XAPP130 (v1.4) December 18, 2000

Using the Virtex Block SelectRAM+ Features

Summary

The Virtex™ series provides dedicated blocks of on-chip, true dual-read/write port synchronous RAM, with 4096 memory cells. Each port of the block SelectRAM+™ memory can be independently configured as a read/write port, a read port, or a write port, and each port can be configured to a specific data width. The block SelectRAM+ memory offers new capabilities allowing the FPGA designer to simplify designs.

Introduction

The Virtex series of devices now includes the block SelectRAM+ feature, while maintaining the distributed RAM available in the XC4000X families. The distributed RAM allows the creation of shallow RAM structures throughout the device, with the RAM co-located in each CLB. The block SelectRAM+ memory offers the additional capability of fast, discrete, and large blocks of RAM in the device. Each block SelectRAM+ cell, located in the columns of the device, spans a height of four CLBs.

The block SelectRAM+ memory is a fully synchronous, true dual-read/write port RAM with 4096 memory cells. Each port contains independent control signals, allowing designers to create flexible RAM structures. The data width of each port is independently configurable, providing built-in bus width conversion.

Table 1 describes the depth and width aspect ratios for the block SelectRAM+ memory.

Table 1: **Block SelectRAM+ Port Aspect Ratios**

Width	Depth	ADDR Bus	Data Bus
1	4096	ADDR<11:0>	DATA<0>
2	2048	ADDR<10:0>	DATA<1:0>
4	1024	ADDR<9:0>	DATA<3:0>
8	512	ADDR<8:0>	DATA<7:0>
16	256	ADDR<7:0>	DATA<15:0>

Synchronous Memory Basics

Dual-port and single-port synchronous RAM as discrete components have several different modes of operation for read and write operations. The four major modes of operation are:

Read Through (One Clock Edge)

The read address is registered on the read port clock edge and data appears on the output after the RAM access time. Some memories may place the latch/register at the outputs depending on the desire to have a faster clock-to-out versus setup time. This is generally considered to be an inferior solution, since it changes the read operation to an asynchronous function. This raises the possibility of missing an address/control line transition during the generation of the read pulse clock.

Read Pipelined (Two Clock Edges)

The read address is registered on the read port clock edge; the data is registered and appears on the output after the second read clock edge.

Write Back (One Clock Edge)

The write address is registered on the write port clock edge; the data input is written to the memory and mirrored on the write port input.

Write Through (One Clock Edge)

The write address is registered on the write port clock edge; the data is mirrored on the write port input as well as on the read port output if the write and read addresses match. Data is written to and read from the memory in the same cycle.

The block SelectRAM+ memory has the “read through” and “write back” functions. Simply by adding a CLB register to the outputs, designs requiring a “read pipeline” function can be done. This has the effect of improving the perceived clock-to-out timing of the block SelectRAM+ memory with little device area cost.

Block SelectRAM+ Characteristics

1. All inputs are registered with the port clock and have a setup to clock timing specification.
2. All outputs have a read through or write back function depending on the state of the port WE pin. The outputs relative to the port clock are available after the clock-to-out timing specification.
3. The block RAMs are true SRAM memories and do not have a combinatorial path from the address to the output. The LUT SelectRAM+ cells in the CLBs are still available with this function.
4. The ports are completely independent of each other (i.e., clocking, control, address, read/write function, and data width) without arbitration.
5. A write operation requires only one clock edge.
6. A read operation requires only one clock edge.
7. The output ports are latched with a self-timed circuit to guarantee a glitch-free read. The state of the output port will not change until the port executes another read or write operation.

Library Primitives

Figure 1 and Figure 2 show the two generic library block SelectRAM+ primitives. Table 2 describes all of the available primitives for synthesis and simulation.

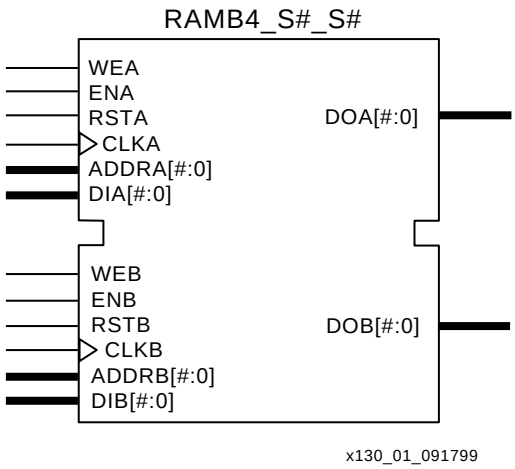

Figure 1: **Dual-Port Block SelectRAM+ Memory**

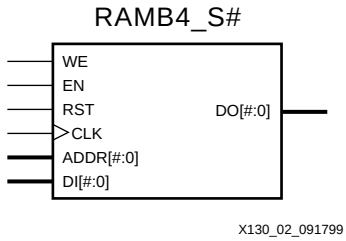

Figure 2: **Single-Port Block SelectRAM+ Memory**

Table 2: **Available Library Primitives**

Primitive	Port A Width	Port B Width
RAMB4_S1	1	N/A
RAMB4_S1_S1		1
RAMB4_S1_S2		2
RAMB4_S1_S4		4
RAMB4_S1_S8		8
RAMB4_S1_S16		16
RAMB4_S2	2	N/A
RAMB4_S2_S2		2
RAMB4_S2_S4		4
RAMB4_S2_S8		8
RAMB4_S2_S16		16
RAMB4_S4	4	N/A
RAMB4_S4_S4		4
RAMB4_S4_S8		8
RAMB4_S4_S16		16
RAMB4_S8	8	N/A
RAMB4_S8_S8		8
RAMB4_S8_S16		16
RAMB4_S16	16	N/A
RAMB4_S16_S16		16

Port Signals

Each block SelectRAM+ port operates independently of the other while accessing the same set of 4096 memory cells.

Clock — CLK[A|B]

Each port is fully synchronous with independent clock pins. All port input pins have setup time referenced to the port CLK pin. The data output bus has a clock-to-out time referenced to the CLK pin.

Enable — EN[A|B]

The enable pin affects the read, write and reset functionality of the port. Ports with an inactive enable pin keep the output pins in the previous state and do not write data to the memory cells.

Write Enable — WE[A|B]

Activating the write enable pin allows the port to write to the memory cells. When active, the contents of the data input bus are written to the RAM at the address pointed to by the address bus, and the new data is also reflected on the data out bus. When inactive, a read operation occurs and the contents of the memory cells referenced by the address bus are reflected on the data out bus.

Reset — RST[A|B]

The reset pin forces the data output bus latches to zero synchronously. This does not affect the memory cells of the RAM and does not disturb a write operation on the other port.

Address Bus — ADDR[A|B]<#:0>

The address bus selects the memory cells for read or write. The width of the port determines the required width of this bus, as shown in Table 1.

Data In Bus — DI[A|B]<#:0>

The data in bus provides the new data value to be written into the RAM. This bus and the port have the same width, as shown in Table 1.

Data Output Bus — DO[A|B]<#:0>

The data out bus reflects the contents of the memory cells referenced by the address bus at the last active clock edge. During a write operation, the data out bus reflects the data in bus. The width of this bus equals the width of the port. The allowed widths appear in Table 1.

Inverting Control Pins

The four control pins (CLK, EN, WE and RST) for each port have independent inversion control as a configuration option.

Address Mapping

Each port accesses the same set of 4096 memory cells using an addressing scheme dependent on the width of the port. The physical RAM locations addressed for a particular width are described in the following formulas (of interest only when the two ports use different aspect ratios):

$$Start = ([ADDR_{port} + 1] \times Width_{port}) - 1$$

$$End = ADDR_{port} \times Width_{port}$$

Table 3 shows low-order address mapping for each port width.

Table 3: **Port Address Mapping**

Port Width	Port Addresses															
1	4095...	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
2	2047...	07		06		05		04		03		02		01		00	
4	1023...	03				02				01				00			
8	511...	01								00							
16	255...	00															

Creating Larger RAM Structures

The block SelectRAM+ columns have specialized routing to allow cascading blocks together with minimal routing delays. This achieves wider or deeper RAM structures with a smaller timing penalty than when using normal routing channels.

Location Constraints

Block SelectRAM+ instances can have LOC properties attached to them to constrain the placement. The block SelectRAM+ placement locations are separate from the CLB location naming convention, allowing the LOC properties to transfer easily from array to array.

The LOC properties use the following form:

LOC = RAMB4_R#C#

RAMB4_R0C0 is the upper left RAMB4 location on the device.

Conflict Resolution

The block SelectRAM+ memory is a true dual-read/write port RAM that allows simultaneous access of the same memory cell from both ports. When one port writes to a given memory cell, the other port must not address that memory cell (for a write or a read) within the clock-to-clock setup window. Specifics of port and memory cell write conflict resolution follow:

- If both ports write to the same memory cell simultaneously, thus violating the clock-to-clock setup requirement, the data stored is invalid
- If one port attempts a read of a memory cell while the other port simultaneously writes to it, thus violating the clock-to-clock setup requirement, the following occurs:
  - The write succeeds
  - The data out on the writing port accurately reflects the data written
  - The data out on the reading port is invalid
- Conflicts do not cause any physical damage

Read and Write Operations

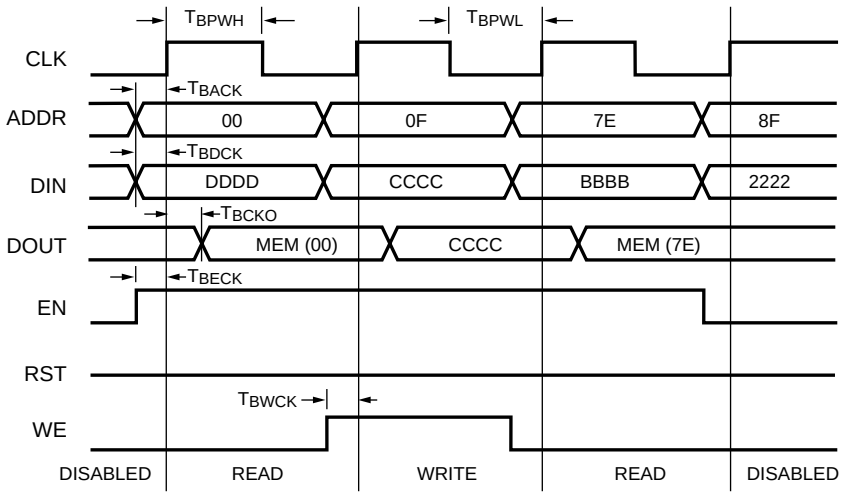

Figure 3: **Timing Diagram for Single-Port Block SelectRAM+ Memory**

Single-Port Timing

Figure 3 shows a timing diagram for a single port of a block SelectRAM+ memory. The block RAM AC switching characteristics are specified in the datasheet. The block RAM memory is initially disabled.

Refer to Figure 3 in conjunction with the following steps:

1. At the first rising edge of the CLK pin, the ADDR, DI, EN, WE, and RST pins are sampled. The EN pin is High and the WE pin is Low, indicating a read operation. The DO bus contains the contents of the memory location, 0x00, as indicated by the ADDR bus.
2. At the second rising edge of the CLK pin, the ADDR, DI, EN, WR, and RST pins are sampled again. The EN and WE pins are High, indicating a write operation. The DO bus mirrors the DI bus. The DI bus is written to the memory location 0x0F.
3. At the third rising edge of the CLK pin, the ADDR, DI, EN, WR, and RST pins are sampled again. The EN pin is High and the WE pin is Low, indicating a read operation. The DO bus contains the contents of the memory location 0x0F as indicated by the ADDR bus.
4. At the fourth rising edge of the CLK pin, the ADDR, DI, EN, WR, and RST pins are sampled again. The EN pin is Low, indicating that the block RAM is now disabled. The DO bus retains the last value.

Dual-Port Timing

Figure 4 shows a timing diagram for a true dual-port read/write block RAM. The clock on Port A has a longer period than the clock on Port B. The timing parameter T_{BCCS} (clock-to-clock setup) is shown on this diagram. The parameter T_{BCCS} is violated once in the diagram. All other timing parameters are identical to the single-port version shown in Figure 3.

T_{BCCS} is only of importance when the address of both ports are the same and at least one port is performing a write operation. When the clock-to-clock setup parameter is violated for a WRITE-WRITE condition, the contents of the memory at that location will be invalid. When the clock-to-clock setup parameter is violated for a WRITE-READ condition, the contents of the memory will be correct, but the read port will have invalid data.

Refer to Figure 4, page 7, in conjunction with the following steps:

1. At the first rising edge of CLKA, memory location 0x00 is to be written with the value 0xAAAA and is mirrored on the DOA bus. The last operation of Port B was a read to the same memory location 0x00. The DOB bus of Port B does not change with the new value on Port A, and retains the last read value. A short time later, Port B executes another read to memory location 0x00, and the DOB bus now reflects the new memory value written by Port A.
2. At the second rising edge of CLKA, memory location 0x7E is written with the value 0x9999 and is mirrored on the DOA bus. Port B then executes a read operation to the same memory location without violating the T_{BCCS} parameter, and the DOB reflects the new memory values written by Port A.
3. At the third rising edge of CLKA, the T_{BCCS} parameter is violated with two writes to memory location 0x0F. The DOA and DOB busses reflect the contents of the DIA and DIB busses, but the stored value at 0x0F is invalid.
4. At the fourth rising edge of CLKA, a read operation is performed at memory location 0x0F, and invalid data is present on the DOA bus. Port B also executes a read operation to memory location 0x0F, and also reads invalid data.
5. At the fifth rising edge of CLKA a read operation is performed that does not violate the T_{BCCS} parameter to the previous write of 0x7E by Port B. The DOA bus reflects the recently written value by Port B.

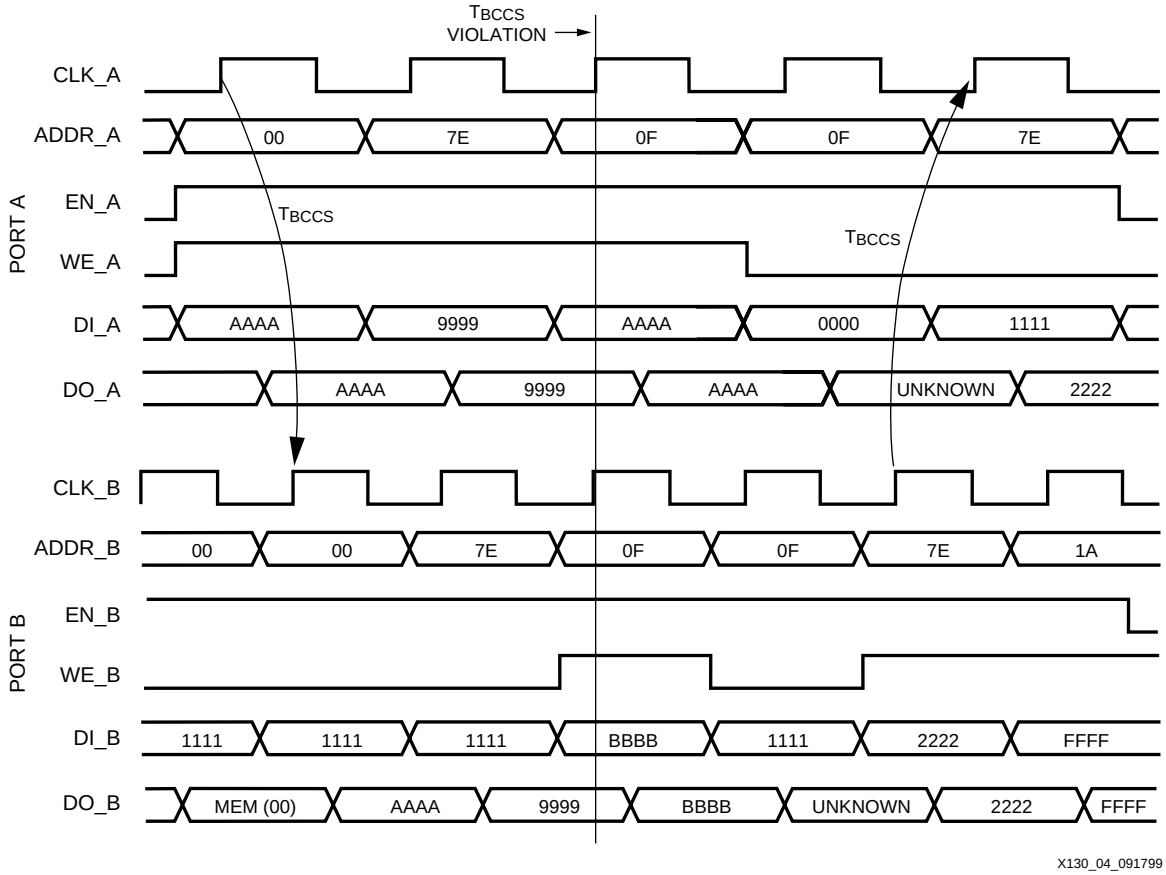

Figure 4: **Timing Diagram for a True Dual-Port Read/Write Block SelectRAM+ Memory**

Design Examples

Creating a 32-bit Single-Port RAM

The true dual-read/write port functionality of the block RAM allows a single port, 128 deep by 32-bit wide RAM to be created using a single block RAM cell as shown in Figure 5.

Interleaving the memory space, setting the LSB of the address bus of Port A to 1 (V_{CC}), and the LSB of the address bus of Port B to 0 (GND), allows a 32-bit wide single-port RAM to be created.

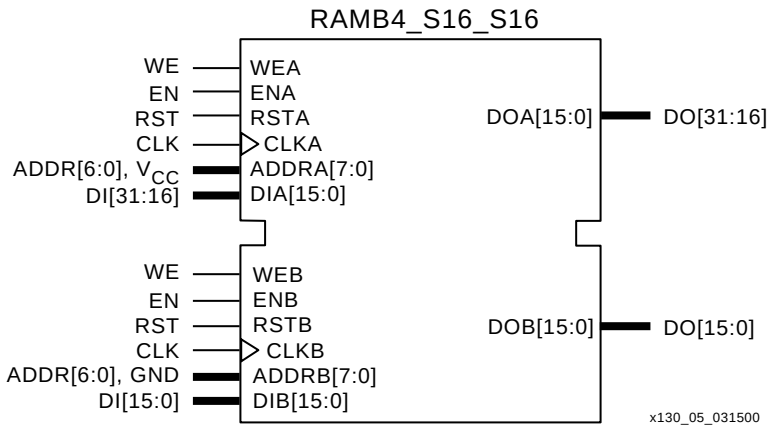

Figure 5: **Single-Port 128 x 32 RAM**

Creating Two Single-Port RAMs

The true dual-read/write port functionality of the block RAM allows a single RAM to be split into two single port memories of 2K bits each, as shown in Figure 6.

In this example, a 512K x 4 RAM (Port A) and a 128 x 16 RAM (Port B) are created out of a single block RAM. The address space for the RAM is split by fixing the MSB of Port A to 1 (V_{CC}) for the upper 2K bits and the MSB of Port B to 0 (GND) for the lower 2K bits.

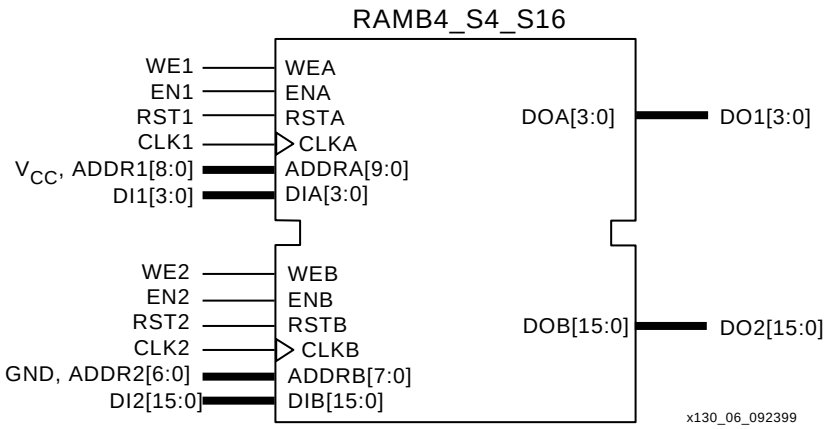

Figure 6: **512 x 4 RAM and 128 x 16 RAM**

Block Memory Generation

The CoreGen program generates memory structures using the block SelectRAM+ features. This program outputs VHDL or Verilog simulation code templates and an EDIF file for inclusion in a design.

Initialization

The block SelectRAM+ memory can initialize during the device configuration sequence. The 16 initialization properties of 64 hex values each (a total of 4096 bits) set the initialization of each RAM. These properties appear in Table 4. Any initialization properties that are not explicitly set to a value configure as zeros. Partial initialization strings pad with zeros. Initialization strings greater than 64 hex values generate an error.

The RAMs can be simulated with the initialization values using generics in VHDL simulators and parameters in Verilog simulators.

Initialization in VHDL and Synopsys

The block SelectRAM+ structures may be initialized in VHDL for both simulation and synthesis for inclusion in the EDIF output file. The simulation of the VHDL code uses a generic to pass the initialization. Synopsys FPGA compiler does not presently support generics; the initialization values instead are attached as attributes to the RAM by a built-in Synopsys **dc_script**. The **translate_off** statement stops synthesis translation of the generic statements. The following code illustrates a module that employs these techniques.

Table 4: **RAM Initialization Properties**

Property	Memory Cells
INIT_00	255 to 0
INIT_01	511 to 256
INIT_02	767 to 512
INIT_03	1023 to 768
INIT_04	1279 to 1024

Table 4: **RAM Initialization Properties *(Continued)***

Property	Memory Cells
INIT_05	1535 to 1280
INIT_06	1791 to 2047
INIT_07	2047 to 1792
INIT_08	2303 to 2048
INIT_09	2559 to 2304
INIT_0a	2815 to 2560
INIT_0b	3071 to 2816
INIT_0c	3327 to 3072
INIT_0d	3583 to 3328
INIT_0e	3839 to 3584
INIT_0f	4095 to 3840

VHDL Initialization Example

```
library IEEE;
use IEEE.std_logic_1164.all;

entity MYMEM is
port (CLK, WE:in std_logic;
ADDR: in std_logic_vector(8 downto 0);
DIN: in std_logic_vector(7 downto 0);
DOUT: out std_logic_vector(7 downto 0));
end MYMEM;

architecture BEHAVE of MYMEM is
signal logic0, logic1: std_logic;

component RAMB4_S8
--synopsys translate_off
generic( INIT_00,INIT_01, INIT_02, INIT_03, INIT_04, INIT_05, INIT_06,
INIT_07,
INIT_08, INIT_09, INIT_0a, INIT_0b, INIT_0c, INIT_0d, INIT_0e, INIT_0f :
BIT_VECTOR(255 downto 0)
:= X"0000000000000000000000000000000000000000000000000000000000000000");
--synopsys translate_on
port (WE, EN, RST, CLK: in STD_LOGIC;
ADDR: in STD_LOGIC_VECTOR(8 downto 0);
DI: in STD_LOGIC_VECTOR(7 downto 0);
DO: out STD_LOGIC_VECTOR(7 downto 0));
end component;

--synopsys dc_script_begin
--set_attribute ram0 INIT_00
"0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF" -type
string
--set_attribute ram0 INIT_01
"FEDCBA9876543210FEDCBA9876543210FEDCBA9876543210FEDCBA9876543210" -type
string
--synopsys dc_script_end

begin
logic0 <='0';
```

```
logic1 <='1';

ram0: RAMB4_S8
--synopsys translate_off
generic map (
INIT_00 =>
X"0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF",
INIT_01 =>
X"FEDCBA9876543210FEDCBA9876543210FEDCBA9876543210FEDCBA9876543210")
--synopsys translate_on
port map (WE=>WE, EN=>logic1, RST=>logic0, CLK=>CLK,ADDR=>ADDR, DI=>DIN,
DO=>DOUT);

end BEHAVE;
```

Initialization in Verilog and Synopsys

The block SelectRAM+ structures may be initialized in Verilog for both simulation and synthesis for inclusion in the EDIF output file. The simulation of the Verilog code uses a defparam to pass the initialization. The Synopsys FPGA compiler does not presently support defparam. The initialization values instead attach as attributes to the RAM by a built-in Synopsys dc_script. The translate_off statement stops synthesis translation of the defparam statements. The following code illustrates a module that employs these techniques.

Verilog Initialization Example

```
module MYMEM (CLK, WE, ADDR, DIN, DOUT);
input CLK, WE;
input [8:0] ADDR;
input [7:0] DIN;
output [7:0] DOUT;

wire logic0, logic1;

//synopsys dc_script_begin
//set_attribute ram0 INIT_00
"0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF" -type
string
//set_attribute ram0 INIT_01
"FEDCBA9876543210FEDCBA9876543210FEDCBA9876543210FEDCBA9876543210" -type
string
//synopsys dc_script_end

assign logic0 = 1'b0;
assign logic1 = 1'b1;

RAMB4_S8 ram0 (.WE(WE), .EN(logic1), .RST(logic0), .CLK(CLK), .ADDR(ADDR),
.DI(DIN), .DO(DOUT));
//synopsys translate_off
defparam ram0.INIT_00 =
256h'0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF;
defparam ram0.INIT_01 =
256h'FEDCBA9876543210FEDCBA9876543210FEDCBA9876543210FEDCBA9876543210;
//synopsys translate_on
endmodule
```

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
10/16/98	1.0	Initial release
09/24/99	1.1	Information added and updated in the complete document.
12/29/99	1.2	Reformatted document.
3/16/00	1.3	Revised Figure 5.
11/20/00	1.4	Reformatted document and corrected memory location on page 6.