Ryerson University

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

Course Title:	Computer Organization and Architecture
Course Number:	COE608
Semester/Year:	W2024

Instructor:	Dr. Khalid Abdel Hafeez

Lab Number:	6
Lab Title:	Complete CPU

Submission Date:	Apr. 12, 2024
Due Date:	Apr. 12, 2024

Last Name	First Name	Student Number	Section Number	Signature
Thangeswaran	Charran			

Objective

The objective of this lab is to implement a complete CPU that consists of the data-path and the control unit that have been designed and implemented in Labs 4b and 5 respectively, allowing the overall design to implement the features described in the CPU specification document.

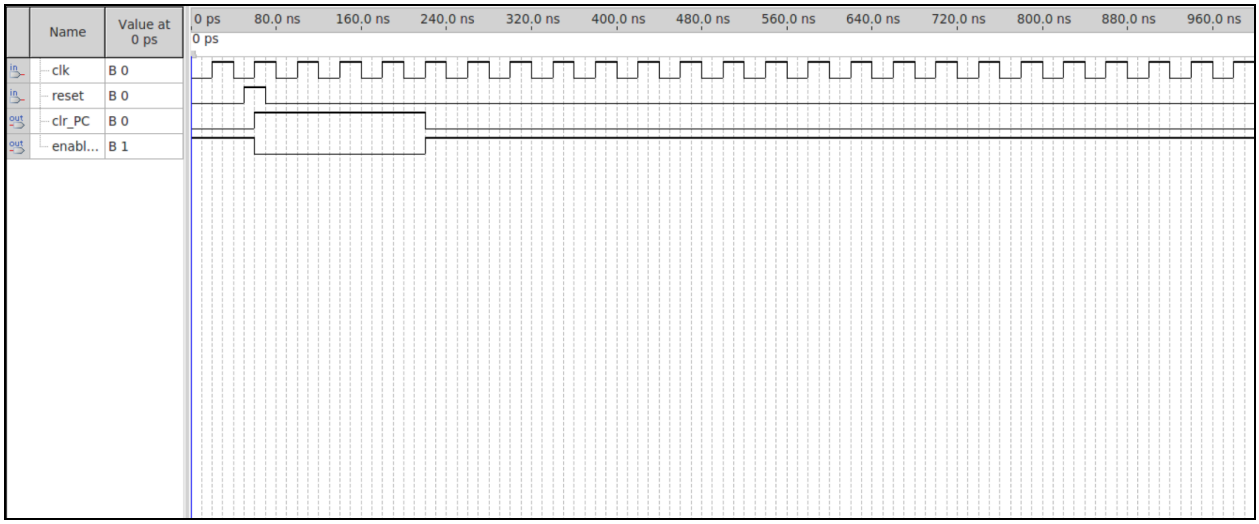
Experiment & Results

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

entity reset_circuit is
    port(
        reset : in std_logic;
        clk : in std_logic;
        enable_PD : out std_logic :='1';
        clr_PC : out std_logic
    );
end reset_circuit;

architecture Behavior of reset_circuit is
    type clkNum is (clk0, clk1, clk2, clk3);
    signal present_clk: clkNum;
begin
    process(clk)begin
        if rising_edge(clk) then
            if reset = '1' then
                clr_PC <= '1';
                enable_PD <= '0';
                present_clk <= clk0;
            elsif present_clk <= clk0 then
                present_clk <= clk1;
            elsif present_clk <= clk1 then
                present_clk <= clk2;
            elsif present_clk <= clk2 then
                present_clk <= clk3;
            elsif present_clk <= clk3 then
                clr_PC <= '0';
                enable_PD <= '1';
            end if;
        end if;
    end process;
end Behavior;
```

Reset Circuit VHDL Code



Reset Circuit Waveform Simulation

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

ENTITY CPU_TEST_Sim IS
   PORT(
      cpuClk : in std_logic;
      memClk : in std_logic;
      rst : in std_logic;
      -- Debug data.
      outA, outB : out std_logic_vector(31 downto 0);
      outC, outZ : out std_logic;
      outIR : out std_logic_vector(31 downto 0);
      outPC : out std_logic_vector(31 downto 0);
      -- Processor-Inst Memory Interface.
      addrOut : out std_logic_vector(5 downto 0);
      wEn : out std_logic;
      memDataOut : out std_logic_vector(31 downto 0);
      memDataIn : out std_logic_vector(31 downto 0);
      -- Processor State
      T_Info : out std_logic_vector(2 downto 0);
      --data Memory Interface
      wen_mem, en_mem : out std_logic);
END CPU_TEST_Sim;

ARCHITECTURE behavior OF CPU_TEST_Sim IS
   COMPONENT system_memory
      PORT(
         address  : IN  STD_LOGIC_VECTOR (5 DOWNTO 0);
```

```
        clock     : IN  STD_LOGIC ;
        data      : IN  STD_LOGIC_VECTOR (31 DOWNTO 0);
        wren      : IN  STD_LOGIC ;
        q         : OUT STD_LOGIC_VECTOR (31 DOWNTO 0)
);
END COMPONENT;

  COMPONENT cpu1
     PORT(
        clk       : in std_logic;
        mem_clk   : in std_logic;
        rst       : in std_logic;
        dataIn    : in std_logic_vector(31 downto 0);
        dataOut   : out std_logic_vector(31 downto 0);
        addrOut   : out std_logic_vector(31 downto 0);
        wEn : out std_logic;
        dOutA, dOutB : out std_logic_vector(31 downto 0);
        dOutC, dOutZ : out std_logic;
        dOutIR : out std_logic_vector(31 downto 0);
        dOutPC : out std_logic_vector(31 downto 0);
        outT : out std_logic_vector(2 downto 0);
        wen_mem, en_mem : out std_logic);
  END COMPONENT;

  signal cpu_to_mem: std_logic_vector(31 downto 0);
  signal mem_to_cpu: std_logic_vector(31 downto 0);
  signal add_from_cpu: std_logic_vector(31 downto 0);
  signal wen_from_cpu: std_logic;
```

```
BEGIN
 -- Component instantiations.
  main_memory : system_memory
     PORT MAP(
         address => add_from_cpu(5 downto 0),
         clock => memClk,
         data => cpu_to_mem,
         wren => wen_from_cpu,
         q => mem_to_cpu
      );
  main_processor : cpu1
     PORT MAP(
      clk => cpuClk,
      mem_clk => memClk,
      rst => rst,
      dataIn => mem_to_cpu,
      dataOut => cpu_to_mem,
      addrOut => add_from_cpu,
      wEn => wen_from_cpu,
      dOutA => outA,
      dOutB => outB,
      dOutC => outC,
      dOutZ => outZ,
      dOutIR => outIR,
      dOutPC => outPC,
      outT => T_Info,
      wen_mem => wen_mem,
```

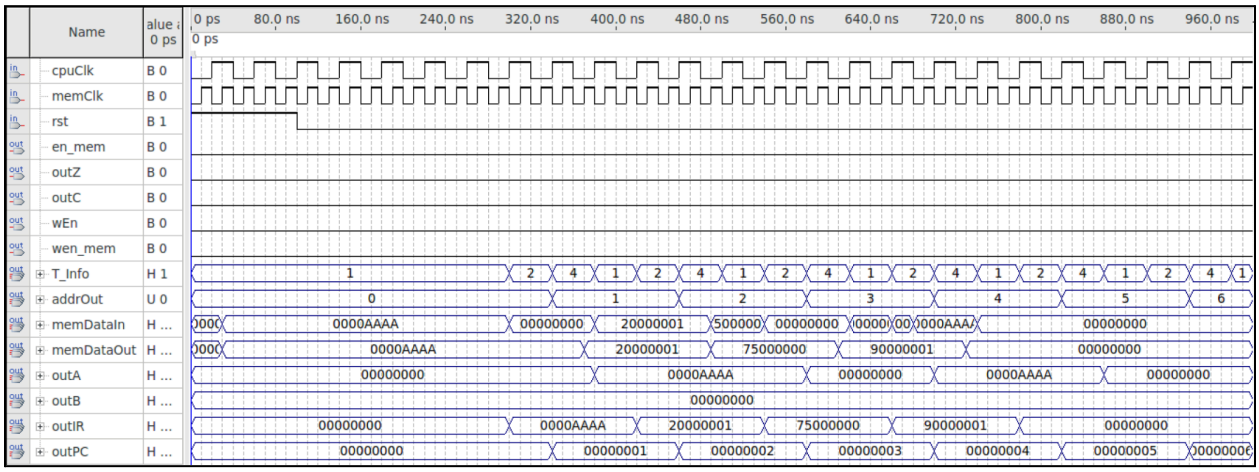
```
      en_mem => en_mem
      );

  addrOut <= add_from_cpu(5 downto 0);
  wEn <= wen_from_cpu;
  memDataOut <= mem_to_cpu;
  memDataIn <= cpu_to_mem;
END behavior;
```

CPU Test Simulation VHDL Code

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	0000AAAA	20000001	75000000	90000001	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

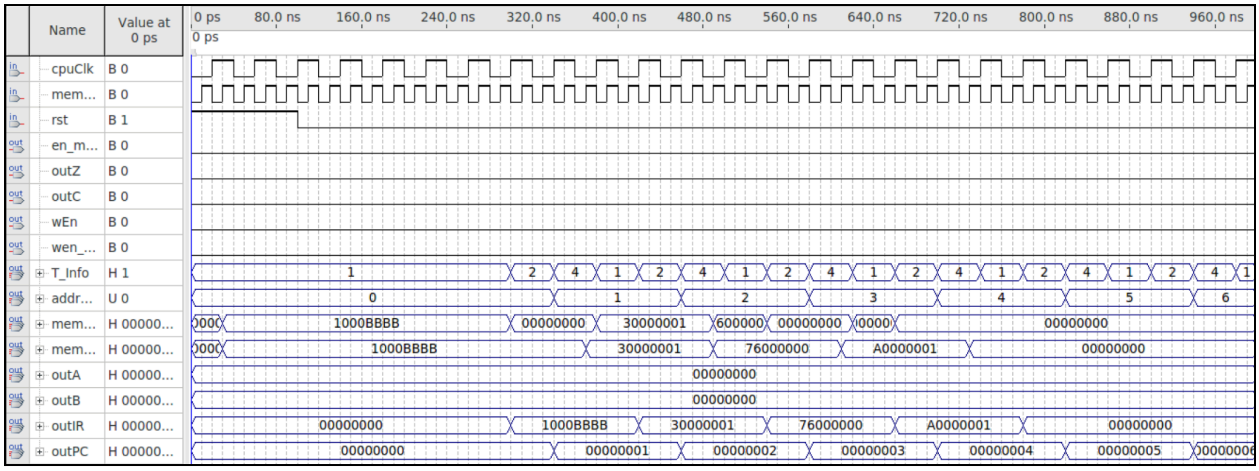
System Memory Implementation (LDAI, STA, CLRA, LDA)



Functional Simulation (LDAI, STA, CLRA, LDA)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	1000BBBB	30000001	76000000	A0000001	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

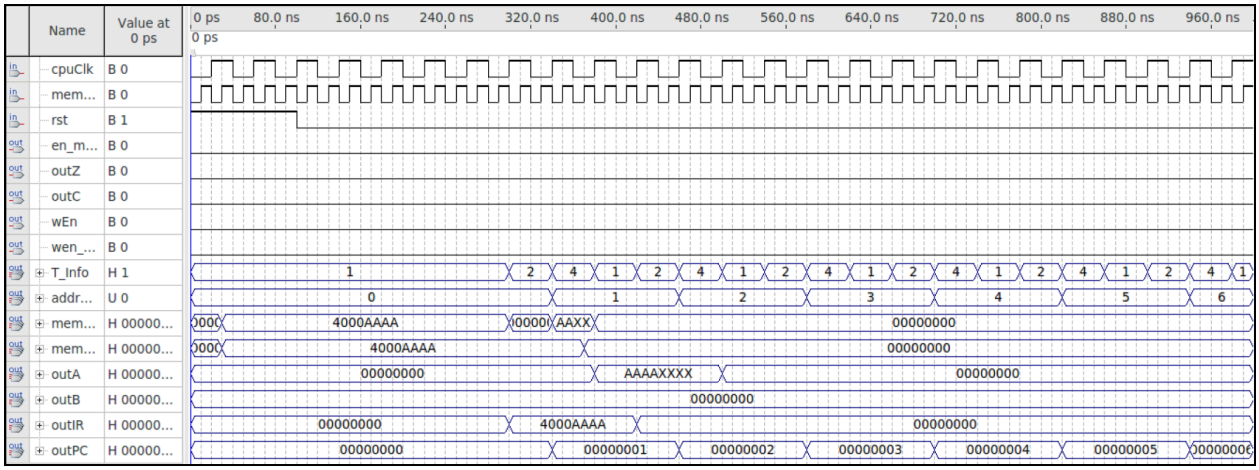
System Memory Implementation (LDBI, STB, CLRB, LDB)



Functional Simulation (LDBI, STB, CLRB, LDB)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	4000AAAA	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

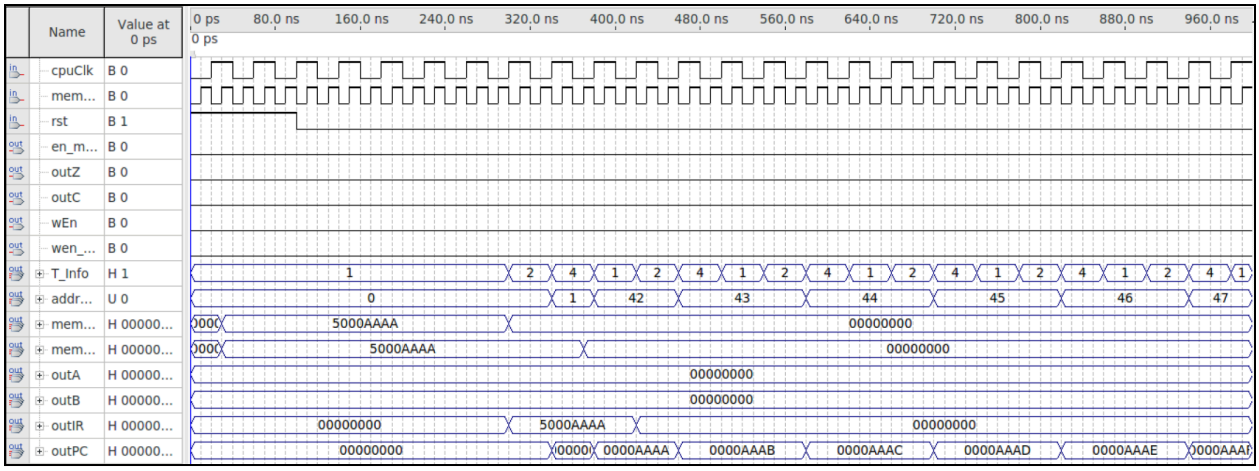
System Memory Implementation (LUI)



Functional Simulation (LUI)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	5000AAAA	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

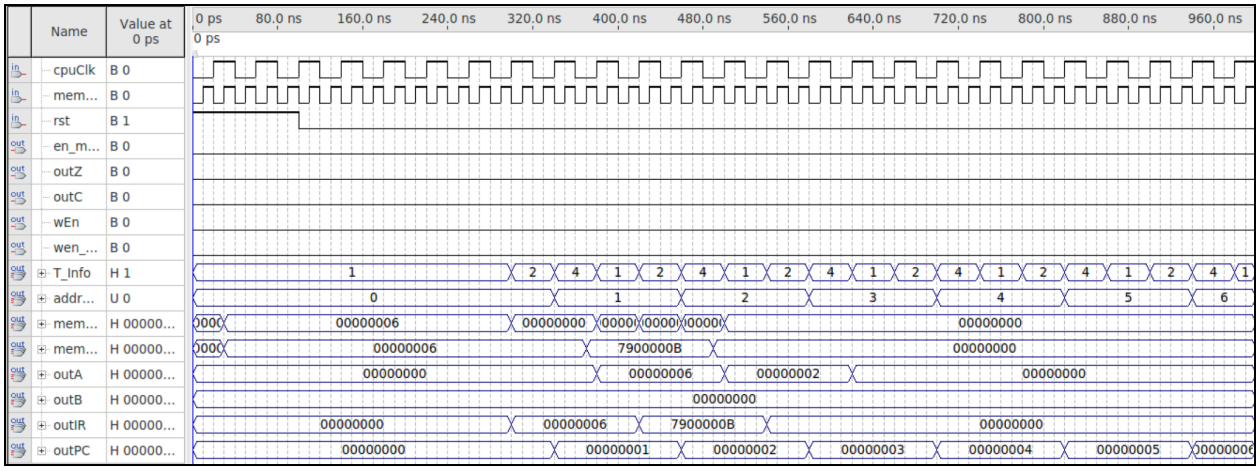
System Memory Implementation (JMP)



Functional Simulation (JMP)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	00000006	7900000B	00000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

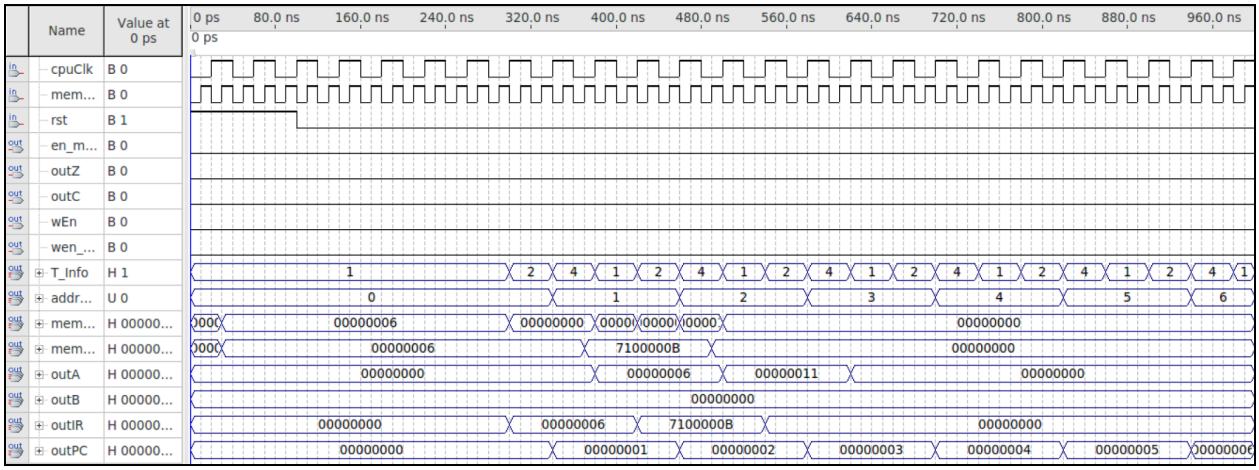
System Memory Implementation (ANDI)



Functional Simulation (ANDI)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	00000006	7100000B	00000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

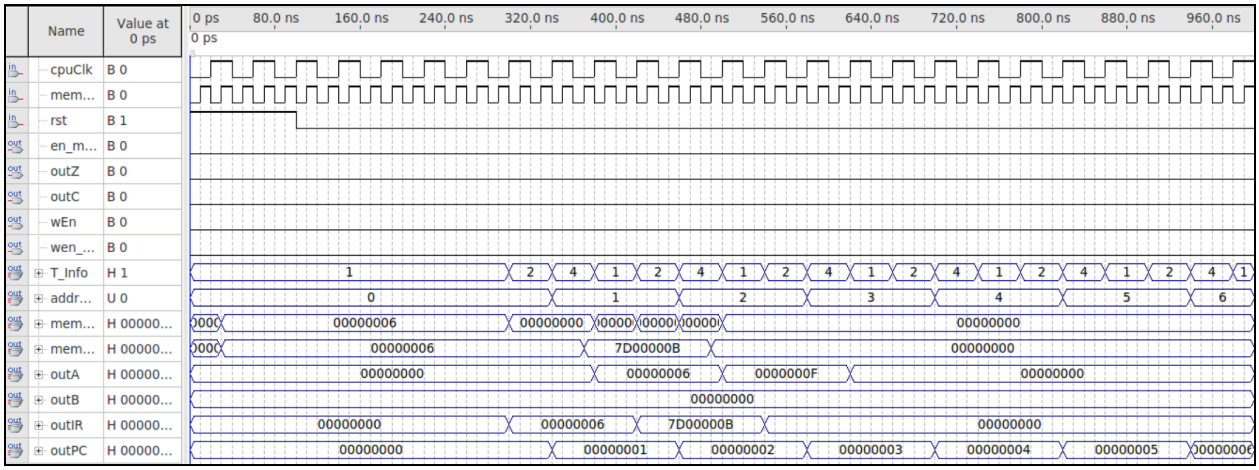
System Memory Implementation (ADDI)



Functional Simulation (ADDI)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	00000006	7D00000B	00000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

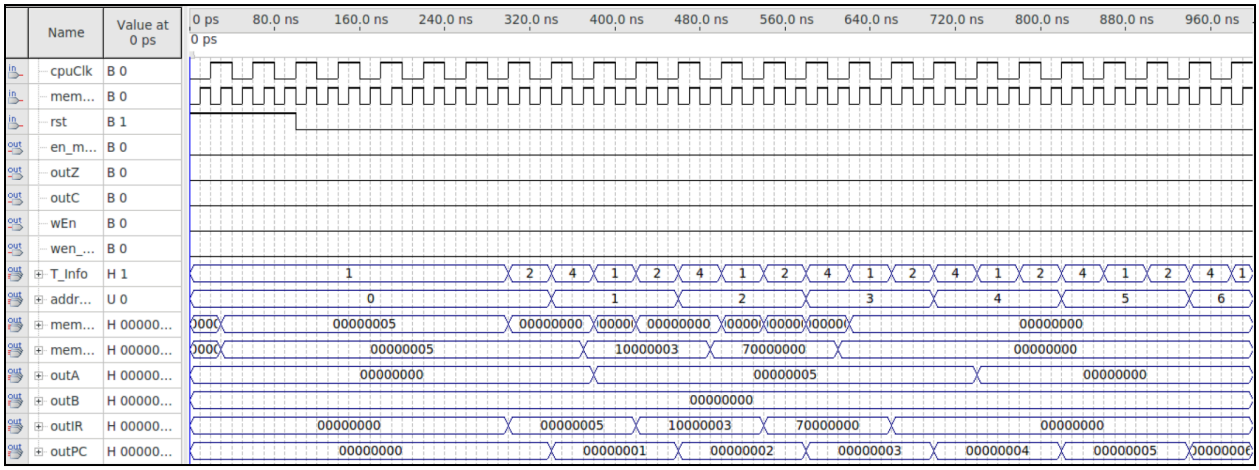
System Memory Implementation (ORI)



Functional Simulation (ORI)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	00000005	10000003	70000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

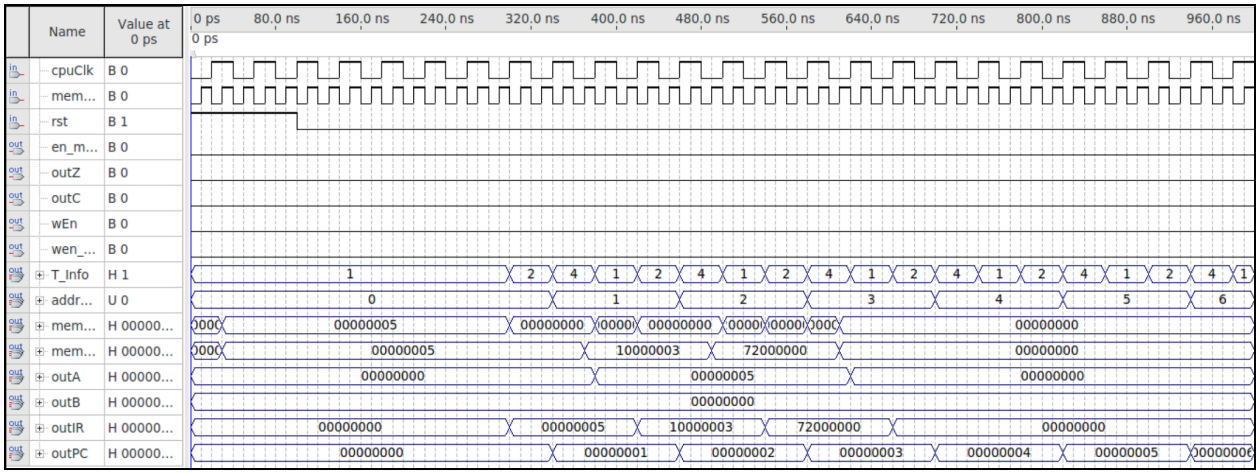
System Memory Implementation (ADD)



Functional Simulation (ADD)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	00000005	10000003	72000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

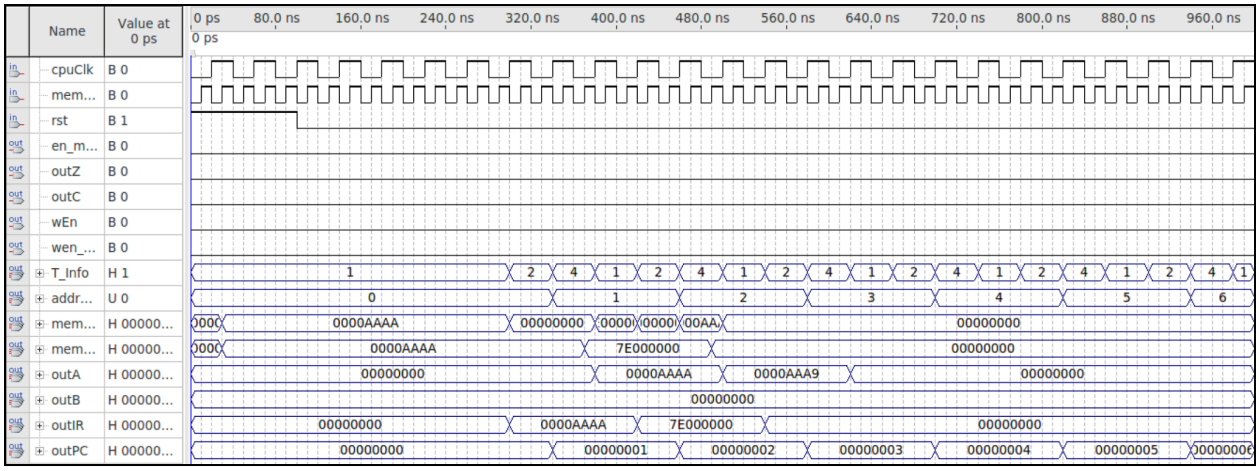
System Memory Implementation (SUB)



Functional Simulation (SUB)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	0000AAAA	7E000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

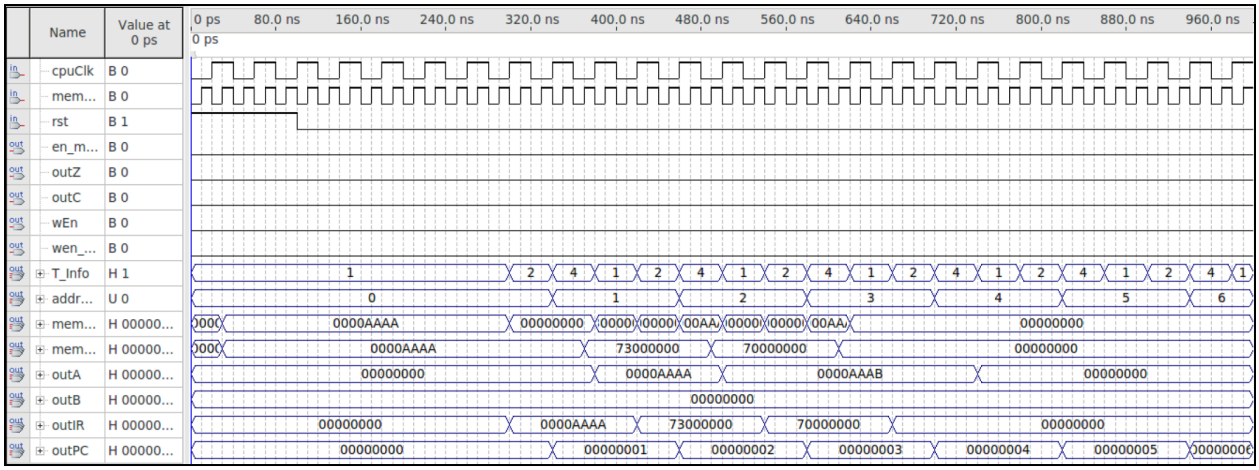
System Memory Implementation (DECA)



Functional Simulation (DECA)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	0000AAAA	73000000	70000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

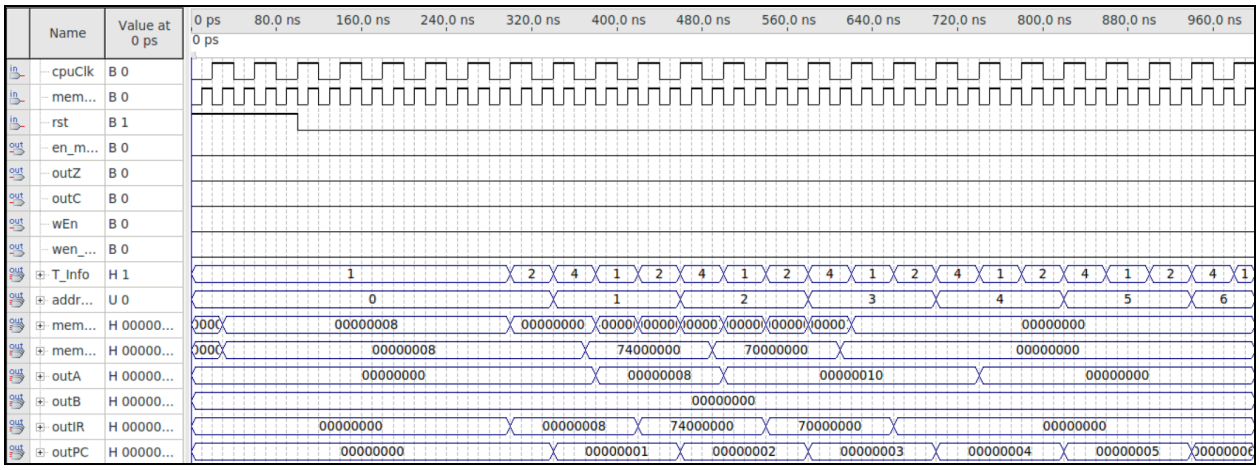
System Memory Implementation (INCA)



Functional Simulation (INCA)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	00000008	74000000	70000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

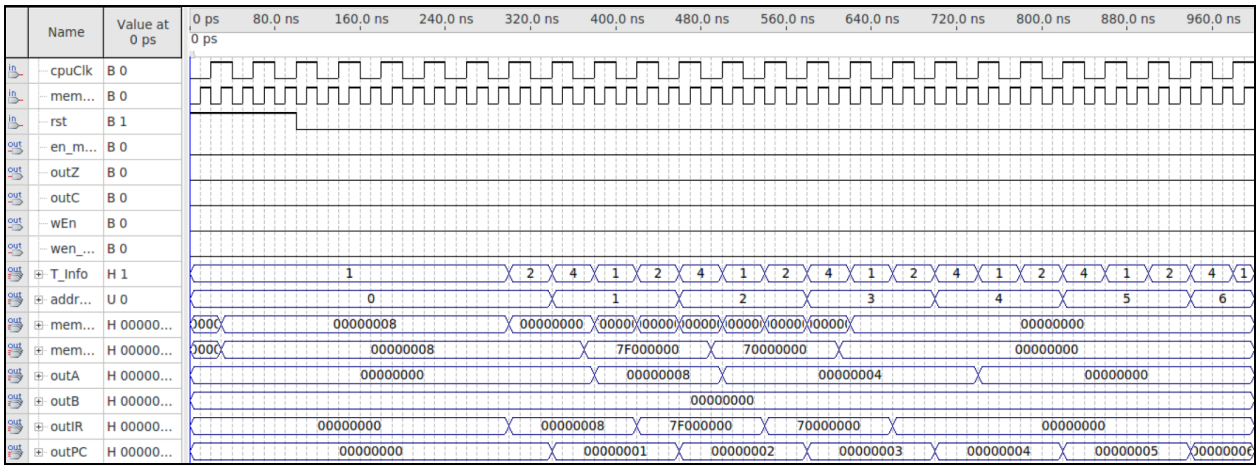
System Memory Implementation (ROL)



Functional Simulation (ROL)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	00000008	7F000000	70000000	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

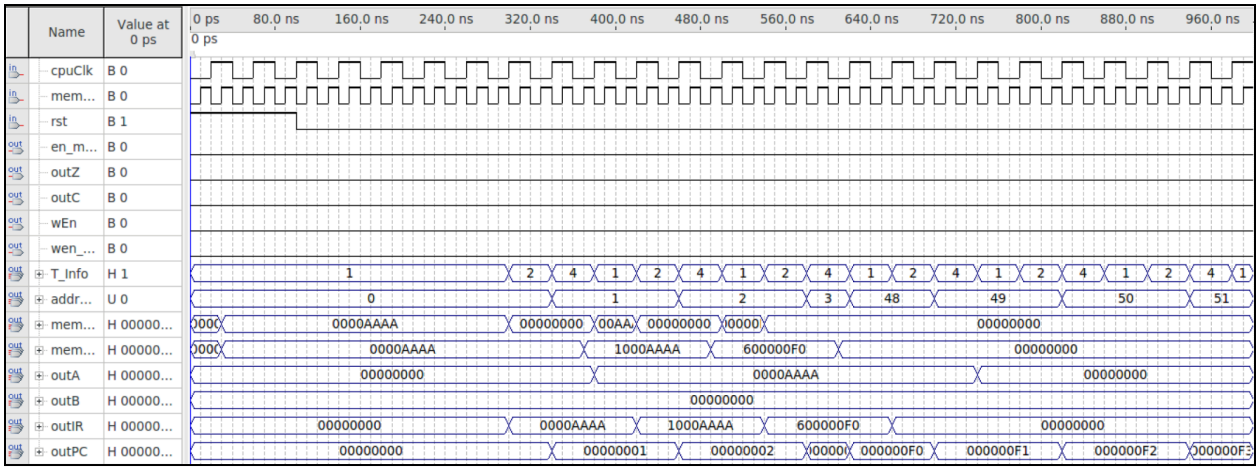
System Memory Implementation (ROR)



Functional Simulation (ROR)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	0000AAAA	1000AAAA	600000F0	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

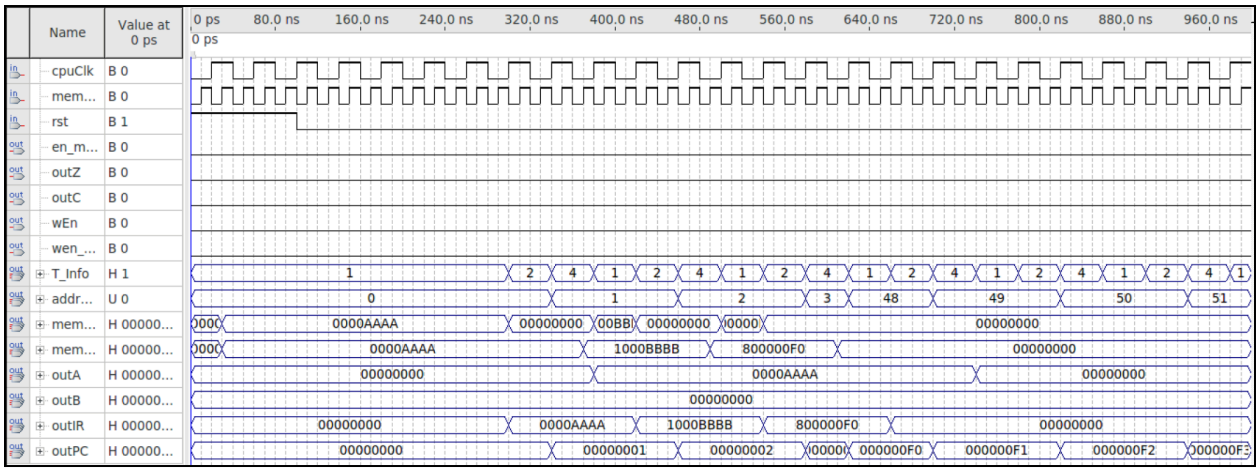
System Memory Implementation (BEQ)



Functional Simulation (BEQ)

Addr	+0	+1	+2	+3	+4	+5	+6	+7	ASCII
0	0000AAAA	1000BBBB	800000F0	00000000	00000000	00000000	00000000	00000000	.....
8	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
16	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
24	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
32	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
40	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
48	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....
56	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	.....

System Memory Implementation (BNE)



Functional Simulation (BNE)

Discussion

The CPU was able to demonstrate the correct operation as shown with the simulation results, thanks to the manipulation of the System Memory.

References

Al-Qawasmi M. (2021). Lab 6 Tutorial: The Complete CPU (Overall Project). *COE 608 COMPUTER ORGANIZATION AND ARCHITECTURE.*

Toronto Metropolitan University. (2024). COE 608 Lab 6 – The Complete CPU (Overall Project).