Lab 4a Tutorial

Data Memory Module

OVERVIEW

In this lab we will implement and simulate a memory unit that is capable of reading and writing data within a clock cycle. The data memory functions are listed in the table below.

en	wen	Function	data_out
0	X	N/A	0
1	0	Read	M[addr]
1	1	Write M[addr] <= data_in	0

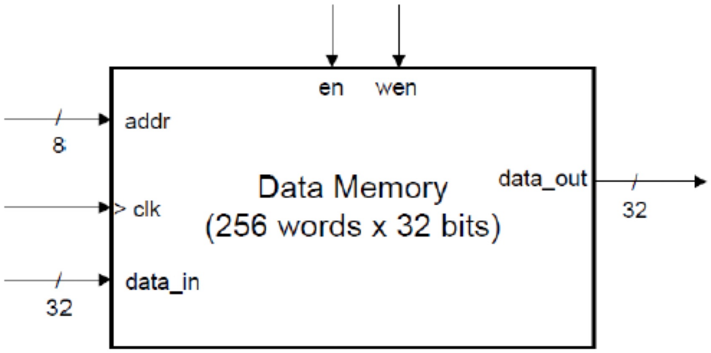


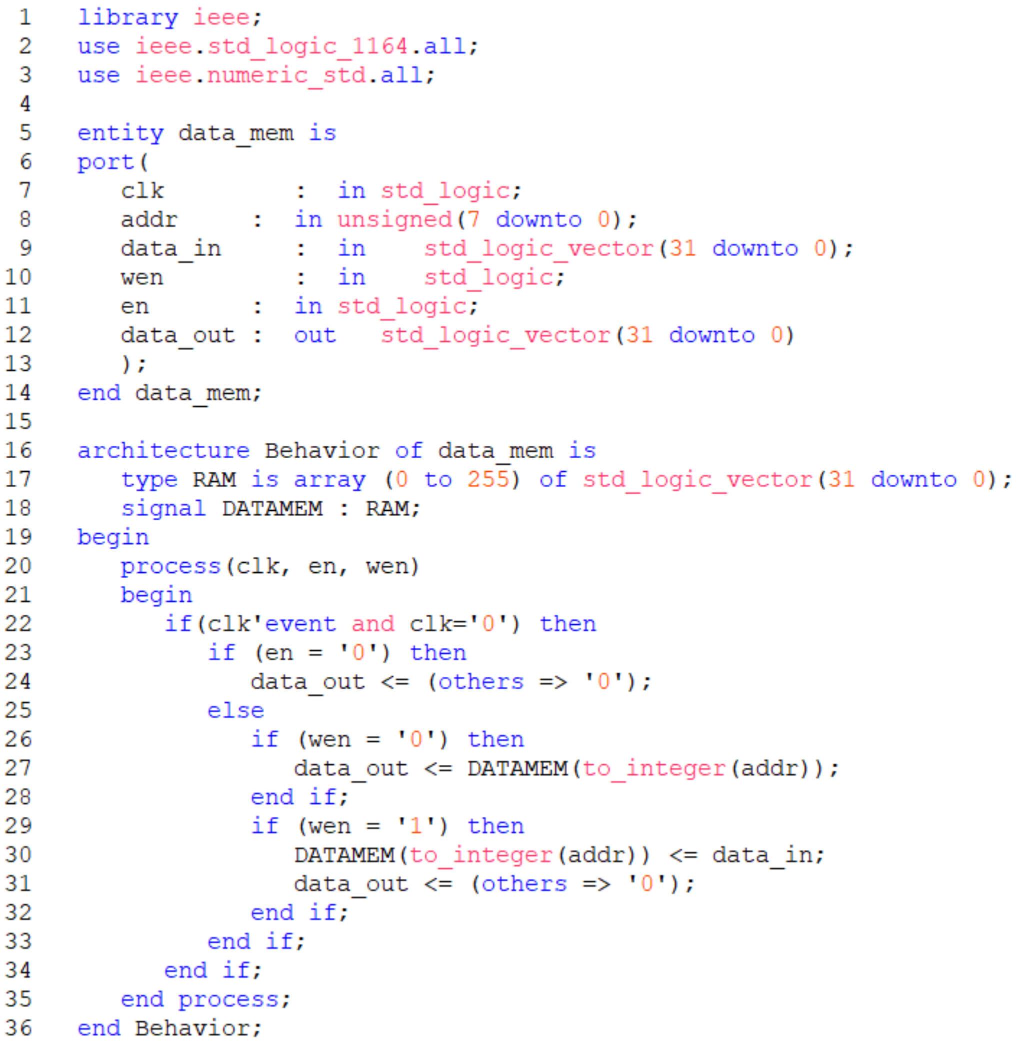
Figure 1: 256-word x 32-bit Data Memory Unit

Figure 1: 256-word x 32-bit Data Memory Unit

The memory unit has an 8-bit address input (addr), 32-bit data input and output (data_in and data_out), and other control lines such as clock, write enable (wen), and enable (en). There are many ways to implement a memory module, however memories are typically implemented as a 2D array of registers. The data memory unit designed in this lab will also be simulated on the Cyclone IV device.

PROCEDURE

Data Memory Module (data_mem.vhd)

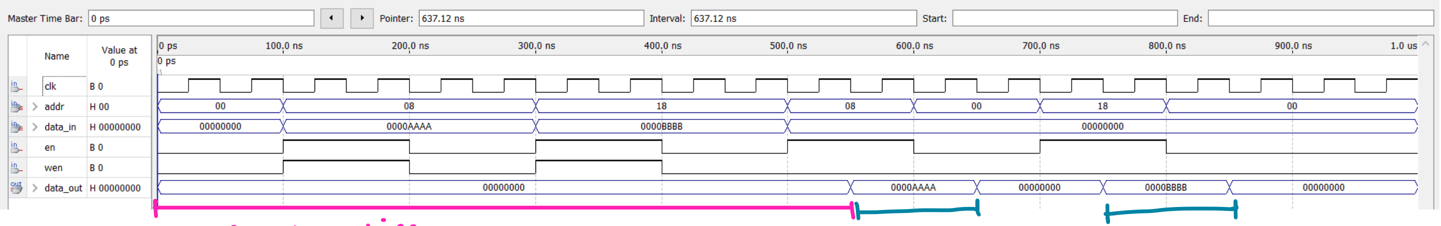


```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity data_mem is
port(
   clk         :  in std_logic;
   addr     :  in unsigned(7 downto 0);
   data_in     :  in    std_logic_vector(31 downto 0);
   wen         :  in    std_logic;
   en       :  in std_logic;
   data_out :  out   std_logic_vector(31 downto 0)
   );
end data_mem;

architecture Behavior of data_mem is
   type RAM is array (0 to 255) of std_logic_vector(31 downto 0);
   signal DATAMEM : RAM;
begin
   process(clk, en, wen)
   begin
      if(clk'event and clk='0') then
         if (en = '0') then
            data_out <= (others => '0');
         else
            if (wen = '0') then
               data_out <= DATAMEM(to_integer(addr));
            end if;
            if (wen = '1') then
               DATAMEM(to_integer(addr)) <= data_in;
               data_out <= (others => '0');
            end if;
         end if;
      end if;
   end process;
end Behavior;
```

FUNCTION SIMULATION



Master Time Bar: 0 ps Pointer: 637.12 ns Interval: 637.12 ns Start: End:

Name	Value at 0 ps
clk	B 0
addr	H 00
data_in	H 00000000
en	B 0
wen	B 0
data_out	H 00000000

en = 1 } writing
wen = 1 } data into
data_mem unit

en = 1 } reading
wen = 0 } stored data
from data_mem
unit.