**史上最详细IIC教程**

2018年07月20日 17:17:07 [河间老王](https://me.csdn.net/Lingdongtianxia) 阅读数：9203更多

所属专栏： [通讯协议详解](https://blog.csdn.net/column/details/19051.html)

 版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/Lingdongtianxia/article/details/81135456

**基础知识介绍**

时序图

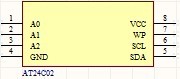
本文所有时序图均来自AT24C02的芯片手册

IIC

     I2C 总线是一种串行数据总线，只有二根信号线，一根是双向的数据线SDA，另一根是时钟线SCL，两条线可以挂多个设备。 IIC设备（绝大多数）里有个固化的地址，只有在两条线上传输的值等于IIC设备的固化地址时，其才会作出响应。通常我们为了方便把IIC设备分为主设备和从设备，基本上谁控制时钟线（即控制SCL的电平高低变换）谁就是主设备。

AT24C02

AT24C02是一个2K位串行CMOS E2PROM,有一个16字节页写[缓冲器](https://www.baidu.com/s?wd=%E7%BC%93%E5%86%B2%E5%99%A8&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)。该器件通过IIC[总线接口](https://baike.baidu.com/item/%E6%80%BB%E7%BA%BF%E6%8E%A5%E5%8F%A3)进行操作，有一个专门的[写保护](https://baike.baidu.com/item/%E5%86%99%E4%BF%9D%E6%8A%A4)功能, 如果WP管脚连接到Vcc，所有的内容都被写保护只能读。当WP管脚连接到Vss or GND 或悬空允许器件进行正常的读/写操作。



AT24C02的存储容量为2K bit，内容分成32页，每页8Byte，共256Byte，操作时有两种[寻址方式](https://baike.baidu.com/item/%E5%AF%BB%E5%9D%80%E6%96%B9%E5%BC%8F)：芯片寻址和片内子地址寻址。

（1）芯片寻址：AT24C02的芯片地址为1010，其地址控制字格式为1010A2A1A0R/W。其中A2，A1，A0可编程地址选择位，A2，A1，A0[引脚](https://baike.baidu.com/item/%E5%BC%95%E8%84%9A)接高、低电平后得到确定的三位编码，与1010形成7位编码，即为该器件的[地址码](https://baike.baidu.com/item/%E5%9C%B0%E5%9D%80%E7%A0%81)。A0、A1、A2引脚是器件地址输入端，这些输入脚用于多个器件级联时设置器件地址，当这些脚悬空时默认值为0，通过改变A2，A1，A0的引脚状态，共有8种组合，所以一个IIC总线最大可级联8个器件。R/W为芯片读写控制位，该位为0，表示芯片进行写操作。在下面的分析及程序中，只有一个AT24C02器件，A2，A1，A0全部悬空，即全部为0，地址控制字为1010000\*，\*为R/W位。

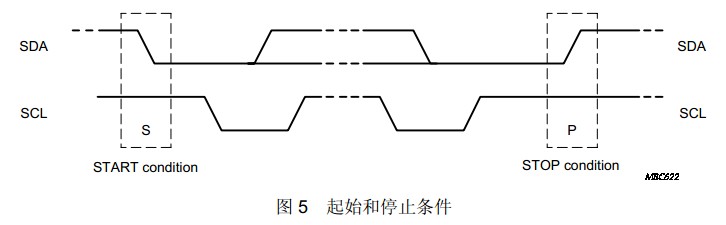
（2）片内子地址寻址：芯片寻址可对内部256B中的任一个进行读/写操作，其寻址范围为00~FF，共256个寻址单位。

主要C语言程序声明

1. sbit sda = P2^0; *// I2C接口SDA定义*
2. sbit scl = P2^1; *// I2C接口SCL定义*
3. void flash() *// 延时子函数*
4. {
5. ; ; ;
6. }
8. void delay(unsigned char i) *// 延时程序*
9. {
10. for( j=i ; j>0 ; j-- )
11. for( k=125; k>0; k-- );
12. }

**IIC通信时序图及子程序讲解**

初始化、起始和停止



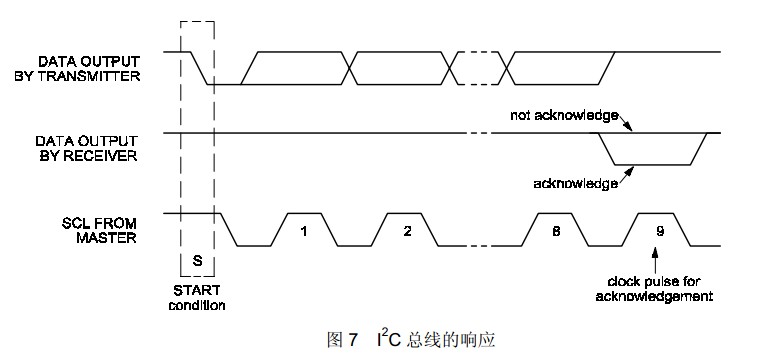
初始化：IIC的初始化为SDA和SCL均为高。

开始信号：处理器让SCL时钟保持高电平，然后让SDA数据信号由高变低就表示一个开始信号。同时IIC总线上的[设备检测](https://www.baidu.com/s?wd=%E8%AE%BE%E5%A4%87%E6%A3%80%E6%B5%8B&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)到这个开始信号它就知道处理器要发送数据了。

停止信号：处理器让SCL时钟保持高电平，然后让SDA数据信号由低变高就表示一个停止信号。同时IIC总线上的设备检测到这个停止信号它就知道处理器已经结束了数据传输，我们就可以各忙各个的了，如休眠等。

1. void x24c08\_init()*// 2402初始化子程序*
3. {
4. scl = 1;*//首先把时钟线拉高*
5. flash();*//延时函数，后面不再注释*
6. sda = 1;*//在SCL为高的情况下把SDA拉高，实际上是在IIC总线上发送了一个停止信号*
7. flash();
8. }
10. void start()*// 启动I2C总线*
12. {
13. sda = 1;*//确保SDA线为高电平*
14. flash();
15. scl = 1; *//确保SCL高电平*
16. flash();
17. sda = 0;*//在SCL为高时拉低SDA线，即为起始信号*
18. flash();
19. scl = 0;*//将SCL线拉低，为后面SDA的电平变化做准备*
20. flash();
21. }
23. void stop()*// 停止I2C总线*
24. {
25. sda = 0;*//确保SDA线为低电平*
26. flash();
27. scl = 1;*//确保SCL高电平*
28. flash();
29. sda = 1;*//在SCL为高电平时拉高SDA线，即为停止信号，此时SDA和SCL均为高，同时完成IIC初始化过程*
30. flash();
31. }

数据传输和响应信号



数据传输：SDA上的数据只能在SCL为低电平期间翻转变化，在SCL为高电平期间必须保持稳定，IIC设备只在SCL为高电平期间采集SDA数据。

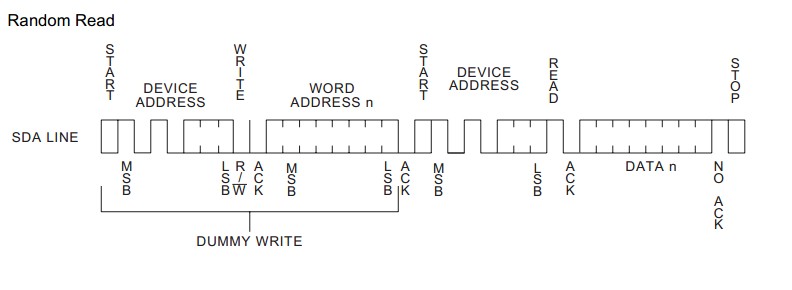
响应信号（ACK）：[单片机](https://www.baidu.com/s?wd=%E5%8D%95%E7%89%87%E6%9C%BA&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)发完8bit数据后就不再驱动总线了（SDA引脚变输入），而SDA和SDL硬件设计时都有上拉电阻，所以这时候SDA变成高电平。那么在第8个数据位，如果外接IIC设备能收到信号的话接着在第9个周期把SDA拉低，那么处理器检测到SDA拉低就能知道外接IIC设备数据已经收到。IIC数据从最高位开始传输(小端传输)。

1. void writex(unsigned char j)*// 向IIC总线上写8位*
2. {
3. unsigned char i,temp;
4. temp = j;
5. for(i=0; i<8; i++)*//循环8次，每次发送一位数据*
7. {
9. temp = temp<<1;*//将temp左移一位，如果temp最高位是1，左移后将产生进位，进位标志CY=1，如果temp最高位是0，左移后不会产生进位，进位标志CY=0*
10. scl = 0;*//将SCL总线拉低，为SDA电平变化做准备*
11. flash();
12. sda = CY;*//根据进位标志CY更改SDA的电平*
13. flash();
14. scl = 1;*//将SCL拉高，告诉IIC器件现在可以检测SDA的电平了*
15. flash();
16. }
18. scl = 0;*//8位数据传输完成后，将SCL拉低*
19. flash();
20. sda = 1;*//将SDL拉高，开始为IIC设备的应答ACK做准备*
21. flash();
22. }
24. void clock()*// I2C总线应答子函数*
25. {
26. unsigned char i = 0;
27. scl = 1;*//将SCL拉高*
28. flash();
29. while((sda==1)&&(i<255)) i++;*//在这里检测SDA的电平变化，若SDA被拉低，即收到应答信号，则退出while循环，若i=255，则按超时处理，同样退出循环*
30. scl = 0;*//将SCL拉低，为后续SDA电平变化做准备*
31. flash();
32. }
34. unsigned char readx()*//从IIC总线读8位*
35. {
36. unsigned char i,j,k=0;
37. scl = 0;*//确保SCL为低，为后续SDA电平变化做准备*
38. flash();
39. sda = 1;*//从单片机角度释放SDA总线，为IIC设备操作SDA线做准备*
40. for(i=0; i<8; i++)*//循环8次，读取8位SDA值*
41. {
42. flash();
43. scl = 1;*//将SCL拉高，此时IIC设备的SDA电平不再变化，为单片机读SDA做准备*
44. flash();
45. if(sda==1) j = 1;
46. else j = 0;*//将SDA电平赋值给j*
47. k = (k<<1)|j;*//将j左移赋值给k，小端传输*
48. scl = 0;*//将SCL拉低，告诉IIC设备现在可以改变SDA的值了，进行下一位的传输*
49. }
50. flash();
51. return(k);*//将8位数据返回，完成一个数据读取过程*
52. }

**AT24C02通信时序图及程序讲解**

单片机从AT24C02读数据的时序图及程序

先对照AT24C02读数据时序图把程序看懂，再结合程序把AT24C02读数据时序图看懂。

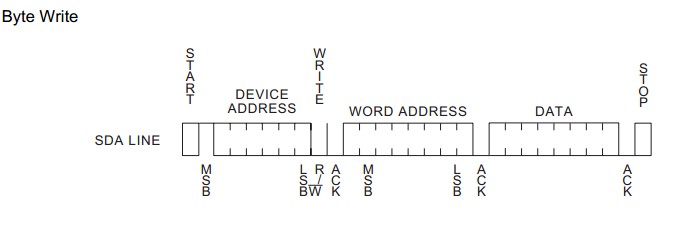
单片机读操作的大致流程：首先单片机发送起始信号，接着使用写操作进行芯片寻址0xa0，接着定位片内子地址address，因为我们要读数据，但是刚才使用的是写操作进行芯片寻址，我们要把R/W位设置成读操作才可以对address处数据进行读操作，所以接下来需要重新发送一个起始位，接着使用读操作进行芯片寻址0xa1，接着单片机从IIC总线中读取8位数据保存到变量中，最后单片机发送一个结束信号完成整个读过程，其中中间夹杂着应答过程。

1. unsigned char x24c08\_read(unsigned char address)  *// 从24c02的地址address中读取一个字节数据*
2. {
3. unsigned char i;
4. start();*//向IIC总线发送一个起始位*
5. writex(0xa0);*// 芯片寻址，向IIC总线写入0xa0,即10100000，R/W位是0，代表写操作*
6. clock();*//等待AT24C02应答*
7. writex(address);*// 片内子地址寻址,将address写入IIC总线，AT24C02下次将操作此地址处的数据*
8. clock();*//等待AT24C02应答*
9. start();*//向IIC总线发送一个起始位*
10. writex(0xa1); *//芯片寻址，向IIC总线写入0xa0,即10100001，R/W位是1，代表读操作*
11. clock();*//等待AT24C02应答*
12. i = readx();单片机从IIC总线中读取8位数据放入i中
13. stop();*//向IIC总线发送一个停止位*
14. delay(10);*//此处延时是为了系统稳定*
15. return(i);*//将读到的数值返回*
16. }

单片机从AT24C02写数据的时序图及程序

先对照AT24C02写数据时序图把程序看懂，再结合程序把AT24C02写数据时序图看懂。

单片机写数据的大致流程：首先发送起始信号，接着使用写操作芯片寻址0xa0，接着进行片内子地址寻址address，接着向address写入info,最后发送停止信号，其中中间夹杂着应答的过程。



1. void x24c08\_write(unsigned char address,unsigned char info)*// 向2402的address地址中写入一个字节数据*
2. {
3. start();*//向IIC总线发送一个起始位*
4. writex(0xa0); *// 芯片寻址，向IIC总线写入0xa0,即10100000，R/W位是0，代表写操作*
5. clock();*//等待AT24C02应答*
6. writex(address); *// 片内子地址寻址,将address写入IIC总线，AT24C02下次将操作此地址处的数据*
7. clock();*//等待AT24C02应答*
8. writex(info);*//将info写入AT24C02的address处*
9. clock();*//等待AT24C02应答*
10. stop();*//向IIC总线发送一个停止位*
11. delay(10); *//此处延时是为了系统稳定*
12. }

**附51单片机读取AT24C02的完整代码**

实现功能：0-99秒的自动计时器，随机关断电源，在通电以后计时器接着断电时的状态继续计时

1. #include<reg51.h>
2. #include<intrins.h>
3. #define uint unsigned int
4. #define uchar unsigned char
5. unsigned char sec; *// 定义计数值，每过1s，sec加1*
6. unsigned int tcnt; *// 定时中断次数*
7. bit write = 0; *// 写2408的标志*
8. sbit sda = P2^0; *// I2C接口SDA定义*
9. sbit scl = P2^1; *// I2C接口SCL定义*
10. sbit dula = P2^6;
11. sbit wela = P2^7;
12. unsigned char j,k;
14. void delay(unsigned char i) *// 延时程序*
15. {
16. for( j=i ; j>0 ; j-- )
17. for( k=125; k>0; k-- );
18. }
20. uchar code table[] = {0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71}; *// 数码管编码*
22. void display(uchar bai\_c,uchar sh\_c) *// 显示程序*
23. {
24. dula = 0;
25. P0 = table[bai\_c];*// 显示百位*
26. dula = 1;
27. dula = 0;
28. wela = 0;
29. P0 = 0x7e;
30. wela = 1;
31. wela = 0;
32. delay(5);
33. dula = 0;
34. P0 = table[sh\_c];
35. dula = 1;
36. dula = 0;
37. wela = 0;
38. P0 = 0x7d;
39. wela = 1;
40. wela = 0;
41. delay(5);
42. }
44. *// 24c02读写驱动程序*
45. void delay1(unsigned char x)
46. {
47. unsigned int i ;
48. for( i=0; i<x; i++);
49. }
51. void flash() *// 延时子函数*
52. {
53. ; ; ;
54. }
56. void x24c08\_init()*// 2402初始化子程序*
57. {
58. scl = 1;
59. flash();
60. sda = 1;
61. flash();
62. }
64. void start()*// 启动I2C总线*
65. {
66. sda = 1;
67. flash();
68. scl = 1;
69. flash();
70. sda = 0;
71. flash();
72. scl = 0;
73. flash();
74. }
76. void stop()*// 停止I2C总线*
77. {
78. sda = 0;
79. flash();
80. scl = 1;
81. flash();
82. sda = 1;
83. flash();
84. }
86. void writex(unsigned char j)*// 写一个字节*
87. {
88. unsigned char i,temp;
89. temp = j;
90. for(i=0; i<8; i++)
91. {
92. temp = temp<<1;
93. scl = 0;
94. flash();
95. sda = CY;
96. flash();
97. scl = 1;
98. flash();
99. }
100. scl = 0;
101. flash();
102. sda = 1;
103. flash();
104. }
106. unsigned char readx()*// 读一个字节*
107. {
108. unsigned char i,j,k=0;
109. scl = 0;
110. flash();
111. sda = 1;
112. for(i=0; i<8; i++)
113. {
114. flash();
115. scl = 1;
116. flash();
117. if(sda==1) j = 1;
118. else j = 0;
119. k = (k<<1)|j;
120. scl = 0;
121. }
122. flash();
123. return(k);
124. }
126. void clock()*// I2C总线应答子函数*
127. {
128. unsigned char i = 0;
129. scl = 1;
130. flash();
131. while((sda==1)&&(i<255)) i++;
132. scl = 0;
133. flash();
134. }

137. unsigned char x24c08\_read(unsigned char address) *// 从24c02的地址address中读取一个字节数据*
138. {
139. unsigned char i;
140. start();
141. writex(0xa0);
142. clock();
143. writex(address);
144. clock();
145. start();
146. writex(0xa1);
147. clock();
148. i = readx();
149. stop();
150. delay(10);
151. return(i);
152. }
154. void x24c08\_write(unsigned char address,unsigned char info)*// 向2402的address地址中写入一个字节数据*
155. {
156. EA = 0;
157. start();
158. writex(0xa0);
159. clock();
160. writex(address);
161. clock();
162. writex(info);
163. clock();
164. stop();
165. EA = 1;
166. delay1(50);
167. }
169. void t0(void) interrupt 1 using 0 *// 定时中断服务函数*
170. {
171. TH0 = (65536-50000)/256; *// 对TH0，TL0赋值*
172. TL0 = (65536-50000)%256;
173. tcnt++; *// 每过250us tcnt加1*
174. if(tcnt == 20)
175. {
176. tcnt = 0; *// 重新再计*
177. sec++;
178. write = 1; *// 1s写一次24c02*
179. if(sec == 100)
180. {
181. sec = 0; *// 定时100s,从0开始计时*
182. }
183. }
184. }
186. void main() *// 主函数*
187. {
188. unsigned char i;
189. TMOD = 0x01; *// 定时器工作在方式1*
190. ET0 = 1;
191. EA = 1; *// 开中断*
192. x24c08\_init();*// 初始化24c08*
193. sec = x24c08\_read(2); *// 读出保存的数据赋予sec*
194. TH0 = (65536-50000)/256;
195. TL0 = (65534-50000)%256 ; *// 使定时器0.05s中断一次*
196. TR0 = 1;*// 启动定时器*
197. while(1)
198. {
199. i = 10;
200. while(i--)
201. {
202. display(sec/10,sec%10);
203. }
205. if(write = 1) *// 判断计时器是否计时1s*
206. {
207. write = 0; *// 清0*
208. x24c08\_write(2,sec); *// 在24c02的地址中写入数据sec*
209. }
210. }
211. }