Block RAM

- Dedicated FPGA resource, separate columns from CLBs
- Designed to implement large (Kb) memories
- Multi-port capabilities
- Multi-clock capabilities
- FIFO capabilities
- Built-in error detection and correction
Block RAM

• Size
  – Virtex-4 and older: 18Kb
  – Virtex-5 and newer: 36Kb, can function as two 18Kb blocks

• Ports
  – Native size 18b/36b (16/32b data and 2/4b ECC spare capability)
  – Variable aspect ratio: 1b to 72b wide
  – Variable depth: 1K to 36K words
HDL Guide to Block RAM

• Automatic inference of block RAM requires:
  – Synchronous Write
  – Synchronous Read (Asynch read will infer Distributed RAM)

• Block RAM may use:
  – Optional “Enable” signal for the RAM, which must have higher priority than the “Write Enable”, and must affect both the read and write operations
  – Optional per-byte write enables
  – Optional output pipeline registers
HDL Guide to Block RAM

• Correct code:
  
  ```verilog
  always @(posedge clock) begin
      if (en) begin
          if (we)            ram[addr] <= wdata;
          rdata <= ram[addr];
      end
  end
  end
  ```

• Code results in “Read-First” Block RAM
• Useful for Read-and-Replace application (e.g., write-back caches)
HDL Guide to Block RAM

• Misleading code:
  ```
  always @(posedge clock) begin
    if (en) begin
      if (we) 
        ram[addr] <= wdata;
    end 
    rdata <= ram[addr];
  end
  ```
• Code results in always-enabled block RAM and Write-enable connected to (en & we)
• Bad input timing (1 LUT delay) and extra area (1 LUT)
HDL Guide to Block RAM

• Correct code:
  
  ```verilog
  always @(posedge clock)
  if (en) begin
    if (we) begin
      ram[addr] <= wdata;
      rdata <= wdata;
    end else
    rdata <= ram[addr];
  end
  ```

• Results in “Write-First” Block RAM

• Useful for write-through applications (e.g., Write-Through Caches)
HDL Guide to Block RAM

- Correct code:
  ```verilog
  always @(posedge clock)
    if (en) begin
      if (we) begin
        ram[addr] <= wdata;
      end else
        rdata <= ram[addr];
    end
  end
  ```

- Results in “No-Change” Block RAM

- Exercise: how to code per-byte write enables?
HDL Guide to Block RAM

• Improve output timing by using optional output pipeline register:
  – Describe an extra register in HDL on the output data path
  – Available control signals are architecture-dependent; on Virtex-4:
    • No reset (otherwise will be implemented with Slice FFs)
    • No clock enable or same enable signal as RAM port (otherwise will be implemented with Slice FFs)

• Output timing is 0.72ns with the pipeline register, 1.64ns without (!!)
HDL Guide to Block RAM

• Dual port modes:
  – Simple dual port (1 read port, 1 write port)
  – True dual port (1 read-write port, 1 read/write/read-write port)
  – Combinations of write modes are possible (e.g., read-first on port A and write-first on port B)

• Clocking
  – Port clocks are independent (asynchronous ports)
  – Synchronous ports simulated by connecting same clock to both clock ports
In dual-port sync/async modes, ports can conflict
- A conflict occurs when both ports access the same memory location and at least one access is for write

Conflict effects:
- In asynchronous mode, simultaneous write/write or read/write access to the same location results in corrupted data
- In synchronous mode, WRITE_FIRST or NOCHANGE ports become corrupted on read/write accesses
- READ_FIRST allows read/write access

!! HDL descriptions will not properly simulate conflicts !!
HDL Guide to Block RAM

- Output latch and RAM initialization

    initial begin
      rdataA = 0;
      rdataB = 1;
      ram[0] = 'h1234;
      ram[1] = 'h5678;
      //ALL the RAM locations
    end

- Missing RAM locations will result in ignored initialization
- Modern tools will initialize RAM contents on configuration
- Registers are also initialized on reset (Local or GSR)
Alternative uses of Block RAM

• Block RAM can be used to implement logic (functions as ROM memory)
  – Determine the number of inputs and outputs of your logic function
  – Inputs form the address (like the LUT structure)
  – If truth table of logic function is less than Block RAM size, truth table can be implemented using INIT parameters (or “initial” blocks)
  – E.g. 18 different 10-input functions simultaneously in a 18Kb Block RAM
Features not available in HDL

• Asymmetric port widths (case by case)
• Built-in FIFO logic
• Built-in parity/ECC logic
• Instantiated as macros, primitives or CoreGen blocks
HDL Guide to Asymmetric Ports

• HDL description of asymmetric ports available for Virtex-6 or newer, on ISE 14.2
  – Least wide port defines RAM parameters (word width and number of words)
  – Wider port utilizes several RAM locations for reads or writes
Further Reading

- Virtex-4 FPGA User Guide, Chapter 4
- Virtex-6 FPGA Memory Resources User Guide
- 7-Series FPGA Memory Resources User Guide
- XST User Guide, 7-Series, Spartan-6 and Virtex-6, Chapter 7