

```

1 /*-----
2 * main.cpp
3 *
4 * A simple program to poll three Dallas DS18B20 sensor and provide an output
5 * to the serial interface as a CSV format, with option for heat control.
6 *
7 * Author:
8 * Created: 23 Jul 2015
9 * Modified: 24 Jul 2015
10 *
11 * Code based from https://github.com/milesburton/Arduino-Temperature-Control-Library/blob/master/examples/Multiple/Multiple.pde
12 *
13 * Notes:
14 *   OneWire http://playground.arduino.cc/Learning/OneWire
15 *
16 * Code will check for correct communications three DS18B20 sensors.
17 * Any error will be reported and functionality for reading sensor disabled.
18 *
19 * The sensors will be polled every 10 seconds for their temperature and printed out.
20 *
21 * Output to the serial interface should be as follows:
22 *   1234567890, 21.5, 22.7, 23.9
23 *   1234567891, 21.5, 22.6, 23.8
24 *   and so on ....
25 *
26 * The first item is the internal timer millis() and is only used
27 * as a reference and can be removed or altered as required.
28 *
29 *
30 * Heating:
31 *   The heating control relay is connected to pin 4.
32 *   The heatin is controlled by the temperature of sensor_1 and is operated
33 *   when the temperature is below the HEAT_ON setpoint.
34 *   The heat control is removed when the temperature is above
35 *   the HEAT_OFF setpoint.
36 *
37 */
38 #include <OneWire.h>
39 #include <DallasTemperature.h>
40
41 #define ONE_WIRE_BUS 3 // Data wire is plugged into pin 3 on the Arduino:
42 #define TEMPERATURE_PRECISION 9 // sensor precision set to 9 bit, can be higher, 10, 11 or 12 bits:
43 #define HEAT_RELAY 4 // heat relay is connected to pin 4:
44 #define HEAT_ON 18.5 // heat on temperature:
45 #define HEAT_OFF 24.0 // heat off temperature:
46
47 //-----
48 // const type variable = value; comment
49
50 boolean heat_relay = false; // state of heat relay:
51 uint32_t lastScan = 0; // last time sensors are scanned:
52 const uint32_t periodScan = 10000; // rate period to scan sensors, every 10000mSec, or 10 seconds:
53
54 OneWire oneWire(ONE_WIRE_BUS); // oneWire instance to communicate with OneWire devices
55 DallasTemperature sensors(&oneWire); // Pass oneWire reference to Dallas Temperature.
56
57 struct sensor_t { // structure to hold info for each sensor:
58     DeviceAddress address; // sensor unique address:
59     float value; // actual temperature value from sensor in °C:
60     boolean enabled; // set if sensor is enabled and online:
61     uint16_t error_count; // error count if needed:
62 };
63
64 sensor_t sensor_1 = {0x28, 0xBC, 0xBF, 0xE9, 0x03, 0x00, 0x00, 0x7A}; // instance and address for sensor 1:
65 sensor_t sensor_2 = {0x28, 0x09, 0xA9, 0xC0, 0x03, 0x00, 0x00, 0x95}; // instance and address for sensor 2:
66 sensor_t sensor_3 = {0x25, 0x04, 0xBC, 0xC7, 0x03, 0x00, 0x00, 0x85}; // instance and address for sensor 3:
67
68 /*-----
69 * void setup()
70 * Initial setup of board:
71 */
72 void setup() {
73
74 }
75
76 /*-----
77 * void loop()
78 * Mission Control:
79 * Routine to get sensor data and print out at a scan rate of every 10000mSec:
80 */
81 void loop() {
82     if (millis() - lastScan > periodScan) { // call routine at a set rate:
83         sensors.requestTemperatures(); // initiate all sensors to do conversion:
84
85         Serial.print(millis()); // print something first such as time if needed:
86         Serial.print(", "); // insert delimiter:
87
88         if (sensor_1.enabled)
89         {
90             sensor_1.value = sensors.getTempC(sensor_1.address); // get sensor 1 temperature:
91             Serial.print(sensor_1.value, 1);
92             Serial.print(", "); // insert delimiter:
93         }
94
95         if (sensor_2.enabled)
96         {
97
98         }
99     }
100 }

```

main.cpp

```
129     sensor_1.value = sensors.getTempC(sensor_1.address); // get sensor 2 temperature:
130     Serial.print(sensor_2.value, 1);
131     Serial.print(", "); // insert delimiter:
132 }
133
134 if (sensor_3.enabled)
135 {
136     sensor_1.value = sensors.getTempC(sensor_1.address); // get sensor 3 temperature:
137     Serial.print(sensor_3.value, 1);
138     Serial.print("\n"); // if needed, print a newline:
139 }
140
141 if (sensor_1.enabled) // this sensor used for relay control:
142 {
143     if ((sensor_1.value <= HEAT_ON) && !heat_relay) // if temp below setpoint and not heating:
144     { // then turn on heat:
145         digitalWrite(HEAT_RELAY, true);
146         heat_relay = true;
147     }
148
149     if ((sensor_1.value >= HEAT_OFF) && heat_relay) // if temp above setpoint and heating:
150     { // then turn off heating:
151         digitalWrite(HEAT_RELAY, false);
152         heat_relay = false;
153     }
154 }
155
156 lastScan = millis();
157 }
158 }
159
```