ELEC 204 / FALL 18

LAB PROJECT

WASHING MACHINE

## Hasan Can Aslan

Description

The washing machine has 4 states of operation: *WASH*, *SPIN*, *RINSE, IDLE.* And has 5 states of mode: *SHEET, WHITE, COTTON, COLOR, RAPID.* The washing machine synchronous circuit with clock signal CLK. There are two external 1-bit inputs: *START* and *RESET* buttons. The external output is a 2-bit signal S1S0 that shows which state the washing machine is in (*WASH*, *SPIN*, *RINSE, IDLE are 00, 01, 10, 11, respectively*). *WASH*, *SPIN* and *RINSE* states use a down counter. The counters all use a 1-bit internal variable *DEC*. The counters synchronously decrement once each second for *DEC* = 1, It has a single internal output, *ZERO*, which is 1 whenever the counter reaches zero and is 0 otherwise.  
The counters for *WASH, SPIN* and *RINSE* states count for different integers for every states of mode. Next, the counter for *WASH* state starts counting down (S1S0 = 00). As soon as this counter reaches zero, *ZERO* becomes 1, the counter for *WASH* state resets and the system transitions to *SPIN* state. Then, the counter for *SPIN* state (S1S0 = 01) starts counting down. As soon as this counter reaches zero, *ZERO* becomes 1 again, the counter for *SPIN* state resets and the system transitions to *RINSE* state. In this final operational state which is *RINSE* (S1S0 = 10), the counter for *RINSE* state starts counting down. As soon as this counter reaches zero, *ZERO* becomes 1 again, the counter for *SPIN* state resets and the system transitions to *RINSE* state. *RESET* input aborts the entire washing procedure, sets DEC = 0, and the machine resets to the *IDLE* state (S1S0 = 11).

**Codes**

*Clock*

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity clock\_divider is

Port (

MCLK : in STD\_LOGIC;

HUNDREDHZCLOCK : out STD\_LOGIC);

end clock\_divider;

architecture Behavioral of clock\_divider is

SIGNAL COUNTER : STD\_LOGIC\_VECTOR(18 DOWNTO 0) := "0000000000000000000";

begin

CLK\_PROCESS: PROCESS(MCLK)

BEGIN

if (MCLK'EVENT AND MCLK = '1') then

if COUNTER <= "0011110100001001000" then

COUNTER <= COUNTER + 1;

else

COUNTER <= "0000000000000000000";

end if;

end if;

END PROCESS;

HUNDREDHZCLOCK <= '1' WHEN COUNTER = "0001111010000100100" else '0';

end Behavioral;

*Decoder*

LIBRARY IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

ENTITY SevSeg\_Decoder IS

PORT ( INPUT : IN STD\_LOGIC\_VECTOR (4 DOWNTO 0);

SEVSEG\_BUS : OUT STD\_LOGIC\_VECTOR (6 DOWNTO 0));

END SevSeg\_Decoder;

ARCHITECTURE BEHAVIORAL OF SevSeg\_Decoder IS

BEGIN

WITH INPUT SELECT SEVSEG\_BUS <=

"0000001" WHEN "00000", --0

"1001111" WHEN "00001", --1 or I

"0010010" WHEN "00010", --2

"0000110" WHEN "00011", --3

"1001100" WHEN "00100", --4

"0100100" WHEN "00101", --5

"0100000" WHEN "00110", --6

"0001111" WHEN "00111", --7

"0000000" WHEN "01000", --8

"0000100" WHEN "01001", --9

"0001000" WHEN "01010", --A

"1100000" WHEN "01011", --B

"0110001" WHEN "01100", --C

"1000010" WHEN "01101", --D

"0110000" WHEN "01110", --E

"1110001" WHEN "01111", --L

"1100001" WHEN "10000", -- W Part 1

"1000011" WHEN "10001", -- W Part 2

"0001001" WHEN "10010", -- N

"0011000" WHEN "10011", -- P

"0111001" WHEN "10100", -- R

"0100100" WHEN "10101", -- S

"1110000" WHEN "10110", -- T

"1001000" WHEN "10111", -- H

"0000000" WHEN "11000", -- K

"0011000" WHEN "11001", -- P

"1111111" WHEN "11010", -- Blank

"0100100" WHEN "11011", --

"1110110" WHEN "11100", --

"0100001" WHEN "11101", --

"0110000" WHEN "11110", --

"1110001" WHEN "11111", --

"1111111" WHEN OTHERS;

END BEHAVIORAL;

*Main*

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity main is

Port (

Clock : in std\_logic;

start : in std\_logic;

reset : in std\_logic;

leds : out std\_logic\_vector(9 downto 0);

userMode : in std\_logic\_vector(4 downto 0);

SevSegData : out std\_logic\_vector(6 downto 0);

SevSegDriver : out std\_logic\_vector(7 downto 0));

end main;

architecture Behavioral of main is

signal ledOut : std\_logic\_vector(9 downto 0);

signal time\_sn : std\_logic\_vector(4 downto 0);

signal dataSevSeg : std\_logic\_vector(4 downto 0);

signal hz : std\_logic;

signal tempLed : std\_logic\_vector (9 downto 0);

Component washing\_machine

Port (

CK : in std\_logic;

start : in std\_logic;

reset : in std\_logic;

userMode : in std\_logic\_vector(4 downto 0);

state : out std\_logic\_vector(9 downto 0);

time\_sn : out std\_logic\_vector(4 downto 0));

end Component;

Component clock\_divider

PORT(

MCLK : IN std\_logic;

HUNDREDHZCLOCK : OUT std\_logic

);

END Component;

COMPONENT SevSeg\_Decoder

PORT(

INPUT : IN std\_logic\_vector(4 downto 0);

SEVSEG\_BUS : OUT std\_logic\_vector(6 downto 0)

);

END COMPONENT;

COMPONENT SevSeg\_Driver

PORT(

time\_sn : IN std\_logic\_vector(4 downto 0);

leds : IN std\_logic\_vector(9 downto 0);

userMode : IN STD\_LOGIC\_VECTOR(4 downto 0);

CLK : IN std\_logic;

start : IN STD\_LOGIC;

SEV\_SEG\_DATA : OUT std\_logic\_vector(4 downto 0);

SEV\_SEG\_DRIVER : OUT std\_logic\_vector(7 downto 0)

);

END COMPONENT;

begin

clk : clock\_divider Port Map(Clock, hz);

wsh : washing\_machine Port Map(hz, start, reset, userMode, templed, time\_sn);

driver : SevSeg\_Driver PORT MAP(time\_sn, tempLed, userMode, hz, start, dataSevSeg, SevsegDriver);

decoder : SevSeg\_Decoder PORT MAP(dataSevSeg, SevSegData);

leds <= tempLed;

end Behavioral;

*Driver*

library IEEE;

USE IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

USE IEEE.NUMERIC\_STD.ALL;

ENTITY SevSeg\_Driver IS

PORT (

time\_sn : IN STD\_LOGIC\_VECTOR (4 DOWNTO 0);

leds : IN STD\_LOGIC\_VECTOR(9 downto 0);

userMode : IN STD\_LOGIC\_VECTOR(4 downto 0);

CLK : IN STD\_LOGIC;

start : IN STD\_LOGIC;

SEV\_SEG\_DATA : OUT STD\_LOGIC\_VECTOR(4 DOWNTO 0);

SEV\_SEG\_DRIVER : OUT STD\_LOGIC\_VECTOR (7 DOWNTO 0));

END SevSeg\_Driver;

ARCHITECTURE BEHAVIORAL OF SevSeg\_Driver IS

-- Characters

signal A : std\_logic\_vector(4 downto 0);

signal B : std\_logic\_vector(4 downto 0);

signal C : std\_logic\_vector(4 downto 0);

signal D : std\_logic\_vector(4 downto 0);

signal E : std\_logic\_vector(4 downto 0);

signal L : std\_logic\_vector(4 downto 0);

signal W1 : std\_logic\_vector(4 downto 0);

signal W2 : std\_logic\_vector(4 downto 0);

signal N : std\_logic\_vector(4 downto 0);

signal P : std\_logic\_vector(4 downto 0);

signal R : std\_logic\_vector(4 downto 0);

signal S : std\_logic\_vector(4 downto 0);

signal T : std\_logic\_vector(4 downto 0);

signal H : std\_logic\_vector(4 downto 0);

signal K : std\_logic\_vector(4 downto 0);

signal O : std\_logic\_vector(4 downto 0);

signal Blank : std\_logic\_vector(4 downto 0);

SIGNAL COUNTER : STD\_LOGIC\_VECTOR(2 DOWNTO 0) := "000";

BEGIN

--INCREMENT COUNTER

A <= "01010"; --A

B <= "01011"; --B

C <= "01100"; --C

D <= "01101"; --D

E <= "01110"; --E

L <= "01111"; --L

W1 <= "10000"; -- W Part 1

W2 <= "10001"; -- W Part 2

N <= "10010"; -- N

P <= "10011"; -- P

R <= "10100"; -- R

S <= "10101"; -- S

T <= "10110"; -- T

H <= "10111"; -- H

K <= "11000"; -- K

Blank <= "11010"; -- Blank

PROCESS\_CLK : PROCESS(CLK)

BEGIN

if rising\_edge(CLK) then

if COUNTER = "111" then

COUNTER <= "000";

ELSE

COUNTER <= COUNTER + 1;

end if;

end if;

END PROCESS PROCESS\_CLK;

SEV\_SEG\_DATA <=

-- Washing Modes

S WHEN COUNTER = "000" and userMode = "10000" and start = '0' ELSE

H WHEN COUNTER = "001" and userMode = "10000" and start = '0' ELSE

E WHEN COUNTER = "010" and userMode = "10000" and start = '0' ELSE

E WHEN COUNTER = "011" and userMode = "10000" and start = '0' ELSE

T WHEN COUNTER = "100" and userMode = "10000" and start = '0' ELSE

Blank WHEN COUNTER = "101" and userMode = "10000" and start = '0' ELSE

"00001" WHEN COUNTER = "110" and userMode = "10000" and start = '0' ELSE

"00001" WHEN COUNTER = "111" and userMode = "10000" and start = '0' ELSE

W1 WHEN COUNTER = "000" and userMode = "01000" and start = '0' ELSE

W2 WHEN COUNTER = "001" and userMode = "01000" and start = '0' ELSE

H WHEN COUNTER = "010" and userMode = "01000" and start = '0' ELSE

"00001" WHEN COUNTER = "011" and userMode = "01000" and start = '0' ELSE

T WHEN COUNTER = "100" and userMode = "01000" and start = '0' ELSE

E WHEN COUNTER = "101" and userMode = "01000" and start = '0' ELSE

"00001" WHEN COUNTER = "110" and userMode = "01000" and start = '0' ELSE

"00010" WHEN COUNTER = "111" and userMode = "01000" and start = '0' ELSE

C WHEN COUNTER = "000" and userMode = "00100" and start = '0' ELSE

"00000" WHEN COUNTER = "001" and userMode = "00100" and start = '0' ELSE

T WHEN COUNTER = "010" and userMode = "00100" and start = '0' ELSE

T WHEN COUNTER = "011" and userMode = "00100" and start = '0' ELSE

"00000" WHEN COUNTER = "100" and userMode = "00100" and start = '0' ELSE

N WHEN COUNTER = "101" and userMode = "00100" and start = '0' ELSE

"00001" WHEN COUNTER = "110" and userMode = "00100" and start = '0' ELSE

"00101" WHEN COUNTER = "111" and userMode = "00100" and start = '0' ELSE

C WHEN COUNTER = "000" and userMode = "00010" and start = '0' ELSE

"00000" WHEN COUNTER = "001" and userMode = "00010" and start = '0' ELSE

L WHEN COUNTER = "010" and userMode = "00010" and start = '0' ELSE

"00000" WHEN COUNTER = "011" and userMode = "00010" and start = '0' ELSE

R WHEN COUNTER = "100" and userMode = "00010" and start = '0' ELSE

Blank WHEN COUNTER = "101" and userMode = "00010" and start = '0' ELSE

"00001" WHEN COUNTER = "110" and userMode = "00010" and start = '0' ELSE

"00000" WHEN COUNTER = "111" and userMode = "00010" and start = '0' ELSE

R WHEN COUNTER = "000" and userMode = "00001" and start = '0' ELSE

A WHEN COUNTER = "001" and userMode = "00001" and start = '0' ELSE

P WHEN COUNTER = "010" and userMode = "00001" and start = '0' ELSE

"00001" WHEN COUNTER = "011" and userMode = "00001" and start = '0' ELSE

D WHEN COUNTER = "100" and userMode = "00001" and start = '0' ELSE

Blank WHEN COUNTER = "101" and userMode = "00001" and start = '0' ELSE

Blank WHEN COUNTER = "110" and userMode = "00001" and start = '0' ELSE

"00110" WHEN COUNTER = "111" and userMode = "00001" and start = '0' ELSE

-- States of America -\_-

D WHEN COUNTER = "000" and leds = "1000000011" and start = '1' ELSE

"00000" WHEN COUNTER = "001" and leds = "1000000011" and start = '1' ELSE

N WHEN COUNTER = "010" and leds = "1000000011" and start = '1' ELSE

E WHEN COUNTER = "011" and leds = "1000000011" and start = '1' ELSE

D WHEN COUNTER = "100" and leds = "1000000011" and start = '1' ELSE

"00000" WHEN COUNTER = "101" and leds = "1000000011" and start = '1' ELSE

N WHEN COUNTER = "110" and leds = "1000000011" and start = '1' ELSE

E WHEN COUNTER = "111" and leds = "1000000011" and start = '1' ELSE

W1 WHEN COUNTER = "000" and leds = "0100000000" and start = '1' ELSE

W2 WHEN COUNTER = "001" and leds = "0100000000" and start = '1' ELSE

A WHEN COUNTER = "010" and leds = "0100000000" and start = '1' ELSE

S WHEN COUNTER = "011" and leds = "0100000000" and start = '1' ELSE

H WHEN COUNTER = "100" and leds = "0100000000" and start = '1' ELSE

Blank WHEN COUNTER = "101" and leds = "0100000000" and start = '1' ELSE

Blank WHEN COUNTER = "110" and leds = "0100000000" and start = '1' ELSE

time\_sn WHEN COUNTER = "111" and leds = "0100000000" and start = '1' ELSE

S WHEN COUNTER = "000" and leds = "0010000001" and start = '1' ELSE

P WHEN COUNTER = "001" and leds = "0010000001" and start = '1' ELSE

"00001" WHEN COUNTER = "010" and leds = "0010000001" and start = '1' ELSE

N WHEN COUNTER = "011" and leds = "0010000001" and start = '1' ELSE

Blank WHEN COUNTER = "100" and leds = "0010000001" and start = '1' ELSE

Blank WHEN COUNTER = "101" and leds = "0010000001" and start = '1' ELSE

Blank WHEN COUNTER = "110" and leds = "0010000001" and start = '1' ELSE

time\_sn WHEN COUNTER = "111" and leds = "0010000001" and start = '1' ELSE

R WHEN COUNTER = "000" and leds = "0001000010" and start = '1' ELSE

"00001" WHEN COUNTER = "001" and leds = "0001000010" and start = '1' ELSE

N WHEN COUNTER = "010" and leds = "0001000010" and start = '1' ELSE

S WHEN COUNTER = "011" and leds = "0001000010" and start = '1' ELSE

E WHEN COUNTER = "100" and leds = "0001000010" and start = '1' ELSE

Blank WHEN COUNTER = "101" and leds = "0001000010" and start = '1' ELSE

Blank WHEN COUNTER = "110" and leds = "0001000010" and start = '1' ELSE

time\_sn WHEN COUNTER = "111" and leds = "0001000010" and start = '1' ELSE

Blank;

SEV\_SEG\_DRIVER <=

"01111111" WHEN COUNTER = "000" ELSE

"10111111" WHEN COUNTER = "001" ELSE

"11011111" WHEN COUNTER = "010" ELSE

"11101111" WHEN COUNTER = "011" ELSE

"11110111" WHEN COUNTER = "100" ELSE

"11111011" WHEN COUNTER = "101" ELSE

"11111101" WHEN COUNTER = "110" ELSE

"11111110" WHEN COUNTER = "111" ELSE

"11111111";

END BEHAVIORAL;

*Wash*

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

-- Uncomment the following library declaration if using

-- arithmetic functions with Signed or Unsigned values

use IEEE.NUMERIC\_STD.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

-- Uncomment the following library declaration if instantiating

-- any Xilinx primitives in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity washing\_machine is

Port(

CK : in std\_logic;

start : in std\_logic;

reset : in std\_logic;

userMode : in std\_logic\_vector(4 downto 0);

state : out std\_logic\_vector(9 downto 0);

time\_sn : out std\_logic\_vector(4 downto 0));

end washing\_machine;

architecture Behavioral of washing\_machine is

signal hz : std\_logic;

signal tState : std\_logic\_vector(1 downto 0);

signal washC : integer;

signal spinC : integer;

signal rinseC : integer;

signal modes : std\_logic\_vector(3 downto 0);

begin

with userMode select washC <=

5 when "10000",

6 when "01000",

7 when "00100",

5 when "00010",

3 when "00001",

0 when others;

with userMode select spinC <=

3 when "10000",

4 when "01000",

5 when "00100",

3 when "00010",

2 when "00001",

0 when others;

with userMode select rinseC <=

3 when "10000",

2 when "01000",

3 when "00100",

2 when "00010",

1 when "00001",

0 when others;

process(CK,start,reset)

constant freq : integer := 400;

variable curState : UNSIGNED(1 downto 0) := "11";

variable tick : integer := 0;

variable zero : std\_logic := '0';

variable counter : integer;

variable hold : std\_logic;

begin

if rising\_edge(CK) then

if curState = "11" and start = '1' and hold = '0' then

counter := washC;

curState := "00";

elsif curState = "11" and start = '1' and hold = '0' then

curState := curState;

end if;

if reset = '1' then

curState := "11";

zero := '0';

counter := 0;

tick := 0;

end if;

if zero = '1' then

curState := curState + "01";

zero := '0';

tick := 0;

end if;

if curState = "00" and zero = '0' then

if tick < freq then

tick := tick + 1;

else

tick := 0;

counter := counter - 1;

end if;

if counter = 0 then

zero := '1';

counter := spinC;

end if;

end if;

if curState = "01" and zero = '0' then

if tick < freq then

tick := tick + 1;

else

tick := 0;

counter := counter - 1;

end if;

if counter = 0 then

zero := '1';

counter := rinseC;

end if;

end if;

if curState = "10" and zero = '0' then

if tick < freq then

tick := tick + 1;

else

tick := 0;

counter := counter - 1;

end if;

if counter = 0 then

curState := "11";

end if;

end if;

hold := start;

end if;

tState <= std\_logic\_vector(curState);

time\_sn <= std\_logic\_vector(to\_unsigned(counter, time\_sn'length));

end process;

with tState select modes <=

"1000" when "11",

"0100" when "00",

"0010" when "01",

"0001" when "10",

"1111" when others;

state <= modes & "0000" & tState;

end Behavioral;

*UCF*

NET "leds[9]" LOC = P77;

NET "leds[8]" LOC = P83;

NET "leds[7]" LOC = P84;

NET "leds[6]" LOC = P86;

NET "leds[5]" LOC = P89;

NET "leds[4]" LOC = P93;

NET "leds[3]" LOC = P3;

NET "leds[2]" LOC = P6;

NET "leds[1]" LOC = P13;

NET "leds[0]" LOC = P16;

NET "Clock" LOC = P40;

NET "reset" LOC = P35;

NET "start" LOC = P15;

NET "SevsegDriver[7]" LOC = P61;

NET "SevsegDriver[6]" LOC = P60;

NET "SevsegDriver[5]" LOC = P57;

NET "SevsegDriver[4]" LOC = P59;

NET "SevsegDriver[3]" LOC = P56;

NET "SevsegDriver[2]" LOC = P52;

NET "SevsegDriver[1]" LOC = P49;

NET "SevsegDriver[0]" LOC = P50;

NET "SevsegData[6]" LOC = P71;

NET "SevsegData[5]" LOC = P62;

NET "SevsegData[4]" LOC = P65;

NET "SevsegData[3]" LOC = P72;

NET "SevsegData[2]" LOC = P73;

NET "SevsegData[1]" LOC = P98;

NET "SevsegData[0]" LOC = P64;

NET "userMode[4]" LOC = P78;

NET "userMode[3]" LOC = P82;

NET "userMode[2]" LOC = P85;

NET "userMode[1]" LOC = P88;

NET "userMode[0]" LOC = P90;

RLT Schematics

,



