**Video link: https://youtu.be/Ds\_K42JTFVQ**

**Toplevel**

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity toplevel is

port (clk,r: in std\_logic;

coin: in std\_logic;

starttimer: in std\_logic;

startwashing: in std\_logic;

ultrawashing: in std\_logic;

lidcloser: in std\_logic;

tempo: out std\_logic;

coininside: out std\_logic;

donewash: out std\_logic;

lidajar: out std\_logic;

stages: out std\_logic\_vector(3 downto 0);

seven1: out std\_logic\_vector(6 downto 0);

seven2: out std\_logic\_vector(6 downto 0));

end;

architecture beh of toplevel is

signal clockfeeder:std\_logic;

signal resetfeeder: std\_logic;

signal lidfeeder: std\_logic;

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

signal sevenfeeder2: std\_logic\_vector(3 downto 0);

component clockdiv is

port(clk: in STD\_logic;

start\_timer: in STD\_logic;

FastClock , MediumClock , SlowClock , led0 : out STD\_LOGIC );

end component ;

component washingmachinefsm is

port(clk,r: in std\_logic;

coin: in std\_logic;

startwash: in std\_logic;

ultrawash: in std\_logic;

closelid: in std\_logic;

coinin: out std\_logic;

stages: out std\_logic\_vector(3 downto 0);

washdone: out std\_logic;

lidcounter: out std\_logic;

resetimer: in std\_logic);

end component;

component lidcontroller is

port(clk,r, lidcontroller: in std\_logic;

lidopen: out std\_logic;

counter1: out std\_logic\_vector(3 downto 0));

end component;

component washcontroller is

port(clk,r: in std\_logic;

startwash: in std\_logic;

ultrawash: in std\_logic;

resettimer: out std\_logic;

counter2: out std\_logic\_vector(3 downto 0));

end component;

component sevenseg is

Port ( aa : in STD\_LOGIC\_VECTOR(3 downto 0);

seven : out STD\_LOGIC\_VECTOR(6 downto 0));

end component;

begin

clock: clockdiv port map(clk=>clk, start\_timer=> starttimer, slowclock=>clockfeeder,led0=>tempo);

wash: washingmachinefsm port map(clk=>clockfeeder, r=>r, coin=>coin,startwash=>startwashing,ultrawash=>ultrawashing,closelid=>lidcloser,coinin=>coininside,stages=>stages,washdone=>donewash,resetimer=>resetfeeder,lidcounter=>lidfeeder);

lid : lidcontroller port map(clk=>clockfeeder, r=>r, lidcontroller=>lidfeeder,lidopen=>lidajar, counter1=>sevenfeeder);

washercon: washcontroller port map(clk=>clockfeeder,r=>r,startwash=>startwashing,ultrawash=>ultrawashing,resettimer=>resetfeeder, counter2=>sevenfeeder2);

seven0: sevenseg port map(aa=>sevenfeeder,seven=>seven1);

seven00: sevenseg port map(aa=>sevenfeeder2,seven=>seven2);

end;

Washing controller

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity washcontroller is

port(clk,r: in std\_logic;

startwash: in std\_logic;

ultrawash: in std\_logic;

resettimer: out std\_logic;

counter2: out std\_logic\_vector(3 downto 0));

end;

architecture beh of washcontroller is

type state is (s0,s1,s2,s3,s4);

signal cs,ns:state;

begin

process(clk,r)

begin

if(r='1') then

cs<=s0;

elsif (clk'event and clk='1') then

cs<=ns;

end if;

end process;

process(cs,startwash,ultrawash)

begin

case cs is

when s0=>

if (startwash='1') then

counter2<="0000";

ns<=s1;

elsif (ultrawash='1') then

counter2<="0000";

ns<=s1;

else

ns<=s0;

counter2<="0000";

end if;

when s1=>

if (startwash='1') then

counter2<="0001";

ns<=s2;

elsif (ultrawash='1') then

counter2<="0001";

ns<=s2;

else

ns<=s0;

counter2<="0000";

end if;

when s2=>

if (startwash='1') then

counter2<="0010";

ns<=s4;

elsif (ultrawash='1') then

counter2<="0010";

ns<=s4;

else

ns<=s0;

counter2<="0000";

end if;

when s3=>

if (startwash='1') then

counter2<="0011";

ns<=s4;

elsif (ultrawash='1') then

counter2<="0011";

ns<=s4;

else

ns<=s0;

counter2<="0000";

end if;

when s4=>

if (startwash='1') then

counter2<="0100";

ns<=s0;

resettimer<='1';

elsif (ultrawash='1') then

counter2<="0100";

ns<=s0;

resettimer<='1';

else

ns<=s0;

counter2<="0000";

end if;

when others=>

null;

end case;

end process;

end;

Lid controller

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity lidcontroller is

port (clk,r, lidcontroller: in std\_logic;

lidopen: out std\_logic;--These are all signals that the fsm will send and start the counters

counter1: out std\_logic\_vector(3 downto 0));--this counter will go back to fsm

end;

architecture beh of lidcontroller is

type state is (s0,s1,s2,s3,s4,s5,s6,s7,s8,s9,s10);

signal cs,ns:state;

signal lidopentmp:std\_logic;

begin

process(clk,r)

begin

if(r='1') then

cs<=s0;

elsif (clk' event and clk='1') then

cs<=ns;

end if;

end process;

process(cs,lidcontroller)

begin

case cs is

when s0=>

if (lidcontroller='0') then

counter1<="0000";

lidopentmp<='1';

ns<=s1;

else

ns<=s0;

counter1<="0000";

end if;

when s1=>

if (lidcontroller='0') then

counter1<="0001";

lidopentmp<='1';

ns<=s2;

else

ns<=s0;

counter1<="0000";

end if;

when s2=>

if (lidcontroller='0') then

counter1<="0010";

lidopentmp<='1';

ns<=s3;

else

ns<=s0;

counter1<="0000";

end if;

when s3=>

if (lidcontroller='0') then

counter1<="0011";

lidopentmp<='1';

ns<=s4;

else

ns<=s0;

counter1<="0000";

end if;

when s4=>

if (lidcontroller='0') then

counter1<="0100";

lidopentmp<='1';

ns<=s5;

else

ns<=s0;

counter1<="0000";

end if;

when s5=>

if (lidcontroller='0') then

counter1<="0101";

lidopentmp<='1';

ns<=s6;

else

ns<=s0;

counter1<="0000";

end if;

when s6=>

if (lidcontroller='0') then

counter1<="0110";

lidopentmp<='1';

ns<=s7;

else

ns<=s0;

counter1<="0000";

end if;

when s7=>

if (lidcontroller='0') then

counter1<="0111";

lidopentmp<='1';

ns<=s8;

else

ns<=s0;

counter1<="0000";

end if;

when s8=>

if (lidcontroller='0') then

counter1<="1000";

lidopentmp<='1';

ns<=s9;

else

ns<=s0;

counter1<="0000";

end if;

when s9=>

if (lidcontroller='0') then

counter1<="1001";

lidopentmp<='1';

ns<=s10;

else

ns<=s0;

counter1<="0000";

end if;

when s10=>

if (lidcontroller='0') then

counter1<="1111";

lidopentmp<='1';

ns<=s10;

else

ns<=s0;

counter1<="0000";

end if;

when others=>

counter1<="1111";

end case;

end process;

lidopen<=lidopentmp;

end;

Clock div

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.NUMERIC\_STD.ALL;

entity ClockDiv is

port(clk: in STD\_logic;

start\_timer: in STD\_logic;

FastClock , MediumClock , SlowClock , led0 : out STD\_LOGIC );

end Clockdiv ;

architecture beh of Clockdiv is

signal slowClock\_sig : STD\_LOGIC;

begin

process

variable cnt : STD\_LOGIC\_VECTOR (26 downto 0):= "000000000000000000000000000";

begin

wait until (( clk 'EVENT) AND ( clk = '1'));

if ( start\_timer = '0') then

cnt := "000000000000000000000000000" ;

else

cnt := STD\_LOGIC\_VECTOR( unsigned ( cnt ) + 1);

end if ;

FastClock <= cnt (22);

MediumClock <= cnt (24);

SlowClock <= cnt (26);

slowClock\_sig <= cnt (26);

if ( slowClock\_sig = '1') then

led0 <= '1';

else

led0 <= '0';

end if ;

end process;

end beh;

Seven decorder

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity sevenseg is

Port ( aa : in STD\_LOGIC\_VECTOR(3 downto 0);

seven : out STD\_LOGIC\_VECTOR(6 downto 0));

end sevenseg;

architecture Behavioral of sevenseg is

begin

process(aa)

begin

Case aa is

when "0000"=> seven <="1000000"; -- '0'

when "0001"=> seven <="1111001"; -- '1'

when "0010"=> seven <="0100100"; -- '2'

when "0011"=> seven <="0110000"; -- '3'

when "0100"=> seven <="0011001"; -- '4'

when "0101"=> seven <="0010010"; -- '5'

when "0110"=> seven <="0000010"; -- '6'

when "0111"=> seven <="1111000"; -- '7'

when "1000"=> seven <="0000000"; -- '8'

when "1001"=> seven <="0011000"; -- '9'

when "1111"=>seven<="1111001"; -- 'E'

when others=> seven <= "1111001";-- everything else would be 0

end case;

end process;

end Behavioral;

Washing fsm

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_arith.all;

use ieee.std\_logic\_unsigned.all;

entity washingmachinefsm is

port(clk,r: in std\_logic;

coin: in std\_logic;

startwash: in std\_logic;

ultrawash: in std\_logic;

closelid: in std\_logic;

coinin: out std\_logic;

stages: out std\_logic\_vector(3 downto 0);

washdone: out std\_logic;

lidcounter: out std\_logic;

resetimer: in std\_logic);

end;

architecture beh of washingmachinefsm is

type state is (s0,soak, wash, rinse, spin,soak2, rinse2);

signal cs, ns: state;

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

begin

process(clk,r)

begin

if( r='1') then

cs<=s0;

elsif(clk'event and clk='1') then

cs<=ns;

end if;

end process;

process(coin,cs,startwash, ultrawash,closelid)

begin

stagestmp<="0000";

coinin<='0';

lidcounter<='0';

washdone<='0';

case cs is

when s0=>

if(coin='0') then

ns<= s0;

if (closelid='0') then

lidcounter<='1';

else

lidcounter<='0';

end if;

elsif (coin='1' and startwash='1' and closelid='1' and resetimer='1') then

ns<=soak;

coinin<='1';

elsif( coin='1' and ultrawash='1' and closelid='1' and resetimer='1') then

ns<=soak;

coinin<='1';

end if;

stagestmp<="0001";

when soak=>

if(coin='0') then

ns<=s0;

if (closelid='0') then

lidcounter<='1';

else

lidcounter<='0';

end if;

elsif(coin='1' and startwash='1' and closelid='1' and resetimer='1') then

ns<=wash;

coinin<='1';

elsif(coin='1' and ultrawash='1' and closelid='1' and resetimer='1') then

ns<=wash;

coinin<='1';

else

end if;

stagestmp<="0010";

when wash=>

if(coin='0') then

ns<=s0;

if (closelid='0') then

lidcounter<='1';

else

lidcounter<='0';

end if;

elsif(coin='1' and startwash='1' and closelid='1' and resetimer='1') then

ns<=rinse;

coinin<='1';

elsif(coin='1' and ultrawash='1' and closelid='1' and resetimer='1') then

ns<=rinse;

coinin<='1';

else

end if;

stagestmp<="0010";

when rinse=>

if(coin='0') then

ns<=s0;

if (closelid='0') then

lidcounter<='1';

else

lidcounter<='0';

end if;

elsif(coin='1' and startwash='1' and closelid='1' and resetimer='1') then

ns<=spin;

coinin<='1';

elsif(coin='1' and ultrawash='1' and closelid='1' and resetimer='1' ) then

ns<=spin;

coinin<='1';

end if;

stagestmp<="0100";

when spin=>

if(coin='0') then

ns<=s0;

if (closelid='0') then

lidcounter<='1';

else

lidcounter<='0';

end if;

elsif(coin='1' and startwash='1' and closelid='1' and resetimer='1' ) then

ns<=s0;

washdone<='1';

coinin<='0';

elsif(coin='1' and ultrawash='1' and closelid='1' and resetimer='1') then

ns<=soak2;

coinin<='1';

end if;

stagestmp<="1000";

when soak2=>

if(coin='0') then

ns<=s0;

if (closelid='0') then

lidcounter<='1';

else

lidcounter<='0';

end if;

elsif(coin='1' and ultrawash='1' and closelid='1' and resetimer='1') then

ns<=rinse2;

coinin<='1';

end if;

stagestmp<="1001";

when rinse2=>

if(coin='0') then

ns<=s0;

if (closelid='0') then

lidcounter<='1';

else

lidcounter<='0';

end if;

elsif(coin='1' and ultrawash='1' and closelid='1' and resetimer='1') then

ns<=s0;

washdone<='1';

coinin<='0';

end if;

stagestmp<="1010";

when others=>

null;

end case;

end process;

stages<=stagestmp;

end;