**ELEC 204 Digital Design Lab Report**

Lab 5

Name: Walid Baroudi

Date: 12/15/2019

I worked on this lab with my partner Abdul Rahman Hamadeh.

1. **Introduction and objectives**

In this lab we had to design a state diagram of a washing machine, implement its corresponding state diagram on FPGA using a modular design and experimentally demonstrate the operation of the state machine.

1. **Methods**

Inputs:

START which is a 1-bit input that changes the state from IDLE to WASH state.

RESET which is a 1-bit input that changes the state from any state to IDLE state.

Outputs:

S0 and S1 is a 2-bit output that is responsible for showing the current state.

DEC a 1-bit output that is 1 in every state other than IDLE state.

ZERO which is a 1-bit output that changes between 1 and 0. It becomes 1 during the state change and 0 otherwise.

The VHDL code had to simulate the states of a washing machine. The simulated washing machine had 4 states: WASH, SPIN, RINSE and IDLE.

How the code works:

First of all, we had to implement a clock divider in order to get the counter for each state. When the code is run, the code always starts with the IDLE state. After the START button is clicked the state changes from IDLE to WASH and stays in WASH for 4 seconds, then moves to SPIN, stays there for 3 second and, lastly, The RINSE state which lasts for 2 seconds and after that it returns back to the IDLE state.

There is a RESET button that will change the state, from any state, to the IDLE state when clicked.

The COUNTER in this table is the inner counter of each state. When the COUNTER reaches 0 the state changes to the next state.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| CURRENT STATE | | | | NEXT STATE | | | |
| START | RESET | COUNTER | STATE | ZERO | S1 | S0 | STATE |
| 0 | 0 | 0 | WASH | 1 | 0 | 0 | SPIN |
| 0 | 0 | 0 | SPIN | 1 | 0 | 1 | RINSE |
| 0 | 0 | 0 | RINSE | 1 | 1 | 0 | IDLE |
| 1 | 0 | 0 | IDLE | 1 | 1 | 1 | WASH |
| 0 | 1 | 0 | WASH | 1 | 1 | 1 | IDLE |
| 0 | 1 | 0 | SPIN | 1 | 1 | 1 | IDLE |
| 0 | 1 | 0 | RINSE | 1 | 1 | 1 | IDLE |
| 0 | 1 | 0 | IDLE | 1 | 1 | 1 | IDLE |

S1 and S0 in the following tables are outputs.

The WASH state:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CURRENT STATE | | | | | NEXT STATE | | | | | | |
| DEC | RESET | Inner counter (3-bits) | | | ZERO | Inner counter (3-bits) | | | S1 | S0 | DEC |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | NEXT STATE | | | 0 | 0 | 1 |
| 1 | 1 | X | X | X | 1 | IDLE STATE | | | 1 | 1 | 0 |

The SPIN state:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CURRENT STATE | | | | NEXT STATE | | | | | |
| DEC | RESET | Inner counter (2-bits) | | ZERO | Inner counter (2-bits) | | S1 | S0 | DEC |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | NEXT STATE | | 0 | 1 | 1 |
| 1 | 1 | X | X | 1 | IDLE STATE | | 1 | 1 | 0 |

The RINSE state:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CURRENT STATE | | | | NEXT STATE | | | | | |
| DEC | RESET | Inner counter (2-bits) | | ZERO | Inner counter (2-bits) | | S1 | S0 | DEC |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | NEXT STATE | | 1 | 0 | 1 |
| 1 | 1 | X | X | 1 | IDLE STATE | | 1 | 1 | 0 |

The IDLE state:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CURRENT STATE | | NEXT STATE | | | | |
| RESET | START | Next state | S1 | S0 | DEC  0 | ZERO |
| 0 | 1 | WASH STATE | 1 | 1 | 0 | 1 |
| 1 | X | IDLE STATE | 1 | 1 | 0 | 1 |

1. **Problems encountered, errors and warnings resolved**

There were no errors in the final run of the code, however, we faced some minor errors that we were able to fix.

1. **Conclusion**

References

1. Please cite any resource (web site, book, youtube video) you used for this lab.

**Appendix 1. Lab source code**

**Appendix 2. RTL schematics**

**Appendix 3. FPGA Board photos showing working code**

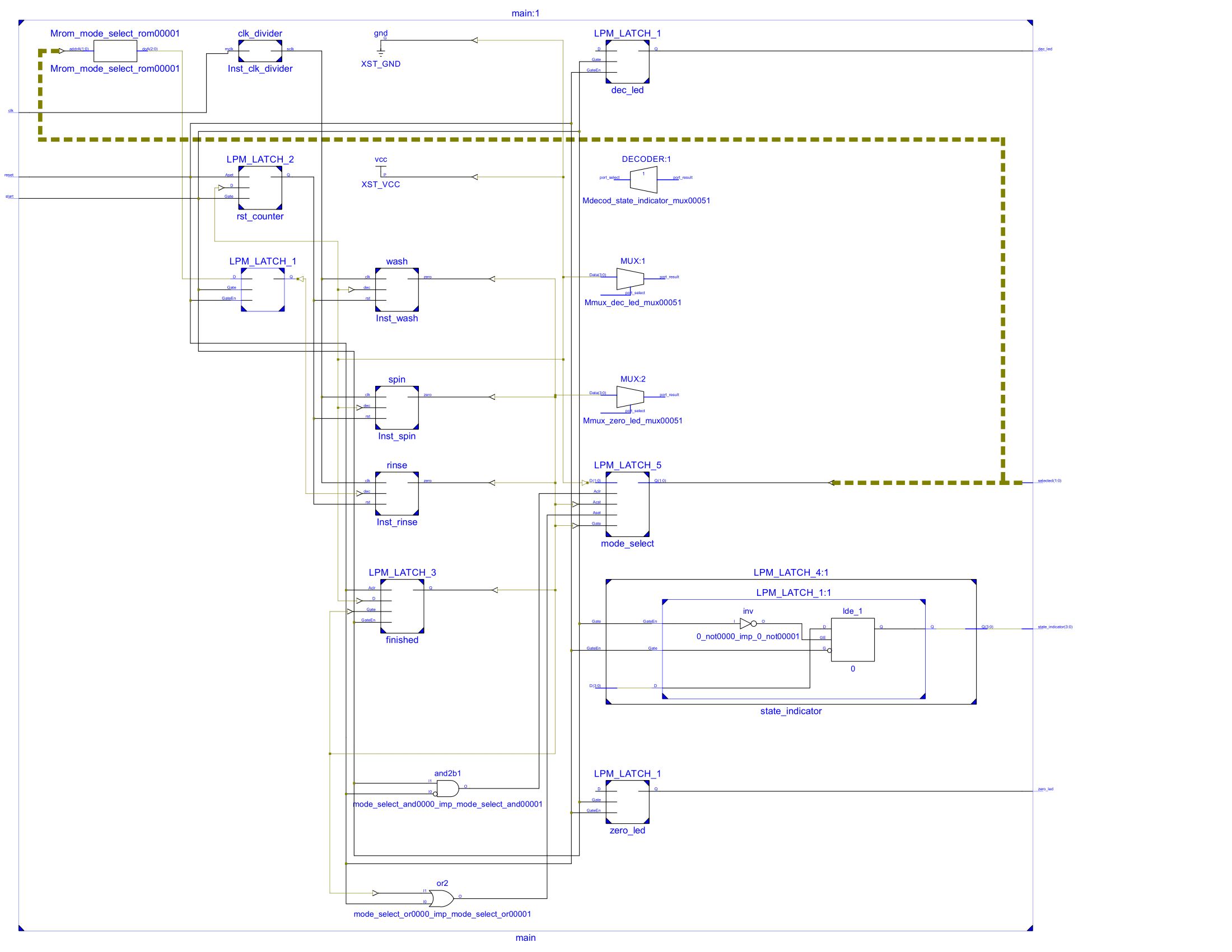
The other states modules are not included because they are exactly the same as the WASH state module

But with different divider value.

|  |  |
| --- | --- |
| **Clock Divider** | **Wash Behavioral** |
| signal slow\_clock: std\_logic;  signal mode\_select: std\_logic\_vector(1 downto 0):="11"; --selected state  signal wash\_dec: std\_logic;  signal wash\_zero: std\_logic:='0';  signal spin\_dec: std\_logic;  signal spin\_zero: std\_logic:='0';  signal rinse\_dec: std\_logic;  signal rinse\_zero: std\_logic:='0';  signal finished: std\_logic:='0'; --to distinguish whether the transition to IDLE was due to finishing or restarting.  signal rst\_counter: std\_logic; --will be passed to state modules as '1' when the the restart button is clicked.  begin  process  begin  if(reset='1') then --selects idle, resets counter, and indicates that the cycle was not completed.  mode\_select <= "11";  finished <='0';  rst\_counter <= '1';  elsif(start='1') then  rst\_counter <= '0';  mode\_select <= "00";  else  if(mode\_select = "00") then --the decrement of the selected mode is 1 while for others is 0;  wash\_dec <= '1';  spin\_dec <= '0';  rinse\_dec <= '0';  dec\_led <= wash\_dec;  zero\_led <= wash\_zero;  state\_indicator <= "0001";  end if;  if(mode\_select = "01") then  wash\_dec <= '0';  spin\_dec <= '1';  rinse\_dec <= '0';  dec\_led <= spin\_dec;  zero\_led <= spin\_zero;  state\_indicator <= "0010";  end if;  if(mode\_select = "10") then  wash\_dec <= '0';  spin\_dec <= '0';  rinse\_dec <= '1';  dec\_led <= '1';  zero\_led <= rinse\_zero;  state\_indicator <= "0100";  end if;  if(mode\_select = "11") then  wash\_dec <= '0';  spin\_dec <= '0';  rinse\_dec <= '0';  dec\_led <= '0';  zero\_led <= finished;  state\_indicator <= "1000";  end if;  if(wash\_zero = '1') then --going to the next state when the current state's counter reaches zero.  mode\_select <= "01";  end if;  if(spin\_zero = '1') then  mode\_select <= "10";  end if;  if(rinse\_zero = '1') then  finished <='1';  mode\_select <= "11";  end if;  end if;  end process;  selected <= mode\_select; | architecture Behavioral of wash is  signal counter: integer:=1;  signal increment: integer:=0;  begin  process  begin  if(dec = '1') then --if block to convert dec from std\_logic to integer  increment <= 1;  else  increment <= 0;  end if;  if(rst = '1') then --rst input to reset counter and zero when the reset button is clicked.  counter <= 1;  zero <= '0';  elsif(rising\_edge(clk)) then --1 sec clock  if(counter = 4) then --counting up to four  zero <= '1';  counter <= 1;  else  zero <= '0';  counter <= counter + increment;  end if;  end if;  end process;  end Behavioral; |

|  |  |
| --- | --- |
| INSTANCES | CLOCK DIVIDER |
| Inst\_clk\_divider: clk\_divider PORT MAP(  mclk => clk,  sclk => slow\_clock  );  Inst\_wash: wash PORT MAP(  clk => slow\_clock,  dec => wash\_dec,  rst => rst\_counter,  zero => wash\_zero  );  Inst\_spin: spin PORT MAP(  clk => slow\_clock,  dec => spin\_dec,  rst => rst\_counter,  zero => spin\_zero  );    Inst\_rinse: rinse PORT MAP(  clk => slow\_clock,  dec => rinse\_dec,  rst => rst\_counter,  zero => rinse\_zero  ); | architecture Behavioral of clk\_divider is --producess and clock with period 1 s (1Hz frequency)  signal temp: std\_logic:='1';  signal counter: integer:=0;  begin  Process\_clk: process(mclk)  begin  if(rising\_edge(mclk)) then  counter <= counter + 1;  if(counter = 50000000) then  counter <= 0;  temp <= (not temp);  end if;  end if;  end process;  sclk <= temp;  end Behavioral; |

|  |  |
| --- | --- |
| INPUTS AND OUTPUTS OF MAIN | UCF |
| entity main is  port(  clk: in std\_logic;  start: in std\_logic;  reset: in std\_logic;  zero\_led: out std\_logic;  dec\_led: out std\_logic;  state\_indicator: out std\_logic\_vector(3 downto 0); --the 4 leds indicating the states  selected: out std\_logic\_vector(1 downto 0)  );  end main; | NET "clk" LOC = P40;  NET "dec\_led" LOC = P13;  NET "zero\_led" LOC = P16;  NET "selected[0]" LOC = P6;  NET "selected[1]" LOC = P3;  NET "state\_indicator[0]" LOC = P86;  NET "state\_indicator[1]" LOC = P84;  NET "state\_indicator[2]" LOC = P83;  NET "state\_indicator[3]" LOC = P77;  NET "start" LOC = P37;  NET "reset" LOC = P32;  NET "reset" CLOCK\_DEDICATED\_ROUTE = FALSE; |

****