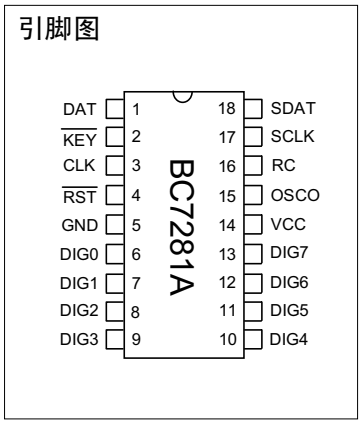
BC7281A
128段LED显示及64键键盘控制芯片

特点

- 可驱动8位或16位数码管显示或64/128只独立LED
- 具有64键键盘接口，内含去抖功能
- 具有2种键盘工作模式，适应不同应用需求
- 独具光柱译码方式，可独立控制两条64段光柱显示
- 段寻址功能便于控制独立 LED
- 16位均可独立控制闪烁属性，闪烁速度软件可调
- 段驱动极性及移位脉冲时序均可控，可配合各种形式的驱动电路
- 键盘部分具有键值锁存功能
- 内部显示寄存器和控制寄存器的内容均可读出
- 2线高速串行接口

引脚图



简述

BC7281A是BC7281芯片的升级换代产品，是8位/16位LED数码管显示及键盘接口专用控制芯片。通过外接移位寄存器(典型芯片如74HC164,74LS595等)，最多可以控制16位数码管显示或128只独立的LED。BC7281A的驱动输出极性及输出时序均为软件可控，从而可以和各种外部电路配合，适用于任何尺寸的数码管。

BC7281A各位可独立按不同的译码方式译码或不译码显示，译码方式显示时小数点不受译码影响，使用更方便；较之BC7281，BC7281A对16个显示位均可以独立地控制闪烁属性；BC7281A内部还具有一闪烁速度控制寄存器，使用者可随时改变闪烁频率；译码方式除了常用的BCD译码等2种方式外，还有专用于光柱(Bar)显示的光柱译码方式，只要送一个字节，就可以完成光柱显示的控制。128段被分成2个各自独立的64段光柱，可以分别控制。另外，128个显示段同时被分配了128个地址，利用段寻址功能可以独立地控制每一个段，便于使用独立的LED。

BC7281A芯片可以连接最多64键(8*8)的键盘矩阵，内部具有去抖动功能。BC7281A的键盘具有2种工作模式，详细情况请参见内文。

BC7281A内部共有26个寄存器，包括16个显示寄存器和10个特殊(控制)寄存器，所

有的操作均通过对这26个寄存器的访问完成。

BC7281A采用高速二线接口与MCU进行通讯，只占用很少的I/O口资源和主机时间。且BC7281A和BC7280/81在软件和硬件上均兼容，BC728X的用户代换为BC7281A时,软件和硬件不需做任何改动。

极限参数

注：超出所列的极限参数有可能造成器件的永久性损坏。

储存温度	-65至+150℃
工作温度	0至+70℃(商业级)
	-40至+85℃(工业级)
任意脚对地电压	-0.3至6.0V

电特性 （除特别说明外,T_A=25℃,Vcc=5.0V）

参数	最小值	典型值	最大值	单位	备注
电源电压	2.7	5.0	5.5	V	
工作电流		2	4	mA	R=3.3K, C=20pF, 所有引脚悬空
输入低电平			0.8	V	
输入高电平	2.0			V	
输出低电平			0.45	V	I_{OL}=5mA
输出高电平	2.4			V	I_{OH}=-5mA
显示扫描周期	8	12	20	ms	R=3.3K, C=20pF
按键响应时间	16	24	40	ms	R=3.3K, C=20pF

引脚说明

名称	引脚号	说明
DAT	1	与MCU串行通讯数据端，为双向数据传输口，作为输出时为漏极开路(Open Drain)输出，需要外接上拉电阻。
KEY	2	键盘有效输出端，低电平有效，检测到有效按键后该引脚变为低电平，并一直保持到键值锁存器内容被读出。
CLK	3	与MCU串行通讯时钟端，下降沿有效。
RST	4	复位端，低电平有效。BC728x内部具有上电复位电路，故一般可将该引脚与Vcc直接相连
GND	5	接地端
DIG0-DIG7	6-13	位驱动输出端，第8-15位显示位驱动与第0-7位共用。同时也是键盘矩阵的‘行’
VCC	14	电源输入端
OSCO	15	RC振荡器输出端，一般应悬空
RC	16	外接RC振荡器端，该引脚连接一RC电路形成振荡，给内部扫描等电路提供时钟。
SCLK	17	外接段驱动用移位寄存器时钟端
SDAT	18	外接段驱动用移位寄存器数据端，输出段驱动数据，低位在前。

内部寄存器

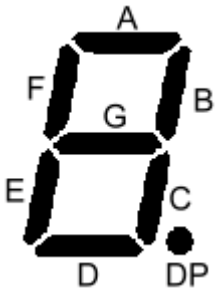
BC7281A芯片内部具有 26 个寄存器，包括16个显示寄存器和10个特殊功能寄存器，共用一段连续的地址，其地址范围是00H-19H，其中00H-0FH为显示寄存器，其余为特殊寄存器。

BC7281A内部寄存器见下表：

地址	内容	上电缺省	允许操作	备注
00H	第0位显示寄存器	FFH	读写	
01H	第1位显示寄存器	FFH	读写	
02H	第2位显示寄存器	FFH	读写	
• • •	• • •	FFH	读写	
0EH	第14位显示寄存器	FFH	读写	
0FH	第15位显示寄存器	FFH	读写	
10H	闪烁开关控制寄存器	FFH	读写	
11H	闪烁速度控制寄存器	40H	读写	
12H	工作模式控制寄存器	00H	读写	
13H	键值锁存器	FFH	只读	
14H	译码寄存器1（BCD）	-	只写	
15H	译码寄存器2（HEX）	-	只写	
16H	译码寄存器3（光柱一）	-	只写	
17H	译码寄存器4（光柱二）	-	只写	
18H	译码寄存器5（段寻址）	-	只写	
19H	扩展闪烁控制寄存器	FFH	读写	BC7281A独有

显示寄存器——地址00H-0FH

显示寄存器共 16 个，分别对应 16 个显示位，其内容为各显示位的映射，各数据位与显示段对应关系如下：



D7	D6	D5	D4	D3	D2	D1	D0
DP	G	F	E	D	C	B	A

用户可以直接修改显示寄存器的内容，达到不译码显示的目的，也可以通过译码寄存器（见后文）间接地改变其内容。显示寄存器为可读写的寄存器，用户既可以直接写入数据，也可以读出其内容。

芯片复位后（上电复位或中途复位），所有显示寄存器被恢复为缺省值（FFH）。当执行了写入译码寄存器（参见后文）的操作后，需要点亮的段被置为‘0’。

闪烁开关控制寄存器——地址10H，19H

闪烁开关控制寄存器控制显示位的闪烁属性。BC7281A的闪烁控制有两种工作模式，由工作模式控制寄存器（地址12H）中的闪烁模式选择位BMS控制（详见后文），缺省的工作模式（BMS=0）与BC7281相同，第8-15位显示与第0-7位共用同一个闪烁控制寄存器10H，第8－15位不能单独控制闪烁属性，控制寄存器数据位与显示位的对应关系如下：

寄存器10H	D7	D6	D5	D4	D3	D2	D1	D0
显示位	DIG7	DIG6	DIG5	DIG4	DIG3	DIG2	DIG1	DIG0
显示位	DIG15	DIG14	DIG13	DIG12	DIG11	DIG10	DIG9	DIG8

DIG0-DIG15分别代表显示位0到显示位15。相应数据位为‘1’时，表示该位正常显示；为‘0’时，该位按闪烁方式显示。

当BMS控制位设为1时，BC7281A工作于扩展闪烁控制模式，这时16个显示位均可以单独控制闪烁状态，10H控制DIG0－DIG7，而扩展闪烁控制寄存器19H控制DIG8－DIG15。对应关系见下表：

寄存器10H	D7	D6	D5	D4	D3	D2	D1	D0
显示位	DIG7	DIG6	DIG5	DIG4	DIG3	DIG2	DIG1	DIG0
寄存器19H	D7	D6	D5	D4	D3	D2	D1	D0
显示位	DIG15	DIG14	DIG13	DIG12	DIG11	DIG10	DIG9	DIG8

闪烁控制寄存器可由用户在任何时刻改写，也可随时将其内容读出。

复位后，闪烁开关控制寄存器10H和19H的初始值均为FFH，即各位均按不闪烁方式显示。

闪烁速度控制寄存器——地址11H

BC7281A的闪烁速度可以控制，通过将不同的值写入闪烁速度控制寄存器，可以改变闪烁显示的闪烁速度，其值的范围是00H-FFH。值越小则闪烁速度越快。

同闪烁开关控制寄存器一样，该寄存器也可由用户任意读写。

闪烁速度控制寄存器复位后的初始值为40H，在该值下，当BC7281A的外接RC的值分别为3.3K和20pF时，闪烁的频率大约为2Hz。

工作模式控制寄存器——地址12H

为了能够和各种驱动电路配合，BC7281A具有多种工作模式，其工作模式由工作控制寄存器控制，其各数据位意义如下：

D7	D6	D5	D4	D3	D2	D1	D0
SCN	X	X	X	BMS	KMS	INV	MOD

MOD：移位寄存器模式控制。当MOD=0时，为普通移位寄存器模式，SDAT端每输出一位数据，SCLK端输出一移位脉冲，8个数据位共输出8个移位脉冲，此种模式适用于一般的移位寄存器，典型器件如74HC164等，故此种模式也称为164模式；当MOD=1时，除了象普通模式一样输出8位数据和8个移位脉冲外，在数据的末尾还多输出一额外的移位脉冲，此种模式适用于带有二级锁存的移位寄存器，典型器件如74HC595等，因此又叫做595模式。

INV：段驱动数据输出极性控制。当INV=0时，各位显示寄存器的数据直接通过移位寄存器输出作为段驱动数据；而当INV=1时，显示寄存器的内容经过反相后才从移位寄存器输出。

KMS：键盘工作模式选择。当KMS=0时，工作模式与BC7281相同，为带锁存的互锁模式，即有效按键发生后KEY即为低电平，直至MCU读取键值后KEY恢复高电平，这期间不响应任何新的按键。而当KMS=1时，KEY电平随按键情况变化，有有效按键时，KEY即为低电平，直至按键释放才恢复高电平，而无论MCU是否读取了键值。

BMS：闪烁控制模式选择。BMS=0时，工作模式与BC7281相同，采用一个闪烁开关控制寄存器（10H）控制各显示位的闪烁属性，第8—15个显示位不能单独控制闪烁属性。当BMS=1时，工作于扩展模式，由10H控制0—7位的闪烁属性，由扩展控制寄存器19H控制8—15位的闪烁属性。

SCN：扫描使能控制。当SCN=0时，扫描被禁止，包括显示扫描和键盘扫描，位驱动DIG0-DIG7输出低电平。当SCN=1时，扫描被使能。

X：无影响。

复位后，工作模式控制寄存器的初始值为00H，扫描被禁止，使用者必须给该控制寄存器设置适当的值后器件才能正常工作。

键值锁存器——地址13H

该寄存器为只读寄存器。当BC7281A检测到有效按键后，其键值将被存储在此寄存器中，$\overline{KEY}$引脚也将输出低电平。在KMS=0时，这种状态将一直保持到锁存的键值被读出，MCU读取13H后，KEY将恢复为高电平，13H的值也恢复为FFH。在上一个键值被读出之前，BC7281A将不会接受新的按键输入；当KMS=1时，KEY的电平随有效按键的情况而变化，存在有效按键时，KEY为低电平，13H中的值为键值，按键释放，KEY恢复为高电平，13H的内容也恢复为FFH。MCU读取键值，不会改变13H中的值，KEY引脚和13H的值

仅取决于有效按键的情况。

相关如果没有按健而执行读键值锁存器的操作，读出的值将为FFH。

键盘矩阵的连接请参阅后文。矩阵中各键的键值见下表：

段＼位	DIG0	DIG1	DIG2	DIG3	DIG4	DIG5	DIG6	DIG7
A	00	01	02	03	04	05	06	07
B	08	09	0A	0B	0C	0D	0E	0F
C	10	11	12	13	14	15	16	17
D	18	19	1A	1B	1C	1D	1E	1F
E	20	21	22	23	24	25	26	27
F	28	29	2A	2B	2C	2D	2E	2F
G	30	31	32	33	34	35	36	37
DP	38	39	3A	3B	3C	3D	3E	3F

BC7281A不接受多个按键同时按下的情况，见下表两种键盘模式的对比：

	KMS=0	KMS=1
无有效按键	KEY=1, 13H=0xFF	KEY=1, 13H=0xFF
一个有效按键发生	KEY=0, 13H=键值	KEY=0, 13H=键值
一个有效按键保持	只要MCU读取键值，即恢复为KEY=1及13H=0xFF, 无论该键是否保持	KEY=0, 13H=键值，不论MCU是否读取键值
一个有效按键释放	不改变，只要MCU没有读取键值，则KEY一直为低电平，13H=键值	KEY恢复高电平，13H恢复0xFF, 无论键值是否已被读取
先有一个键按下，然后第二个键按下	如果第一个键值没有读走，则不相应第二个键	第一个键按下时，KEY=0, 13H=键值，当第二个键按下后，13H变为0xFF,但KEY保持为0
两个键同时从‘关’的状态被按下	KEY=1, 13H=0xFF	KEY=1, 13H=0xFF
从两个键按下的情况恢复成一个按键	如果上一个键值已被读走，则KEY=0，13H=有效按键键值	KEY=0, 13H=有效按键键值

译码寄存器1-5——地址14H-18H

该寄存器负责对输入的数据完成显示译码。译码寄存器不是物理寄存器，使用者并不能实际地向这些地址内写入数据，对这些地址的写入操作的结果是间接地改变显示寄存器的内容。译码寄存器是‘只写’寄存器，只能写入不能读出，对译码寄存器读操作的结果将始终为00H。分别介绍各译码寄存器如下：

1、BCD译码器——地址14H

写入数据的格式如下：

D7	D6	D5	D4	D3	D2	D1	D0
a_3	a_2	a_1	a_0	d_3	d_2	d_1	d_0

其中，a_3:a_0为显示位地址，其取值范围为0-F，对应第0-第15位显示。d3:d0为待译码的数据，其译码方式如下表：

d_3:d_0	d_3	d_2	d_1	d_0	显示
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	2
3	0	0	1	1	3
4	0	1	0	0	4
5	0	1	0	1	5
6	0	1	1	0	6
7	0	1	1	1	7
8	1	0	0	0	8
9	1	0	0	1	9
a	1	0	1	0	-
b	1	0	1	1	E
c	1	1	0	0	H
d	1	1	0	1	L
e	1	1	1	0	P
f	1	1	1	1	(空白)

需要注意的是，通过BCD译码寄存器改变显示内容，不会影响该显示位的小数点段(DP段)的状态，这样对于一般的显示数据刷新，就可以不用考虑小数点的显示，而对于那些将小数点段用作单独的指示灯的用户，这更是一个非常有用的特性；如果用户需要改变小数点的状态，让其点亮或熄灭，可以通过段寻址指令(见后文)。

2、HEX译码器——地址15H

该寄存器数据格式、使用方式同BCD译码器，请参见上文。所不同之处在于译码方式，见下表：

d_3:d_0	d_3	d_2	d_1	d_0	显示
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	2
3	0	0	1	1	3
4	0	1	0	0	4
5	0	1	0	1	5
6	0	1	1	0	6
7	0	1	1	1	7
8	1	0	0	0	8
9	1	0	0	1	9
a	1	0	1	0	A
b	1	0	1	1	b
c	1	1	0	0	C
d	1	1	0	1	d
e	1	1	1	0	E
f	1	1	1	1	F

通过HEX译码器同样不会对小数点产生影响。

3、光柱译码器(一、二)——地址16H、17H

BC7281A将16位显示共128段分为2部分，第一部分为第0-63段(段地址定义请参见后文段寻址部分)，第二部分为第64-127段，2个光柱译码寄存器分别控制二段的译码。这样既可以作为一个128级的整体光柱显示，也可以分作2个64级的光柱，以适应不同的使用需求。译码器一(地址16H)负责第一部分的译码，译码器二(地址17H)负责第二部分。

其数据格式为：

D7	D6	D5	D4	D3	D2	D1	D0
0	d_6	d_5	d_4	d_3	d_2	d_1	d_0

d6:d0为相应部分内地址从低到高所点亮的显示段个数，其取值范围为0000000B(00H，全熄)-1000000B(40H，全亮)。

BC7281A并没有“清屏”或者“全亮”的指令，但使用者可以通过使用光

柱译码方式达到同样的效果。

4、段寻址译码寄存器——地址18H

16位显示共128个显示段，通过段寻址寄存器，可以分别独立控制每一个显示段。BC7281A为每一个显示段定义了一个地址，其定义如下表：

段地址 \ 段位	DP	G	F	E	D	C	B	A
0	07	06	05	04	03	02	01	00
1	0F	0E	0D	0C	0B	0A	09	08
2	17	16	15	14	13	12	11	10
3	1F	1E	1D	1C	1B	1A	19	18
4	27	26	25	24	23	22	21	20
5	2F	2E	2D	2C	2B	2A	29	28
6	37	36	35	34	33	32	31	30
7	3F	3E	3D	3C	3B	3A	39	38
8	47	46	45	44	43	42	41	40
9	4F	4E	4D	4C	4B	4A	49	38
10	57	56	55	54	53	52	51	50
11	5F	5E	5D	5C	5B	5A	59	58
12	67	66	65	64	63	62	61	60
13	6F	6E	6D	6C	6B	6A	69	68
14	77	76	75	74	73	72	71	70
15	7F	7E	7D	7C	7B	7A	79	78

写入段寻址译码寄存器的数据格式如下：

D7	D6	D5	D4	D3	D2	D1	D0
SD	A_6	A_5	A_4	A_3	A_2	A_1	A_0

A_6:A_0为段地址，范围是0000000-1111111，也即00H-7FH，SD为段状态，即写入段寻址寄存器指令执行完后相应显示寄存器位的状态，SD=0，则相应位被置为‘0’（点亮），反之则被置为‘1’（熄灭）。当然，因为BC7281A的段驱动输出的极性是软件可控的，因此显示寄存器位的‘0’造成对应显示段实际是‘开’还是‘关’，还取决于输出的极性及驱动电路的结构。

串行接口

一、基本格式

BC7281A与MCU之间通讯采用2线高速串行接口，二根连线分别是数据线DAT和同步

时钟线CLK，其中DAT为双向数据传输线，BC7281A既用该线从MCU接收数据，也用该线向MCU发送数据。BC7281A的DAT引脚为漏极开路输出（OPEN DRAIN）结构，使用时需要在该线上加一20KΩ左右的上拉电阻。CLK引脚为串行接口的同步时钟，由MCU控制，下降沿有效。

串行接口数据宽度为8位，两个字节为一组，构成一条完整的指令。第一个字节为命令字，第二个字节为数据。字节在传送时高位（MSB）在前。串行接口数据结构如下：

指令字节									数据字节							
D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
R/W	0	0	a_4	a_3	a_2	a_1	a_0		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

指令字节中R/W为读写控制，当R/W=0时，由MCU向BC7281A的内部寄存器内写入数据；当R/W=1时，MCU读出BC7281A内部寄存器的数据。a_0-a_4为目标寄存器的地址，其范围为00H-19H

数据字节即写入或从寄存器读出的数据。写入指令（R/W=0）时，数据由MCU传向BC7281A，读出指令（R/W=1）时，数据由BC7281A传向MCU。具体各寄存器数据的含义请参阅上文有关各寄存器的叙述。

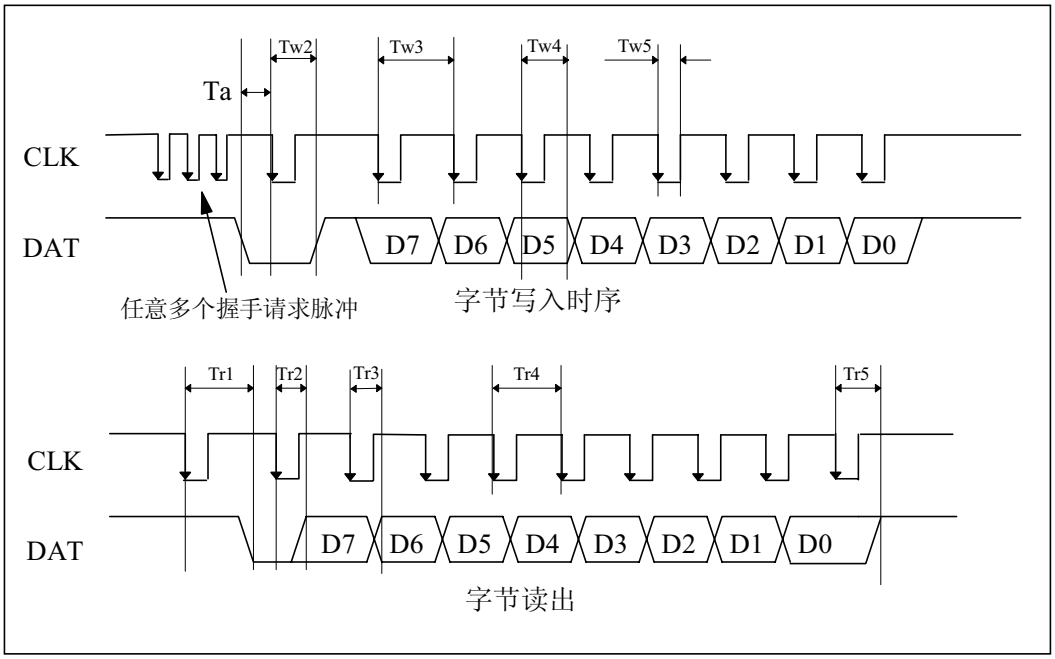
二、时序

1、字节写入BC7281A——指令字节及数据字节

字节写入BC7281A，包括指令字节和写入BC7281A的数据字节，一个写入寄存器的指令，由两个字节写入操作组成，第一个字节为指令字节，而第二个字节则为数据字节。在接口空闲的情况下，BC7281A的DAT引脚处于高阻输入状态，而MCU端也应将DAT线置于输入状态，上拉电阻使得DAT线上为高电平。传送开始时，首先需要建立握手信号，MCU先向BC7281A发出一系列CLK脉冲，脉冲的数量可以是任意多个，并同时监测DAT线，而BC7281A则在收到该握手脉冲后在DAT线上输出一低电平，表示准备好可以接收MCU的数据，MCU一旦检测到BC7281A的响应信号后，立刻停止发送握手脉冲，并在15us之内（时序图中Ta)在CLK线上再次发出一个脉冲，该脉冲使得BC7281A的DAT引脚恢复高阻输入状态，因为DAT线上有上拉电阻，因此DAT线上恢复成高电平，MCU在检测到DAT线恢复成高电平后，即开始发送数据。发送时数据的高位（MSB）在前。每发送一位，即输出一CLK脉冲，开始部分及数据传送部分的CLK脉冲均为下降沿有效。需要注意的是，MCU检测到BC7281A的低电平握手信号后，应该在15us之内给出下一个CLK脉冲，同时请注意数据传送时的数据保持时间，如果数据保持时间小于规定的值，则有可能造成通讯错误。

2、字节从BC7281A读出——读出数据字节

读寄存器操作，由一个字节写入操作和一个字节读出操作两部分组成，字节写入操作写入指令字，数据则由字节读出操作读出。MCU在传送完指令字后，应将DAT线置于输入状态，以便从BC728X接收数据。读出数据时，也需要建立握手信号，过程与写入数据时相似，但有不同，数据读出时，MCU仅发送一个单一的握手脉冲，而不是像写入数据时不停的发送直到收到BC7281A响应信号。具体过程是：MCU首先向BC7281A发出一个起始CLK脉冲，BC7281A在收到该脉冲后在DAT上响应一低电平，表示准备好输出数据，此后MCU再发出一CLK脉冲，BC7281A的DAT脚开始输出数据。此后BC7281A每收到一个脉冲，即在DAT上输出一个数据位。一个与写入指令不同的地方是，当8个数据位均读出了以后，MCU还必须再多发出一个CLK脉冲，表示数据接收完毕，BC7281A才能从数据输出状态转成输入状态，准备接收下一个指令。



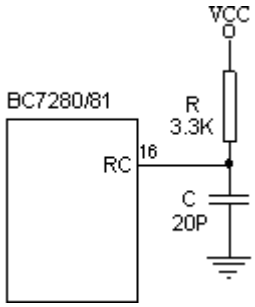
符号	说明	最小值	典型值	最大值	备注
		单位：uS			
Ta	写入数据握手时MCU响应时间			15	
Tw1	写入数据时握手信号响应时间		15	10ms	
Tw2	写入数据时握手信号完成到接收状态准备好建立时间	5	6	10	
Tw3	写入移位脉冲周期	20			
Tw4	写入数据数据保持时间	8	11		
Tw5	写入数据CLK脉冲宽度	1			
Tr1	读出数据时握手信号响应时间		15	55	
Tr2	读出数据时握手信号完成到输出数据准备好建立时间	7	9	12	
Tr3	输出数据建立时间		10	14	
Tr4	读出CLK脉冲周期	22			
Tr5	DAT线恢复输入状态响应时间	10	15	20	

注意：在数据传送期间，BC7281A将不会进行显示和键盘扫描扫描，因此虽然数据传送的速度并没有下限的限制，但如果速度过慢（指令的传送时间>扫描周期），将会对显示造成影响。

外部电路

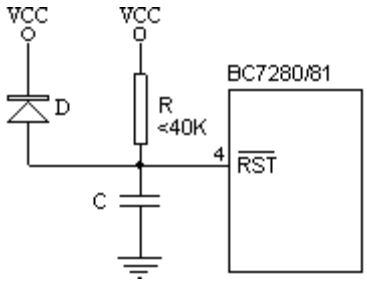
1、振荡电路

BC7281A采用外接的RC振荡电路为显示和键盘扫描提供时钟驱动，外接元件的典型参数为R=3.3K, C=20pF，见下图。在Vcc=5V的情况下，振荡电路的典型振荡频率约为4.5MHz。BC7281A的CLKO端为内部振荡电路的输出端，一般此脚悬空即可。在电路板布线时，振荡电路的元件应尽可能地靠近BC7281A芯片，并尽量使连线最短。



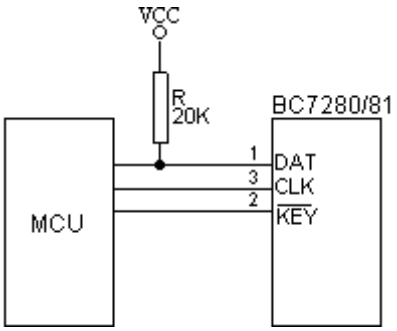
2、复位电路

芯片的$\overline{\text{RST}}$引脚为复位端。因为BC7281A的内部有上电复位电路，因此在一般情况下不需要特殊的复位电路，只需将$\overline{\text{RST}}$引脚直接连接到VCC端就可以了。如果您需要外部的复位电路，可以按照如下的接法，或自行设计的其他电路。$\overline{\text{RST}}$上的复位脉冲的最小宽度为20μS，复位电路中电阻R的阻值一般不要超过40KΩ。另外的一种方法是直接由MCU控制BC7281A的复位。BC7281A的复位过程大约需要25mS的时间，也即$\overline{\text{RST}}$为高电平约25mS后，BC7281A才开始工作。



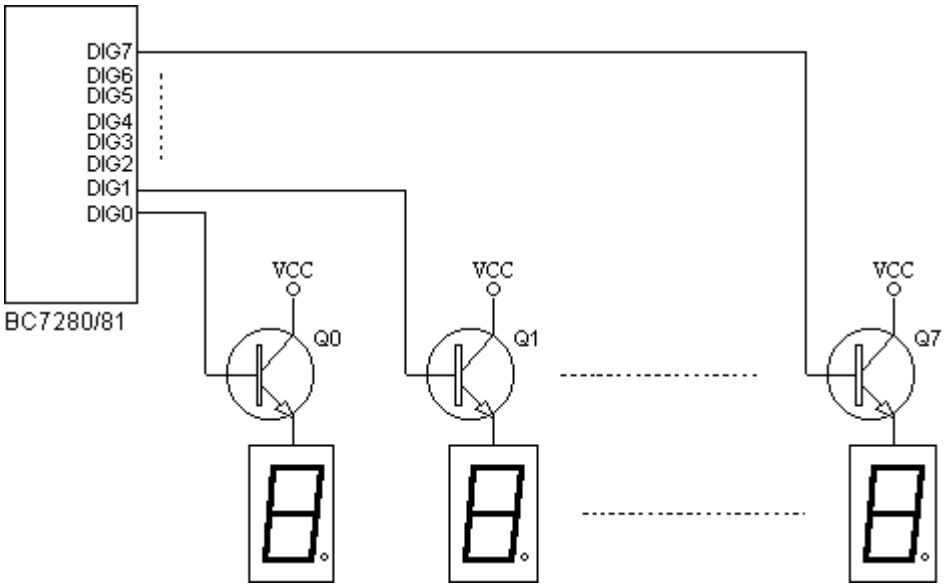
3、与MCU接口电路

BC7281A与MCU的接口共需要三根线，数据线DAT，时钟线CLK和按键指示$\overline{\text{KEY}}$，其中CLK和$\overline{\text{KEY}}$引脚分别为输入和输出引脚，而DAT脚则为双向口，其内部为OPEN DRAIN结构，需要外接一20KΩ左右的上拉电阻，以使其能可靠地输出高电平。

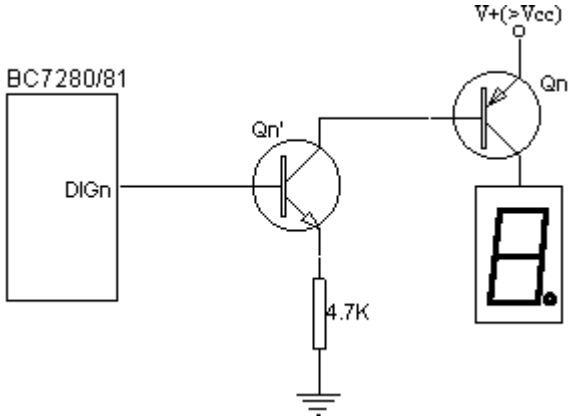


4、位驱动电路

DIG0-DIG7为位驱动输出，BC7281A适合连接共阳式的数码管，虽然DIG0-DIG7本身具有一定的驱动能力，但为了保证足够的显示亮度，应该另加以外部驱动电路。外部驱动电路比较简单，只需要8只NPN型三极管即可，三极管接成射极跟随器形式，因此基极无需限流电阻。



对于大型的LED数码管，其内部各显示段往往由若干个LED串联而成，这样如按照标准的接法，则无法提供足够的驱动电压，可按照如下电路连接：



三极管的选取，并无特殊的要求，只需注意两点：

1、Qn三极管最大电流Ice>8*Iseg（如果是16位的显示系统，则要求Ice>16*Iseg），Iseg为每个显示段的峰值电流。

2、如果V+比较低，则应适当减小Qn'发射极的电阻，以使电路满足

βn*（V_+-0.7)/R > 8*Iseg

的要求（如果是16位的显示系统，则要求 >16*Iseg），其中βn为Qn的放大倍数，R为Qn'的发射极电阻，Iseg的单位为mA。

5、段驱动电路

BC7281A需要外接移位寄存器构成段驱动电路。

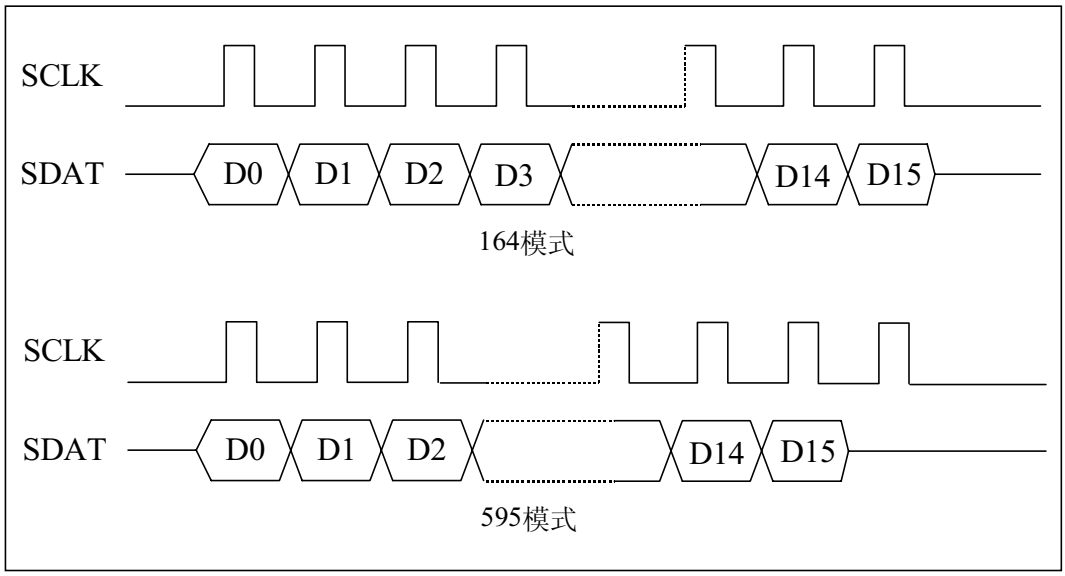
BC7281A送出的数据共16位，数据结构如下表，发送的顺序是高位（MSB）在前：

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
DP_2	G_2	F_2	E_2	D_2	C_2	B_2	A_2	DP_1	G_1	F_1	E_1	D_1	C_1	B_1	A_1

其中，前8位（D15-D8）为第8-15位显示的段驱动数据，后8位（D7-D0）为第0-7位显示的段驱动数据。

对于8位(64段)以内的显示系统，只需要接入对应DIG0-DIG7的一片8位的移位寄存器，而对于多于8位的显示系统，则需要2片8位移位寄存器或一片16位的移位寄存器。由于BC7281A的段驱动数据输出极性及移位脉冲的时序均可调，因此可以适应与各种各样的移位寄存器相连。移位脉冲有两种模式，分别是164模式和595模式（见上文中工作模式控制寄存器部分），164模式为普通的移位寄存器模式，每一位数据对应着一个移位脉冲；而595模式适应于象74xx595等内部有二级缓存触发器的移位寄存器，使用时将二级触发器的锁存时钟与一级锁存（移位寄存器）的时钟并联在SCLK上，这样二

级锁存器与移位寄存器同步更新但其内容比移位寄存器滞后一个脉冲，在595模式下，SCLK在最后一个脉冲后，还会输出一额外的脉冲，这样数据便可以正确地锁存到二级缓存中。移位寄存器串行数据输出的时序图如下：

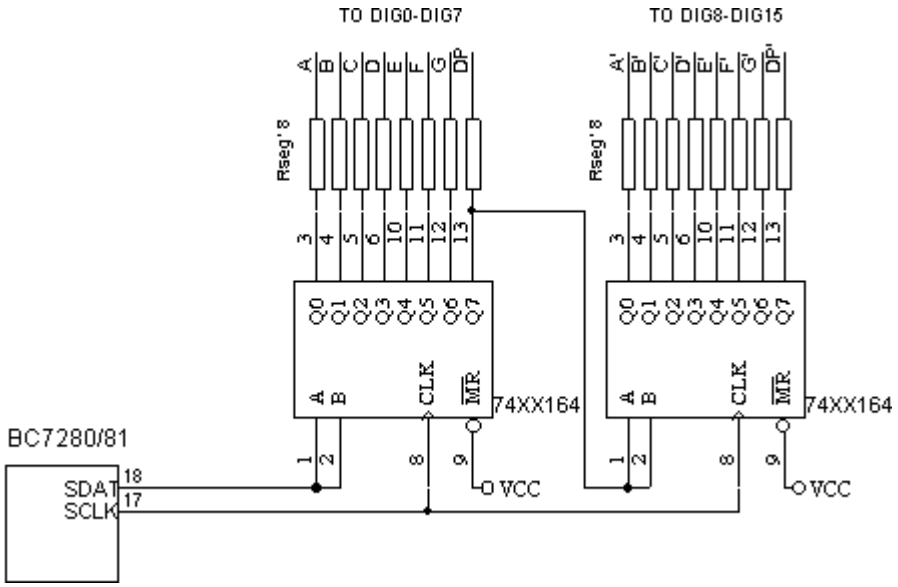


BC7281A的段驱动数据的输出极性可以由软件控制，这样使用者可以灵活地使用各种外围驱动电路。BC728X缺省的工作模式是164模式、不反相输出，也就是说，输出的段驱动数据中，点亮的段对应的数据为‘0’。如果电路中在移位寄存器的后面又使用了反相输出的驱动器，或者使用的是反相输出的移位寄存器，则需要将BC7281A的工作模式置为反相模式（INV=1）。

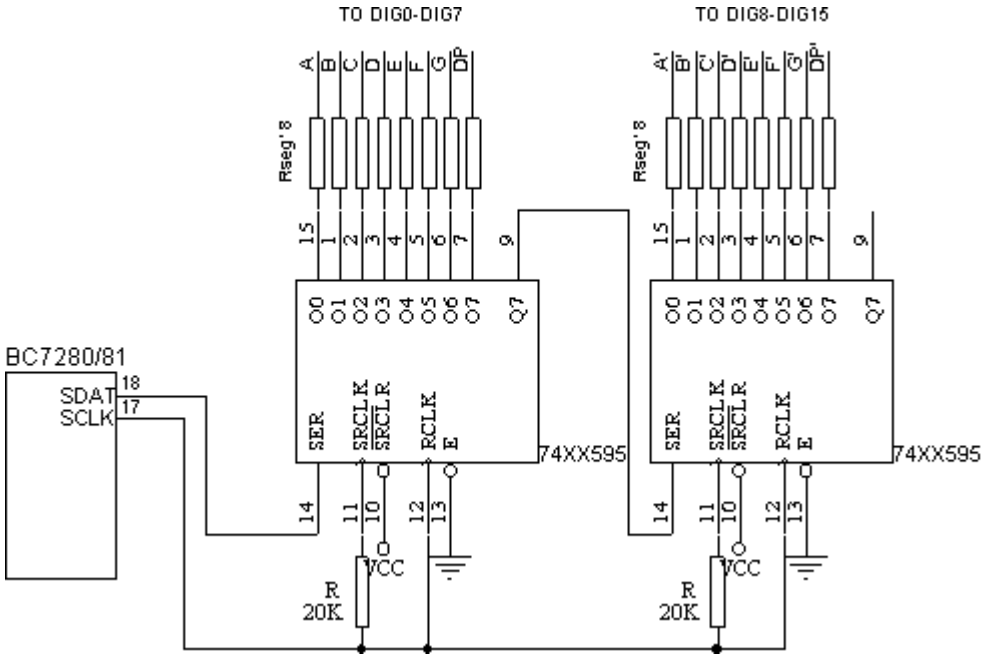
需要指出的是，INV位所影响的只是段驱动数据的极性，而对键盘扫描的极性没有影响，因此如果使用了反相输出的驱动器，而同时又使用键盘的话，键盘矩阵必须连接在反相驱动器之前。

可以和BC7281A配合的电路有很多，下面是其中比较典型的几种：

1、74xx164——MOD=0，INV=0



2、74xx595——MOD=1，INV=0

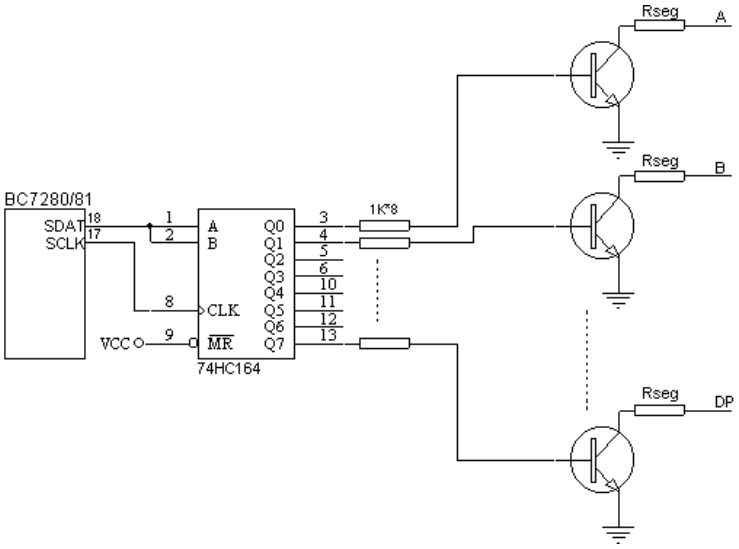


串入SRCLK端的电阻的作用在于与芯片的输入电容及电路的分布电容构成一积分电路，从而对输入SRCLK的时钟产生一微小延时，这样可以保证移位脉冲SRCLK迟于二级锁存脉冲RCLK到达，从而保证移位寄存器的内容在变化之前可靠地进入二级锁存器。

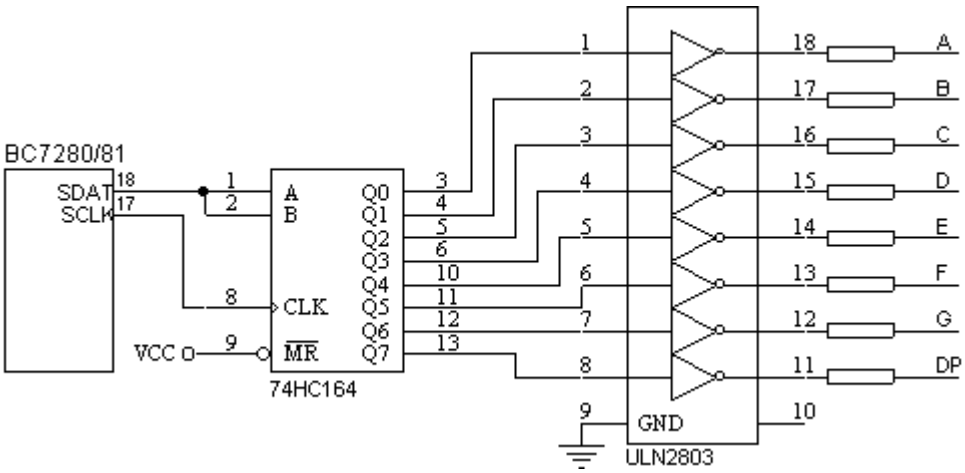
电阻的阻值没有严格要求，并且不同厂家及不同型号的芯片对应的阻值范围也会有所不同，一般而言，这个范围是比较宽的，从几K到几十K都可以，20K的阻值可以适用于绝大多数芯片。

典型的逻辑电路的驱动能力有限，只能驱动较小型的数码管，如果需要驱动较大的数码管（≥1英寸），则需要另加驱动或选用专用的驱动芯片，下面是几种常见方案：

1、锁存器+三极管——INV=1



2、锁存器+达林顿驱动阵列——INV=1



另外，还可以选用专门的驱动芯片，下面这些电路均可以直接和BC7281A配合使用：

TPIC6B595——TI公司，8位，段电流150mA，MOD=1，INV=1

HEF4794B——Philips公司，8位，段电流40mA，MOD=0，INV=0

M66312——Mitsubishi公司，8位，段电流16mA，MOD=1，INV=0

A6275——Allegro公司，8位，段电流90mA，MOD=0，INV=0

UCN5821——Allegro公司，8位，段电流350mA，MOD=0，INV=1

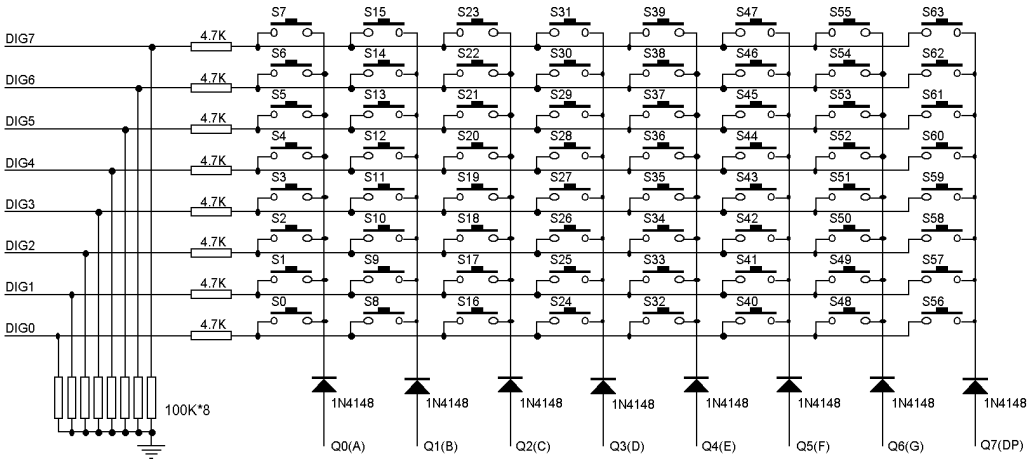
除了采用恒流源电路的驱动芯片（如A6275等）外，段驱动电路中应加入限流电阻。对LED显示器来说，段电流越大，则显示亮度越高，因此在可能的情况下，应根据所选用的数码管和驱动电路的特性，尽可能选取较大的段电流，确定了段电流后，再根据以下公式计算限流电阻的阻值：

$$R_{seg}=(V_{cc}-0.7-V_{led}-V_{ol})/I_{seg}$$

其中,Vcc为BC7281A的电源电压，V_{led}为LED的管压降，V_{ol}为段驱动电路的低电平输出电压，I_{seg}为段电流。

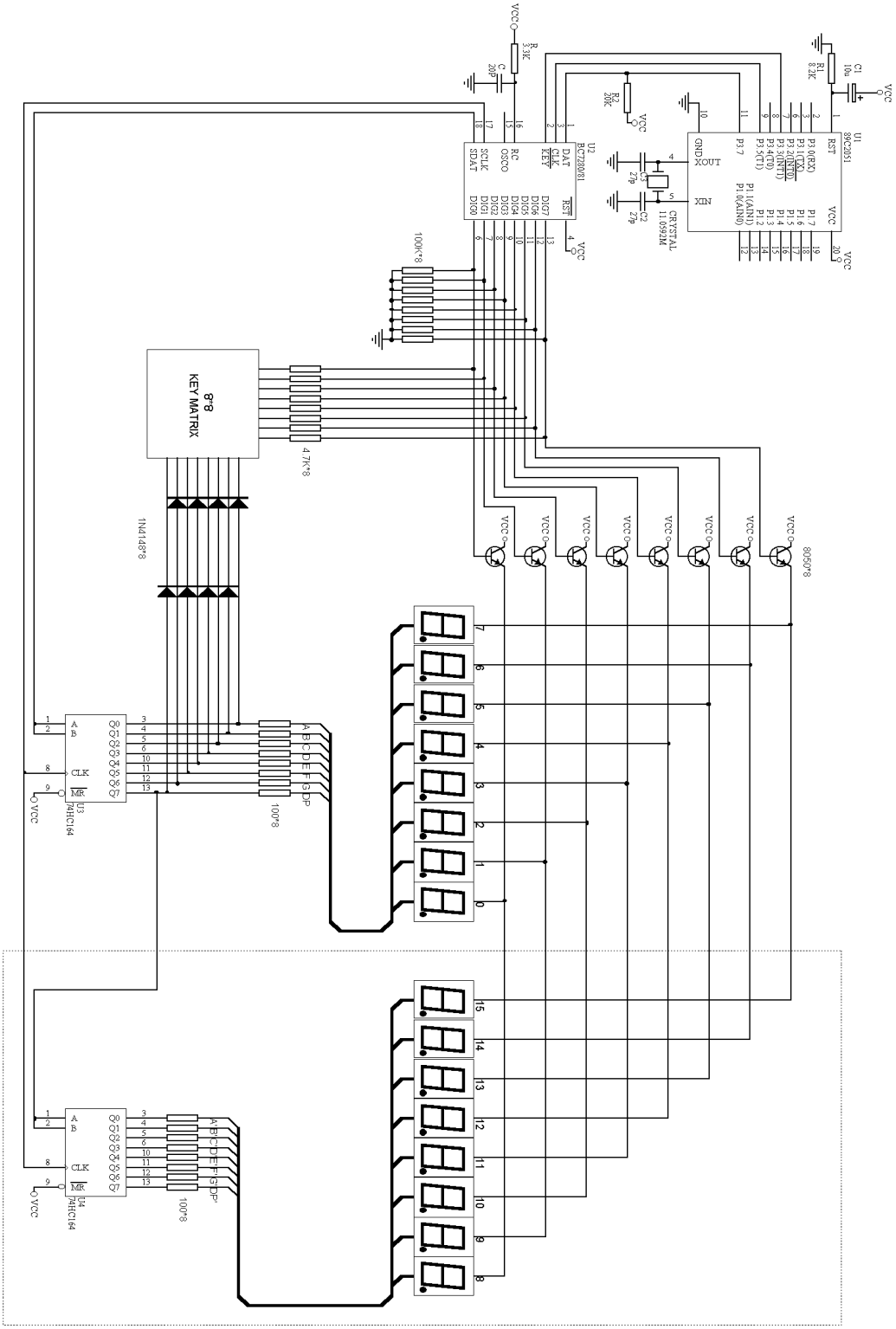
6、键盘矩阵

BC7281A最多可以连接64个按键，按8*8矩阵排列，矩阵的‘行’连接到BC7281A的位驱动DIG0-DIG7，矩阵的‘列’连接到第0-7位显示的段驱动移位寄存器的输出，为了防止对显示部分的影响，键盘矩阵与显示电路之间必须加入二极管和4.7K的隔离电阻。当使用BC7281A的键盘功能时，DIG0-DIG7上应加以100K的下拉电阻，且8根引脚必须都接入下拉电阻，即使所用到的键比较少时，也不能将其中未连接键盘的引脚上的下拉电阻省略。



典型应用

下面给出BC7281A的一个典型应用电路，显示位数为16位，连接的键盘个数为64个，控制单片机采用ATMEL公司的AT89C2051，为简单起见，图中仅画出与BC7281A相关的电路。如果使用的是8位显示，只需要将虚线框内的电路去掉即可。电路中BC7281A的振荡电路的元件参数为：R=3.3K, C=20pF；AT89C2051单片机的工作频率为11.0592MHz。稍后部分给出了一个简单的应用程序，程序的功能为读入按键码，然后显示出来。因为BC7281A在KMS=0模式时具有按键保持功能，在键值被读走之前，$\overline{\text{KEY}}$将一直维持低电平，因此对按键使用查询方式即可，不过为了今后升级方便，$\overline{\text{KEY}}$仍然连接到了单片机的$\overline{\text{INT1}}$引脚上，以便于日后改为中断处理方式。程序为通用程序，不光适用于BC7281A，也适用于较早的BC7280/7281，程序分为 C 语言和汇编语言两个版本。



1、测试程序，C语言版本

```
#include <reg51.h>

//*** 函数定义 ***
void delay(unsigned char);                        // 短暂延时
void write728x(unsigned char, unsigned char);  // 写入到BC728x
unsigned char read728x(unsigned char);          // 从BC728x读出
void send_byte(unsigned char);                    // 发送一个字节
unsigned char receive_byte(void);                // 接收一个字节

//*** 变量及I/O口定义 ***
unsigned char key_number;
unsigned int tmr;
sbit clk=P3^5;              // clk 连接于 P3.5
sbit dat=P3^7;              // dat 连接于 P3.7
sbit key=P3^3;              // key 连接于 P3.3(INT1)

//*** 主程序 ***
main()
{
       for (tmr=0;tmr<0xffff;tmr++);     // 等待 BC728x 完成复位
       write728x(0x12,0x80);      // 初始化BC728x为164模式，不反相
       while (1)
       {
              while(key);                 // 等待按键
              key_number=read728x(0x13); // 读键值
              write728x(0x15,0x10+(key_number&0xf0)/16);
                     // 在第1位上以HEX译码方式显示键码的高4位
              write728x(0x15,key_number&0x0f);
                     // 在第0位上以HEX译码方式显示键码的低4位
       }
}

// ***********************************************************
// * 写入BC728X, 第一个参数为目标寄存器地址, 第二个参数为要写入的数据 *
// ***********************************************************
void write728x(unsigned char reg_add, unsigned char write_data)
```



```
{
       send_byte(reg_add);         // 发送寄存器地址
       send_byte(write_data);      // 发送数据字节
}

// ********************************************************
// *       读出 BC728X 内部寄存器的值, 调用参数为寄存器地址         *
// ********************************************************
unsigned char read728x(unsigned char reg_add)
{
       send_byte(reg_add|0x80);    // 发送读指令(bit7=1)
       return(receive_byte());     // 接收数据字节并返回
}

// ********************************************************
// *                 向 BC728X 发送一个字节                    *
// ********************************************************
void send_byte(unsigned char send_byte)
{
       unsigned char bit_counter;
       clk=0;              // 产生一 clk 脉冲
       clk=1;
       do {
              clk=0;       // 发送 clk 脉冲直至dat为低电平
              clk=1;
       } while (dat);
       clk=0;              // 15us之内再次输出一 clk 脉冲
       clk=1;
       while (!dat);       // 等待 BC728X 进入接收状态
       for (bit_counter=0;bit_counter<8;bit_counter++)
       {                   // 发送 8 个比特
              if ((send_byte&0x80)==0)
              {
                     dat=0;// 如果待发bit为0, 置 dat 为 0
              }
              else
              {
                     dat=1;       // 反之置为 1
```



```
            }
            send_byte=send_byte*2;     // send_byte 左移一位
            clk=0;              // 输出一 clk 脉冲
            clk=1;
            delay(1);           // 短暂延时
      }
      dat=1;                    // 恢复 dat 为高电平
      delay(2);                 // 指令间设置一微小延时
}

// ************************************************************
// *                  从 BC728X 接收一个字节                   *
// ************************************************************
unsigned char receive_byte(void)
{
      unsigned char bit_counter, in_byte;
      clk=0;              // 只发送一个单一的 clk 脉冲
      clk=1;
      while (dat);        // 等待 BC728X 响应dat低电平
      clk=0;       // 收到响应, 再发一脉冲等待接收数据
      clk=1;
      for (bit_counter=0;bit_counter<8;bit_counter++)
      {                   // 接收8个bit
            delay(1);     // 短暂延时
            in_byte=in_byte*2;  // in_byte 左移一位
            if (dat)      // 如果 dat 为'1'
            {
                  in_byte=in_byte|0x01;      // bit0=1
            }
            clk=0;        // 输出一 clk 脉冲
            clk=1;
      }
      delay(2);           // 指令间设置一微小延时
      return(in_byte);
}

// ************************************************************
// *  短暂延时程序, 延时时间与参数time成正比, 范围是几个uS到几百个uS *
```



```
// ************************************************************
void delay(unsigned char time)
{
       while (time!=0)
       {
              time--;
       }
}
```

2、汇编语言版本

```
       $ title (BC728X Test Program, AT89C2051 @ 11.0592MHz)
       $ DB

BIT_COUNT     DATA   07FH
TIMER         DATA   07EH
TIMER1        DATA   07DH
TEMP          DATA   07CH
DATA_IN       DATA   021H
DATA_OUT      DATA   020H

CLK           BIT    P3.5                ;定义I/O口
DAT           BIT    P3.7                ;
KEY           BIT    P3.3                ;

              ORG     000H
              JMP     START

;************************************************************
;* 上电初始化
;************************************************************
              ORG     100H
START:        MOV     SP,#2FH             ;设置堆栈
              MOV     TIMER,#50
START_DELAY:  MOV     TIMER1,#255         ;延时以确保BC728X完成复位
START_DELAY1: DJNZ    TIMER1,START_DELAY1
              DJNZ    TIMER,START_DELAY
```



```
                MOV     DATA_OUT,#12H       ;BC728X初始化
                CALL    SEND
                MOV     DATA_OUT,#80H       ;设定为164模式, 不反相
                CALL    SEND
;*************************************************************
;* 主程序
;*************************************************************
MAIN:           JB      KEY,MAIN            ;等待按键
                MOV     DATA_OUT,#93H       ;读键值锁存器指令(地址13H)
                CALL    SEND
                CALL    RECEIVE             ;读出数据
                MOV     DATA_OUT,#15H       ;HEX译码指令
                CALL    SEND
                MOV     TEMP,DATA_IN
                ANL     TEMP,#0F0H          ;键码高4位在第1位显示
                MOV     A,TEMP
                SWAP    A
                ORL     A,#10H
                MOV     DATA_OUT,A
                CALL    SEND
                MOV     DATA_OUT,#15H       ;HEX译码指令
                CALL    SEND
                MOV     A,DATA_IN
                ANL     A,#0FH              ;取键码的低4位
                MOV     DATA_OUT,A          ;在第0位显示
                CALL    SEND
                JMP     MAIN

;*************************************************************
;* 向BC728X发送一个字节子程序,待发送数据存于DATA_OUT
;*************************************************************
SEND:           CLR     CLK                 ;输出CLK脉冲
                SETB    CLK
WAIT1:          JB      DAT,SEND            ;检测DAT如为高继续输出脉冲
                CLR     CLK                 ;15us之内再输出一CLK脉冲
                SETB    CLK
WAIT2:          JNB     DAT,WAIT2     ;等待DAT恢复高电平(输入状态)
                MOV     BIT_COUNT,#8
```



```
SEND_LOOP:    MOV    C,DATA_OUT.7          ;输出BIT7
              MOV    DAT,C
              CLR    CLK                   ;输出一CLK脉冲
              SETB   CLK
              MOV    A,DATA_OUT
              RL     A
              MOV    DATA_OUT,A            ;DATA_OUT左移一位
              NOP                          ;短暂延时
              NOP
              NOP
              DJNZ   BIT_COUNT,SEND_LOOP
              SETB   DAT                   ;恢复DAT为高电平
              NOP                          ;指令间设置一微小延时
              NOP
              NOP
              NOP
              RET

;*************************************************************
;* 从BC728X接收一个字节子程序,接收到的数据存于DATA_IN
;*************************************************************
RECEIVE:      CLR    CLK                   ;发出一单一CLK脉冲
              SETB   CLK
WAIT3:        JB     DAT,WAIT3             ;等待DAT低电平响应信号
              CLR    CLK           ;再发出一CLK脉冲,准备接收数据
              SETB   CLK
              MOV    BIT_COUNT,#8
RECV_LOOP:    NOP                          ;短暂延时
              NOP
              NOP
              NOP
              NOP
              NOP
              NOP
              NOP
              NOP
              MOV    A,DATA_IN
              MOV    C,DAT                 ;读入一位
```

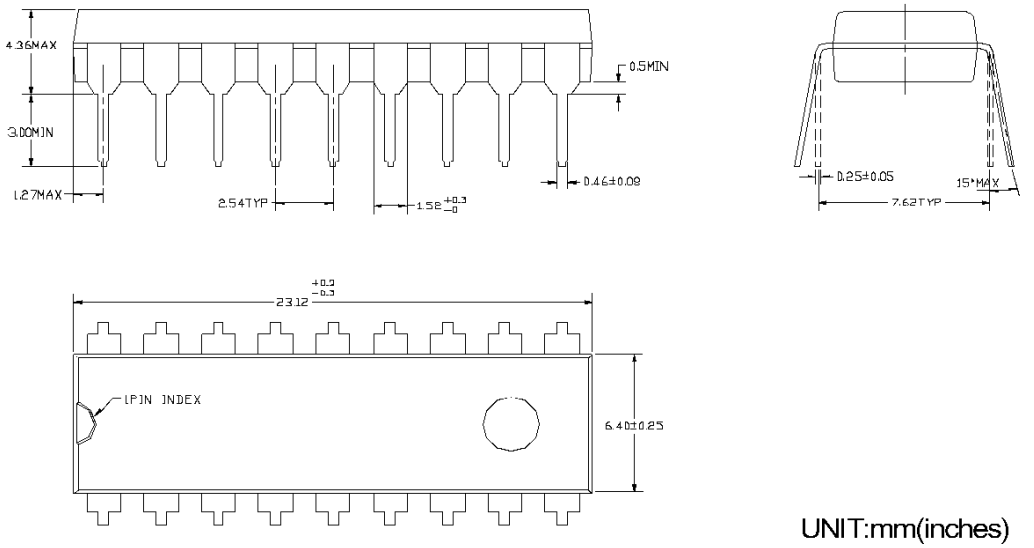


```
RLC    A
MOV    DATA_IN,A
CLR    CLK                    ;发出CLK脉冲
SETB   CLK
DJNZ   BIT_COUNT,RECV_LOOP
NOP                           ;指令间增加一微小延时
NOP
NOP
NOP
RET

END
```


附录1：封装尺寸

Plastic DIP-18pin



UNIT:mm(inches)

附录2

BC728X应用笔记（一）

噪声敏感电路的电路板布线

作者：凌志比高科技有限公司 矫健

BC728X是LED显示驱动电路，而LED显示器是属于高功耗的显示器件，特别是BC7281最多可以同时驱动16位显示，如果以一个显示段的驱动电流为25MA计算，则显示电流最大可以达到25*8*2=400MA，而且因为BC728X是分时扫描的显示方式，所以显示电流并不是一个持续稳定的量，而是一个快速瞬变的量，这个电流的变化噪声通过电源耦合到系统的其它部分电路，很可能会对电路其它部分的正常工作造成影响，因此象此种LED扫描显示电路如果设计者在设计时不对这些干扰加以考虑，则很有可能会给系统中其他部分，尤其是模拟电路部分带来难以去除的干扰，甚至影响到数字电路的正常工作。曾经有过因为电源布线不合理，MCU距离显示驱动电路过近，从而造成MCU程序无规律地跑飞和复位的例子。

下面是为避免噪声干扰而在电路板设计时应注意的几个方面：

1、尽可能将显示部分的电源和系统的其它部分的电源分开独立布线，两部分电路只在电源处才汇合成一点，这样可以非常显著地减少互相间的干扰，当然，如果对两部分电路分别供电，是最理想的情况，某些敏感的模拟电路系统，可能会需要用到这种方法。

2、尽可能使用较粗的地线和电源走线，从而减小电源部分的内阻，进而减小噪音，在功率驱动器件和MCU器件的旁边加上滤除高频干扰的电容，也是比较有效的办法。

3、扫描显示方式的特点是，电路的瞬间电流可能比其平均电流大很多倍，因此，在有可能的情况下，尽可能选用功率容量比较大的电源。

附录3：

BC728X应用笔记（二）

键盘使用技巧

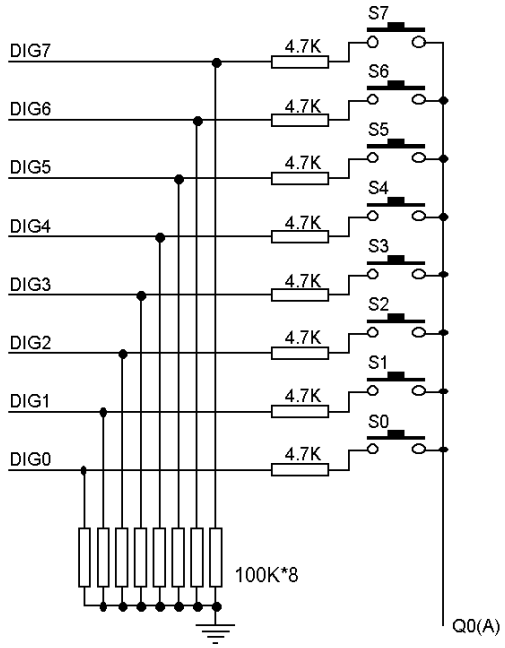
作者：凌志比高科技有限公司 矫健

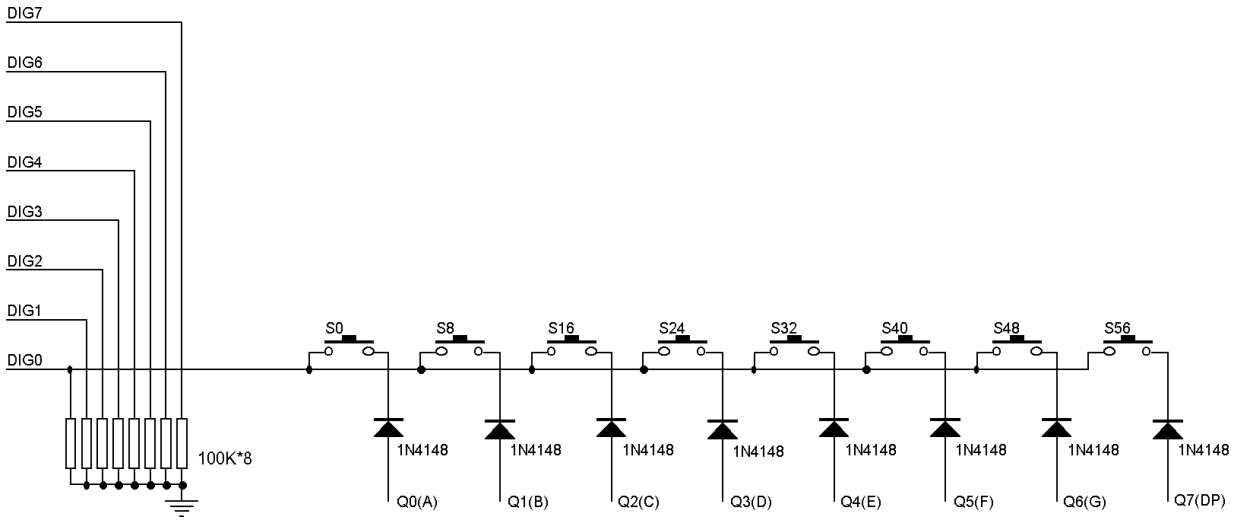
BC7281A具有多达16位的数码管驱动和64键的键盘管理能力，这足以满足一般的单片机系统的要求。事实上，对于大多数的仪器仪表类应用来说，一般的键盘数量不会多于8个，在这种情况下，BC7281A的键盘电路还可以进行简化。

BC7281A型键盘电路见内文，当按键数少于8个时，可以将图中的4.7k电阻或二极管省略。

图中的4.7k电阻和二极管的作用，均在于防止由键盘引起的短路。如果没有电阻和二极管的保护，当同一行或同一列里面有两个键同时按下时，则会造成相应列或相应行之间的短路，对显示电路造成影响。例如，S0和S16键同时按下时，会造成Q0(A)和Q2(C)之间的短路，而S9和S10同时按下时，则会造成DIG1和DIG2之间的短路。

而当按键数少于8个时，我们可以将所有按键集中到一行或一列中，这样也就避免了相应的列之间或行之间的短路，从而可以将保护电阻或二极管省略，见下面二图：





二图分别标明了按键集中于同一列和同一行的情况，推荐采用集中于同一列略去二极管的方案，一方面因为二级管比电阻的成本高，另一方面也因为没有了二级管的管压降，键盘的可靠性会更高。

附录4：

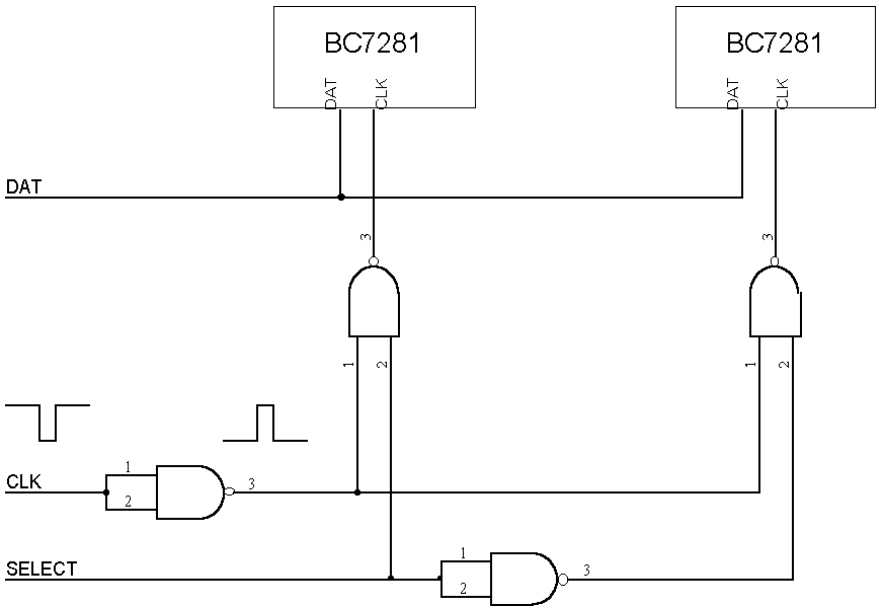
BC728X应用笔记（三）

BC7281的多片联用

作者：凌志比高科技有限公司 矫健

单片BC7281具有多达16位的数码管驱动能力，但是，在某些需要更多位数的LED显示的应用中，因为BC7281并没有“片选”信号的输入端，有些用户会对如何在一个系统中同时使用多片BC7281感到困惑，其实因为BC7281的通讯开始时是由CLK引脚启动握手信号，而DAT引脚在建立握手信号之前一直是高阻状态，所以只要将单片机送出的CLK信号按需要分配给指定的BC7281芯片就行了。

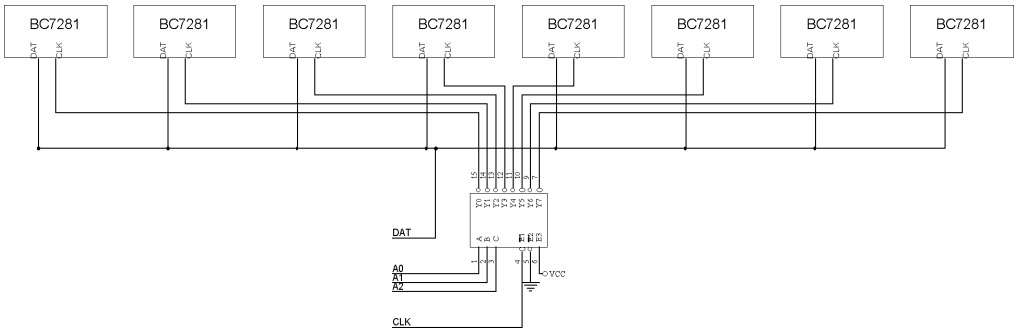
如果系统中有两片BC7281，用几个与非门，即可达到目的，见下图：



图中的SELECT线起到选择显示芯片的作用，当SELECT=1时，选择左边的芯片，SELECT=0时，选择右边芯片。

而当系统中需要用到多片BC7281时，则可以参考以下的电路，该电路中使用了地

址译码器74HC138，从而可以将系统中的BC7281数量扩大到8片。



图中A0、A1、A2为地址线，其状态决定当前选中的芯片，使用时应当先设置地址，然后再开始握手信号。

不论采用本文中的哪种方法，均应注意在通讯的过程中不能做芯片选择的转换，否则会造成通讯错误，选择芯片的转换应该在上一个完整的指令（无论写入或读出）传送结束、BC7281的DAT引脚进入高阻状态之后。图中没有标出DAT线上的上拉电阻，该电阻的阻值与单片应用时一致，只需要接一只，而无需每只芯片都接。

