

目 录

第 1 章	概述.....	1
1.1	ZLG7289B 的主要特性	1
1.2	描述.....	1
1.3	引脚图.....	1
第 2 章	引脚功能说明.....	2
第 3 章	典型应用电路图.....	3
3.1	电路原理图.....	3
3.2	电路简析.....	3
第 4 章	SPI 串行接口	4
4.1	SPI 串行接口信号	4
4.2	SPI 总线时序图	4
4.3	时序图中的各项延迟时间	4
第 5 章	控制指令详解.....	6
5.1	单字节纯指令	6
5.1.1	复位（清除）指令	6
5.1.2	测试指令	6
5.1.3	左移指令	6
5.1.4	右移指令	6
5.1.5	循环左移指令	6
5.1.6	循环右移指令	7
5.2	双字节带数据指令	7
5.2.1	下载数据并且按方式 0 进行译码	7
5.2.2	下载数据并且按方式 1 进行译码	7
5.2.3	下载数据但不译码.....	8
5.2.4	闪烁控制.....	8
5.2.5	消隐控制.....	8
5.2.6	段点亮指令	9
5.2.7	段关闭指令	9
5.2.8	读键盘数据指令	9
第 6 章	实际应用中要注意的若干问题.....	10
6.1	ZLG7289B 要跟着控制面板走	10
6.2	复位引脚可以由主控制器直接控制	10
6.3	驱动 1 英寸以上的大数码管时，要另外加驱动电路	10
6.4	键盘的使用注意事项	10
6.5	降低晶振频率.....	11
第 7 章	特殊应用.....	12
7.1	只使用键盘.....	12
7.2	只使用数码管.....	12
7.3	驱动大型数码管的方法.....	13
第 8 章	C51 驱动程序软件包.....	16
8.1	软件包说明.....	16

8.2	头文件的程序清单.....	16
8.3	C 语言文件的程序清单	18
第 9 章	C51 演示程序.....	24
9.1	演示程序说明.....	24
9.2	演示程序 1.....	24
9.3	演示程序 2.....	28
第 10 章	参考文献.....	34

第1章 概述

1.1 ZLG7289B 的主要特性

- 很宽的工作电压范围：+ 2.7 ~ 6V；
- 直接驱动 8 位共阴式数码管（1 英寸以下）或 64 只独立的 LED；
- 能够管理多达 64 只按键，自动消除抖动；
- 段电流可达 15mA 以上，字电流可达 100mA；
- 利用功率电路可以方便地驱动 1 英寸以上的大型数码管；
- 具有左移、右移、闪烁、消隐、段点亮等强大功能；
- 要显示的数据提供有两种不同的译码方式（也可以选择不译码）；
- 不接数码管而仅使用键盘管理功能时，工作电流可降至 3mA；
- 与微控制器之间采用 SPI 串行总线接口，操作方便，占用 I/O 资源少；
- 工作温度范围：- 40 ~ + 85 ；
- 封装：SOP-28，DIP-28。

1.2 描述

ZLG7289B 是广州周立功单片机发展有限公司自行设计的数码管显示驱动及键盘扫描管理芯片，可直接驱动 8 位共阴式数码管（或 64 只独立 LED），同时还可以扫描管理多达 64 只按键。ZLG7289B 内部含有显示译码器，可直接接受 BCD 码或 16 进制码，并同时具有 2 种译码方式。此外，还具有多种控制指令，如消隐、闪烁、左移、右移、段寻址等。ZLG7289B 采用 SPI 串行总线与微控制器接口，仅占用少数几根 I/O 口线。利用片选信号，多片 ZLG7289B 还可以并接在一起使用，能够方便地实现多于 8 位的显示或多于 64 只按键的应用。

ZLG7289B 可广泛地应用于仪器仪表，工业控制器，条形显示器，控制面板等领域。

1.3 引脚图

1	RTCC	$\overline{\text{RST}}$	28
2	VCC	OSC1	27
3	NC	OSC2	26
4	GND	KC7/DIG7	25
5	NC	KC6/DIG6	24
6	$\overline{\text{CS}}$	KC5/DIG5	23
7	CLK	KC4/DIG4	22
8	$\overline{\text{DIO}}$	KC3/DIG3	21
9	INT	KC2/DIG2	20
10	SG/KR0	KC1/DIG1	19
11	SF/KR1	KC0/DIG0	18
12	SE/KR2	KR7/DP	17
13	SD/KR3	KR6/SA	16
14	SC/KR4	KR5/SB	15

图 1.1 ZLG7289B 引脚图（SOP-28，DIP-28）

第2章 引脚功能说明

表 2.1 ZLG7289B 引脚功能

引脚序号	引脚名称	功能描述
1	RTCC	接电源
2	Vcc	电源, +2.7 ~ 6V
3	NC	悬空
4	GND	接地
5	NC	悬空
6	$\overline{\text{CS}}$	SPI 总线片选信号, 低电平有效
7	CLK	SPI 总线时钟输入信号, 上升沿有效
8	DIO	SPI 总线数据信号, 双向
9	$\overline{\text{INT}}$	键盘中断请求信号, 低电平 (下降沿) 有效
10	SG/KR0	数码管 g 段 / 键盘行信号 0
11	SF/KR1	数码管 f 段 / 键盘行信号 1
12	SE/KR2	数码管 e 段 / 键盘行信号 2
13	SD/KR3	数码管 d 段 / 键盘行信号 3
14	SC/KR4	数码管 c 段 / 键盘行信号 4
15	SB/KR5	数码管 b 段 / 键盘行信号 5
16	SA/KR6	数码管 a 段 / 键盘行信号 6
17	DP/KR7	数码管 dp 段 / 键盘行信号 7
18	DIG0/KC0	数码管字选信号 0 / 键盘列信号 0
19	DIG1/KC1	数码管字选信号 1 / 键盘列信号 1
20	DIG2/KC2	数码管字选信号 2 / 键盘列信号 2
21	DIG3/KC3	数码管字选信号 3 / 键盘列信号 3
22	DIG4/KC4	数码管字选信号 4 / 键盘列信号 4
23	DIG5/KC5	数码管字选信号 5 / 键盘列信号 5
24	DIG6/KC6	数码管字选信号 6 / 键盘列信号 6
25	DIG7/KC7	数码管字选信号 7 / 键盘列信号 7
26	OSC2	晶振输出信号
27	OSC1	晶振输入信号
28	$\overline{\text{RST}}$	复位信号, 低电平有效

第3章 典型应用电路图

3.1 电路原理图

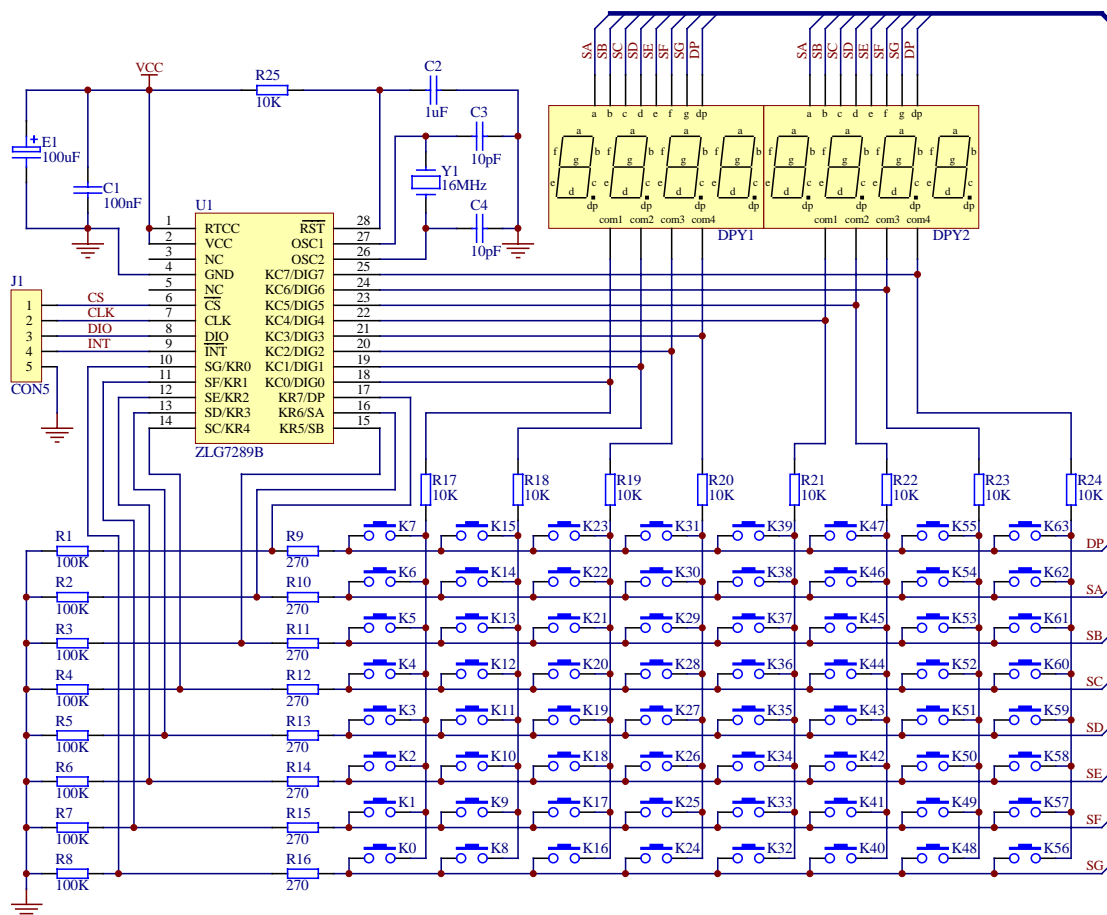


图 3.1 ZLG7289B 典型应用电路原理图

3.2 电路简析

U1 就是 ZLG7289B。为了使电源更加稳定，一般要在 V_{cc} 到 GND 之间接入 $47 \sim 470\mu F$ 的电解电容。J1 是 ZLG7289B 与微控制器的接口。晶振 Y1 取 $4 \sim 16MHz$ ，调节电容 C3 和 C4 通常取值在 $10pF$ 左右。复位信号是低电平有效，一般只需外接简单的 RC 复位电路，也可以通过直接拉低 RST 引脚的方法进行复位。

数码管必须是共阴式的，不能直接使用共阳式的。DPY1 和 DPY2 是 4 位联体式数码管，共同组成完整的 8 位，当然还可以采用其它的组合方式，如 4 只双联体式数码管。数码管在工作时要消耗较大的电流，R9 ~ R16 是限流电阻，典型值是 270Ω 。如果要增大数码管的亮度，可以适当减小电阻值，最低 200Ω 。

64 只按键中，左下角是 K0，右上角是 K63。为了使键盘扫描得以正常进行，下拉电阻 R1 ~ R8 和位选电阻 R17 ~ R24 是必须的。它们之间还要遵从一定的比例关系，比值在 5:1 到 50:1 之间，典型值是 10:1。下拉电阻取值范围在 $10 \sim 100K \Omega$ ，位选电阻取值范围在 $1 \sim 10K \Omega$ 。在多数应用当中可能用不到太多的按键，建议按列裁减键盘，则相应列的位选电阻可以省略。但是下拉电阻一个都不能省去，除非完全不使用键盘。完全不使用键盘的具体电路图请参见第 7 章图 7.2。

第4章 SPI 串行接口

4.1 SPI 串行接口信号

ZLG7289B 与微控制器的接口采用 3 线制 SPI 串行总线，由 \overline{CS} 、CLK 和 DIO 这 3 根信号线组成。 \overline{CS} 和 CLK 是输入信号，由微控制器提供。DIO 信号是双向的，必须接到微控制器上具有双向功能的 I/O 上。SPI 信号线的具体意义请参见表 4.1。操作 SPI 总线的时序图请参见图 4.1、图 4.2 和图 4.3。其中图 4.3 是读按键值的时序图，只有当 INT 引脚出现下跳沿时才允许去读取按键值，否则将得不到有意义的数。

表 4.1 ZLG7289B 的 SPI 串行接口信号

信号名称	引脚序号	描述
\overline{CS}	6	SPI 总线片选输入信号，低电平有效
CLK	7	SPI 总线时钟输入信号，上升沿有效
DIO	8	SPI 总线数据信号，双向

4.2 SPI 总线时序图

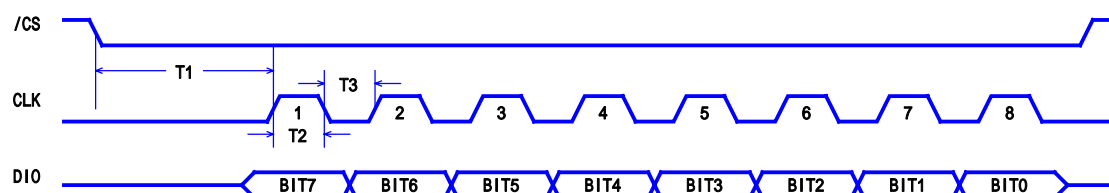


图 4.1 纯指令时序图 (单字节命令)

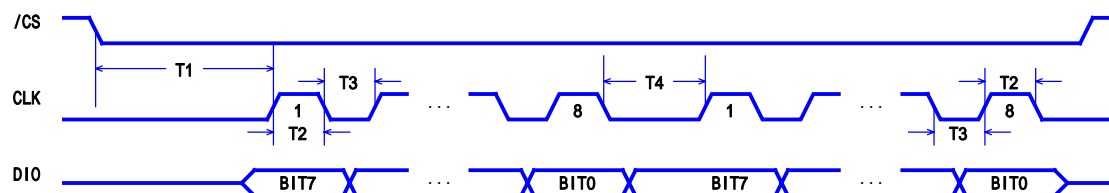


图 4.2 带数据指令时序图 (命令字在前，输入数据在后)

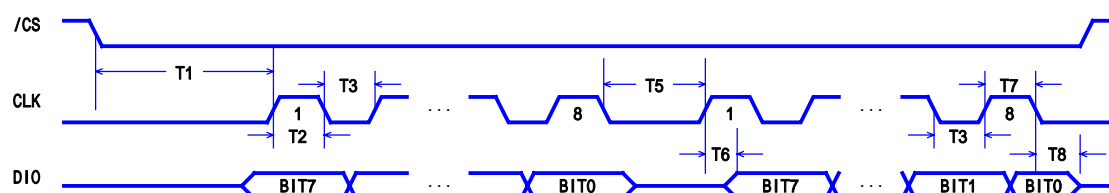


图 4.3 读键盘指令时序图 (命令字在前，键值在后)

4.3 时序图中的各项延迟时间

表 4.2 中各项参数的测试条件为：Vcc=5.0V，Fosc=16MHz。

表 4.2 时序图中的各项延迟时间

符号	名称	最小值	典型值	单位
T1	片选信号 \overline{CS} 的建立时间	25	50	μs
T2	CLK 信号高电平的宽度	5	8	μs
T3	CLK 信号低电平的宽度	5	8	μs
T4	命令字与输入数据之间的时间间隔	15	25	μs
T5	命令字与输出数据（按键值）之间的时间间隔	15	25	μs
T6	输出数据（按键值）建立时间	5	8	μs
T7	读取输出数据（按键值）时 CLK 信号高电平的宽度	5	8	μs
T8	DIO 信号从输出状态切换到输入状态的时间	-	5	μs

第5章 控制指令详解

ZLG7289B 的控制指令分为单字节纯指令和双字节带数据指令两大类。操作这些指令的相关时序请参考第 4 章。

5.1 单字节纯指令

所有这些指令的长度都是 1 个字节。执行这一类指令时，不需要附带任何其它数据。

5.1.1 复位（清除）指令

D7	D6	D5	D4	D3	D2	D1	D0
1	0	1	0	0	1	0	0

这时一条软复位指令，执行后将数码管所有的显示内容清除掉，原先设置的所有闪烁、消隐等属性也一并被清除，就像硬件复位一样。

5.1.2 测试指令

D7	D6	D5	D4	D3	D2	D1	D0
1	0	1	1	1	1	1	1

该指令使所有的数码管各段包括小数点在内全部点亮，并处于不断闪烁之中。这条指令可用于生产测试，以确定 ZLG7289B 或数码管是否有问题。

5.1.3 左移指令

D7	D6	D5	D4	D3	D2	D1	D0
1	0	1	0	0	0	0	0

该指令使数码管所有的显示自右向左移动一位（以第 3 章中图 3.1 的接法为准），处于闪烁和消隐状态的显示位也一起被移动。原来最左边的显示数据被移出后自动丢弃，最右边的一位用无任何显示的空白代替。每执行一次该指令，就左移一位。例如，数码管原来的显示为：

1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---

执行一次左移指令后，显示就变成了：

2	3	4	5	6	7	8	
---	---	---	---	---	---	---	--

5.1.4 右移指令

D7	D6	D5	D4	D3	D2	D1	D0
1	0	1	0	0	0	0	1

与左移指令类似，执行该指令后，数码管的数据显示向右移动一位，原来最右边的一位被丢弃，而最左边的一位用空白代替。

5.1.5 循环左移指令

D7	D6	D5	D4	D3	D2	D1	D0
1	0	1	0	0	0	1	0

与左移指令类似，但原来最左边被移出的显示数据不是被丢弃，而是补在最右边。例如数码管原来的显示为：

1	2	3	4	5	6	7	8
---	---	---	---	---	---	---	---

执行一次循环左移指令后，显示就变成了：

2	3	4	5	6	7	8	1
---	---	---	---	---	---	---	---

5.1.6 循环右移指令

D7	D6	D5	D4	D3	D2	D1	D0
1	0	1	0	0	0	1	1

与右移指令类似，但原来最右边被移出的显示数据不是被丢弃，而是补在最左边。

5.2 双字节带数据指令

所有这些指令的长度都是 2 个字节。第 1 字节是命令字，第 2 字节是输入或输出的数据。

5.2.1 下载数据并且按方式 0 进行译码

D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	a ₂	a ₁	a ₀		dp	x	x	x	d ₃	d ₂	d ₁	d ₀

在指令格式中，高 5 位的 10000 是命令字段；a₂a₁a₀ 是数码管显示数据的位地址，位地址编号按从左到右的顺序依次为 0、1、2、3、4、5、6、7（以第 3 章中的图 3.1 为准）；dp 控制小数点是否显示，dp = 0 时该位的小数点被点亮，dp = 1 时该位的小数点被熄灭；xxx 是无关位；d₃d₂d₁d₀ 是要显示的数据。显示数据按照下表中的规则进行译码：

表 5.1 下载数据命令译码方式 0

d ₃ d ₂ d ₁ d ₀ (二进制)				d ₃ d ₂ d ₁ d ₀ (十六进制)				显示结果
0	0	0	0	00H				0
0	0	0	1	01H				1
0	0	1	0	02H				2
0	0	1	1	03H				3
0	1	0	0	04H				4
0	1	0	1	05H				5
0	1	1	0	06H				6
0	1	1	1	07H				7
1	0	0	0	08H				8
1	0	0	1	09H				9
1	0	1	0	0AH				-
1	0	1	1	0BH				E
1	1	0	0	0CH				H
1	1	0	1	0DH				L
1	1	1	0	0EH				P
1	1	1	1	0FH				(无显示)

5.2.2 下载数据并且按方式 1 进行译码

D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
1	1	0	0	1	a ₂	a ₁	a ₀		dp	x	x	x	d ₃	d ₂	d ₁	d ₀

在指令格式中，高 5 位的 11001 是命令字段；a₂a₁a₀ 是数码管显示数据的位地址，位地

址编号按从左到右的顺序依次为 0、1、2、3、4、5、6、7 (以第 3 章中的图 3.1 为准); dp 控制小数点是否显示, dp = 0 时该位的小数点被点亮, dp = 1 时该位的小数点被熄灭; xxx 是无关位; d₃d₂d₁d₀ 是要显示的数据。显示数据按照下表中的规则进行译码:

表 5.2 下载数据命令译码方式 1

d ₃ d ₂ d ₁ d ₀ (二进制)				d ₃ d ₂ d ₁ d ₀ (十六进制)	显示结果
0	0	0	0	00H	0
0	0	0	1	01H	1
0	0	1	0	02H	2
0	0	1	1	03H	3
0	1	0	0	04H	4
0	1	0	1	05H	5
0	1	1	0	06H	6
0	1	1	1	07H	7
1	0	0	0	08H	8
1	0	0	1	09H	9
1	0	1	0	0AH	A
1	0	1	1	0BH	b
1	1	0	0	0CH	C
1	1	0	1	0DH	d
1	1	1	0	0EH	E
1	1	1	1	0FH	F

5.2.3 下载数据但不译码

D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	0	a ₂	a ₁	a ₀		dp	a	b	c	d	e	f	g

在指令格式中, 高 5 位的 10010 是命令字段; a₂a₁a₀ 是数码管显示数据的位地址, 位地址编号按从左到右的顺序依次为 0、1、2、3、4、5、6、7 (以第 3 章中的图 3.1 为准); dp 控制小数点是否显示, dp = 0 时该位的小数点被点亮, dp = 1 时该位的小数点被熄灭; abcdefg 对应数码管内部的 7 个 LED 字段。

不译码的数据下载方式给用户提供了最大的灵活性, dp 连同 abcdefg 一共有 256 种不同的组合, 想怎样显示就怎样显示。

5.2.4 闪烁控制

D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	1	0	0	0		d ₇	d ₆	d ₅	d ₄	d ₃	d ₂	d ₁	d ₀

该指令控制数码管各位的闪烁属性。在指令格式中, 第 1 字节是命令字段; 第 2 字节位的 d₇d₆d₅d₄d₃d₂d₁d₀ 分别对应数码管的第 7 至第 0 位, 0 - 闪烁, 1 - 不闪烁。复位后, 所有位都不闪烁。

5.2.5 消隐控制

D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	1	0	0	0		d ₇	d ₆	d ₅	d ₄	d ₃	d ₂	d ₁	d ₀

该指令控制数码管各位的消隐属性。在指令格式中, 第 1 字节是命令字段; 第 2 字节的

$d_7d_6d_5d_4d_3d_2d_1d_0$ 分别对应数码管的第 0 至第 7 位, 0 - 消隐, 1 - 显示。复位后, 所有位都不消隐。

当数码管的某一位被设置成消隐属性后, ZLG7289B 在进行扫描显示时将跳过该位, 该位的扫描时间将分配给其它位。一旦某一位设置了消隐属性, 则无论对该位写入什么样的数据都不会被显示出来。写入的数据不是被丢弃, 而是保存在内部的数据寄存器中。如果去掉该位的消隐属性, 则最后一次写入的数据有效并立即显示出来。

消隐功能的用途: 如果实际使用的数码管位数不足 8 位, 则可以将不用的位设为消隐属性, 这样可以提高有用位的显示亮度。

注意: 至少应有 1 位保持显示状态。如果在消隐控制指令中所有位全部为 0, 则该指令将不被接受, ZLG7289B 仍然保持原有的消隐状态不变。

5.2.6 段点亮指令

D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	0	0	0	0	0		x	x	d_5	d_4	d_3	d_2	d_1	d_0

该指令可以单独点亮数码管中的某一指定的段, 或者 LED 矩阵中某一指定的 LED。在指令格式中, 第 1 字节是命令字段; xx 表示无关位; $d_5d_4d_3d_2d_1d_0$ 是 6 位段地址。在某位数码管里, 各段的点亮顺序按照 “g,f,e,d,c,b,a,dp” 进行。

5.2.7 段关闭指令

D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
1	1	0	0	0	0	0	0		x	x	d_5	d_4	d_3	d_2	d_1	d_0

该指令可以单独熄灭数码管中的某一指定的段, 或者 LED 矩阵中某一指定的 LED。在指令格式中, 第 1 字节是命令字段; xx 表示无关位; $d_5d_4d_3d_2d_1d_0$ 是 6 位段地址。在某位数码管里, 各段的关闭顺序按照 “g,f,e,d,c,b,a,dp” 进行。

5.2.8 读键盘数据指令

D7	D6	D5	D4	D3	D2	D1	D0		D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	1	0	1	0	1		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0

当有键按下时, ZLG7289B 的 INT 引脚会变成低电平。这时利用该指令可以读出当前的键值。与其它带数据指令不同的是, 第 2 字节是 ZLG7289B 向微控制器返回的键值, 而不是输入数据。正常情况下, 键值的范围是 0 ~ 63 (00H ~ 3FH), 无按键的状态用 255 (FFH) 表示。在第 3 章图 3.1 中, K0 ~ K63 所对应的键值就是 0 ~ 63。

第6章 实际应用中要注意的若干问题

6.1 ZLG7289B 要跟着控制面板走

ZLG7289B 可广泛应用于仪器仪表, 工业控制器, 条形显示器, 控制面板等领域。在实际应用中, 控制面板和主机板往往是分离的, 它们之间有几十厘米的距离, 要用长长的排线相连。键盘和数码管一般都位于控制面板上, 主控制器则在主机板上。在设计时千万注意: ZLG7289B 一定要跟着控制面板走, 而不要放在主机板上。ZLG7289B 驱动数码管显示采用的是动态扫描法, 为了防止显示出现闪烁, 采用了比较高的扫描频率。扫描键盘同样用的也是频率较高的信号。如果 ZLG7289B 放在主机板上, 这些扫描信号势必要走长线, 而高频信号最忌讳走长线了, 这容易导致显示混乱、按键失灵等故障。如果 ZLG7289B 放在控制面板上, 由于走的是短线, 就不易出现上述问题了。不必担心 ZLG7289B 与主控制器之间通信的 SPI 总线会有问题。因为 SPI 总线的通信速率是由主控制器控制的, 可以做得低一些, 所以允许走长线。

6.2 复位引脚可以由主控制器直接控制

在工业控制应用中, 为了增强抗干扰能力, 建议采用独立的稳定直流电源给 ZLG7289B 供电, V_{cc} 与 GND 之间的电容也要相应加大。另外复位引脚最好由主控制器来控制, 每隔几分钟强制复位一次, 复位脉冲宽度可以在 50ms 左右, 一闪而过, 肉眼很难察觉。定时强制复位可以有效防止偶尔由于电磁干扰而产生的显示不正常和按键失灵的现象。

6.3 驱动 1 英寸以上的大数码管时, 要另外加驱动电路

ZLG7289B 的驱动能力毕竟是有限的, 如果直接驱动 1 英寸以上的大数码管则可能会导致显示亮度不够。这时可以适当减小限流电阻 (最低 200 Ω) 以增加亮度。如果亮度仍然不够, 就必须另外添加驱动电路。更深入的讨论请参见第 7.3 节。

6.4 键盘的使用注意事项

ZLG7289B 在扫描键盘时, 已经采取了消抖动措施, 因此在程序中不必另外编写消抖动的代码。

如果用了键盘, 哪怕只有一个按键, 则 R1 ~ R8 (参见图 3.1) 统统不能省略。但如果某一系列键盘未使用, 则相应的位选电阻可以省略。

某个按键按下时, ZLG7289B 的 INT 引脚会出现低电平, 向主控制器发出中断请求。主控制器既可以采用中断方式处理, 也可以采用查询 INT 引脚电平状态的方法处理。但是要避免通过 SPI 总线用软件命令的方式去查询是否有键按下, 这将导致 SPI 总线频繁处于活动状态, 不利于抗干扰。应当在 INT 引脚出现低电平时及时地读取键值。读取键值后, INT 引脚并不会自动恢复为高电平, 一定要等到按键抬起为止。如果没有及时读取按键值, 则按键抬起后 INT 引脚也将恢复到高电平, 而在 INT 引脚处于高电平期间, 试图去读取键值将可能得不到有意义的数据。

利用中断方式处理按键时, 建议将微控制器外部中断的触发方式设置成负边沿触发, 而不要设置成低电平触发。按下某个键时, ZLG7289B 会在 INT 引脚产生低电平信号, 这个低电平信号直到松开按键时才会撤除。如果程序中采用低电平触发中断, 则进入中断完成读取键值操作后, 还必须要等待 INT 信号恢复为高电平, 即等待操作者放键, 在等待期间, CPU 几乎不能再干其它事情, 造成浪费。如果不等待, 读完键值后就直接从中断返回主程序, 那么由于 INT 信号还是低电平, 这将再次触发中断, 从而导致程序错误。如果设置成负边沿触发方式, 则进入中断读完键值后不必等待即可退出, 返回主程序后也不会再次触发中断。

6.5 降低晶振频率

在 ZLG7289B 的典型应用电路图当中，晶振用的是 16MHz。但是在电磁环境恶劣的现场，还是应该把晶振频率降下来为妙。许多本来“有问题”的电路，在把晶振速度降下来之后就完全正常了。降到多少合适呢？这里推荐值为 1 ~ 4MHz。晶振频率降低后，SPI 总线的通信速率也要适当降低。ZLG7289B 的闪烁显示功能将受到影响，闪烁速度将因晶振频率的下降而跟着变慢。

第7章 特殊应用

7.1 只使用键盘

ZLG7289B 在某些应用中，可能不需要使用数码管，而只使用其键盘扫描管理功能，这时，工作电流可降至 3mA。在典型应用电路图（图 3.1）中，用于数码管限流的 8 只 270 电阻可以省略（省略后用导线代替）。只使用键盘的具体用法请参考下面的电路图（图 7.1）。

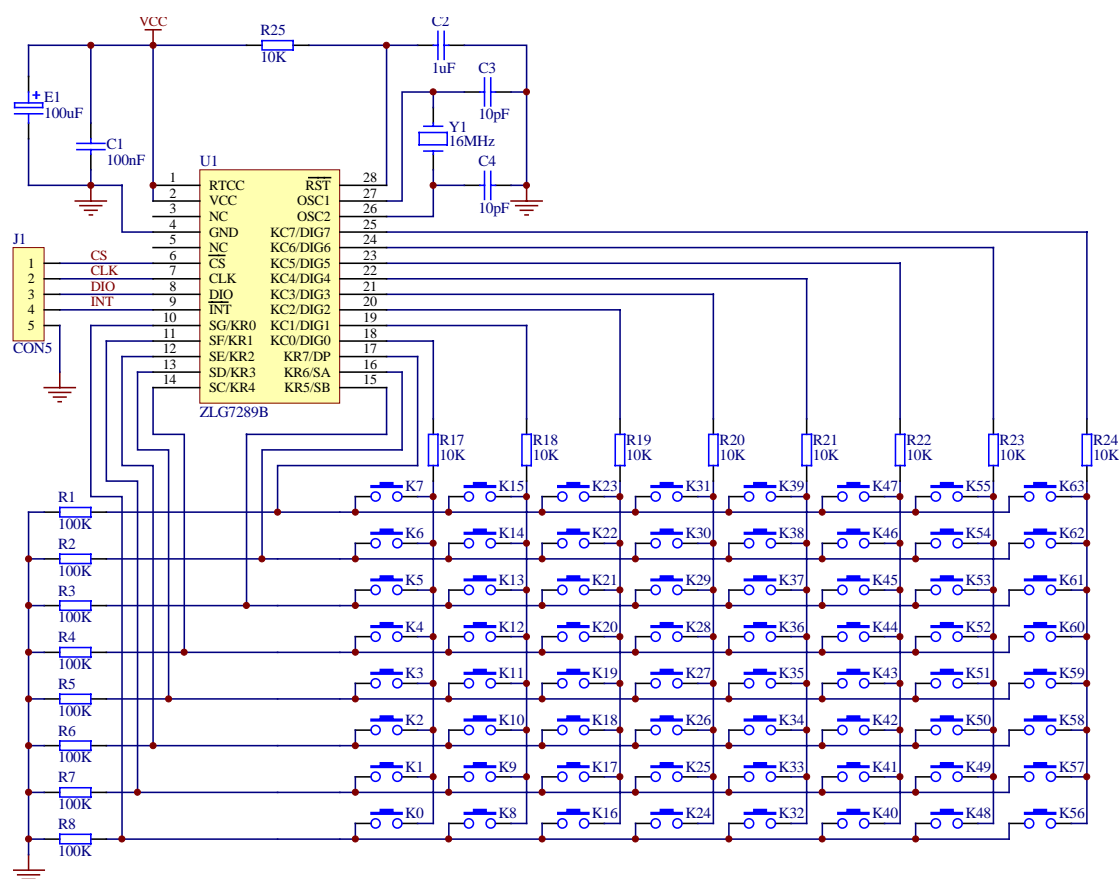


图 7.1 只使用键盘的应用电路图

7.2 只使用数码管

ZLG7289B 在某些应用中，可能不需要使用键盘，而只使用其数码管显示驱动功能。用于键盘扫描的下拉电阻和位选电阻都可以去掉不要，电路因此大为简化。只使用数码管的具体用法请参考下面的电路图（图 7.2）。

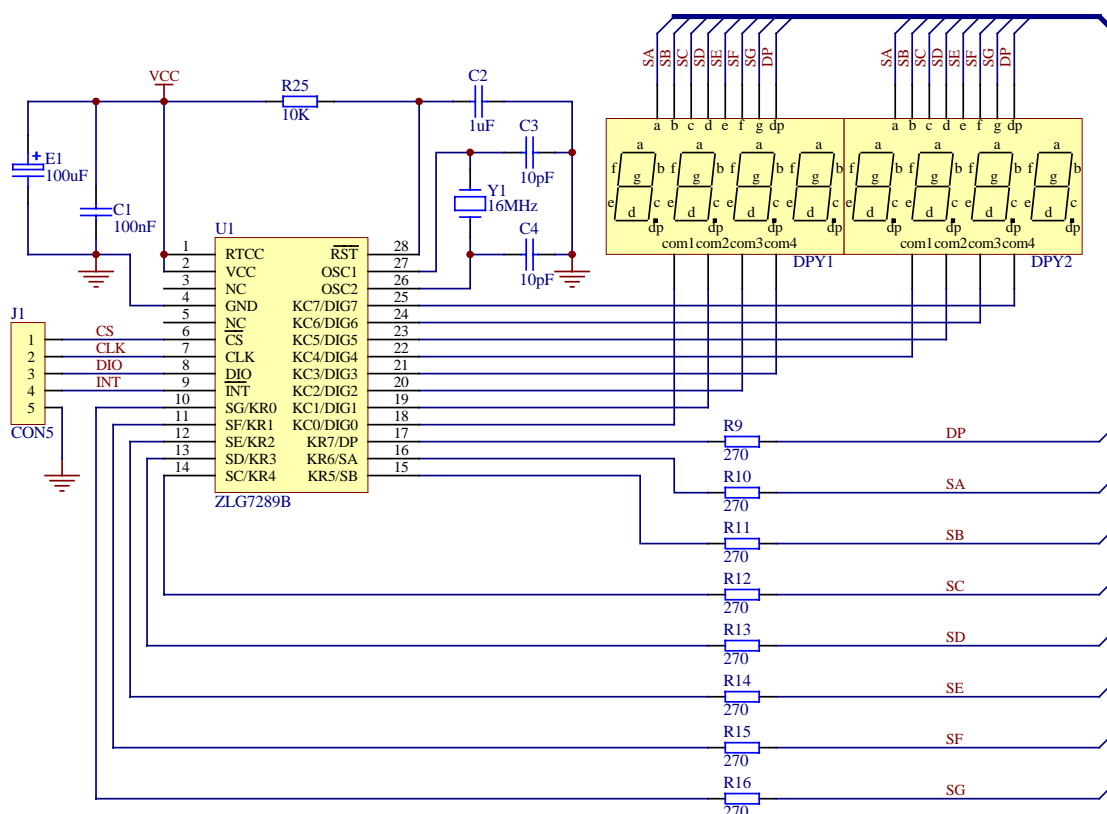


图 7.2 只使用数码管的应用电路图

7.3 驱动大型数码管的方法

ZLG7289B 的驱动能力毕竟有限，如果要使用 1 英寸以上的大型数码管，则显示亮度可能不够，这时可以考虑适当减小数码管的限流电阻（参见第 3 章图 3.1 中的 R9 ~ R16）的阻值以增加亮度，阻值最小为 200 Ω 。如果亮度依然不够，就必须另外加入功率驱动电路。采用功率晶体管作为驱动电路是很容易想到的方案，其原理如图 7.3 所示。

在图 7.3 中，以第 1 路驱动为例，ZLG7289B 的段选信号 SA 是高电平有效的，DIG0 则是低电平有效。SA 为高电平时，使 Q1（NPN 通用型）导通，Q1 的导通又使得 Q2（PNP 功率型）导通；DIG0 为低电平时，使 Q17（PNP 通用型）导通，Q17 的导通又使得 Q18（NPN 功率型）导通；这样就有电流从 Vs 经限流电阻 R3（功率电阻，阻值视具体情况而定）和数码管流到 GND，于是相应的数码管字段就被点亮。如果 SA 和 DIG0 信号不是一高一低的组合，则相应的数码管字段就不会亮。这种接法完全符合 ZLG7289B 的动态扫描工作方式，并且不会影响键盘扫描管理功能。注意，图中的电源 Vs 可以是与 ZLG7289B 相同的电源 Vcc，也可以是单独的高压电源（20V 以内）。

用分立功率晶体管作为驱动电路显然太麻烦，要动用 16 只通用型晶体管和 16 只功率型晶体管，以及若干只电阻。那么有没有替代它们的功率集成电路呢？Allegro 公司的 8 路达林顿阵列 UDN2981A 和 UDN2596A 就是一对很好的组合，它们的驱动电流高达 1A。其内部结构请参考图 7.4，其中 UDN2981A 相当于图 7.3 中 Q1 与 Q2 的组合，UDN2596A 相当于图 7.3 中 Q17 与 Q18 的组合。ZLG7289B 与它们相配合一起驱动大型数码管的完整电路图请参考图 7.5，可以看出，这比分立功率晶体管的电路简单多了。图 7.5 中 UDN2981A 的电源 Vs 可以是单独的高压电源（20V 以内），R26 ~ R33 是限流电阻（功率型），阻值视具体情况而定。

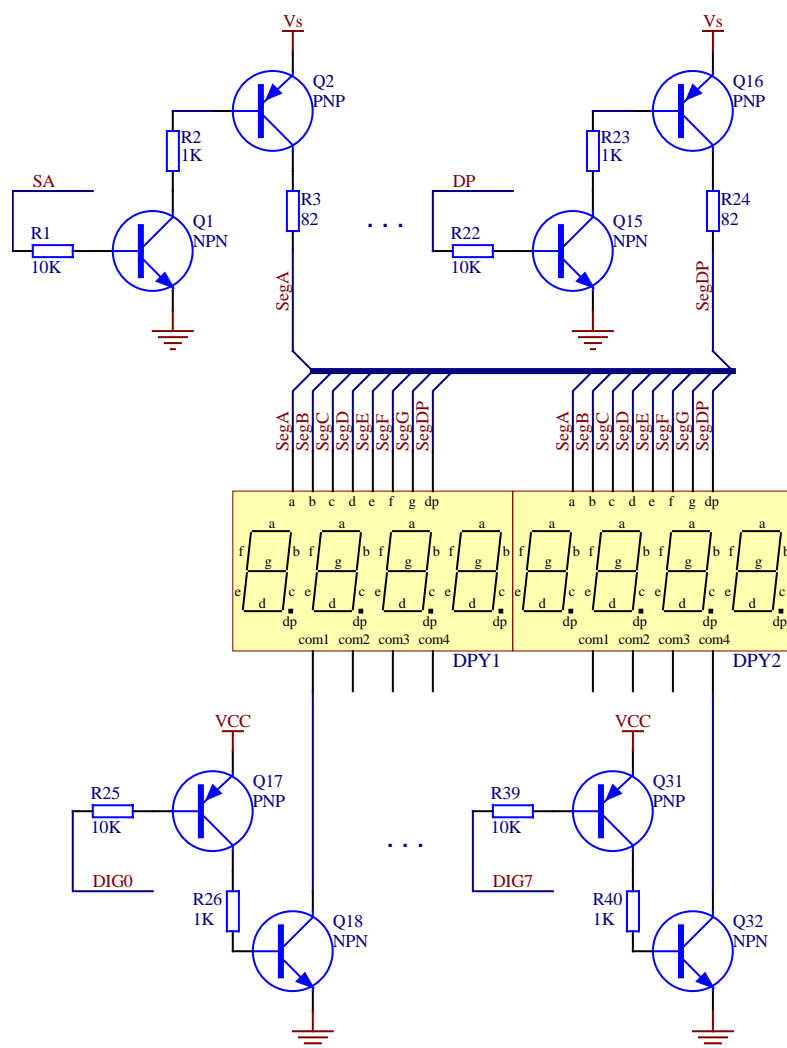


图 7.3 使用分立功率晶体管驱动大型数码管

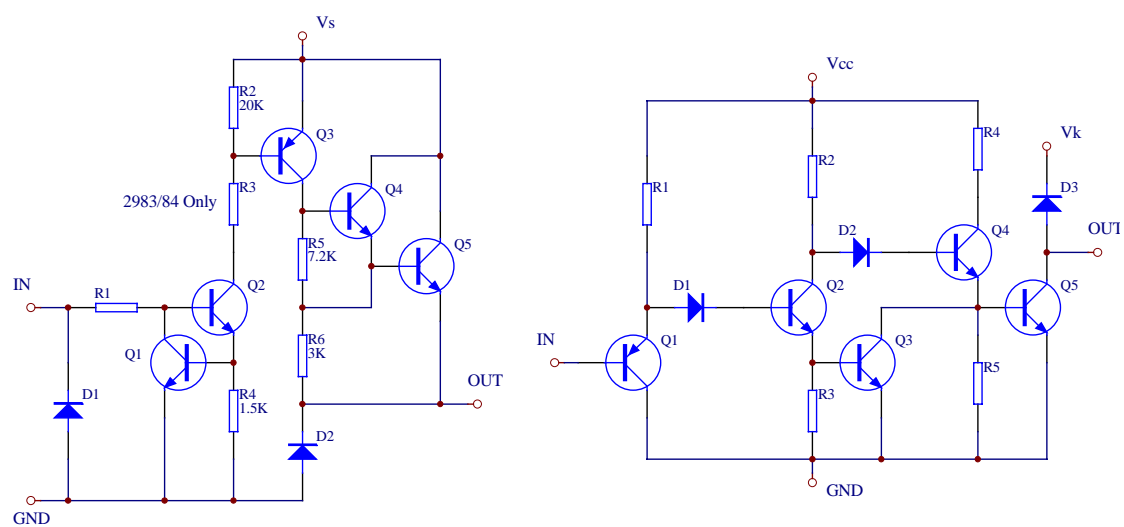


图 7.4 UDN2981A (左) 和 UDN2596A (右) 其中一路的内部结构

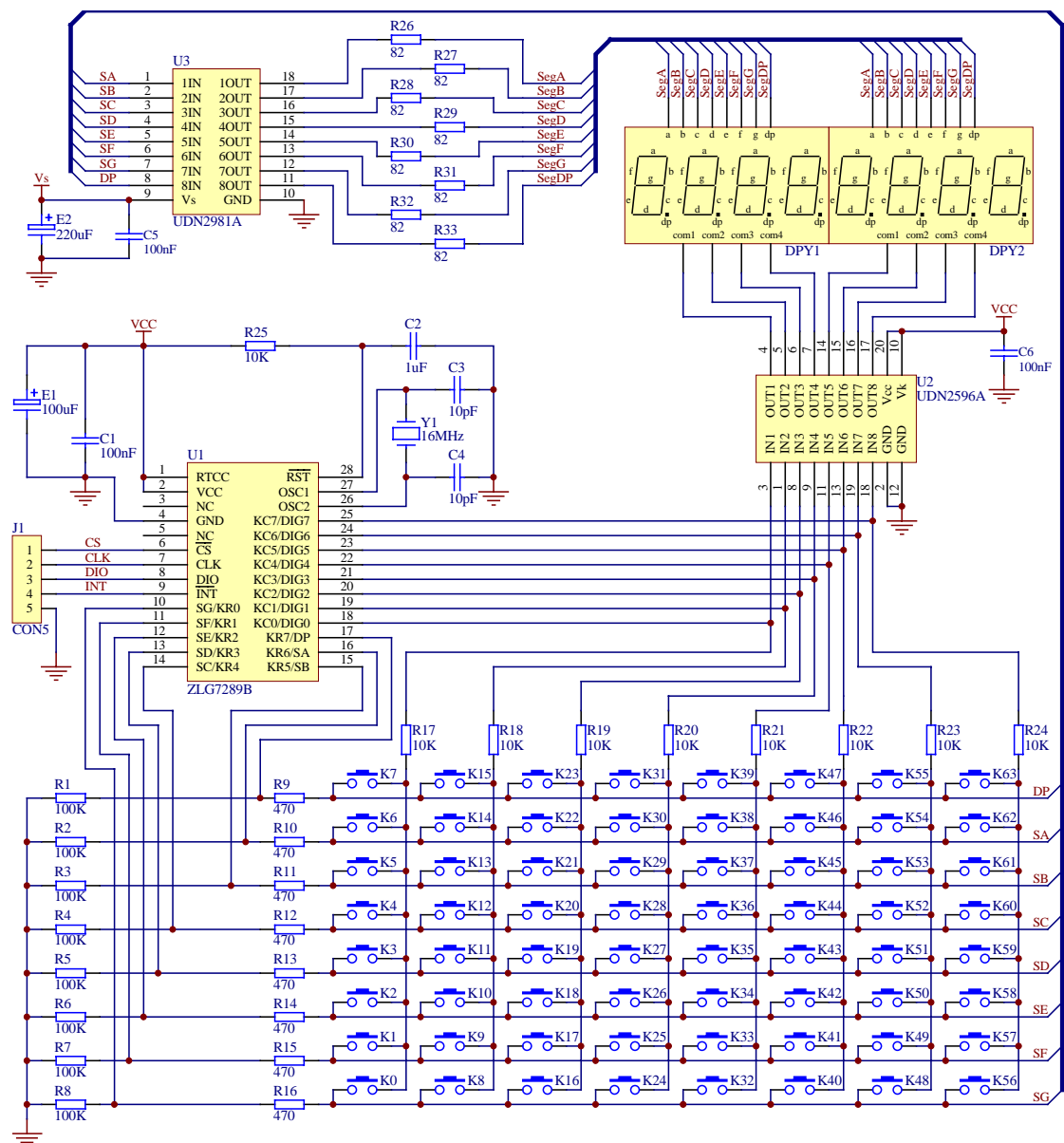


图 7.5 ZLG7289B 利用达林顿阵列驱动大型数码管

第8章 C51 驱动程序软件包

8.1 软件包说明

ZLG7289B 的 80C51 单片机 C51 驱动程序软件包由两个文件组成：“ZLG7289.h”和“ZLG7289.c”。头文件“ZLG7289.h”包括 ZLG7289B 的 I/O 接口定义和用户指令集声明，C 语言文件“ZLG7289.c”是这些指令的具体实现。

以 Keil C51 为例，该软件包的具体用法是：

- 将文件“ZLG7289.c”和“ZLG7289.h”一起复制到您的工程文件夹下；
- 根据电路的实际情况，在文件“ZLG7289.h”中重新定义 I/O 接口；
- 把文件“ZLG7289.c”添加进工程中，在需要的地方包含头文件“ZLG7289.h”；
- 在 main() 函数的开始处应当调用一次初始化函数 ZLG7289_Init()；
- 以后在程序中可以直接使用 ZLG7289B 的用户指令集了；
- 读取键值时使用函数 ZLG7289_Key()，查询方式、中断方式皆可。

用户指令集汇总：

- 复位指令：ZLG7289_Reset()；
- 测试指令：ZLG7289_Test()；
- 左移指令：ZLG7289_SHL()；
- 右移指令：ZLG7289_SHR()；
- 循环左移：ZLG7289_ROL()；
- 循环右移：ZLG7289_ROR()；
- 下载数据：ZLG7289_Download(mod, x, dp, dat)；
- 闪烁控制：ZLG7289_Flash(x)；
- 消隐控制：ZLG7289_Hide(x)；
- 段点亮控制：ZLG7289_SegOn(seg)；
- 段关闭控制：ZLG7289_SegOff(seg)；
- 读键盘数据指令：char ZLG7289_Key()。

8.2 头文件的程序清单

程序清单 8.1 ZLG7289B 的 C51 驱动程序头文件 (ZLG7289.h)

```
/*
ZLG7289.h
数码管显示与键盘管理芯片 ZLG7289 的标准 80C51 驱动程序头文件
Copyright (c) 2005 , 广州周立功单片机发展有限公司
All rights reserved.
本程序仅供学习参考，不提供任何可靠性方面的担保；请勿用于商业目的。
*/

#ifndef _ZLG7289_H_
#define _ZLG7289_H_
```

```
#include <reg52.h>

//定义 I/O 接口
sbit ZLG7289_pinCS = P1^0;      //片选信号，低电平有效
sbit ZLG7289_pinCLK = P1^1;     //时钟信号，上升沿有效
sbit ZLG7289_pinDIO = P1^2;     //数据信号，双向
sbit ZLG7289_pinINT = P3^2;     //键盘中断请求信号，低电平（负边沿）有效

//执行 ZLG7289 纯指令
extern void ZLG7289_cmd(char cmd);

//执行 ZLG7289 带数据指令
extern void ZLG7289_cmd_dat(char cmd, char dat);

//以下是 ZLG7289 的用户指令集

//复位（清除）指令
#define ZLG7289_Reset()          ZLG7289_cmd(0xA4)

//测试指令
#define ZLG7289_Test()           ZLG7289_cmd(0xBF)

//左移指令
#define ZLG7289_SHL()             ZLG7289_cmd(0xA0)

//右移指令
#define ZLG7289_SHR()             ZLG7289_cmd(0xA1)

//循环左移指令
#define ZLG7289_ROL()             ZLG7289_cmd(0xA2)

//循环右移指令
```

```
#define ZLG7289_ROR()          ZLG7289_cmd(0xA3)

//下载数据
extern void ZLG7289_Download(unsigned char mod, char x, bit dp, char dat);

//闪烁控制
// x 的 8 个位分别对应数码管的 8 个位 , 0 - 闪烁 , 1 - 不闪烁
#define ZLG7289_Flash(x)  ZLG7289_cmd_dat(0x88,(x))

//消隐控制
// x 的 8 个位分别对应数码管的 8 个位 , 0 - 消隐 , 1 - 显示
#define ZLG7289_Hide(x)    ZLG7289_cmd_dat(0x98,(x))

//段点亮指令
//seg=0 ~ 63 , 8 只数码管被看成 64 只独立的 LED
//每只数码管中各段的点亮顺序按照 “ g , f , e , d , c , b , a , dp ” 进行
#define ZLG7289_SegOn(seg)  ZLG7289_cmd_dat(0xE0,(seg))

//段关闭指令
//seg=0 ~ 63 , 8 只数码管被看成 64 只独立的 LED
//每只数码管中各段的熄灭顺序按照 “ g , f , e , d , c , b , a , dp ” 进行
#define ZLG7289_SegOff(seg) ZLG7289_cmd_dat(0xC0,(seg))

//执行 ZLG7289 键盘命令
extern char ZLG7289_Key() reentrant;

//ZLG7289 初始化
extern void ZLG7289_Init(unsigned char t);

#endif    //_ZLG7289_H_
```

8.3 C 语言文件的程序清单

程序清单 8.2 ZLG7289B 的 C51 驱动程序 C 语言文件 (ZLG7289.c)

```
/*
    ZLG7289.c
```

```
数码管显示与键盘管理芯片 ZLG7289 的标准 80C51 驱动程序 ( C51 )
Copyright (c) 2005 , 广州周立功单片机发展有限公司
All rights reserved.
本程序仅供学习参考 , 不提供任何可靠性方面的担保 ; 请勿用于商业目的。

*/

#include "ZLG7289.h"

//定义一个全局变量 , 用于延时
//该变量在调用函数 ZLG7289_Init()时被初始化
unsigned char ZLG7289_Delay_t;

/*
宏定义 : ZLG7289_ShortDelay()
功能 : 短延时
说明 : 延时(ZLG7289_Delay_t*2+2)个机器周期
*/
#define ZLG7289_ShortDelay() \
{ \
    unsigned char t = ZLG7289_Delay_t;\
    while ( --t != 0 );\
}

/*
宏定义 : ZLG7289_LongDelay()
功能 : 长延时
说明 : 延时(ZLG7289_Delay_t*12+8)个机器周期
*/
#define ZLG7289_LongDelay() \
{ \
    unsigned char t = ZLG7289_Delay_t * 6;\
    while ( --t != 0 );\
}

/*
函数 : ZLG7289_SPI_Write()
功能 : 向 SPI 总线写入 1 个字节的数据
参数 :
    dat : 要写入的数据
*/
void ZLG7289_SPI_Write(char dat) reentrant
```

```

{
    unsigned char t = 8;
    do
    {
        ZLG7289_pinDIO = (bit)(dat & 0x80);
        dat <<= 1;
        ZLG7289_pinCLK = 1;
        ZLG7289_ShortDelay();
        ZLG7289_pinCLK = 0;
        ZLG7289_ShortDelay();
    } while ( --t != 0 );
}

/*
函数：ZLG7289_SPI_Read()
功能：从 SPI 总线读取 1 个字节的数据
返回：
    读取到的数据
*/
char ZLG7289_SPI_Read() reentrant
{
    char dat;
    unsigned char t = 8;
    ZLG7289_pinDIO = 1;          //读取数据之前 DIO 引脚要置 1 以切换到输入状态
    do
    {
        ZLG7289_pinCLK = 1;
        ZLG7289_ShortDelay();
        dat <<= 1;
        if ( ZLG7289_pinDIO ) dat++;
        ZLG7289_pinCLK = 0;
        ZLG7289_ShortDelay();
    } while ( --t != 0 );
    return dat;
}

/*
函数：ZLG7289_cmd()
功能：执行 ZLG7289 纯指令
参数：
    cmd：命令字
*/

```

```
void ZLG7289_cmd(char cmd)
{
    char sav = IE;
    IE &= 0xFA;          //关闭外部中断
    ZLG7289_pinCS = 0;
    ZLG7289_LongDelay();
    ZLG7289_SPI_Write(cmd);
    ZLG7289_pinCS = 1;
    ZLG7289_LongDelay();
    IE = sav;            //恢复 IE
}
```

/*

函数：ZLG7289_cmd_dat()

功能：执行 ZLG7289 带数据指令

参数：

cmd：命令字

dat：数据

*/

```
void ZLG7289_cmd_dat(char cmd, char dat)
{
    char sav = IE;
    IE &= 0xFA;          //关闭外部中断
    ZLG7289_pinCS = 0;
    ZLG7289_LongDelay();
    ZLG7289_SPI_Write(cmd);
    ZLG7289_LongDelay();
    ZLG7289_SPI_Write(dat);
    ZLG7289_pinCS = 1;
    ZLG7289_LongDelay();
    IE = sav;            //恢复 IE
}
```

/*

函数：ZLG7289_Download()

功能：下载数据

参数：

mod=0：下载数据且按方式 0 译码

mod=1：下载数据且按方式 1 译码

mod=2：下载数据但不译码

x：数码管编号（横坐标），取值 0~7

dp=0：小数点不亮

dp=1 : 小数点亮
dat : 要显示的数据

说明 :

译码方式的具体含义请参考数据手册

*/

void ZLG7289_Download(unsigned char mod, char x, bit dp, char dat)

```
{
    code char ModDat[3] = {0x80,0xC8,0x90};
    char d1;
    char d2;
    if ( mod > 2 ) mod = 2;
    d1 = ModDat[mod];
    x &= 0x07;
    d1 |= x;
    d2 = dat & 0x7F;
    if ( dp ) d2 |= 0x80;
    ZLG7289_cmd_dat(d1,d2);
}
```

/*

功能 : 执行 ZLG7289 键盘命令

返回 :

返回读到的按键值 : 0 ~ 63

如果返回 0xFF 则表示没有键按下

*/

char ZLG7289_Key() reentrant

```
{
    char key;
    ZLG7289_pinCS = 0;
    ZLG7289_LongDelay();
    ZLG7289_SPI_Write(0x15);
    ZLG7289_LongDelay();
    key = ZLG7289_SPI_Read();
    ZLG7289_pinCS = 1;
    ZLG7289_LongDelay();
    return key;
}
```

/*

函数 : ZLG7289_Init()

功能 : ZLG7289 初始化

参数 :

t : SPI 总线的延时值设置, 取值 1-40 (超出范围可能导致错误)
说明:
t 的取值可以参照以下公式: $t \geq 5 * f1 / f2$, 其中 f1 表示 80C51 的晶振频率, f2 表示 ZLG7289 的晶振频率

```
*/  
void ZLG7289_Init(unsigned char t)  
{  
    unsigned char x;  
    //I/O 口初始化  
    ZLG7289_pinCS = 1;  
    ZLG7289_pinCLK = 0;  
    ZLG7289_pinDIO = 1;  
    ZLG7289_pinINT = 1;  
    //延时初始化  
    ZLG7289_Delay_t = t;  
    //点亮所有数码管  
    for ( x=0; x<8; x++ )  
    {  
        ZLG7289_Download(1,x,1,8);  
    }  
}
```

第9章 C51 演示程序

9.1 演示程序说明

ZLG7289B 的 C51 演示程序共有两个,“ZLG7289Demo1.c”和“ZLG7289Demo2.c”。这两个程序都是以第 3 章的典型应用电路图(图 3.1)为基础的,首先详细演示了 ZLG7289B 的各项数码管显示功能,最后清除所有显示,并等待某个键按下并将按键值显示出来。这两个演示程序的数码管显示部分是相同的,区别在于键盘演示部分处理按键中断请求的方式不同,前者通过查询 $\overline{\text{INT}}$ 引脚来判断是否有键按下,后者采用负边沿中断方式处理。这两个演示程序都将用到第 8 章中 ZLG7289B 的 C51 驱动程序软件包,并且已经在 ZLG7289B 的 Test 电路板上成功运行。

9.2 演示程序 1

程序清单 9.1 ZLG7289B 演示程序 1

```
/*  
    ZLG7289Demo1.c  
    ZLG7289 演示程序  
说明:  
    采用查询 INT 引脚状态的方式读取键盘值  
*/  
  
#include "ZLG7289.h"  
  
/*  
函数: Delay()  
功能: 延时 0.01 ~ 655.36s  
参数:  
    t>0 时, 延时(t*0.01)s  
    t=0 时, 延时 655.36s  
说明:  
    晶振采用 11.0592MHz  
*/  
void Delay(unsigned int t)  
{  
    do  
    {  
        TH0 = 0xDC;  
        TL0 = 0x00;  
        TR0 = 1;  
        while ( !TF0 );  
        TF0 = 0;  
        TR0 = 0;
```

```

    } while (--t);
}

void main()
{
    code char DispDat[16] =
    { //字母 AbCdEFgHIJkLoPqs 的字形数据
        0x77,0x1F,0x4E,0x3D,0x4F,0x47,0x7B,0x37,
        0x06,0x38,0x07,0x0E,0x1D,0x67,0x73,0x5B
    };
    unsigned char mod;
    unsigned char x;
    bit dp;
    unsigned char dat;
    unsigned char k;
    unsigned char n;
    TMOD = 0x01;
    Delay(30);           //延时 300ms , 等待 ZLG7289 复位完毕
    ZLG7289_Init(4);     //调用 ZLG7289 的初始化函数
    Delay(100);

    //测试
    ZLG7289_Test();
    Delay(200);

    //复位
    ZLG7289_Reset();
    Delay(100);

    //用译码方式 0 和方式 1 下载数据 , 并显示
    for ( mod=0; mod<2; mod++ )
    {
        for ( dat=0; dat<16; dat++ )
        {
            dp = 0;
            for ( x=0; x<8; x++ )
            {
                ZLG7289_Download(mod,x,dp,dat);
            }
            Delay(80);
            dp = 1;
            for ( x=0; x<8; x++ )
            {
                ZLG7289_Download(mod,x,dp,dat);
            }
            Delay(90);
        }
    }
}

```

```
    }
    Delay(100);
}

//用不译码的方式显示自己定义的数据
for ( n=0; n<16; n++ )
{
    dp = 0;
    dat = DispDat[n];
    for ( x=0; x<8; x++ )
    {
        ZLG7289_Download(2,x,dp,dat);
    }
    Delay(80);
}
Delay(100);

//显示 0.1234567
ZLG7289_Download(1,0,1,0);
for ( x=1; x<8; x++ )
{
    dat = x;
    ZLG7289_Download(1,x,0,dat);
}
Delay(100);

//循环左移
for ( n=0; n<8; n++ )
{
    ZLG7289_ROL();
    Delay(120);
}
Delay(100);

//循环右移
for ( n=0; n<8; n++ )
{
    ZLG7289_ROR();
    Delay(120);
}

//左移 2 位
for ( n=0; n<2; n++ )
{
    ZLG7289_SHL();
    Delay(150);
}

//右移 3 位
for ( n=0; n<3; n++ )
```

```

    {
        ZLG7289_SHR();
        Delay(150);
    }
//显示 01234567
for ( x=0; x<8; x++ )
{
    dat = x;
    ZLG7289_Download(1,x,0,dat);
}
Delay(100);
//闪烁控制
dat = 0xFE;
for ( n=0; n<8; n++ )           //逐位闪烁
{
    ZLG7289_Flash(dat);
    dat <<= 1; dat++;           //数据 0xFE 循环左移
    Delay(150);
}
ZLG7289_Flash(0x5A);           //多位可以同时闪烁
Delay(200);
ZLG7289_Flash(0xFF);           //停止闪烁
Delay(100);
//消隐控制
dat = 0xFE;
for ( n=0; n<8; n++ )
{
    ZLG7289_Hide(dat);
    dat <<= 1; dat++;           //数据 0xFE 循环左移
    Delay(150);
}
ZLG7289_Hide(0x5A);           //多位可以同时消隐
Delay(200);
ZLG7289_Hide(0xFF);           //停止消隐
Delay(100);
//复位
ZLG7289_Reset();
Delay(100);
//段点亮
for ( n=0; n<64; n++ )
{
    dat = n;
    ZLG7289_SegOn(dat);
    Delay(30);
}

```

```
    }  
//段关闭  
    for ( n=0; n<64; n++ )  
    {  
        dat = n;  
        ZLG7289_SegOff(dat);  
        Delay(30);  
    }  
//键盘测试  
    ZLG7289_Reset();  
    for (;;)   
    {  
        if ( ZLG7289_pinINT == 0 )    //有键按下  
        {  
            //读取按键值  
            k = ZLG7289_Key();  
            //显示按键值  
            dat = k / 10;  
            ZLG7289_Download(1,0,0,dat);  
            dat = k - dat * 10;  
            ZLG7289_Download(1,1,0,dat);  
            //等待按键抬起  
            while ( !ZLG7289_pinINT );  
        }  
        Delay(5);  
    }  
}
```

9.3 演示程序 2

程序清单 9.2 ZLG7289B 演示程序 2

```
/*  
    ZLG7289Demo2.c  
    ZLG7289 演示程序  
说明：  
    采用中断方式读取按键值  
*/  
  
#include "ZLG7289.h"  
  
//定义全局变量 Key , 用来保存按键值  
volatile unsigned char Key = 0xFF;    //0xFF 表示未按键的状态
```

```
/*
功能：ZLG7289 键盘中断服务程序
参数：
    读到的键盘值放在全局变量 Key 中
说明：
    中断触发方式要设置成负边沿触发
*/
void INTO_SVC() interrupt 0
{
    Key = ZLG7289_Key();
}

/*
函数：Delay()
功能：延时 0.01 ~ 655.36s
参数：
    t>0 时，延时(t*0.01)s
    t=0 时，延时 655.36s
说明：
    晶振采用 11.0592MHz
*/
void Delay(unsigned int t)
{
    do
    {
        TH0 = 0xDC;
        TL0 = 0x00;
        TR0 = 1;
        while ( !TF0 );
        TF0 = 0;
        TR0 = 0;
    } while (--t);
}

void main()
{
    code char DispDat[16] =
    { //字母 AbCdEFgHIJkLoPqs 的字形数据
        0x77,0x1F,0x4E,0x3D,0x4F,0x47,0x7B,0x37,
        0x06,0x38,0x07,0x0E,0x1D,0x67,0x73,0x5B
```

```

};
unsigned char mod;
unsigned char x;
bit dp;
unsigned char dat;
unsigned char k;
unsigned char n;
TMOD = 0x01;
Delay(30);          //延时 300ms , 等待 ZLG7289 复位完毕
ZLG7289_Init(4);    //调用 ZLG7289 的初始化函数
Delay(100);

//测试
ZLG7289_Test();
Delay(200);

//复位
ZLG7289_Reset();
Delay(100);

//用译码方式 0 和方式 1 下载数据 , 并显示
for ( mod=0; mod<2; mod++ )
{
    for ( dat=0; dat<16; dat++ )
    {
        dp = 0;
        for ( x=0; x<8; x++ )
        {
            ZLG7289_Download(mod,x,dp,dat);
        }
        Delay(80);
        dp = 1;
        for ( x=0; x<8; x++ )
        {
            ZLG7289_Download(mod,x,dp,dat);
        }
        Delay(90);
    }
    Delay(100);
}

//用不译码的方式显示自己定义的数据
for ( n=0; n<16; n++ )
{
    dp = 0;
    dat = DispDat[n];
    for ( x=0; x<8; x++ )
    {

```

```
        ZLG7289_Download(2,x,dp,dat);
    }
    Delay(80);
}
Delay(100);
//显示 0.1234567
ZLG7289_Download(1,0,1,0);
for ( x=1; x<8; x++ )
{
    dat = x;
    ZLG7289_Download(1,x,0,dat);
}
Delay(100);
//循环左移
for ( n=0; n<8; n++ )
{
    ZLG7289_ROL();
    Delay(120);
}
Delay(100);
//循环右移
for ( n=0; n<8; n++ )
{
    ZLG7289_ROR();
    Delay(120);
}
//左移 2 位
for ( n=0; n<2; n++ )
{
    ZLG7289_SHL();
    Delay(150);
}
//右移 3 位
for ( n=0; n<3; n++ )
{
    ZLG7289_SHR();
    Delay(150);
}
//显示 01234567
for ( x=0; x<8; x++ )
{
    dat = x;
    ZLG7289_Download(1,x,0,dat);
}
```

```

    Delay(100);
//闪烁控制
    dat = 0xFE;
    for ( n=0; n<8; n++ )           //逐位闪烁
    {
        ZLG7289_Flash(dat);
        dat <<= 1; dat++;           //数据 0xFE 循环左移
        Delay(150);
    }
    ZLG7289_Flash(0x5A);           //多位可以同时闪烁
    Delay(200);
    ZLG7289_Flash(0xFF);           //停止闪烁
    Delay(100);
//消隐控制
    dat = 0xFE;
    for ( n=0; n<8; n++ )
    {
        ZLG7289_Hide(dat);
        dat <<= 1; dat++;           //数据 0xFE 循环左移
        Delay(150);
    }
    ZLG7289_Hide(0x5A);           //多位可以同时消隐
    Delay(200);
    ZLG7289_Hide(0xFF);           //停止消隐
    Delay(100);
//复位
    ZLG7289_Reset();
    Delay(100);
//段点亮
    for ( n=0; n<64; n++ )
    {
        dat = n;
        ZLG7289_SegOn(dat);
        Delay(30);
    }
//段关闭
    for ( n=0; n<64; n++ )
    {
        dat = n;
        ZLG7289_SegOff(dat);
        Delay(30);
    }
//键盘测试
    EA = 0;

```

```
IT0 = 1;    //负边沿触发中断
EX0 = 1;    //允许外部中断
EA = 1;
ZLG7289_Reset();
for (;;)
{
    k = Key;          //Key 的值复制到临时变量 k 中
    Key = 0xFF;       //Key 恢复为无按键状态
    if ( k != 0xFF )   //通过临时变量 k 判断是否有键按下，有则显示出来
    {
        dat = k / 10;
        ZLG7289_Download(1,0,0,dat);
        dat = k - dat * 10;
        ZLG7289_Download(1,1,0,dat);
    }
    Delay(5);
}
```

第10章 参考文献

- [1]周立功单片机. ZLG7289B 串行接口 LED 数码管及键盘管理器件数据手册.<http://www.zlgmcu.com>
- [2]马忠梅,戚军,刘滨,马岩.单片机 C 语言 Windows 环境编程宝典.北京航空航天大学出版社.2003