Report of Lab 2.1

Lalit Meena,2019CS50439

For this Lab 2 we are building microarchitecture for subset of arm instructions.

For this aim we are first building basic modules such as ALU, Register File 16x32, Program Memory and Data Memory 64x32.They are tested thoroughly with separate testbench files.

I have made 4 module design files , 4 testbench file for each module and two helping files.

Zip folder name – L2.1_2019CS50439

<u>Submission folder, **L2.1_2019CS50439** contains-</u>

1. **alu.vhd –** alu module design file
2. **data_mem.vhd -** data memory 64x32 module design file
3. **prog_mem.vhd –** program memory 64x32 module design file
4. **rfile.vhd -** design file for module register file 16x32

package.vhd -package to declare some types-words,bytes,etc

run.do- edaplayground file to help in syntheses of modules to give useful info about resources

<u>4 testbench files-</u>

testbench_alu.vhd – testbench file for alu module

testbench_dm.vhd – testbench file for data memory 64x32 module

testbench_pm.vhd –– testbench file for program memory 64x32 module

testbench_rf.vhd - testbench file for module register file 16x32

Report of Lab 2.1-lab 2 stage 1 description and test cases(also screenshots)

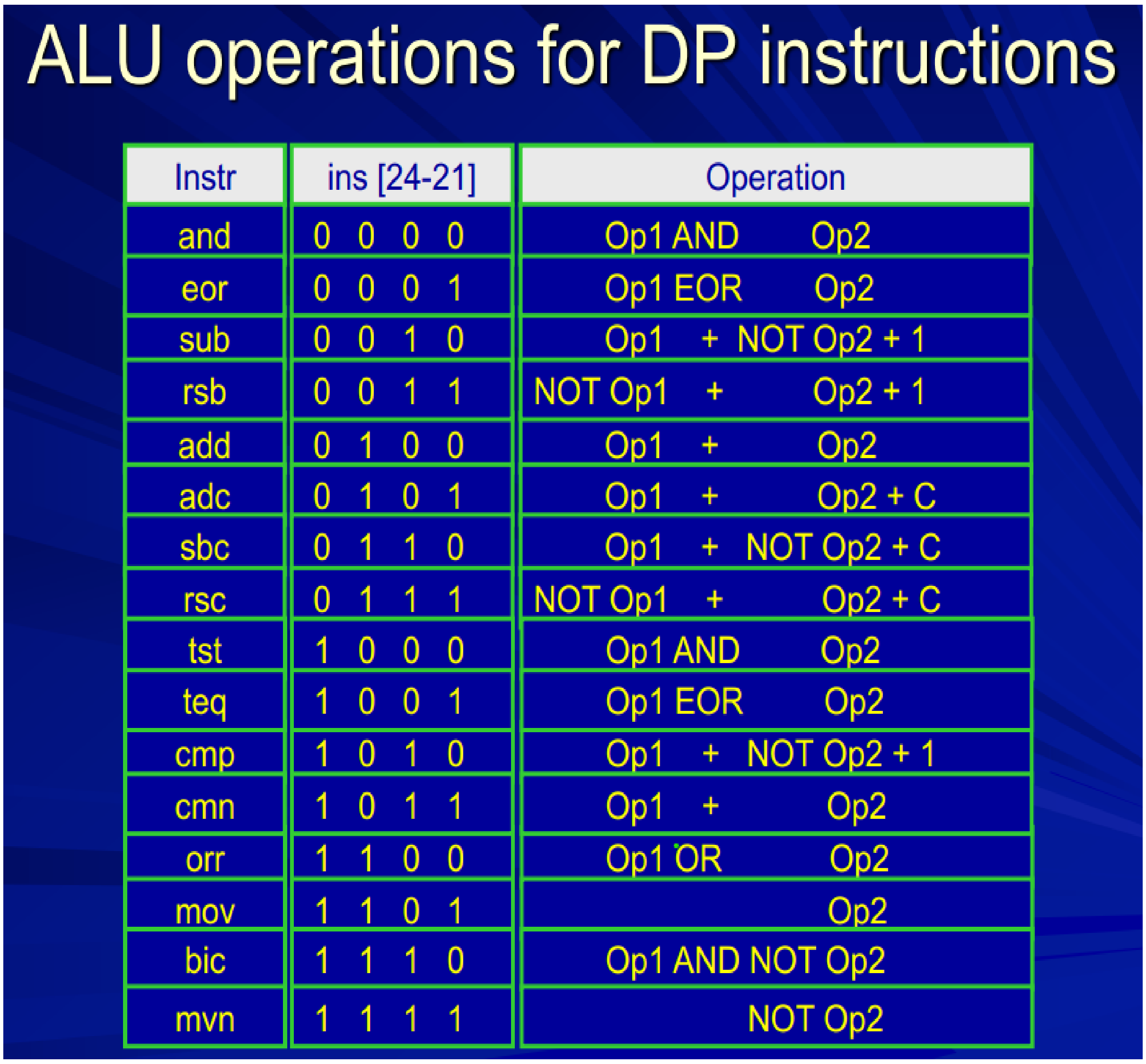
More details are in next pages of each module-

1. alu.vhd – alu module design file

explanation-

alu module is combinational circuit that takes two operands, does one of the 16 operations and produces a result. Operands and results are 32-bit std_logic_vectors. there is also a carry input and a carry output. operation to be done is specified by another input that modelled as a 4-bit std_logic_vector.

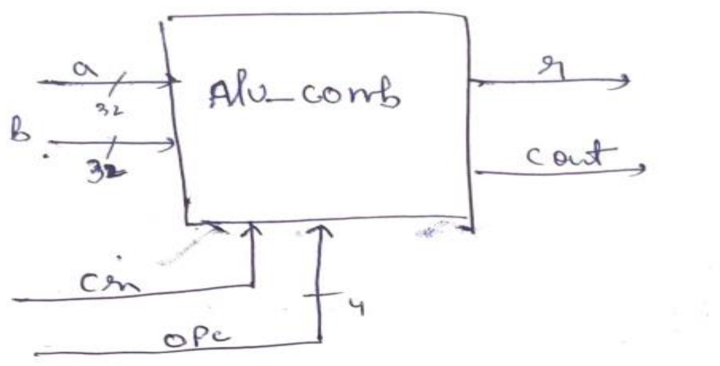
For **implementation** used simple case statement and then for arithmetic instructions ,used full adder to perform computation as shown below.



ALU operations for DP instructions

Instr	ins [24-21]	Operation
and	0 0 0 0	Op1 AND Op2
eor	0 0 0 1	Op1 EOR Op2
sub	0 0 1 0	Op1 + NOT Op2 + 1
rsb	0 0 1 1	NOT Op1 + Op2 + 1
add	0 1 0 0	Op1 + Op2
adc	0 1 0 1	Op1 + Op2 + C
sbc	0 1 1 0	Op1 + NOT Op2 + C
rsc	0 1 1 1	NOT Op1 + Op2 + C
tst	1 0 0 0	Op1 AND Op2
teq	1 0 0 1	Op1 EOR Op2
cmp	1 0 1 0	Op1 + NOT Op2 + 1
cmn	1 0 1 1	Op1 + Op2
orr	1 1 0 0	Op1 OR Op2
mov	1 1 0 1	Op2
bic	1 1 1 0	Op1 AND NOT Op2
mvn	1 1 1 1	NOT Op2

diagram-



entity and ports info-

```
-- entity
entity alu_comb is
port(
  a: in word ;
  b: in word;
  opc:in optype;
  cin:in std_logic;
  r: out word;
  cout:out std_logic);
end entity;
```

synthesized resources-

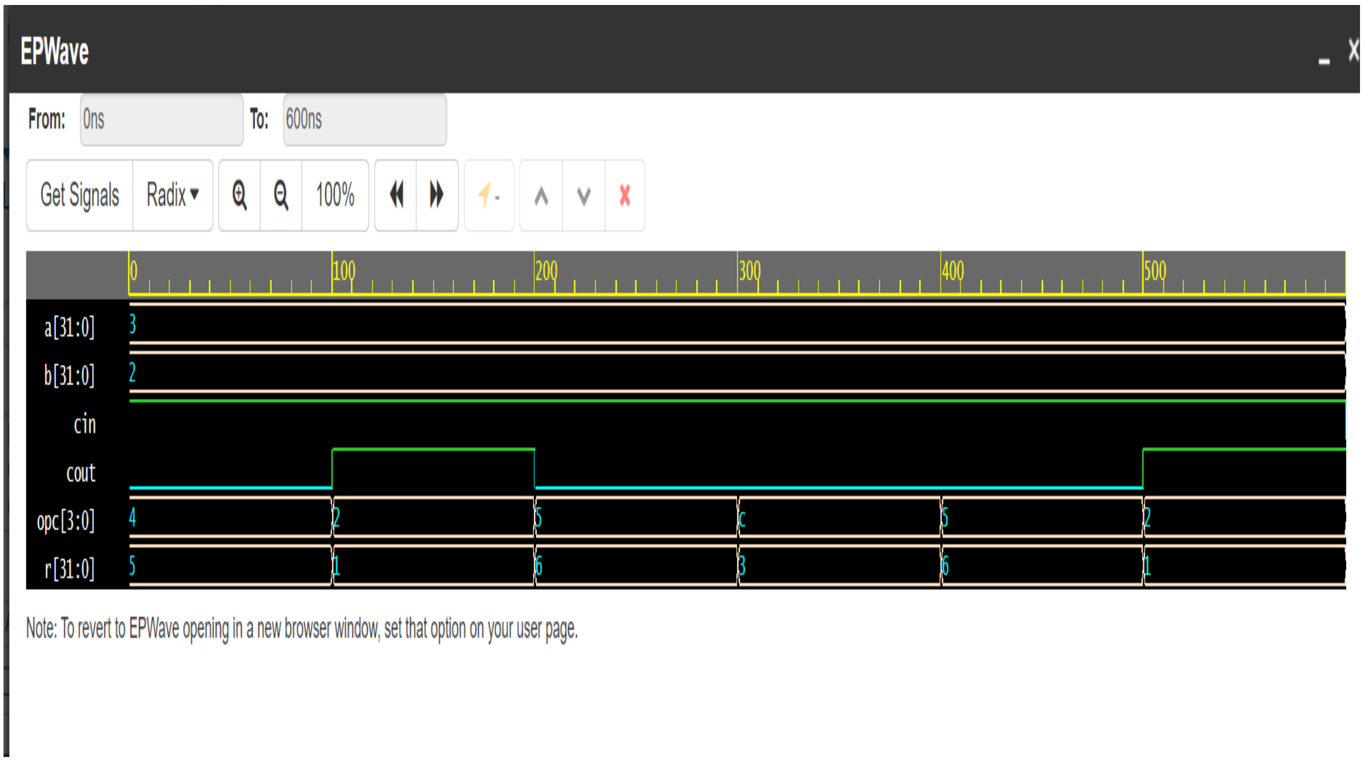
```
# Info: ***************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ***************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ---------------------------------------------------------------
# Info: IOs                               102     210      48.57%
# Info: Global Buffers                    0       32        0.00%
# Info: LUTs                              99      63400     0.16%
# Info: CLB Slices                        24      15850     0.15%
# Info: Dffs or Latches                   1       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ---------------------------------------------------------------
# Info: ***********************************************************
# Info: Library: work    Cell: alu_comb    View: alu_behv
# Info: ***********************************************************
# Info:  Number of ports :                          102
# Info:  Number of nets :                           339
# Info:  Number of instances :                      270
# Info:  Number of references to this view :          0
# Info: Total accumulated area :
# Info:  Number of Dffs or Latches :                  1
# Info:  Number of LUTs :                            99
# Info:  Number of Primitive LUTs :                 101
# Info:  Number of LUTs with LUTNM/HLUTNM :           4
# Info:  Number of MUX CARRYs :                      32
# Info:  Number of accumulated instances :          270
# Info: *****************************
```

simulated with help of testbench_alu.vhd-

as combinational circuit so clear timings .

to show various testcases varying opc with time and ,kept a=1,cin=1 and b=2 as same throughout

therefore for eg opc-5(adc) result is 6.



Note: To revert to EPWave opening in a new browser window, set that option on your user page.

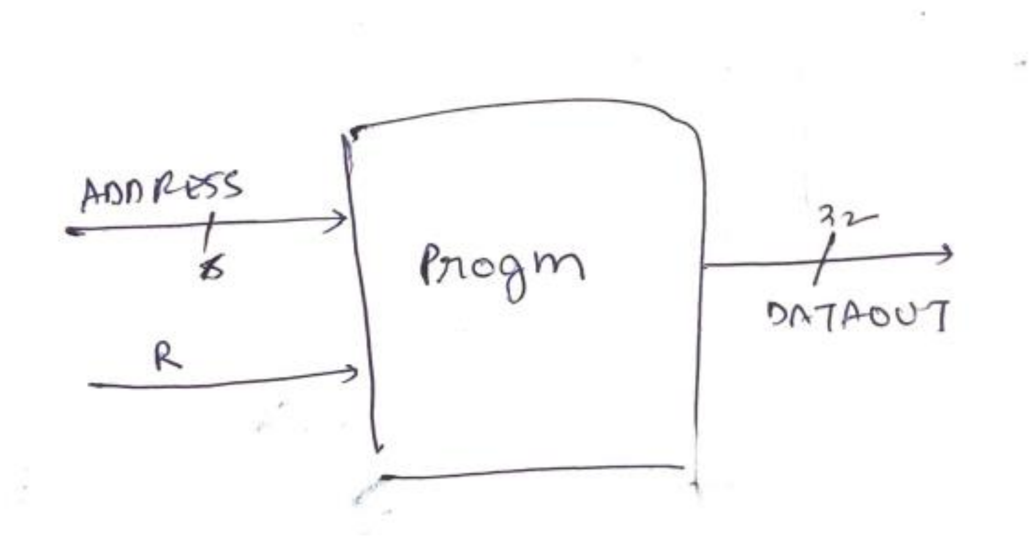
2. **prog_mem.vhd –** program memory 64x32 module design file

explanation-

it is program memory module with one read port. It contains an array of 64 std_logic_vectors of 32-bits .Its memory initialized with for loop counting.

Combinational so Read to DATAOUT when read=1 else impedence.

diagram-



synthesized resources-

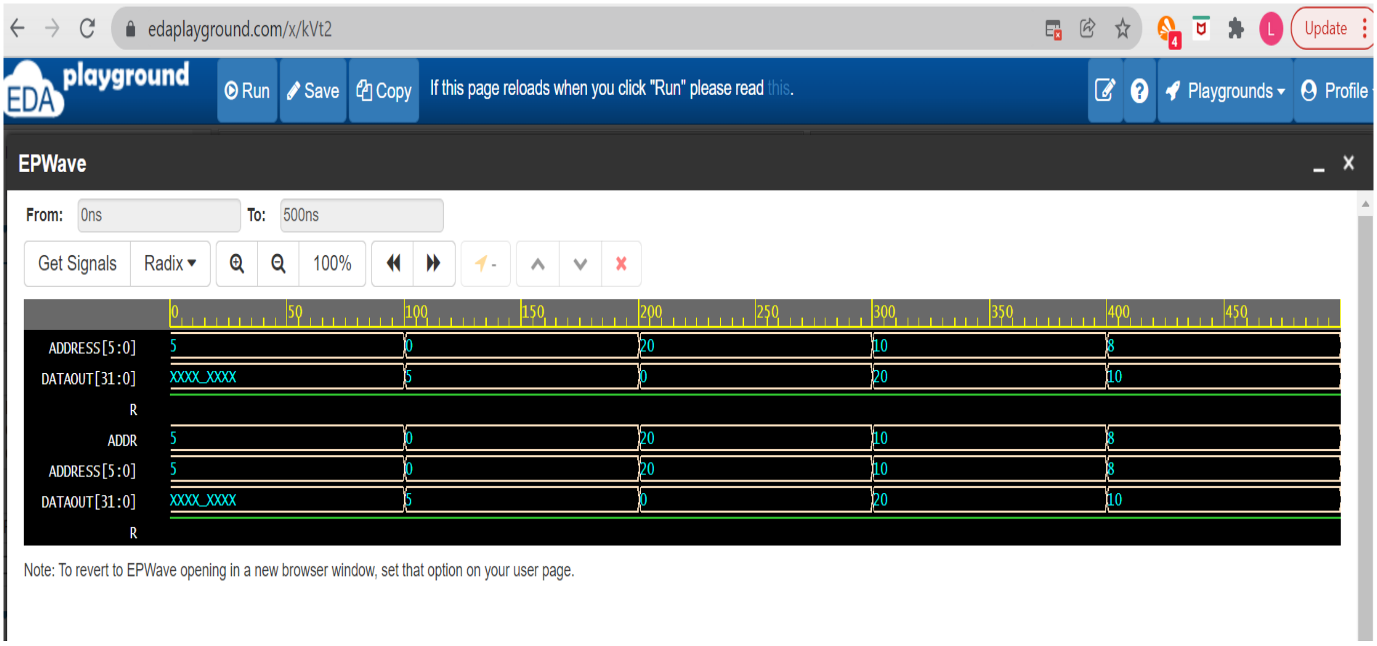
```
# Info: ***************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ***************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ---------------------------------------------------------------
# Info: IOs                               39      210      18.57%
# Info: Global Buffers                    0       32        0.00%
# Info: LUTs                              1       63400     0.00%
# Info: CLB Slices                        1       15850     0.01%
# Info: Dffs or Latches                   0       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: DSP48E1s                          0       240       0.00%
# Info: ---------------------------------------------------------------
# Info: *************************************************
# Info: Library: work    Cell: progm    View: pm
# Info: *************************************************
# Info:  Number of ports :                           39
# Info:  Number of nets :                            48
# Info:  Number of instances :                       41
# Info:  Number of references to this view :          0
# Info: Total accumulated area :
# Info:  Number of LUTs :                             1
# Info:  Number of Primitive LUTs :                   1
# Info:  Number of accumulated instances :           41
# Info: *****************************
# Info:  IO Register Mapping Report
# Info: *****************************
```

entity and ports info-

```
ENTITY progm IS
  PORT(
        ADDRESS : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
        -- read when R is 1, else
"ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ"
        R : IN STD_LOGIC;
        DATAOUT : OUT word
        );
END ENTITY;
```

simulated with help of testbench_pm.vhd-

Here in epwave ADDRESS gives location of reading and reads as DATAOUT .



Note: To revert to EPWave opening in a new browser window, set that option on your user page.

3. data_mem.vhd – data memory 64x32 module design file

explanation-

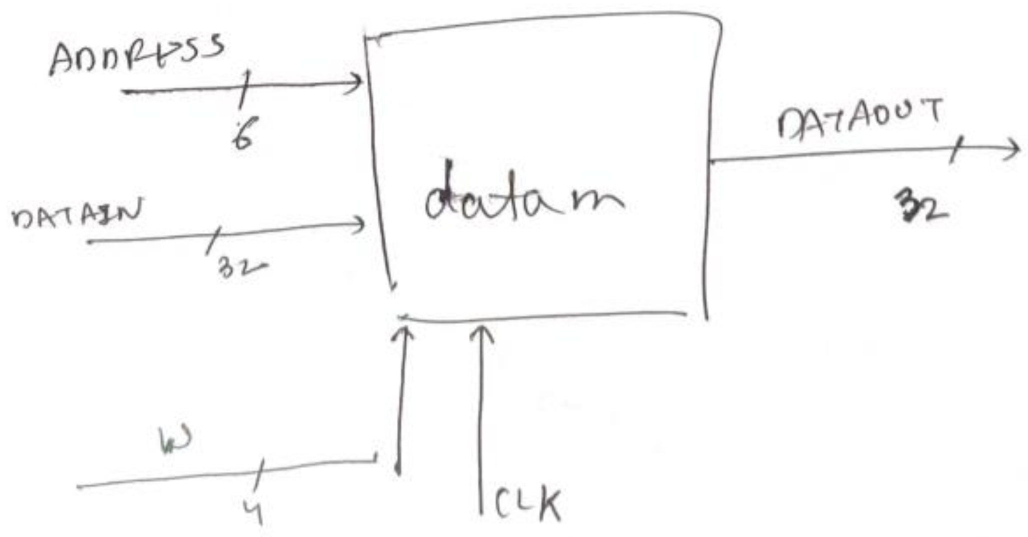
it is data memory module with one read and one write port. It contains an array of 64 std_logic_vectors of 32-bits.

read/write operations- write is clocked wheras read is unclocked (like a combinational circuit).

here W-write enable signals is of 4 bit to provide byte level write operations.

According to W, we put values in VARIABLE W0,W1,W2,W3: std_logic_vector (7 downto 0).At last concatenate write finally word.

diagram-



synthesized resources-

```
# Info: *************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: *************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: -------------------------------------------------------------
# Info: IOs                               75      210      35.71%
# Info: Global Buffers                    1       32        3.12%
# Info: LUTs                              49      63400     0.08%
# Info: CLB Slices                        1       15850     0.01%
# Info: Dffs or Latches                   0       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: Distributed RAMs
# Info:    RAM64M                         8
# Info: DSP48E1s                          0       240       0.00%
# Info: -------------------------------------------------------------
# Info: **************************************************
# Info: Library: work    Cell: datam    View: dm
# Info: **************************************************
# Info:  Number of ports :                          75
# Info:  Number of nets :                          183
# Info:  Number of instances :                     109
# Info:  Number of references to this view :         0
# Info: Total accumulated area :
# Info:  Number of LUTs :                           49
# Info:  Number of Primitive LUTs :                 65
# Info:    Number of LUTs as Distributed RAM :      32
# Info:  Number of LUTs with LUTNM/HLUTNM :         32
# Info:  Number of accumulated instances :         116
# Info: *****************************
```

entity and ports info-

```
--  entity
ENTITY datam IS
  PORT(
    ADDRESS : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
       DATAIN: IN word; --std_logic_vector (31 downto 0);
       W:IN std_logic_vector (3 downto 0);
       --in W four write enable bits to help in writing byte ,word,etc
       CLK: IN STD_LOGIC;
       DATAOUT : OUT word
       );
END ENTITY;
```

simulated with help of testbench_dm.vhd-

Not sure of delay and timings but results with clock are correct.Here in epwave ADDRESS gives location of reading or writing and reads as DATAOUT .

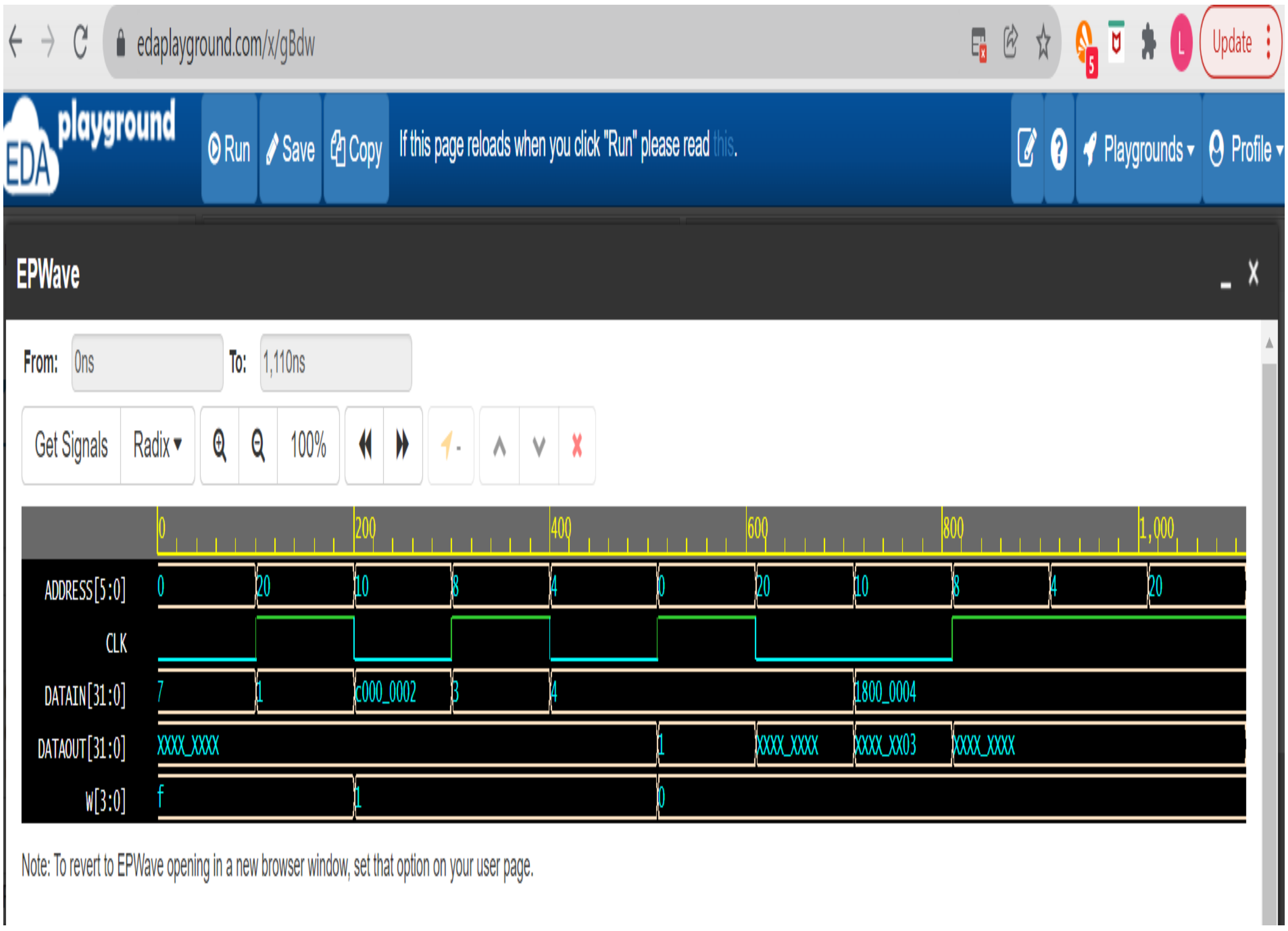
In this test first with clock we are writing 1,with W-“1111” means word write for hex address 0.

When later read at hex address 0,DATAOUT is 1

With next clock cycle we are also writing 3,with W-“0001” means byte write for hex address 10.

When later read at hex address 10,DATAOUT is XXXX_XX03.

Can also be validated that without clock or W-“0000”,nothing written at that location(identify with reading locations later).



Note: To revert to EPWave opening in a new browser window, set that option on your user page.

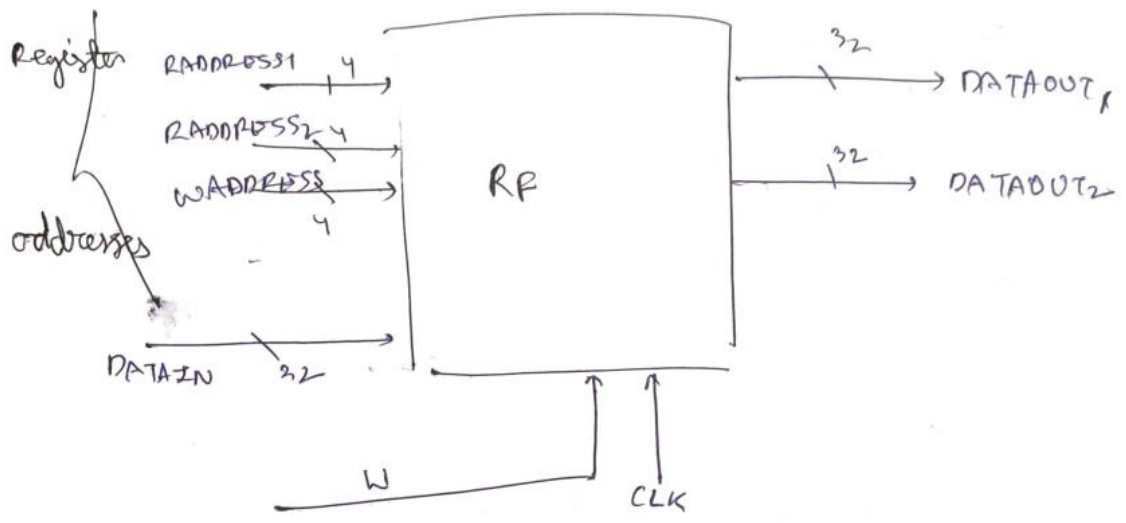
4. **rfile.vhd –** register file 16x32 module design file

explanation-

this module contains an array of 16 std_logic_vectors of 32-bits each.There are two data outputs on which contents of the array elements selected by read addresses are continuously available.Here read is unclocked (like a combinational circuit) but write is clocked.

If write enable is active, at clock edge the input data gets written in the array element selected by write address.

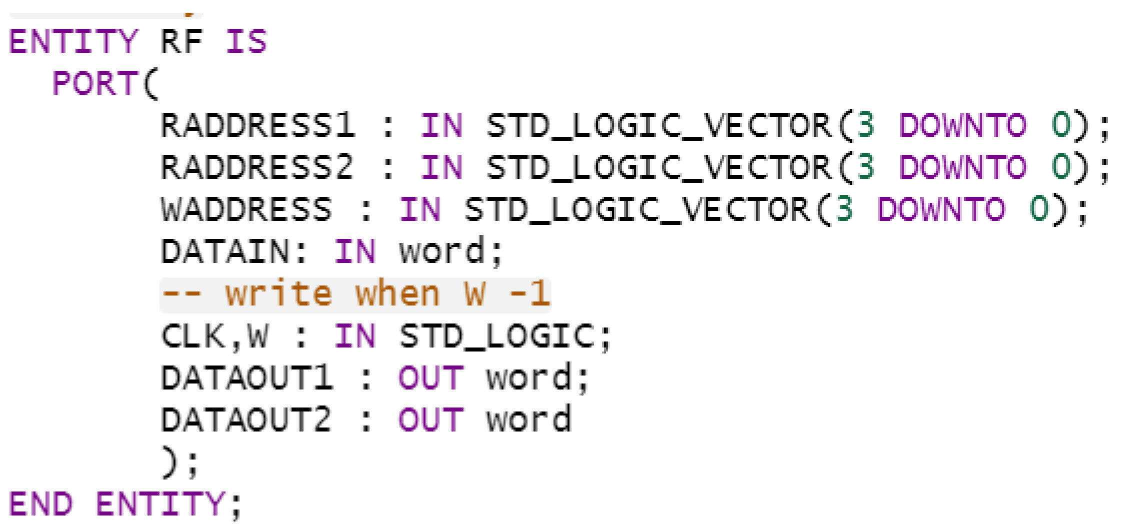
diagram-



synthesized resources-

```
# Info: ***************************************************************
# Info: Device Utilization for 7A100TCSG324
# Info: ***************************************************************
# Info: Resource                          Used    Avail   Utilization
# Info: ---------------------------------------------------------------
# Info: IOs                               110     210      52.38%
# Info: Global Buffers                    1       32        3.12%
# Info: LUTs                              48      63400     0.08%
# Info: CLB Slices                        12      15850     0.08%
# Info: Dffs or Latches                   0       126800    0.00%
# Info: Block RAMs                        0       135       0.00%
# Info: Distributed RAMs
# Info:    RAM32M                         10
# Info:    RAM64M                         2
# Info: DSP48E1s                          0       240       0.00%
# Info: ---------------------------------------------------------------
# Info: ***************************************************
# Info: Library: work    Cell: RF    View: RF_BEH
# Info: ***************************************************
# Info:  Number of ports :                              110
# Info:  Number of nets :                               220
# Info:  Number of instances :                          111
# Info:  Number of references to this view :              0
# Info: Total accumulated area :
# Info:  Number of LUTs :                                48
# Info:  Number of Primitive LUTs :                      48
# Info:    Number of LUTs as Distributed RAM :           48
# Info:  Number of accumulated instances :              123
# Info: *****************************
```

entity and ports info-



```
ENTITY RF IS
   PORT(
          RADDRESS1 : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
          RADDRESS2 : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
          WADDRESS : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
          DATAIN: IN word;
          -- write when W -1
          CLK,W : IN STD_LOGIC;
          DATAOUT1 : OUT word;
          DATAOUT2 : OUT word
          );
END ENTITY;
```

simulated with help of testbench_rf.vhd-

Not sure of delay and timings but results with clock are correct.Here in epwave RADDRESS gives location of reading and RADDRESS for writing and reads as DATAOUT1 and DATAOUT2 .

All these below read observations can be found at RADDRESS1 and DATAOUT1-

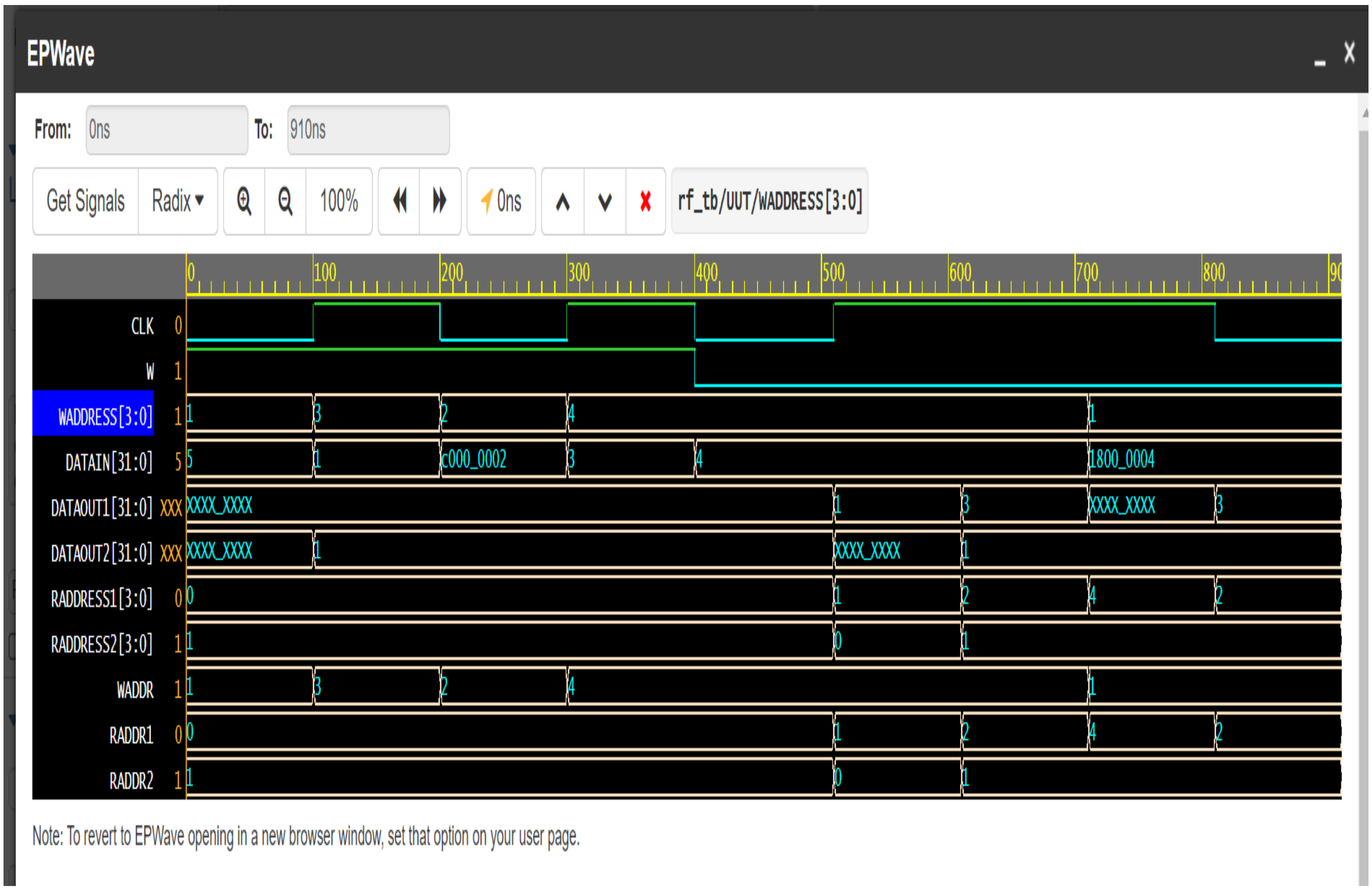
In this test with first clock we are writing 1,with W-'1' at hex address 1.

When later read at hex address 1,DATAOUT is 1

With next clock cycle we are also writing 3,with W-'1' AT hex address 2.

When later read at hex address 2,DATAOUT is 3.

Can also be validated that without clock or W-'o',nothing written at that location(identify with reading locations later) for example when WADDRESS-4 but W-'0' SO later read as "XXXX_XXXX"



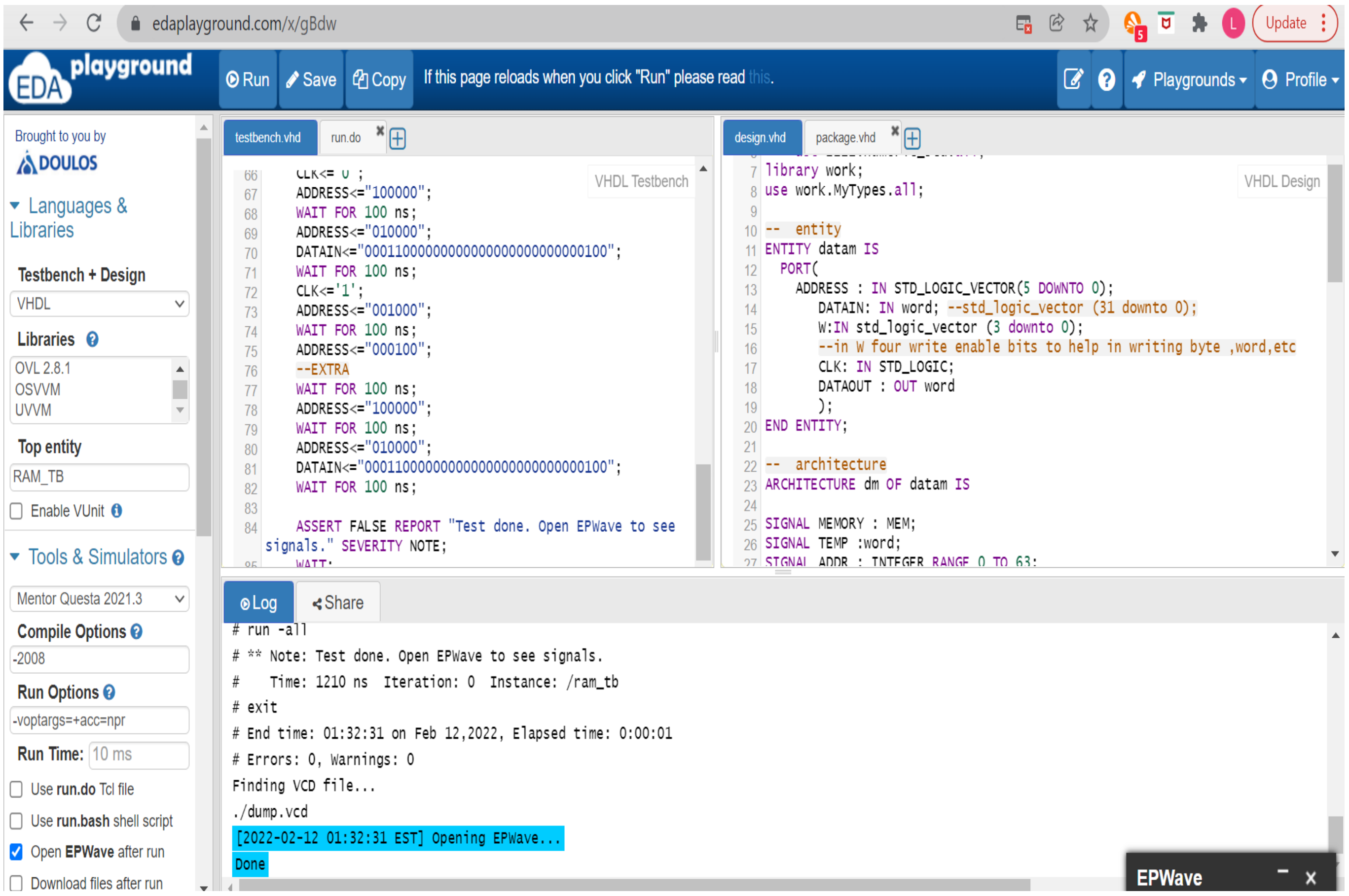
Note: To revert to EPWave opening in a new browser window, set that option on your user page.

For each module,work and analysis I have put files in edaplayground like below-

Used for simulation- Mentor Questa 2021.3

Used for synthesis- Mentor Precision 2021.1

simulation settings For eg data mem-



Testbench + Design: VHDL

Libraries: OVL 2.8.1, OSVVM, UVVM

Top entity: RAM_TB

☐ Enable VUnit

Mentor Questa 2021.3

Compile Options: -2008

Run Options: -voptargs=+acc=npr

Run Time: 10 ms

☐ Use run.do Tcl file

☐ Use run.bash shell script

☑ Open EPWave after run

☐ Download files after run

```vhdl
    ADDRESS<="100000";
    WAIT FOR 100 ns;
    ADDRESS<="010000";
    DATAIN<="00011000000000000000000000000100";
    WAIT FOR 100 ns;
    CLK<='1';
    ADDRESS<="001000";
    WAIT FOR 100 ns;
    ADDRESS<="000100";
    --EXTRA
    WAIT FOR 100 ns;
    ADDRESS<="100000";
    WAIT FOR 100 ns;
    ADDRESS<="010000";
    DATAIN<="00011000000000000000000000000100";
    WAIT FOR 100 ns;

    ASSERT FALSE REPORT "Test done. Open EPWave to see
signals." SEVERITY NOTE;
```

```vhdl
library work;
use work.MyTypes.all;

-- entity
ENTITY datam IS
  PORT(
    ADDRESS : IN STD_LOGIC_VECTOR(5 DOWNTO 0);
      DATAIN: IN word; --std_logic_vector (31 downto 0);
      W:IN std_logic_vector (3 downto 0);
      --in W four write enable bits to help in writing byte ,word,etc
      CLK: IN STD_LOGIC;
      DATAOUT : OUT word
      );
END ENTITY;

-- architecture
ARCHITECTURE dm OF datam IS

SIGNAL MEMORY : MEM;
SIGNAL TEMP :word;
```

```
# run -all
# ** Note: Test done. Open EPWave to see signals.
#    Time: 1210 ns  Iteration: 0  Instance: /ram_tb
# exit
# End time: 01:32:31 on Feb 12,2022, Elapsed time: 0:00:01
# Errors: 0, Warnings: 0
Finding VCD file...
./dump.vcd
[2022-02-12 01:32:31 EST] Opening EPWave...
Done
```

For more testcases and live running ,edaplayground can be played in demo for this stage and public link of it can be shared.