Block RAM (BRAM) Block (v1.00a)

DS444 March 12, 2007 **Product Specification**

Introduction

The BRAM Block is a configurable memory module that attaches to a variety of BRAM Interface Controllers.

The BRAM Block structural HDL is generated by the EDK design tools based on the configuration of the BRAM interface controller IP. All BRAM Block parameters are automatically calculated and assigned by the EDK tools Platgen and Simgen.

Features

- Fully automated generation and configuration of HDL through EDK Platgen/Simgen tools.
- Number of BRAM primitives utilized is a function of the configuration parameters for: memory address range, number of byte-write enables, the data width, and the targeted architecture
- Both Port A and Port B of the memory block can be connected to independent BRAM Interface Controllers: LMB (Local Memory Bus), OPB (On-chip Peripheral Bus), PLB (Processor Local Bus), and OCM (On-Chip Memory).
- Supports byte, half-word, word, and doubleword transfers provided the correct number of byte-write enables have been configured

LogiCORE™ Facts		
Core Specifics		
Supported Device Family	QPro™-R Virtex-II™, QPro Virtex-II, Spartan™-II, Spartan-IIE, Spartan-3, Spartan-3E, Virtex, Virtex-II, Virtex-II Pro, Virtex-4, Virtex-E, Virtex-5	
Version of Core	bram_block	v1.00a
Resources Used		
	Min	Max
I/O	N/A	N/A
LUTs	0	0
FFs	0	0
Block RAMs	1	64
Provided with Core		
Documentation	Product Specification	
Design File Formats	VHDL	
Constraints File	N/A	
Verification	N/A	
Instantiation Template	N/A	
Reference Designs	None	
Design Tool Requirements		
Xilinx Implementation Tools	5.1i or later	
Verification	N/A	
Simulation	ModelSim SE/PE 5.7b or later	
Synthesis	XST	
Support		
Support provided by Xilinx, Inc.		

Functional Description

The BRAM Block is a structural design that instantiates a number of RAMB primitives, depending on specific factors. An example is shown in Figure 1.

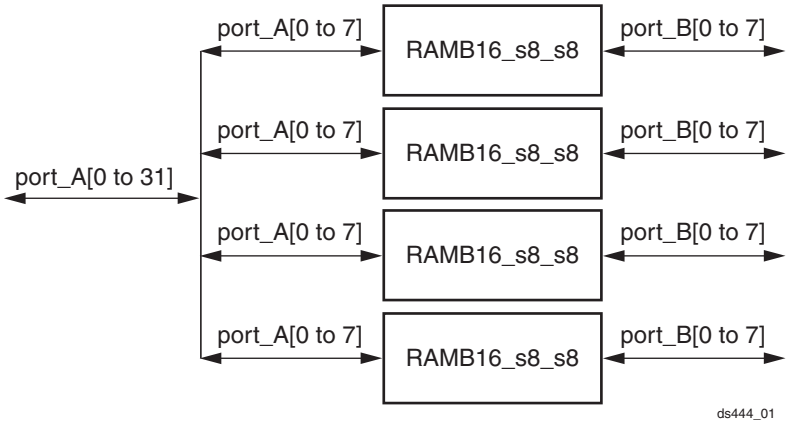

Figure 1: **Example of BRAM Block implementation with 4 RAMB16 primitives**

BRAM Block I/O Signals

The I/O signals for the BRAM Block are shown in Figure 1 and described in Table 1. Note that the data in/out signals are named the opposite of their actual direction. This is because the signals are referenced from the point of view of the BRAM controller so that "data in" on the controller connects to "data in" on the BRAM block, and "data out" on the controller connects to "data out" on the BRAM block.

Table 1: **BRAM Block I/O Signals**

Signal Name	Interface	I/O	Initial State	Description
BRAM_Rst_A	Port A	I		BRAM Reset, Active High
BRAM_Clk_A	Port A	I		BRAM Clock
BRAM_EN_A	Port A	I		BRAM Enable, Active High
BRAM_WEN_A	Port A	I		BRAM Write Enable, Active High
BRAM_Addr_A (0:C_PORT_AWIDTH-1)	Port A	I		BRAM Address
BRAM_Din_A (0:C_PORT_DWIDTH-1)	Port A	O	0	BRAM Data **Output**. Note signal name is referenced from the point of view of the controller (on which it is an input).
BRAM_Dout_A (0:C_PORT_DWIDTH-1)	Port A	I		BRAM Data **Input**. Note signal name is referenced from the point of view of the controller (on which it is an output).
BRAM_Rst_B	Port B	I		BRAM Reset, Active High
BRAM_Clk_B	Port B	I		BRAM Clock
BRAM_EN_B	Port B	I		BRAM Enable, Active High
BRAM_WEN_B	Port B	I		BRAM Write Enable, Active High

Table 1: **BRAM Block I/O Signals *(Contd)***

Signal Name	Interface	I/O	Initial State	Description
BRAM_Addr_B (0:C_PORT_AWIDTH-1)	Port B	I		BRAM Address
BRAM_Din_B (0:C_PORT_DWIDTH-1)	Port B	O	0	BRAM Data **Output**. Note signal name is referenced from the point of view of the controller (on which it is an input).
BRAM_Dout_B (0:C_PORT_DWIDTH-1)	Port B	I		BRAM Data **Input**. Note signal name is referenced from the point of view of the controller (on which it is an output).

BRAM Block Parameters

Table 2: **BRAM block Parameters**

Parameter Name	Feature/Description	Allowable Values	Tool Calculated	VHDL Type
C_PORT_ AWIDTH	Port A and B Address Width	9 – 17	Yes	integer
C_PORT_ DWIDTH	Port A and B Data Width	32, 64	Yes	integer
C_NUM_WE	Number of Write Enables (Byte Enables for Write)	1, 2, 4, 8	Yes	integer
C_FAMILY	Target FPGA family of bram_block	spartan2, spartan2e, spartan3, virtex, virtexe, virtex2, qvirtex2, qrvirtex2, virtex2p, virtex4	Yes	string
C_MEMSIZE	Size of BRAM(s) in bytes	512 - 131072	Yes	integer

Allowable Parameter Combinations

For architectures with 4kb RAMB primitives (C_FAMILY equals one of spartan2, spartan2e, virtex, or virtexe), the maximum value for C_MEMSIZE is 32kB.

For architectures with 16kb RAMB primitives (C_FAMILY equals one of spartan3, virtex2, qvirtex2, qrvirtex2, virtex2p, or virtex4), the minimum value for C_MEMSIZE is 2kB.

Parameter - Port Dependencies

The width of many of the BRAM Block signals depends on the number of memories in the system and the width of the various data and address buses. The dependencies between the BRAM design parameters and I/O signals are shown in Table 3.

Table 3: **Parameter-Port Dependencies**

Name	Affects	Depends	Relationship Description
Design Parameters			
C_PORT_ DWIDTH	BRAM_Din_A/B BRAM_Dout_A/B	0 to C_PORT_DWIDTH -1 0 to C_PORT_DWIDTH -1	Width of BRAM interface Width of BRAM interface
C_PORT_ AWDITH	BRAM_Addr_A/B	0 to C_PORT_AWIDTH -1	Width of the BRAM Address Bus
C_NUM_WE	BRAM_WEN_A/B	0 to C_NUM_WE -1	Number of byte enable signals
I/O Signals			
BRAM_Addr_A		C_PORT_AWIDTH	Width varies with the width of the BRAM Address Bus
BRAM_Din_A		C_PORT_DWIDTH	Width varies with the width of the BRAM Data Bus
BRAM_Dout_A		C_PORT_DWIDTH	Width varies with the width of the BRAM Data Bus
BRAM_WEN_A		C_NUM_WE	Width varies with the number of byte-write enable signals
BRAM_Addr_B		C_PORT_AWIDTH	Width varies with the width of the BRAM Address Bus.
BRAM_Din_B		C_PORT_DWIDTH	Width varies with the width of the BRAM Data Bus.
BRAM_Dout_B		C_PORT_DWIDTH	Width varies with the width of the BRAM Data Bus.
BRAM_WEN_B		C_NUM_WE	Width varies with the number of byte-write enable signals

BRAM Block Register Descriptions

Not applicable.

BRAM Block Interrupt Descriptions

Not applicable.

The BRAM Block is a structural design that instantiates a number of RAMB primitives. The number of block RAM primitives instantiated in a BRAM Block depends on the following factors

- Each primitive is either 4kb or 16kb depending on architecture. The address range (C_PORT_AWIDTH) times the accessed data width (C_PORT_DWIDTH) defines the total number of bits required.
- Instantiated RAMB primitives can be configured to have at most a 32 bit wide data interface. Thus a 64 bit interface will require at least 2 BRAM primitives in parallel.
- RAMB primitive in architectures prior to Virtex-4 only have a single write enable per port. Thus if byte-write enable is required on a 32 bit data port (C_NUM_WE=4), these architectures will use a minimum of 4 BRAM primitives.

Design Implementation

Design Tools

The BRAM Block design is generated by the EDK tools.

XST is the synthesis tool used for synthesizing the BRAM Block. The EDIF netlist output from XST is then input to the Xilinx Alliance tool suite for actual device implementation.

Target Technology

The intended target technology is an FPGA in one of the following families: Virtex, Virtex-E, Spartan-II, Spartan-IIE, Spartan-3, Virtex-II, QPro Virtex-II, QPro-R Virtex-II, Virtex-II Pro, or Virtex-4.

Device Utilization and Performance Benchmarks

The device utilization depends on the configured BRAM Block size in relation to the RAMB primitive resources of the targeted device. See the user guide for the respective FPGA family for details on RAMB primitive performance and available resources.

Specification Exceptions

Not applicable.

Reference Documents

None

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
01/07/03	1.3	Updated for EDK SP3.
07/22/03	1.4	Updated to new template.
08/03/04	1.5	Updated with new families and added detail on BRAM primitive usage. Reformatted
8/13/04	1.5.1	Updated for Gmm: reviewed and corrected trademarks.
4/2/05	1.6	Updated for EDK 7.1.1 SP1; updated supported device family listing.
8/1/05	1.7	Converted to new DS template; no content edited.
12/1/05	1.8	Added Spartan-3E to supported device listing.
8/21/06	1.9	C_MEMSIZE description corrected
3/12/07	1.10	Added Virtex-5 support.