Verilog code

module Slave(

inout wire SDA,

input wire SCL);

reg [4:0] IDLE = 4'b0000;

reg [4:0] START = 4'b0001;

reg [4:0] READ\_ADDRESS = 4'b0010;

reg [4:0] READ\_WRITE = 4'b0011;

reg [4:0] DATA = 4'b0100;

reg [4:0] DATA\_ACK = 4'b0101;

reg [4:0] STOP = 4'b0110;

reg [4:0] ADDRESS\_ACK = 4'b0111;

reg [4:0] state = 4'b0010;

reg [6:0] slaveAddress = 7'b000\_1000;

reg [6:0] addr = 7'b000\_0000;

reg [6:0] addressCounter = 7'b000\_0000;

reg [7:0] data = 8'b0000\_0000;

reg [6:0] dataCounter = 7'b000\_0000;

reg readWrite = 1'b0;

reg start = 0;

reg write\_ack = 0;

assign SDA = (write\_ack == 1) ? 0 : 'b1z;

always @(negedge SDA) begin

if ((start == 0) && (SCL == 1))

begin

start <= 1;

addressCounter <= 0;

dataCounter <= 0;

end

end

always @(posedge SDA) begin

if (state == DATA\_ACK && SCL == 1)

begin

start <= 0;

state <= READ\_ADDRESS;

end

end

always @(posedge SCL)

begin

if (start == 1)

begin

case (state)

READ\_ADDRESS:

begin

addr[addressCounter] <= SDA;

addressCounter <= addressCounter + 1;

if (addressCounter == 6)

begin

state <= READ\_WRITE;

end

end

READ\_WRITE:

begin

readWrite <= SDA;

state <= ADDRESS\_ACK;

end

ADDRESS\_ACK:

begin

write\_ack <= 1;

state <= DATA;

end

DATA:

begin

write\_ack <= 0;

data[dataCounter] <= SDA;

dataCounter <= dataCounter + 1;

if (dataCounter == 8)

begin

state <= DATA\_ACK;

write\_ack <= 1;

end

end

DATA\_ACK:

begin

write\_ack <= 0;

state <= STOP;

end

STOP:

begin

start <= 0;

state <= READ\_ADDRESS;

end

endcase

end

end

endmodule