Distributed and Block RAM Memory Internal to the FPGA

David R. Bull

March 25, 2003

This document discusses the implementation of distributed and block RAM in the Xilinx Spartan II FPGA.

There are two types of on-chip memory, namely:

1. Distributed
2. Block

Distributed RAM is built using the logic resources and is ideal for small RAMs that are closely integrated with external logic. Alternately, block RAM is implemented in dedicated blocks of true RAM divided into 4K sections. In the XSA100 there are 10, 4K blocks for a total of 40K.

Distributed and block RAM also differ in the way they operate. Distributed RAM is read asynchronously whereas in block RAM the address is latched on the clock edge.

Either memory can be instantiated using the library primitives or can be inferred in the VHDL code. A listing of the library primitives for the Spartan II FPGA can be found at http://toolbox.xilinx.com/docsan/xilinx5/data/docs/lib/lib0024_8.html.

Below are examples on inferring both distributed and block RAMs.

Example 1: Distributed RAM

Here we infer a 640-bit (32-by-20) single port distributed RAM. Data on the input lines is written to the RAM, at the location selected by the address lines, when the enable signal is high on the rising edge of the clock. The RAM is read asynchronously.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity distram is

  generic (
    portwidth : integer := 20;
    addwidth  : integer := 5);

  port (
    clk : in  std_logic;
    we  : in  std_logic;
    a   : in  std_logic_vector(addwidth-1 downto 0);
    di  : in  std_logic_vector(portwidth-1 downto 0);
    do  : out std_logic_vector(portwidth-1 downto 0));

end distram;

architecture ramarch of distram is

  type ram_type is array (2**addwidth-1 downto 0) of
    std_logic_vector (portwidth-1 downto 0);
  signal RAM : ram_type;

begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (we = '1') then
        RAM(conv_integer(a)) <= di;
      end if;
    end if;
  end process;
  do <= RAM(conv_integer(a));
end ramarch;
```

Example 2: Block RAM

Here we infer a 640-bit (32-by-20) single port Block RAM. Like the distributed RAM, the data on the input lines is written to the RAM, at the location selected by the address lines, when the enable signal is high on the rising edge of the clock. Unlike distributed RAM, the address is latched on the clock edge, so the RAM can only be read synchronously.

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity rambank is

  generic (
    portwidth : integer := 20;
    addwidth  : integer := 5);

  port (
    clk : in  std_logic;
    we  : in  std_logic;
    a   : in  std_logic_vector(addwidth-1 downto 0);
    di  : in  std_logic_vector(portwidth-1 downto 0);
    do  : out std_logic_vector(portwidth-1 downto 0));

end rambank;

architecture ramarch of rambank is

  type ram_type is array (2**addwidth-1 downto 0) of
    std_logic_vector (portwidth-1 downto 0);
  signal RAM    : ram_type;
  signal read_a : std_logic_vector(addwidth-1 downto 0);

begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (we = '1') then
        RAM(conv_integer(a)) <= di;
      end if;
      read_a <= a;
    end if;
  end process;
  do <= RAM(conv_integer(read_a));
end ramarch;
```

Other resources that discuss implementing on-chip RAM are the XST User Guide and Xilinx Application Note XAPP173, which can be found on the Xilinx web page (www.xilinx.com).