SHTxx
Humidity & Temperature Sensmitter

Application Note
CRC

1 Introduction

A CRC checksum is calculated over the whole transmission. If a CRC mismatch is detected, the SHTxx should be reset (command “00011110”) and the measurement should be repeated.

2 Theory

CRC stands for Cyclic Redundancy Check. It is one of the most effective error detection schemes and requires a minimal amount of hardware.
For in-depth information on CRC we recommend the comprehensive: “A painless guide to CRC error detection algorithms” available at: http://www.repairfaq.org/filipg/LINK/F_crc_v3.html

The polynomial used in the SHTxx is: $x^8+ x^5+ x^4$. The types of errors that are detectable with this polynomial are:
1. Any odd number of errors anywhere within the transmission.
2. All double-bit errors anywhere within the transmission.
3. Any cluster of errors that can be contained within an 8-bit “window” (1-8 bits incorrect).
4. Most larger clusters of errors.

The CRC register initializes with the value of the lower nibble of the status register (“0000’$s_3s_2s_1s_0$”, default “00000000”). It covers the whole transmission (command and response bytes) without the acknowledge bits. See the datasheet SHT11 on page 4 for an example of CRC readout.
The receiver can perform the CRC calculation upon the first part of the original message and then compare the result with the received CRC- 8. If a CRC mismatch is detected, the SHTxx should be reset (command “00011110”) and the measurement should be repeated.

This application note will cover two methods for checking the CRC. The first “Bitwise” is more suited for hardware or lowlevel implementation while the later “Bytewise” is the preferred method for more powerful microcontroller solutions.

2.1 Bitwise

With the bitwise method, the receiver copies the structure of the CRC generator in hard- or software.
An algorithm to calculate this could look like this:

1) Initialise CRC Register to low nibble of status register (reversed ($s_0s_1s_2s_3$'0000))
2) Compare each (transmitted and received) bit with bit 7
3) If the same: shift CRC register, bit0='0'
   else: shift CRC register and then invert bit4 and bit5, bit0='1' (see figure 1)
4) receive new bit and go to 2)
5) The CRC value retrieved from the SHTxx must be reversed (bit 0 = bit 7, bit 1=bit 6 … bit 7 = bit 0) and can then be compared to the final CRC value.(2)

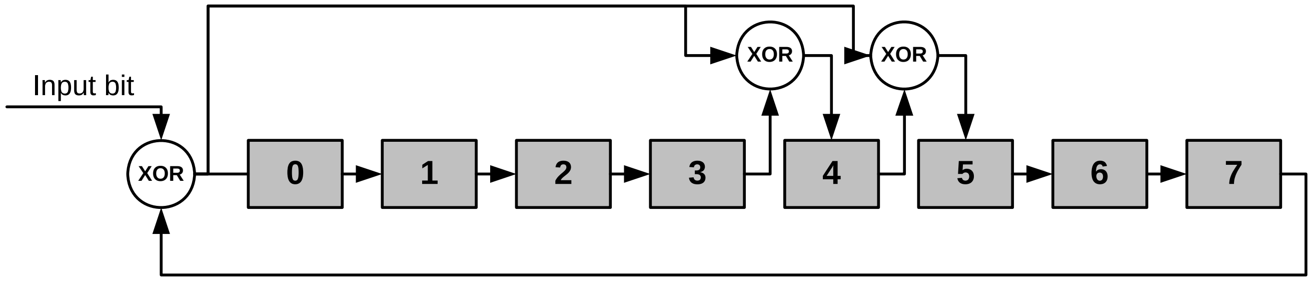

Figure 1 Internal structure of the SHTxx CRC-8 generator

2.1.1 *Example for bitwise*

Example 2: RH Measurement (as example in datasheet)

Input bit's	bit7 … bit0	0x	dec	Comment
	0000'0000			Start value see below(1)
0	0000'0000	00	0	1st bit of command
0	0000'0000	00	0	2nd bit of command
0	0000'0000	00	0	…
0	0000'0000	00	0	
0	0000'0000	00	0	
1	0011'0001			CRC EXOR polynom
0	0110'0010			
1	**1111'0101**	**F5**	**245**	**CRC after command**
0	1101'1011			1st byte (MSB) of measurement
0	1000'0111			
0	0011'1111			
0	0111'1110			
1	1100'1101			
0	1010'1011			
0	0110'0111			
1	**1111'1111**	**FF**	**255**	**CRC value**
0	1100'1111			2nd byte (LSB) of measurement
0	1010'1111			
1	0101'1110			
1	1000'1101			
0	0010'1011			
0	0101'0110			
0	1010'1100			
1	**0101'1000**	**58**	**88**	**Final CRC value**

Example 1: readout of status register containing 0x40

Input bit's	bit7 … bit0	0x	dec	Comment
	0000'0000			Start value see below(1)
0	0000'0000	00	0	1st bit of command
0	0000'0000	00	0	2nd bit of command
0	0000'0000	00	0	…
0	0000'0000	00	0	
0	0000'0000	00	0	
1	0011'0001			CRC EXOR polynom
1	0101'0011			
1	**1001'0111**	**97**	**151**	**CRC after command**
0	0001'1111			1st bit (MSB) of status register
1	0000'1111			
0	0001'1110			
0	0011'1100			
0	0111'1000			
0	1111'0000			
0	1101'0001			
0	**1001'0011**	**93**	**147**	**Final CRC value**

---

(1) Low nibble only, whole byte reversed (Statusregister = [$s_7s_6s_5s_4$'$s_3s_2s_1s_0$] -> Startvalue = [$s_0s_1s_2s_3$'0000])
(2) This is different to other CRC implementations

2.2 Bytewise

With this implementation the CRC data is stored in a 256 byte lookup table.
Perform the following operations:

1. Initialize the CRC register with the value of the lower nibble of the value of the status register (reversed ($s_0s_1s_2s_3$‘0000)). (default ‘00000000’ = 0)
2. XOR each (transmitted and received) byte with the previous CRC value. The result is the new byte that you need to calculate the CRC value from.
3. Use this value as the index to the table to obtain the new CRC value.
4. Repeat from 2.) until you have passed all bytes through the process.
5. The last byte retrieved from the table is the final CRC value.
6. The CRC value retrieved from the SHTxx must be reversed (bit 0 = bit 7, bit 1=bit 6 … bit 7 = bit 0) and can then be compared to the final CRC value.(2)

2.2.1 256 byte CRC Lookup table

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	49	98	83	196	245	166	151	185	136	219	234	125	76	31	46	67	114	33	16	135	182	229	212	250	203	152	169	62	15	92	109
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
134	183	228	213	66	115	32	17	63	14	93	108	251	202	153	168	197	244	167	150	1	48	99	82	124	77	30	47	184	137	218	235
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
61	12	95	110	249	200	155	170	132	181	230	215	64	113	34	19	126	79	28	45	186	139	216	233	199	246	165	148	3	50	97	80
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
187	138	217	232	127	78	29	44	2	51	96	81	198	247	164	149	248	201	154	171	60	13	94	111	65	112	35	18	133	180	231	214
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
122	75	24	41	190	143	220	237	195	242	161	144	7	54	101	84	57	8	91	106	253	204	159	174	128	177	226	211	68	117	38	23
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
252	205	158	175	56	9	90	107	69	116	39	22	129	176	227	210	191	142	221	236	123	74	25	40	6	55	100	85	194	243	160	145
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
71	118	37	20	131	178	225	208	254	207	156	173	58	11	88	105	4	53	102	87	192	241	162	147	189	140	223	238	121	72	27	42
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
193	240	163	146	5	52	103	86	120	73	26	43	188	141	222	239	130	179	224	209	70	119	36	21	59	10	89	104	255	206	157	172

---

(2) This is different to other CRC implementations

2.2.2 Code example for lookup table

The following procedure calculates the CRC-8. The result accumulates in the variable CRC.

```
Var
CRC : Byte;
Procedure calc_CRC(X: Byte);

Const
CRC_Table : Array[0..255] of Byte = (
0,  49, 98, 83, 196,245,166,151,185,136,219,234,125,76, 31, 46, 67, 114,33, 16,
135,182,229,212,250,203,152,169,62, 15, 92, 109,134,183,228,213,66, 115,32, 17,
63, 14, 93, 108,251,202,153,168,197,244,167,150,1,  48, 99, 82, 124,77, 30, 47,
184,137,218,235,61, 12, 95, 110,249,200,155,170,132,181,230,215,64, 113,34, 19,
126,79, 28, 45, 186,139,216,233,199,246,165,148,3,  50, 97, 80, 187,138,217,232,
127,78, 29, 44, 2,  51, 96, 81, 198,247,164,149,248,201,154,171,60, 13, 94, 111,
65, 112,35, 18, 133,180,231,214,122,75, 24, 41, 190,143,220,237,195,242,161,144,
7,  54, 101,84, 57, 8,  91, 106,253,204,159,174,128,177,226,211,68, 117,38, 23,
252,205,158,175,56, 9,  90, 107,69, 116,39, 22, 129,176,227,210,191,142,221,236,
123,74, 25, 40, 6,  55, 100,85, 194,243,160,145,71, 118,37, 20, 131,178,225,208,
254,207,156,173,58, 11, 88, 105,4,  53, 102,87, 192,241,162,147,189,140,223,238,
121,72, 27, 42, 193,240,163,146,5,  52, 103,86, 120,73, 26, 43, 188,141,222,239,
130,179,224,209,70, 119,36, 21, 59, 10, 89, 104,255,206,157,172);

Begin
CRC := CRC_Table[X xor CRC];
End;
```

3 Revision history

Date	Revision	Changes
December 30, 2001	0.9 (Preliminary)	Initial revision
February 18, 2001	1.01	
February 27, 2001	1.02	corrected bug in CRC register init. (byte must be reversed)
May 16, 2002	1.03	emphasize that command to SHTxx is also in CRC
Oct. 17, 2003	1.04	Changed download link
December 16, 2003	1.05	Improved bitwise example table
May 25, 2005	1.06	Changed company address
Oct 3, 2006	1.07	Sensirion Inc. address added

The latest version of this document and all application notes can be found at:
www.sensirion.com/humidity

Headquarters and Sales Office

SENSIRION AG
Laubisrütistr. 50
CH-8712 Stäfa ZH
Switzerland

Phone: + 41 (0)44 306 40 00
Fax: + 41 (0)44 306 40 30
e-mail: info@sensirion.com
http://www.sensirion.com/

SENSIRION Inc
Westlake Pl. Ctr. I, suite 240
2801 Townsgate Road
Westlake Village, CA 91361
USA

Phone: 805-409 4900
Fax: 805-435 0467
e-mail: michael.karst@sensirion.com
http://www.sensirion.com/