Lab 7 Report

ECE 4525

Mack Usmanova

Jena Francis

10/22/25

Introduction

The purpose of this lab was to design and implement a controller for a DRAM chip using VHDL. The main goal of this lab was to design a controller that can execute both read and write operations for a DRAM chip while meeting its AC timing specifications. The first task involved developing VHDL code to control a simple read cycle. We ensured the DRAM control signals were correctly generated, and timing constraints were verified through post-route simulation and the oscilloscope. The second task was similar but instead of a read cycle we developed a write cycle. Again, timing constraints were verified through post-route simulation and the oscilloscope. Lastly, task three involved integrating both read and write operations into one. Overall, this lab provided valuable experience in timing diagrams and DRAM implementation.

Procedure

Task One:

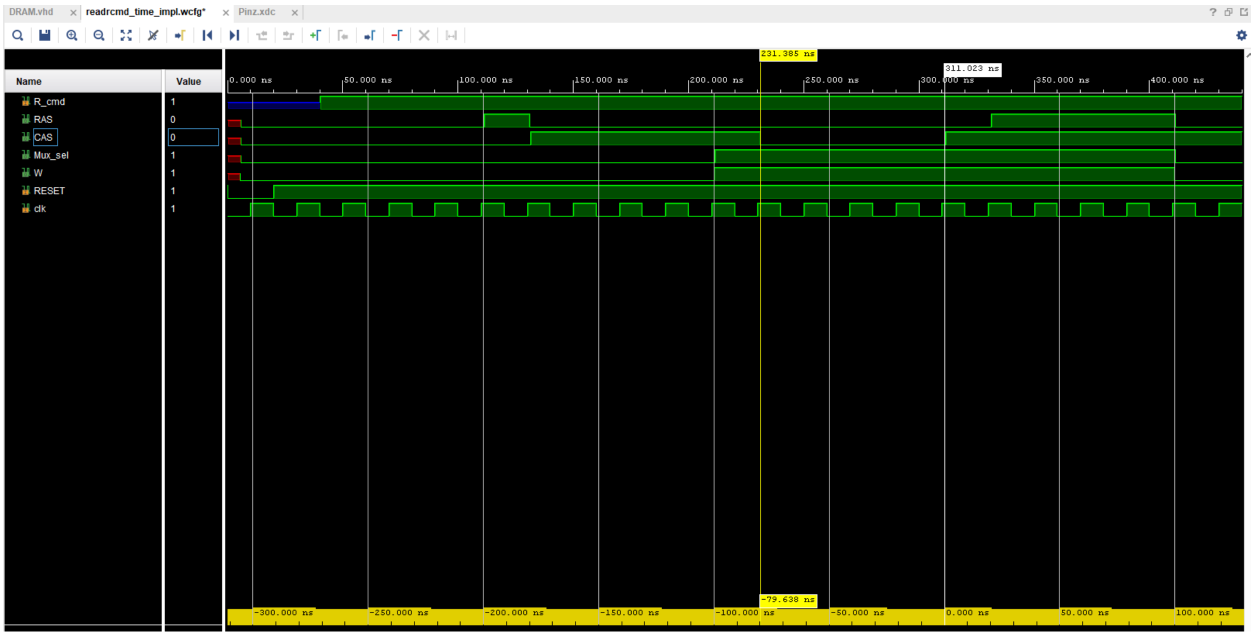
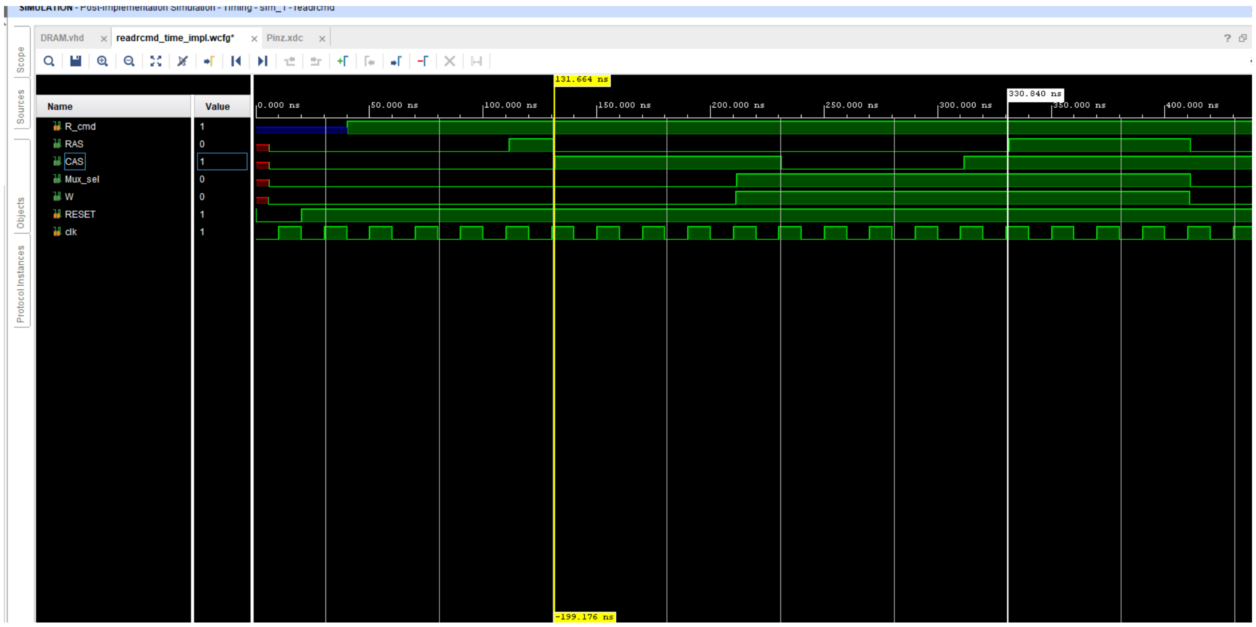
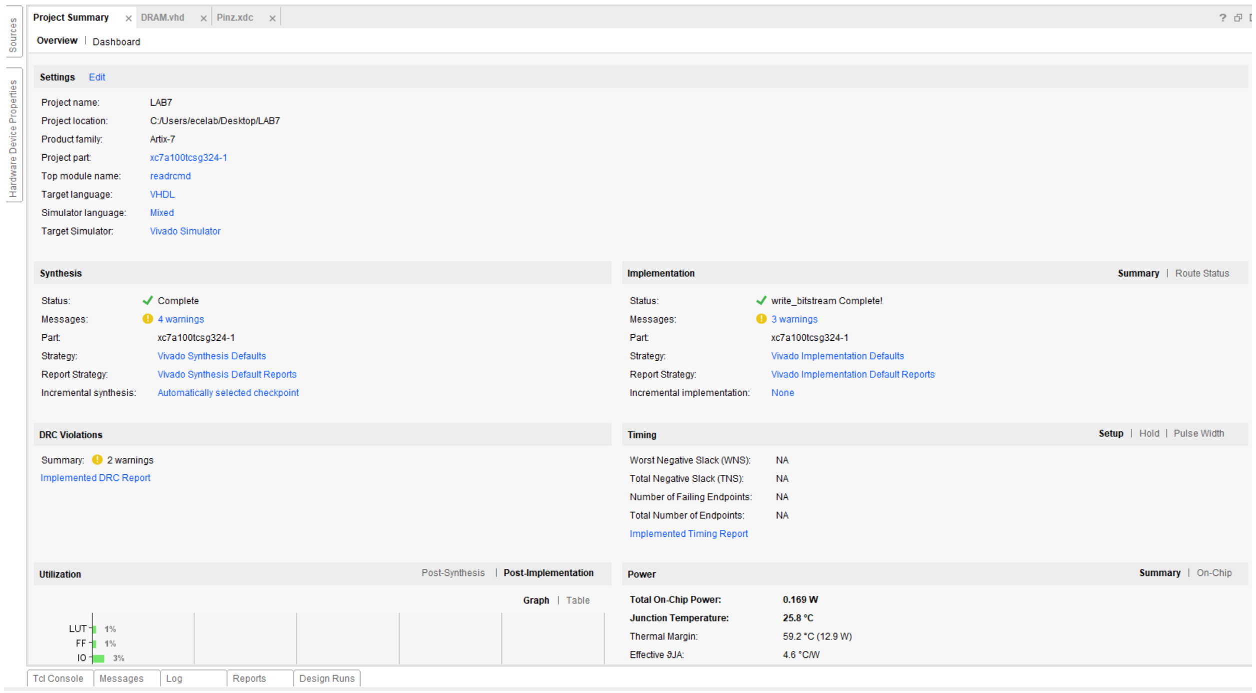


Figure 1: shows the post implementation timing simulation for task 1. the outputs are delayed until 110ns is reached and the rest of the output matches the desired output given in the datasheet.



Fifure 2: the minimum for t_RAS is 70ns and max is 10,000ns, we have 199.176 which is in between the limit.


Project Summary | DRAM.vhd | Pinz.xdc

Overview | Dashboard

Settings Edit

Project name:	LAB7
Project location:	C:/Users/ecelab/Desktop/LAB7
Product family:	Artix-7
Project part:	xc7a100tcsg324-1
Top module name:	readrcmd
Target language:	VHDL
Simulator language:	Mixed
Target Simulator:	Vivado Simulator

Synthesis

Status:	✓ Complete
Messages:	4 warnings
Part:	xc7a100tcsg324-1
Strategy:	Vivado Synthesis Defaults
Report Strategy:	Vivado Synthesis Default Reports
Incremental synthesis:	Automatically selected checkpoint

Implementation — Summary | Route Status

Status:	✓ write_bitstream Complete!
Messages:	3 warnings
Part:	xc7a100tcsg324-1
Strategy:	Vivado Implementation Defaults
Report Strategy:	Vivado Implementation Default Reports
Incremental implementation:	None

DRC Violations

Summary: 2 warnings

Implemented DRC Report

Timing — Setup | Hold | Pulse Width

Worst Negative Slack (WNS):	NA
Total Negative Slack (TNS):	NA
Number of Failing Endpoints:	NA
Total Number of Endpoints:	NA

Implemented Timing Report

Utilization — Post-Synthesis | Post-Implementation — Graph | Table

LUT	1%
FF	1%
IO	3%

Power — Summary | On-Chip

Total On-Chip Power:	**0.169 W**
Junction Temperature:	**25.8 °C**
Thermal Margin:	59.2 °C (12.9 W)
Effective ϑJA:	4.6 °C/W

Tcl Console | Messages | Log | Reports | Design Runs

Fig 3: This is the project summary page for task one.

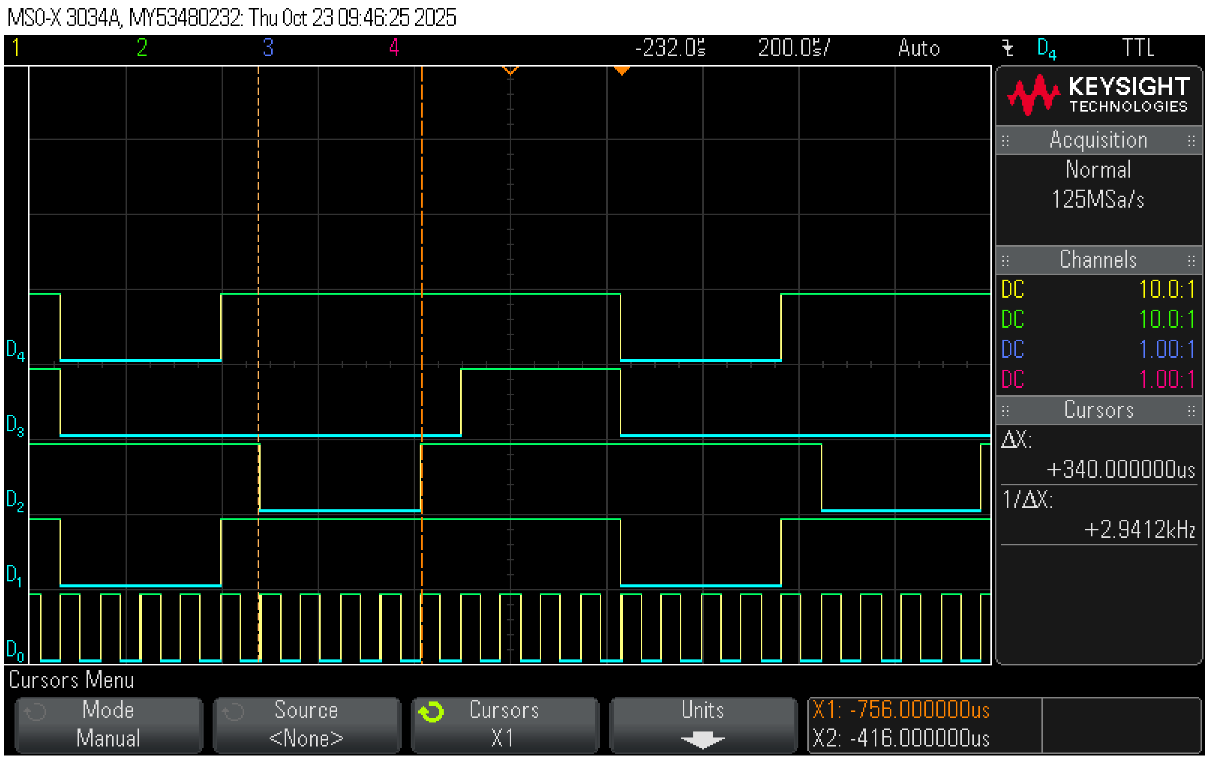


Fig 4: This is the oscilloscope view. D4 is Mux_sel, D3 is CAS, D2 is RAS, D1 is W and D0 is the clk.

MSO-X 3034A, MY53480232: Thu Oct 23 09:44:58 2025

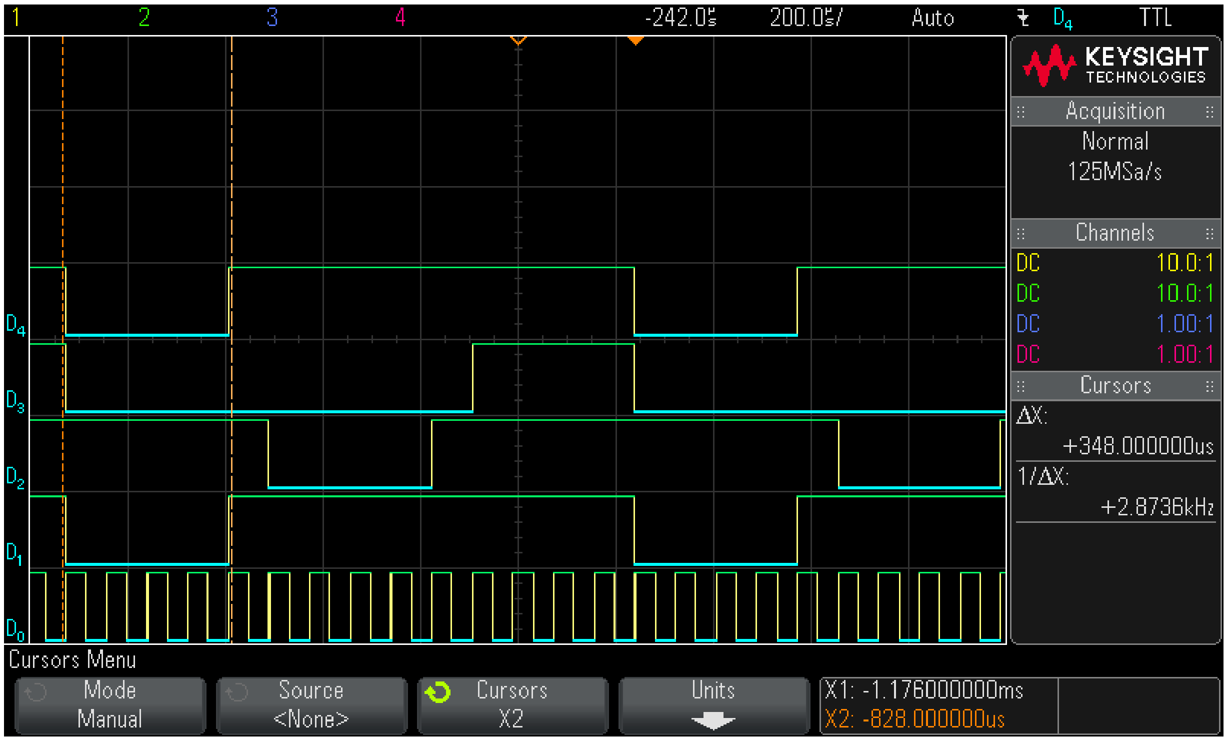


Fig 5: This is the oscilloscope view. D4 is Mux_sel, D3 is CAS, D2 is RAS, D1 is W and D0 is the clk.

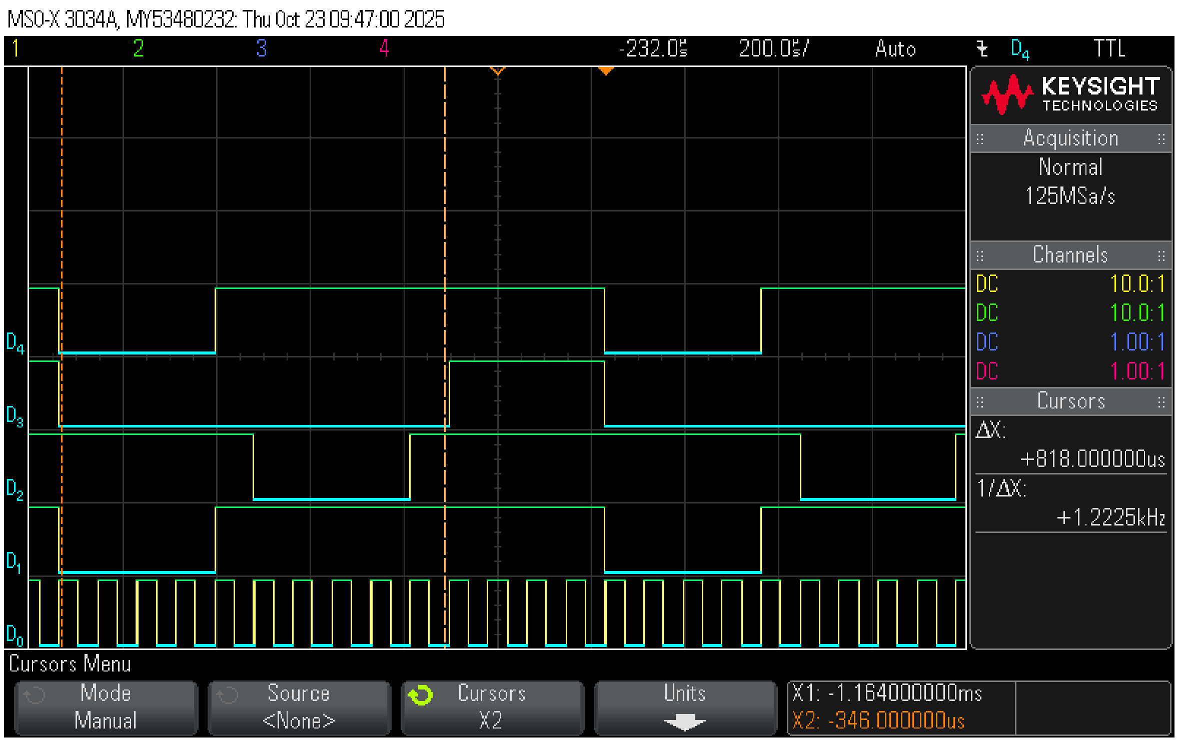


Figure 6: This is the oscilloscope view. D1 is Mux_sel, D2 is CAS, D3 is RAS, D4 is W and D0 is the clk. The outlined timeline above shoves how long RAS stayed low for which as mentioned above, the max limit is 10,000ns

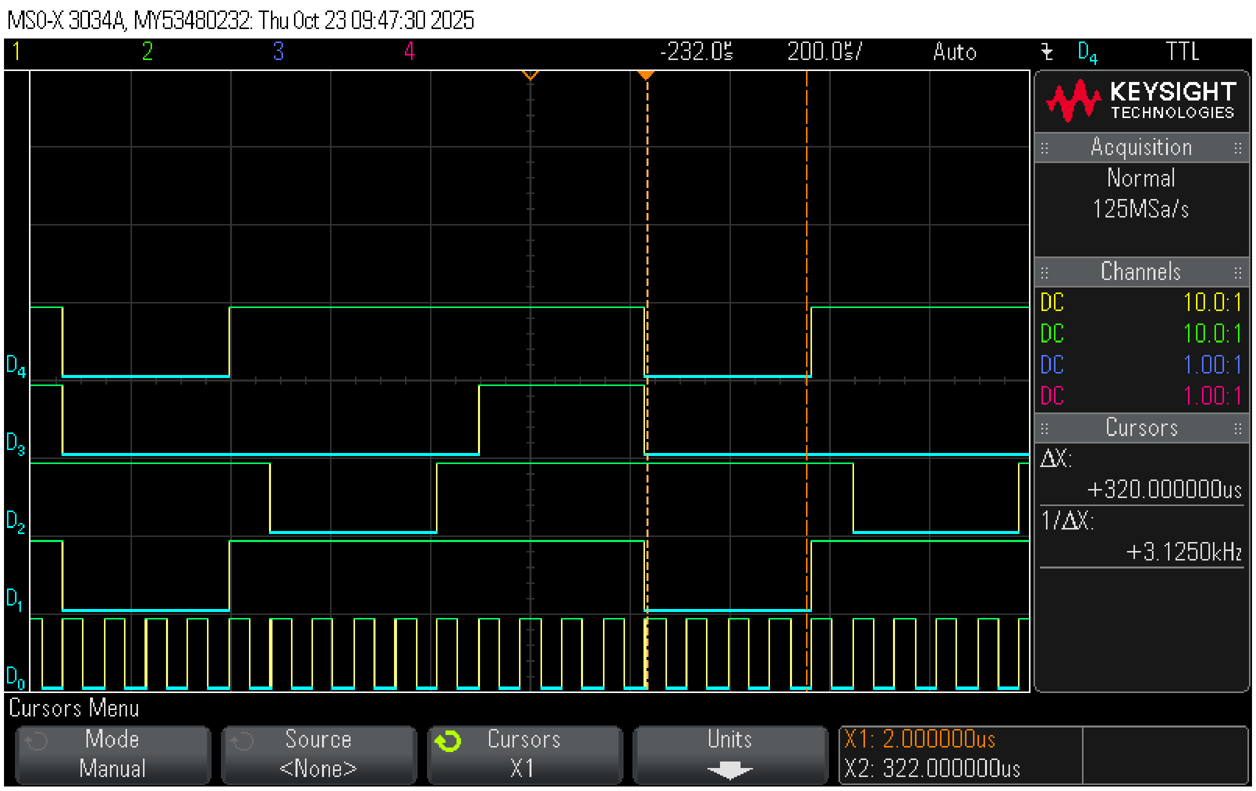


Figure 7: the outlined timeline above displays how long it took for column address to be selected by mux_sel and how long it too for write to be enabled.

Task Two:

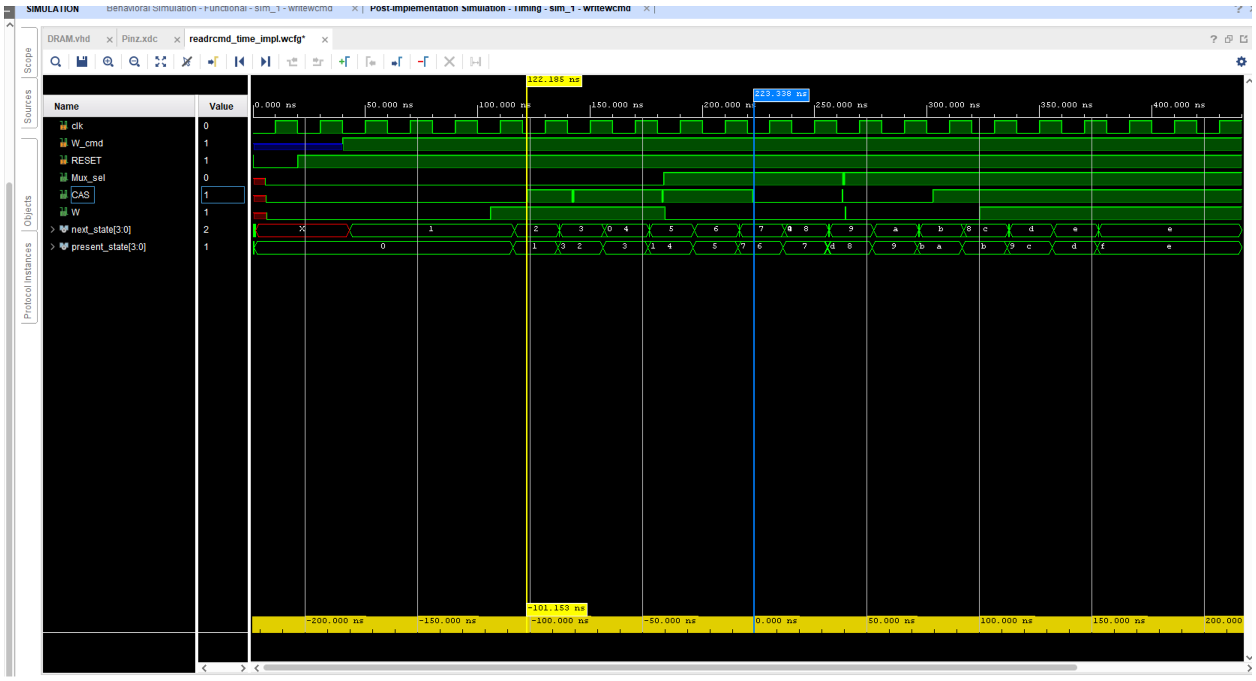
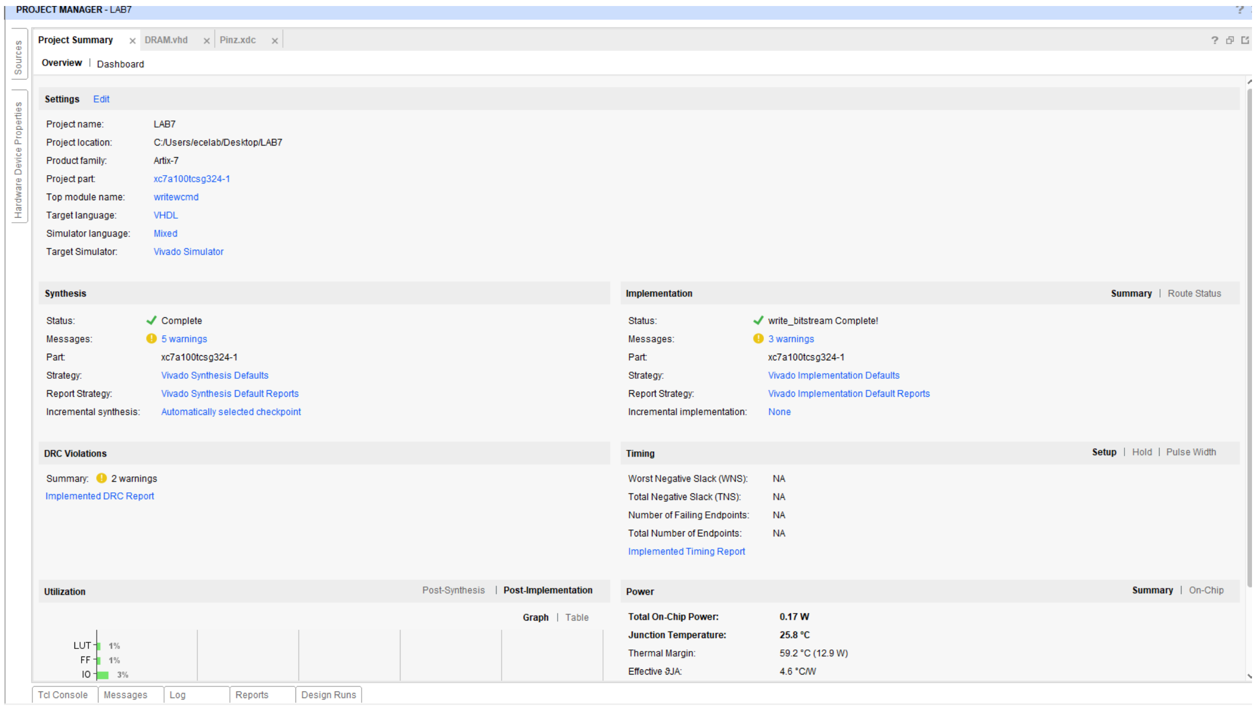


Figure 8: the outlined timeline displayed above shows t_RCD+t_CRP the minimum of which seems to be 25 plus some missing nanoseconds and the maximum is undefined



PROJECT MANAGER - LAB7

Project Summary | DRAM.vhd | Pinz.xdc

Overview | Dashboard

Settings Edit

Project name:	LAB7
Project location:	C:/Users/ecelab/Desktop/LAB7
Product family:	Artix-7
Project part:	xc7a100tcsg324-1
Top module name:	writewcmd
Target language:	VHDL
Simulator language:	Mixed
Target Simulator:	Vivado Simulator

Synthesis

Status:	✓ Complete
Messages:	5 warnings
Part:	xc7a100tcsg324-1
Strategy:	Vivado Synthesis Defaults
Report Strategy:	Vivado Synthesis Default Reports
Incremental synthesis:	Automatically selected checkpoint

Implementation Summary | Route Status

Status:	✓ write_bitstream Complete!
Messages:	3 warnings
Part:	xc7a100tcsg324-1
Strategy:	Vivado Implementation Defaults
Report Strategy:	Vivado Implementation Default Reports
Incremental implementation:	None

DRC Violations

Summary: 2 warnings

Implemented DRC Report

Timing Setup | Hold | Pulse Width

Worst Negative Slack (WNS):	NA
Total Negative Slack (TNS):	NA
Number of Failing Endpoints:	NA
Total Number of Endpoints:	NA

Implemented Timing Report

Utilization Post-Synthesis | Post-Implementation

Graph | Table

LUT	1%
FF	1%
IO	3%

Power Summary | On-Chip

Total On-Chip Power:	0.17 W
Junction Temperature:	25.8 °C
Thermal Margin:	59.2 °C (12.9 W)
Effective θJA:	4.6 °C/W

Tcl Console | Messages | Log | Reports | Design Runs

Fig 9: this is the project summary report for task two.

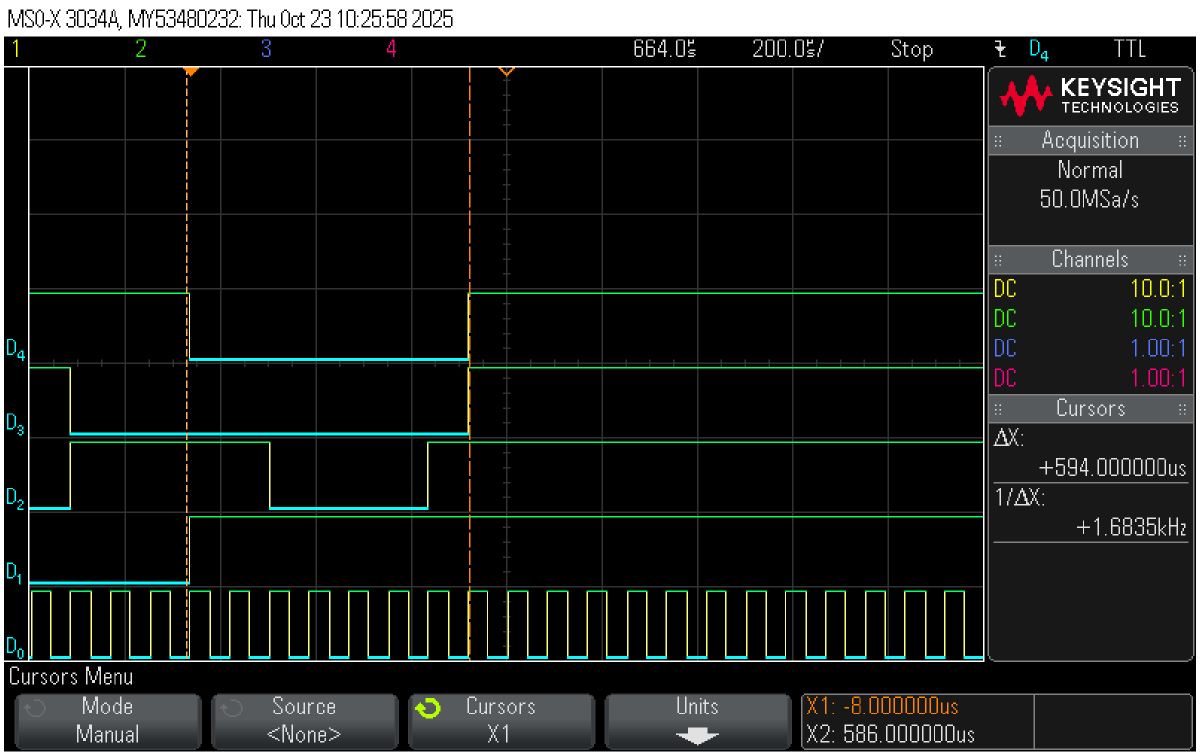


Fig 10: This is the oscilloscope view. D1 is Mux_sel, D2 is CAS, D3 is RAS, D4 is W and D0 is the clk.

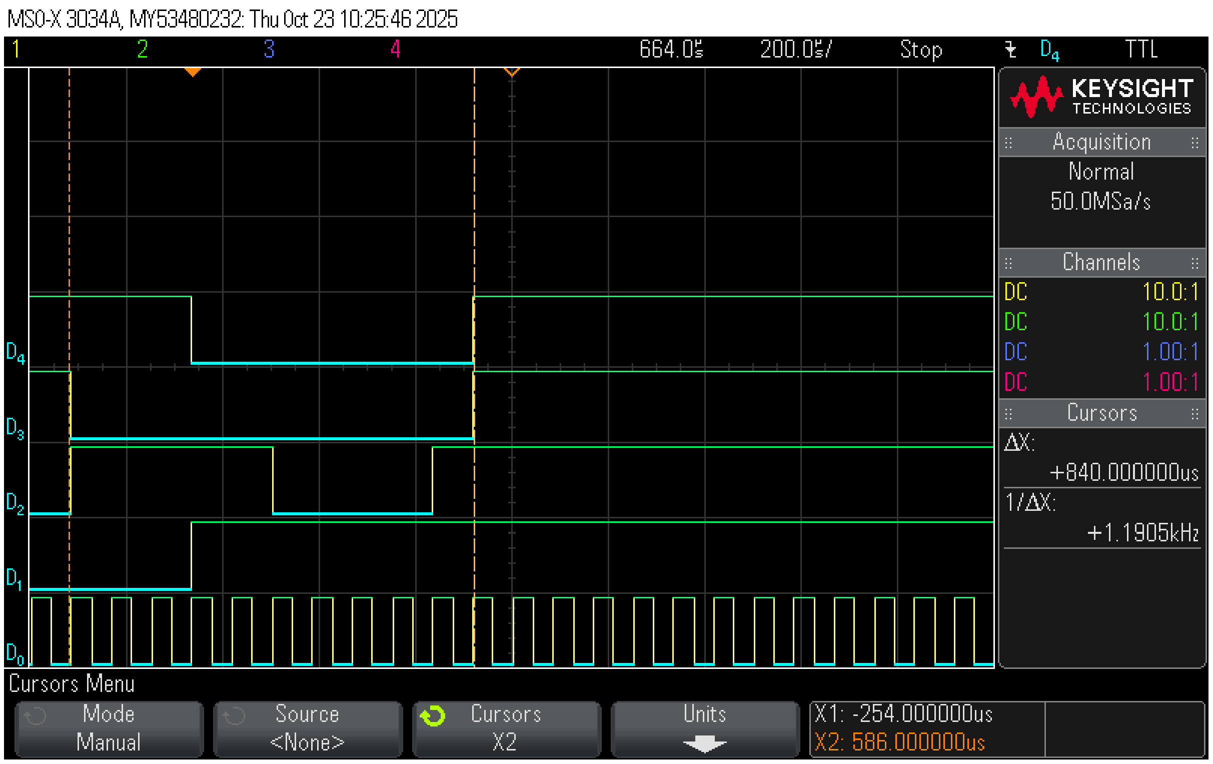


Figure 11: This is the oscilloscope view. D1 is Mux_sel, D2 is CAS, D3 is RAS, D4 is W and D0 is the clk.

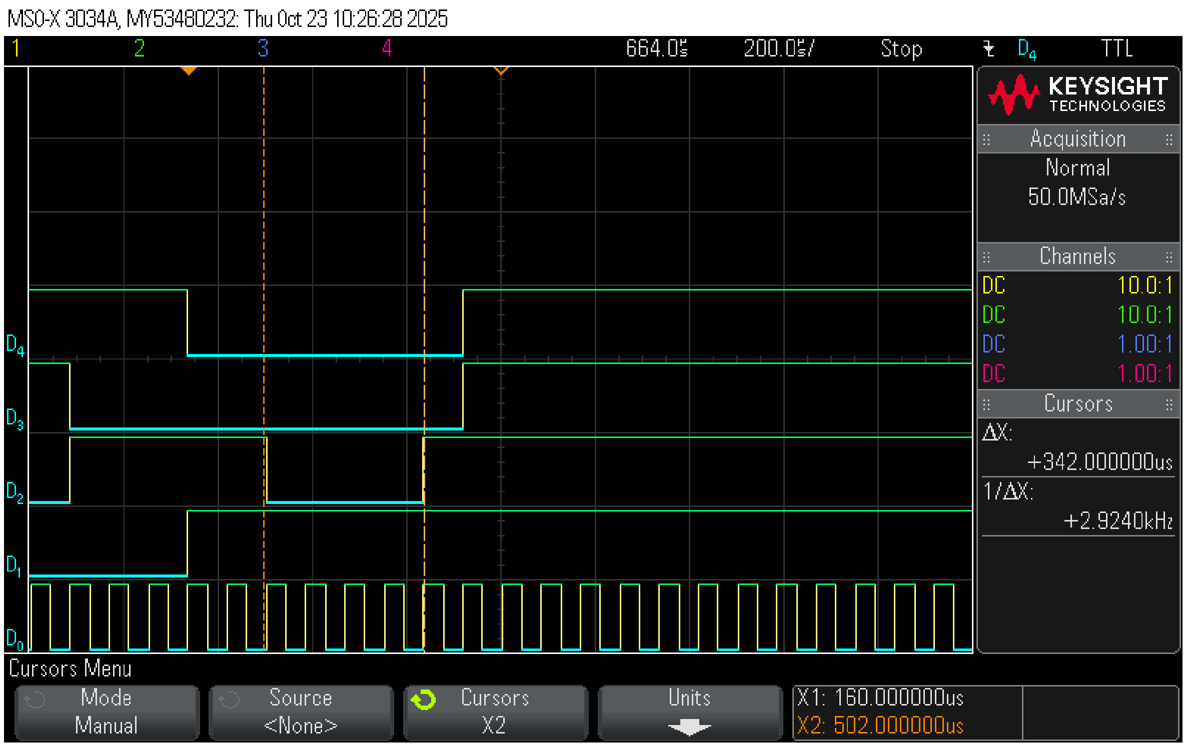


Figure 12: This is the oscilloscope view. D1 is Mux_sel, D2 is CAS, D3 is RAS, D4 is W and D0 is the clk.

Task Three:

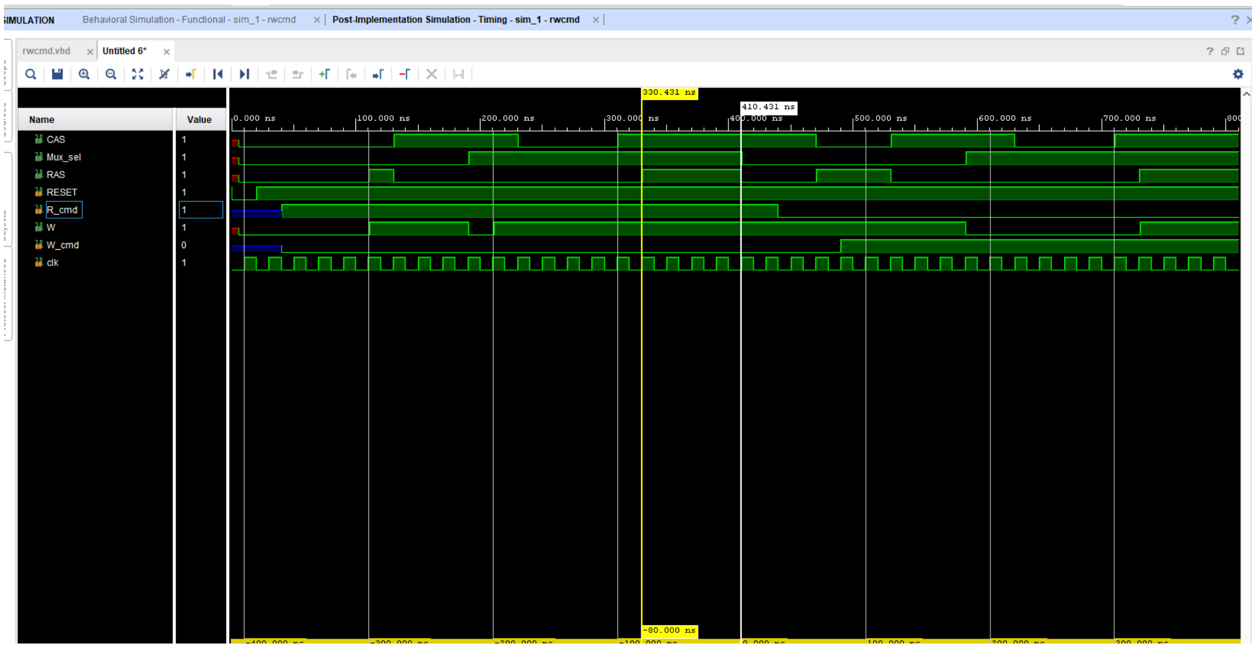
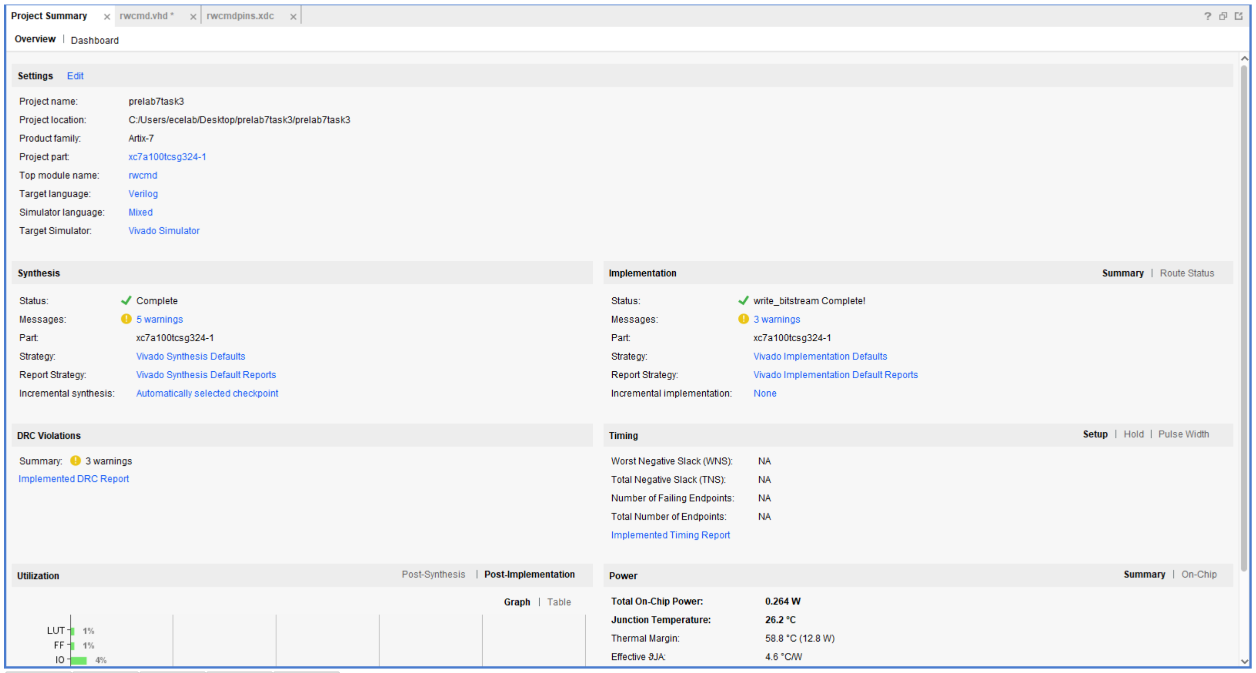


Figure 13: the post implementation timing simulation above has the t_rp outlined for which the minimum is 50ns and max is undefined. We have 80ns.


Project Summary | rwcmd.vhd * | rwcmdpins.xdc

Overview | Dashboard

Settings Edit

Project name:	prelab7task3
Project location:	C:/Users/ecelab/Desktop/prelab7task3/prelab7task3
Product family:	Artix-7
Project part:	xc7a100tcsg324-1
Top module name:	rwcmd
Target language:	Verilog
Simulator language:	Mixed
Target Simulator:	Vivado Simulator

Synthesis

Status:	✓ Complete
Messages:	5 warnings
Part:	xc7a100tcsg324-1
Strategy:	Vivado Synthesis Defaults
Report Strategy:	Vivado Synthesis Default Reports
Incremental synthesis:	Automatically selected checkpoint

Implementation Summary | Route Status

Status:	✓ write_bitstream Complete!
Messages:	3 warnings
Part:	xc7a100tcsg324-1
Strategy:	Vivado Implementation Defaults
Report Strategy:	Vivado Implementation Default Reports
Incremental implementation:	None

DRC Violations

Summary: 3 warnings

Implemented DRC Report

Timing Setup | Hold | Pulse Width

Worst Negative Slack (WNS):	NA
Total Negative Slack (TNS):	NA
Number of Failing Endpoints:	NA
Total Number of Endpoints:	NA

Implemented Timing Report

Utilization Post-Synthesis | Post-Implementation

Graph | Table

LUT	1%
FF	1%
IO	4%

Power Summary | On-Chip

Total On-Chip Power:	0.264 W
Junction Temperature:	26.2 °C
Thermal Margin:	58.8 °C (12.8 W)
Effective θJA:	4.6 °C/W

Fig 14: this shows the project summary report for task three.

MSO-X 3034A, MY53480232: Thu Oct 23 10:50:12 2025

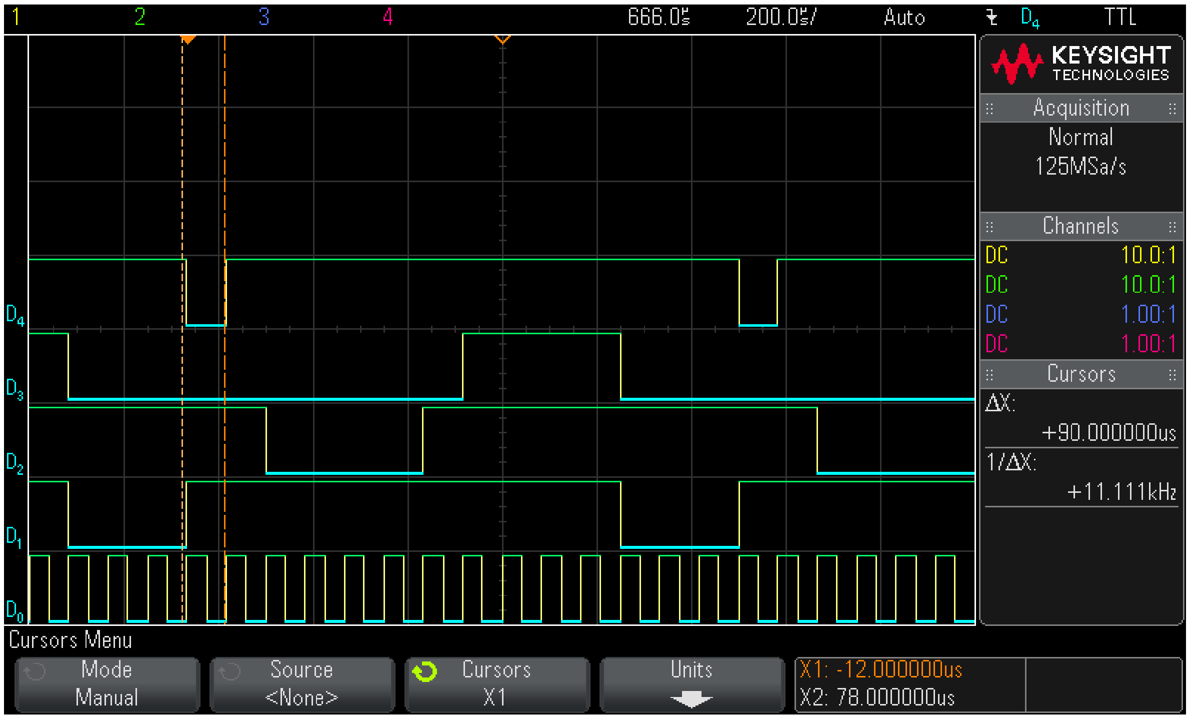
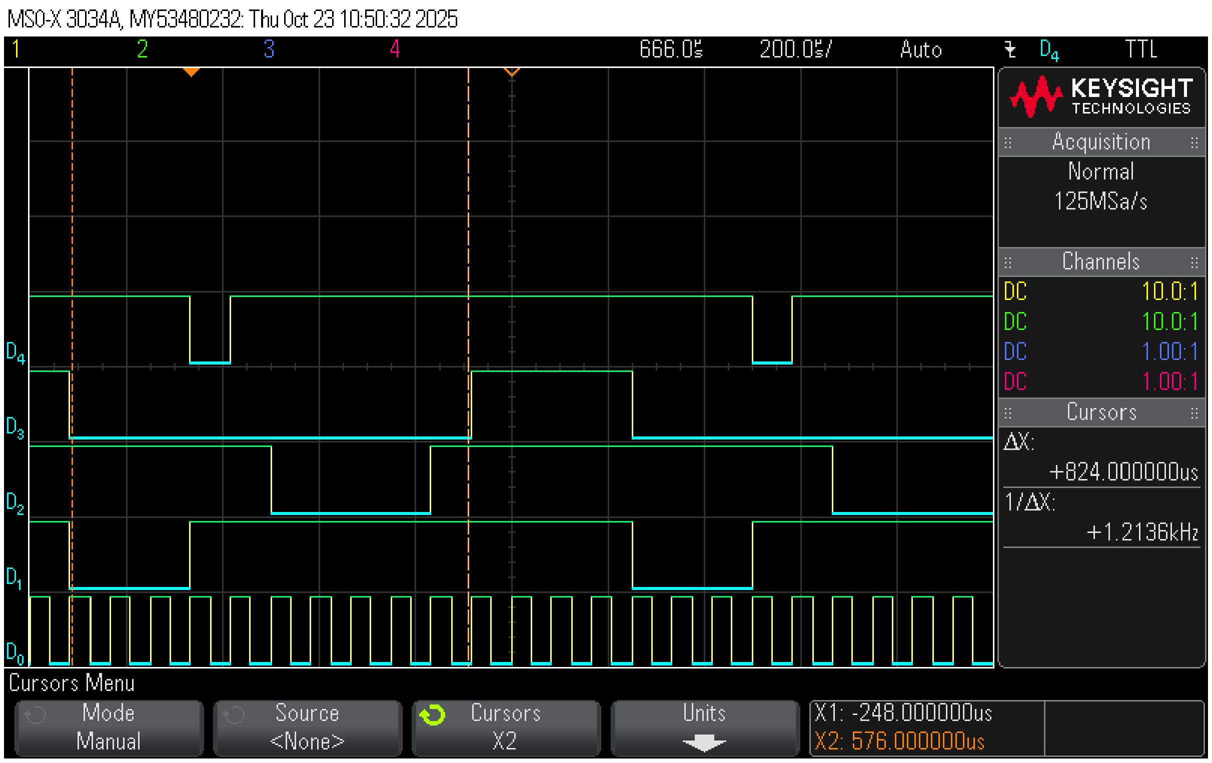


Fig 15: This is the oscilloscope view. D1 is Mux_sel, D2 is CAS, D3 is RAS, D4 is W and D0 is the clk.


MSO-X 3034A, MY53480232: Thu Oct 23 10:50:32 2025

Acquisition
Normal
125MSa/s

Channels

DC	10.0:1
DC	10.0:1
DC	1.00:1
DC	1.00:1

Cursors
ΔX:
+824.000000us
1/ΔX:
+1.2136kHz

Cursors Menu

Mode
Manual
Source
<None>
Cursors
X2
Units

X1: -248.000000us
X2: 576.000000us

Figure 16: This is the oscilloscope view. D4 is Mux_sel, D3 is CAS, D2 is RAS, D1 is W and D0 is the clk.

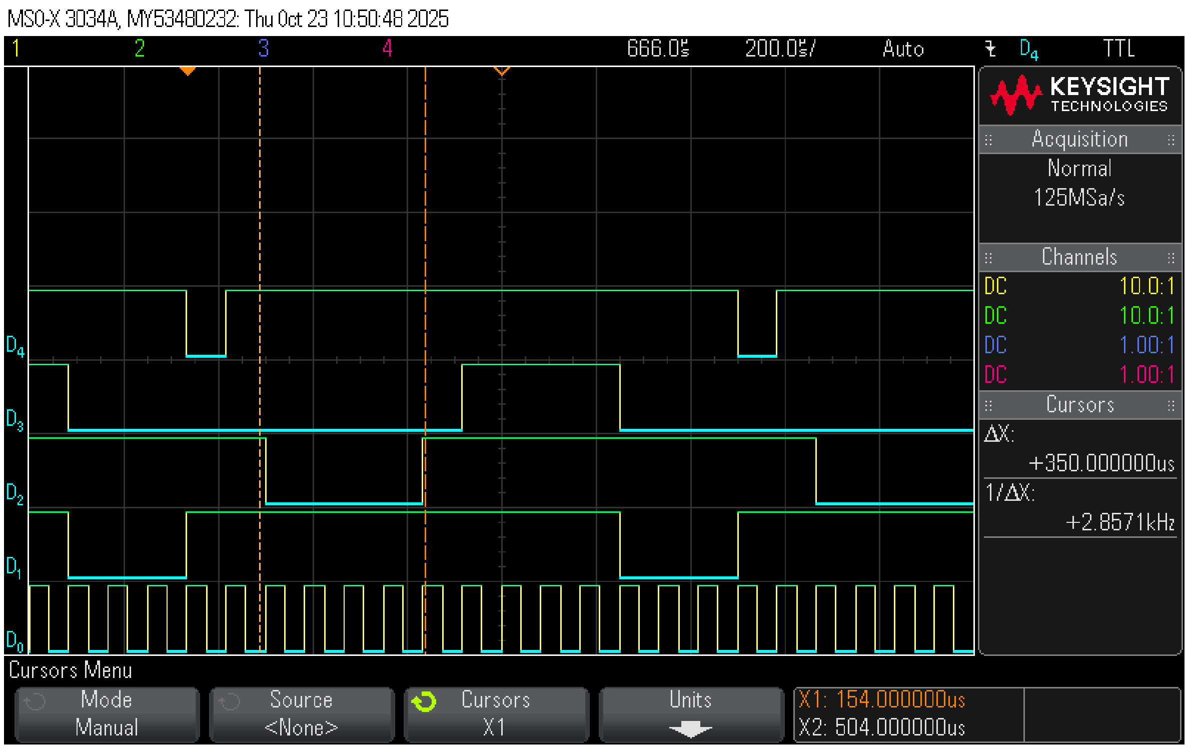


Figure 17: This is the oscilloscope view. D1 is Mux_sel, D2 is CAS, D3 is RAS, D4 is W and D0 is the clk.

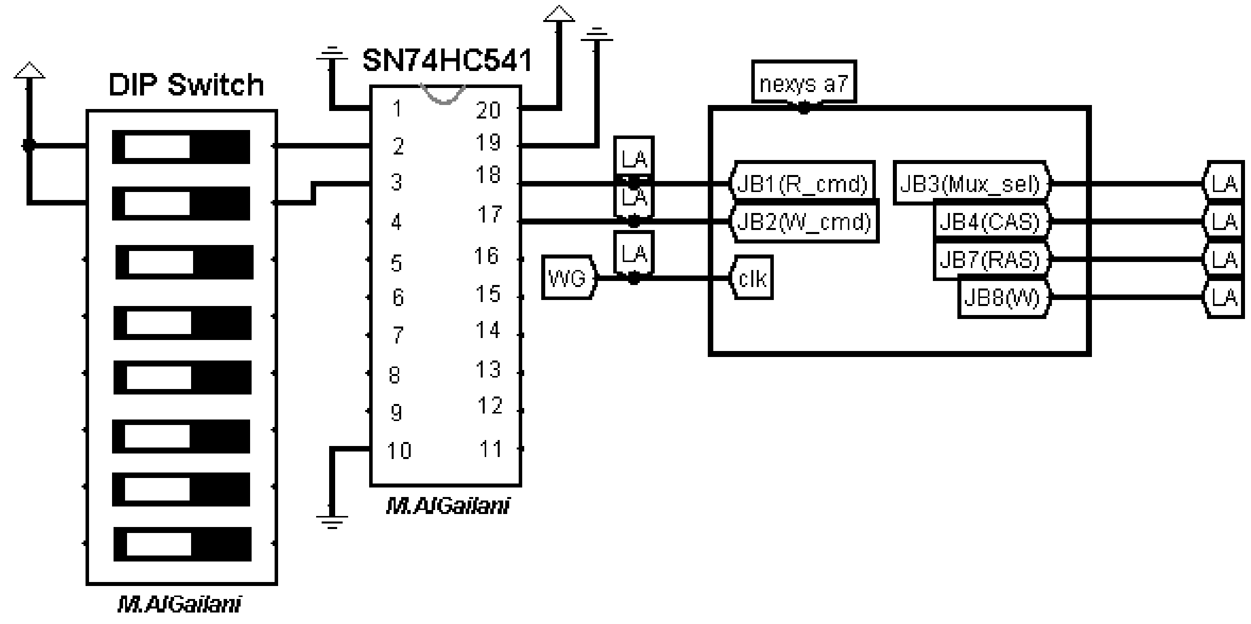


Figure 18: shows the schematic for task 3, there was no need for a schematic in lab report therefore we volunterely provided it for task3 only where la stands for logic analyzer and wg for wave generator.

Conclusion

In the lab, the design and implementation of a DRAM controller provided practical knowledge of synchronous sequential circuit design and finite state machines. Task one involved creating a VHDL code for a simple read cycle for the DRAM chip. For task two, we implemented a write cycle for the DRAM chip. In task three, both read and write operations were integrated into a single DRAM controller. Simulations and the oscilloscope verified that the state machine successfully generated the proper signal transitions for each task. In conclusion, this lab helped reinforce key skills in digital design and memory interface development.

Appendix

Task One

Vhd

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity readrcmd is
  Port (
  clk, R_cmd, RESET: in std_logic;
  Mux_sel, CAS, RAS, W: out std_logic
  );
end readrcmd;

architecture Behavioral of readrcmd is
type state is (idle,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14);
signal present_state, next_state: state:= idle;
begin
process(R_cmd, present_state)
begin
case present_state is
when idle =>
CAS <= '0';
RAS <= '1';
W <= '0';
Mux_sel <= '0';
if R_cmd = '1' then
    next_state <= s1;
else
    next_state <= idle;
end if;

when s1 =>
CAS <= '1';
RAS <= '0';
W <= '0';
Mux_sel <= '0';
next_state <= s2;

when s2 =>
CAS <= '1';
RAS <= '0';
W <= '0';
Mux_sel <= '0';
next_state <= s3;

when s3 =>
CAS <= '1';
RAS <= '0';
W <= '0';
Mux_sel <= '0';
next_state <= s4;

when s4 =>
```

```
Mux_sel <= '0';
CAS <= '1';
RAS <= '0';
W <= '0';
next_state <= s5;

when s5 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '0';
W <= '1';
next_state <= s6;

when s6 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '1';
next_state <= s7;

when s7 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '1';
next_state <= s8;

when s8 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '1';
next_state <= s9;

when s9 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '1';
next_state <= s10;

when s10 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '0';
W <= '1';
next_state <= s11;

when s11 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
next_state <= s12;

when s12 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
next_state <= s13;

when s13 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
next_state <= s14;

when s14 =>
Mux_sel <= '1';
```

```
CAS <= '1';
RAS <= '1';
W <= '1';
if R_cmd = '1' then
    next_state <= s1;
else
    next_state <= idle;
end if;

when others =>
    next_state <= idle;

end case;
end process;

process
begin
wait until clk'event and clk = '1';
     if reset <= '0' then
        present_state <= idle;
     else
        present_state <= next_state;
     end if;
end process;

end Behavioral;
```

Tcl

```
restart
add_force clk {0 0ns} {1 10ns} -repeat_every 20ns

add_force RESET {0 0ns}
run 20ns
add_force RESET {1 0ns}
run 20ns
add_force R_cmd {1 0ns}
run 300ns
```

Xdc

```
## Clock signal
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk]

#set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports { CLK100MHZ }];
#IO_L12P_T1_MRCC_35 Sch=clk100mhz

create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}];

##Buttons
set_property -dict { PACKAGE_PIN C12 IOSTANDARD LVCMOS33 } [get_ports { RESET }];
#IO_L3P_T0_DQS_AD1P_15 Sch=cpu_resetn

##Pmod Header JB
set_property -dict { PACKAGE_PIN D14 IOSTANDARD LVCMOS33 } [get_ports { R_cmd }];
#IO_L1P_T0_AD0P_15 Sch=jb[1]

#set_property -dict { PACKAGE_PIN F16 IOSTANDARD LVCMOS33 } [get_ports { JB[2] }];
#IO_L14N_T2_SRCC_15 Sch=jb[2]

set_property -dict { PACKAGE_PIN G16 IOSTANDARD LVCMOS33 } [get_ports { Mux_sel }];
#IO_L13N_T2_MRCC_15 Sch=jb[3]

set_property -dict { PACKAGE_PIN H14 IOSTANDARD LVCMOS33 } [get_ports { CAS }];
#IO_L15P_T2_DQS_15 Sch=jb[4]
```

```
set_property -dict { PACKAGE_PIN E16 IOSTANDARD LVCMOS33 } [get_ports { RAS }];
#IO_L11N_T1_SRCC_15 Sch=jb[7]

set_property -dict { PACKAGE_PIN F13 IOSTANDARD LVCMOS33 } [get_ports { W }];
#IO_L5P_T0_AD9P_15 Sch=jb[8]

set_property -dict { PACKAGE_PIN H16 IOSTANDARD LVCMOS33 } [get_ports { clk }];
#IO_L13P_T2_MRCC_15 Sch=jb[10]
```

Task Two

Vhd

```


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity readrcmd is
  Port (
  clk, R_cmd, RESET: in std_logic;
  Mux_sel, CAS, RAS, W: out std_logic
  );
end readrcmd;

architecture Behavioral of readrcmd is
type state is (idle,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14);
signal present_state, next_state: state:= idle;
begin
process(R_cmd, present_state)
begin
case present_state is
when idle =>
CAS <= '0';
RAS <= '1';
W <= '0';
Mux_sel <= '0';
if R_cmd = '1' then
    next_state <= s1;
else
    next_state <= idle;
end if;

when s1 =>
CAS <= '1';
RAS <= '0';
W <= '0';
Mux_sel <= '0';
next_state <= s2;

when s2 =>
CAS <= '1';
RAS <= '0';
W <= '0';
Mux_sel <= '0';
next_state <= s3;

when s3 =>
CAS <= '1';
```

```
RAS <= '0';
W <= '0';
Mux_sel <= '0';
next_state <= s4;

when s4 =>
Mux_sel <= '0';
CAS <= '1';
RAS <= '0';
W <= '0';
next_state <= s5;

when s5 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '0';
W <= '1';
next_state <= s6;

when s6 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '1';
next_state <= s7;

when s7 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '1';
next_state <= s8;

when s8 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '1';
next_state <= s9;

when s9 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '1';
next_state <= s10;

when s10 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '0';
W <= '1';
next_state <= s11;

when s11 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
next_state <= s12;

when s12 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
next_state <= s13;

when s13 =>
Mux_sel <= '1';
CAS <= '1';
```

```
RAS <= '1';
W <= '1';
next_state <= s14;

when s14 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
if R_cmd = '1' then
    next_state <= s1;
else
    next_state <= idle;
end if;

when others =>
    next_state <= idle;

end case;
end process;

process
begin
wait until clk'event and clk = '1';
     if reset <= '0' then
        present_state <= idle;
     else
        present_state <= next_state;
     end if;
end process;

end Behavioral;
```

Tcl

```
restart
add_force clk {0 0ns} {1 10ns} -repeat_every 20ns

add_force RESET {0 0ns}
run 20ns
add_force RESET {1 0ns}
run 20ns
add_force W_cmd {1 0ns}
run 300ns
```

Xdc

```
## Clock signal
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk]

#set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports { CLK100MHZ }];
#IO_L12P_T1_MRCC_35 Sch=clk100mhz

create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}];

##Buttons
set_property -dict { PACKAGE_PIN C12 IOSTANDARD LVCMOS33 } [get_ports { RESET }];
#IO_L3P_T0_DQS_AD1P_15 Sch=cpu_resetn

##Pmod Header JB
#set_property -dict { PACKAGE_PIN D14 IOSTANDARD LVCMOS33 } [get_ports { JB[1] }];
#IO_L1P_T0_AD0P_15 Sch=jb[1]

set_property -dict { PACKAGE_PIN F16 IOSTANDARD LVCMOS33 } [get_ports { W_cmd }];
#IO_L14N_T2_SRCC_15 Sch=jb[2]
```

```
set_property -dict { PACKAGE_PIN G16 IOSTANDARD LVCMOS33 } [get_ports { Mux_sel }];
#IO_L13N_T2_MRCC_15 Sch=jb[3]

set_property -dict { PACKAGE_PIN H14 IOSTANDARD LVCMOS33 } [get_ports { CAS }];
#IO_L15P_T2_DQS_15 Sch=jb[4]

set_property -dict { PACKAGE_PIN E16 IOSTANDARD LVCMOS33 } [get_ports { RAS }];
#IO_L11N_T1_SRCC_15 Sch=jb[7]

set_property -dict { PACKAGE_PIN F13 IOSTANDARD LVCMOS33 } [get_ports { W }];
#IO_L5P_T0_AD9P_15 Sch=jb[8]

set_property -dict { PACKAGE_PIN H16 IOSTANDARD LVCMOS33 } [get_ports { clk }];
#IO_L13P_T2_MRCC_15 Sch=jb[10]
```

Task Three

vhd

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity rwcmd is
  Port (
  R_cmd, W_cmd, RESET, clk: in std_logic;
  Mux_sel, CAS, RAS, W: out std_logic
   );
end rwcmd;

architecture Behavioral of rwcmd is
type state is (s1, s2, s3, s4, s5, s6, s7, s8, s9, s10, s11, s12, s13, s14, s15, s16, s17);
signal present_state, next_state: state:=s1;
begin
process(R_cmd, W_cmd, present_state)
begin
case present_state is

when s1 =>
Mux_sel <= '0';
CAS <= '0';
RAS <= '1';
W <= '1';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s2;
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s2;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s2 =>
Mux_sel <= '0';
CAS <= '1';
RAS <= '0';
W <= '1';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s3;
```

```
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s3;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s3 =>
Mux_sel <= '0';
CAS <= '1';
RAS <= '0';
W <= '1';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s4;
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s4;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s4 =>
Mux_sel <= '0';
CAS <= '1';
RAS <= '0';
W <= '1';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s5;
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s5;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s5 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '0';
W <= '0';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s6;
    W <= '0';
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s6;
    W <= '0';
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s6 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '0';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s7;
    W <= '1';
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s7;
    W <= '0';
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;
```

```
when s7 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s8;
    W <= '1';
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s8;
    W <= '0';
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s8 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '0';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s9;
    W <= '1';
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s9;
    W <= '0';
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s9 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '0';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s10;
    W <= '1';
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s10;
    W <= '0';
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s10 =>
Mux_sel <= '1';
CAS <= '0';
RAS <= '0';
W <= '0';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s11;
    W <= '1';
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s11;
    W <= '0';
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s11 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '0';
```

```vhdl
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s12;
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s12;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s12 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s13;
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s13;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s13 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s14;
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s14;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s14 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s15;
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s15;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when s15 =>
Mux_sel <= '1';
CAS <= '1';
RAS <= '1';
W <= '1';
if R_cmd = '1' and W_cmd = '0' then
    next_state <= s2;
elsif R_cmd = '0' and W_cmd = '1' then
    next_state <= s2;
elsif R_cmd = '0' and W_cmd = '0' then
    next_state <= s1;
else
    next_state <= s2;
end if;

when others =>
```

```
    next_state <= s1;

end case;
end process;

process
begin
wait until clk'event and clk = '1';
if reset = '0' then
    present_state <= s1;
else
    present_state <= next_state;
end if;
end process;


end Behavioral;
```

tcl

```
restart
add_force clk {0 0ns} {1 10ns} -repeat_every 20ns

add_force RESET {0 0ns}
run 20ns
add_force RESET {1 0ns}
run 20ns
add_force R_cmd {1 0ns}
add_force W_cmd {1 0ns}
run 160ns
add_force R_cmd {0 0ns}
run 50ns
add_force W_cmd {0 0ns}
run 20ns
```

xdc

```
## Clock signal
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets clk]
set_property -dict { PACKAGE_PIN E3 IOSTANDARD LVCMOS33 } [get_ports { CLK100MHZ }];
#IO_L12P_T1_MRCC_35 Sch=clk100mhz

create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}];

##Buttons
set_property -dict { PACKAGE_PIN C12 IOSTANDARD LVCMOS33 } [get_ports { RESET }];
#IO_L3P_T0_DQS_AD1P_15 Sch=cpu_resetn

##Pmod Headers
##Pmod Header JB
set_property -dict { PACKAGE_PIN D14 IOSTANDARD LVCMOS33 } [get_ports { R_cmd }];
#IO_L1P_T0_AD0P_15 Sch=jb[1]

set_property -dict { PACKAGE_PIN F16 IOSTANDARD LVCMOS33 } [get_ports { W_cmd }];
#IO_L14N_T2_SRCC_15 Sch=jb[2]

set_property -dict { PACKAGE_PIN G16 IOSTANDARD LVCMOS33 } [get_ports { Mux_sel }];
#IO_L13N_T2_MRCC_15 Sch=jb[3]

set_property -dict { PACKAGE_PIN H14 IOSTANDARD LVCMOS33 } [get_ports { CAS }];
#IO_L15P_T2_DQS_15 Sch=jb[4]

set_property -dict { PACKAGE_PIN E16 IOSTANDARD LVCMOS33 } [get_ports { RAS }];
#IO_L11N_T1_SRCC_15 Sch=jb[7]
```

```
set_property -dict { PACKAGE_PIN F13 IOSTANDARD LVCMOS33 } [get_ports { W }];
#IO_L5P_T0_AD9P_15 Sch=jb[8]

set_property -dict { PACKAGE_PIN H16 IOSTANDARD LVCMOS33 } [get_ports { clk }];
#IO_L13P_T2_MRCC_15 Sch=jb[10]
```