**Protokollierung zu Versuch 8**

**Gruppe 12**

Yazan Al-Kabbani (1648827)

Osama Hammoud (1536285)

Ahmet Taner Kahraman (1660926)

**Protokollierung Aufgabe 1**

**1)** Der gemessene Strom für **eine LED** mit einem Duty Cycle von **100% : - mA**

**2)** Der gemessene Strom für **vier LEDs** mit einem Duty Cycle von **100% : - mA**

**3)** Der gemessene Strom für **vier LEDs** mit einem Duty Cycle von **50% : - mA**

**4)** Der gemessene Strom für **vier LEDs** mit einem Duty Cycle von **50% im Energiesparmodus : - mA**

**Protokollierung Aufgabe 3**

Wie in Aufgabenstellung, leuchten alle LEDs wenn der eingegebene Code richtig ist und sonst gar keine

**Zusätzlich** wird es mit einzelnen LEDs dargestellt, in welchem Zustand man aktuell ist. Es leuchtet nur eine LED, jenachdem in welchem Zustand man ist. Somit kann man auch z.B. verstehen, dass auf die Eingabe eines Codes gewartet wird usw.

**Protokollierung Aufgabe 2**

tuerschloss.vhd

*----------------------------------------------------------------------------------*

*-- Company:*

*-- Engineer:*

*--*

*-- Create Date: 17.06.2023 15:30:08*

*-- Design Name:*

*-- Module Name: tuerschloss - 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 leaf cells in this code.*

*--library UNISIM;*

*--use UNISIM.VComponents.all;*

entity Tuerschloss is

    Port ( Clock : in STD\_LOGIC;

           ButtonsIn : in STD\_LOGIC\_VECTOR(3 downto 0); *-- Button0 als Reset implementiert*

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

           LEDsOut : out STD\_LOGIC\_VECTOR(3 downto 0));

end Tuerschloss;

architecture Behavioral of Tuerschloss is

*-- Debounced Buttons*

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

*-- Reset Signal*

signal Reset : std\_logic;

*-- State Signals*

type state\_type is (

    Init, Init2NewCode, Init2EnterCode,

    NewCode, NewCode2, EnterCode, EnterCode2Success, EnterCode2Fail,

    Success, Fail

);

signal current\_s, next\_s : state\_type;

*-- code and led*

signal code: std\_logic\_vector(3 downto 0) := "0000";

signal next\_code: std\_logic\_vector(3 downto 0) := "0000";

signal led: std\_logic\_vector(3 downto 0) := "0000";

signal next\_led: std\_logic\_vector(3 downto 0) := "0000";

*--  Entprellung – es wurde das Modul vom Asha-Prjekt verwendet*

component Debounce is

    Port (

        clk     :   in   std\_logic;                     *--! Taktsignal*

        keyin    :  in   std\_logic\_vector(3 downto 0);  *--! bouncing input*

        keyout  :   out  std\_logic\_vector(3 downto 0)   *--! debounced output*

     );

end component;

begin

Debouncing  :   Debounce

port map    (

    clk=>Clock,     *--! Taktsignal*

    keyin=>ButtonsIn,   *--! bouncing buttons*

    keyout=>Buttons *--! debounced buttons*

);

*-- nebenl�ufige Anweisungen*

Reset <= Buttons(0);

seq: process(Clock)

begin

    if rising\_edge(Clock) then

        if(Reset = '1') then

            current\_s <= Init; *-- or Init2 ????*

        else

            current\_s <= next\_s;

            code <= next\_code;

            led <= next\_led;

        end if;

    end if;

end process;

comb: process(current\_s, Buttons, Switches, led, code) is

begin

*--  Latches vermeiden*

    next\_s <= current\_s;

    next\_code <= code;

    next\_led <= led;

    case current\_s is

        when Init =>

            next\_led <= b"0001";

            if (Buttons(2) = '1') then next\_s <= Init2EnterCode;

            elsif (Buttons(3) = '1') then next\_s <= Init2NewCode;

            end if;

        when Init2EnterCode =>

            next\_led <= b"0001";

            if (Buttons(2) = '0') then next\_s <= EnterCode;

            end if;

        when Init2NewCode =>

            next\_led <= b"0001";

            if (Buttons(3) = '0') then next\_s <= NewCode;

            end if;

        when NewCode =>

            next\_led <= b"0010";

            if (Buttons(3) = '1') then *-- Code �bernehmen*

                next\_code <= Switches(3 downto 0);

                next\_s <= NewCode2;

            end if;

        when NewCode2 =>

            next\_led <= b"0010";

            if (Buttons(3) = '0') then

                next\_s <= Init;

            end if;

*-- Zustand, Code wird eingegeben und �berpr�ft*

        when EnterCode =>

            next\_led <= b"0100";

*-- Solange Button2 = 0, kann man den Code eingeben.*

*-- Wenn Button2 = 1, wirde der Code �berpr�ft*

*-- und es wird in den n�chsten Zustand gewechselt*

            if (Buttons(2) = '1') then *-- Code eingegeben*

                if ( next\_code = Switches(3 downto 0)) then *-- Code ist korrekt*

                    next\_s <= EnterCode2Success;

                else *-- Code ist falsch*

                    next\_s <= EnterCode2Fail;

                end if;

            end if;

*-- �bergangszustand zu Success*

        when EnterCode2Success =>

            next\_led <= b"0100";

            if (Buttons(2) = '0') then

                next\_s <= Success;

            end if;

*-- �bergangszustand zu Fail*

        when EnterCode2Fail =>

            next\_led <= b"0100";

            if (Buttons(2) = '0') then

                next\_s <= Fail;

            end if;

*-- LEDs ausgeschaltet*

         when Fail => next\_led <= b"0000";

*-- LEDs eingeschaltet*

         when Success => next\_led <= b"1111";

    end case;

    LEDsOut <= next\_led;

end process;

end Behavioral;

LED.vhd

*----------------------------------------------------------------------------*

*-- Company:*

*-- Engineer:*

*--*

*-- Create Date: 13.06.2023 14:57:09*

*-- Design Name:*

*-- Module Name: LED - 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;

use ieee.numeric\_std.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 leaf cells in this code.*

*--library UNISIM;*

*--use UNISIM.VComponents.all;*

entity LED is

    Port ( Clock : in STD\_LOGIC;

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

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

           LEDsOut : out STD\_LOGIC\_VECTOR(3 downto 0));

end LED;

architecture Behavioral of LED is

signal counter: unsigned(8 downto 0 ) := (others => '0');

signal PWMSignalLow: std\_logic :='0';

signal PWMSignalHigh: std\_logic :='1';

signal PWMClock: std\_logic :='0';

begin

*-- Es wird ein CLock-Signal von 50KHz f�r PWMSignal erzeugt*

clockdivider: process(Clock)

begin

    if rising\_edge(Clock) then

        counter <= counter + 1;

        if (counter = 500) then

            PWMClock <= '1';

            counter <= (others => '0');

        else

            PWMClock <= '0';

        end if;

    end if;

end process;

*-- Es werden in diesem Prozess LEDs gesteuert*

*-- Es wird auch PWM Signal mit Hilfe von PWMClock aktualisiert*

LED : process(Clock, PWMClock)

begin

    if rising\_edge(PWMClock) then

        PWMSignalLow <= not PWMSignalLow;

        PWMSignalHigh <= not PWMSignalHigh;

    end if;

    if rising\_edge(Clock) then

*-- Impelementierung:*

*-- Wenn der erste von zu dem betroffenen LED geh�rigen Schaltern (Switches) auf 1*

*--      -> Wenn Schalter 2 (Buttons) auf 0: LED an*

*--      -> Wenn Schalter 2 (Buttons) auf 1: PWM - Energiesparmodus*

*-- Ansonsten betrachte Schalter 2 (Buttons):*

*--      -> Wenn der auf 1:      PWM*

*--      -> Wenn auch der auf 0: LED aus*

*-- LEDsOut(0): Switches(0), Buttons(0)*

        if (Switches(0) = '0' and Buttons(0) = '0') then *-- aus*

            LEDsOut(0) <= '0';

        elsif (Switches(0) = '1' and Buttons(0) = '0') then *-- an*

            LEDsOut(0) <= '1';

        elsif (Switches(0) = '0' and Buttons(0) = '1') then *-- PWM*

            LEDsOut(0) <= PWMSignalHigh;

        else *-- PWM Energiesparmodus -- Beide Schalter auf 1*

            LEDsOut(0) <= PWMSignalLow;

        end if;

*-- LEDsOut(1): Switches(1), Buttons(1)*

        if (Switches(1) = '0' and Buttons(1) = '0') then *-- aus*

            LEDsOut(1) <= '0';

        elsif (Switches(0) = '1' and Buttons(1) = '0') then *-- an*

            LEDsOut(1) <= '1';

        elsif (Switches(1) = '0' and Buttons(1) = '1') then *-- PWM*

            LEDsOut(1) <= PWMSignalHigh;

        else *-- PWM Energiesparmodus -- Beide Schalter auf 1*

            LEDsOut(1) <= PWMSignalLow;

        end if;

*-- LEDsOut(2): Switches(2), Buttons(2)*

        if (Switches(2) = '0' and Buttons(2) = '0') then *-- aus*

            LEDsOut(2) <= '0';

        elsif (Switches(2) = '1' and Buttons(2) = '0') then *-- an*

            LEDsOut(2) <= '1';

        elsif (Switches(2) = '0' and Buttons(2) = '1') then *-- PWM*

            LEDsOut(2) <= PWMSignalHigh;

        else *-- PWM Energiesparmodus -- Beide Schalter auf 1*

            LEDsOut(2) <= PWMSignalHigh;

        end if;

*-- LEDsOut(3): Switches(3), Buttons(3)*

        if (Switches(3) = '0' and Buttons(3) = '0') then *-- aus*

            LEDsOut(3) <= '0';

        elsif (Switches(3) = '1' and Buttons(3) = '0') then *-- an*

            LEDsOut(3) <= '1';

        elsif (Switches(3) = '0' and Buttons(3) = '1') then *-- PWM*

            LEDsOut(3) <= PWMSignalHigh;

        else *-- PWM Energiesparmodus -- Beide Schalter auf 1*

            LEDsOut(3) <= PWMSignalHigh;

        end if;

    end if;

end process;

end Behavioral;