J-Component - Report Format

EEE-4019

Title Page

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Implementation of Control system for Automatic Washing Machine with FPGA Verilog

TEAM MEMBERS

DIBYAN GOSWAMI – 20BEE0059

ROSHAN KOVILAPATI – 20BEE0057

Under the Guidance

Dr.S. Balamurugan sir

Associate Professor

School of Electrical Engineering

VIT, Vellore

Winter Semester 2022

INTRODUCTION:

Washing machines are an integral part of our houses. They are important domestic appliances that save a lot of human labour. Washing machines were invented to reduce human effort and keeping this in mind, our manual washing machines are being replaced by automatic washing machines. Automatic washing machines allow us to wash our clothes with least possible human effort: pressing a button. In this project, we aim to design a control system for an automatic washing machine that will help it decide its control flow and execute its actions based on this control flow.

Problem Identification:

With progressive industrialization, many new jobs have been created. With rise of employment, people have started earning more but at the same time have much lesser time to invest in domestic labour like washing clothes.

Automatic washing machines ease our labour by fully automating the cleaning process. However, we need to make the washing machine learn what to do and when in order to make it "smart" or automatic. Automatic washing machines require a control flow to automate their process. This control flow is a state diagram that represents several states of the washing machines. Corresponding to a particular set of inputs, the washing machine will move from one state to the other giving us a flowchart or state diagram.

Objective:

To implement a Control System that controls the fully automatic washing machine's workflow using an FPGA with Verilog HDL as the language of instruction.

LiTERATURE SURVEY:

https://ieeexplore.ieee.org/abstract/document/7406105: This given paper explores the implementation of a washing machine control system to control the wash time, wash strength, temperature, water level and detergent amount. However, it does not organize the workflow or state diagram in order to schedule these tasks. Our project aims to implement this work flow for the automatic washing machine so that the machine can decide which task to complete and when, for example, checking if door is closed, filling the water, adding detergent, etc.

METHODOLOGY:

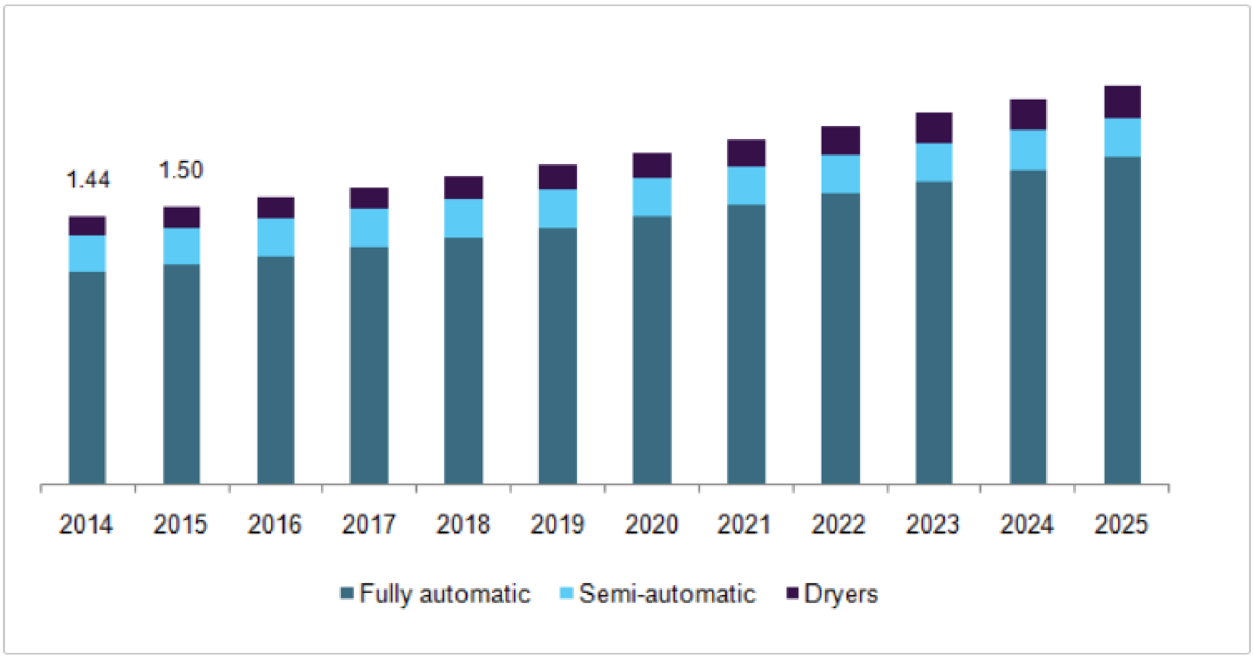
Various real life scenarios can be represented by Finite State Machines like control system of an automatic washing machine. We will add stages of our automatic washing machine as, processes like the closed, or fill water, or add detergent cycle, drain & spin as various states that can be implemented as State machine. We will work how the actual working will done with of all of these stages mentioned above. We will write verilog code for it and testbench , we will debug and we can observe in Xilinx Vivado.

Outcome Expected:

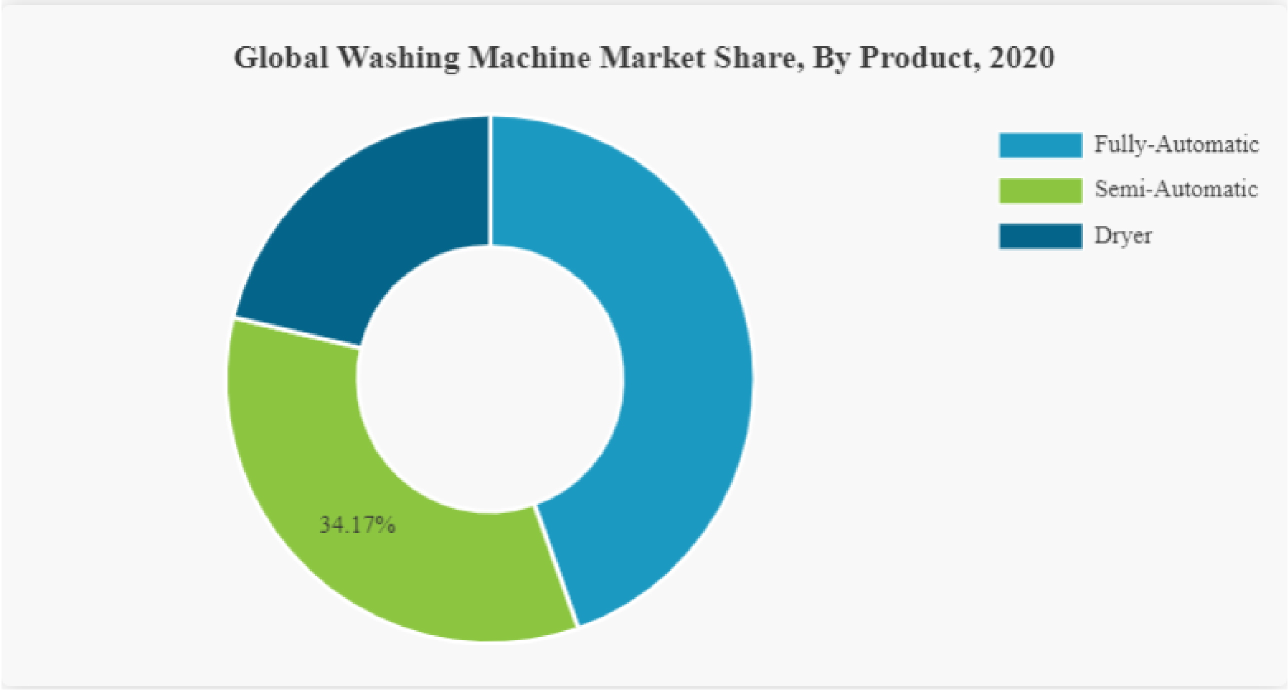
We are expecting as we will include of 6 stages(approx) like Door, Water fill, Detergent, Cycle, Drain & Spin. And also we will add algorithm that has respective max input like the water fill, detergent. We will add registers as per transitions. The stages will be attained one by one in process, if any of them haven't achieved, then it will remain current state. We will expect the same simulation result as the transition between state.

Data Collection:

Germany washing machine market, by product, 2014 - 2025 (USD Billion)



Source: Grand View Research

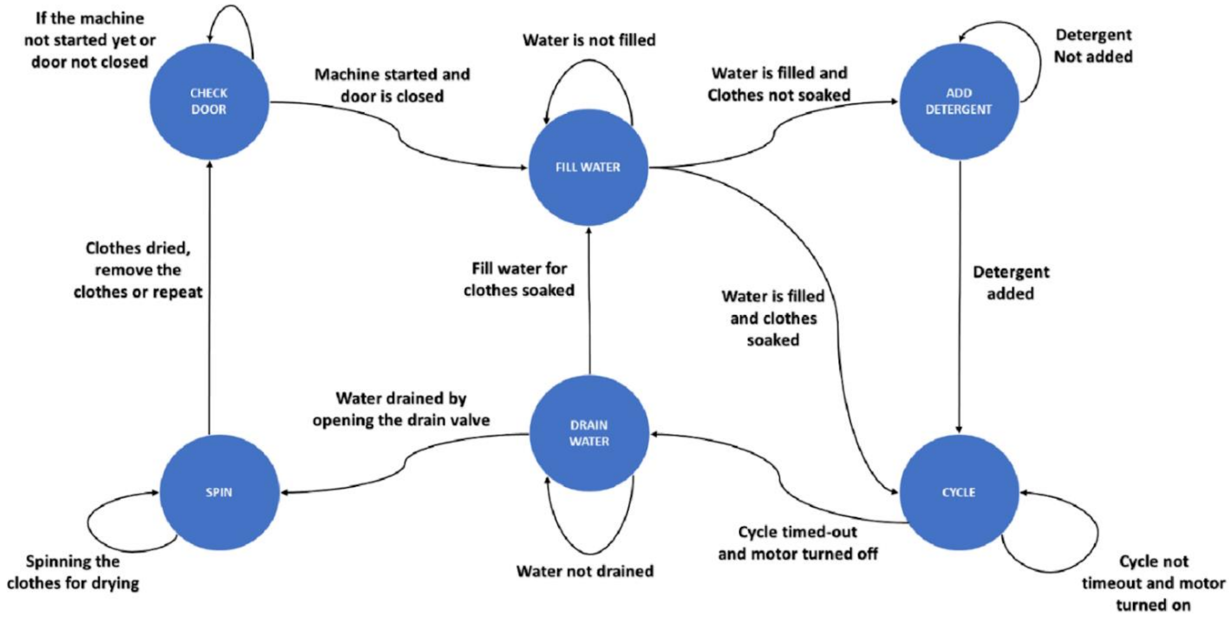


Fortune Business Insights

Data Analysis:

Fully-automatic washing machines are seeing a great surge in usage in recent years. The global washing machine market size was USD 52.8 bn in the year 2020. Of which nearly USD 23.76 bnis attributed to automatic washing machines. Thus, automatic washing machines are presently dominating the global demand. The global washing machine market size is expected to rise to USD 390.6 during the years 2021-2025

Algorithm development:


If the machine not started yet or door not closed
CHECK DOOR
Machine started and door is closed
Water is not filled
FILL WATER
Water is filled and Clothes not soaked
ADD DETERGENT
Detergent Not added
Clothes dried, remove the clothes or repeat
Fill water for clothes soaked
Water is filled and clothes soaked
Detergent added
Water drained by opening the drain valve
DRAIN WATER
SPIN
CYCLE
Spinning the clothes for drying
Water not drained
Cycle timed-out and motor turned off
Cycle not timeout and motor turned on

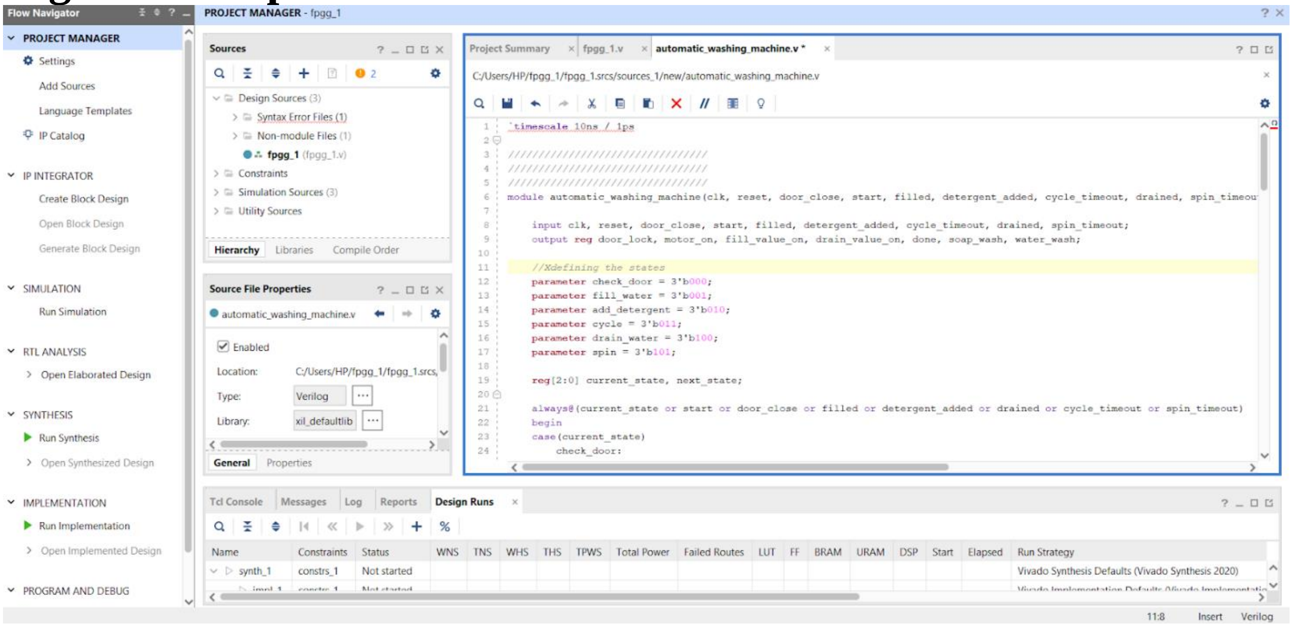
Standards adopted:

Our algorithm implements a finite state machine. In our algorithm, the finite state machine is defined by its present state and transitions. Each state is defined by its value, either one or zero.

Trade off identified:

1. The algorithm simply checks if a task has been completed but doesn’t measure it quantitatively, such as the amount of water or detergent added.
2. No provision to move back in the cycle during the wash, the cycle must complete itself once it has started.

Algorithm implementation:



☑ Enabled

Location: C:/Users/HP/fpgg_1/fpgg_1.srcs,

Type: Verilog

Library: xil_defaultlib

C:/Users/HP/fpgg_1/fpgg_1.srcs/sources_1/new/automatic_washing_machine.v

```
`timescale 10ns / 1ps

////////////////////////////////////
////////////////////////////////////
////////////////////////////////////
module automatic_washing_machine(clk, reset, door_close, start, filled, detergent_added, cycle_timeout, drained, spin_timeou

    input clk, reset, door_close, start, filled, detergent_added, cycle_timeout, drained, spin_timeout;
    output reg door_lock, motor_on, fill_value_on, drain_value_on, done, soap_wash, water_wash;

    //Xdefining the states
    parameter check_door = 3'b000;
    parameter fill_water = 3'b001;
    parameter add_detergent = 3'b010;
    parameter cycle = 3'b011;
    parameter drain_water = 3'b100;
    parameter spin = 3'b101;

    reg[2:0] current_state, next_state;

    always@(current_state or start or door_close or filled or detergent_added or drained or cycle_timeout or spin_timeout)
    begin
    case(current_state)
        check_door:
```

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	Not started															Vivado Synthesis Defaults (Vivado Synthesis 2020)
impl_1	constrs_1	Not started															Vivado Implementation Defaults (Vivado Implementatio

```
`timescale
10ns / 1ps

                ////////////////////////////////////
                ////////////////////////////////////
                ////////////////////////////////////
                module automatic_washing_machine(clk, reset, door_close, start, filled,
                detergent_added, cycle_timeout, drained, spin_timeout, door_lock,
                motor_on, fill_value_on, drain_value_on, done, soap_wash, water_wash);

                        input clk, reset, door_close, start, filled, detergent_added,
                cycle_timeout, drained, spin_timeout;
                        output reg door_lock, motor_on, fill_value_on, drain_value_on,
                done, soap_wash, water_wash;

                        //defining the states
                        parameter check_door = 3'b000;
                        parameter fill_water = 3'b001;
                        parameter add_detergent = 3'b010;
                        parameter cycle = 3'b011;
                        parameter drain_water = 3'b100;
                        parameter spin = 3'b101;

                        reg[2:0] current_state, next_state;

                        always@(current_state or start or door_close or filled or
                detergent_added or drained or cycle_timeout or spin_timeout)
                        begin
                        case(current_state)
```

```
check_door:
    if(start==1 && door_close==1)
    begin
        next_state = fill_water;
        motor_on = 0;
        fill_value_on = 0;
        drain_value_on = 0;
        door_lock = 1;
        soap_wash = 0;
        water_wash = 0;
        done = 0;
    end
    else
    begin
        next_state = current_state;
        motor_on = 0;
        fill_value_on = 0;
        drain_value_on = 0;
        door_lock = 0;
        soap_wash = 0;
        water_wash = 0;
        done = 0;
    end

    fill_water:
    if (filled==1)
    begin
        if(soap_wash == 0)
        begin
            next_state = add_detergent;
            motor_on = 0;
            fill_value_on = 0;
            drain_value_on = 0;
            door_lock = 1;
            soap_wash = 1;
            water_wash = 0;
            done = 0;
        end
        else
        begin
            next_state = cycle;
            motor_on = 0;
            fill_value_on = 0;
            drain_value_on = 0;
            door_lock = 1;
            soap_wash = 1;
            water_wash = 1;
            done = 0;
        end
end
else
begin
        next_state = current_state;
        motor_on = 0;
        fill_value_on = 1;
        drain_value_on = 0;
        door_lock = 1;
        done = 0;
end
add_detergent:
if(detergent_added==1)
begin
        next_state = cycle;
        motor_on = 0;
        fill_value_on = 0;
        drain_value_on = 0;
        door_lock = 1;
        soap_wash = 1;
        done = 0;
end
else
begin
        next_state = current_state;
        motor_on = 0;
        fill_value_on = 0;
        drain_value_on = 0;
        door_lock = 1;
        soap_wash = 1;
        water_wash = 0;
        done = 0;
end
cycle:
if(cycle_timeout == 1)
begin
        next_state = drain_water;
        motor_on = 0;
        fill_value_on = 0;
        drain_value_on = 0;
        door_lock = 1;
        //soap_wash = 1;
        done = 0;
end
else
begin
        next_state = current_state;
        motor_on = 1;
```

```
       fill_value_on = 0;
       drain_value_on = 0;
       door_lock = 1;
       //soap_wash = 1;
       done = 0;
end
drain_water:
 if(drained==1)
 begin
       if(water_wash==0)
       begin
             next_state = fill_water;
             motor_on = 0;
             fill_value_on = 0;
             drain_value_on = 0;
             door_lock = 1;
             soap_wash = 1;
             //water_wash = 1;
             done = 0;
       end
       else
       begin
       next_state = spin;
             motor_on = 0;
             fill_value_on = 0;
             drain_value_on = 0;
             door_lock = 1;
             soap_wash = 1;
             water_wash = 1;
             done = 0;
       end
end
else
begin
       next_state = current_state;
       motor_on = 0;
       fill_value_on = 0;
       drain_value_on = 1;
       door_lock = 1;
       soap_wash = 1;
       //water_wash = 1;
       done = 0;
end
spin:
if(spin_timeout==1)
begin
       next_state = door_close;
       motor_on = 0;
```

```
                fill_value_on = 0;
                drain_value_on = 0;
                door_lock = 1;
                soap_wash = 1;
                water_wash = 1;
                done = 1;
            end
            else
            begin
                next_state = current_state;
                motor_on = 0;
                fill_value_on = 0;
                drain_value_on = 1;
                door_lock = 1;
                soap_wash = 1;
                water_wash = 1;
                done = 0;
            end
            default:
                next_state = check_door;

            endcase
    end

    always@(posedge clk or negedge reset)
    begin
        if(reset)
        begin
            current_state<=3'b000;
        end
        else
        begin
            current_state<=next_state;
        end
    end

endmodule
```

Problems faced:

1. The process of developing an algorithm was time-consuming.

It was difficult to get acquainted with Xilinx coding environment.
It took some practice to get used to the coding style.
The only provision to check code is by running testbench and no other intermediate method.

Individual contribution:

Roshan Kovilapati- Designed the flow-chart for the finite state machine. Worked on the algorithm and pseudocode. Researched on the coding perspective and coded one half.

Dibyan Goswami- Collected and analyzed the data. Worked on the algorithm and pseudocode. Researched on the coding syntax and worked on one half of the code.

MILESTONE & TIMEFRAME:

Ideation- 06-02-22, We fixed with our idea.
Algorithm- 26-02-22, We will come up with exact algorithm.
Verilog coding- 12-03-22, We will finish first V code.
DOCUMENTATION- 27-04-22,We will finish official documentation
Testing- 10-04-22,We will finish testing our model with Testbench & .V

REFERENCES:

https://ieeexplore.ieee.org/abstract/document/7406105