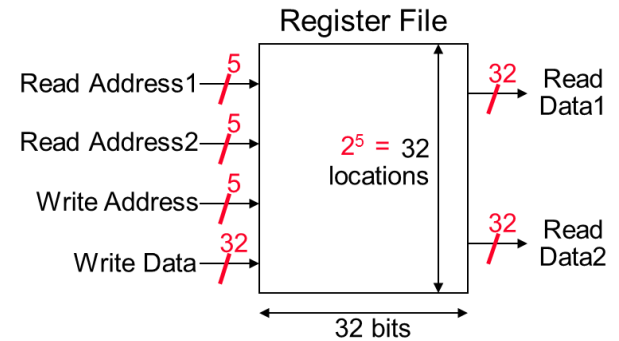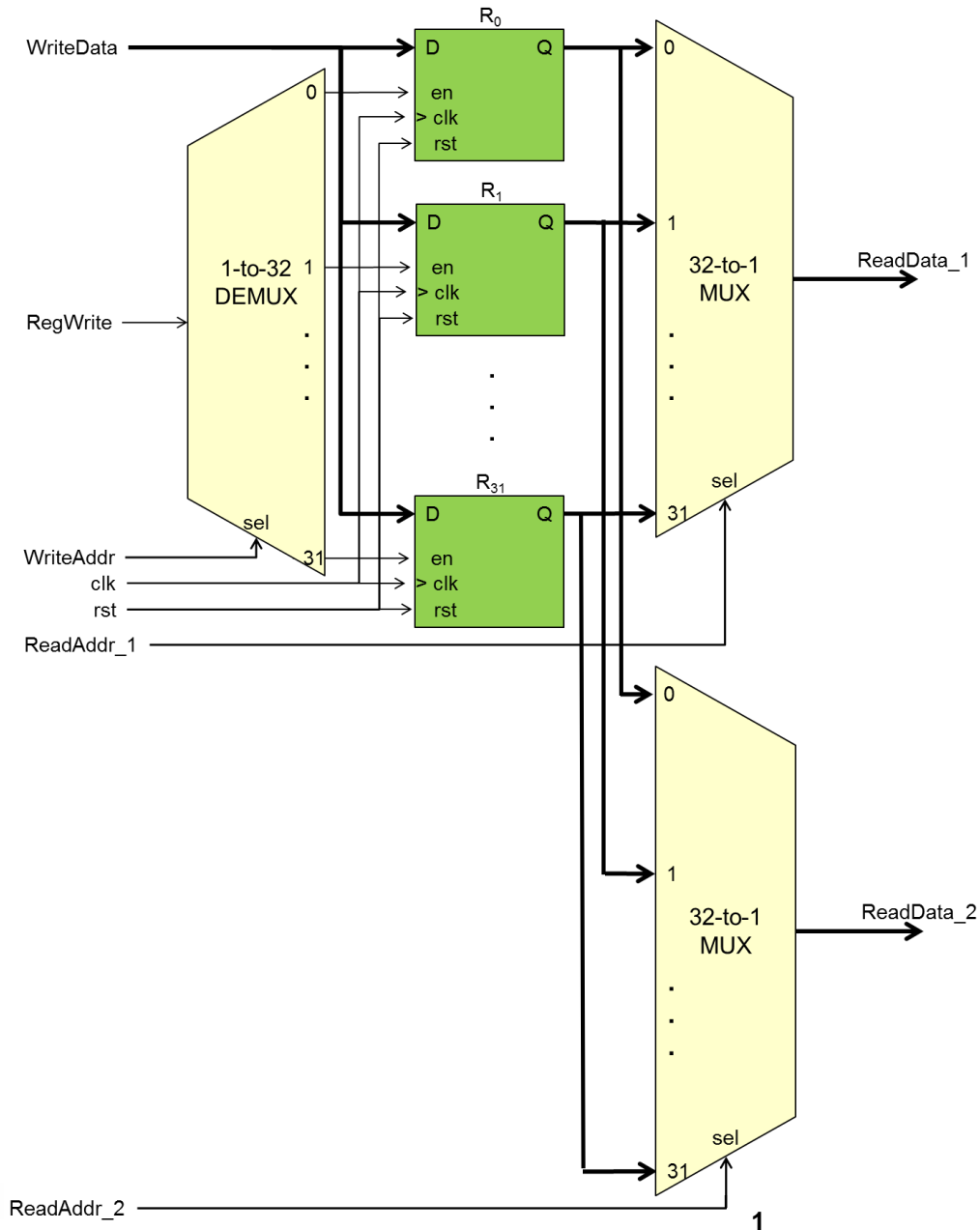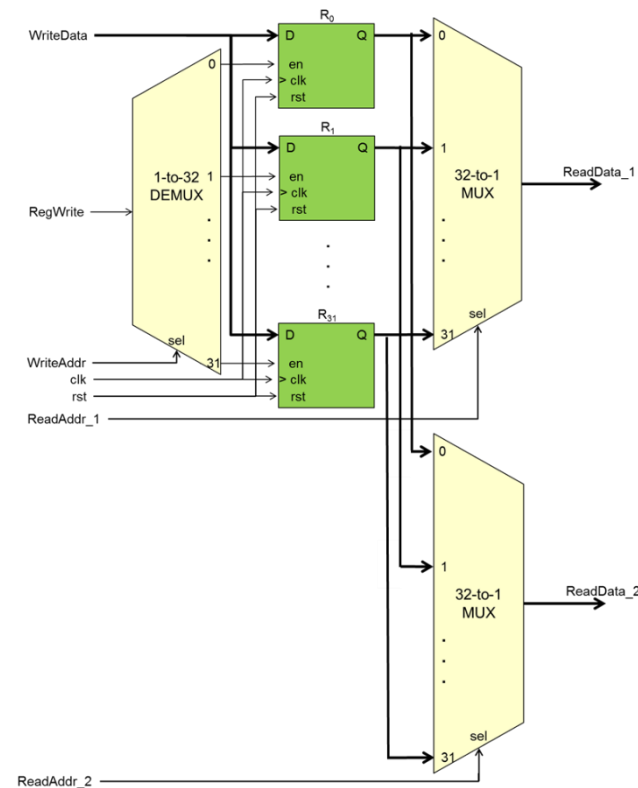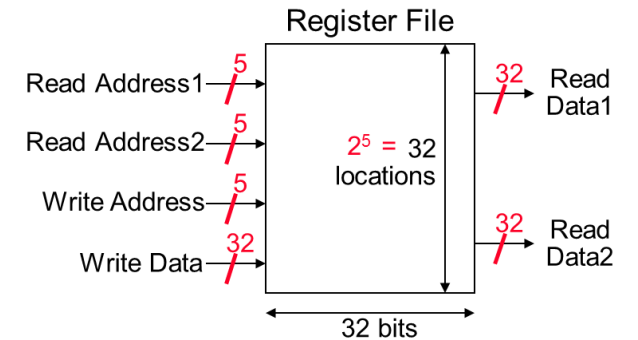# MIPS Register File

MIPS Register File

```vhdl
1   LIBRARY ieee;
2   USE ieee.std_logic_1164.all;
3   USE ieee.std_logic_unsigned.all;
4
5   ENTITY RegFile IS
6       PORT(
7           ReadAddr_1 : IN      std_logic_vector (4 DOWNTO 0);
8           ReadAddr_2 : IN      std_logic_vector (4 DOWNTO 0);
9           RegWrite   : IN      std_logic;
10          WriteAddr  : IN      std_logic_vector (4 DOWNTO 0);
11          WriteData  : IN      std_logic_vector (31 DOWNTO 0);
12          clk        : IN      std_logic;
13          rst        : IN      std_logic;
14          ReadData_1 : OUT     std_logic_vector (31 DOWNTO 0);
15          ReadData_2 : OUT     std_logic_vector (31 DOWNTO 0)
16      );
17  END RegFile ;
18
19  ARCHITECTURE struct OF RegFile IS
20      type reg_file_mem_type is array(0 to 31) of std_logic_vector(31 downto 0);
21      constant zero_register : std_logic_vector(31 downto 0) := (others => '0');
22      constant initial_reg_file : reg_file_mem_type := (others => zero_register);
23      SIGNAL reg_file_mem : reg_file_mem_type;
24
25  BEGIN
26
27
28
29
30
31
32              Insert Your Code Here
33
34
35
36
37
38  END struct;
```

Register File

Read Address1 — 5 →

Read Address2 — 5 →

Write Address — 5 →

Write Data — 32 →

$2^5 = 32$ locations

32 → Read Data1

32 → Read Data2

32 bits

WriteData

R_0 D Q en > clk rst

1-to-32 DEMUX

RegWrite

R_1 D Q en > clk rst

32-to-1 MUX ReadData_1

WriteAddr
clk
rst

R_{31} D Q en > clk rst

sel

ReadAddr_1

32-to-1 MUX ReadData_2

ReadAddr_2

MIPS Register File

```vhdl
1   LIBRARY ieee;
2   USE ieee.std_logic_1164.ALL;
3
4   -- Uncomment the following library declaration if using
5   -- arithmetic functions with Signed or Unsigned values
6   --USE ieee.numeric_std.ALL;
7
8   ENTITY regfile_tb IS
9   END regfile_tb;
10
11  ARCHITECTURE behavior OF regfile_tb IS
12
13      -- Component Declaration for the Unit Under Test (UUT)
14
15      COMPONENT RegFile
16      PORT(
17          ReadAddr_1 : IN  std_logic_vector(4 downto 0);
18          ReadAddr_2 : IN  std_logic_vector(4 downto 0);
19          RegWrite : IN  std_logic;
20          WriteAddr : IN  std_logic_vector(4 downto 0);
21          WriteData : IN  std_logic_vector(31 downto 0);
22          clk : IN  std_logic;
23          rst : IN  std_logic;
24          ReadData_1 : OUT  std_logic_vector(31 downto 0);
25          ReadData_2 : OUT  std_logic_vector(31 downto 0)
26          );
27      END COMPONENT;
28
29
30      --Inputs
31      signal ReadAddr_1 : std_logic_vector(4 downto 0) := (others => '0');
32      signal ReadAddr_2 : std_logic_vector(4 downto 0) := (others => '0');
33      signal RegWrite : std_logic := '0';
34      signal WriteAddr : std_logic_vector(4 downto 0) := (others => '0');
35      signal WriteData : std_logic_vector(31 downto 0) := (others => '0');
36      signal clk : std_logic := '0';
37      signal rst : std_logic := '0';
38
39          --Outputs
40      signal ReadData_1 : std_logic_vector(31 downto 0);
41      signal ReadData_2 : std_logic_vector(31 downto 0);
42
43      -- Clock period definitions
44      constant clk_period : time := 10 ns;
45
46  BEGIN
47
48          -- Instantiate the Unit Under Test (UUT)
49      uut: RegFile PORT MAP (
50          ReadAddr_1 => ReadAddr_1,
51          ReadAddr_2 => ReadAddr_2,
52          RegWrite => RegWrite,
53          WriteAddr => WriteAddr,
54          WriteData => WriteData,
55          clk => clk,
56          rst => rst,
57          ReadData_1 => ReadData_1,
58          ReadData_2 => ReadData_2
59          );
60
61      -- Clock process definitions
62      clk_process :process
63      begin
64              clk <= '0';
65              wait for clk_period/2;
66              clk <= '1';
67              wait for clk_period/2;
68      end process;
69
70
71      -- Stimulus process
72      stim_proc: process
73      begin
74          -- hold reset state for 100 ns.
75              rst <= '1';
76          ReadAddr_1 <= "00000";
77              ReadAddr_2 <= "00000";
78              RegWrite <= '0';
79          WriteAddr <= "00000";
80          WriteData <= "00000000000000000000000000000000";
81      wait for clk_period*10;
82
83              -- insert stimulus here
84              rst <= '0';
85              RegWrite <= '1';
86          WriteAddr <= "00101";
87          WriteData <= "00000000000000000000000000001111";
88      wait for clk_period*3;
89
90              RegWrite <= '1';
91          WriteAddr <= "01100";
92          WriteData <= "00000000000000000000000000011111";
93      wait for clk_period*3;
94
95          ReadAddr_1 <= "00101";
96              ReadAddr_2 <= "01100";
97              RegWrite <= '0';
98      wait for clk_period*3;
99
100             rst <= '1';
101     wait for clk_period*3;
102
103         wait;
104     end process;
105
106 END;
```

MIPS Register File