----------------------------------------------------------------------------------

-- Company:

-- Engineer:

--

-- Create Date: 21:51:49 12/03/2018

-- Design Name:

-- Module Name: controller - Behavioral

-- Project Name:

-- Target Devices:

-- Tool versions:

-- Description:

--

-- Dependencies:

--

-- Revision:

-- Revision 0.01 - File Created

-- Additional Comments:

--

----------------------------------------------------------------------------------

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;

-- Uncomment the following library declaration if instantiating

-- any Xilinx primitives in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity controller is

Port (

clk: in STD\_LOGIC;

reset: in STD\_LOGIC;

wash\_time:in std\_logic\_vector(7 downto 0);

Zeroflag:in STD\_LOGIC;

empty:in STD\_LOGIC;

full:in STD\_LOGIC;

start:in STD\_LOGIC;

turn: out STD\_LOGIC;

hot:out STD\_LOGIC;

cold:out STD\_LOGIC;

drain:out STD\_LOGIC;

Load:out STD\_LOGIC;

Dec:out STD\_LOGIC);

end controller;

architecture Behavioral of controller is

type state is (T0, T1, T2,T3,T4,T5,T6);

signal current\_state, next\_state: state;

signal turns : STD\_LOGIC;

signal hots : STD\_LOGIC;

signal colds : STD\_LOGIC;

signal decs : STD\_LOGIC;

signal drains : STD\_LOGIC;

signal loads : STD\_LOGIC;

begin

process (clk, reset, current\_state)

variable washtime: std\_logic\_vector(7 downto 0);

begin

if (reset = '1') then

current\_state <= T0;

startS <= '0';

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

-- T0

if(current\_state = T0) then

if(start = '1') then

hots <= '1';

decs <= '0';

turns <= '0';

colds <= '0';

loads <= '0';

drains <= '0';

end if;

-- T1

elsif(current\_state = T1) then

if(full = '1') then

washtime := wash\_time;

hots <= '0';

decs <= '1';

turns <= '1';

loads <= '1';

end if ;

-- T2

elsif ( current\_state = T2) then

if(zeroflag = '1' ) then

decs <= '0';

turns <= '0';

drains <= '1';

loads<= '0';

end if;

-- T3

elsif (current\_state = T3) then

if(empty = '1') then

Washtime := wash\_time;

decs <= '1';

turns <= '1';

loads <='1';

if(zeroflag = '1') then

decs <= '0';

turns <= '0';

drains <= '0';

loads <='0';

end if;

end if;

-- T4

elsif(current\_state = T4) then

if(full = '1') then

Washtime := wash\_time;

decs <= '1';

turns <= '1';

colds <= '0';

loads <='1';

end if;

-- T5

elsif (current\_state = T5) then

colds <= '1';

if (zeroflag = '0') then

decs <= '0';

turns <= '0';

drains <= '1';

loads <='0';

end if;

-- T6

elsif (current\_state = T6) then

if(empty = '1') then

Washtime := wash\_time;

decs <= '1';

turns <= '1';

loads <='1';

if(zeroflag = '1') then

decs <= '0';

turns <= '0';

drains <= '0';

loads <='0';

end if;

end if;

end if; -- for all states

turn <= turns;

hot <= hots;

cold <= colds;

drain <= drains;

Load <= loads;

Dec <= decs;

current\_state<= next\_state;

end if; -- for clock

End process;

process (current\_state,start,full,Zeroflag,empty,wash\_time)

begin

case current\_state is

when T0 =>

if ( start = '1') then

next\_state <= T1;

else

next\_state <= T0;

end if;

when T1 =>

if( full = '1') then

next\_state <= T2;

else

next\_state <= T1;

end if;

when T2 =>

if(Zeroflag = '1') then

next\_state <= T3;

else

next\_state<=T2;

end if;

when T3 =>

if (empty = '1') then

if(zeroflag = '1') then

next\_state <=T4;

else

next\_state <=T3;

end if;

else

next\_state <=T3;

end if;

when T4 =>

if( full = '1') then

next\_state <= T5;

else

next\_state <= T4;

end if;

when T5 =>

if(Zeroflag = '1') then

next\_state <= T6;

else

next\_state<=T5;

end if;

when T6 =>

if (empty = '1') then

if(zeroflag = '1') then

next\_state <=T0;

else

next\_state <=T6;

end if;

else

next\_state <=T6;

end if;

End case;

End process;

end Behavioral;