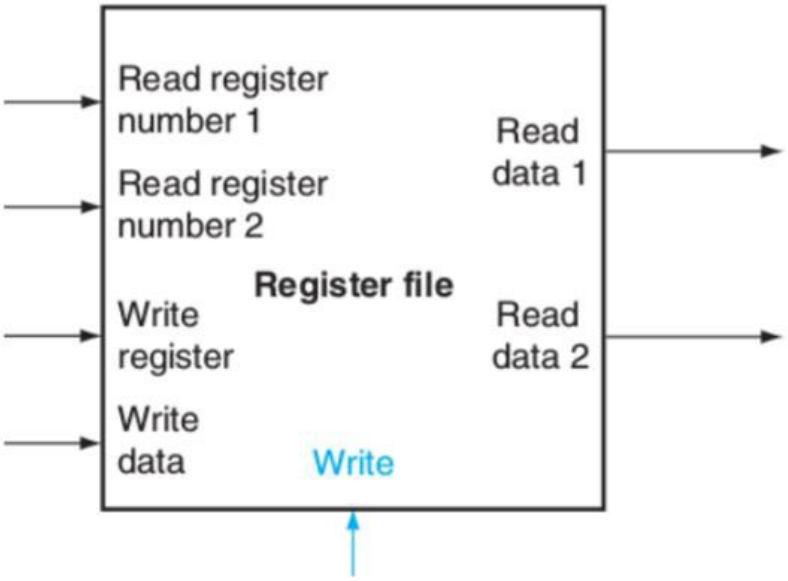
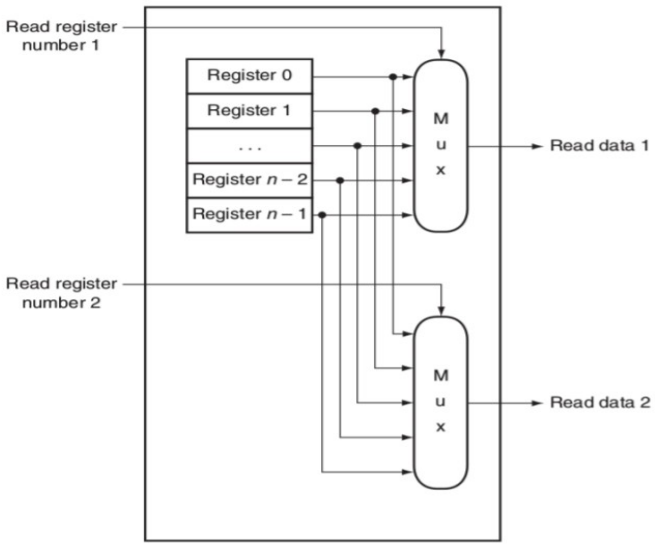
Register File



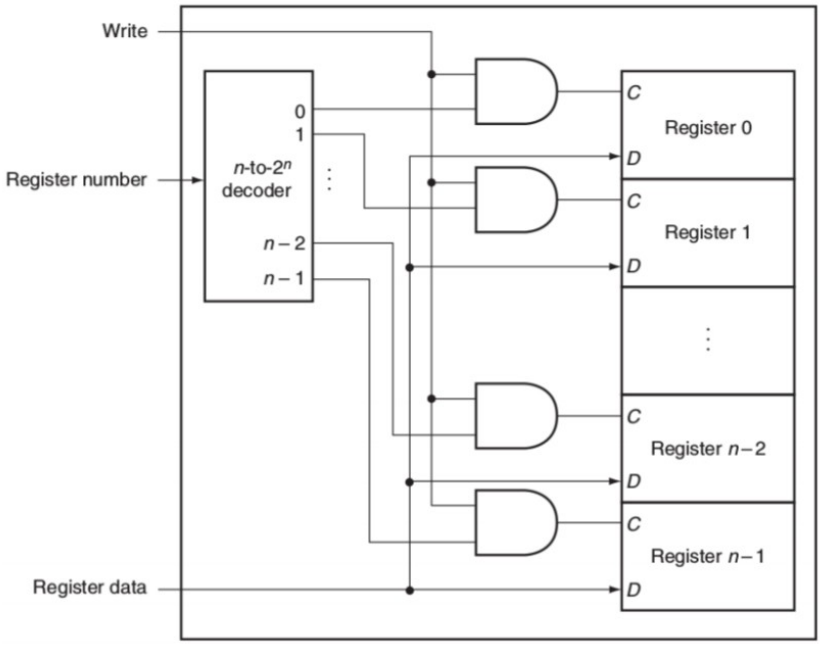
1. 32 Registers.
2. Read Register No 1 (Address of the register to read data from).
3. Read Register No 2 (Address of the register to read data from).
4. Read Data 1 (Data from the register pointed to by read register no 1).
5. Read Data 2 (Data from the register pointed to by read register no 2).
6. Write signal (Corresponds to writing data or not).
7. Write Register (Address of the data to which data has to be written)
8. Write data (Data to be written in the register pointed by write register).

Reading from Register File:



The multiplexer operation for decoding the address and 32 lines each of 32 bit wide is done using behavioural model in verilog. Decoding the address is done using switch cases in Verilog.

Writing to a Register File



The decoding of which register to write to, while taking care of the write signal is done using register is done using cases in behavioural model in Verilog.

VERILOG CODES :

File: **register_file.v**

```
`include"Reg32b.v"

module register_file (output reg [31:0] Read_Data1, output reg [31:0] Read_Data2, input RegWrite, input [31:0] Write_Data, input [4:0] Write_
Register, input [4:0] Read_Register1, input[4:0] Read_Register2);
    reg [31:0][31:0] Data_Write;
    reg [31:0] Write;
    wire [31:0][31:0] Out;
    integer i;
    Reg32b r0 (Data_Write[0], Write[0], Out[0]);
    Reg32b r1 (Data_Write[1], Write[1], Out[1]);
    Reg32b r2 (Data_Write[2], Write[2], Out[2]);
    Reg32b r3 (Data_Write[3], Write[3], Out[3]);
    Reg32b r4 (Data_Write[4], Write[4], Out[4]);
    Reg32b r5 (Data_Write[5], Write[5], Out[5]);
    Reg32b r6 (Data_Write[6], Write[6], Out[6]);
    Reg32b r7 (Data_Write[7], Write[7], Out[7]);
    Reg32b r8 (Data_Write[8], Write[8], Out[8]);
    Reg32b r9 (Data_Write[9], Write[9], Out[9]);
    Reg32b r10 (Data_Write[10], Write[10], Out[10]);
    Reg32b r11 (Data_Write[11], Write[11], Out[11]);
    Reg32b r12 (Data_Write[12], Write[12], Out[12]);
    Reg32b r13 (Data_Write[13], Write[13], Out[13]);
    Reg32b r14 (Data_Write[14], Write[14], Out[14]);
    Reg32b r15 (Data_Write[15], Write[15], Out[15]);
    Reg32b r16 (Data_Write[16], Write[16], Out[16]);
    Reg32b r17 (Data_Write[17], Write[17], Out[17]);
    Reg32b r18 (Data_Write[18], Write[18], Out[18]);
    Reg32b r19 (Data_Write[19], Write[19], Out[19]);
    Reg32b r20 (Data_Write[20], Write[20], Out[20]);
    Reg32b r21 (Data_Write[21], Write[21], Out[21]);
    Reg32b r22 (Data_Write[22], Write[22], Out[22]);
    Reg32b r23 (Data_Write[23], Write[23], Out[23]);
    Reg32b r24 (Data_Write[24], Write[24], Out[24]);
    Reg32b r25 (Data_Write[25], Write[25], Out[25]);
    Reg32b r26 (Data_Write[26], Write[26], Out[26]);
    Reg32b r27 (Data_Write[27], Write[27], Out[27]);
    Reg32b r28 (Data_Write[28], Write[28], Out[28]);
```

```
View  Search  Terminal  Help
    Reg32b r9 (Data_Write[9], Write[9], Out[9]);
    Reg32b r10 (Data_Write[10], Write[10], Out[10]);
    Reg32b r11 (Data_Write[11], Write[11], Out[11]);
    Reg32b r12 (Data_Write[12], Write[12], Out[12]);
    Reg32b r13 (Data_Write[13], Write[13], Out[13]);
    Reg32b r14 (Data_Write[14], Write[14], Out[14]);
    Reg32b r15 (Data_Write[15], Write[15], Out[15]);
    Reg32b r16 (Data_Write[16], Write[16], Out[16]);
    Reg32b r17 (Data_Write[17], Write[17], Out[17]);
    Reg32b r18 (Data_Write[18], Write[18], Out[18]);
    Reg32b r19 (Data_Write[19], Write[19], Out[19]);
    Reg32b r20 (Data_Write[20], Write[20], Out[20]);
    Reg32b r21 (Data_Write[21], Write[21], Out[21]);
    Reg32b r22 (Data_Write[22], Write[22], Out[22]);
    Reg32b r23 (Data_Write[23], Write[23], Out[23]);
    Reg32b r24 (Data_Write[24], Write[24], Out[24]);
    Reg32b r25 (Data_Write[25], Write[25], Out[25]);
    Reg32b r26 (Data_Write[26], Write[26], Out[26]);
    Reg32b r27 (Data_Write[27], Write[27], Out[27]);
    Reg32b r28 (Data_Write[28], Write[28], Out[28]);
    Reg32b r29 (Data_Write[29], Write[29], Out[29]);
    Reg32b r30 (Data_Write[30], Write[30], Out[30]);
    Reg32b r31 (Data_Write[31], Write[31], Out[31]);

    always @ (RegWrite or Write_Register or Read_Register1 or Read_Register2 or Write_Data) begin
        if(RegWrite == 1'b1) begin
            Write[Write_Register] <= 1'b1;
            Data_Write[Write_Register] <= Write_Data;
        end

        else begin
            Write[Read_Register1] <= 1'b0;
            Read_Data1 <= Out[Read_Register1];
            Write[Read_Register2] <= 1'b0;
            Read_Data2 <= Out[Read_Register2];
        end

    end
endmodule
```

OUTPUT :

```
File  Edit  View  Search  Terminal  Help
VCD info: dumpfile register_file.vcd opened for output.
Read Register1 =  0             Read Data1 =          x
Read Register2 =  0             Read Data2 =          x

Read Register1 =  0             Read Data1 = 2878068402
Read Register2 =  0             Read Data2 = 2878068402

Read Register1 =  0             Read Data1 = 2878068402
Read Register2 =  1             Read Data2 = 2878068403

Read Register1 =  0             Read Data1 =  969834398
Read Register2 =  1             Read Data2 = 2878068403

Read Register1 =  0             Read Data1 =  969834398
Read Register2 =  1             Read Data2 = 3222275104

Read Register1 = 26             Read Data1 = 2878068396
Read Register2 =  1             Read Data2 = 3222275104

Read Register1 = 26             Read Data1 = 2878068396
Read Register2 =  3             Read Data2 = 2878068373

Read Register1 =  6             Read Data1 = 2878068376
Read Register2 =  3             Read Data2 = 2878068373

Read Register1 =  6             Read Data1 = 2878068376
Read Register2 =  9             Read Data2 = 2878068379

Read Register1 = 21             Read Data1 = 2878068391
Read Register2 =  9             Read Data2 = 2878068379

Read Register1 = 21             Read Data1 = 2878068391
Read Register2 = 29             Read Data2 = 2878068399

Read Register1 =  6             Read Data1 = 2878068376
Read Register2 = 29             Read Data2 = 2878068399

Read Register1 =  6             Read Data1 = 2878068376
Read Register2 = 25             Read Data2 = 2878068395

hackspot@hackspot-inspiron-3521:~/code/Verilog/A5$
```