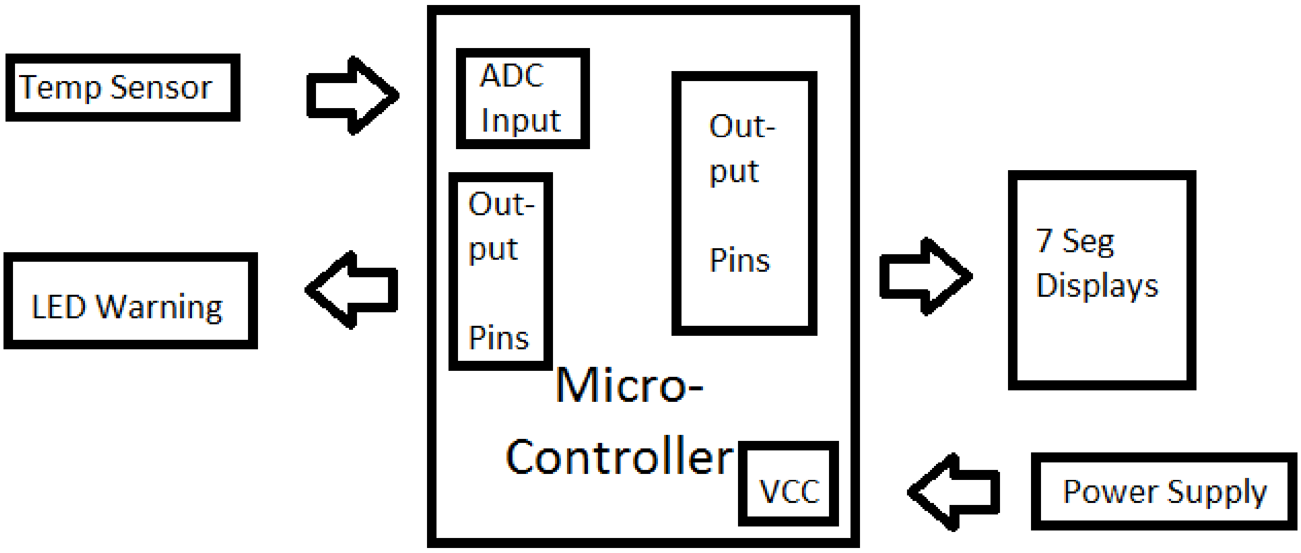
Functional

This water temperature bath device will allow users to monitor the temperature of the bath to ensure the baby is put in the water to prevent burns or sickness that can occur with cold baths. The device is compact which allow user to carry anywhere. The Device will be placed away from water but the sensor will be inside the bath to read temperature of the water and be then displayed on the device. There will be a display to show the temperature of the water and also a warning light to see the status of the water. There is a switch on the power supply to ensure the device isn't always on. A RED led will light up if the power supply is on.

Technical

System Block Diagram



Processor: ATtiny43U

Specs:
8-bit Microcontroller; 16 Programmable I/O Pins; Operation at 1.8v – 5.5V; Low consumption Active mode 400microA @ 3V; EEPROM; inbuilt clock system; Maximum current I/O PIN 40mA; Maximum VCC Current 200mA; Operating Temperature -55 degree to 125 Degree

Reason: The Processor has enough pins to function the device. There are 3 additional pins on the microcontroller which aren't used, which will allow the developer to improve the product by implementing more functions if needed. This micro controller is power efficient, cheap, has enough output current and operates at adequate temperature.

Operation: The microcontroller will control the display and also retrieve data from the temperature sensor through pins. 7 Pins will control the 7 segments, 2 pins will control the display switch, 1 ADC pin will control the temperature signals, 3 Pins will control the Red, Green, and Blue LED.
The voltage supply to the microcontroller is 3.3V. The micro controller will draw 164mA maximum with this device setup, which is less than the microcontroller current input of 200mA
16mA x 7 (*7- Seg display*) + 2mA x2 (*7 Seg display Switch*) + 16mA x3 (*RGB LED*) totalling 164mA.

Cost: $5.58 NZD @ Digi-key

Datasheet: http://www.mouser.com/ds/2/36/atmel_doc8048-611492.pdf

Power Supply: Battery, Voltage Regulator, Led, Power Switch, Decoupling capacitor

Parts:

Battery- Panasonic 9V BSG 6LF22XWA/B Battery
Specs: Zinc Carbon
Reason: Normal house hold 9V Battery which is easy to find and replace. The battery has a large capacity to story energy which allows the device to last longer without constantly replacing the battery. The battery has snap terminals which should cause no rust or deformation.
Operation: The battery is connected in parallel with the components for voltage regulator to help supply the initial voltage and current source for components.
Cost: $3.15 NZD @ Digi-Key
Datasheet: http://www.farnell.com/datasheets/84662.pdf

Voltage Regulator- Voltage XP TR05S3V3
Specs: 3.3V output @ 500mA; Operating temperature 5°C to +35°C;
Reason: This product is a proven to be a reliable component with a 3 year warranty. I have chosen an output of 3.3V so it won't consume as much power and reduce cost on other components like resistor as it doesn't need high amount of resistance.
Operation: This voltage regulator needs to be connected with 2 µF capacitors and also a 6.8 µH inductor. The instructions on how to connect them are on the datasheet below.
Cost: $5.78 NZD @ Digi-Key
Datasheet: http://www.farnell.com/datasheets/1707818.pdf

Red Led- MCL034SRT LED
Specs: 3mm Super Red LED; -40°C to 85°C Operating Temperature; Maximum Reverse voltage of 5V; Maximum reverse current 10µA
Reason: To let the user know if there is power is on, this lets the user know if the power is on so the user can hit the switch to prevent any damages/waste of energy when the device isn't used.
Operation: The LED light will turn on depending if the switch is turned on or off. The LED will require a 68 ohm resister as the LED requires 1.9V (Typical) to power the LED and the common resistor closest to 70 ohm is 68.
$\frac{3.3V-1.9V}{20mA}$ = 70 Ω => 68Ω
Cost: $0.11 NZD @ Digi-Key
Datasheet: http://www.farnell.com/datasheets/1671473.pdf

Power Switch: Toggle Switch 1MB1R1B5M1QE
Specs: Maximum AMP 5A; Maximum Voltage 250V; Silver and Gold Plated; -20°C to 85°C Operating Temperature; Maximum insulation 1000M Ω **;** 40,000 make-and-break cycles at full load;
Reason: This was the cheapest switch I could find on element 14. This switch is compatible to 250V and 5A. This switch is more than adequate to switch off the power. This is placed after the battery to prevent any voltage to going pass. This helps the user save battery and also helps user to prevent damage to the device if there is a break on the circuit. This is gold and silver plated which allows the prevention of rust.
Operation: The user can toggle the switch and turn off the power from the power supply. The led will turn off when the switch is turned off and vice versa.
Cost: $1.19 NZD @ Element14
Datasheet: http://www.farnell.com/cad/1520810.pdf

Decoupling Capacitor: Decoupling Capacitor, Two Capacitor
Specs:
Reason: The reason for putting this on the circuit is to prevent any voltage rippling which can cause a fluctuation of voltage to the microcontroller this can cause the microcontroller to not have the adequate amount of voltage for the pins which can cause malfunction of components or components won't work efficiently.
Operation: The capacitors and decoupling capacitors are connected in-between the supply voltage and ground. The capacitors and decoupling are all connected into parallel. The decoupling capacitor is 10 micro and the capacitors are both 0.1µF.
Cost:
Datasheet:

Display: 7 Segment Red Display anode x2, RGB Display
7 Segment- Red Display Kingbright SA56-11EWA
Specs: Common Anode; Operating Temperature -40°C to 85°C; Power Dissipation 75 mW; Maximum Current 30mA; Low Current operation;
Reason: Two segment Display was used to show the temperature in two digits. This allows the user to see values of the temperature with the display. The reason why I used an anode display instead of cathode as the micro controller cannot supply enough amperage to turn on the display.
Using the voltage from the supply will not stress the microcontroller.
Operation: The segment displays will show the temperature of the water. These displays are controlled by the micro controller via 7 pins for segment and 2 pins each segment display. Two segment display are wire together and then connected to the micro controller pins through resistors. The Resistors I have used is 82ohms as I want 16mA to the 7 segment LED.
$\frac{3.3V-2.5V}{16mA}$= 80ohm => 82 ohm.
I have also used transistors to switch on the segment displays. The signal is outputting to the transistor at 2mA which gains to 200mA (MAX) into the segment display. Also the supply for the segment display is from the supply.
Signal:
$\frac{3.3V}{2mA}$ = 1600 ohms
Gain:
2mA x 100(Gain) = 200mA
Cost: 2.82 NZD x2 @ Digi-Key
Datasheet: http://www.mouser.com/ds/2/216/SA56-11EWA-10234.pdf

RGB Display: KingBright L-154A4SUREQBFZGEW LED
Specs: Common Cathode; Red 2.5V Max ; Blue 4V Max; Green 4V Max; Operating Temperature -40°C - 85°C; Power Dissipation 75-120 Max Depending colour; long life-solid state;
Reason: This component is a compact display that allows the display of the state of the water.
The led will appear blue if the water is cold, the led will appear green at optimum temperature and the led will appear red when the temperature is too hot. The component I have choose is relatively cheap and doesn't require a lot more components to get it functioning.
Operation: The LED is controlled by 3 pins on the micro controller. The microcontroller will turn on the red led when then the temperature exceeds 38°C. The blue led will turn on when temperature falls below 36°C. The green led will turn on in between 36°C and 38°C.
The LEDs will use 16mA per pin. The blue signal won't need any resistance as the output from the microcontroller is 3.3V. The calculation of resistors are bellow.

Red LED:

$\frac{3.3V-1..9V}{16mA}$= 87.5 ohm = 91 ohm (Closest common resistor)

Blue LED:

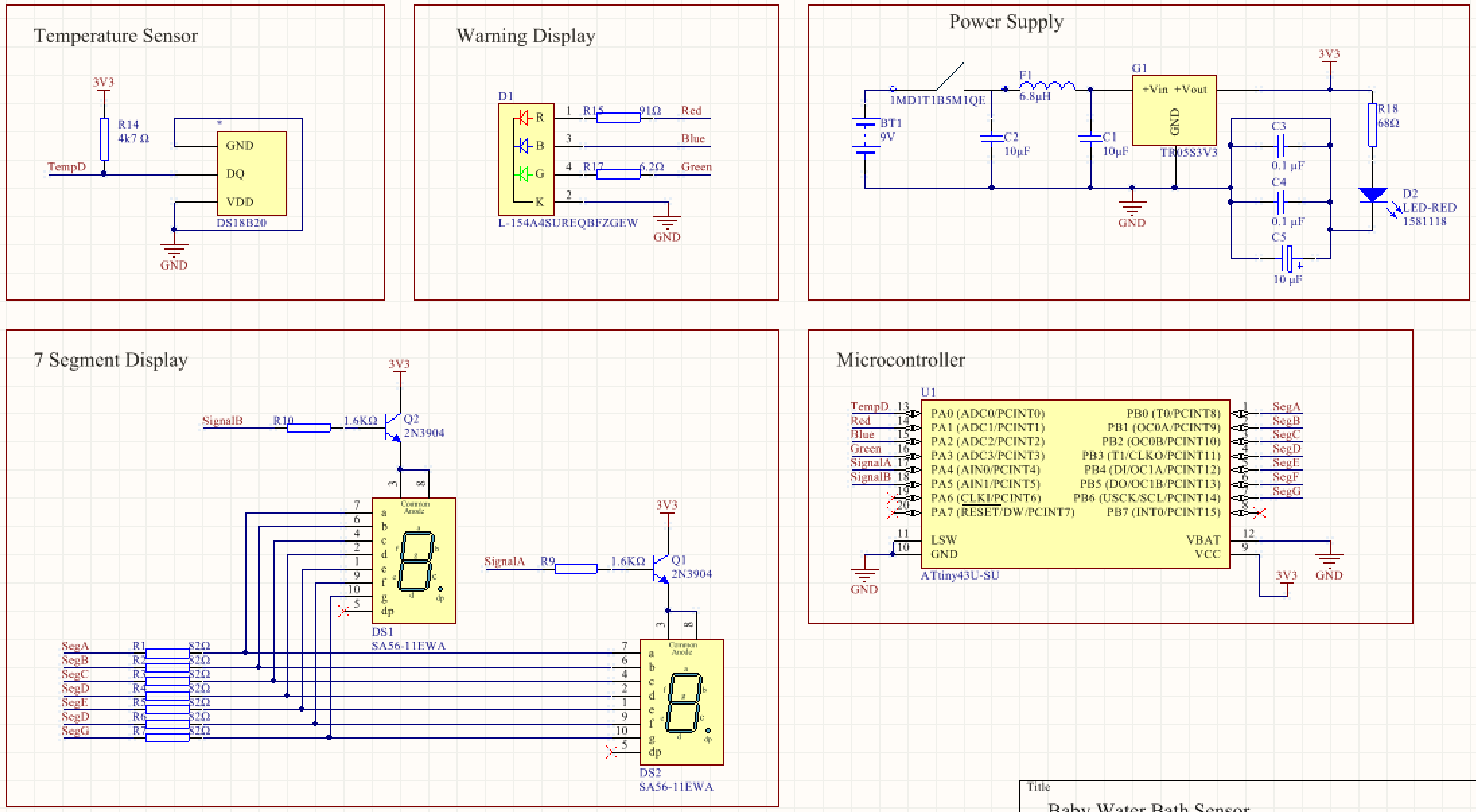
$\frac{3.3V-2.2}{16mA}$ = 0 ohm

Green LED:

$\frac{3.3V-3.2V}{16mA}$ = 6.25ohm => 6.2ohm (Closest common resistor)

Cost: $4.19 NZD @ Digi-Key
Datasheet: http://www.farnell.com/datasheets/1683536.pdf

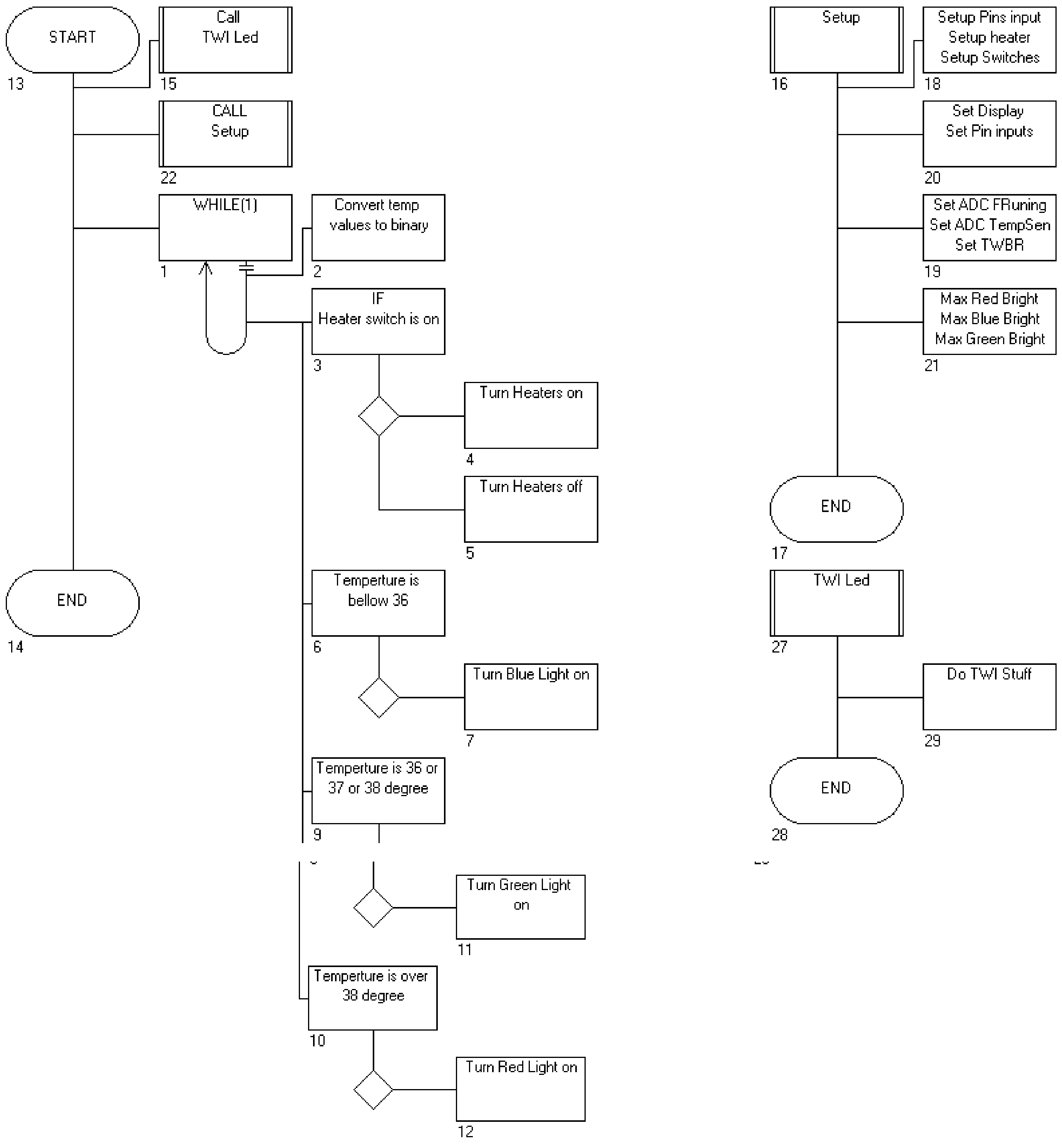
Analogue inputs – Temperature Sensor
Temperature Sensor: DS18B20
Specs: 8 Bit, Parasite Power, -10 C to 85 C , Power supply range 3V-5V, 1 wire Communication , 0.5C Accuracy,
Reason: The reason for picking this sensor over another is due to accuracy. This sensor is accurate to the 0.5C. Since the optimum baby bath temperature is 37 to 38 C. If the accuracy is 2 degrees out it would cause the reading to be not accurate and cause harm to the baby. The sensor is easily wired using just one pin for the data. This allow less external components. This allows wiring more sensors easily if you want to add additional sensors around the bath (find the average temperature at different position of the bath). This sensor is waterproof which allows the user to be able to place it into the water without causing any damage to the device.
Operation: The water temperature is placed in the water to retrieve the temperature values. The VDD pin must be grounded to enable parasite mode. DQ pin wires into an ADC pin to receive data to be converted and displayed on the 7 segment display. GND is wired into the ground. While using parasite mode a 4400 ohm resistor must be placed in between the supply voltage to the DQ which is stated on the datasheet below.
Cost: $4.70 NZD @ Digi-Key
Datasheet: http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf



Title Baby Water Bath Sensor		
Size A4	Number Assignment One	Revision
Date: 19/08/2015		Sheet 1of 1
File: H:\Embedded System 2015 Sem2\...\Assignment 1.SchDoc		Drawn By:

Matthew Chu-1088515

Structure Diagram:



C CODE

```
/*
 *
 *
 * Created: 16/08/2015 12:06:56 p.m.
 *  Author: wgb8191
 */


#define F_CPU 8000000UL
#include <avr/io.h>

#define Temperture ADCW                                 //Sets the ADC value as Temperture
#define heaterSwitch PINA & (1<<0)      //Set pin0 as heater switch
#define HeaterOn (PORTB |= (1<<7))              //Turns Heaters on for testing
#define HeaterOff (PORTB &= ~(1<<7))    //Turns Heaters on for testing
#define display PORTC                                   //Define display as output for degrees
#define char hex;
#define char ten;
#define char ones;




char TWIwrite(char address, char ref_addr , char data); //Calls the TWI for light colours
void setup(void);


int main(void)
{
        setup();

   while(1)
   {

//                      Calculations:
//                      5v/1024(Revolutions)
//                      50mv / 5mv = 10

                        //Converting values
                        hex  = Temperture/10;
                        ten = hex/10;
                        ones = hex%10;
                        result = (ten<<4)+(ones);

                        display = result; // Displays temperture on display


                        if(heaterSwitch)//Turns Heater On when switch is on
                        {
                                HeaterOn;
                        }
                        else
                                HeaterOff;


                        if(hex <= 35){ // Bellow 36
                        TWIwrite(0b11001100, 0b00000011, 0b00110000); //Blue light turns on when it's bellow 36
degree
                        }

                        if(hex == 36 || hex == 37 ||hex = 38) {  //37-38
```

```
                    TWIwrite(0b11001100, 0b00000011, 0b00000011); //Green light turns on when it's 36-38
degrees
                    }

                    if(hex >= 39){//39degree+
                        TWIwrite(0b11001100, 0b00000011, 0b00001100);//Red light turns on when its
above 38 degrees
                    }
                }



        }

        char TWIwrite(char address, char reg_addr, char data)
        {
                char x, error = 0;

                TWCR = (1<<TWINT)|(1<<TWSTA |(1<<TWEN));
                while(!(TWCR & (1<<TWINT)));
                if((TWSR & 0xF8)!= 0x08)
                    error= 1;
                else{
                    TWDR = address;
                    TWCR = (1<<TWINT)| (1<<TWEN);
                    while(!(TWCR & (1<<TWINT)));
                    if((TWSR & 0xF8)!= 0x18)
                        error = 1;
                    else
                    {
                        TWDR = reg_addr;
                        TWCR = (1<<TWINT)| (1<<TWEN);
                        while(!(TWCR & (1<<TWINT)));
                        if((TWSR & 0xF8)!= 0x28)
                            error = 1;
                        else
                        {
                            TWDR = data;
                            TWCR = (1<<TWINT)| (1<<TWEN);
                            while(!(TWCR & (1<<TWINT)));

                            if((TWSR & 0xF8)!= 0x28)
                                error = 1;
                            else
                            {
                                TWCR = (1<<TWINT)| (1<< TWEN)|(1<<TWSTO);
                                for (x=0; x<50; x++)
                                asm("NOP");
                            }
                        }
                    }
                }
                return(error);
        }



                void setup(void)
                {
                DDRA = 0x00;// Inputs
                DDRB = 0xFF;// Setup for Heater
                DDRE = 0x03;// Using switches for pin
                DDRC = 0xFF;// Setup for Display

                ADCSRA = 0b11100111;// free running adc
```

```
ADMUX =  0b01000011;// left adjusted ADC3 for variable resistor 1

TWBR = 0x20; //How fast it travels default values
TWIwrite(0b11001100, 0b00000000, 0b00111111); //turns all Blue to maximum
TWIwrite(0b11001100, 0b00000001, 0b00111111); //turns all Red to maximum
TWIwrite(0b11001100, 0b00000010, 0b00111111); //turns all Green to maximum
}
```