```
`timescale 1ns / 1ps
module rom(romaddress,instr_reg);
//////////////////////// ROM I/O declaration //////////////
input [4:0] romaddress;
output [31:0] instr_reg;
wire [7:0] instr_1;
wire [7:0] instr_2;
wire [7:0] instr_3;
wire [7:0] instr_4;
reg [7:0] ROM [31:0];
integer i;
//////////////////// assign 4x8 wires to read 32-bit data/////////
assign instr_1 = ROM[romaddress];
assign instr_2 = ROM[romaddress+32'd1];
assign instr_3 = ROM[romaddress+32'd2];
assign instr_4 = ROM[romaddress+32'd3];
assign instr_reg = {instr_4,instr_3,instr_2,instr_1};
////////////////// ROM initial values ///////////////////////
initial
begin
ROM[0] = 8'b00010011;
ROM[1] = 8'b10001001;
ROM[2] = 8'b00110000;
ROM[3] = 8'b00000000;
ROM[4] = 8'b10010011;
ROM[5] = 8'b10001001;
ROM[6] = 8'b00100000;
ROM[7] = 8'b00000000;
ROM[8] = 8'b01100011;
ROM[9] = 8'b10000010;
ROM[10] = 8'b10011100 ;
ROM[11] = 8'b00000001;
ROM[12] = 8'b00010011;
ROM[13] = 8'b00010011;
ROM[14] = 8'b00010011;
ROM[15] = 8'b00010011;
ROM[16] = 8'b00110011;
ROM[17] = 8'b10000011;
ROM[18] = 8'b00101001;
ROM[19] = 8'b00000001;
ROM[21] = 8'b11111111;
for(i=22;i<=1023;i=i+1)
begin
ROM[i] = 8'd0;
end
end
```

```
endmodule
```