import com.pi4j.io.i2c.I2CBus;

import com.pi4j.io.i2c.I2CDevice;

import com.pi4j.io.i2c.I2CFactory;

import java.io.IOException;

public class SHT25

{

public static void main(String args[]) throws Exception

{

// Create I2C bus

I2CBus Bus = I2CFactory.getInstance(I2CBus.BUS\_1);

// Get I2C device, SHT25 I2C address is 0x40(64)

I2CDevice device = Bus.getDevice(0x40);

// Send temprature measurement command, NO HOLD master

device.write((byte)0xF3);

Thread.sleep(500);

// Read 2 bytes of data

// temp msb, temp lsb

byte[] data = new byte[2];

device.read(data, 0, 2);

// Convert the data

double cTemp = (((((data[0] & 0xFF) \* 256) + (data[1] & 0xFF)) \* 175.72) / 65536.0) - 46.85;

double fTemp = (cTemp \* 1.8) + 32;

// Send humidity measurement command, NO HOLD master

device.write((byte)0xF5);

Thread.sleep(500);

// Read 2 bytes of data

// humidity msb, humidity lsb

device.read(data, 0, 2);

// Convert the data

double humidity = (((((data[0] & 0xFF) \* 256) + (data[1] & 0xFF)) \* 125.0) / 65536.0) - 6;

// Output data to screen

System.out.printf("Relative Humidity : %.2f %% RH %n", humidity);

System.out.printf("Temperature in Celsius : %.2f C %n", cTemp);

System.out.printf("Temperature in Farhenheit : %.2f F %n", fTemp);

}

}

|  |  |
| --- | --- |
|  | import java.io.IOException; |
|  | import java.util.Date; |
|  | import java.util.Properties; |
|  | import java.util.Timer; |
|  | import java.util.TimerTask; |
|  |  |
|  | import upm\_grove.GroveTemp; |
|  |  |
|  | public class FireAlarm { |
|  |  |
|  | private static AlarmLcd lcdScreen; |
|  | private static AlarmBuzzer buzzer; |
|  | private static GroveTemp temperatureSensor; |
|  | private static int lastTemperature = -1; |
|  | private static int temperatureThreshold; |
|  | private static TimerTask alarmTask; |
|  | private static Timer FireTimer; |
|  | private static boolean alarmTick = true; |
|  | private static boolean isAlertOn = false; |
|  | private static Properties config = new Properties(); |
|  |  |
|  | public static void main(String[] args) { |
|  | // TODO Auto-generated method stub |
|  | loadConfigurationFile(); |
|  | initiateSensors(); |
|  | listenToTemperatureChanges(); |
|  | } |
|  |  |
|  |  |
|  | /\*\* |
|  | \* load configuration file and retrieve temperature threshold |
|  | \*/ |
|  | private static void loadConfigurationFile() { |
|  | // TODO Auto-generated method stub |
|  | try { |
|  | // Load configuration data from `config.properties` file. Edit this file |
|  | // to change to correct values for your configuration |
|  | config.load(FireAlarm.class.getClassLoader().getResourceAsStream("config.properties")); |
|  | temperatureThreshold =Integer.parseInt(config.getProperty("TEMPERATURE\_THRESHOLD")); |
|  | } catch (IOException e) { |
|  | e.printStackTrace(); |
|  | } |
|  | } |
|  |  |
|  |  |
|  | /\*\* |
|  | \* when alarm is on this method will be called multiple times to simulate an alarm with the |
|  | \* LCD screen and buzzer. |
|  | \*/ |
|  | private static void handleFireAlertTick() { |
|  | if(alarmTick){ |
|  | lcdScreen.setLcdColor("white"); |
|  | buzzer.stopBuzzing(); |
|  | } |
|  | else{ |
|  | lcdScreen.setLcdColor("red"); |
|  | buzzer.buzz(); |
|  | } |
|  | alarmTick = !alarmTick; |
|  | } |
|  |  |
|  | /\*\* |
|  | \* instantiate sensor helper objects |
|  | \*/ |
|  | private static void initiateSensors() { |
|  | // TODO Auto-generated method stub |
|  | lcdScreen = new AlarmLcd(); |
|  | buzzer = new AlarmBuzzer(5); |
|  | temperatureSensor = new GroveTemp(0); |
|  | } |
|  |  |
|  | /\*\* |
|  | \* run periodic task which checks change in temperature and alerts a fire if detected |
|  | \* or stops the alarm if the fire is out. |
|  | \*/ |
|  | private static void listenToTemperatureChanges() { |
|  | // TODO Auto-generated method stub |
|  | Timer temperatureCheckTimer = new Timer(); |
|  | temperatureCheckTimer.schedule(new TimerTask() { |
|  |  |
|  | @Override |
|  | public void run() { |
|  | // TODO Auto-generated method stub |
|  | int currentTemperature = temperatureSensor.value(); |
|  | lcdScreen.displayMessageOnLcd("temperature: " + currentTemperature, 0); |
|  | System.out.println("value: " + currentTemperature); |
|  | if(lastTemperature < temperatureThreshold && currentTemperature >= temperatureThreshold && !isAlertOn){ |
|  | alertFire(); |
|  | } |
|  | if(lastTemperature >= temperatureThreshold && currentTemperature < temperatureThreshold && isAlertOn){ |
|  | stopAlertingFire(); |
|  | } |
|  | lastTemperature = currentTemperature; |
|  | } |
|  |  |
|  |  |
|  | },0,2000); |
|  | } |
|  |  |
|  | /\*\* |
|  | \* start alerting a fire. |
|  | \*/ |
|  | private static void alertFire() { |
|  | // TODO Auto-generated method stub |
|  | Utils.sendMessageWithTwilio("fire alarm", config); |
|  | Utils.notifyAzure((new Date().toString()), config); |
|  |  |
|  | System.out.println("fire!"); |
|  | isAlertOn = true; |
|  | lcdScreen.displayMessageOnLcd("fire detected", 1); |
|  | lcdScreen.setLcdColor("red"); |
|  | buzzer.buzz(); |
|  | alarmTask = new TimerTask() { |
|  |  |
|  | @Override |
|  | public void run() { |
|  | // TODO Auto-generated method stub |
|  | handleFireAlertTick(); |
|  | } |
|  | }; |
|  | FireTimer = new Timer(); |
|  | FireTimer.schedule(alarmTask, 0,250); |
|  | } |
|  |  |
|  | /\*\* |
|  | \* stop alerting a fire. |
|  | \*/ |
|  | private static void stopAlertingFire() { |
|  | // TODO Auto-generated method stub |
|  | isAlertOn = false; |
|  | System.out.println("fire stopped!"); |
|  | lcdScreen.displayMessageOnLcd("fire stopped", 1); |
|  | lcdScreen.setLcdColor("white"); |
|  | buzzer.stopBuzzing(); |
|  | FireTimer.cancel(); |
|  | } |
|  |  |
|  | } |

\*SAMPLE CODE SOURCE\*

<https://github.com/intel-iot-devkit/how-to-code-samples/blob/master/fire-alarm/java/src/FireAlarm.java>

<https://www.instructables.com/id/Temperature-and-Humidity-Monitoring-Using-SHT25-Se/> (IMPORTANT)

Write java program to read temperature and relative humidity data.

Tutorial to setup sensor with the raspberry pi (IMPORTANT after collecting raspberry pi material)

<https://www.youtube.com/watch?v=i-wnEwMVtpA>

What kind of API(eg. Java library) to draw chart/flame graph

<https://netflixtechblog.com/java-in-flames-e763b3d32166>

**Exploration**

Show sensor data to android app firebase console

<https://www.youtube.com/watch?v=UlJvG2nebCg>

**Note that we are not using Arduino, this is just a reference**

Arduino and raspberry pi

<https://oscarliang.com/raspberry-pi-arduino-connected-i2c/>

Programming an Arduino from Raspberry Pi

<https://www.youtube.com/watch?v=mfIacE-SPvg>

**Backup cloud service for Bentley**

Cloud Service to store sensor data

<https://grovestreams.com/>