Project report on

Fully Automatic Washing Machine

Group Members:

1. Mohammed Hussain BT22ECI001
2. Shashank Rao BT22ECI031
3. Utkarsh Ghore BT22ECI032
4. Arjun Garg BT22ECI037
5. Sahil Kumar BT22ECI043

A report submitted for the partial fulfilment of the requirements of the course

ECL-106 DSDg

Submission Date: 03/07/2023

Under the guidance of:

Dr. Mayank B. Thacker

Department of Electronics and Communication Engineering

भारतीय सूचना प्रौद्योगिकी संस्थान, नागपुर

Indian Institute of Information Technology, Nagpur

Table of Contents:

Chapter 1: Introduction

This project proposes to demonstrate the capabilities and scope of Verilog HDL by implementing the control system of an automatic washing machine. The above mentioned objective is achieved by implementing the Control System of an automatic washing using the Finite State Machine model. The washing machine control system generates all the control signals required for the operation of washing machine and is designed using Verilog HDL.

The module uses a finite-state machine (FSM) to control the operation of the washing machine. The internal states are defined using parameters. In this example, we have five states: IDLE,READY, SOAK, RINSE, and SPIN.

STATE DIADRAM:

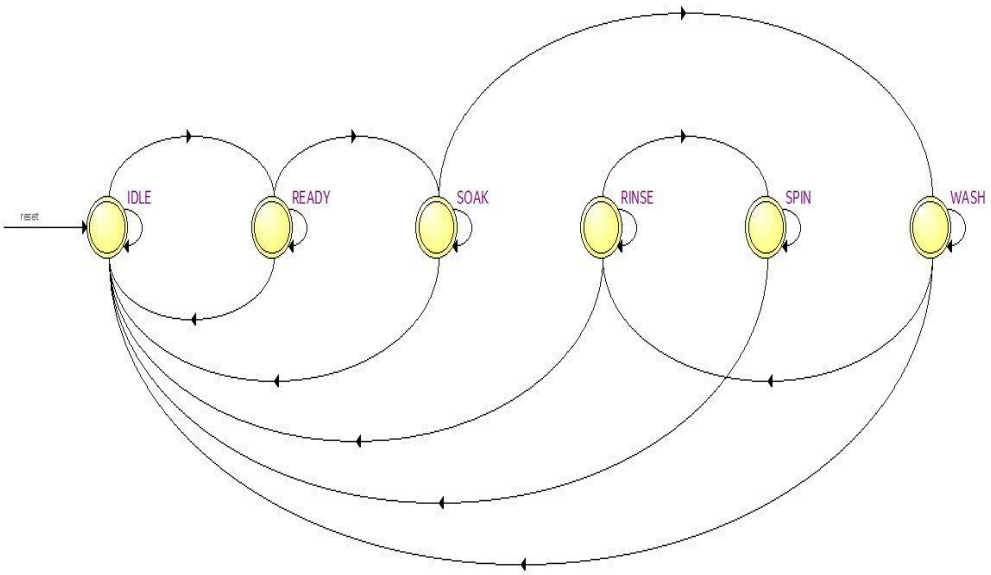


Fig 1.1

Chapter 2: DE10 Lite Board

Terasic DE10-Lite is a cost-effective Intel® MAX® 10 based FPGA board. The board utilizes the maximum capacity Intel® MAX® 10 FPGA, which has around 50K logic elements (LEs) and on-die analog-to-digital converter (ADC). It features on-board USB-Blaster, SDRAM, accelerometer, VGA output, 2x20 GPIO expansion connector, and an expansion connector. The kit provides the perfect system-level prototyping solution for industrial, automotive, consumer, and many other market applications. The DE10-Lite kit also contains lots of reference designs and software utilities for users to easily develop their applications based on these design resources.

DE10 Lite Board :

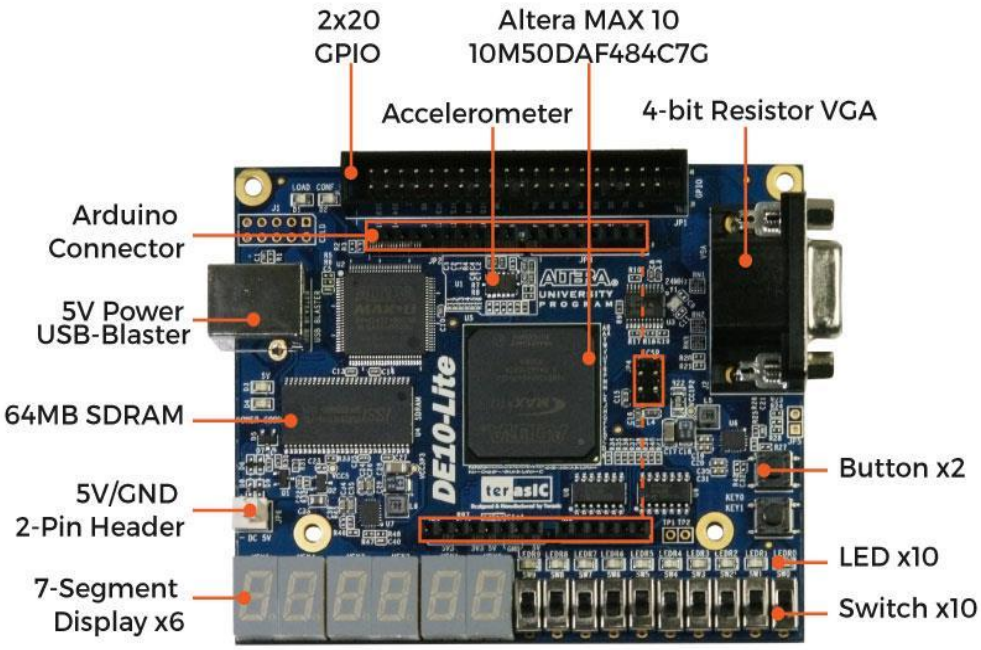


Fig 1.2

Chapter 3:Working of Machine

The Verilog code implements a state machine-based control system for an automatic washing machine, providing different operations based on the selected mode and handling inputs such as lid status, start button, cancel button, and coin input.

The washing machine module consists of various inputs and outputs, including the clock signal (i_clk), lid status (i_lid), start button (i_start), cancel button (i_cancel), coin input (i_coin), mode selection inputs (i_mode_1, i_mode_2, i_mode_3), and various control signals and display outputs.

The core of the washing machine's functionality is implemented using a state machine. The state machine represents different states of the washing machine, such as IDLE, READY, SOAK, WASH, RINSE, and SPIN. Each state corresponds to a specific operation of the washing machine.

The state machine is driven by a clock signal (i_clk). On each positive edge of the clock, the current state (PS) is updated based on the inputs and the current state. The next state (NS) is determined using a case statement that considers the current state and input conditions.

The present state (PS) and next state (NS) of the state machine are stored in registers, allowing for sequential execution of operations. The initial state of the machine is set to IDLE.

The output logic determines the state of various signals based on the current state. For example, the o_idle signal is asserted when the washing machine is in the IDLE state, indicating that it is not performing any operation. Similarly, the o_ready signal is asserted when the washing machine is in the READY state, indicating that it is ready to start a new wash cycle.

The o_soak, o_wash, o_rinse, and o_spin signals are used to indicate the corresponding operations' progress. These signals are asserted when the washing machine is in the respective states, allowing external components to monitor the progress of each operation.

The o_waterinlet signal is asserted when the washing machine is in the SOAK, WASH, or RINSE states, indicating that water should be supplied to the machine for these operations.

The o_coinreturn signal is asserted when the washing machine is in the READY state and the cancel button (i_cancel) is pressed. This signal indicates that the coin inserted by the user should be returned.

The o_done signal is asserted when the washing machine completes the SPIN state and the spin_done flag is set. This signal can be used to trigger external actions or indicate the end of the wash cycle.

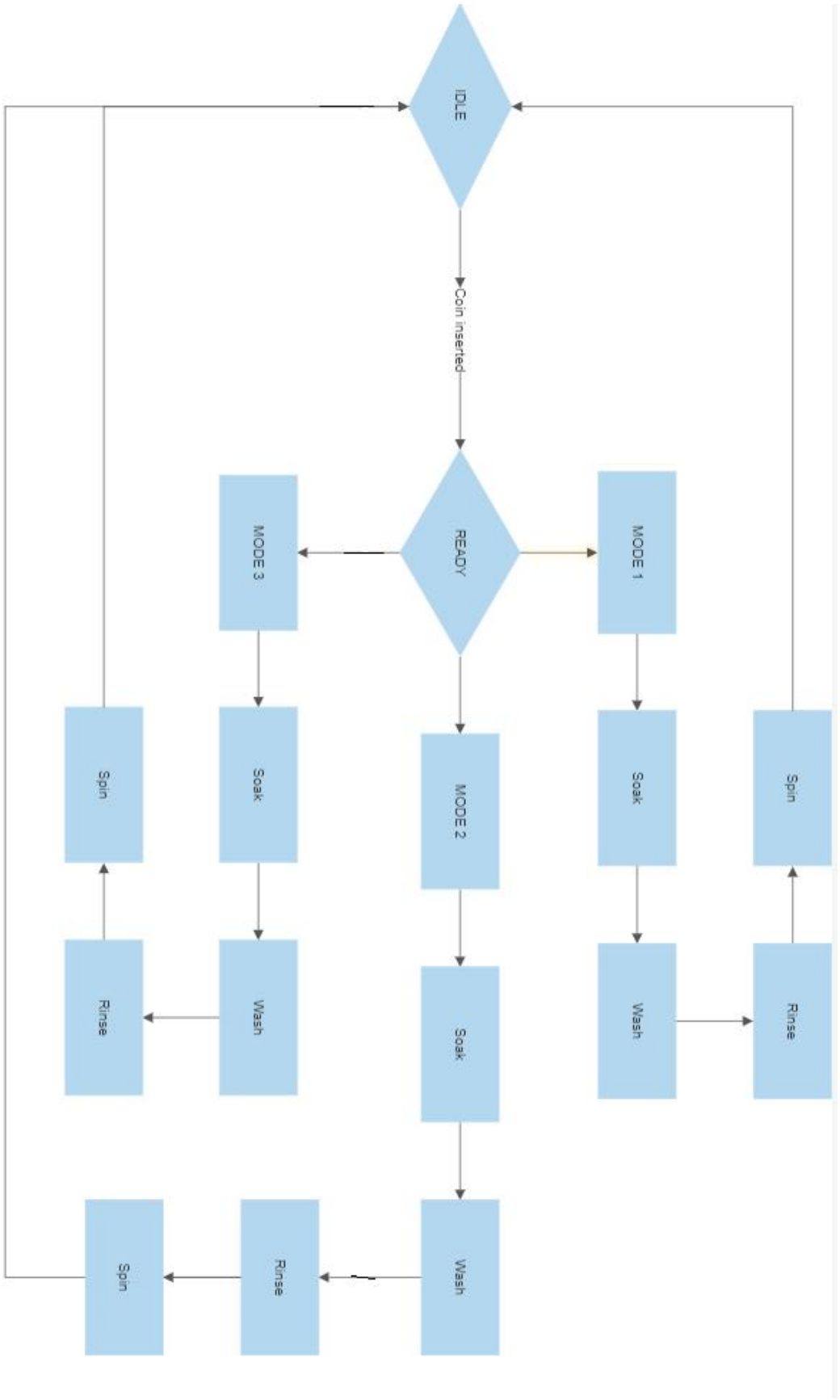
To implement the timers for each operation, the code uses counters (soakcounter, washcounter, rinsecounter, spincounter). These counters are incremented on each positive edge of the clock when the corresponding operation is active (soak_up, wash_up, rinse_up, spin_up). The duration for each operation varies based on the selected mode (i_mode_1, i_mode_2, i_mode_3), as mentioned in the comments. When the corresponding operation's timer reaches its maximum value, the respective done flag (soak_done, wash_done, rinse_done, spin_done) is set, indicating that the operation is complete.

To handle the case when the lid is open during an operation, the code includes logic to pause the timers. The soak_pause, wash_pause, rinse_pause, and spin_pause signals are asserted when the corresponding operation is active (PS == SOAK, PS == WASH, PS == RINSE, PS == SPIN) and the lid is open (i_lid). This prevents the timers from incrementing while the lid is open.

The code also includes display outputs (hex0 to hex5) that represent the current state of the washing machine on a seven-segment display. Each state corresponds to a specific number or character, allowing the user to visualize the current state of the machine.

Overall, the project implements a state machine-based control system for an automatic washing machine. It provides different operations based on the selected mode and handles inputs such as lid status, start button, cancel button, and coin input. The timers ensure that each operation is performed for the appropriate duration, while the display outputs allow the user to monitor the machine's current state.

Chapter4: Flowchart



Chapter 5:Verilog Code

```
`timescale 1ms / 1ms

 module
washingmachine(i_clk,i_lid,i_start,i_cancel,i_coin,i_mode_1,i_mode_2,i
_mode_3,
o_idle,o_ready,o_soak,o_wash,o_rinse,o_spin,o_coinreturn,o_waterinlet,
o_done,hex0,hex1,hex2,hex3,hex4,hex5);


input i_clk,i_start,i_cancel,i_coin,i_lid,i_mode_1,i_mode_2,i_mode_3;

output o_idle,o_ready,o_soak,o_wash,o_rinse,o_spin,o_waterinlet;

output o_coinreturn,o_done;

output reg [0:6] hex0,hex1,hex2,hex3,hex4,hex5;


parameter    IDLE      = 6'b000001,
             READY  = 6'b000010,
             SOAK     = 6'b000100,
             WASH    = 6'b001000,
             RINSE    = 6'b010000,
             SPIN       = 6'b100000;
  // i_clk = 50 Mhz means 1sec = 50 * 10^6 clock cycles
/* Washing Modes:
mode1 - cloth < 2kg
      Soak  : 2 seconds   (2 * 50 * 10^6  = 100,000,000)
      Wash  : 2 seconds   (2 * 50 * 10^6  = 100,000,000)
      Rinse : 2 seconds   (2 * 50 * 10^6  = 100,000,000)
```

Spin : 2 seconds (2 * 50 * 10^6 = 100,000,000)

mode2 - 2kg < cloth < 4kg

Soak : 5 seconds (5 * 50 * 10^6 = 250,000,000)

Wash : 5 seconds (5 * 50 * 10^6 = 250,000,000)

Rinse : 5 seconds (5 * 50 * 10^6 = 250,000,000)

Spin : 5 seconds (5 * 50 * 10^6 = 250,000,000)

mode3 - 4kg < cloth < 6kg

Soak : 10 seconds (10 * 50 * 10^6 = 500,000,000)

Wash : 10 seconds (10 * 50 * 10^6 = 500,000,000)

Rinse : 10 seconds (10 * 50 * 10^6 = 500,000,000)

Spin : 10 seconds (10 * 50 * 10^6 = 500,000,000)

(The values have been set significantly lesser than real world scenario for easier demonstration in the lab)

*/

```
reg [5:0] PS,NS;
reg soak_done,wash_done,rinse_done,spin_done;
wire soak_up,wash_up,rinse_up,spin_up;
wire soak_pause,wash_pause,rinse_pause,spin_pause;
reg [30:0] soakcounter,washcounter,rinsecounter,spincounter; //30 bits can count upto 1,073,741,824
```

```
//------- Timer pause logic when lid is open ------------------
                assign soak_pause  = ( PS == SOAK )  && (i_lid) ;
                assign wash_pause  = ( PS == WASH )  && (i_lid) ;
                assign rinse_pause = ( PS == RINSE ) && (i_lid) ;
                assign spin_pause  = ( PS == SPIN )  && (i_lid) ;

assign soak_up  = (PS == SOAK) && (i_mode_1 || i_mode_2 ||
i_mode_3);
assign wash_up  = (PS == WASH);
assign rinse_up = (PS == RINSE);
assign spin_up  = (PS == SPIN);
//---------- SOAK DONE LOGIC ----------------------------

always@(i_mode_1,i_mode_2,i_mode_3,soakcounter)
   begin
   if(i_mode_1)
         soak_done = (soakcounter == 100000000) ? 1'b1 : 1'b0;
               else if(i_mode_2)
                 soak_done = (soakcounter == 250000000) ? 1'b1 : 1'b0;
               else if(i_mode_3)
             soak_done = (soakcounter == 500000000) ? 1'b1 : 1'b0;
      end
     //---------- WASH DONE LOGIC ----------------------------
     always@(i_mode_1,i_mode_2,i_mode_3,washcounter)
          begin
```

```
        if(i_mode_1)
        wash_done = (washcounter == 100000000) ? 1'b1 : 1'
              else if(i_mode_2)
                wash_done = (washcounter == 250000000) ? 1'b1 : 1'b0;
                  else if(i_mode_3)
                wash_done = (washcounter == 500000000) ? 1'b1 : 1'b0;
end
 //---------- RINSE DONE LOGIC ----------------------------
always@(i_mode_1,i_mode_2,i_mode_3,rinsecounter)
        begin
          if(i_mode_1)
              rinse_done = (rinsecounter == 100000000) ? 1'b1 : 1'b0;
              else if(i_mode_2)
                rinse_done = (rinsecounter == 250000000) ? 1'b1 : 1'b0;
              else if(i_mode_3)
                rinse_done = (rinsecounter == 500000000) ? 1'b1 : 1'b0;
        end
//---------- SPIN DONE LOGIC ----------------------------
 always@(i_mode_1,i_mode_2,i_mode_3,spincounter)
   begin
   if(i_mode_1)
        spin_done = (spincounter == 100000000) ? 1'b1 : 1'b0;
              else if(i_mode_2)
                spin_done = (spincounter == 250000000) ? 1'b1 : 1'b0;
           else if(i_mode_3)
                spin_done = (spincounter == 500000000) ? 1'b1 : 1'b0;
```

```
end
//----------- SOAK TIMER ----------------------------------
always@(posedge i_clk)
        begin
          if(i_start)
            soakcounter <= 0;
            if(soak_done)
               soakcounter <= 0;
                  else if(soak_pause)
                        soakcounter <= soakcounter;
                        else if(soak_up)
                                soakcounter <= soakcounter + 1'b1;
end
 //----------- WASH TIMER ----------------------------------
 always@(posedge i_clk)
     begin
         if(i_start)
                washcounter <= 0;
                 else if(wash_done)
                washcounter <= 0;
                 else if(wash_pause)
                washcounter <= washcounter;
                 else if(wash_up)
                      washcounter <= washcounter + 1'b1;
end
```

```
//----------- RINSE TIMER ----------------------------------
always@(posedge i_clk)
begin
        if(i_start)
              rinsecounter <= 0;
                  else if(rinse_done)
               rinsecounter <= 0;
                        else if(rinse_pause)
               rinsecounter <= rinsecounter;
            else if(rinse_up)
               rinsecounter <= rinsecounter + 1'b1;
          end
//----------- SPIN TIMER ----------------------------------
 always@(posedge i_clk)
          begin
            if(i_start)
           spincounter <= 0;
                  else if(spin_done)
                       spincounter <= 0;
                          else if(spin_pause)
                                spincounter <= spincounter;
                                else if(spin_up)
                                spincounter <= spincounter + 1'b1;
        end
```

```
//---------------- Present state logic ----------------------
always@(posedge i_clk)
 Begin
        if(i_start)
             PS <= IDLE;
             else if(i_cancel)
              PS <= IDLE;
              else
               PS <= NS;
end
//----------- Next state decoder logic ---------------------------------
always@(*)
begin
        case(PS)
        IDLE : begin
                    // display IdLE
                     hex0 = 7'b1111111;
                     hex1 = 7'b1111111;
                     hex2 = 7'b0110000;
                     hex3 = 7'b1110001;
                     hex4 = 7'b1000010;
                     hex5 = 7'b1111001;
                     if(~i_coin && !i_lid && !i_cancel)
                     NS <= READY;
```

```
                else
                    NS <= PS;
                                end

                    READY : begin
                                    // display rEAdy
                hex0 = 7'b1111111;
                hex1 = 7'b1000100;
                hex2 = 7'b1000010;
                hex3 = 7'b0001000;
                hex4 = 7'b0110000;
                hex5 = 7'b1111010;

                                if(!i_lid && !i_cancel &&
(i_mode_1 || i_mode_2 || i_mode_3))
                                  NS <= SOAK;
                                else
                    NS <= PS;
            end
            SOAK : begin
              //display SoAk
                hex0 = 7'b1111111;
                hex1 = 7'b1111111;
                hex2 = 7'b1111000;
                hex3 = 7'b0001000;
```

```
        hex4 = 7'b1100010;
        hex5 = 7'b0100100;
        if(!i_lid && !i_cancel && soak_done)
          NS <= WASH;
          else
           NS <= PS;
         end
   WASH : begin
    //display WASh
      hex0 = 7'b1111111;
      hex1 = 7'b1111111;
      hex2 = 7'b1101000;
      hex3 = 7'b0100100;
      hex4 = 7'b0001000;
      hex5 = 7'b1100001;
      if(!i_lid && !i_cancel && wash_done)
            NS <= RINSE;
           else
          NS <= PS;
   end
 RINSE : begin
    //display rInSE
        hex0 = 7'b1111111;
        hex1 = 7'b0110000;
        hex2 = 7'b0100100;
```

```
                hex3 = 7'b1101010;
                hex4 = 7'b1111001;
                hex5 = 7'b1111010;
            if(!i_lid && !i_cancel && rinse_done)
                NS <= SPIN;
                else
                NS <= PS;
            end
        SPIN : begin
            //display SPIn
                hex0 = 7'b1111111;
                hex1 = 7'b1111111;
                hex2 = 7'b1101010;
                hex3 = 7'b1111001;
                hex4 = 7'b0011000;
                hex5 = 7'b0100100;
            if(!i_lid && !i_cancel && spin_done)
                NS <= IDLE;
                else
                NS <= PS;
            end
        default :  NS <= IDLE;
    endcase
end
```

```
//--------- output logic ------------
assign o_idle  = (PS == IDLE);
assign o_ready = (PS == READY);
assign o_soak  = (PS == SOAK);
assign o_wash  = (PS == WASH);
assign o_rinse = (PS == RINSE);
assign o_spin  = (PS == SPIN);
assign o_waterinlet = (PS == SOAK) || (PS == WASH) || (PS == RINSE) ;
assign o_coinreturn = (PS == READY)         && (i_cancel);
assign o_done = (PS == SPIN) && (spin_done);


endmodule
```

Chapter 6: Refrences

https://www.mouser.in/new/terasic-technologies/terasic-de10-lite-board/

https://github.com/mnmhdanas/Automatic-washing-machine