Department of Electrical, Computer, & Biomedical Engineering
Faculty of Engineering & Architectural Science

Course Title:	Computer Organization and Architecture
Course Number:	COE 608
Semester/Year (e.g. F2017)	W2023

Instructor	Demetres Kostas

Assignment/Lab Number:	4a
Assignment/Lab Title:	Data Memory

Submission Date:	March 1, 2023
Due Date:	March 1, 2023

Student LAST Name	**Student FIRST Name**	**Student Number**	**Section**	**Signature***
Hernandez	Spencer	501033228	11	Spencer H
Dumlao	Carlo	501018239	11	C.D

*By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: http://www.ryerson.ca/senate/current/pol60.pdf

Summary

The purpose of this lab is to construct a data memory for the cpu project.

Data Memory

Description

The purpose of the data memory is to store 1024 bits of data organized as an array of 256 data slots containing 32 bits of data. Each data slot can be accessed when provided with an address which is then sent to the output terminal. There are three operating modes of the data memory: read, write and high impedance. During read mode, the data memory can output any value when provided with an address, during write mode, the data memory can override any data slot provided with an address and input value. During high impedance mode, the data memory outputs neither a high or low value and allows for the flow of data. These operating modes can be accessed using the “en” and “wen” inputs which control what mode is in use.

VHDL Code

```
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;
```

Waveform

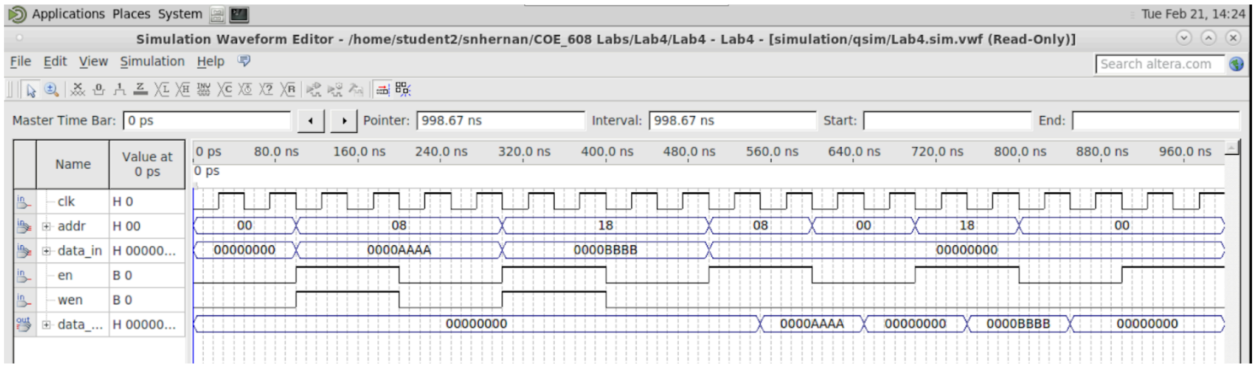

Figure 1: Functional waveform of the data memory

Explanation

As you can see, this is a sequential device which operates on a rising edge. when “en” and “wen” are both equal to 1 then the device operates in write mode. As you can see on the first clock the value “0000AAAA” is being loaded at address 8 of the device then “0000BBBB” at address 18 at the next clock cycle. Next when “en” is set to high, it is operating at read mode which the value “0000AAAA” being the output since the value “08” is given as the address. Similarly, the same thing occurs when the value “18” is fed into the address input; its corresponding value of “0000BBBB” is also observed to be the output.

Discussion

Based on the timing waveform of the data memory in Figure 2 below, the worst case path T_{WC} is independent of the address input *addr* when the write pin $wen = LOW$. Notice that in the time interval $[500ns,\ 1000ns]$, this component requires approximately $T_{WC} = 60ns$ or $f_{WC} = 16.7GHz$ to properly read a data from a specified address, as well as to disable it which sets the output to a 0 value. Put it simply, this designed data memory is able to read up to 16.7 million instructions (32-bit) per second – accessing more than this will surely make the component to output unwanted data. In terms of writing data into the data memory, the delay is much shorter – in Figure 3, specifically at $t = 100ns$, this delay is approximately $40ns$ or $25GHz$.

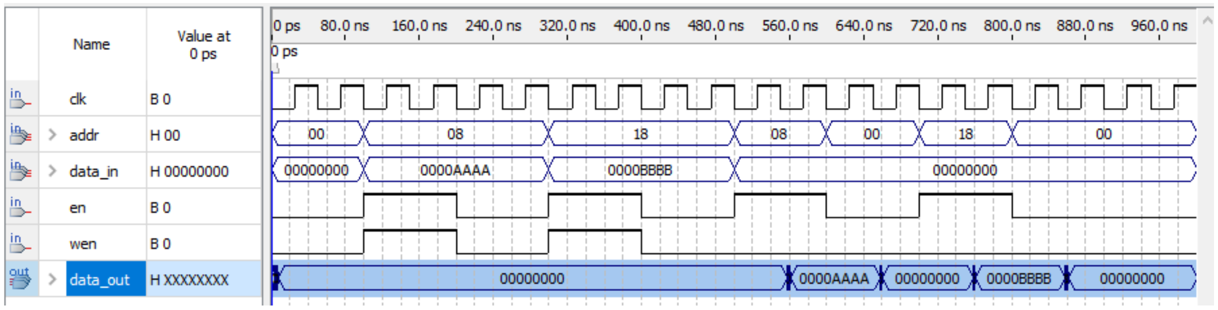

Figure 2: Timing waveform of the data memory

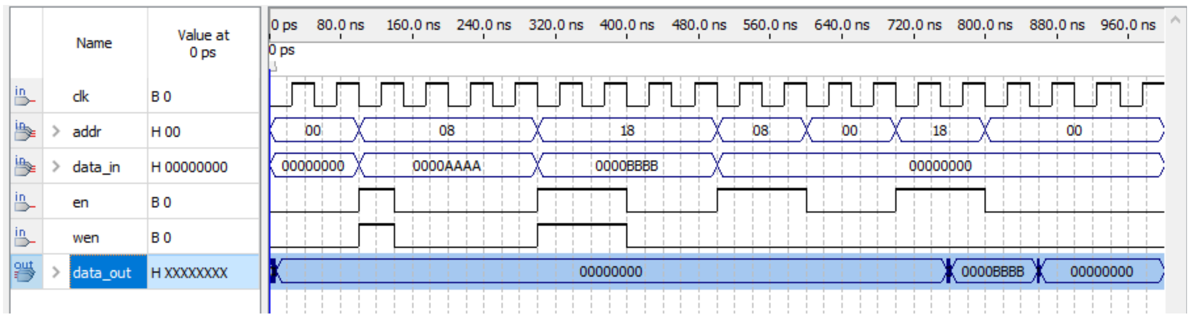

Figure 3: Timing waveform showing the effect of writing data in a short amount of time at $t = 100ns$