1. Überarbeitung

Nachdem nunmehr auch bei mir die Temperaturmessung funktioniert, hier unverbindlich und ohne Gewähr eine Zusammenfassung der Erfahrung mit den hygrosens-Modulen zur Ermittlung der relativen Feuchtigkeit / Temperatur.

Die TWI-Adresse

Eine TWI-/I2C-Adresse ist zunächst 7 Bit lang (also von 0 ... 127).
Zusätzlich wird ein Read/Write Bit angehängt.
Das R/W-Bit entscheidet, ob - aus der Sicht des Masters - eine Schreibvorgang (R/W = 0) oder ein Lesevorgang (R/W = 1) ausgeführt werden soll.
Insgesamt werden zur Adressierung eines TWI-Teilnehmers 8 Bit = 1 Byte erwartet.

Bit.7	Bit.6	Bit.5	Bit.4	Bit.3	Bit.2	Bit.1	Bit.0
A.6	A.5	A.4	A.3	A.2	A.1	A.0	R/W

Die HYT-xxx haben die 7-Bit Adresse 0b010 1000 (könnte man lesen als 0x28).
Diese Adresse muss aber nach links geschoben werden, damit an der richtigen Stelle im niedrigstwertigen Bit Platz für das Read/Write Bit bleibt.

Bit.7	Bit.6	Bit.5	Bit.4	Bit.3	Bit.2	Bit.1	Bit.0
0	1	0	1	0	0	0	R/W

Als Hexadezimalwert könnte man hier lesen 0x50 (wenn das R/W Bit nicht gesetzt ist) bzw. 0x51 (wenn das R/W-Bit gesetzt ist).

Die Daten aus dem Modul auslesen

Um Daten auszulesen, schickt man zuerst die TWI-Adresse mit R/W-Bit = 0 an das Modul.

Bit.7	Bit.6	Bit.5	Bit.4	Bit.3	Bit.2	Bit.1	Bit.0
0	1	0	1	0	0	0	0

Zur Kontrolle sollte man prüfen, ob der Slave (das Modul) mit einem ACK antwortet (er zieht kurz die SDA-Leitung auf Masse und lässt sie dann wieder "los").
Ist keine Reaktion an SDA zu erkennen, dann liegt ein Fehler vor, der beseitigt werden muss.

Wird ein ACK empfangen, dann muss man dem Modul Zeit gewähren, um die Sensoren für Feuchtigkeit und Temperatur auszulesen, in kalibrierte Werte umzurechnen und das Ergebnis in seinem Ausgangspuffer bereitzustellen (ca. 100ms).

Anschließend wird die TWI-Adresse mit gesetztem R/W-Bit gesendet:

Bit.7	Bit.6	Bit.5	Bit.4	Bit.3	Bit.2	Bit.1	Bit.0
0	1	0	1	0	0	0	1

und es werden vier Byte aus dem Modul ausgelesen.
Diese Bytes mögen in einem Array namens Buffer[] abgelegt sein.

Die Daten interpretieren

In Buffer[0] signalisieren die beiden höchstwertigen Bits:

Bit.7	Bit.6	Bit.5	Bit.4	Bit.3	Bit.2	Bit.1	Bit.0
CmdMode	Busy	x	x	x	x	x	x

Bit.7 sollte nie gesetzt sein, sofern man nicht absichtlich den 'Command Mode' aktiviert hat.
Ein gesetztes Bit.6 zeigt an, dass das Modul noch keine neuen Daten bereitgestellt hat (z.B., weil es noch mit der Konversion beschäftigt ist).

Sind weder Bit.7 noch Bit.6 gesetzt, dann können man die Daten ausgewertet werden:

Die Raw-Daten für Relative Feuchtigkeit

Jeweils zwei Bytes müssen zu einem 16 Bit-Wert zusammengefügt werden.
Dazu benötigt man eine entsprechende Variable vom Typ unsigned int.

raw_RH16 = Buffer[0] * 256 + Buffer[1]; oder
raw_RH16 = Buffer[0] < 8 | Buffer[1];

Beide Schreibweisen führen zum gleichen Ergebnis.

Die Raw-Daten für Temperatur

Es gilt dieselbe Überlegung wie bei der relativen Luftfeuchtigkeit, die Bytes Buffer[2] und Buffer[3] werden zu einem 16 Bit-Wert in der Variablen raw_T16 zusammengeführt:

raw_T16 = Buffer[2] * 256 + Buffer[3]; oder
raw_T16 = Buffer[2] < 8 | Buffer[3];

Der 14 Bit lange "Messwert" steht nun linksbündig in der Variablen raw_T16.
Der Hersteller empfiehlt, die beiden niedrigswertigen Bits zu entfernen (maskieren).
Darauf kann man verzichten, wenn man den Wert rechtsbündig ausrichtet, also 2x nach rechts shiftet (was identisch ist mit der Division durch 2^2 = 4).

Umrechnung der Raw-Daten in relative Feuchtigkeit und Temperatur

Die Feuchtigkeit wird als Wert zwischen 0 ... 2^14 übermittelt und muss umgerechnet werden auf den Wertebereich 0 ... 100%:

RH = (raw_RH16 * 100) / 2^14

Bei der Umrechnung der Temperatur gilt grundsätzlich dieselbe Überlegung, allerdings muss noch eine Konstante subtrahiert werden:

raw_T16 hat den Wertebereich 0 ... 2^14 und entspricht dem Temperaturbereich -40 bis + 125°.

T = ((raw_T16 * 165) / 2^14) - 40

Der Faktor 165 ergeben sich aus dem Wertebereich der Tempertur (-40 bis 125 = 165).

Die Subtraktion ist erforderlich, um negative Temperaturwerte zu erhalten.
Dazu muss die verwendete Variable vom Typ signed werden.

Will man noch eine Nachkommastelle erhalten, dann multipliziert man mit 1650 und fügt vor der letzten Ziffer ein Dezimaltrennzeichen ein.
Auf diese Weise kann mit Ganzzahlen gerechnet werden.

Anstatt den Messwert raw_T16 zuerst rechtsbündig auszurichten und anschließend durch 2^14 zu dividieren, kann man beide Divisionen auch zusammenfassen zu:

T = ((raw_T16 * 165) / 2^16) - 40

Zur Sicherheit sollten dann allerdings die oben erwähnten beiden niedrigstwertigen Bits der raw_T16 maskiert werden.

Wer mit einem 8-Bit Microcontroller arbeitet, der sollte die Umrechnungen nicht mit Fließkomma-Arithmetik sondern mit Ganzzahlen ausführen: das 'kostet' wesentlich weniger Speicherplatz.

18.05.11

Michael S.