

```

1  /*
2  * This is the .h file for the SHT35D Tempearture
3  * and relative humidity sensor.
4  */
5
6  #ifndef SHT35D
7  #define SHT35D
8  #define MAX_READ_COUNT 5
9  #define ADDR_SHT 0x45
10
11  #include <Arduino.h>
12
13  //List of Commands for SHT35D Sensor:
14  typedef enum {
15      SHT3XD_CMD_READ_SERIAL_NUMBER = 0x3780,
16
17      SHT3XD_CMD_READ_STATUS = 0xF32D,
18      SHT3XD_CMD_CLEAR_STATUS = 0x3041,
19
20      SHT3XD_CMD_HEATER_ENABLE = 0x306D,
21      SHT3XD_CMD_HEATER_DISABLE = 0x3066,
22
23      SHT3XD_CMD_SOFT_RESET = 0x30A2,
24
25      SHT3XD_CMD_CLOCK_STRETCH_H = 0x2C06,
26      SHT3XD_CMD_CLOCK_STRETCH_M = 0x2C0D,
27      SHT3XD_CMD_CLOCK_STRETCH_L = 0x2C10,
28
29      SHT3XD_CMD_POLLING_H = 0x2400,
30      SHT3XD_CMD_POLLING_M = 0x240B,
31      SHT3XD_CMD_POLLING_L = 0x2416,
32
33      SHT3XD_CMD_ART = 0x2B32,
34
35      SHT3XD_CMD_PERIODIC_HALF_H = 0x2032,
36      SHT3XD_CMD_PERIODIC_HALF_M = 0x2024,
37      SHT3XD_CMD_PERIODIC_HALF_L = 0x202F,
38      SHT3XD_CMD_PERIODIC_1_H = 0x2130,
39      SHT3XD_CMD_PERIODIC_1_M = 0x2126,
40      SHT3XD_CMD_PERIODIC_1_L = 0x212D,
41      SHT3XD_CMD_PERIODIC_2_H = 0x2236,
42      SHT3XD_CMD_PERIODIC_2_M = 0x2220,
43      SHT3XD_CMD_PERIODIC_2_L = 0x222B,
44      SHT3XD_CMD_PERIODIC_4_H = 0x2334,
45      SHT3XD_CMD_PERIODIC_4_M = 0x2322,
46      SHT3XD_CMD_PERIODIC_4_L = 0x2329,
47      SHT3XD_CMD_PERIODIC_10_H = 0x2737,
48      SHT3XD_CMD_PERIODIC_10_M = 0x2721,
49      SHT3XD_CMD_PERIODIC_10_L = 0x272A,
50
51      SHT3XD_CMD_FETCH_DATA = 0xE000,
52      SHT3XD_CMD_STOP_PERIODIC = 0x3093,
53
54      SHT3XD_CMD_READ_ALR_LIMIT_LS = 0xE102,
55      SHT3XD_CMD_READ_ALR_LIMIT_LC = 0xE109,
56      SHT3XD_CMD_READ_ALR_LIMIT_HS = 0xE11F,
57      SHT3XD_CMD_READ_ALR_LIMIT_HC = 0xE114,
58      SHT3XD_CMD_WRITE_ALR_LIMIT_HS = 0x611D,
59      SHT3XD_CMD_WRITE_ALR_LIMIT_HC = 0x6116,
60      SHT3XD_CMD_WRITE_ALR_LIMIT_LC = 0x610B,
61      SHT3XD_CMD_WRITE_ALR_LIMIT_LS = 0x6100,
62
63      SHT3XD_CMD_NO_SLEEP = 0x303E,
64  } SHT31D_Commands;
65
66  // List of repeatability options for SHT35D:
67  typedef enum {
68      SHT3XD_REPEATABILITY_HIGH,
69      SHT3XD_REPEATABILITY_MEDIUM,

```

```

70     SHT3XD_REPEATABILITY_LOW,
71 } SHT31D_Repeatability;
72
73 // List of modes:
74 typedef enum {
75     SHT3XD_MODE_CLOCK_STRETCH,
76     SHT3XD_MODE_POLLING,
77 } SHT31D_Mode;
78
79 // List of frequency choices
80 typedef enum {
81     SHT3XD_FREQUENCY_HZ5,
82     SHT3XD_FREQUENCY_1HZ,
83     SHT3XD_FREQUENCY_2HZ,
84     SHT3XD_FREQUENCY_4HZ,
85     SHT3XD_FREQUENCY_10HZ
86 } SHT31D_Frequency;
87
88 // List of errors:
89 typedef enum {
90     SHT3XD_NO_ERROR = 0,
91
92     SHT3XD_CRC_ERROR = -101,
93     SHT3XD_TIMEOUT_ERROR = -102,
94
95     SHT3XD_PARAM_WRONG_MODE = -501,
96     SHT3XD_PARAM_WRONG_REPEATABILITY = -502,
97     SHT3XD_PARAM_WRONG_FREQUENCY = -503,
98     SHT3XD_PARAM_WRONG_ALERT = -504,
99
100 // Wire I2C translated error codes
101
102     SHT3XD_WIRE_I2C_DATA_TOO_LOG = -10,
103     SHT3XD_WIRE_I2C_RECEIVED_NACK_ON_ADDRESS = -20,
104     SHT3XD_WIRE_I2C_RECEIVED_NACK_ON_DATA = -30,
105     SHT3XD_WIRE_I2C_UNKNOW_ERROR = -40
106 } SHT31D_ErrorCode;
107
108 // List of statuses:
109 typedef union {
110     uint16_t rawData;
111     struct {
112         uint8_t WriteDataChecksumStatus : 1;
113         uint8_t CommandStatus : 1;
114         uint8_t Reserved0 : 2;
115         uint8_t SystemResetDetected : 1;
116         uint8_t Reserved1 : 5;
117         uint8_t T_TrackingAlert : 1;
118         uint8_t RH_TrackingAlert : 1;
119         uint8_t Reserved2 : 1;
120         uint8_t HeaterStatus : 1;
121         uint8_t Reserved3 : 1;
122         uint8_t AlertPending : 1;
123     };
124 } SHT31D_RegisterStatus;
125
126 struct SHT31D {
127     /*
128     * Structure for SHT31D
129     * t - temperature
130     * rh - relative humidity
131     * error - error of type SHT31D_ErrorCode
132     */
133     float t;
134     float rh;
135     SHT31D_ErrorCode error;
136 };
137
138 class ClosedCube_SHT31D {

```

```

139  /*
140  * Class definition for ClosedCube_SHT31D
141  */
142  public:
143      ClosedCube_SHT31D();
144
145      bool start_sht(void);
146      void run_sht(void);
147      float get_t_ave(void);
148      float get_rh_ave(void);
149
150
151      SHT31D_ErrorCode begin(uint8_t address);
152      SHT31D_ErrorCode clearAll();
153      SHT31D_RegisterStatus readStatusRegister();
154
155      SHT31D_ErrorCode heaterEnable();
156      SHT31D_ErrorCode heaterDisable();
157
158      SHT31D_ErrorCode softReset();
159      SHT31D_ErrorCode generalCallReset();
160
161      SHT31D_ErrorCode artEnable();
162
163      uint32_t readSerialNumber();
164
165      SHT31D printResult(String text, SHT31D result);
166      SHT31D readTempAndHumidity(SHT31D_Repeatability repeatability, SHT31D_Mode mode,
167                                uint8_t timeout);
168      SHT31D readTempAndHumidityClockStretch(SHT31D_Repeatability repeatability);
169      SHT31D readTempAndHumidityPolling(SHT31D_Repeatability repeatability, uint8_t
170                                        timeout);
171
172      SHT31D_ErrorCode periodicStart(SHT31D_Repeatability repeatability, SHT31D_Frequency
173                                     frequency);
174      SHT31D periodicFetchData();
175      SHT31D_ErrorCode periodicStop();
176
177      SHT31D_ErrorCode writeAlertHigh(float temperatureSet, float temperatureClear, float
178                                     humiditySet, float humidityClear);
179      SHT31D readAlertHighSet();
180      SHT31D readAlertHighClear();
181
182      SHT31D_ErrorCode writeAlertLow(float temperatureClear, float temperatureSet, float
183                                     humidityClear, float humiditySet);
184      SHT31D readAlertLowSet();
185      SHT31D readAlertLowClear();
186
187  private:
188      float t_buf[MAX_READ_COUNT];
189      float rh_buf[MAX_READ_COUNT];
190      bool is_average_taken;
191      int read_count;
192      float t_average;
193      float rh_average;
194
195      SHT31D save_to_buffer(SHT31D result);
196      SHT31D read_sht(void);
197      void calculate_average(void);
198
199      uint8_t _address;
200      SHT31D_RegisterStatus _status;
201
202      SHT31D_ErrorCode writeCommand(SHT31D_Commands command);
203      SHT31D_ErrorCode writeAlertData(SHT31D_Commands command, float temperature, float
204                                     humidity);
205
206      uint8_t checkCrc(uint8_t data[], uint8_t checksum);
207      uint8_t calculateCrc(uint8_t data[]);

```

```
202
203     float calculateHumidity(uint16_t rawValue);
204     float calculateTemperature(uint16_t rawValue);
205
206     uint16_t calculateRawHumidity(float value);
207     uint16_t calculateRaWTemperature(float value);
208
209     SHT31D readTemperatureAndHumidity();
210     SHT31D readAlertData(SHT31D_Commands command);
211     SHT31D_ErrorCode read(uint16_t* data, uint8_t numOfPair);
212
213     SHT31D returnError(SHT31D_ErrorCode command);
214 };
215
216 #endif
217
```