

BC7281B

128 段 LED 显示及 64 键键盘控制芯片

特点

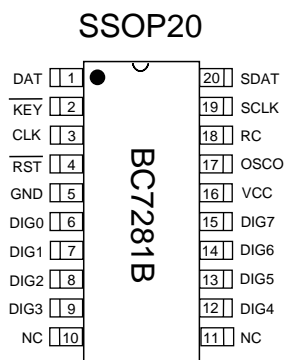
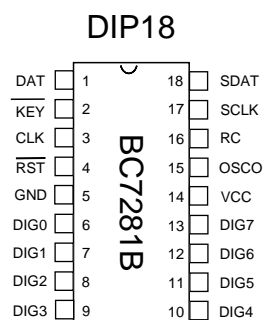
- 可驱动 8 位或 16 位数码管显示或 64/128 只独立 LED
- 具有 64 键键盘接口，内建去抖功能
- 具有 2 种键盘工作模式，适应不同应用需求
- 显示亮度控制
- 具有寄存器保护(抗干扰)模式，可靠性更高
- 16 位均可独立控制闪烁属性，闪烁速度软件可调
- 独具光柱译码方式，可独立控制两条 64 段光柱显示
- 段寻址功能便于控制独立 LED
- 适用于任何型号的数码管/发光二极管
- 键盘部分具有键值锁存功能
- 内部显示寄存器和控制寄存器的内容均可读出
- 2 线高速串行接口
- 兼容 BC7281/BC7281A

简述

BC7281 系列是 8 位/16 位 LED 数码管显示及键盘接口专用控制芯片，BC7281B 是 BC7281A 芯片的升级换代产品，全面兼容 BC7281A，同时提供更强的性能。通过外接移位寄存器(典型芯片如 74HC164, 74LS595 等)，BC7281 最多可以控制 16 位数码管显示或 128 只独立的 LED。BC7281 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。

BC7281 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；BC7281A/B 对 16 个显示位均可以独立地控制闪烁属性；同时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；较之 BC7281A，BC7281B 增加了显示亮度的控制，可以在正常、半亮和关闭之间切换。BC7281 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。

引脚图



BC7281 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/B 的键盘具有 2 种工作模式，详细情况请参见内文。

BC7281B 内部共有 31 个寄存器，包括 16 个显示寄存器和 15 个特殊(控制)寄存器，比 BC7281A 增加了 5 个，所有的操作均通过对这 32 个寄存器的访问完成。

BC7281 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。BC7281B 和较早的 BC7281/BC7281A 在软件和硬件上均兼容。内部设计的改进使得 BC7281B 比 BC7281/BC7281A 具有了更强的抗干扰性能，BC728X 的用户代换为 BC7281B 时，软件和硬件不需做任何改动，即可获得更强的抗干扰能力。同时，BC7281B 还增加了“寄存器保护”（抗干扰）模式，使得抗干扰能力进一步提高。

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, $T_A=25$, $V_{CC}=5.0V$)

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	1.0	1.2	1.4	mA	R=3.3K, C=20pF, 所有引脚悬空
	1.5	1.7	1.9	mA	R=1.5K, 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	12	17	20	ms	R=3.3K, C=20pF
	6.5	7.2	8	ms	R=1.5K, C=20pF
按键响应时间	12	20	40	ms	R=3.3K, C=20pF
	6.5	10	16	ms	R=1.5K, 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

BC7281B 芯片内部具有 31 个寄存器，包括 16 个显示寄存器和 15 个特殊功能寄存器，共用一段连续的地址，其地址范围是 00H-1FH，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	
1AH	(保留)	-	-	BC7281B 独有
1BH	10H 的镜像寄存器		只写	BC7281B 独有
1CH	11H 的镜像寄存器		只写	BC7281B 独有
1DH	12H 的镜像寄存器		只写	BC7281B 独有
1EH	19H 的镜像寄存器		只写	BC7281B 独有
1FH	通用镜像寄存器		只写	BC7281B 独有

显示寄存器——地址 00H-0FH，镜像寄存器地址 1FH

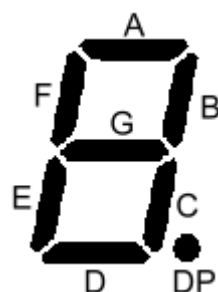
显示寄存器共 16 个，分别对应 16 个显示位，其内容为各显示位的映射，各数据位与显示段对应关系如下

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’。

当 BC7281B 工作于“寄存器保护”模式时，改写显示寄存器之前，必须先向通用镜像寄存器内写入镜像的数据，以达到防止数据被错误改写的目的。详见后文有关寄存器保护模式的说明。



闪烁开关控制寄存器——地址 10H, 19H, 镜像寄存器地址 1BH, 1EH

闪烁开关控制寄存器控制显示位的闪烁属性。BC7281A/B 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/B 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。

当 BC7281B 工作于“寄存器保护”模式时，闪烁开关控制寄存器的改写必须通过先向其镜像寄存器内写入镜像数据的方式来进行。详见后文有关寄存器保护模式的说明。

闪烁速度控制寄存器——地址 11H，镜像寄存器地址 1CH

BC7281 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。

闪烁速度控制寄存器复位后的初始值为 40H，在该值下，当 BC7281B 的外接 RC 的值分别为 3.3K 和 20pF 时，闪烁的频率大约为 2Hz。

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。当 BC7281B 工作于“寄存器保护”模式时，闪烁速度控制寄存器的改写必须通过先向其镜像寄存器内写入镜像数据的方式来进行。详见后文有关寄存器保护模式的说明。

工作模式控制寄存器——地址 12H，镜像寄存器地址 1DH

为了能够和各种驱动电路配合，BC7281B 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	RP	KO	ES	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，工作模式与原 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，工作模式与原 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。

ES：节能模式。当该位置 1 时，有效驱动电流减小为正常状态的一半（显示亮度随之降低）。

K0: 显示关闭模式。当该位置 1 时，显示扫描被关闭，但是键盘部分仍保持工作。（显示寄存器内容不被清除，并可被更新）

RP: 寄存器保护模式。当 RP=1 时，BC7281B 内部的寄存器（包括显示寄存器和控制寄存器，但不包括译码寄存器 14H-18H）不能够直接改写，必须先向其镜像寄存器写入反相（逐位取反）的数据后，才可向该寄存器写入数据，只有当写入寄存器的数据与镜像寄存器中的反相数据相符时，该数据才可被接受，否则会被忽略。这种模式可以防止因通讯过程中受到干扰而造成寄存器被写入错误数据。镜像寄存器的地址如下：

寄存器	镜像寄存器
显示寄存器 00H-0FH	1FH
闪烁开关控制寄存器 10H	1BH
闪烁速度控制寄存器 11H	1CH
工作模式控制寄存器 12H	1DH
扩展闪烁控制寄存器 19H	1EH

需要说明的是，RP 位不影响译码寄存器的写入方式，RP=1 时，译码显示仍然只需直接将数据写入译码寄存器即可。因此寄存器保护模式在多数情况下并不影响显示刷新的速度。

例 1：RP=1 时，需向显示寄存器 05H 写入数据 F9H。则第一步应向 1FH 地址写入数据 06H(F9H 按位取反得 06H)，第二步向 05H 地址内写入数据 F9H。

例 2：RP=1 时，需向闪烁开关控制寄存器 10H 写入数据 FEH。则第一步应向 1BH 地址写入 01H(FEH 按位取反得 01H)，第二步向 10H 地址写入 FEH。

例 3：RP=1 时，用 BCD 译码方式在第 2 位显示“5”。译码寄存器写入不受 RP 位的影响，因此直接向 14H(BCD 译码寄存器)写入 25H。

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。

复位后，工作模式控制寄存器的初始值为 00H，扫描被禁止，不会有显示，键盘也不工作，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 BC7281 检测到有效按键后，其键值将被存储在此寄存器中，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，BC7281 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。

相关如果没有按键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：

	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/B 不接受多个按键同时按下的情况，见下表两种键盘模式下的对比：

	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。译码寄存器的写入不受RP位的影响。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，a₃:a₀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

BC7281 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 000000B(00H, 全熄)-100000B(40H, 全亮)。

BC7281 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

。BC7281 为每一个显示段定义了一个地址，其定义如下表：

段地	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

A₆:A₀ 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 BC7281 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。

镜像寄存器——地址 1BH,1CH,1DH,1EH,1FH

当 BC7281B 工作于寄存器保护模式（RP=1）时，对于寄存器的写入需要分为 2 个步骤，第一步需要先向该寄存器的镜像寄存器内写入待写入数据的“镜像”值，即待写入数据“按位取反”后的值，第二步才是向目标寄存器内写入数据。只有当写入的数据与镜像寄存器内数据按位取反后相等时，数据才会被接受，否则将被忽略。这样，可以防止通讯过程中的错误所造成的寄存器被错误改写。所有的显示寄存器（00H-0FH）共享一个镜像寄存器 1FH，而 4 个控制寄存器则每个各有一个镜像寄存器与其对应。镜像寄存器只能写入，不能读出。镜像寄存器内的值写入后会一直保持，直至下一次有新的数据写入。

串行接口

一、基本格式

BC7281 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，BC7281 既用该线从 MCU 接收数据，也用该线向 MCU 发送数据。BC7281 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上拉电阻，阻值可以在 10K-20K 之间选取。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 BC7281 的内部寄存器内写入数据；当 R/W=1 时，MCU 读出 BC7281 内部寄存器的数据。a₀-a₄ 为目标寄存器的地址，其范围为 00H-1FH

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 BC7281，读出指令（R/W=1）时，数据由 BC7281 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。

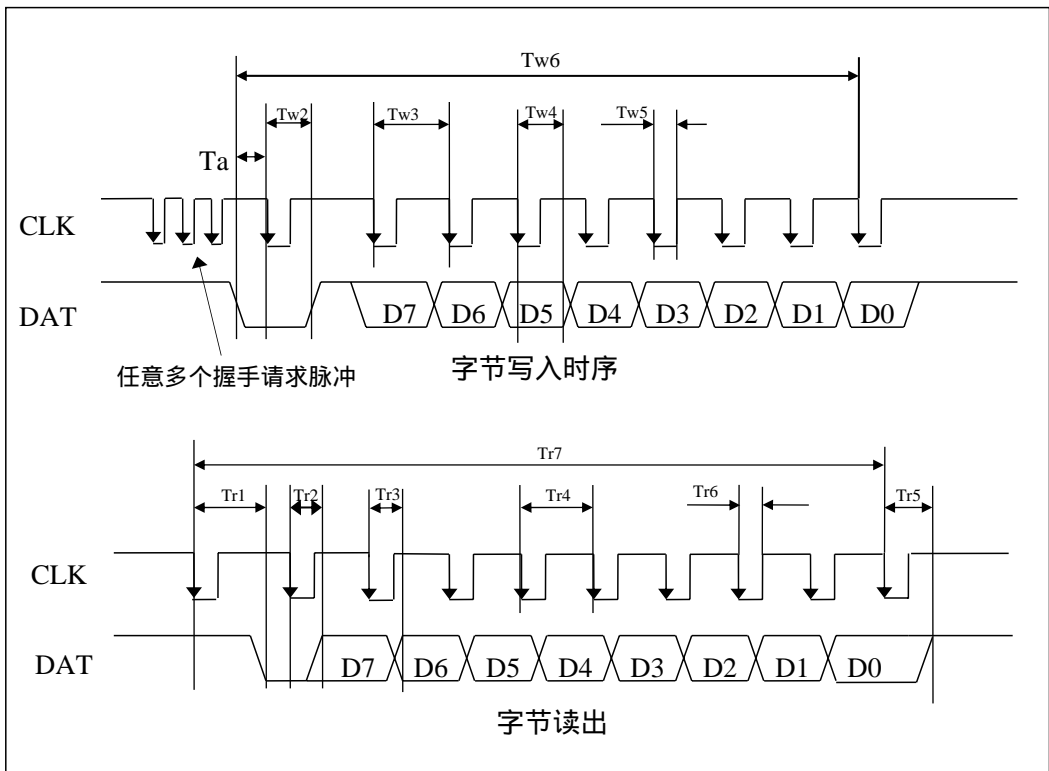
二、时序

1、字节写入 BC7281——指令字节及数据字节

字节写入 BC7281，包括指令字节和写入 BC7281 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，BC7281 的 DAT 引脚处于高阻输入状态，而 MCU 端也将 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 BC7281 发出一系列 CLK 脉冲，脉冲的数量可以是任意多个，MCU 同时监测 DAT 线，而 BC7281 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 BC7281 的响应信号后，立刻停止发送握手脉冲，并在规定时间内（时序图中 Ta，R=3.3K 时为 15uS，R=1.5K 时为 6.8uS）在 CLK 线上再次发出一个脉冲，该脉冲使得 BC7281 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 BC7281 的低电平握手信号后，应该在 Ta 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 BC7281 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 BC7281 响应信号。具体过程是：MCU 首先向 BC7281 发出一个起始 CLK 脉冲，BC7281 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，BC7281 的 DAT 脚开始输出数据。此后 BC7281 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，BC7281 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(R=1.5K/R=3.3K)			
Ta	写入数据握手时 MCU 响应时间			7 / 15	
Tw1	写入数据时握手信号响应时间	0.5 / 1	7 / 15	0.9 / 2ms	
Tw2	写入数据时握手信号完成到 BC7281 接收状态准备好时间	1 / 2	1.5 / 3	2 / 4	
Tw3	写入移位脉冲周期	4 / 8			
Tw4	写入数据数据保持时间	1.8 / 3.5	4 / 8		
Tw5	写入数据 CLK 脉冲宽度	0.5 / 1			
Tw6	写入字节总时间			30ms	仅 BC7281B
Tr1	读出数据时握手信号响应时间	10 / 19	11 / 22	13 / 25	
Tr2	读出数据时握手信号完成到输出数据准备好建立时间	2.1 / 4	2.3 / 4.4	2.6 / 5	
Tr3	输出数据建立时间	2.1 / 4	2.3 / 4.4	2.6 / 5	
Tr4	读出 CLK 脉冲周期	2.6 / 5			
Tr5	DAT 线恢复输入状态响应时间	1.3 / 2.4	1.5 / 3	2 / 4	
Tr6	读出时 CLK 脉冲宽度	0.5 / 1			
Tr7	读出字节总时间			30ms	仅 BC7281

注意：在数据传送期间，BC7281 将不会进行显示和键盘扫描扫描，因此如果通讯速度过慢（指令的传送时间>扫描周期），将会对显示造成影响。

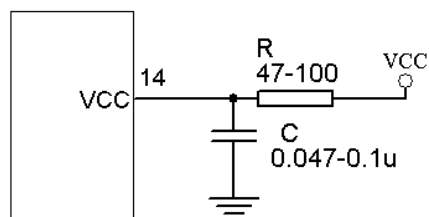
外部电路

1、电源电路

BC7281 的 14 脚为电源引脚，虽然可以直接将其与电路中电源相连，但是为了获得更好的抗干扰能力，建议采用如下的电路，以滤除电源中的尖峰和毛刺：

在 VCC 引脚和电源之间，串入了一个 RC 滤波电路，元件的取值范围如下：R=47-100

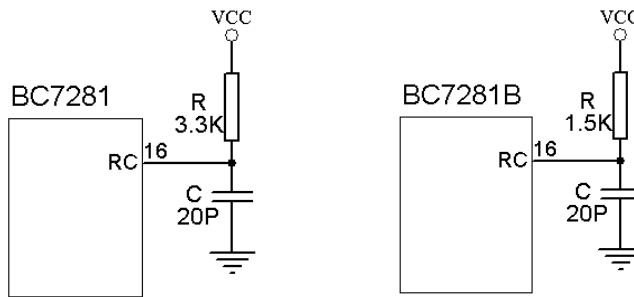
BC7280/81



, $C=0.047-0.1\mu\text{F}$ 。电容应该选用高频特性好的瓷片电容。

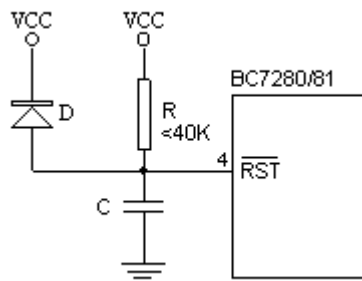
2、振荡电路

BC7281B 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 $R=1.5\text{K}-3.3\text{K}$ ， $C=20\text{pF}$ ，见下图。当采用比较小的电阻值时，震荡频率较高，可以获得更快的通讯速率和扫描频率，使得占用 CPU 时间更少，显示也更加稳定，而当采用 3.3K 电阻时，BC7281B 芯片的各项工作参数与 BC7281 及 BC7281A 保持一致，从而可以直接替换 BC7281(A)。在 $V_{\text{CC}}=5\text{V}$ ， $R=3.3\text{K}$ 的情况下，振荡电路的典型振荡频率约为 5.0MHz ；而当 $R=1.5\text{K}$ 时，震荡频率约为 9.5MHz 。BC7281 的 OSC0 端为内部振荡电路的输出端，其输出的频率为 RC 震荡频率的一半，可用作测量震荡频率之用（请不要直接测量 RC 引脚上的频率，因为测量仪器会引起震荡频率的改变，从而无法得到真实的结果），电路中一般此脚悬空即可。在电路板布线时，振荡电路的元件应尽可能地靠近 BC7281 芯片，并尽量使连线最短。



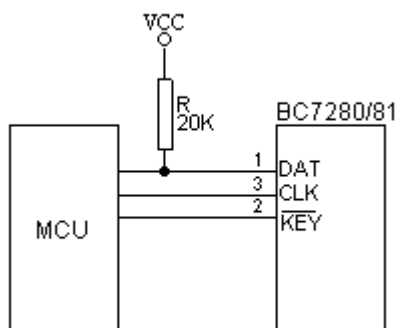
2、复位电路

芯片的 RST 引脚为复位端。因为 BC7281 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 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。RST 上的复位脉冲的最小宽度为 $20\mu\text{S}$ ，复位电路中电阻 R 的阻值一般不要超过 40K 。另外的一种方法是直接由 MCU 控制 BC7281 的复位。BC7281 的复位过程大约需要 25ms 的时间，也即 RST 为高电平约 25ms 后，BC7281 才开始工作。



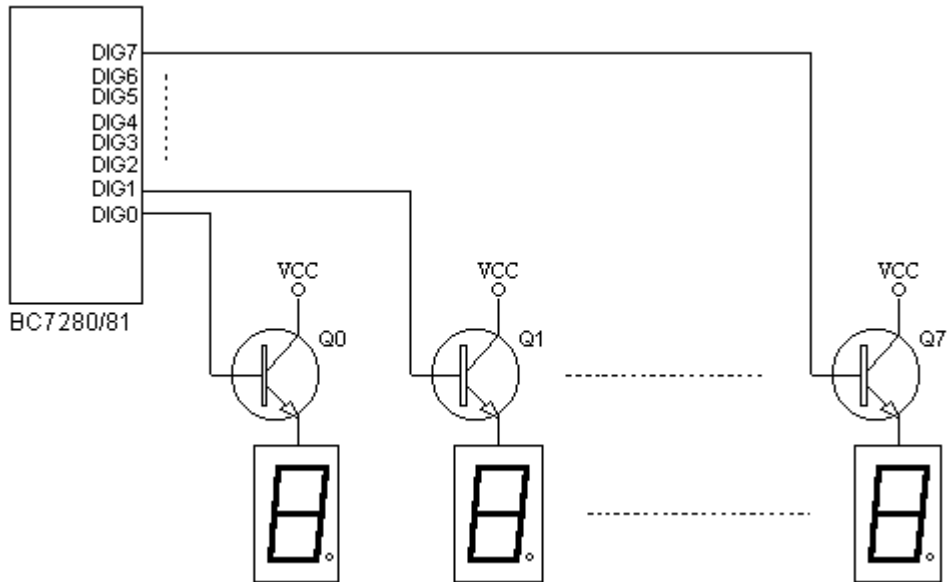
3、与 MCU 接口电路

BC7281 与 MCU 的接口共需要三根线，数据线 DAT，时钟线 CLK 和按键指示 KEY，其中 CLK 和 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 10-20K 的上拉电阻，以使其能可靠地输出高电平。

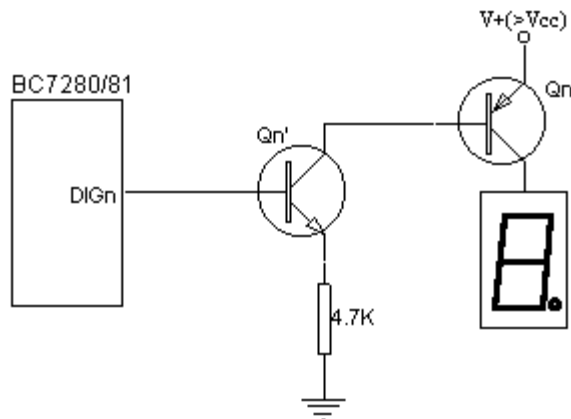


4、位驱动电路

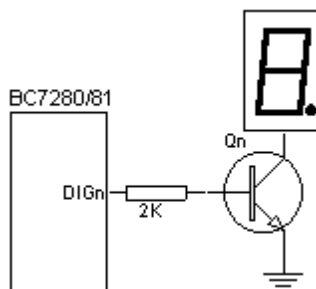
DIG0-DIG7 为位驱动输出，BC7281 适合连接共阳式的数码管，外部驱动电路比较简单，只需要 8 只 NPN 型三极管即可，三极管接成射极跟随器形式，因此基极无无限流电阻。



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：



BC7281 也可以接成共阴的方式，这时候需要对位驱动进行反相，基本电路如下图，采用 NPN 型三极管达到反相作用，也可以采用集成达林顿驱动器，作为位驱动电路。而段驱动输出部分则可以直接通过软件来控制输出的反相。需要注意的是，如果是用共阴的



接法，则移位寄存器应该使用 HC 系列的，因为 TTL 电路的高电平输出能力相对很小，会造成段驱动能力不足。而且，因为接成共阴接法时 DIGn 引脚上对地的阻抗很小因此会造成键盘部分的不能正常工作，无法读取键值，因此如果需要使用 BC7281 的键盘功能，则不应使用共阴接法。

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 $I_{ce} > 8 * I_{seg}$ （如果是 16 位的显示系统，则要求 $I_{ce} > 16 * I_{seg}$ ）， I_{seg} 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 $n * (V_+ - 0.7) / R > 8 * I_{seg}$ 的要求（如果是 16 位的显示系统，则要求 $> 16 * I_{seg}$ ），其中 n 为 Qn 的放大倍数， R 为 Qn' 的发射极电阻， I_{seg} 的单位为 mA。

5、段驱动电路

BC7281 需要外接移位寄存器构成段驱动电路。

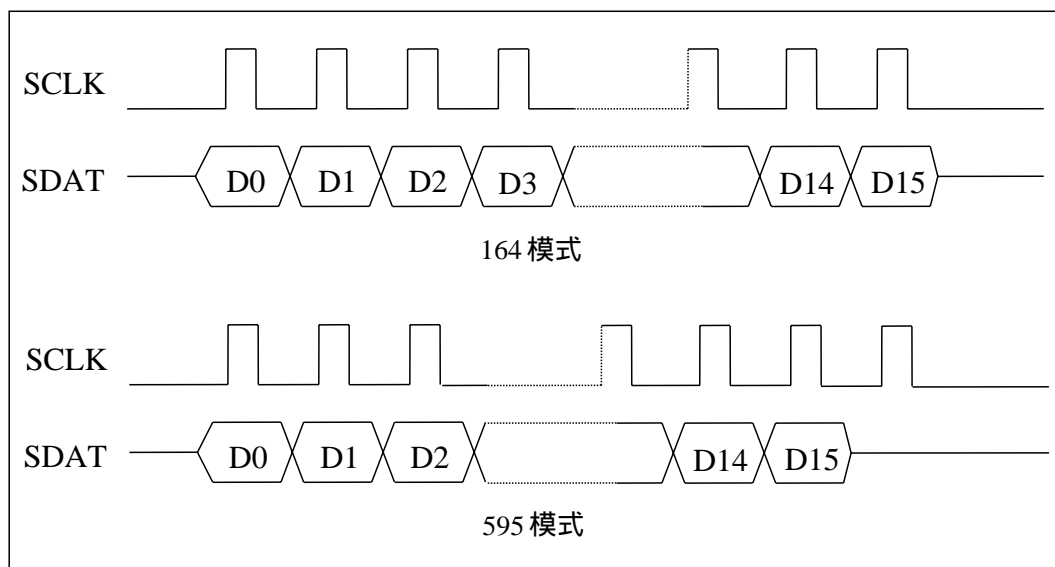
BC7281 送出的数据共 16 位，数据结构如下表，发送的顺序是高位（MSB）在前：

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 BC7281 的段驱动数据输出极性及时序均可调，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：

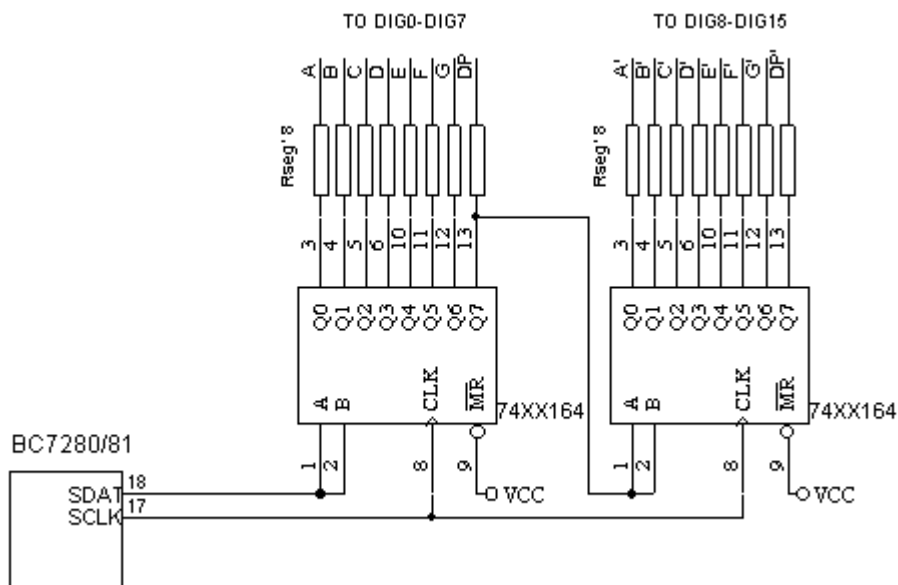


BC7281 的段驱动数据的输出极性可以由软件控制，这样使用者可以灵活地使用各种外围驱动电路。BC728X 缺省的工作模式是 164 模式、不反相输出，也就是说，输出的段驱动数据中，点亮的段对应的数据为‘0’。如果电路中在移位寄存器的后面又使用了反相输出的驱动器，或者使用的是反相输出的移位寄存器，则需要将 BC7281 的工作模式置为反相模式（INV=1）。

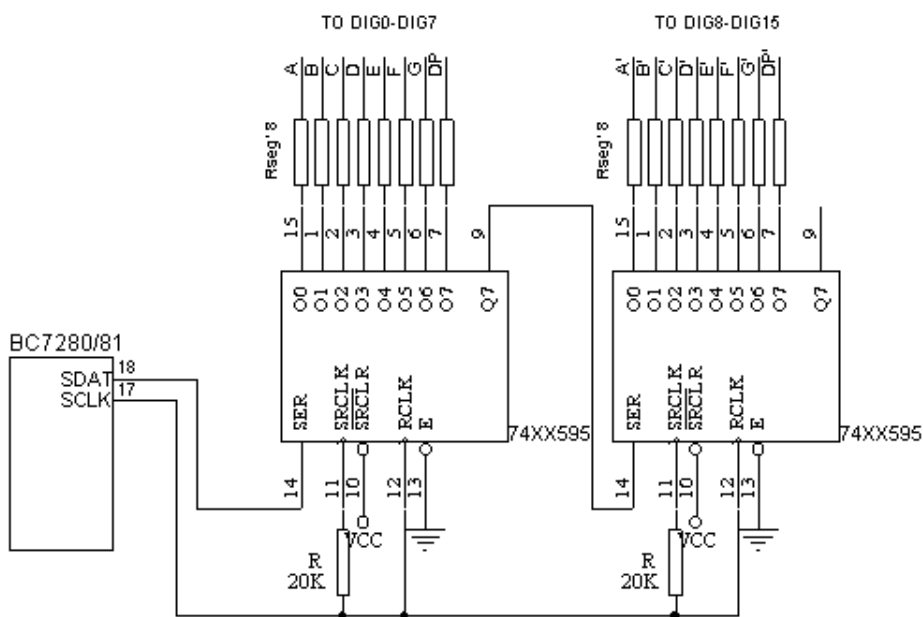
需要指出的是，INV 位所影响的只是段驱动数据的极性，而对键盘扫描的极性没有影响，因此如果使用了反相输出的驱动器，而同时又使用键盘的话，键盘矩阵必须连接在反相驱动器之前。

可以和 BC7281 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

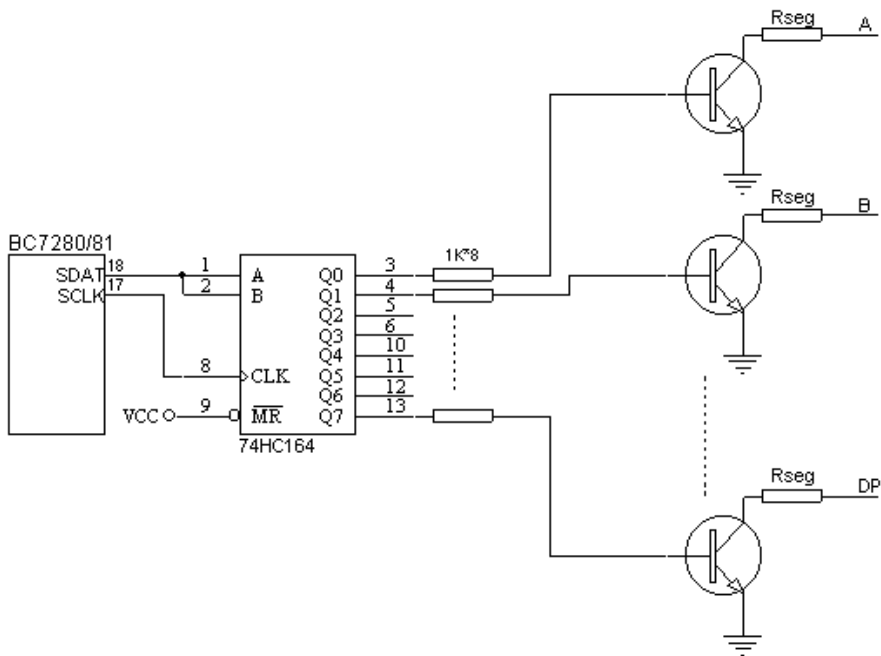


串入 SRCLK 端的电阻的作用在于与芯片的输入电容及电路的分布电容构成一积分电路，从而对输入 SRCLK 的时钟产生一微小延时，这样可以保证移位脉冲 SRCLK 迟于二级锁存

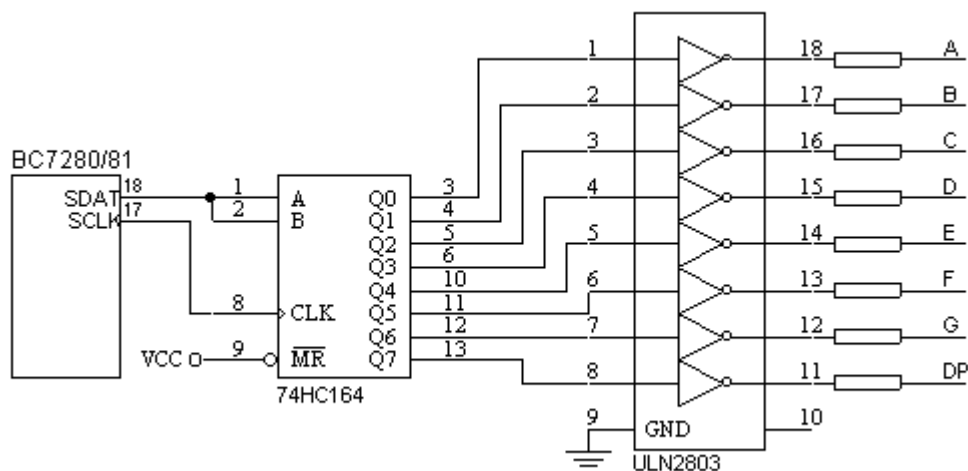
脉冲 RCLK 到达，从而保证移位寄存器的内容在变化之前可靠地进入二级锁存器。电阻的阻值没有严格要求，并且不同厂家及不同型号的芯片对应的阻值范围也会有所不同，一般而言，这个范围是比较宽的，从几 K 到几十 K 都可以，20K 的阻值可以适用于绝大多数芯片。

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 BC7281 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：

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

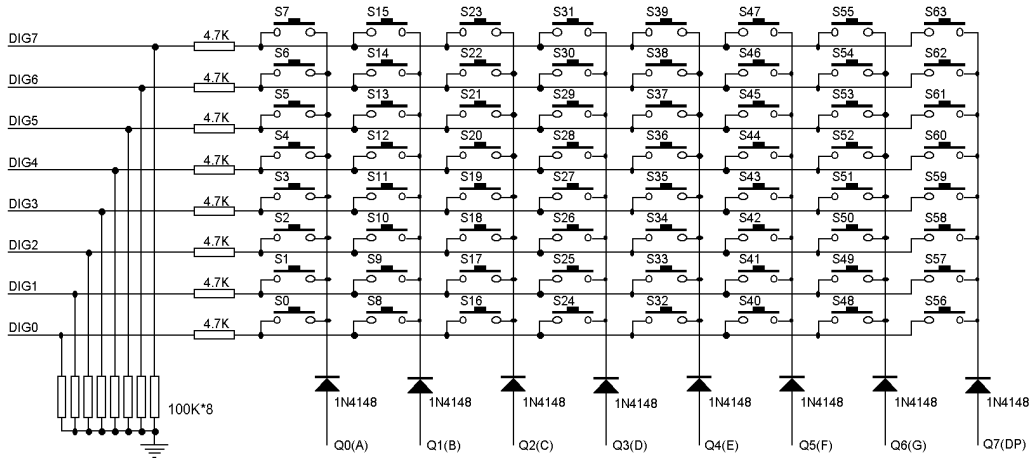
其中， V_{cc} 为 BC7281 的电源电压， V_{led} 为 LED 的管压降， V_{ol} 为段驱动电路的低电平输出电压， I_{seg} 为段电流。

注意：如果设计数码管的驱动电压大于移位寄存器的电源电压，则不管驱动电流的大小，必须采取此 2 种段驱动方案，否则因为数码管电源电压大于移位寄存器的电源电压，将造成漏电或者严重情况下移位寄存器的损坏。

6、键盘矩阵

BC7281 最多可以连接 64 个按键，按 8*8 矩阵排列，矩阵的‘行’连接到 BC7281 的驱动 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 BC7281 的键盘功能时，DIG0-DIG7 上应加以 100K 的下拉电阻，且 8 根引脚都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 BC7281B 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 ATMELE 公司的 AT89C2051，为简单起见，图中仅画出与 BC7281B 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 BC7281B 的振荡电路的元件参数为： $R=1.5K$ ， $C=20pF$ ；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 BC7281B 在 $KMS=0$ 模式时具有按键保持功能，在键值被读走之前，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，KEY 仍然连接到了单片机的 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 BC7281B，也适用于较早的 BC7280/7281/7281A，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(0x12,0xC8);         // 初始化 BC7281B 为 SCN=1,
                                // RP=1,BMS=1
    write728x(0x15,0x00);         // 在第 0 位显示 ' 0 '
    write728x(0x15,0x10);         // 在第 1 位显示 ' 0 '
    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;            // 使用指令周期小于 1uS 的高速单片机的用户
                          // 请注意在这两条语句间加入延时, 以使脉冲
                          // 宽度不小于 Tw5
    } while (dat);
    clk=0;                // Ta 时间之内再次输出一 clk 脉冲
    clk=1;                // 使用指令周期小于 1uS 的高速单片机的用户
                          // 请注意在这两条语句间加入延时, 以使脉冲
                          // 宽度不小于 Tw5

    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;          // 使用指令周期小于 1uS 的高速单片机的用户
                  // 请注意在这两条语句间加入延时, 以使脉冲
                  // 宽度不小于 Tw5
    // delay(1); // 请根据单片机的速度和 BC7281 的 RC 值决定
                  // 是否加入延时, 以使得送入每一位的总时间不
                  // 小于 Tw3 且数据保持时间不小于 Tw4,
                  // 本例中不需要
}
// delay(1); // 请根据单片机的速度和 BC7281 的 RC 值决定
                // 是否加入延时, 以使得最后一位的数据保持时间不
                // 小于 Tw4, 本例中不需要
dat=1;           // 恢复 dat 为高电平
}

// *****
// *           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;        // 使用指令周期小于 1uS 的高速单片机的用户
                  // 请注意在这两条语句间加入延时, 以使脉冲
                  // 宽度不小于 Tr6

    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;        // 使用指令周期小于 1uS 的高速单片机的用户
                  // 请注意在这两条语句间加入延时, 以使脉冲
                  // 宽度不小于 Tr6

    for (bit_counter=0; bit_counter<8; bit_counter++)
    {
        // 接收 8 个 bit
        // delay(1); // 请根据单片机的速度和 BC7281 的 RC 值决定
                    // 是否加入延时, 以满足数据建立时间 Tr2、Tr3
                    // 本例中不需要

        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;           // 使用指令周期小于 1uS 的高速单片机的用户
    }
}

```

```

// 请注意在这两条语句间加入延时，以使脉冲
// 宽度不小于 Tr6

}
// delay(2); // 请根据单片机的速度和 BC7281 的 RC 值决定
// 是否加入延时，以满足 DAT 恢复时间 Tr5
// 本例中不需要

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
;*****

```

```

START:      ORG      100H
            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                ;BC7281B 初始化
CALL    SEND
MOV     DATA_OUT, #C8H                ;设定为 SCN=1, RP=1, BMS=1
CALL    SEND
MOV     DATA_OUT, #15H                ;HEX 译码指令
CALL    SEND
MOV     DATA_OUT, #00H                ;在第 0 位显示 '0'
CALL    SEND
MOV     DATA_OUT, #15H                ;
CALL    SEND
MOV     DATA_OUT, #10H                ;在第 1 位显示 '0'
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    ; 使用指令周期小于 1uS 的高速单片机的用户
                ; 请注意在这两条语句间加入延时, 以使脉冲
                ; 宽度不小于 Tw5
WAIT1: JB      DAT, SEND    ; 检测 DAT 如为高继续输出脉冲
                CLR     CLK    ; Ta 时间之内再输出一 CLK 脉冲
                SETB    CLK    ; 使用指令周期小于 1uS 的高速单片机的用户
                ; 请注意在这两条语句间加入延时, 以使脉冲
                ; 宽度不小于 Tw5
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    ; 使用指令周期小于 1uS 的高速单片机的用户
                ; 请注意在这两条语句间加入延时, 以使脉冲
                ; 宽度不小于 Tw5
                MOV     A, DATA_OUT
                RL      A
                MOV     DATA_OUT, A    ; DATA_OUT 左移一位
                ; NOP        ; 请根据单片机的速度和 BC7281 的 RC 值决定
                ; NOP        ; 是否加入延时, 以使得送入每一位的总时间不
                ; NOP        ; 小于 Tw3 且数据保持时间不小于 Tw4
                ; NOP        ; 本例中不需要
                DJNZ   BIT_COUNT, SEND_LOOP
                ; NOP        ; 请根据单片机的速度和 BC7281 的 RC 值决定
                ; NOP        ; 是否加入延时, 以使得最后一位的数据保持时间
                ; NOP        ; 不小于 Tw4, 本例中不需要
                SETB   DAT    ; 恢复 DAT 为高电平
                RET

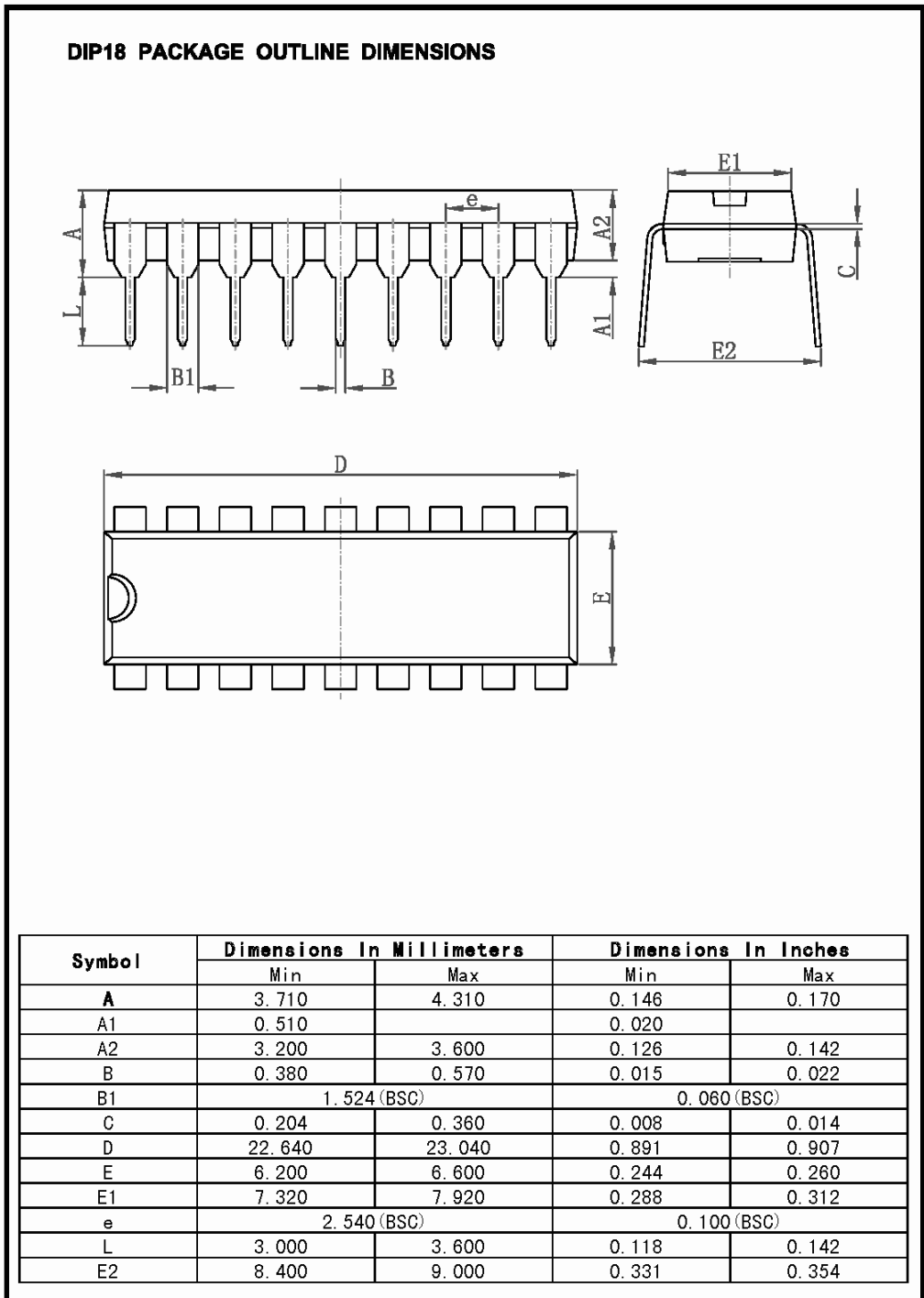
; *****
; * 从 BC728X 接收一个字节子程序, 接收到的数据存于 DATA_IN
; *****
RECEIVE:      CLR     CLK    ; 发出一单一 CLK 脉冲
                SETB   CLK    ; 使用指令周期小于 1uS 的高速单片机的用户
                ; 请注意在这两条语句间加入延时, 以使脉冲
                ; 宽度不小于 Tr6
WAIT3: JB     DAT, WAIT3    ; 等待 DAT 低电平响应信号
                CLR     CLK    ; 再发出一 CLK 脉冲, 准备接收数据
                SETB   CLK    ; 使用指令周期小于 1uS 的高速单片机的用户
                ; 请注意在这两条语句间加入延时, 以使脉冲
                ; 宽度不小于 Tr6
                MOV     BIT_COUNT, #8

```

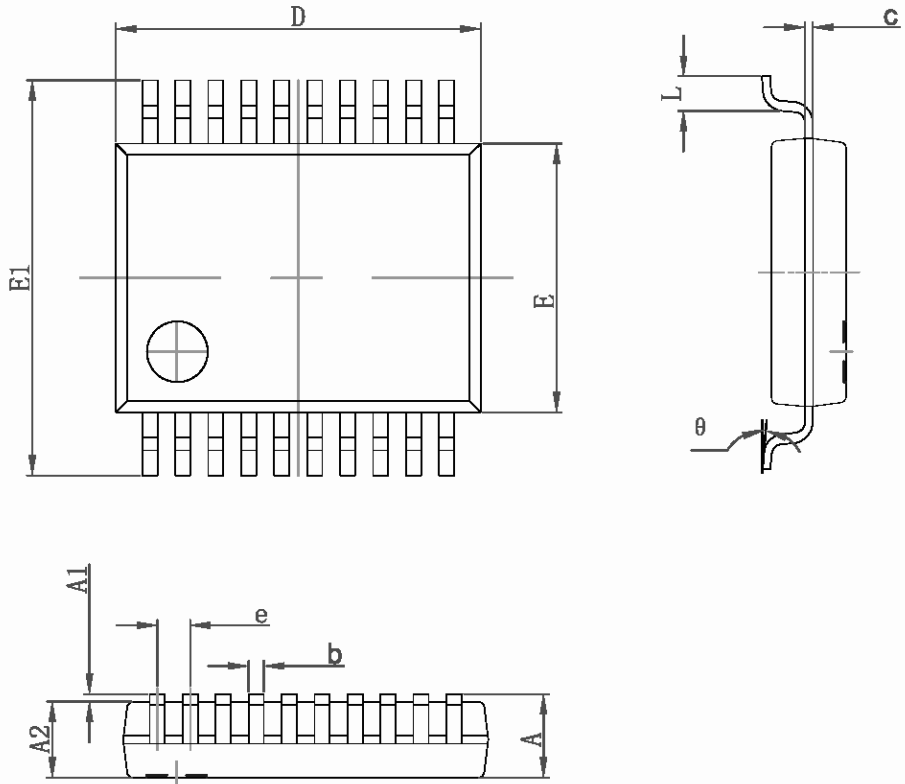
```
RECV_LOOP:    ;NOP      ;请根据单片机的速度和 BC7281 的 RC 值决定
              ;NOP      ;是否加入延时，以满足数据建立时间 Tr2、Tr3
              ;NOP      ;本例中不需要
MOV           A,DATA_IN
MOV           C,DAT           ;读入一位
RLC          A
MOV           DATA_IN,A
CLR          CLK           ;发出 CLK 脉冲
SETB        CLK
DJNZ        BIT_COUNT,RECV_LOOP
              ;NOP      ;请根据单片机的速度和 BC7281 的 RC 值决定
              ;NOP      ;是否加入延时，以满足 DAT 恢复时间 Tr5
              ;NOP      ;本例中不需要
RET
```

END

附录 1：封装尺寸



SSOP20(209mil) PACKAGE OUTLINE DIMENSIONS



Symbol	Dimensions In Millimeters		Dimensions In Inches	
	Min	Max	Min	Max
A		1.730		0.068
A1	0.050	0.230	0.002	0.009
A2	1.400	1.600	0.055	0.063
b	0.220	0.380	0.009	0.015
c	0.090	0.250	0.004	0.010
D	7.000	7.400	0.276	0.291
E	5.100	5.500	0.201	0.217
E1	7.600	8.000	0.299	0.315
e	0.65(BSC)		0.026(BSC)	
L	0.550	0.950	0.022	0.037
theta	0°	8°	0°	8°

附录 2

BC728X 应用笔记（一）

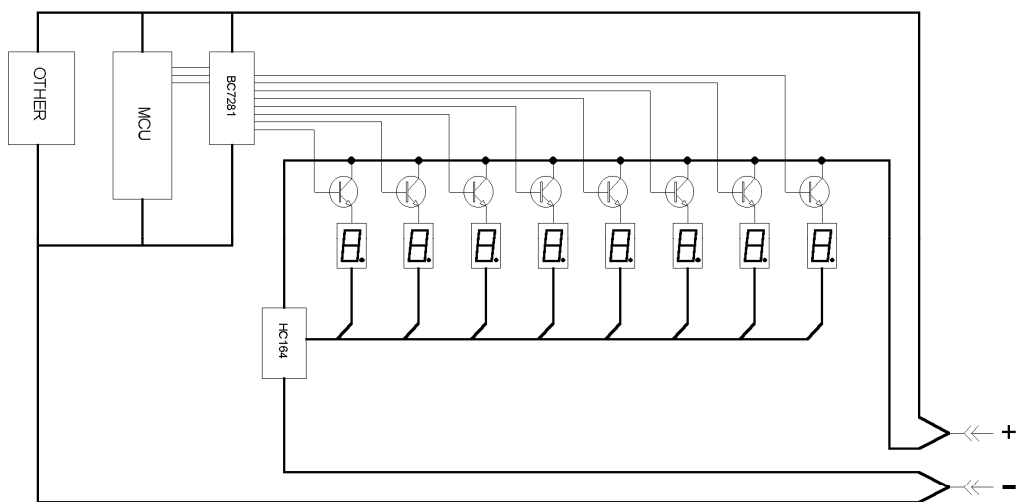
电路板的布线

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 $25*8*2=400\text{MA}$ ，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。建议的电路板走线方式如下图：



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

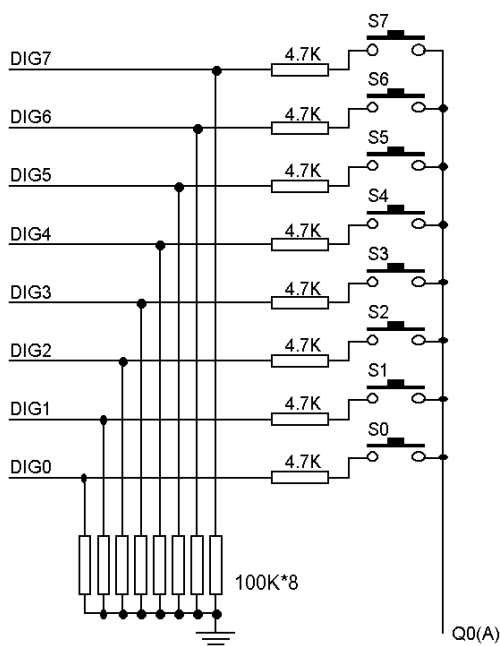
作者：凌志比高科技有限公司 矫健

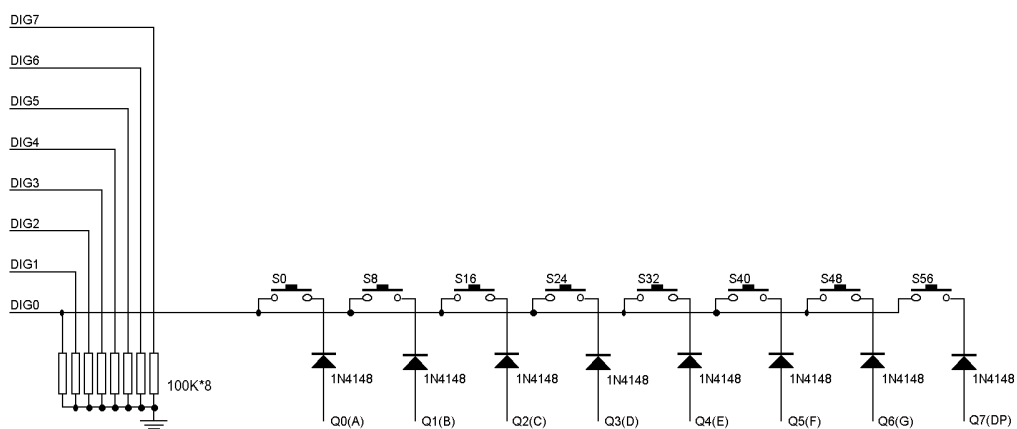
BC7281 具有多达 16 位的数码管驱动和 64 键的键盘管理能力，这足以满足一般的单片机系统的要求。事实上，对于大多数的仪器仪表类应用来说，一般的键盘数量不会多于 8 个，在这种情况下，BC7281 的键盘电路还可以进行简化。

BC7281 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，一方面因为二极管比电阻的成本高，另一方面也因为没有了二极管的管压降，键盘的可靠性会更高。

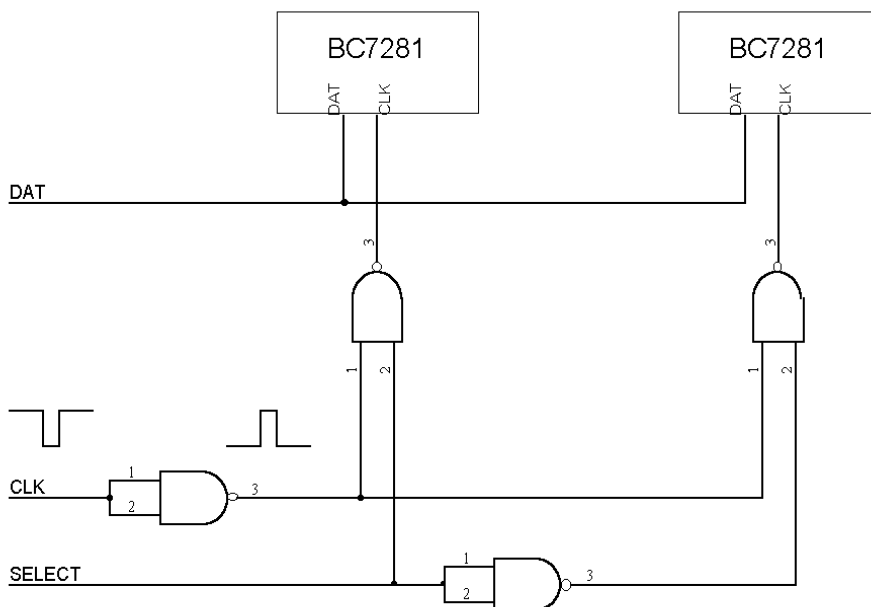
附录 4 :

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

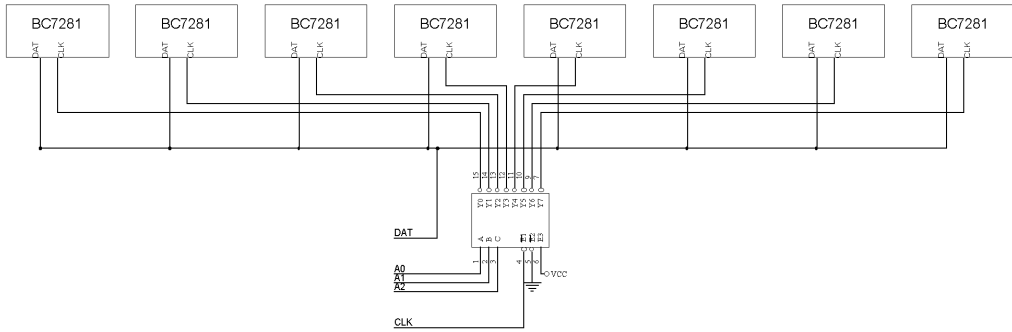
单片 BC7281 具有多达 16 位的数码管驱动能力，但是，在某些需要更多位数的 LED 显示的应用中，因为 BC7281 并没有“片选”信号的输入端，有些用户会对如何在一个系统中同时使用多片 BC7281 感到困惑，其实因为 BC7281 的通讯开始时是由 CLK 引脚启动握手信号，而 DAT 引脚在建立握手信号之前一直是高阻状态，所以只要将单片机送出的 CLK 信号按需要分配给指定的 BC7281 芯片就行了。



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。

附录 5 :

BC7281 系列型号性能对照表

特性	BC7281	BC7281A	BC7281B
抗干扰模式	-	-	有
节能模式 (亮度控制)	-	-	有
显示关闭, 保留键盘扫描功能	-	-	有
键盘工作模式	1 种	2 种, 兼容 BC7281	2 种, 兼容 BC7281/BC7281A
闪烁控制模式	1 种	2 种, 兼容 BC7281	2 种, 兼容 BC7281/BC7281A
震荡电路参数 (R/C)	3.3K/20P	3.3K/20P	1.5K-3.3K/20P
移位寄存器脉冲	标准宽度	标准宽度	适当增加宽度, 使其在高工作频率时可以适应低速的移位寄存器
键盘扫描脉冲	标准宽度	标准宽度	适当增加宽度, 使其在配合分布电容比较大的薄膜键盘使用时可靠性更好