TECHNISCHE HOCHSCHULE DEGGENDORF

FPGA-Programming

Summer Semester 2020

Group 4 -Project report

Washing Machine

Abhi Akbari (00802927)

Constantin Kriegl (00627494)

Kishankumar Vaidya (00617830)

Mateus Teller-Nebias (00802324)

Muhammad Ali (00732065)

Submitted To:

Prof. Dr. Martin Schramm

26.07.2020

TECHNISCHE HOCHSCHULE DEGGENDORF

Report Washing Machine

Clarification of the independent work

Hereby, we assure that everyone in the team spent roughly the same effort and resources on the project. Furthermore, we assure that we did not copy work without mentioning and adhered to academic writing guidelines in the report.

Team members: Abhi Akbari (00802927)
Constantin Kriegl (00627494)
Kishankumar Vaidya (00617830)
Mateus Telles Nebias (00802324)
Muhammad Ali (00732065)

Examiner: **Prof. Dr. Martin Schramm**

Submission date: 26.07.2020

Contents

1 Introduction

We decided to implement a washing machine as a project. The project will have files for the main program, RAM, 2to4Decoder, hex7seg, and a modulo timer. Through press buttons, the user will decide which of the four washing programs he wants to select. He can select light fabric, cotton, heavy fabric and hospital cloth.

The selected program will be displayed on the 7 Segment Display. The **LEDs (LEDR)** will show in which state the washing machine is now. The state machine will operate according to the chosen washing program. The finite state machine has the states **S_SBY, S_INITIAL, S_EMPTY1, S_FILLING2, S_DRYER, S_ULTRA, S_EMPTY2, S_HEAVY, S_LIGHT, S_FILLING1, S_RINSE**. Furthermore, the selected washing programs will be shown on the 7-segment displays. The selection of the washing programs will be done with the keys.

The circuit also has counters that make sure the right time for each washing program. We also use a memory to store the current running program in it.

The FPGA Board we used was the Cyclon 5 FPGA, which you can see in **Figure 1.**

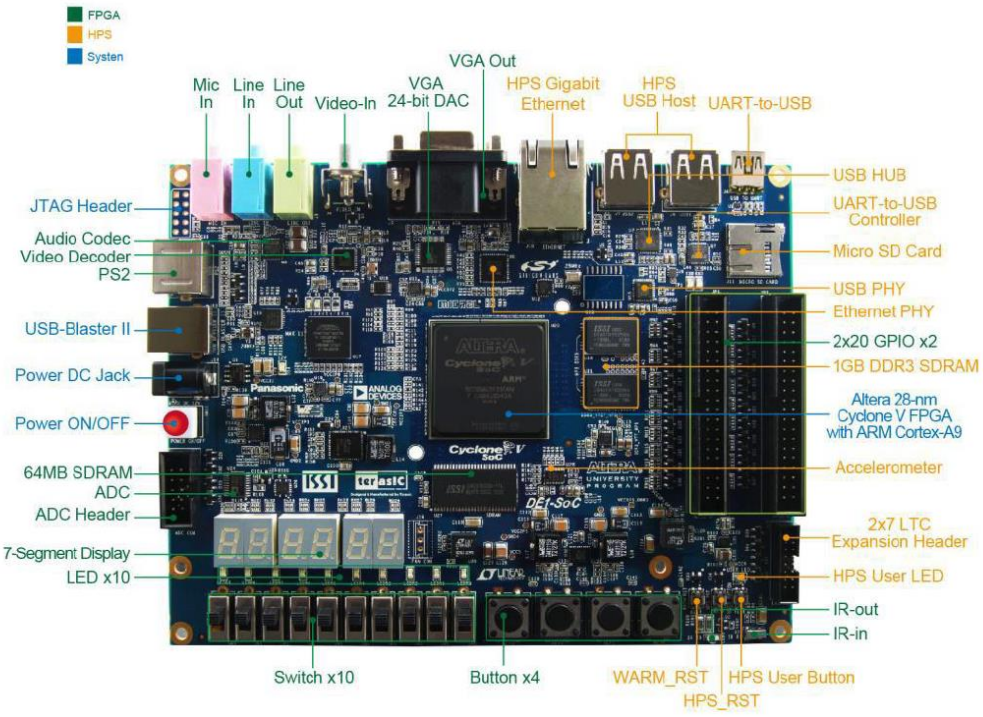


Figure 1

2 Implementation

Configuration:

KEY[1-0]	Chose Operation Mode
KEY[2]	Option
KEY[3]	Write Signal
SW[8]	Address Signal
LEDR[9-0]	Show which state is running
HEX[4-1]	Show operating modes
CLOCK_50	For clock signal
Selection Signal	Data in signal
Open (no data out)	Data out signal

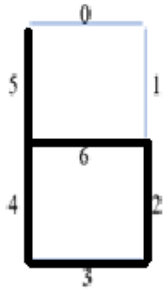
2.1 Operating modes (KEYS [1-0]):

The selection of the modes is done with the keys and then transferred with 2to4decoder to the selection signal.

Key[1]	Key[0]	Selection signal	Mode
0	0	0001	Light fabric
0	1	0010	Cotton
1	0	0100	Heavy fabric
1	1	1000	Hospital cloth

In the code we used the short forms: LIFb, COtn, HEFb, HPCL. So we had to define some letters on the 7 segment display, which you can see below. This was done for example with: constant c_B: std_logic_vector(0 to 6) := "1100000";

On the example for letter b you can see that 0 and 1 are both 1. This means that they are not visible, and only 2, 3, 4, 5 and 6 are displayed. So “b” appears.



The Letters are shown in the hex7Display as shown in following table:

	HEX5	HEX4	HEX3	HEX2	HEX1	HEX0
reset	-	-	-	-	-	-
Light fabric	-	L	I	F	b	-
cotton	-	C	O	t	n	-
Heavy fabric	-	H	E	F	b	-
Hospital cloth	-	H	P	C	L	-

2.2 State description

The 10 LEDs on the FPGA board show the user in which state the washing machine is. This is useful for to get a better estimation when the washing machine is done.

LEDR(9 to 0)	All On	s-by
LEDR(9)	On	Initial
LEDR(8)	On	s-filling(1)
LEDR(7)	On	s-rinse
LEDR(6)	On	s-empty(1)
LEDR(5)	On	s-filling(2)
LEDR(4)	On	s-light
LEDR(3)	On	s-heavy
LEDR(2)	On	s-empty(2)
LEDR(1)	On	s-dryer

LEDR(0)	On	s-ultra
LEDR(9 to 0)	All off	otherwise

2.3 Diagrams for Overview

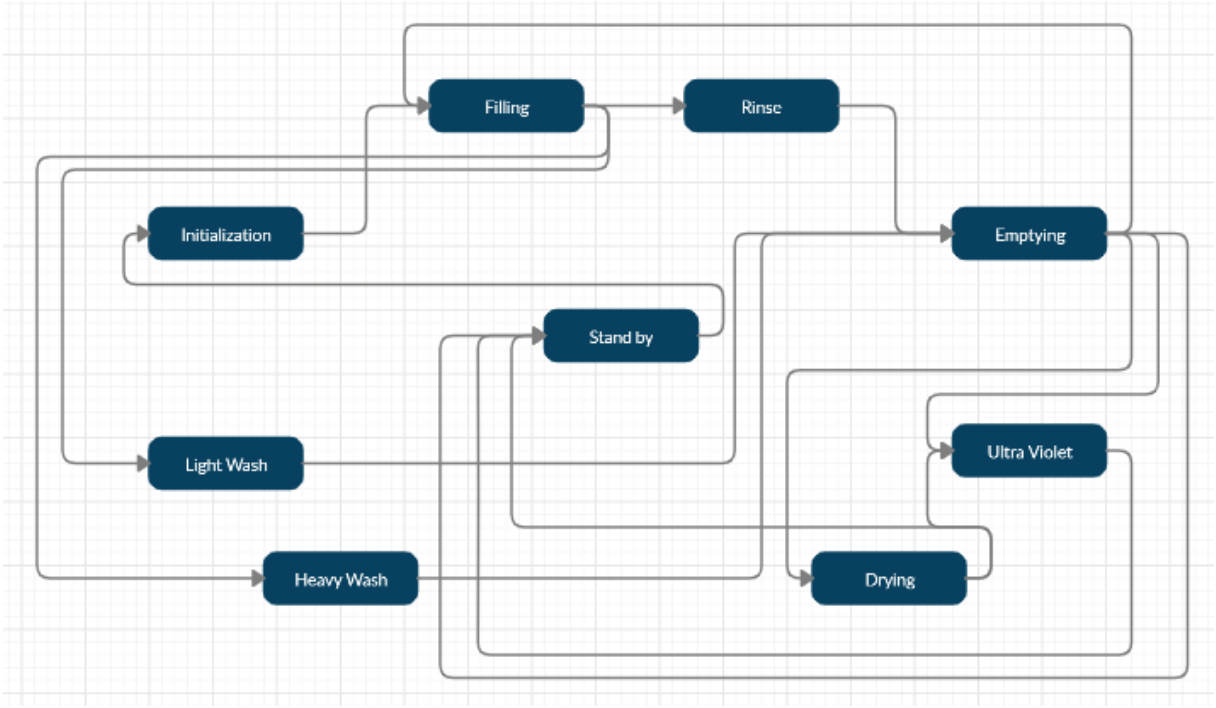


Figure 2: state diagram of washing machine

In Figure 2 the state diagram of the washing machine is shown. The placing was made to make the diagram more readable. We placed stand by in the middle because it's the beginning and the ending state of each cycle. Are the remaining states were placed trying to imitate a clock's work?

To facilitate the understanding of the machine's operation, all arrows enter from the left and exit from the right. Some blocks have more than one arrow going in or out. This indicates that it can be triggered by different modes in the case of multiple inputs or that they can result in different modes in the case of multiple outputs. For example, it is possible to enter Stand-by mode by exiting three other modes (drying, emptying or ultraviolet).

The generated state machine in **Figure 1 Figure 3** shows the correct work of the planned state diagram.

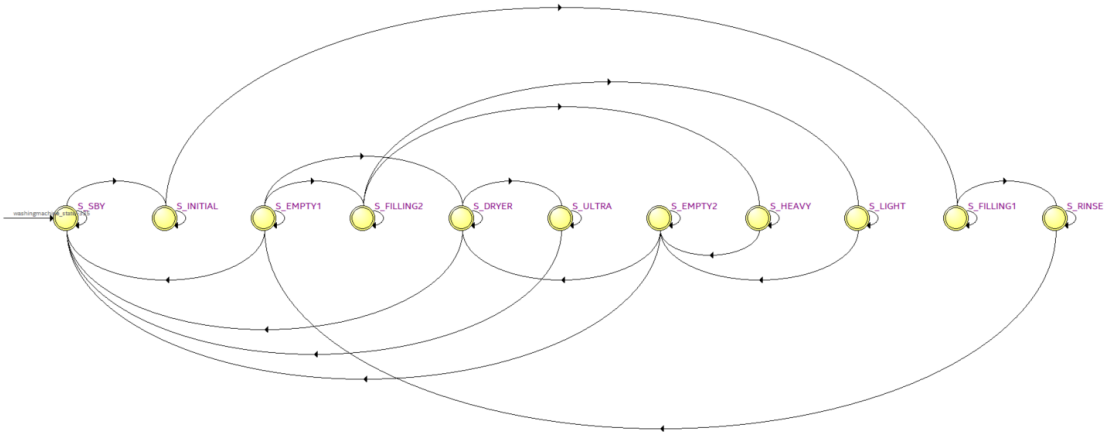


Figure 3: State machine

It works appropriate to the operation cycles. As you can see in the table below, we have defined the time for each state. For this we wrote the clock file.

Operation cycle:

mode									**Option(KEY[2])**		
[0][0] Light fabric	s-by	(2 s) Initial	(5 s) Filling(1)	(5 m) Rinse	(5 s) Empty(1)				(15m) Dryer		s-by
[0][1] cotton	s-by	Initial	Filling(1)	Rinse	Empty(1)	(5 s) Filling(2)	(1h20 m) Light	(5 s) Empty(2)	Dryer		s-by
[1][0] heavy fabric	s-by	Initial	Filling(1)	Rinse	Empty(1)	Filling(2)	(1h40 m) Heavy	Empty(2)	Dryer		s-by
[1][1] hospital cloth	s-by	Initial	Filling(1)	Rinse	Empty(1)	Filling(2)	Heavy	Empty(2)	Dryer	(1h40 m) Ultra	s-by

In **Figure 4** the block diagram for the washing machine is shown. It demands the file e_my_wasingmachine.vhd.

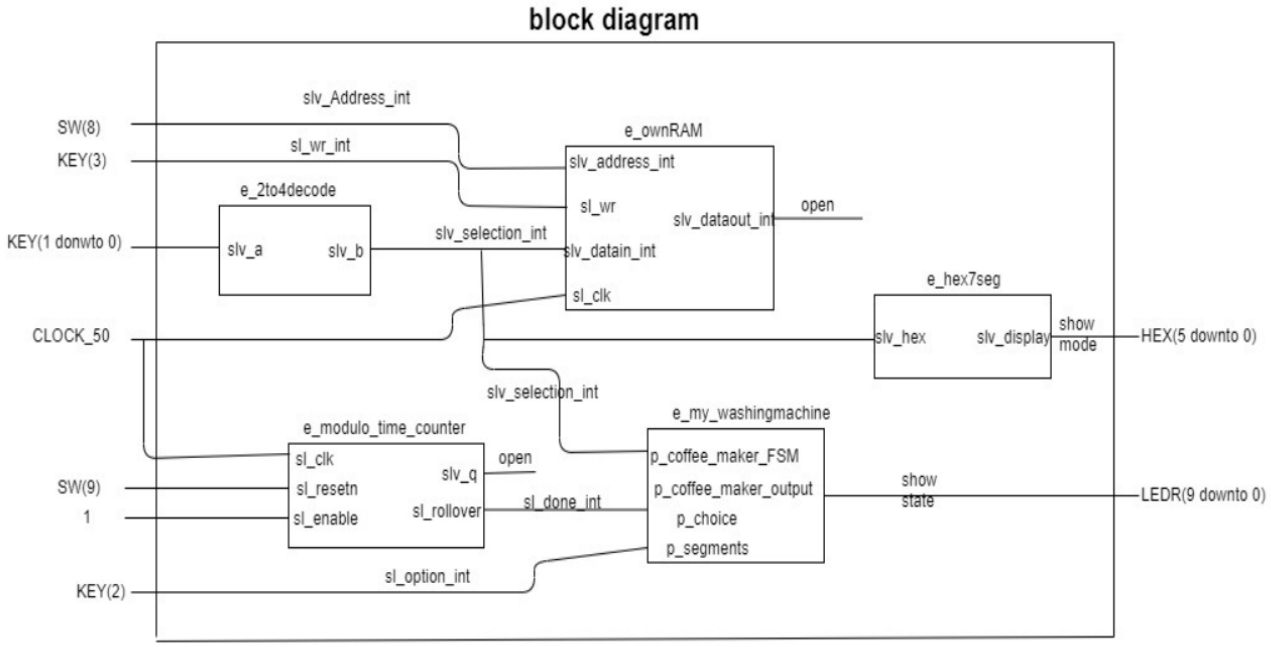


Figure 4: block diagram from washing machine

2.4 Memory

A problem we had to deal with was to find a useful function for the memory. Normally a washing machine has no memory, as you can see in the block diagram in **Figure 5**.

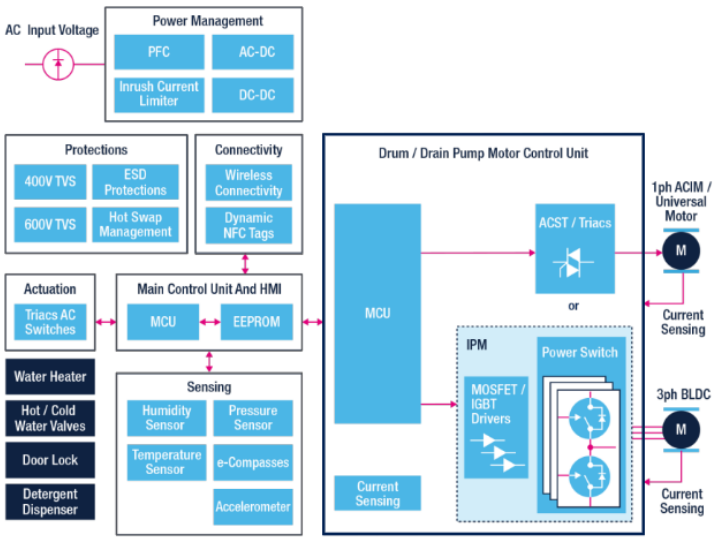


Figure 5: Block Diagram [https://www.st.com/en/applications/home-and-professional-appliances/washing-machines.html]

But anyway, we added a so-called Random-Access Memory (RAM) which commonly contains the programs or parts of the program that are currently running, and the data required. Our RAM stores the values of the selection signal in the memory every time.

2.5 Testing the functionality

To test the functionality, we created a testbench. To do this we called the Keys for the washing programs in the vht file. After that the files were compiled in ModelSim. There we noticed that an error is in the code for the 2to4decode file. The error was that we forgot to put the line:
when others => slv_b <= "0000"; in the code.
It was possible to compile all files error free, but at first it was not possible to open a do file to pass them into it. The following error message was shown:
wrong # args: should be "load file ?pkg? ?interp?"
With research we could solve this problem. We changed the location of the program from C:\ Programme (x86) to C:\. With these changes made it was now possible to create the do file and put the compiled files in it. After that we saved the do script with the name run.do. Now the testbench file was selected in the library to locate the testbench entity with a right click → simulate. The view of ModelSim changes with performing this step. Now the simulation folder is visible. It is important to add the command vsim work.e_my_wasingmachine_vhd_tst to the do script.
In the next step we can execute the script with typing the command: do run.do in the transcript. The advantage of this is a fast compilation of the complete circuit. It is much faster than a full compilation.
Following step was to set up a wave with putting in some signals. So i1 was selected, which is our design device under test. It was added in the wave window. Again, a file was created, and the line do wave.do was added to the run.do script because every time we do that the wave will be loaded and the signals will be added to this wave. The last line we need now is run -all and with typing in do run.do in the transcript we can start the complete compilation with the command run -all. A screenshot of the wave window is shown in following **Figure 6.**

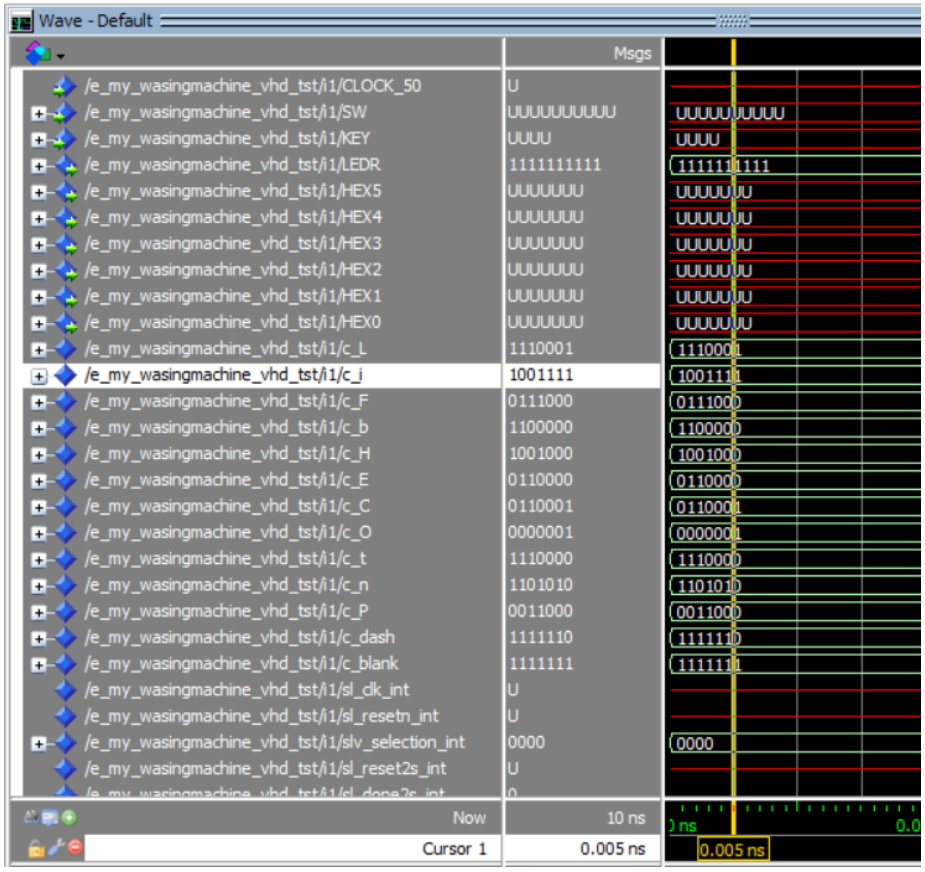


Figure 6: Wave window in ModelSim

It shows that the wave gets a lot of undefined values because we are in the standby state. All LEDRs are 1 and this was the definition of the idle state and by scrolling down you can see S SBY on the wave. So, the correct functionality of the standby state is proven. But, the states of all washing programs should be displayed by scrolling to the right in the wave window.

Maybe the reason for this is that the washing programs are selected with two keys and the first program should run when no key is pressed but when no key is pressed it could be possible that it stays in the standby state. Maybe it is also not possible to reset the keys to 0 because if two keys are 0 the first washing program is selected.

3 Summary

Summed up the project was good for developing a good knowledge in VHDL and digital technology. We had to develop and implement our own project idea. For this it was a challenge to involve everybody in the project because we were five people in a group.

Unfortunately, we couldn't test our implementation on a real FPGA board, but in our opinion, it isn't so important for the learning effect, because we know how the FPGA board look like. It would only be interesting to see if the programmed idea works.