

## SD01-SPI 温湿度模块

### 特性:

- 温度和相对湿度传感器
- 预先计算的温度和湿度输出, 无需额外计算
- 可以计算露点
- 完全校准, 数字输出
- 良好的长期稳定性
- 无需额外的组件
- 超低功耗
- 完全可互换
- 尺寸小
- 自动断电



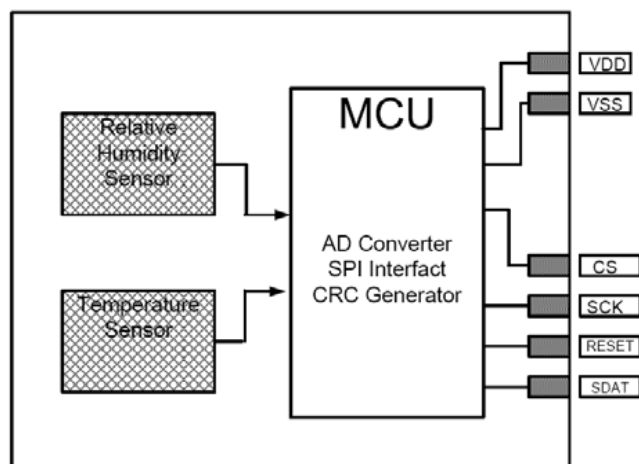
### 产品简介:

该SD01 是一款包括检测温度和相对湿度的传感器功能模块, 包括一个SPI 接口(主模式), 可以直接读出温度和湿度。数字输出是预先计算好的, 在单片机中并无需再作外计算。该系统用了两个传感器元件: 高精度型NTC温度传感器和来自日本的电阻式相对湿度传感器。系统采用了一个非常独特相对湿度的计算算法专利, 并通过良好的温度调节补偿机制以保证准确相对湿度输出。在整个温度范围(0-50C)都可以保证此高精度湿度读数。

### 适用产品

- HVAC 产品
- 消费类产品
- 气象站
- 加湿机
- 抽湿机
- 测试与测量
- 数据记录
- 白色家电

### 模块组成



1. 性能规格

参数	条件	Min	Typ	Max	单位
<b>湿 度</b>					
分辨率				1	%
重复性			1		%
精度 不确定性	0-50度	0	±3	±5	%
互换性		完全直接互换			
非线性			1	2	%
范围	0-50度	18		98	%
响应时间			60		S
湿滞			1	2	%
长期稳定性			2		%/yr
<b>温 度</b>					
分辨率			0.1		°C
可重复性			0.1	0.2	°C
范围		-40		70	°C
精度	25		±0.5	±1.0	°C
响应时间	Delta T=1.0		60		S

2. 传感器管脚连接方式

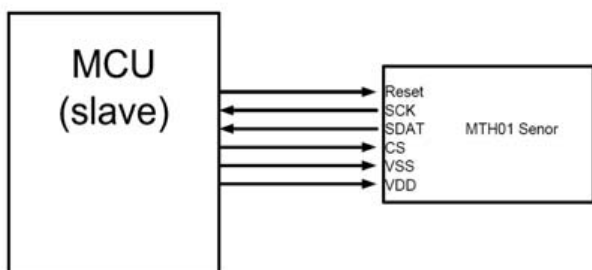


图1 典型应用

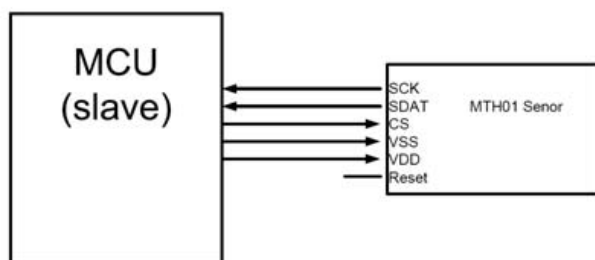


图2 减少I/O口的应用方式

2.1 电源接脚

SD01传感器模块需要2.0到5.5伏的电压供应。上电后，传感器需要20ms时间来进行模块初始化。初始化完成后，传感器开始自动测量，若CS脚被拉低，测量数据自动输出。

电源管脚上需连接一个10-100nF的电容进行退耦。如果在电磁干扰比较厉害的环境下使用该模块，强烈建议增加一个10uF钽电容，以保护传感器免被干扰。

2.2 串行接口

采用串行接口优化设计，是为了能够更加方便读取和减少IO口用量。应用工程师应牢记IO口的特性，对电流消耗有要求的场合是至关重要的。

RESET，输入端，采用50k上拉电阻连接，在一般的应用过程该输入端应该保持高电平除非系统确实需要复位。

CS，输入端，用100k上拉电阻连接。下降过程将触发模块并把先前测到的数据发送出去，然后再开始温度和湿度的测量。只要CS拉低，测量，发数据的过程就会一直持续下去。

OUTPUT管脚SDAT和SCK，是CMOS输出模式。因此，这两个管脚和外部的MCU连接时，应该确保是在输入模式，不用放入电阻以避免高电流

### 2.2.1 CS

CS是传感器的开关，并通过SCK和SDAT触发输出之前测量的温湿度数据。通过拉低CS就会立刻将上次测量数据发送出来。数据输出后，一个内部的AD转换循环将自动开始，随即开始一个新温度湿度数据的测量。如果AD转换完成后CS还保持低电平，新测量数据将会通过数据线输出直到CS线变高电平为止，此时终止数据输出和AD转换过程。当CS为高电平，系统进入睡眠模式，以降低电力消耗。

### 2.2.2 SCK

SCK使MCU和传感器模块间保持同步。它采用1.5千赫的频率输出，这是高速和低速单片机系统都能理想工作的频率。数据输出结束，SCK端变成低电平。

### 2.2.3 SDAT数据输出端

SDAT是连接到微控制器的数据输出端。当SCK线变高电平，SDAT输出有效；数据输出完毕，SDAT线将保持为低电平。

### 2.2.4 RESET复位端

在正常应用期间，复位端应该保持高电平。如果当CS线拉低50ms后仍然不能检测到数据输出，可以拉低复位脚来进行系统复位。在高可靠性要求的应用环境中，建议使用此复位功能。

## 3. 时序图

开始数据读取时，将CS线拉低激活传感器模块并通过数据线开始转换数据。数据时序如下方所示：

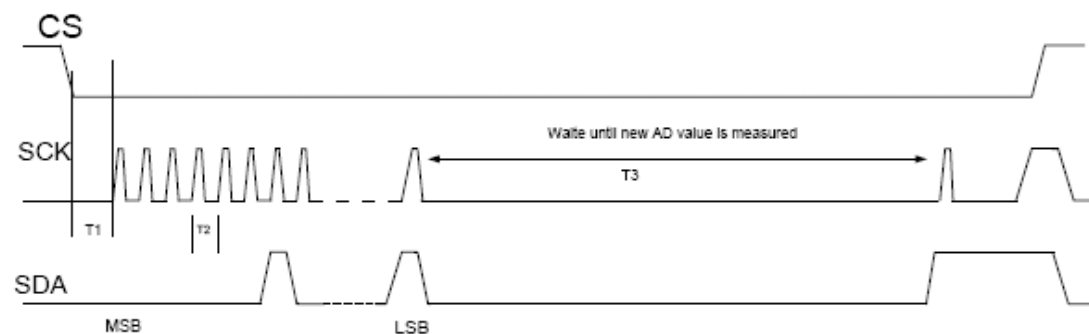


图3: 总线时序图

T1: 名义上为10ms      T2: 900uS      T3: 200mS

一旦CS拉高，传感器模块将会立即终止数据传输进程，并且在进入断电模式前自动开始新的测量。这样做的目的是因为一旦由于拉低CS pin就会启动一个新的读取命令，数据随时准备好被转换。这有利于减少从传感器模块接收反馈的数据前所需等待的时间。一点不足之处是接收的数据是先前测量的，并不是由当前这瞬间的数据。

数据输出位数量先从MSB的温度（2字节）数据开始，随后是一字节的湿度和一字节的CRC。



图4: 数据结构说明

**3.1 把输出数据成温湿度数值:**

温度值增加了一个40C 的抵消值以避免负温度符号，再乘以10。因此，可以通过减掉0190H (Hex) 得到准确的温度。

举例说明:

Data stream: 00000010101100110101000100000000

T\_offset=0x02B3=691 T\_real=691-400=291=29.1C

RH=0x51=81% CRC=0x7F

**3. 直流参数**

参数	条件	Min	Typ	Max	单位
直流供电		2	3	5.5	V
当前电流供应	AD测量		100	200	uA
	数据转换		0.2		mA
	待机		0.2		uA
低水平电压输出		0		20%	Vdd
高水平电压输出		80%		100%	Vdd
低水平输入电压	负极	0%		30%	Vdd
高水平输入电压	正极	70%		100%	Vdd
泄漏电流			1		uA
输出高电流	80%VDD		10		mA
输出低电流	20%VDD		10		mA

**4. 封装形式**

模块尺寸为12×20mm，6个引脚:

VDD VSS Reset SDAT SCK CS.

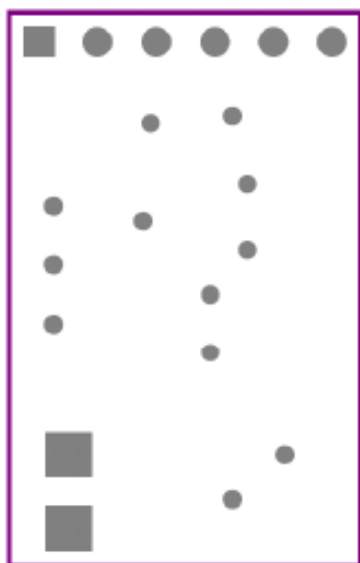


图5: 传感器模块外形图

## 6. CRC

CRC是指循环冗余检查，这是最有效的一个错误检测方案，对硬件的要求很小。

在SD01中的多项式使用是： $x^8+x^5+x^4$ 。以下几种类型的错误用该多项式都可以检测出来：

1. 在传输范围内任何地方任何单数的误差。
2. 在传输范围内任何地方所有双比特错误。
3. 任何群集的错误可以被一个8位的“窗口”包含（1-8位不正确）。
4. 最大群集错误。

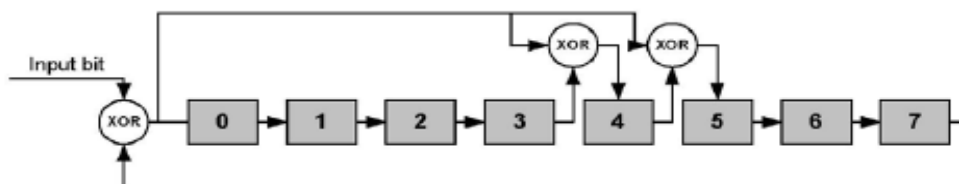
接收存储器可以记录通过源信息的计算得到的CRC的数据，然后与接收到的CRC-8的结果进行对比。如果一个CRC被检测到是不匹配的，则SD01需重置，重复进行新的测试。

该应用指南包括两种检测CRC的方法。第一种适合于硬件或低一级检测的“位运算符”，而第二种的“字节运算符”则是更高级的微控制器解决方案的首选。

### 6.1位运算符

在运算符计算过程，接收器复制CRC中的硬件或软件结构  
一个完整的解决计算问题的运算过程可以看成是这样的：

- 1) 初始化CRC寄存器为0
- 2) 用CRC的第7位与每个（发送和接收）位比较
- 3) 如果相同：bit0='0'，移位CRC寄存器；bit0='1'，移位CRC寄存器并对bit4和bit5取反，（见图1）
- 4) 接收新的位，继续第二步
- 5) 将最后一个字节调好后，在CRC寄存器的数据就被计算出来了。



#### 6.1.1位运算符举例

例子1: 0x05-0x09-0x31 CRC 计算举例

Input bit's	bit7 ... bit0	0x	dec	Comment
	0000'0000			Start value
0	0000'0000	00	0	1 <sup>st</sup> bit of command
0	0000'0000	00	0	2 <sup>nd</sup> 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			1 <sup>st</sup> 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			2 <sup>nd</sup> 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

例子2: 0x40 CRC 计算举例

Input bit's	bit7 ... bit0	0x	dec	Comment
	0000'0000			Start value see below <sup>(1)</sup>

0	0000'0000	00	0	st
				1 bit of command
0	0000'0000	00	0	nd
				2 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			st
				1 bit (MSB) of status register
1	0000'1111			
0	0001'1110			
0	0011'1100			
0	0111'1000			
0	1111'0000			
0	1101'0001			

## 6.2 字节运算符

CRC数据的存储在以下256字节的查询表内。

执行下列步骤：

1. CRC注册值初始化为0
  2. 和上次得到的CRC值XOR(异或)处理，得到的结果可用来计算CRC数值。
  3. 使用该值作为表中的指针以获取新的CRC值。
  4. 从步骤2开始重复，直到所有字节都通过整个检测。
  5. The last byte retrieved from the table is the final CRC value.
- 从该表得到的最后一个字节就是最终的CRC的值。

### 6.2.1 256字节CRC查询表

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	36	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

### 6.2.2 询表代码举例

下面是CRC-8计算举例，CRC作为变量可累加。

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;

## 7. 重要提示:

勿将该产品作为安全或紧急避险系统或其他由于损坏而会导致人身伤害的产品中。未能遵守这项指示，可能会导致死亡或致命的伤害。

### 7.1 ESD 预防措施

为防止公共服务电子化有关的损害和/或退化，处理装置时采取一般**ESD** 预防措施。

### 7.2 保修

至于是否适合其产品在任何特定场合的用途，我们不作任何保修，陈述或保证，我们也不承担任何由应用或使用任何产品或电路而产生的责任，特别声明任何和所有责任，包括但不限于相应而生或附带的损害赔偿。