**Protokollierung zu Versuch 9**

**Gruppe 12**

Yazan Al-Kabbani (1648827)

Osama Hammoud (1536285)

Ahmet Taner Kahraman (1660926)

**Protokollierung Aufgabe 1**

**Licht:**

Es wurde der Wert von SensorDataLight mit konstanten Lux Werte (lux10, lux50, lux200 ) verglichen.

Je nachdem in welchem Bereich der Wert ist, wurde PWM3LightValueControl dementsprechend angepasst.

*-- weniger als 10 lux -> duty cycle: 100%*

*-- weniger als 50 lux -> duty cycle: 50%*

*-- weniger als 200 lux -> duty cycle: 25%*

*-- mehr als 200 lux -> duty cycle: 0%*

**Temperatur:**

Die Werte „SensordataTempIn“ und „SensordataTempOut“ sind unsigned.

Als erstes wurden diese zu int umgewandelt und man hat dann die Differenz gebildet.

Dann wurde, je nachdem wie groß der Unterschied ist, „PeltierDirectionControl“ auf 1 oder 0 gesetzt.

Außerdem wurde der Bereich zwischen 12V - 20V (1.5C – 2.5C) als Toleranzbereich angenommen und PeltierDirectionControl wurde dabei nicht geändert, damit kein ständiger Wechsel stattfindet.

Aber man hat trotzdem die beiden Lüfter laufen lassen.

to\_integer(unsigned(SensordataTempIn)) - to\_integer(unsigned(SensordataTempOut)) < 12

(12V = 1.5C) -> PeltierDirectionControl <= '1'; -- heizen

to\_integer(unsigned(SensordataTempIn)) - to\_integer(unsigned(SensordataTempOut)) > 20

(20V = 2.5C) -> PeltierDirectionControl <= '0'; -- kühlen

**Protokollierung Aufgabe 2**

asha-regelung.vhd

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

*--! Standardbibliothek benutzen*

library IEEE;

*--! Logikelemente verwenden*

use IEEE.STD\_LOGIC\_1164.ALL;

*--! Numerisches Rechnen ermoeglichen*

use IEEE.NUMERIC\_STD.ALL;

*--! @brief ASHA-Modul - Regelung*

*--! @details Dieses Modul enthaelt die Regelung*

entity AshaRegelung is

  Port (

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

    Reset : in std\_logic;                                           *--! Resetsignal*

    EnClockLight : in std\_logic;                                    *--! Enable-Signal fuer die Lichtregelung*

    EnClockTemp  : in std\_logic;                               *--! Enable-Signal fuer die Temperaturregelung*

    SensordataLight   : in std\_logic\_vector(11 downto 0);           *--! Aktuelle Lichtwerte*

    SensordataTempIn  : in std\_logic\_vector(11 downto 0);           *--! Aktuelle Innentemperatur*

     SensordataTempOut : in std\_logic\_vector(11 downto 0);          *--! Aktuelle Außentemperatur*

     PWM1FanInsideValueControl  : out std\_logic\_vector(7 downto 0);     *--! PWM-Wert innerere Luefter*

    PWM2FanOutsideValueControl : out std\_logic\_vector(7 downto 0);   *--! PWM-Wert aeusserer Luefter*

    PWM3LightValueControl   : out std\_logic\_vector(7 downto 0);        *--! PWM-Wert Licht*

    PWM4PeltierValueControl : out std\_logic\_vector(7 downto 0);        *--! PWM-Wert Peltier*

    PeltierDirectionControl : out std\_logic;                              *--! Pelier Richtung heizen (=1)/kuehlen(=0)*

    ControlLightDiffOut : out unsigned(12 downto 0);                *--! Aktuelle Regeldifferenz Licht*

    ControlTempDiffOut  : out unsigned(12 downto 0)            *--! Aktuelle Regeldifferenz Temperatur*

     );

end AshaRegelung;

architecture Behavioral of AshaRegelung is

    constant lux10 : unsigned(11 downto 0) := "111111011010"; *-- 4058*

    constant lux50 : unsigned(11 downto 0) := "111101001000"; *-- 3912*

    constant lux200 : unsigned(11 downto 0) := "110100100100"; *-- 3364*

begin

*-- Versuch 9: Realisierung der Lichtsteuerung*

lightControl: process (Clock)

begin

    if (rising\_edge(Clock)) then

*-- TODO*

        if(Reset = '1') then

            PWM3LightValueControl <= (others => '0');

        else

            if(EnClockLight = '1') then

                if(unsigned(SensordataLight) > lux10 )then *-- weniger als 10 lux*

                    PWM3LightValueControl <= b"11111111"; *-- duty cycle: 100%*

                elsif (unsigned(SensordataLight) > lux50) then *-- weniger als 50 lux*

                    PWM3LightValueControl <= b"10000000"; *-- duty cycle: 50%*

                elsif (unsigned(SensordataLight) > lux200 ) then *-- weniger als 200 lux*

                    PWM3LightValueControl <= b"01000000"; *-- duty cycle: 25%*

                else *-- mehr als 200 lux*

                    PWM3LightValueControl <= b"00000000"; *-- duty cycle: 0%*

                end if ;

            end if;

        end if;

    end if;

end process lightControl;

*-- Versuch 9: Realisierung der Temperatursteuerung*

*-- Ziel: Innen zwei Grad waermer als draussen*

*-- 2�C entsprechen einem Wert von ca. 15;*

*-- um schnelles Umschalten zu verhindern, wird ein Toleranzbereich genommen*

tempControl: process (EnClockTemp)

begin

    if(Reset = '1') then

        PWM1FanInsideValueControl <= b"00000000";

        PWM2FanOutsideValueControl <= b"00000000";

    elsif (rising\_edge(EnClockTemp)) then

        if(to\_integer(unsigned(SensordataTempIn)) - to\_integer(unsigned(SensordataTempOut)) < 12) then *-- < 12v (1.5c)*

            PeltierDirectionControl <= '1'; *-- heizen*

            PWM1FanInsideValueControl <= b"11111111";

            PWM2FanOutsideValueControl <= b"11111111";

        elsif(to\_integer(unsigned(SensordataTempIn)) - to\_integer(unsigned(SensordataTempOut)) > 20) then *-- > 20v (2.5c)*

            PeltierDirectionControl <= '0'; *-- kühlen*

            PWM1FanInsideValueControl <= b"11111111";

            PWM2FanOutsideValueControl <= b"11111111";

        else

*-- PeltierDirectionControl nicht ändern damit kein ständiger Wechsel stattfindet*

            PWM1FanInsideValueControl <= b"11111111";

            PWM2FanOutsideValueControl <= b"11111111";

        end if;

     end if;

end process tempControl;

*-- Versuch 9: Ansteuerung der 7-Seg-Anzeige*

SevenSegOutput: process (Clock)

begin

    if (rising\_edge(Clock)) then

        if(Reset = '1') then

             ControlTempDiffOut(11 downto 0) <= (others => '0');

             ControlLightDiffOut(11 downto 0) <= (others => '0');

        else

             ControlLightDiffOut(11 downto 0) <= unsigned(SensordataLight);

*-- Fallunterscheidung da unsigned*

             if(unsigned(SensordataTempIn)  > unsigned(SensordataTempOut) ) then *-- innen wärmer*

             ControlTempDiffOut(11 downto 0) <= unsigned(SensordataTempIn) - unsigned(SensordataTempOut);

             ControlTempDiffOut(12) <= '1'; *-- SensordataTempIn >= SensordataTempOut*

             else *-- außen wärmer*

             ControlTempDiffOut(11 downto 0) <= unsigned(SensordataTempOut) - unsigned(SensordataTempIn);

             ControlTempDiffOut(12) <= '0'; *-- SensordataTempIn <=  SensordataTempOut*

             end if;

        end if;

     end if;

end process SevenSegOutput;

end Behavioral;