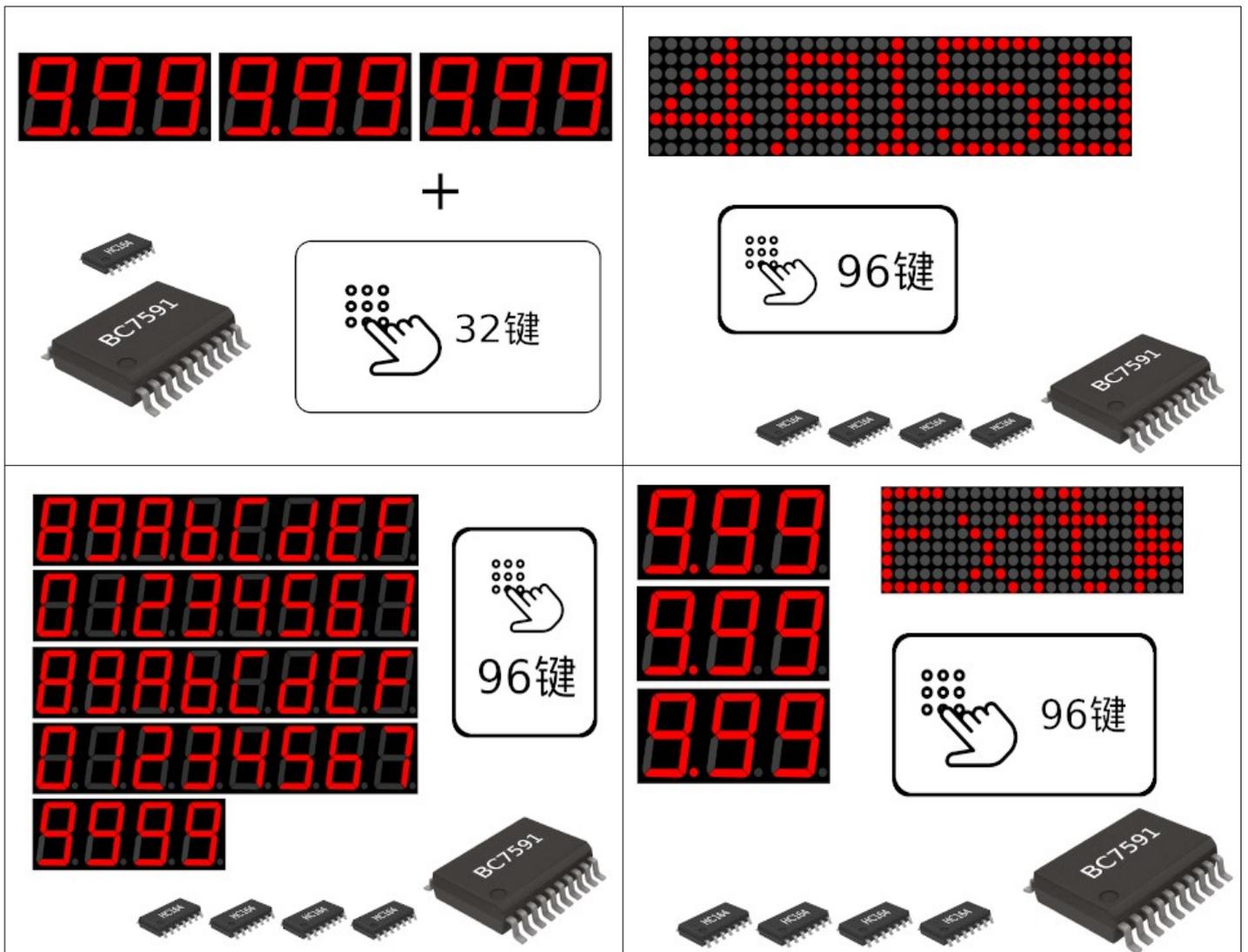


# BC7591

## 256 段 LED 显示及 96 键键盘接口芯片

- 灵活配置为 64 至 256 段(9-36 数码管)显示
- 对应 32 至 96 键键盘接口
- 内置数码管译码器
- 单独 LED 闪烁功能，闪烁速度可调
- 针对点阵显示的画面左右/上下平移功能
- 16 级线性亮度控制
- 256 个 LED 均可独立寻址
- 键盘支持任意数量组合键和长按键
- 支持常开/常闭按键
- 直接键值输出
- UART 接口，并可实现多片联用
- 3.0-5.5V 工作电压
- SSOP20 封装
- 键盘接口与 BC6xxx 芯片协议兼容

### 典型应用方案：



## 简述

BC7591, 提供 LED 显示驱动和键盘接口功能, 仅需外接移位寄存器(HC164)。根据外接移位寄存器的数量不同, 可以灵活配置为从 64 段 LED(9 位数码管)/32 键键盘, 至 256 段 LED(36 位数码管)/96 键键盘的不同驱动电路。每个显示段均可独立寻址, 方便作为指示灯单独控制; 同时具有单独 LED 闪烁功能, 闪烁速度可调, 闪烁过程无需 CPU 参与, 极大程度简化了编程。

用作数码管显示驱动时, 可使用内建的数码管译码器, 只需送入待显示的数值。同时, 数码管译码采用 7 段译码方式, 最高位可作为小数点单独控制, 或将各显示位的小数点用以驱动额外的数码管显示位, 亦有专门的译码指令可直接完成这些扩展的显示位的数字显示。外接 1 片 8 位移位寄存器 74HC164 时, 可构成带负号的 9 位数码管显示; 而外接 2 片时可构成 18 位数码管显示; 最多可构成 36 位数码管显示驱动电路。

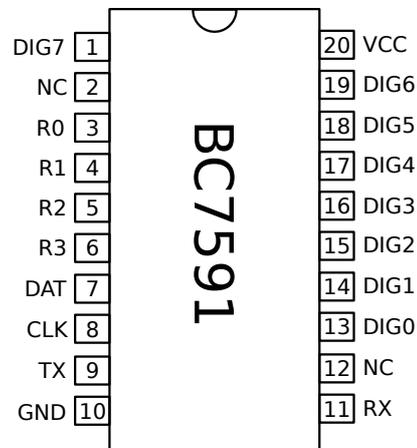
BC7591 还可用于驱动 LED 点阵显示, 或数码管和 LED 点阵的组合。作为 LED 点阵驱动时, 最多可以驱动 32x8 或 16x16 的点阵显示。BC7591 提供了专门用于点阵显示的指令, 可以实现屏幕的左右和上下滚屏显示等常用功能, 而无需大量增加 CPU 的计算开销。BC7591 完成了点阵显示驱动所需的大部分工作, 即便使用低端的 MCU, 也可以提供通常高性能 MCU 才会采用的复杂点阵显示输出, 并且编程得以大幅简化。

根据外接移位寄存器的位数, BC7591 可以提供从 32 键至 96 键的键盘接口。键盘支持常开/常闭按键, 并支持任意数量和形式的组合键和长按键。

BC7591 采用 1/8 的扫描占空比, 相比 1/16 占空比, 在相同亮度下, 切换频率和瞬间驱动电流都可以降低一倍, 更有利于降低扫描电路所产生的噪声干扰。同时, BC7591 可提供 16 级的显示亮度控制。

采用 UART 接口, 可以方便地与各种 MCU 接口, 或直接由 PC 机控制, 搭配 USB-UART 转换器制成电路简洁易用的 PC 外置数码管或点阵显示器。亦可方便地实现隔离或转换为 RS-485/422 接口实现远程显示和键盘。采用简单的电路, 可实现片选功能, 实现一个 UART 口下的多片联用。

## 引脚定义



## 引脚说明

名称	序号	功能
VCC	20	正电源引脚，电压范围 3.0~5.5V
GND	10	接地端
TX	9	UART 输出，漏极开路输出(Open Drain)
RX	11	UART 接收
R0~R3	3,4,5,6	键盘矩阵行接口，内部带上拉电阻
DIG0-DIG7	1,13,14,15,16,17,18,19	数码管位或点阵行驱动输出，低电平有效，高电平为高阻输出
DAT	7	移位寄存器数据线
CLK	8	移位寄存器时钟线
NC	2,12	空脚，无连接

(表 1: 引脚说明)

## 显示驱动

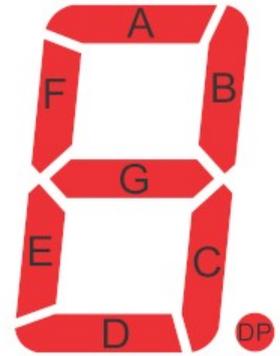
### 显示方式

BC7591 可以驱动数码管，或 LED 点阵显示，或者二者的组合。从 BC7591 的角度，外部显示器的形式对电路的工作并没有影响，不过从使用者的角度，如果显示的内容和所使用的显示器件不匹配，会看到显示为杂乱无意义的内容。比如，用户用数码管译码指令写入数字，但电路中所接的显示器却为点阵显示器，则结果是只能看到点阵显示器上一些杂乱的显示点，并不能看到所希望的数字显示。

## 数码管显示电路

数码管的各显示段定义如图：

数码管电路分为 7 段和 8 段两种连接方式，8 段方式即数码管上从 A 段到小数点 DP 的 8 个 LED 均连接到电路中，因此每个显示位均可以显示小数点。不过在实际应用中，往往并非每个显示位都需要能显示小数点，而且小数点位置往往是固定的，每个显示位的小数点都接入驱动并没有必要。因此，还可以有 7 段接法，即只连接数码管的 A-G 显示段，而将小数点闲置不连接，或者将必要的小数点接入静态驱动。这样每个显示位都可以节约一个显示段，可以用这些闲置的小数点段去驱动额外的数码管，增加显示位数。



BC7591 内含数码管译码器，可以将接收到的数值转换成该数值对应的 7 段显示器字形，数码管的使用得到最大程度的简化(参见后文“译码显示”指令部分)。译码器同样适用于各显示位小数点扩展出来的显示位，有专门的指令用于扩展位的译码显示。所有的译码显示指令，均为 7 段译码，即只更新 A-G 显示段，这样在 8 段连接方式时，小数点的显示不受数码管上数字更新的影响，而在 7 段连接方式时，可以保证扩展显示位的显示内容不受其它显示位的影响。

驱动数码管时，位驱动(点阵显示中行驱动)由 BC7591 直接输出，而段驱动(点阵显示中列驱动)则由外接的移位寄存器输出。移位寄存器的 Q0 对应显示段 A，Q1 对应显示段 B，以此类推，至 Q7 对应小数点 DP。当外接 1 片 8 位移位寄存器(74HC164)时，可以驱动 64 段显示。最多可以连接 4 片 8 位移位寄存器，或 2 片 16 位移位寄存器，驱动 256 段显示。

BC7591 的位驱动具有 100mA 的驱动能力，在显示段数少于 64 段，即使用 1 片 HC164 移位寄存器时，位驱动可以直接由片内驱动，如果使用多于 8 位的移位寄存器，位驱动上应该外接 PNP 型三极管作为外接驱动，以获得理想的显示亮度。

LED 的亮度主要取决于本身发光效率和工作电流两个因素。因为 BC7591 的扫描驱动占空比为 1/8，即只有 1/8 的时间是实际发光的时间，因此应该优先选用亮度高的 LED。LED 的亮度一般用 mcd 做单位，建议使用的 LED 数码管和点阵显示的亮度，在 10mA 下，至少要大于 3 mcd。

LED 的驱动电流，取决于驱动电流的驱动能力和限流电阻的大小。因为 LED 的亮度并非随电流线性增加，电流增大到一定程度后，再增加电流并不会显著增加 LED 的亮度。一般以 5-10mA 区间效率最高。

限流电阻的计算，采用公式

$$R = (V_{\text{seg}} - V_{\text{LED}} - V_{\text{DIG}}) / I_{\text{seg}}$$

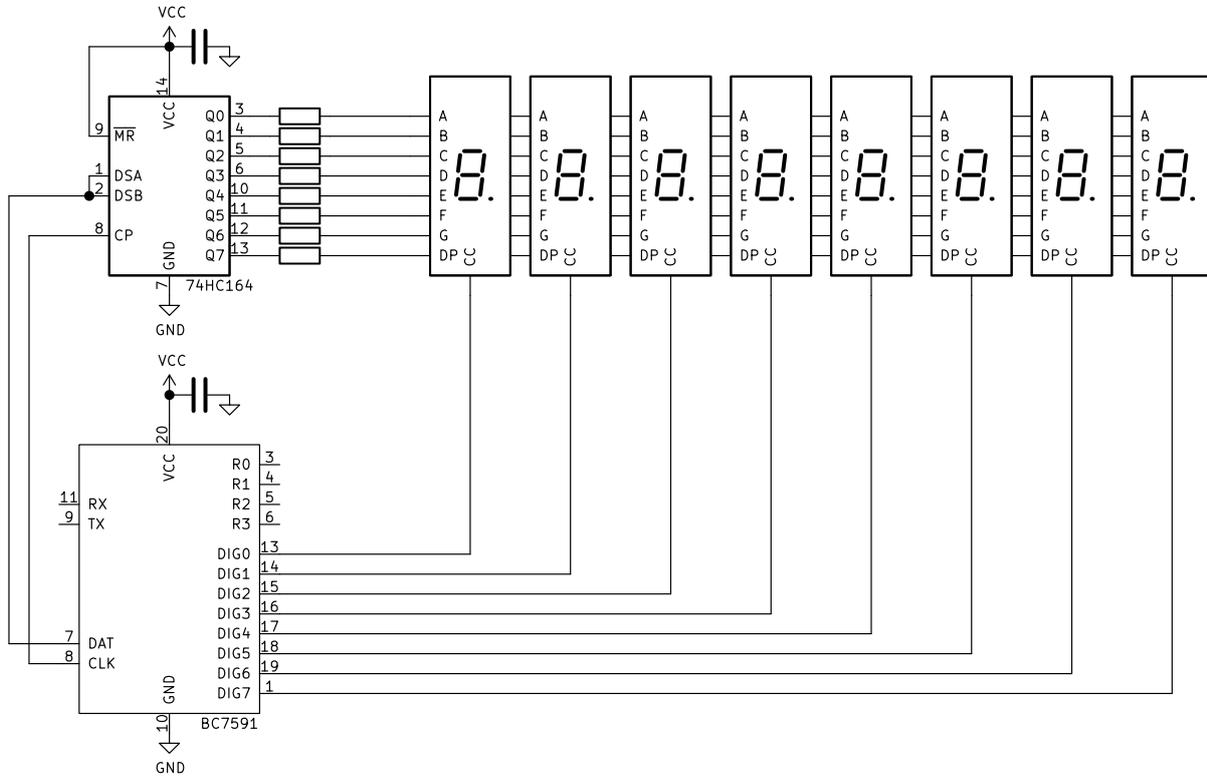
其中， $V_{\text{seg}}$  为移位寄存器在输出 LED 段电流  $I_{\text{seg}}$  时的输出电压， $V_{\text{LED}}$  为 LED 的正向管压降， $V_{\text{DIG}}$  为位驱动输出在  $8 \times I_{\text{seg}}$  时的输出电平。

例 1：电源电压 5V，设计 LED 电流 12mA，LED 管压降 2.0V，移位寄存器输出电压 4.2V，位驱动电压 1V，则  $R = (4.2 - 2.0 - 1) / 0.012 = 100\Omega$

例 2：电源电压 3.3V，设计 LED 电流 10mA，LED 管压降 1.9V，移位寄存器输出电压 2.8V，位驱动电压 0.7V，则  $R = (2.8 - 1.9 - 0.7) / 0.01 = 20\Omega$

## 8 位数码管 8 段连接方式

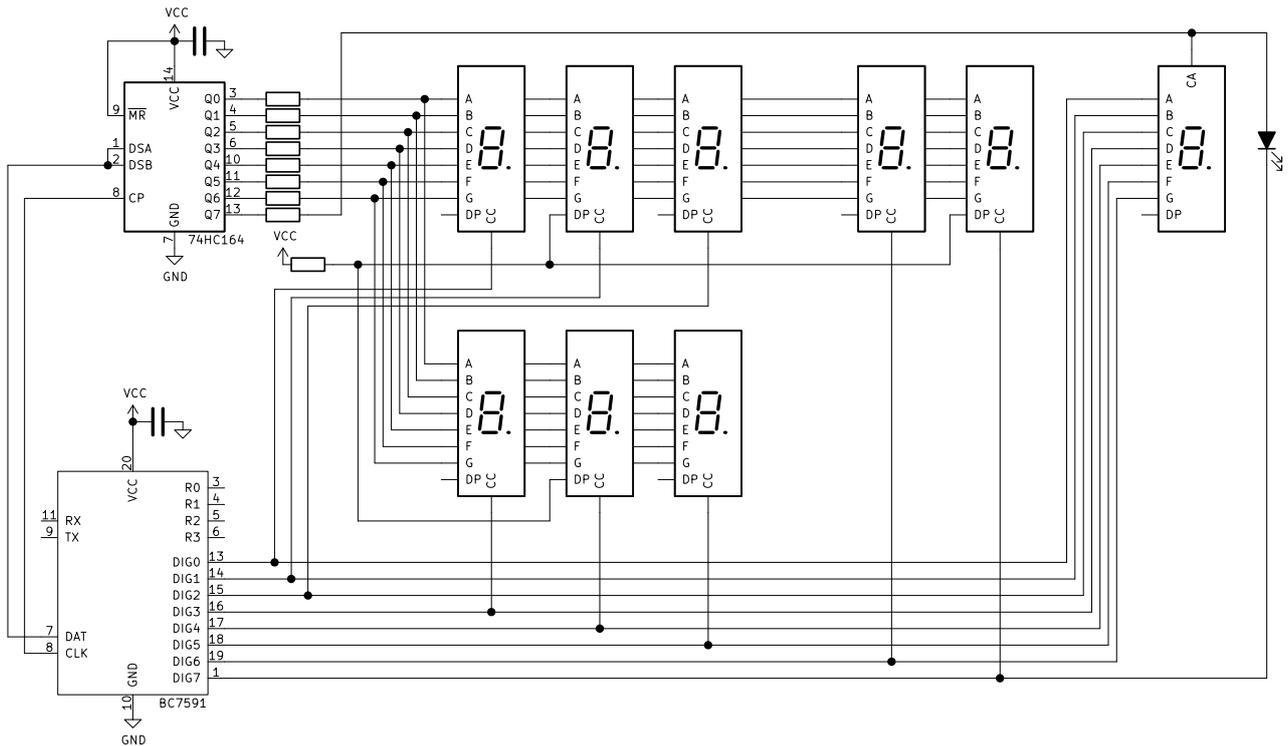
BC7591 的内置位驱动，可以有能力驱动 8 位数码管/64 段 LED 显示，因此显示段数小于等于 64 段时，电路最为简洁，仅需要外接 1 片 74HC164 作为移位寄存器。下图是用 8 段方式连接 8 位数码管，每个显示位的小数点都可单独控制点亮。图中未包含键盘电路，键盘电路请参见后文键盘部分。



(图 1: BC7591-8 位数码管 8 段连接)

## 9 位数码管 7 段连接方式

如果不使用小数点，或者小数点的位置固定，则可以把每个显示位的小数点用作驱动额外的显示位。需要注意的是，电路连接上，普通的显示位使用共阴式数码管，而额外的显示位，须使用共阳式数码管。下图是 BC7591 构成一个 3 组 3 位固定小数点的数码管显示的电路图，每组都最大显示 99.9 的数字。图中显示位 DIG7 和小数点驱动 Q7 所驱动的 LED，即额外显示位的小数点 DP 段，用作了独立的 LED 指示灯。

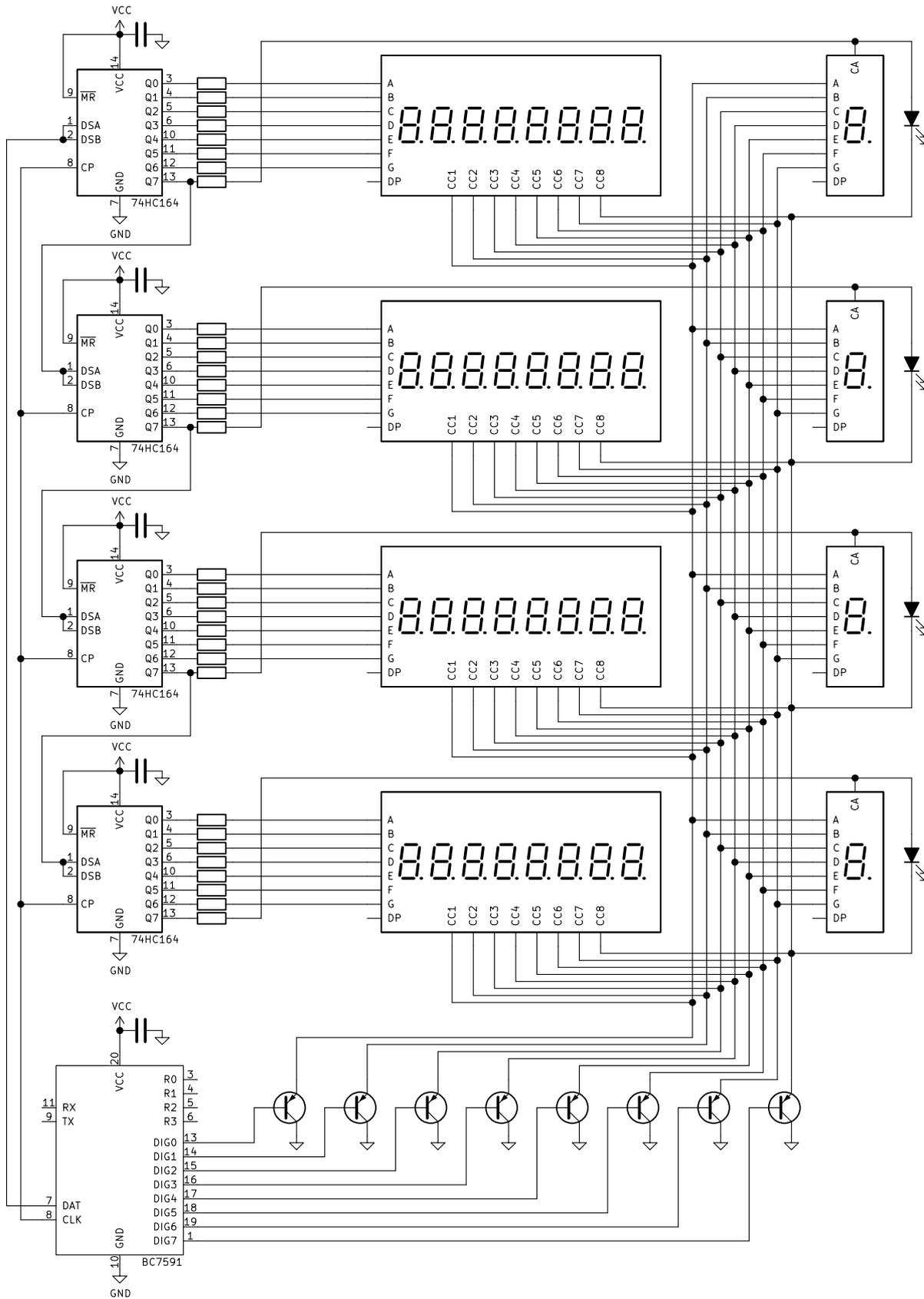


(图 2: BC7591-9 位数数码管 7 段连接)

### 36 位数数码管 7 段连接方式

当显示 LED 数多于数码管 8 位或 64 段，因为 BC7591 自身所提供的位(行)驱动驱动能力不够，会造成显示亮度变低，此时应该使用外部驱动。外部驱动最简单的方式是使用 1 只 PNP 型三极管，连接为电压跟随器的方式，基极无需再串入电阻。

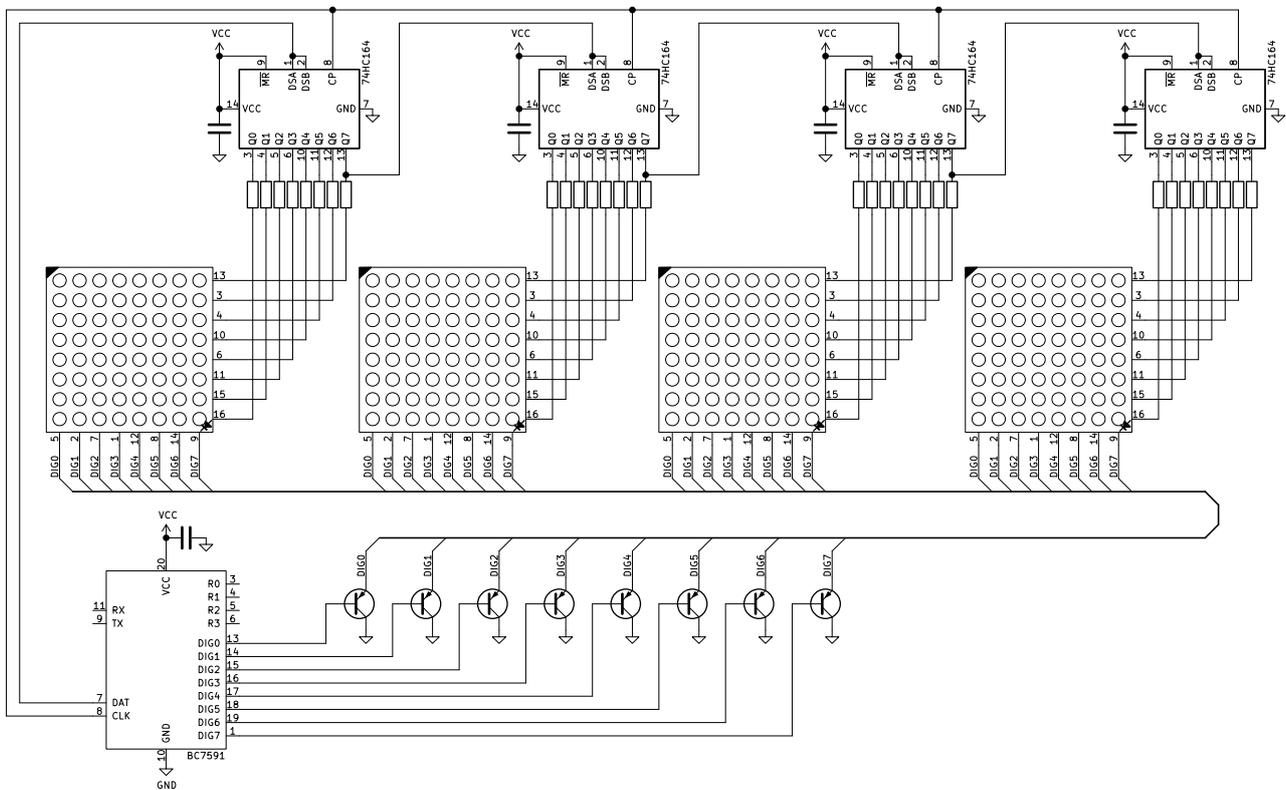
下图是 BC7591 以 7 段方式连接 36 位数数码管应用，请注意额外的 4 位数数码管是与其它数码管不同的共阳型。



(图 3: BC7591-36 位数码管 7 段连接)

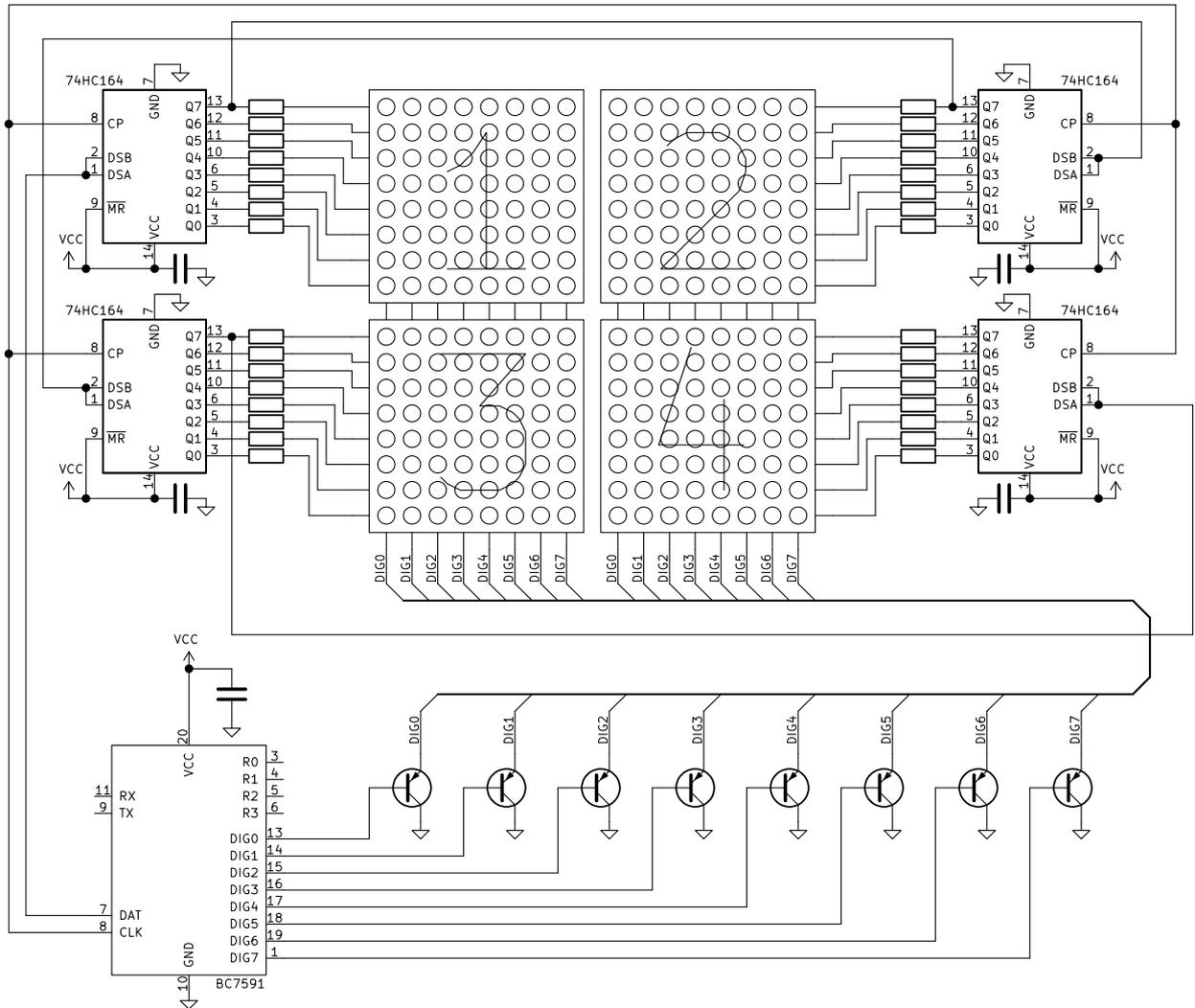
## 点阵显示电路

如果连接 LED 点阵，则 BC7591 可以用作点阵显示驱动器。以 BC7591 的移位寄存器输出为列，位驱动为行，最多可以驱动  $32 \times 8 = 256$  个显示段(点)。点阵的排列方式可以根据需要，排列为单行  $32 \times 8$  的形式可以显示单行英文字符和简单汉字及图形，排列为  $16 \times 16$  则可以显示一个复杂汉字或者符号。



(图 4: BC7591-32x8 点阵)

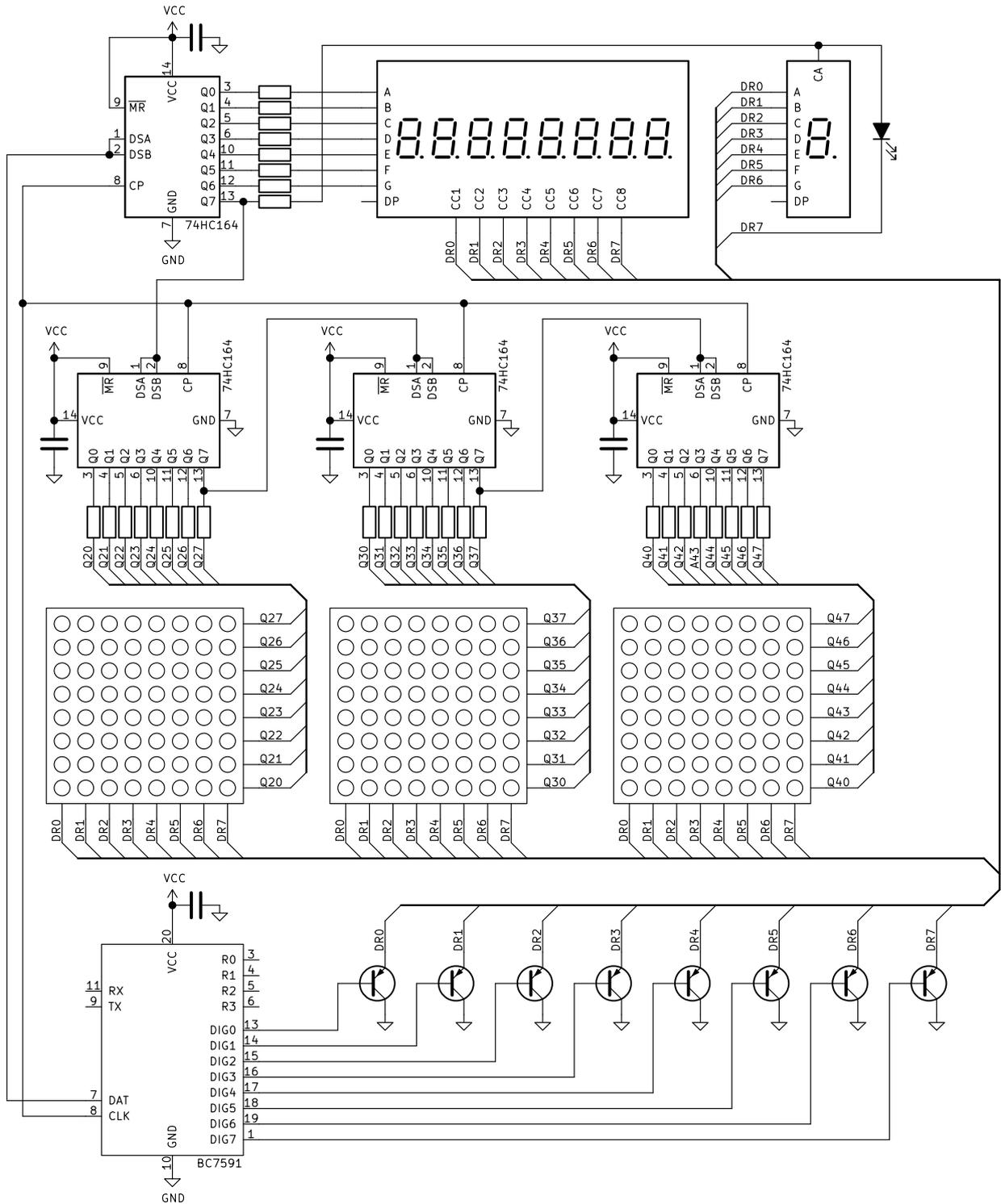
BC7591 驱动的  $16 \times 16$  点阵显示电路实际上与  $32 \times 8$  电路完全相同，仅仅是单行的 4 只  $8 \times 8$  显示模块改为了 2 行排列，外观为  $16 \times 16$  点阵，但电路并非真正的  $16 \times 16$  驱动电路，因为 BC7591 的行驱动输出只有 8 条，无法形成  $16 \times 16$  的驱动输出。



(图 5: BC7591-16x16 点阵)

## 混合显示电路

BC7591 也可以用做混合的数码管显示和点阵显示，下图是一个 BC7591 用作 9 位数码管显示加 24x8 点阵显示的应用电路：



(图 6: BC7591-9 位数码管+24x8 点阵)



因为很多情况并非每个显示位的小数点都会用到，而且小数点的位置经常是固定的，可以直接连接电源采用静态驱动，因此 BC7591 还支持一种方式，即 7 段数码管驱动方式，每个显示寄存器的最高位 bit7，可以单独作为额外的数码管显示位的段，BC7591 有专门的译码指令可以像对待其它显示位一样，直接写入数值完成译码转换。对扩展显示位的译码指令同样是 7 段译码，不包括小数点段，扩展位的小数点，可以用作单独的 LED 指示灯。如果给扩展的数码管位编号 X1 - X4，小数点位命名为 L1-L4，则在 7 段连接方式下，数码管段和显示寄存器的对应关系如下表：

		显示寄存器地址																														
		0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F	0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E
b <sub>7</sub>	A <sub>x1</sub>	B <sub>x1</sub>	C <sub>x1</sub>	D <sub>x1</sub>	E <sub>x1</sub>	F <sub>x1</sub>	G <sub>x1</sub>	L <sub>1</sub>	A <sub>x2</sub>	B <sub>x2</sub>	C <sub>x2</sub>	D <sub>x2</sub>	E <sub>x2</sub>	F <sub>x2</sub>	G <sub>x2</sub>	L <sub>2</sub>	A <sub>x3</sub>	B <sub>x3</sub>	C <sub>x3</sub>	D <sub>x3</sub>	E <sub>x3</sub>	F <sub>x3</sub>	G <sub>x3</sub>	L <sub>3</sub>	A <sub>x4</sub>	B <sub>x4</sub>	C <sub>x4</sub>	D <sub>x4</sub>	E <sub>x4</sub>	F <sub>x4</sub>	G <sub>x4</sub>	L <sub>4</sub>
b <sub>6</sub>	G <sub>1</sub>	G <sub>2</sub>	G <sub>3</sub>	G <sub>4</sub>	G <sub>5</sub>	G <sub>6</sub>	G <sub>7</sub>	G <sub>8</sub>	G <sub>9</sub>	G <sub>10</sub>	G <sub>11</sub>	G <sub>12</sub>	G <sub>13</sub>	G <sub>14</sub>	G <sub>15</sub>	G <sub>16</sub>	G <sub>17</sub>	G <sub>18</sub>	G <sub>19</sub>	G <sub>20</sub>	G <sub>21</sub>	G <sub>22</sub>	G <sub>23</sub>	G <sub>24</sub>	G <sub>25</sub>	G <sub>26</sub>	G <sub>27</sub>	G <sub>28</sub>	G <sub>29</sub>	G <sub>30</sub>	G <sub>31</sub>	G <sub>32</sub>
b <sub>5</sub>	F <sub>1</sub>	F <sub>2</sub>	F <sub>3</sub>	F <sub>4</sub>	F <sub>5</sub>	F <sub>6</sub>	F <sub>7</sub>	F <sub>8</sub>	F <sub>9</sub>	F <sub>10</sub>	F <sub>11</sub>	F <sub>12</sub>	F <sub>13</sub>	F <sub>14</sub>	F <sub>15</sub>	F <sub>16</sub>	F <sub>17</sub>	F <sub>18</sub>	F <sub>19</sub>	F <sub>20</sub>	F <sub>21</sub>	F <sub>22</sub>	F <sub>23</sub>	F <sub>24</sub>	F <sub>25</sub>	F <sub>26</sub>	F <sub>27</sub>	F <sub>28</sub>	F <sub>29</sub>	F <sub>30</sub>	F <sub>31</sub>	F <sub>32</sub>
b <sub>4</sub>	E <sub>1</sub>	E <sub>2</sub>	E <sub>3</sub>	E <sub>4</sub>	E <sub>5</sub>	E <sub>6</sub>	E <sub>7</sub>	E <sub>8</sub>	E <sub>9</sub>	E <sub>10</sub>	E <sub>11</sub>	E <sub>12</sub>	E <sub>13</sub>	E <sub>14</sub>	E <sub>15</sub>	E <sub>16</sub>	E <sub>17</sub>	E <sub>18</sub>	E <sub>19</sub>	E <sub>20</sub>	E <sub>21</sub>	E <sub>22</sub>	E <sub>23</sub>	E <sub>24</sub>	E <sub>25</sub>	E <sub>26</sub>	E <sub>27</sub>	E <sub>28</sub>	E <sub>29</sub>	E <sub>30</sub>	E <sub>31</sub>	E <sub>32</sub>
b <sub>3</sub>	D <sub>1</sub>	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>	D <sub>7</sub>	D <sub>8</sub>	D <sub>9</sub>	D <sub>10</sub>	D <sub>11</sub>	D <sub>12</sub>	D <sub>13</sub>	D <sub>14</sub>	D <sub>15</sub>	D <sub>16</sub>	D <sub>17</sub>	D <sub>18</sub>	D <sub>19</sub>	D <sub>20</sub>	D <sub>21</sub>	D <sub>22</sub>	D <sub>23</sub>	D <sub>24</sub>	D <sub>25</sub>	D <sub>26</sub>	D <sub>27</sub>	D <sub>28</sub>	D <sub>29</sub>	D <sub>30</sub>	D <sub>31</sub>	D <sub>32</sub>
b <sub>2</sub>	C <sub>1</sub>	C <sub>2</sub>	C <sub>3</sub>	C <sub>4</sub>	C <sub>5</sub>	C <sub>6</sub>	C <sub>7</sub>	C <sub>8</sub>	C <sub>9</sub>	C <sub>10</sub>	C <sub>11</sub>	C <sub>12</sub>	C <sub>13</sub>	C <sub>14</sub>	C <sub>15</sub>	C <sub>16</sub>	C <sub>17</sub>	C <sub>18</sub>	C <sub>19</sub>	C <sub>20</sub>	C <sub>21</sub>	C <sub>22</sub>	C <sub>23</sub>	C <sub>24</sub>	C <sub>25</sub>	C <sub>26</sub>	C <sub>27</sub>	C <sub>28</sub>	C <sub>29</sub>	C <sub>30</sub>	C <sub>31</sub>	C <sub>32</sub>
b <sub>1</sub>	B <sub>1</sub>	B <sub>2</sub>	B <sub>3</sub>	B <sub>4</sub>	B <sub>5</sub>	B <sub>6</sub>	B <sub>7</sub>	B <sub>8</sub>	B <sub>9</sub>	B <sub>10</sub>	B <sub>11</sub>	B <sub>12</sub>	B <sub>13</sub>	B <sub>14</sub>	B <sub>15</sub>	B <sub>16</sub>	B <sub>17</sub>	B <sub>18</sub>	B <sub>19</sub>	B <sub>20</sub>	B <sub>21</sub>	B <sub>22</sub>	B <sub>23</sub>	B <sub>24</sub>	B <sub>25</sub>	B <sub>26</sub>	B <sub>27</sub>	B <sub>28</sub>	B <sub>29</sub>	B <sub>30</sub>	B <sub>31</sub>	B <sub>32</sub>
b <sub>0</sub>	A <sub>1</sub>	A <sub>2</sub>	A <sub>3</sub>	A <sub>4</sub>	A <sub>5</sub>	A <sub>6</sub>	A <sub>7</sub>	A <sub>8</sub>	A <sub>9</sub>	A <sub>10</sub>	A <sub>11</sub>	A <sub>12</sub>	A <sub>13</sub>	A <sub>14</sub>	A <sub>15</sub>	A <sub>16</sub>	A <sub>17</sub>	A <sub>18</sub>	A <sub>19</sub>	A <sub>20</sub>	A <sub>21</sub>	A <sub>22</sub>	A <sub>23</sub>	A <sub>24</sub>	A <sub>25</sub>	A <sub>26</sub>	A <sub>27</sub>	A <sub>28</sub>	A <sub>29</sub>	A <sub>30</sub>	A <sub>31</sub>	A <sub>32</sub>

(表 4：显示寄存器 bit7-扩展数码管段)

### 显示寄存器直接写入指令 *DIRECT\_WT* (0x00 - 0x1F)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
0	0	0	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

指令中 A<sub>4</sub>:A<sub>0</sub>为显示寄存器的地址，而数据部分 D<sub>7</sub>:D<sub>0</sub>将直接写入显示寄存器。在使用数码管显示器时，如果需要显示一些特殊字形，比如“H”，“L”，或负号“-”等，将用到此直接写入显示寄存器的指令；在用在点阵显示时，此指令可以直接将点阵数据写入目标显示寄存器。

### 段寻址指令 *SEG\_OFF*, *SEG\_ON* (0xC0, 0xC1)

段熄灭(写 0)指令 0xC0:

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	0	0	0	0	0	0	A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>

段点亮(写 1)指令 0xC1:

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	0	0	0	0	0	1	A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>

BC7591 将显示寄存器中的每一位，赋予了一个段地址，地址范围为 0x00-0xFF，通过段寻址指令，可以单独控制某一个显示段的点亮和熄灭。指令第一个字节为指令本身 C0 或 C1，第二个字节为段地址。显示寄存器中各位的段地址如下表：

显示寄存器地址									
	0x00	0x01	0x02	0x03	...	0x1C	0x1D	0x1E	0x1F
b <sub>7</sub>	0x07	0x0F	0x17	0x1F	...	0xE7	0xEF	0xF7	0xFF
b <sub>6</sub>	0x06	0x0E	0x16	0x1E		0xE6	0xEE	0xF6	0xFE
b <sub>5</sub>	0x05	0x0D	0x15	0x1D		0xE5	0xED	0xF5	0xFD
b <sub>4</sub>	0x04	0x0C	0x14	0x1C		0xE4	0xEC	0xF4	0xFC
b <sub>3</sub>	0x03	0x0B	0x13	0x1B		0xE3	0xEB	0xF3	0xFB
b <sub>2</sub>	0x02	0x0A	0x12	0x1A		0xE2	0xEA	0xF2	0xFA
b <sub>1</sub>	0x01	0x09	0x10	0x19		0xE1	0xE9	0xF0	0xF9
b <sub>0</sub>	0x00	0x08	0x10	0x18		0xE0	0xE8	0xF0	0xF8

(表 5: 显示寄存器-段地址)

### 全局写入指令 *WRITE\_ALL (0xF1)*

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	1	1	0	0	0	1	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

全局写入指令将数据写入到全部所有显示寄存器中，可以用来将所有显示段点亮或清除。指令第一个字节为指令本身 0xF1，第二个字节为写入显示寄存器的数据。

### 数码管显示相关指令

#### 译码显示 *DECODE\_WT (0x80 – 0x9F)*

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	0	0	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	-	-	-	-	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

在使用数码管显示时，译码显示指令将接收到的数据完成 7 段译码，并显示在相应的显示位上。指令字节中，A<sub>4</sub>:A<sub>0</sub>为数码管的位地址，也即显示寄存器的地址，数据字节的低 4 位，为待显示的数据，范围 0-F，高 4 位将被忽略。数据和显示字形对应如下：

D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	D <sub>3</sub> :D <sub>0</sub> (16 进制值)	显示
0	0	0	0	0	

D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>	D <sub>3</sub> :D <sub>0</sub> (16 进制值)	显示
0	0	0	1	1	1
0	0	1	0	2	2
0	0	1	1	3	3
0	1	0	0	4	4
0	1	0	1	5	5
0	1	1	0	6	6
0	1	1	1	7	7
1	0	0	0	8	8
1	0	0	1	9	9
1	0	1	0	A	A
1	0	1	1	B	b
1	1	0	0	C	C
1	1	0	1	D	d
1	1	1	0	E	E
1	1	1	1	F	F

(表 6: 数码管译码字形)

需要注意的是, BC7591 采用 7 段译码的方式, 即只有显示寄存器的低 7 位数据会受到译码显示指令的影响, 而最高位(DP 位)将保持不变。也就是说, 当数码管采用了 8 段的连接方式(如图 1 数码管 8 位连接), 每个数码管的小数点 DP 段都由相应移位寄存器的最高位 Q<sub>7</sub> 来驱动时, 使用译码指令刷新显示数据时, 小数点将不受影响, 维持原来的点亮或熄灭的状态。

译码显示规则有一个特例, 当数据字节为 0x80 时, 按译码表的显示规则应该显示“0”, 但实际 BC7591 在这种情况下不显示任何内容, 小数点位依然遵从 7 段译码规则不受影响。利用这个规则可以用来消除显示数字的最高位的“0”, 使得显示更自然。比如一个三位数字显示, 当显示的数值小于 100

时，例如“56”，如果不加额外处理，显示内容可能成为“056”，不符合人的阅读习惯，需要在程序中增加额外的逻辑判断，让最高位在数值小于100时不予显示。而利用BC7591这个特性，只要固定把写入百位的数值的最高位置1，即可自动实现当百位为0时不显示的功能。

利用这个译码规则的特例，还可以完成清除数码管显示，但不影响小数点状态的功能（如小数点用来做单独的指示灯或用作额外显示位时，此功能较为有用）。

### 扩展位不译码显示 *WRITE\_EXT(0xA8 – 0xAB)*

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	0	1	0	1	0	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

当数码管使用7段连接方式，DP段用作扩展显示位时，此指令用作将数据直接写入扩展的显示位。32个数码管显示位，每8位一组，共可扩展4个数码管显示位，赋予地址0-3。可以理解为：连接第一片HC164的扩展显示位，扩展地址为0，第二位地址为1...至连接到第三片HC164的扩展数码管位扩展地址为3。指令中，A<sub>1</sub>:A<sub>0</sub>为扩展显示位地址，D<sub>7</sub>-D<sub>0</sub>分别写入扩展显示位对应的A-DP段，用此指令可以达到在扩展显示位显示特殊字符的目的。

从寄存器角度，此指令的效果是将数据D<sub>7</sub>:D<sub>0</sub>写入到显示寄存器(Addr\*8+b)的最高位，b的范围是0-7。此指令和后面“顶部¼行写入指令QTR\_WT\_TOP”完全相同，是同一命令的不同叫法。

### 扩展位译码显示 *DECODE\_EXT (0xB0 – 0xB3)*

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	0	1	1	0	0	A <sub>1</sub>	A <sub>0</sub>	-	-	-	-	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

此指令和上条指令类似，区别在于此指令完成译码显示功能。

指令中，最低2位为扩展显示位的地址，范围0-3，而数据字节中，同译码指令相同，低4位为待显示数据，高4位被忽略。

从显示寄存器的角度，扩展位译码指令的作用是将译码后的结果，按位依次写入显示寄存器(Addr\*8+b)的最高位，其中Addr代表扩展显示位地址，b的范围为0-6。扩展位的译码，依然是采用7段译码方式，即不包括小数点，因此显示寄存器地址0x07, 0x0F, 0x17, 0x1F的四个显示寄存器的最高位不会受到此指令的影响。

扩展位译码指令，也如同上面译码显示指令一样，具有当数字字节为0x80时清除显示内容的功能。

## 点阵显示相关指令

### 列写入指令 *COL\_WRITE* (0x00 - 0x1F)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
0	0	0	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

列写入指令，即为上面提到的显示寄存器直接写入指令，是同一个指令的另一名称，用于点阵显示时，列写入的名称更利于理解其作用。LED 点阵与显示寄存器内容的对应关系为：每个显示寄存器对应矩阵中的一个列，显示寄存器的地址，即为列地址。请参见前文中点阵显示与显示寄存器的对应关系。

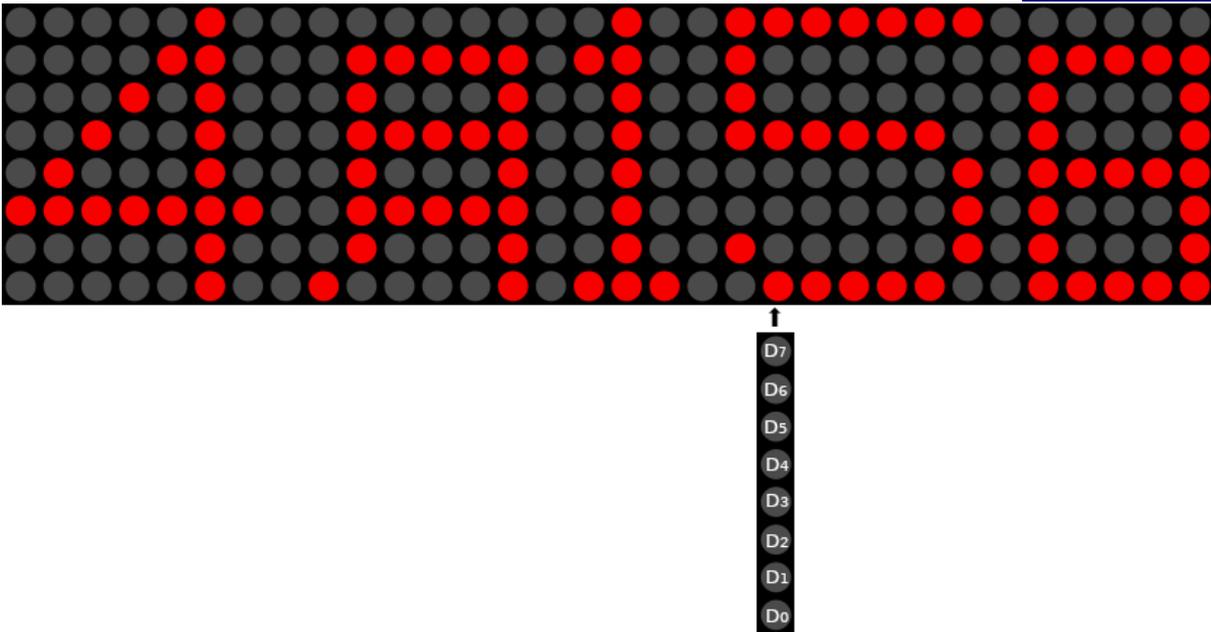
除了直接写入指令，BC7591 还有专门针对点阵显示特点的特殊指令，这些指令包括：

### 列插入并向高地址(右)平移指令 *SHIFT\_H\_WT* (0x40 - 0x5E)

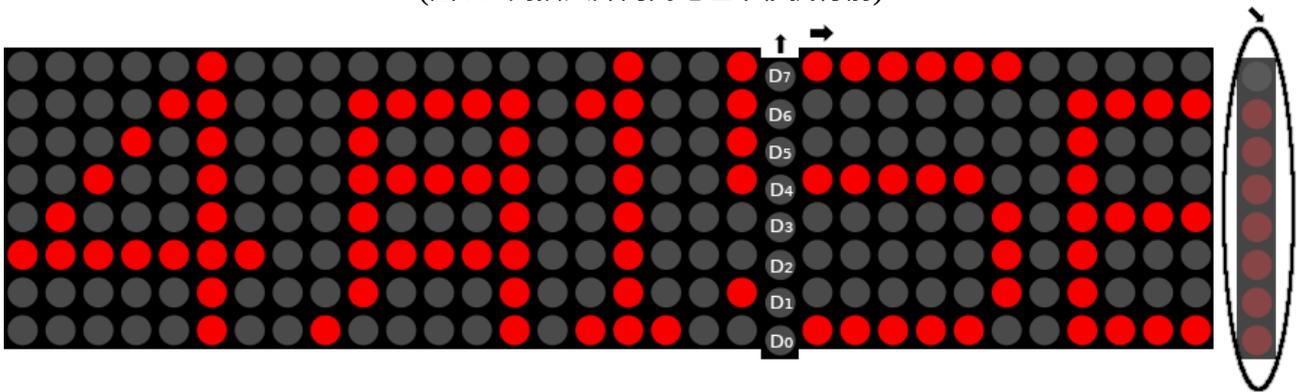
第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
0	1	0	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

指令中 A<sub>4</sub>:A<sub>0</sub> 为显示寄存器地址，范围为 0x00-0x1E。数据字节为待写入的数据。该指令与直接写入指令相同的部分在于，都是将数据写入指定的寄存器。但该指令并不覆盖原显示寄存器中的内容，而是将数据插入到指定寄存器，原寄存器和所有地址高于目标寄存器的显示寄存器中的内容，均向高地址方向平行移动。即地址 n 的寄存器内容，被移动到地址 n+1 的寄存器，而最高地址的显示寄存器的内容，则被抛弃。如果点阵显示是按照示意图中的从左至右地址从低到高排列，则此指令的效果就是，将数据字节的数据插入到目标列，而目标列右侧的显示内容，向右平移一个像素。

假设现在执行命令地址 A<sub>4</sub>:A<sub>0</sub> = 0x14，则此命令执行前后的效果如下图：



(图 7: 列插入并向高地址平移执行前)



丢弃

(图 8: 列插入并向高地址平移执行后)

### 列插入并向低地址(左)平移指令 *SHIFT\_L\_WT* (0x61 – 0x7F)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
0	1	1	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

此指令与上一指令作用类似，但移动的方向相反。地址 A<sub>4</sub>:A<sub>0</sub>的取值范围为 0x01-0x1F.

### 向右(高地址)循环滚屏 *ROTATE\_R*(0x5F)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
0	1	0	1	1	1	1	1	-	-	-	-	-	-	-	-

点阵显示的内容从左向右(显示寄存器低地址向高地址)循环滚动一个像素。第二个字节必须有但可为任意内容，对指令执行无影响。

## 向左(低地址)循环滚屏 ROTATE\_L(0x60)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
0	1	1	0	0	0	0	0	-	-	-	-	-	-	-	-

点阵显示的内容从右向左(显示寄存器高地址向低地址)方向循环滚动一个像素。第二个字节必须有但可为任意内容,对指令执行无影响。

## 底部 1/4 行写入指令 QTR\_WT\_BOT (0xA0 – 0xA3)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	0	1	0	0	0	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

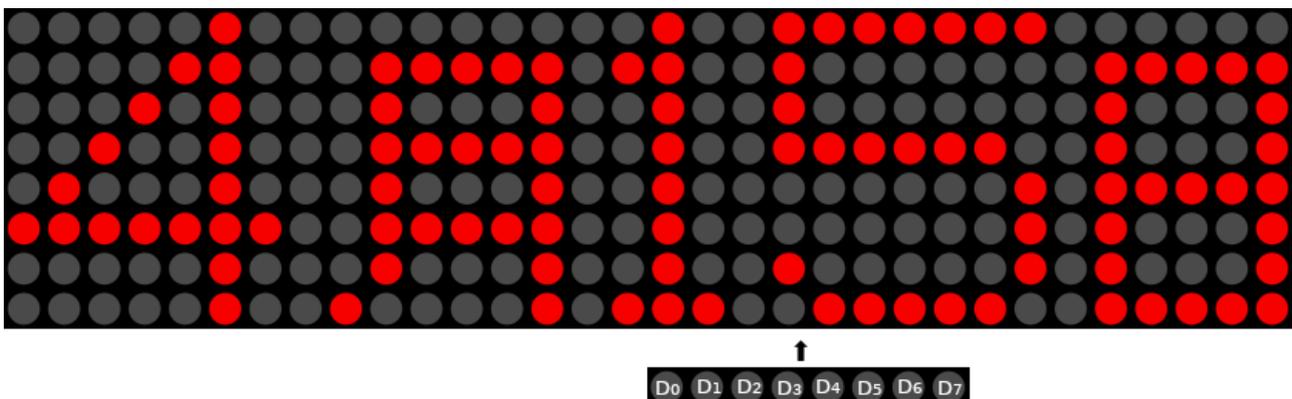
所有的显示段形成一个 32 列 8 行的矩阵,每个显示寄存器对应一个显示列,按列写入是最自然的操作方式,但某些情况下可能需要按行进行更新,矩阵每行包括了 32 个点,而每个指令只能包括一个字节的的数据,因此每个指令只能写入 1/4 行。此指令将 1/4 行的数据写入到显示矩阵的底部。要想完成整个行的内容更新,需要将此指令执行 4 次。

一行中的 32 个显示点,被分为 4 组,每组 8 个,并赋予地址 0-3. 列 0-7 的组地址为 0, 列 8-15 的组地址为 1, 依次类推, 列 24-31 的组地址为 3. 指令中, A<sub>1</sub>:A<sub>0</sub> 表示组地址。

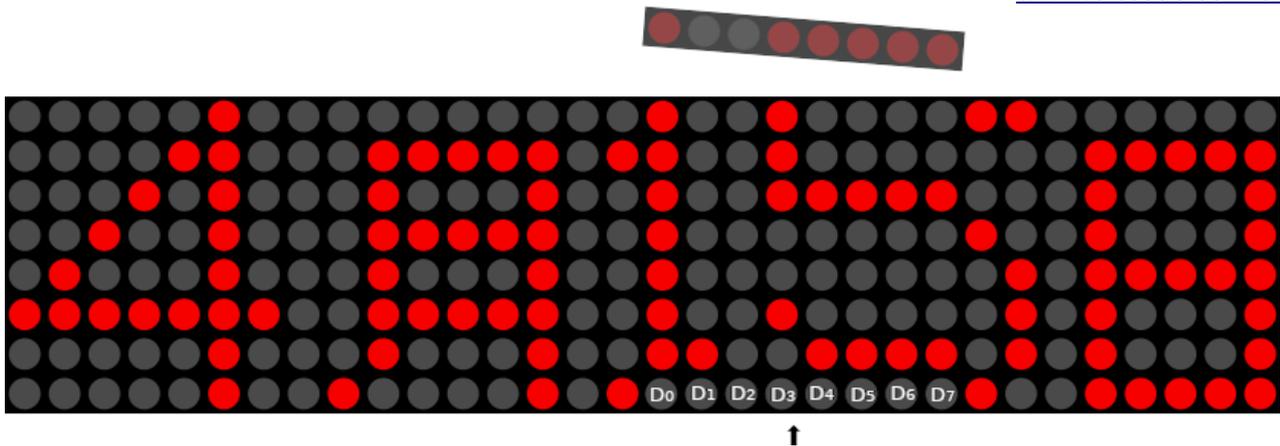
## 底部 1/4 行插入指令 QTR\_INS\_BOT (0xA04– 0xA7)

写入方式和上面 1/4 行写入指令相同,差别在于此指令为插入模式,对应组中的各列显示,都会向上平移一个像素,最顶部的像素被丢弃,而新数据写入最下一行。

假设执行命令时 A<sub>1</sub>:A<sub>0</sub> = 0x02, 则命令执行前后的效果示意如下:



(图 9: 底部 1/4 行插入, 执行前)



(图 10: 底部 ¼ 行插入, 执行后)

从寄存器的角度, 此命令的作用, 是将数据字节的各个位, 依次写入到对应寄存器组各字节的  $b_0$  位, 并将原寄存器内的数据执行左移操作。

### 顶部 ¼ 行写入指令 $QTR\_WT\_TOP$ (0xA8 – 0xAB)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	0	1	0	1	0	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

此指令与上面底部 ¼ 行写入指令类似, 区别在于写入的目标是显示矩阵的最顶行, 即显示寄存器的  $b_7$  位。

此指令的另一个作用是在使用扩展显示位的数码管电路中, 向扩展的显示位直接写入段映射数据, 从而可以显示比如 “H”, “L” 等特殊字形。

### 顶部 ¼ 行插入指令 $QTR\_INS\_TOP$ (0xAC – 0xAF)

此指令与上面 “顶部 ¼ 行写入” 指令相同, 区别在于此指令为插入模式, 被写入的 8 列的原显示内容, 会向下移动一个像素。从显示寄存器的角度, 此命令执行的操作为把 8 个目标显示寄存器的内容先执行右移, 然后把指令的数据分别写入各寄存器的 bit7。

### 坐标点写入指令 $COORD\_OFF, COORD\_ON$ (0xC0, 0xC1)

点熄灭(写 0)指令 0xC0:

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	0	0	0	0	0	0	X <sub>4</sub>	X <sub>3</sub>	X <sub>2</sub>	X <sub>1</sub>	X <sub>0</sub>	Y <sub>2</sub>	Y <sub>1</sub>	Y <sub>0</sub>

点点亮(写 1)指令 0xC1:

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	0	0	0	0	0	1	X <sub>4</sub>	X <sub>3</sub>	X <sub>2</sub>	X <sub>1</sub>	X <sub>0</sub>	Y <sub>2</sub>	Y <sub>1</sub>	Y <sub>0</sub>

坐标点写入指令，是段寻址指令的别名。当把所有显示段理解为一个 32x8 的矩阵，每个显示点可以赋予一个行、列坐标，行坐标 X 的范围为 0-31，而列坐标 Y 的范围为 0-7。实际上每个显示点(显示段)都有一个段地址，段地址的 d<sub>7</sub>:d<sub>3</sub> 位，可以理解为该点的 X 坐标，而 d<sub>2</sub>:d<sub>0</sub> 位，可以理解为 Y 坐标。

## 控制指令

### 段(点)闪烁控制位置 1 指令 *BLINK\_WT\_SET (0x30 – 0x3F)*

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
0	0	1	1	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

BC7591 支持单独显示段(点)的闪烁功能，256 个显示段中的前 128 个，即前 16 个显示寄存器中的每一位，均可以单独控制闪烁功能。指令中，A<sub>3</sub>:A<sub>0</sub> 是显示寄存器地址，而数据字节中 D<sub>7</sub>:D<sub>0</sub> 对应目标显示寄存器中的位(显示段)，如果此命令中对应的 D<sub>n</sub> 位为“1”，该显示段就被赋予闪烁属性；数据字节中为“0”的位，将对闪烁属性无影响，即不改变该位的闪烁属性，如果该位原闪烁属性为 1，则执行指令后仍将保持为 1。当全局的闪烁控制位为开(上电后默认状态)，则闪烁属性为 1 的显示段就会以闪烁的方式显示，闪烁的频率，由闪烁频率控制指令控制。

闪烁控制，芯片内部由专门的闪烁控制寄存器完成，与显示寄存器的内容无关。显示寄存器的内容更新、清零等操作，均不影响其闪烁属性。如果一个显示段被设置了闪烁属性，当显示寄存器清零后，用户将看到该显示段为熄灭状态，而当该显示寄存器被重新写入内容后，该段仍将以闪烁状态显示。闪烁属性只能由闪烁控制指令改变。

上电后，所有的闪烁控制属性，都为“0”，即不闪烁的状态。

### 段(点)闪烁控制位置 0 指令 *BLINK\_WT\_CLR (0x20 – 0x2F)*

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
0	0	1	0	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

此指令作用与上面指令类似，不过作用是清除对应显示段的闪烁属性。数据字节中为“1”的位对应的显示段，其闪烁属性将被清除，而为“0”的位，对应显示段的闪烁属性将不被改变。

### 位闪烁控制指令 *BLINK\_DIG\_CTL (0xD0, 0xD1)*

第 16-23 位(显示寄存器地址 0x10-0x17)闪烁控制 0xD0:

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	0	1	0	0	0	0	D <sub>17</sub>	D <sub>16</sub>	D <sub>15</sub>	D <sub>14</sub>	D <sub>13</sub>	D <sub>12</sub>	D <sub>11</sub>	D <sub>10</sub>

第 24-31 位(显示寄存器地址 0x18-0x1F)闪烁控制 0xD1:

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	0	1	0	0	0	1	D <sub>1F</sub>	D <sub>1E</sub>	D <sub>1D</sub>	D <sub>1C</sub>	D <sub>1B</sub>	D <sub>1A</sub>	D <sub>19</sub>	D <sub>18</sub>

BC7591 的显示寄存器 0x10-0x1F 所对应的显示内容, 不支持单独段的闪烁控制, 只能按显示寄存器, 即数码管的显示位来控制闪烁属性。在指令的数据字节中, D<sub>10</sub>-D<sub>1F</sub> 分别对应显示寄存器 0x10-0x1F, 如果指令中的相应数据位为 1, 该显示寄存器的内容, 就将以闪烁的方式显示, 而为“0”的位, 将不闪烁。

上电后, 所有 0x10-0x1F 的显示寄存器, 闪烁属性都将处于清零的状态。

### 闪烁速度控制指令 *BLINK\_SPEED* (0xF2)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	1	1	0	0	1	0	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

BC7591 的闪烁功能, 其闪烁速度可以由闪烁速度控制指令控制。指令中数据字节的数据, 用来控制闪烁速度, 闪烁的速度, 可以大致由以下公式计算:

$$F_{\text{blink}} = 92/(2*S)$$

公式中, F<sub>blink</sub> 为闪烁的频率, 单位为 Hz, S 为闪烁速度控制值, 即指令中的数据字节的值。上电后, 芯片内部的 S 的缺省值为 50, 即闪烁频率约为 0.92Hz。

### 亮度控制指令 *DIM\_CTL* (0xF3)

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	1	1	0	0	1	1	-	-	-	-	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>

BC7591 具有 16 级线性亮度控制。亮度的控制通过调整显示扫描输出的占空比来实现。因为人眼对亮度的感知特性, 对亮度的感觉并非线性, 对微弱的亮度, 人眼可以感知细小的变化, 而对比较亮的光源, 则对亮度的变化不敏感。BC7591 采用了非线性的占空比调整, 目的是尽量使得人眼感觉到亮度的变化速度比较均匀, 避免了在高亮度时调整变化细微, 而低亮度时感觉每级亮度差别过大的现象。

指令中数据字节的低 4 位为亮度值, 为“0”时, 亮度最大, 而为“0x0F”时, 亮度最低。高 4 位将被忽略。上电后, 默认的亮度值为 0, 即亮度最大的状态。

**整体控制指令 GLOBAL\_CTL (0xF0)**

第 1 字节(指令)								第 2 字节(数据)								
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>		d <sub>0</sub>
1	1	1	1	0	0	0	0	r	r	r	r	r	r	DISP_OFF		BLINK_OFF

整体控制指令控制 BC7591 的两个全局属性：闪烁的开/关，以及整体显示的开/关。分别由指令数据字节的 b<sub>0</sub> 位和 b<sub>1</sub> 位控制，其它数据位，为保留位，应该置为 0。

b<sub>0</sub> 位 BLINK\_OFF：该位置 1 时，芯片的闪烁功能将关闭，无论各显示段和显示寄存器的闪烁属性设置为何值，均不会以闪烁方式显示。不过设置的闪烁属性都将保留，当 BLINK\_OFF 被重新清 0 时，各闪烁属性为闪烁的显示段和显示位，都将恢复闪烁显示。当 BLINK\_OFF 为 1 时，仍可通过闪烁控制指令设置显示段和显示寄存器的闪烁属性。

b<sub>1</sub> 位 DISP\_OFF：该位置 1 时，将关闭芯片的显示输出。不过显示寄存器和闪烁控制的内容，均将被保留。同时当显示输出关闭时，仍可以通过各指令更新显示寄存器和各控制寄存器的内容。关闭芯片的显示输出，不会影响键盘扫描的工作。

**复位指令 RESET (0xFF5A)**

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	1	1	1	1	1	1	0	1	0	1	1	0	1	0

该指令将使得 BC7591 芯片复位，芯片状态恢复到上电时的初始值。如果键盘部分有按键处于按下导通的状态，执行此命令后，芯片将输出所有处于导通状态的按键的键值。详情请参见后面键盘部分。

**UART 发送 0 UART\_SEND\_0(0xFFFF)**

第 1 字节(指令)								第 2 字节(数据)							
d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>	d <sub>7</sub>	d <sub>6</sub>	d <sub>5</sub>	d <sub>4</sub>	d <sub>3</sub>	d <sub>2</sub>	d <sub>1</sub>	d <sub>0</sub>
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

该指令使得 BC7591 的 UART 上输出单个字节 0x00。一般用于测量 BC7591 的波特率，以微调 MCU 的波特率，使二者精确同步。通常用于 MCU 波特率会存在误差时。详见后文《波特率误差》一节。

## BC7591 指令汇总

指令值	指令名称	作用
0x00 - 0x1F	DIRECT_WT COL_WRITE	直接写入显示寄存器 / 点阵显示列
0x20 - 0x2F	BLINK_WT_CLR	清除段闪烁属性。数据字节中为 1 的位对应的显示段的闪烁属性将被清除
0x30 - 0x3F	BLINK_WT_SET	设置段闪烁属性。数据字节中为 1 的位对应的显示段将被设置为的闪烁属性
0x40 - 0x5E	SHIFT_H_WT	矩阵显示列插入，现有显示向高地址侧(右)平移
0x5F	ROTATE_R	矩阵显示向右循环滚动一个像素
0x60	ROTATE_L	矩阵显示向左循环滚动一个像素
0x61 - 0x7F	SHIFT_L_WT	矩阵显示列插入，现有显示向低地址侧(左)平移
0x80 - 0x9F	DECODE_WT	数码管译码写入，将数据字节内数值在数码管显示
0xA0 - 0xA3	QTR_WT_BOT	矩阵显示写入 ¼ 行到矩阵底部
0xA04- 0xA7	QTR_INS_BOT	矩阵显示插入 ¼ 行到矩阵底部
0xA8 - 0xAB	QTR_WT_TOP WRITE_EXT	矩阵显示写入 ¼ 行到矩阵顶部(数码管扩展位不译码写入)
0xAC- 0xAF	QTR_INS_TOP	矩阵显示插入 ¼ 行到矩阵顶部
0xB0 - 0xB3	DECODE_EXT	数码管扩展显示位译码写入
0xC0	SEG_OFF COORD_OFF	显示段熄灭 / 坐标点熄灭
0xC1	SEG_ON COORD_ON	显示段点亮 / 坐标点点亮
0xD0 - 0xD1	BLINK_DIG_CTL	数码管位闪烁属性控制，显示寄存器闪烁属性控制
0xF0	GLOBAL_CTL	整体控制，包括闪烁功能开/关，显示功能开/关
0xF1	WRITE_ALL	全局写入，将数据字节写入全部显示寄存器
0xF2	BLINK_SPEED	闪烁速度控制
0xF3	DIM_CTL	显示亮度控制
0xFF5A	RESET	芯片复位指令
0xFFFF	UART_SEND_0	在 UART 接口输出 0x00

(表 7: BC7591 指令汇总)

此表列举了 BC7591 所接受的所有指令，如果在指令字节收到了超出本表格以外的数值，BC7591 的反应将是“未定义”，即不能确定 BC7591 会有何种反应。

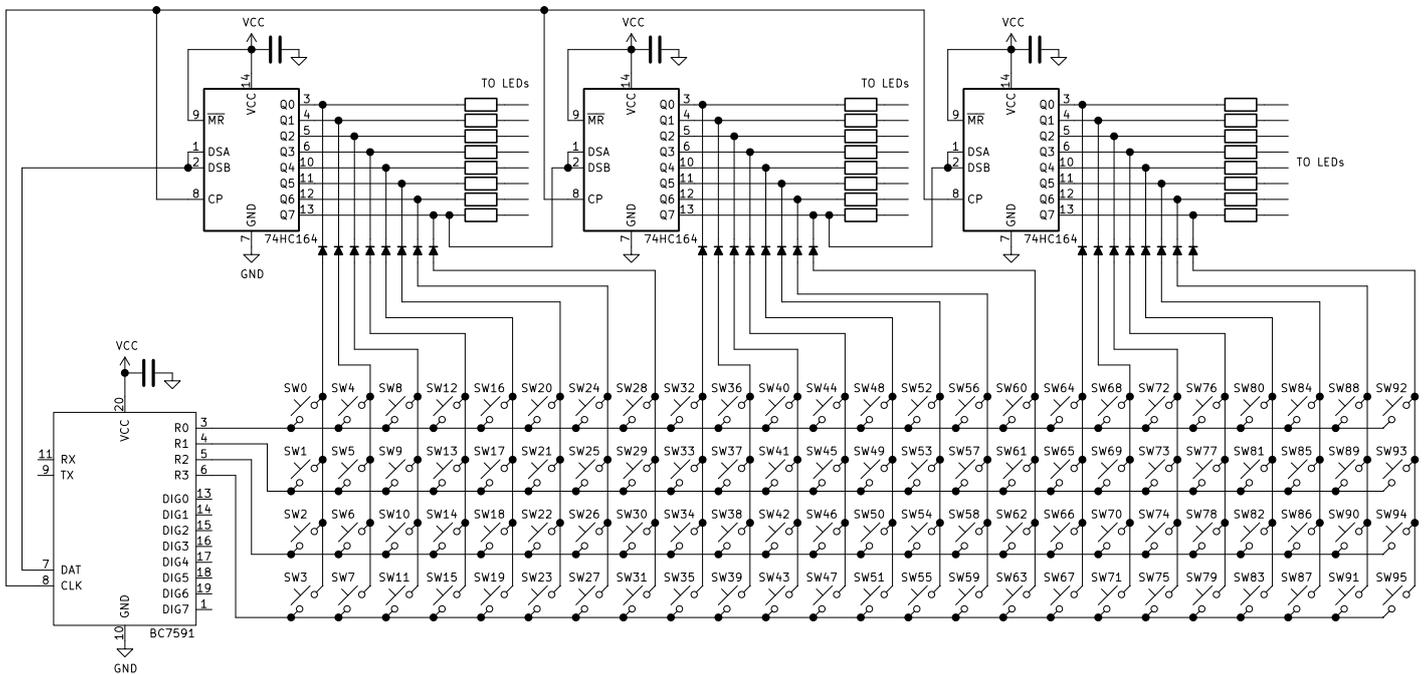
在 BC7591 上电后，将有一约 300ms 的等待时间，以确保 MCU 完成上电过程并可接收串口数据(键盘数据)，在此期间，BC7591 也无法接收指令数据，MCU 对 BC7591 的操作，须在上电 300ms 后。

# 键盘接口

## 接口电路

BC7591 除提供 LED 显示驱动外，还可提供键盘矩阵接口。BC7591 芯片提供 4 个键盘行扫描接口 R0-R3，键盘矩阵的列连接到移位寄存器的输出。根据外接移位寄存器的数量，可以配置为从 32 键 (4x8)到 96 键(4x24)的键盘接口。BC7591 的键盘矩阵，可以支持任意形式的组合键，并可支持任意时间长度的长按键，可以支持常开和常闭型按键，以及两种按键的混合使用。

下图是连接为 96 键键盘接口的完整电路图(不含 LED 显示部分)。图中的按键元件标号，即对应按键的键值，如 SW0 的键值为 0、SW25 的键值为 25(0x19)。



(图 11: 键盘接口电路)

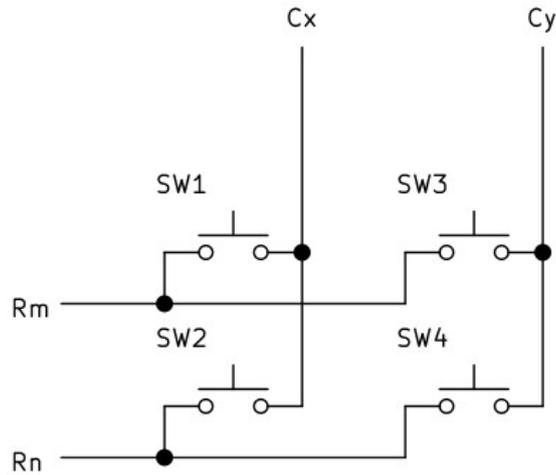
因为键盘扫描电路和显示扫描电路是部分重合的，为了防止当键盘矩阵中同一行中有两个按键同时导通时造成显示驱动电路的短路，必须在键盘的列电路——即连接移位寄存器输出一侧，加入二极管。应该选用正向导通电压较低的二极管，如肖特基二极管等，以确保键盘扫描信号能被可靠接收。这点在电源电压较低时尤其重要，因为 BC7591 芯片的键盘行扫描能接受的最大低电平电压为 0.7V(电源电压 3.3V 时)，如果使用普通二极管，管压降也是 0.6-0.7V，将无法可靠保证按键能被扫描到。

如果键盘部分已经串入了防止影子按键的二极管(参见后文)，这些二极管同时可以起到防止移位寄存器输出短路的作用，故键盘列与移位寄存器间的二极管则可以省略。

BC7591 的键盘矩阵行扫描输出 R0-R3，已经带有内部的上拉电阻，等效阻值约 100K $\Omega$ ，一般使用时无需再另加外部上拉电阻。如果键盘分布面积很大，或者周围环境干扰较大，100K $\Omega$  的输入阻抗可能对减低干扰不利，可以另外再并入外接的上拉电阻，减低输入阻抗。不过此上拉电阻也不能过小，如果过小，当有键按下时将有显著的电流流过显示 LED，对显示造成干扰。

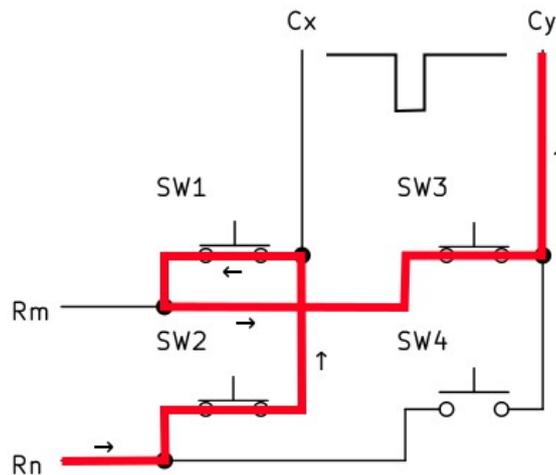
## 消除影子按键

BC7591 具有处理任意多组合按键的能力，但当组合按键处在四边形的四个角上时，需要注意防止影子按键。影子按键是矩形键盘矩阵的特有现象，表现为，当处在一个四边形四个角上的四只按键中有三只处于导通状态时，系统会误认为第四只按键也处于导通状态。如图中当 SW1,SW2,SW3 三个键都按下时，系统会误以为 SW4 也是按下的状态。



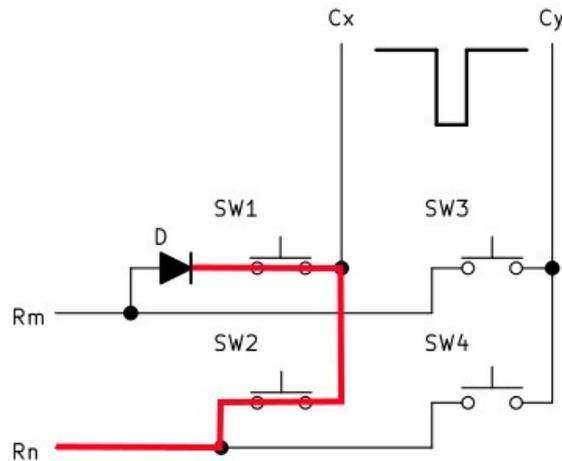
(图 12: 按键矩阵形成 4 边型)

产生这种现象的原因，在于当 SW1-SW3 三只按键均处于导通状态时，因为三只按键所形成的电流回路，实际上第四只按键 SW4 的两端，也是处于连通的状态。假设系统扫描到 SW4 时，Cy 输出低电平，则电流通过如下的回路，会使 Rn 为低电平，和 SW4 被按下效果相同。如图：



(图 13: 四边形三个角按键按下时第 4 键虚假导通)

解决这个问题的方法，是在没有按下的按键的对角线方位的按键 SW1 中，串入一只二极管。该二极管可以在 SW1-SW3 三只按键均导通时，阻止扫描第四只按键 SW4 的电流形成回路，因此系统也就不会认为第四只按键是导通状态了。如图：



(图 14: 二极管防止影子按键)

要彻底消除产生影子按键的可能，需要对所有按键均串入二极管，但这样成本较高，一般实际应用中，仅需对需要防止产生影子按键的个别键采取措施即可。因为二极管会产生正向管压降，从而降低了键盘扫描电路的可靠性，因此，应尽量选用正向导通管压降比较低的二极管，比如肖特基二极管等。

## 输出格式

BC7591 的键盘接口输出数据，与 BC6xxx 系列芯片协议兼容，代码可互换使用。当键盘状态发生变化时，如有按键按下（导通）或释放（断开），BC7591 即输出有变化按键的键值，键值即键值表中的数值。如果按键状态是由断开变为导通状态，则输出不加修改的键值；如果按键状态是由导通变为断开，则输出键值的最高位会被置为 ‘1’，如原键值为 0x01，则变为 0x81，如原值为 0x17，则变化为 0x97，以此类推。如果同时有多个按键状态产生变化，则将所有有变化的按键的值按扫描到的顺序依次输出。同一列中的按键，处在较低行的按键会先输出，而不同列中的按键，输出的顺序则不确定，取决于按键按下时所正在扫描的列号。

在系统上电时，BC7591 芯片默认所有的按键均为断开状态。如果在上电时，有按键已经处于导通状态，则 BC7591 在开始键盘扫描后，会输出所有已经处于导通状态的按键的键值。最极端的情况下，如果所采用的全部是常闭状态的按键，即上电时所有按键都是导通状态，则 BC7591 在上电后将连续输出所有按键的键值。

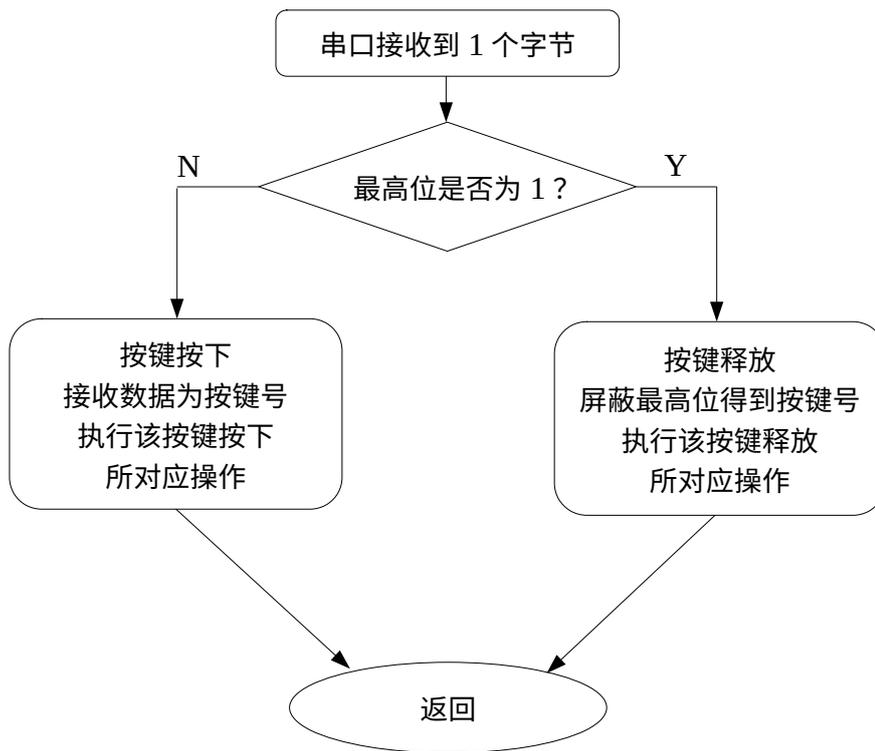
BC7591 在上电后，在开始扫描键盘前，有一约 300ms 的延时，以确保 MCU 完成上电复位过程并能正确接收 UART 数据。

## 键盘处理流程

单片机的硬件 UART 接口，一般均带有中断功能，应该优先使用中断方式处理按键数据，减轻 MCU 的负担。BC7591 直接输出键值数据，因此处理普通的单按键事件非常简单，仅需根据收到的键值，转去执行相应的按键操作即可。而且 BC7591 输出的键值可区分按键的按下和释放，用户可根据需要选用适用的按键事件。

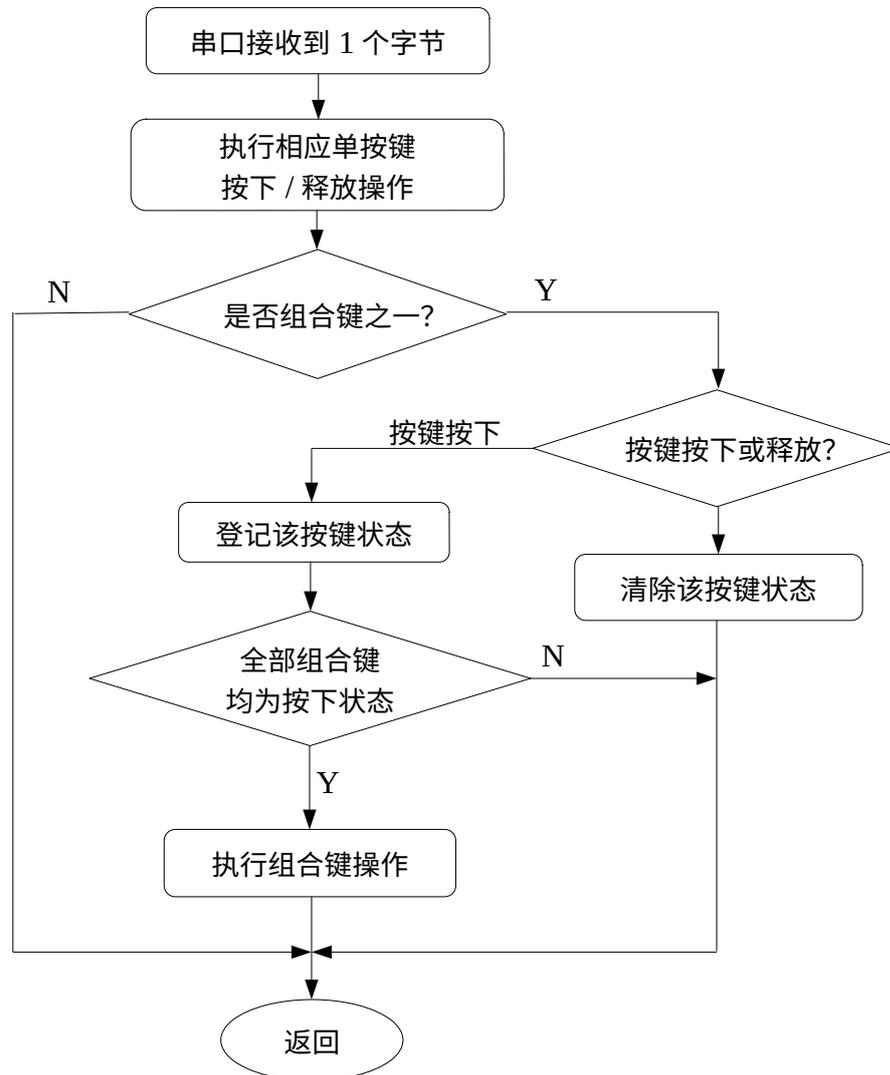
当有多个按键同时发生状态变化时，BC7591 会连续输出所有有变化按键的键值。BC7591 的 UART 波特率为 9600, 输出一个键值的时间大约为 1ms, 也即连续输出时, 数据间的间隔约为 1ms。为了防止按键信息的丢失, MCU 应该确保每个按键事件的处理时间都在 1ms 以内。如果不能保证在 1ms 的时间内处理完, 则需要给串口接收数据建立一个缓存, 防止来不及处理的键值被新数据覆盖。

### 普通单次按键处理流程



(图 15: 普通按键处理流程)

## 组合键



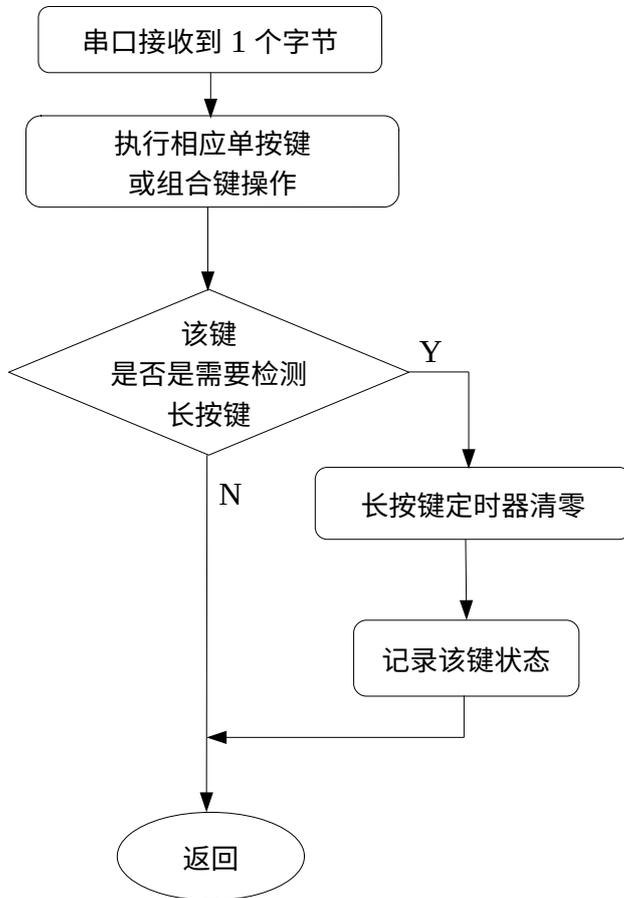
(图 16: 组合键处理流程)

组合键的使用要求程序能记忆按键组合中各键的状态。

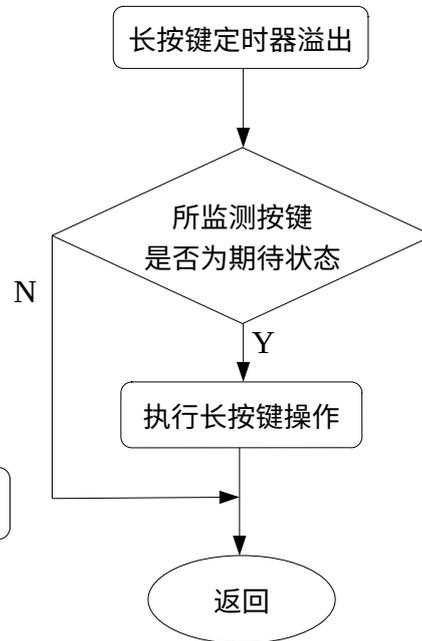
## 长按键

BC7591 在键盘状态发生变化时，即从 UART 接口发出相应信息，无论是有按键按下，或有按键释放。换言之，如果 UART 上没有数据输出，即表明键盘维持在上次数据输出时的状态没有改变。当一个键按下后，如果一段时间内没有收到该键状态改变的数据，就表示这段时间内该键一直处于按下状态。不光是单个按键的长按键检测，组合按键也可以用同样方法检测长按键。用户只需要提供一个计时器，用来确定需要检测的长按键的时长。

串口中断处理程序：



定时器中断处理程序：



(图 17: 长按键处理流程)

## UART 接口

### 接口参数

BC7591 采用 UART 接口，参数设置为：波特率 9600，8 个数据位，1 个起始位，1 个停止位，无奇偶校验。

### 检错机制

BC7591 采用两种方式防止通讯过程中可能出现的错误。首先，对 UART 所接收到的所有数据，当产生帧错误，即没有监测到正确的停止位时，该数据即被抛弃。

同时，BC7591 还采取了独有的时间-复位机制来检测和纠正通讯中出现的错误。BC7591 的指令，均为两个字节一组，第一个字节为指令，第二个字节为数据，如果因某种原因有字节丢失从而使二者产

生了错位，会造成不可预见的后果。BC7591 针对这种情况设计了时间-复位机制，当 UART 接收口线上没有数据超过一定时间后，UART 接收口就会复位。比如因某种原因造成指令字节丢失，此时接下来的数据字节会被作为指令接收，普通情况下，后面下一个指令中的指令字节，会被当做前一条指令的数据，并这样一直错位下去。BC7591 的时间-复位机制，让 UART 口在接收口线上无信号 2 个显示扫描周期后，就会自动复位，上面的情况中，只要 MCU 在两个指令之间插入大于两个扫描周期的时间间隔，第一个没有被完整接收的指令就会因超时而被抛弃，从而保证了第二个指令得以正确被接收和执行。

BC7591 的显示扫描频率约为 93Hz，扫描周期约 10.8ms，因此，只要指令间的间隔时间大于 21.6ms，就可以保证因字节丢失而造成的数据错位得以纠正。程序中也无需在每个指令之间都插入这个延时，因为有时也会需要连续快速地发送多个指令。一般来说，正常使用情况下，不会高频率地长时间不间断发送，正常的指令操作间隔，就足以满足此时间-复位机制的需要。

## 接口电路

### 电平匹配

BC7591 的 UART 输出口 TX，采用漏极开路输出(Open Drain)，可以方便地与不同电源电压的 MCU 接口，仅需在 MCU 接收端连接一上拉电阻至 MCU 的电源。很多的 MCU 和 UART 器件，接收线内部已经带有上拉电阻，这种情况下外部的上拉电阻亦可以省略。

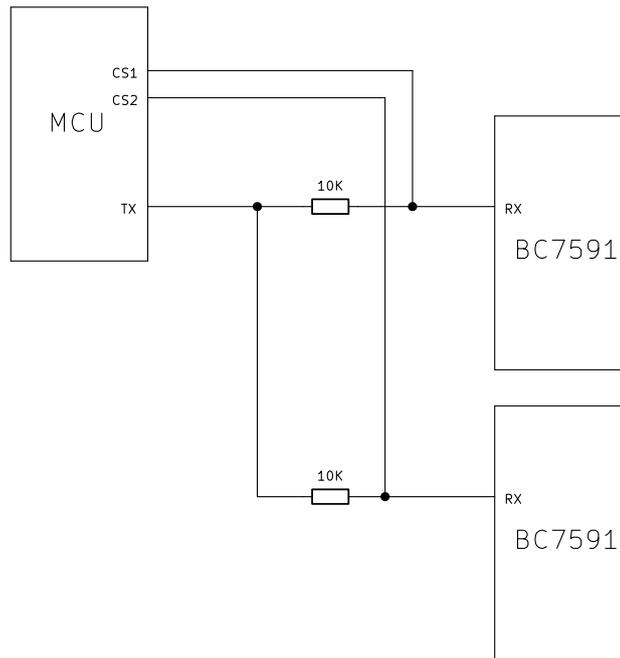
BC7591 的 UART 输入口 RX，内部带有上拉电阻，当 BC7591 和 MCU 两侧电源电压不同，且 MCU 的串口输出又为推挽大电流输出时，需要做特别的考虑。分为 MCU 电压大于和小于 BC7591 电压两种情况。当 MCU 电压大于 BC7591 的电压时，需要在 MCU 的 TX 和 BC7591 的 RX 引脚之中，串入一个电阻，以防止高电压直接加到 BC7591 的 RX 引脚上引起的引脚损坏。电阻的大小并无严格要求，一般选取 10K 可满足多数情况的需要。

当 MCU 的电压小于 BC7591 的电压，需要特别注意。BC7591 的 RX 引脚所能接受的最小高电平电压，为  $0.8V_{cc}$ ，当 BC7591 电源电压为 5V 时，要求 RX 的输入高电平为 4V，如果 MCU 电源电压低于 4V，必须使用电平转换电路。

### 多片联用

UART 接口主要用于 1 对 1 通讯的情况，不过有时系统中会需要用到多片 BC7591，这时，可以用一个简单的电路做为 UART 的“片选”信号，从而实现系统内多片 BC7591 的联合使用。因为 BC7591 芯片上 LED 显示指令均为单向指令，MCU 无需与 BC7591 往复通讯，而 BC7591 的键盘部分亦为单向通讯，且一个系统内一般一片 BC7591 所提供的按键数量已经足够，因此多片联用，仅需考虑 MCU 到 BC7591 的发送信号线即可。

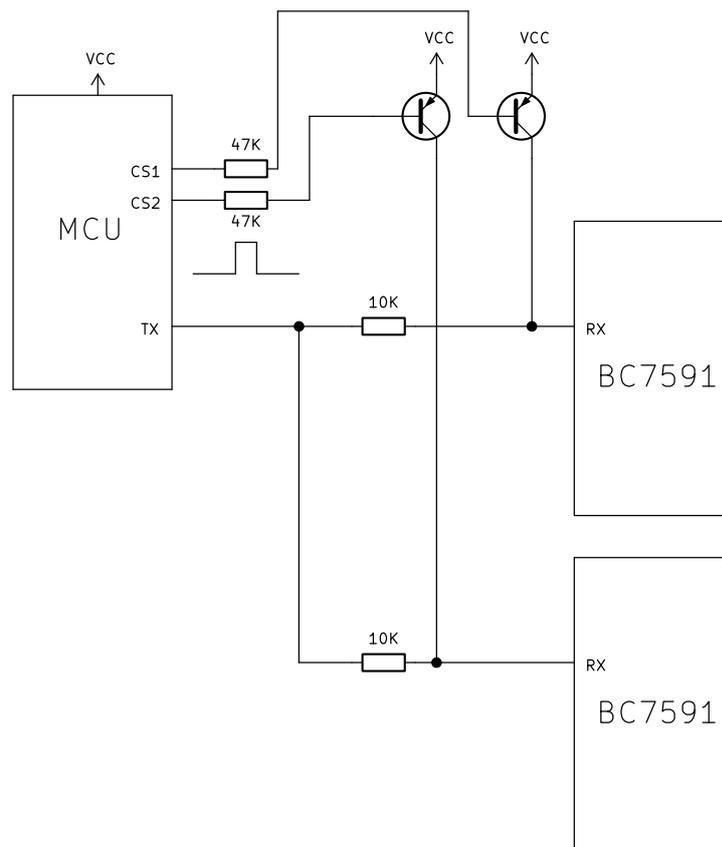
如果所使用的 MCU 的 I/O 支持输出和高阻输入状态的转换，可以仅靠增加每片 1 只电阻，完成 UART 接口的“片选”功能。如下图：



(图 18: UART 使用 I/O 口完成片选)

对不选中的 BC7591 芯片，相应的片选 I/O 口设置为输出状态，并输出高电平。这样不管 MCU 的 TX 输出如何变化，该片 BC7591 的 UART 输入 RX 始终被锁定为高电平，不接收任何数据。而对于需要选中的 BC7591 芯片，相应片选 I/O 口设置为高阻状态，此时 BC7591 的 RX 引脚上电平将通过电阻 R 随 MCU 的 TX 线上电平变化，数据可被 BC7591 接收。

如果 MCU 不支持 I/O 口的高阻状态，则电路会复杂一些，需要使用一个三极管，电路如图：



(图 19: UART 使用三极管完成片选)

当片选线输出低电平时，三极管导通，BC7591 RX 引脚被钳位在高电平，不接收任何数据；当片选 I/O 口输出高电平时，三极管截止，BC7591 被选中，RX 引脚电平随 MCU 的 TX 引脚变化，可接收数据。

## 波特率误差

BC7591 使用内部经校准的 RC 振荡器产生波特率，在全工作温度和电压范围内，保证频率误差在  $\pm 1.5\%$  之内。UART 通讯要求双方的波特率相对误差，必须在  $\pm 5\%$  以内，因此要求 MCU 的波特率误差，必须控制在  $\pm 3.5\%$  以内。

如果 MCU 的波特率可以细微调整，可以通过测量 BC7591 的波特率，使得双边波特率精确一致。测量波特率的简单办法，是从 UART 输出数据 0x00，在理想的 9600 波特率下，BC7591 的 TX 引脚将输出一个宽度为 937.5us 的低电平脉冲，通过测量这个脉冲的宽度，即可计算出 BC7591 的实际波特率与标准波特率的误差。当用户 MCU 没有精确的时间基准时(比如使用 RC 震荡器为时钟源)，并无需计算 BC7591 的精确波特率，仅需参照测量结果调整 MCU 一侧波特率，让二者一致即可保证可靠通讯。

BC7591 输出 0 的方法，一个是按键盘的 0 号键，键按下时(并非按键释放时)，会输出键值“0”；或者发送 0xFF 0xFF 指令（两边波特率相对误差须在 50%以内），也会让 BC7591 输出一个 0。

## 极限参数

储存温度	-65°C - +150°C
工作温度	-40°C - +85°C
任意脚对地电压	-0.5 - +6.0V

(表 8: 极限参数)

## 电气特性

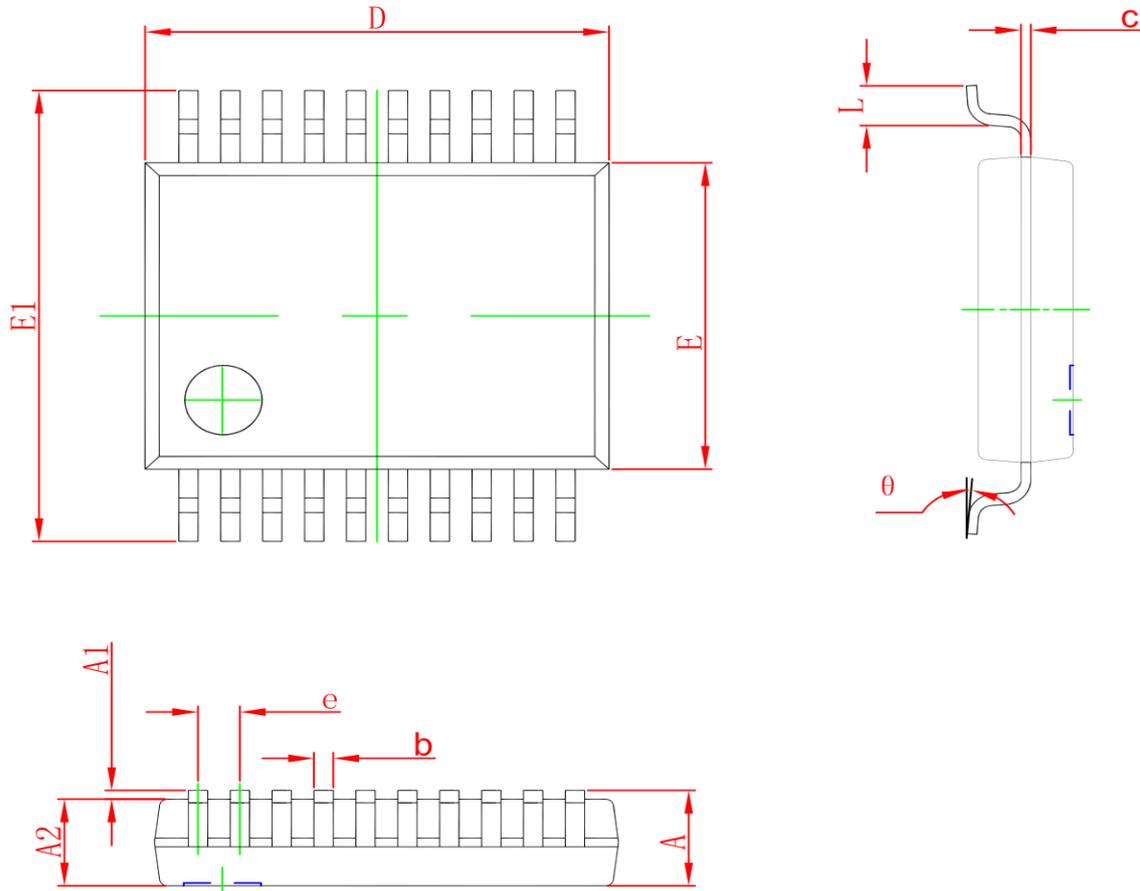
(注: 以下数据除特别注明外,  $T_A=25^\circ\text{C}$ )

参数	最小值	典型值	最大值	备注
$V_{CC}$ 工作电压	3.0V		5.5V	
$I_{CC}$ 工作电流		1.45mA		$V_{CC}=3.3\text{V}$ , 无外接电路
		2.75mA		$V_{CC}=5.0\text{V}$ , 无外接电路
$\Delta_F$ 波特率漂移			$\pm 1.5\%$	$V_{CC}=3.3\text{V}$ , $-40^\circ\text{C} - +85^\circ\text{C}$
			$\pm 1\%$	$V_{CC}=3.3\text{V}$ , $-10^\circ\text{C} - +70^\circ\text{C}$
			$-1\%+0.5\%$	$V_{CC}=5.0\text{V}$ , $-40^\circ\text{C} - +85^\circ\text{C}$
$V_{IL}$ 输入低电平			0.7V	$V_{CC}=3.3\text{V}$
			1.0V	$V_{CC}=5.0\text{V}$
$V_{OL}$ 输出低电平		0.0V		$V_{CC}=5\text{V}$ , $I_{OL}=1\text{mA}$
		1.0V		$V_{CC}=5\text{V}$ , $I_{OL}=100\text{mA}$
		1.0V		$V_{CC}=3.3\text{V}$ , $I_{OL}=60\text{mA}$
$F_{SCN}$ 显示扫描频率		93Hz		
$T_{CL}$ 键盘扫描周期		129ms		即最大按键相应延迟
$T_{db}$ 去抖动时间		11.5ms		即最短可接受按键时间
$T_{pu}$ 上电等待时间		300ms		
$R_{PU}$ 行输入等效上拉电阻		100K		$V_{CC}=3.3\text{V}$
		55K		$V_{CC}=5.0\text{V}$

(表 9: 电气特性)

# 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
θ	0°	8°	0°	8°

## 包装信息

订购型号	包装形式	每包装数量
BC7591EC-T	管装	6600
BC7591EC-RS	盘装(带装)	1000
BC7591EC-RL	盘装(带装)	2000