RTES Quiz 2

1.

#include <ad5933.h>

#include <stdio.h>

#define BUS\_ADDR (0x0D)

#define CONTROL\_REG (0x80)

#define STATUS\_REG (0x8F)

#define REG1 (0x92)

#define REG2 (0x93)

#define CTRL\_TMEASURE (0b10010000)

#define STATUS\_TVALID (0x01)

float getTemperature()

{

    unint8\_t \*bufferArr, tempArr[2];

    int tempVal\_int = 0;

    Start\_I2C(); //starting I2C and initializes all the pins, enables clock etc.

    I2C\_Send\_Start\_Condition(BUS\_ADDR, 1); //addressing the AD5933 device with 1 for writing

    I2C\_Send\_Start\_Condition(CONTROL\_REG, 1); //addressing the control register

    I2C\_Write\_Byte(CTRL\_TMEASURE);            //Setting the Control Register to measure Temperature

    I2C\_Send\_Stop\_Condition();                //stops the sequence for the writing

    I2C\_Send\_Start\_Condition(STATUS\_REG, 0); //addressing the status register to read(0)

    I2C\_RequestRead(\*bufferArr, 1);          //Reading from status register

    if (\*bufferArr == STATUS\_TVALID)

    {

        I2C\_Send\_Start\_Condition(0x90u, 1); //sending the command to measure the temperature

        I2C\_Send\_Stop\_Condition();          //stop sequence for the writing process

        I2C\_Send\_Start\_Condition(BUS\_ADDR, 0); //addressing the AD5933 device with 0 for reading

        I2C\_Send\_Start\_Condition(REG1, 0); //addressing the internal register 1 where the temp. data is stored, with 0 for reading

        I2C\_RequestRead(\*tempArr[0], 1);   //Reading from temperature register 1

        I2C\_Send\_Start\_Condition(REG2, 0); //addressing the internal register 2 where the temp. data is stored, with 0 for reading

        I2C\_RequestRead(\*tempArr[1], 1);   //Reading from temperature register 2

        I2C\_Send\_Stop\_Condition();

        tempVal\_int = (tempArr[0] << 8 | tempArr[1]) & 0x1FFF; // Convert raw temp to integer

        // This conditional statement is used to check the sign of 5th bit

        if ((tempArr[0] & (1 << 5)) == 0)

        {

            return tempVal\_int / 32.0;

        }

        else

        {

            return (tempVal\_int - 16384) / 32.0;

        }

    }

    else{

        return -1000.0; //PLACEHOLDER value can be used

    }

}

int main(){

    // calls the getTemperature function to get the temperature of the AD5933 chip

}

2.

void Timer0\_0V\_Handler();

void pinRising\_Handler();

void pinFallingHandler();

uint32\_t elapsed\_time = 0, start\_time = 0, end\_time = 0, tval = 0, top\_counter = 0;

void Timer0\_0V\_Handler()

{

    if (tval == 250)

    {

        tval = 0;

        top\_counter++;

    }

}

void pinFalling\_Handler()

{

    start\_time = tval + top\_counter \* 250;

}

void pinRising\_Handler()

{

    end\_time = tval + top\_counter \* 250;

    elapsed\_time = end\_time - start\_time;

}