This user guide provides basic information about using the Microtronix HyperDrive Multi-port DDR2 Memory Controller IP Core (PN: 6243-xx-xx). The following table shows the document revision history.

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>July 2007</td>
<td>Initial Release – Version 1.0</td>
</tr>
<tr>
<td>August 2007</td>
<td>Version 1.1 – Added Quartus 7.1 patch info</td>
</tr>
<tr>
<td>August 2008</td>
<td>Version 1.2 – Added Stratix III and Arria GX support</td>
</tr>
<tr>
<td>October 2008</td>
<td>Version 1.3 – Added details about odd-length writes</td>
</tr>
<tr>
<td>November 2008</td>
<td>Version 1.4 – Increase ports to 10</td>
</tr>
<tr>
<td>December 2008</td>
<td>Version 1.5 – Add support for ECC DIMMs</td>
</tr>
<tr>
<td>October 2008</td>
<td>Version 1.6 – Added SDC file creation and Timing Closure advice. Added Arria II GX support.</td>
</tr>
<tr>
<td>August 2011</td>
<td>Version 1.7 – Improved flow control of ports. Added Stratix IV support.</td>
</tr>
</tbody>
</table>

E-mail

Sales Information: sales@microtronix.com
Support Information: support@microtronix.com

Website

General Website: http://www.microtronix.com

Phone Numbers

General: (001) 519-690-0091
Fax: (001) 519-690-0092

Typographic Conventions

<table>
<thead>
<tr>
<th>Path/Filename</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[SOPC Builder]$&lt;cmd&gt;</td>
<td>A command that should be run from within the Cygwin Environment.</td>
</tr>
<tr>
<td>Code</td>
<td>Sample code.</td>
</tr>
<tr>
<td>~</td>
<td>Indicates that there is no break between the current line and the next line.</td>
</tr>
</tbody>
</table>
## Table of Contents

Document Revision History ............................................................................................................. 2

How to Contact Microtronix ............................................................................................................. 2

  E-mail ........................................................................................................................................... 2

  Website ........................................................................................................................................ 2

  Phone Numbers ........................................................................................................................... 2

Typographic Conventions ................................................................................................................ 2

Features ........................................................................................................................................... 4

Introduction ...................................................................................................................................... 5

  Local Ports ................................................................................................................................... 7

  Write Port .................................................................................................................................. 7

  Read Port .................................................................................................................................. 9

  Error Correction Coding ......................................................................................................... 11

Design Flow ................................................................................................................................... 11

  About Tab ................................................................................................................................... 12

  Project Tab ................................................................................................................................ 12

  Memory Tab ................................................................................................................................ 13

  Timing Tab ................................................................................................................................... 16

  Local Ports Tab .......................................................................................................................... 17

  SDC Tab .................................................................................................................................... 18

  Generating the Programming File .............................................................................................. 19

  Design of Quartus II Project ....................................................................................................... 20

  DQ/DQS Pin Assignments ........................................................................................................... 20

Timing ............................................................................................................................................ 21

Performance .................................................................................................................................. 22

Resource Requirements ................................................................................................................ 23

Simulation ...................................................................................................................................... 24

Verification ..................................................................................................................................... 24

Installation ..................................................................................................................................... 25

License .......................................................................................................................................... 25

FAQ ............................................................................................................................................... 26
Features

- Supports DDR2 SDRAM memory devices
- DQS Read Capture – read data clocked using delayed DQS
  - PCB layout independent DDR Round Trip Capture
- Extended DDR2 memory burst accesses
- Requires one PLL with 4 output clocks
- Advanced Performance Architecture
  - Up to 400 MHZ DDR2 memory performance
  - Optimized for streaming transfers
  - Up to 10 local bus Read or Write ports
  - User configurable FIFO for each system port
  - Operates with memory clock independent of system clock speed
- Supports standard Hamming Core Error Correction Coding (ECC)
  - Single bit error correction (SEC) algorithm
  - Double bit error detection (DED) algorithm
- Configuration GUI streamlines design process
- Supports: Stratix III, Stratix II / GX, Arria GX and Arria II GX devices
- TimeQuest Timing Analyzer support
- Support for OpenCore Plus evaluation
- Reference Designs
- Simulation Models
Introduction

The Microtronix HyperDrive Multi-port DDR2 Memory Controller IP-Core provides a complete, easy-to-use solution to interface with industry standard DDR2 memory devices. The core can be configured to support 64-bit DDR2 DIMM modules.

The DDR2 Memory Controller handles all memory tasks, including refresh and device initialization cycles. The IP Core is designed to operate asynchronous to the system clock using a unique source-synchronous design architecture to capture the high-speed double-data rate data from each memory device independent of the round-trip delay freeing the memory design task from PCB layout parameters. This proprietary technology simplifies memory interface design by removing the need for extra resynchronization clocks and maximizes the performance of the memory system.

A block diagram of the core is shown in Figure 1 below. The IP core signal connection diagram is given in Figure 2.

![Block Diagram of Multi-port DDR2 IP Core](image-url)

**Figure 1: Block Diagram of Multi-port DDR2 IP Core**
Figure 2: HyperDrive IP Core Signal Connections
Local Ports
The DDR2 Memory Controller has up to ten local ports. Each port can be configured as either a read or write port. The ports are optimized for streaming transfers. An internal FIFO acts as a bridge between the local interface and the SDRAM memory.

Write Port
Figure 3 shows the beginning of a write transfer. The write clock is unrelated to the SDRAM memory clock. On the rising edge of the INIT signal the starting address is latched. Any remaining data in the FIFO is flushed to the SDRAM memory. Data can be written by activating the write enable (WE). The write enable can be deasserted and reasserted as needed. The byte enable (BE) is active high.

Transfers are not limited to the size of the internal FIFO. The write port will transfer data to the SDRAM as needed in blocks of half the FIFO size. When the write port is unable to accept more data into its FIFO, the BUSY signal is asserted. Once more data has been moved from the FIFO into the SDRAM, BUSY is deasserted. Writing data while BUSY is asserted can overrun the FIFO and cause undefined behavior. Note that BUSY will be asserted one cycle ahead of the FIFO being full, allowing WE to be deasserted on the following cycle (Figure 4).
Figure 4: Write Busy

Figure 5 shows how to force a flush of the write FIFO by asserting INIT after completing a write transfer. Once the INIT is signaled, the port will write out all data remaining to SDRAM. It is not necessary to perform this flush cycle as the data will be flushed the next time INIT is asserted, but it may be useful for coordinating between multiple ports. If BUSY is still asserted from the previous writes, the flush should be deferred until BUSY is deasserted.

Figure 5: Write Flush

The write port has minimum delays between events that must be ensured for proper operation. These delays are listed in Table 1.
Table 1: Write Port Timing

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ti-w</td>
<td>INIT to WE delay</td>
<td>1 clock cycle</td>
</tr>
<tr>
<td>Tw-i</td>
<td>WE to INIT delay</td>
<td>2 clock cycles</td>
</tr>
<tr>
<td>Tb-i</td>
<td>BUSY to INIT delay</td>
<td>1 clock cycle</td>
</tr>
</tbody>
</table>

**Write Length**

Only an even number of words can be written between write init cycles. Writing an odd number of words can result in undefined data being written to the word following the last word of the burst. Additionally, the burst must start at an even address.

**Read Port**

Figure 6 shows a read transfer. The read port clock is independent of the SDRAM memory clock domain. Upon a rising edge of the INIT signal the starting address is latched and the read port starts prefetching the data from the SDRAM memory. This is indicated by the BUSY signal. There is a delay between the rising edge of INIT and the rising edge of BUSY, see Table 2 below.

After the BUSY signal is de-asserted by the read port, data can read by activating the read enable (RE). The read port will assert the data enable (DE) to indicate valid read data. DE assertion will be delayed from RE (see Table 2). DE will be asserted for the same number of clock cycles that RE is asserted. The read enable can be deasserted and reasserted as needed.

![Figure 6: Read Initialization](image-url)
Transfers are not limited to the size of the internal FIFO. After the initial prefetch, the read port will fetch data from the SDRAM as needed in blocks of half the FIFO size. The BUSY signal is asserted when data is no longer available in the FIFO. Once more data is read from the SDRAM, BUSY will be deasserted. Reading while BUSY is asserted can underrun the FIFO and produce undefined behavior.

![Figure 7: Read Busy](image)

Once the last word of data is read from the read port, a new transfer can begin immediately by asserting INIT. If BUSY is still asserted from the previous read transfer, the INIT sequence should be deferred until BUSY is deasserted.

![Figure 8: Read Complete](image)

The UNDERRUN signal is activated when the FIFO has been read empty and is still waiting on data from SDRAM. An INIT will clear the UNDERRUN condition.
Table 2: Read Port Timing

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ti-b</td>
<td>INIT to BUSY delay</td>
<td>2 clock cycles</td>
</tr>
<tr>
<td>Tr-d</td>
<td>RE to DE delay</td>
<td>3 clock cycles</td>
</tr>
</tbody>
</table>

Read Address
All reads must start at an even address.

Error Correction Coding
The Error Correction Coding (ECC) core implements the standard Hamming Code based DRAM single error correction and double error detection algorithm. This algorithm generates an 8 bit ECC word for a 64 bit data bus.

The core consists of an ECC generation path as well as ECC detection and correction. ECC correction is possible for single bit errors occurring in one of combined 64 data bits. ECC detection is possible for errors in two of the combined 64 data.

Status information is provided to the user including whether an error was detected on a single bit or two bits.

NOTE: The ECC core does not support misaligned address bursts.

Design Flow
The following steps describe how to integrate the Microtronix HyperDrive Multi-port DDR2 Memory Controller IP-Core into a Quartus project.

- Open a Windows command prompt or Linux terminal window.
- Browse to the SDRAM wizard directory <install_dir>/wizard
- Start the wizard by typing:
  
  java -jar mtx_hyperdrive_sdram_gui.jar or by running the wizard.bat file
**About Tab**

The About tab summarizes the version and build information.

![Figure 9: About Tab](image)

**Project Tab**

The Project tab is used to select an existing design project or define a new project.

- Click on the Project tab.
- Select the target device and speed grade or use the browse button to load an existing project.
The Memory tab is used to select the memory architecture and device specific properties of the target DDR2 memory system. As the HyperDrive IP Core only supports DDR2 devices, the ‘Architecture’ setting cannot be changed.

- Click on the Memory.
- Configure the ‘Data Width’, ‘Bank Address Bits’, ‘Row Address Bits’ and ‘Column Address Bits’ to match the memory configuration.
- Configure the ‘Clock Pairs’ to match the number required by the memory architecture. For example, DIMM modules typically require three clock pairs.
- The ‘Reduced Drive Strength’ option is used to reduce the memory output drive strength. This option is used in light load memory configurations.
- The ‘On-Die Termination Support’ option is used to support DDR2

**Figure 10: Project Tab**
memory devices incorporating on-die termination. The available termination options are 50, 75 or 150 ohms.

![Microtronix Hyperdrive Multi Port Memory Controller](image)

**Figure 11: Memory Tab**

- ECC DIMM Support option enables error code generation and checking. This setting can only be enabled when using a 64-bit wide dual-inline memory module (DIMM) (Data Width 64, Total Devices 8, DQS signals per Device 1). It requires a DIMM capable of supporting error correction codes (ECC).

Enabling ECC support adds three cycles of latency between RE and DE in read ports.

- This feature adds an additional output to every read port. The 4-bit ERROR signal indicates, for the current word, if an error was detected by the error correction module and if the error was corrected. The low half of ERROR indicates errors in the low half of data. The high half of ERROR indicates errors in the high half of data. The interpretation of the 2-bit error codes is shown in Table 3.
Table 3: ECC Error Status Bits

<table>
<thead>
<tr>
<th>ERROR(3) or ERROR(1)</th>
<th>ERROR(2) or ERROR(0)</th>
<th>Error Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>No Error</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Single-bit Error, Corrected</td>
</tr>
<tr>
<td>1</td>
<td>X</td>
<td>Uncorrectable Error</td>
</tr>
</tbody>
</table>

- The SDRAM GUI uses a Pin Prefix (e.g. sdram_) for the names of all memory interface pins and it expects that these pins have a fixed name as shown in Table 4.

Table 4: DDR2 SDRAM Pin Names

<table>
<thead>
<tr>
<th>SDRAM Pin Function</th>
<th>SDRAM Pin Name (without prefix)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock Enable</td>
<td>cke</td>
</tr>
<tr>
<td>Bank Address</td>
<td>ba</td>
</tr>
<tr>
<td>Address</td>
<td>a</td>
</tr>
<tr>
<td>Chip Select ( (Note 1) )</td>
<td>cs</td>
</tr>
<tr>
<td>Row Address Strobe ( (Note 1) )</td>
<td>ras</td>
</tr>
<tr>
<td>Column Address Strobe ( (Note 1) )</td>
<td>cas</td>
</tr>
<tr>
<td>Write Enable ( (Note 1) )</td>
<td>we</td>
</tr>
<tr>
<td>Data</td>
<td>dq</td>
</tr>
<tr>
<td>Data Mask</td>
<td>dm</td>
</tr>
<tr>
<td>Data Strobe</td>
<td>dqs</td>
</tr>
<tr>
<td>Differential Data Strobe ( when enabled )</td>
<td>dqs_p</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>DDR Clock Out Positive</td>
<td>clk_out_p</td>
</tr>
<tr>
<td>DDR Clock Out Negative</td>
<td>clk_out_n</td>
</tr>
<tr>
<td>On Die Termination</td>
<td>odt</td>
</tr>
</tbody>
</table>

Note 1: These signals are active low.
Timing Tab

The Timing tab is used to enter the DDR2 memory specific timing parameters found in the datasheet supplied by the vendor of the memory device.

- Click the Timing tab.
- Enter the SDRAM memory timing parameters from the product datasheet.

![Figure 12: Timing Tab](image-url)
**Local Ports Tab**

The Local Ports tab is used to configure each of the local ports. In total up to ten ports can be selected. The buffer size selects the number of words that can be temporarily held in the FIFO. A bigger buffer size results in better performance, but will also consume more memory resources and the duration of the busy state will be longer after the start of each memory cycle. (See port-timing diagrams.)

- Click on the Local Ports.
- Define the mode of each port for either Write or Read operation.
- Configure the depth of the (FIFO) buffer.

![Local Ports Tab](image)

*Figure 13: Local Ports Tab*
**SDC Tab**

The SDC tab is used to enter the memory device specific timing parameters found in the datasheet supplied by the vendor of the memory device. This tab is an extension to the timing tab. The information entered under this tab is used in the Synopsys Design Constraint (SDC) script required by TimeQuest. The SDC script defines the timing constraints and specifications to validate the timing performance of the memory controller logic in the FPGA. The script is written in TCL command language and will be automatically generated by the GUI.

- If desired, check the Enable SDC File Creation option (not supported for SDR memory).
- Enter the values for the SDC-specific parameters. Most values will come from the memory device data sheet. Clock cycle time is the period of the memory clock. PLL input clock period is the period of the clock used as input to the PLL that generates the memory clock.

![SDC File tab](image)

*Figure 14: SDC File tab*
Generating the Programming File

The process for generating a sof and pof programming file is as follows:

**NOTE:** A full license if required to generate a pof programming file. If you are using an OpenCores Plus license you will only be able to generate a time-limited sof file.

- Click on Generate to start the SDRAM generation.
- The wizard writes a top level SDRAM entity.
- Also the wizard generates a TCL script (<project>_assignments.tcl), which sets the Quartus SDRAM assignments.
- Start Quartus and open the project.
- Add the SDRAM component to the project and connect the signals.
- Add the mtx_hyperdrive_sdram_package.vhd file to the project (Assignments -> Settings -> Files). The mtx_hyperdrive_sdram_package.vhd file is located in the directory <install_dir>/synthesis.
- Add the <install_dir>/synthesis directory to the Quartus user libraries (Assignments -> Settings -> Libraries)
- Run the TCL script <project>_assignments.tcl (Tools -> TCL Scripts…)
- Add the <project>.sdc timing script to the Quartus project (Assignments->Timing Analysis Settings->TimeQuest Timing Analyzer)
- Start the Quartus compilation.
Design of Quartus II Project

In a double-data rate system, the SDRAM clocks are generated by a PLL. Use the Altera MegaWizard to generate the SDRAM PLL. The SDRAM controller requires several clocks as listed in Table 5.

Table 5: SDRAM Clocks

<table>
<thead>
<tr>
<th>Clock</th>
<th>Frequency</th>
<th>Phase Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDRAM_HS_CLK</td>
<td>SDRAM Frequency</td>
<td>0 degrees</td>
</tr>
<tr>
<td>SDRAM_LS_CLK</td>
<td>½ SDRAM Frequency</td>
<td>0 degrees</td>
</tr>
<tr>
<td>SDRAM_LS_45_CLK</td>
<td>½ SDRAM Frequency</td>
<td>45 degrees</td>
</tr>
<tr>
<td>SDRAM_WRITE_CLK</td>
<td>SDRAM Frequency</td>
<td>-90 degrees</td>
</tr>
</tbody>
</table>

Add the PLL to the top level and start compilation. (Warning: Don’t forget to assign the SDRAM pin locations on the FPGA before starting the compilation.

Figure 15: DDR2 Top Level

DQ/DQS Pin Assignments

The Microtronix HyperDrive Memory Controller requires the use of dedicated DQ / DQS pins. All Quartus DQ-group rules must be followed.
Timing

The HyperDrive configuration GUI generates an SDC (Synopsis Design Constraints) file for use with the TimeQuest Timing Analyzer. Quartus II must be directed to use this file by selecting “Timing Analyzer Settings” from the “Assignments” menu, selecting the “TimeQuest Timing Analyzer” category and then adding the generated SDC file.

The generated SDC file may need to be customized to match the target design. At the top of the file are a number of constants that should be modified to match the routing of the target board. The board_skew_ck_dqs, board_skew_ck_cmd and board_skew_dqs_dq constants describe the skew between the clocks, command/address signals, dqs and dq signals. Under the “Create Clocks” section, the target of the get_ports directive of the create_clock command must be changed to match the name of the pin feeding the SDRAM PLL. The constants that follow create clocks must be modified so that they match the clocking scheme of the target design.

After the compilation process is complete, TimeQuest should be open to check for timing errors.

If errors are reported for SDRAM_A, SDRAM_BA or SDRAM_CS as shown in Figure 16 these pins should be removed from IO Pads. This is done in the Assignment Editor (Figure 17).

![Figure 16: Timing Error Report](image)
Other possible reason to timing errors might be a large fan out for some of the signals which could be helped by using Maximum Fan-Out option in Assignment Editor.

Table 6 shows the typical performance results for the Microtronix HyperDrive Multi-port DDR2 Memory Controller. Actual performance may be affected by the system LE count and the FPGA pin assignments.

### Table 6: Typical DDR2 SDRAM Performance

<table>
<thead>
<tr>
<th>Device</th>
<th>Speed Grade (Commercial)</th>
<th>System Fmax (MHz) DDR2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arria GX</td>
<td>-6</td>
<td>300</td>
</tr>
<tr>
<td>Arria II GX</td>
<td>-4, -5, -6</td>
<td>300, 300, 267</td>
</tr>
<tr>
<td>Stratix IV</td>
<td>-2</td>
<td>400</td>
</tr>
<tr>
<td></td>
<td>-3</td>
<td>333</td>
</tr>
<tr>
<td></td>
<td>-4</td>
<td>300</td>
</tr>
<tr>
<td>Stratix III</td>
<td>-2</td>
<td>400</td>
</tr>
<tr>
<td></td>
<td>-3</td>
<td>333</td>
</tr>
<tr>
<td></td>
<td>-4</td>
<td>300</td>
</tr>
<tr>
<td>Stratix II / GX</td>
<td>-3</td>
<td>333</td>
</tr>
<tr>
<td></td>
<td>-4</td>
<td>300</td>
</tr>
<tr>
<td></td>
<td>-5</td>
<td>267</td>
</tr>
</tbody>
</table>
Table 7 shows the typical size in logic elements (LE) for the various modules of the DDR2 SDRAM IP Core. The actual number of logic elements may vary depending on the device family and Quartus settings. The table shows the minimal M4K RAM blocks usage. If a large buffer size is selected, the number of M4K RAM blocks may increase.

Table 7: FPGA Resource Requirements

<table>
<thead>
<tr>
<th>Module</th>
<th>LE</th>
<th>M4K RAM Blocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR2 SDRAM Controller</td>
<td>600</td>
<td>-</td>
</tr>
<tr>
<td>Write Port</td>
<td>200</td>
<td>1</td>
</tr>
<tr>
<td>Read Port</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>ECC Encoder (one per write port, if enabled)</td>
<td>350</td>
<td>0</td>
</tr>
<tr>
<td>ECC Decoder (one per read port, if enabled)</td>
<td>450</td>
<td>0</td>
</tr>
</tbody>
</table>

*Note: The number of logic element resources depends on the memory architecture, data width and buffer settings. The numbers shown in the table are for a 16-bit DDR2 SDRAM implementation.
**Simulation**

A precompiled simulation library is provided for performing simulations using ModelSim. The library is located in the `<install_dir>/simulation` directory. Perform the following steps to simulate your design with the SDRAM memory controller.

1. Launch ModelSim.
2. Map the SDRAM memory controller library. At the ModelSim prompt type:
   ```
   vmap mtx_hyperdrive_sdram
   <install_dir>/simulation/mtx_hyperdrive_sdram
   ```
   WARNING: If you use a newer version of ModelSim, you must refresh the precompiled library. At the Modelsim prompt type:
   ```
   vcom -refresh -work mtx_hyperdrive_sdram
   ```
3. Compile the sdragram top level. Eg. If the SDRAM controller was named `sdram` in Quartus II, then type:
   ```
   vcom -93 sdram.vhd
   ```
4. Compile all of the design files.
5. Start the ModelSim simulation by typing:
   ```
   vsim -t ps -L mtx_hyperdrive_sdram <top_level>
   ```

**Verification**

The HyperDrive Multi-port DDR2 Memory Controller IP core has been verified on Altera and other FPGA development boards. Table 8 shows the hardware platforms on which the IP core has been tested and the memory devices contained on each board.

**Table 8: Supported Platforms and Tested DDR2 Memory Devices**

<table>
<thead>
<tr>
<th>Development Board</th>
<th>Altera Device</th>
<th>SDRAM Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>Altera Arria GX Development Kit</td>
<td>EP1AGX60DF780</td>
<td>Micron MT47H32M16</td>
</tr>
<tr>
<td>Terasic DE4</td>
<td>EP4SGX230KF40C2</td>
<td>DSL DDR2 800 1GB CL6 Module</td>
</tr>
<tr>
<td>Altera Stratix III FPGA Development Kit</td>
<td>EP3SL150F1152C2</td>
<td>Micron MT47H32M8BP-3</td>
</tr>
<tr>
<td>Microtronix Stratix II Test Board</td>
<td>EP2S15F484C3</td>
<td>Micron MT47H32M16BN-25E</td>
</tr>
<tr>
<td>Altera Audio Video Development Kit, Stratix II GX Edition</td>
<td>EP2GX90FF1508C3</td>
<td>Micron MT9HTF6472AY-53EB3</td>
</tr>
</tbody>
</table>
Installation

Follow these steps to install the Microtronix HyperDrive Multi-port DDR2 Memory Controller IP core on your computer.

1. Unzip the Microtronix IP core file into a temporary directory on your computer.

2. Run the setup program to install the package onto your computer. If it doesn’t, browse to the CD using Windows Explorer and double-click on the setup icon.

3. Follow all the prompts. The setup program will attempt to auto-detect the installation location of the Quartus II. Please correct the specified paths if the setup program doesn’t or incorrectly detects them.

License

A valid IP core license is required from Microtronix to generate program files incorporating the DDR2 SDRAM IP-core. These licenses are generated based on a NIC or Guard ID supplied by the user. They can be either server or workstation based.

After purchasing a license you receive your license file. Copy the license file (license.dat) to your current Quartus license file and the SDRAM controller (CC21_6243) will show in the Quartus License Setup.

With the free OpenCorePlus feature the SDRAM controller can be evaluated in real hardware. If during compilation no valid license is detected, the OpenCorePlus feature is enabled.

Please contact Microtronix for licensing details.
FAQ

Q. Do you have any recommendations for layout of my SDRAM memory board?

A. Microtronix does not have specific layout recommendations. Please follow the layout guidelines of your SDRAM manufacturer. For example, Micron’s recommendations for DDR/DDR2 design can be found at:


Q. My DDR2 SDRAM memory board layout includes external parallel termination resistors. Is this supported?

A. During idle the parallel termination resistor pulls the DQ/DQS signal to the VTT voltage level, which is the same level as the VREF voltage. The SSTL input buffer is a comparator and this causes unwanted glitches inside the programmable device.

For DDR2 memory architectures using On-Die Termination (ODT) is preferable and this feature gives better signal integrity results than external parallel termination.

To support external parallel termination one of the following steps must be applied to the board.

1. In designs using a point-to-point memory architecture (short DQ/DQS traces) and single parallel termination resistors, remove the resistor connected to DQS0.

2. In designs incorporating a more complex memory architecture, (longer traces, DIMM sockets) or resistors packages, adjust the VREF voltage by using the voltage divider.

![Diagram of DDR2 Power Supply](image-url)