

液晶显示模块使用手册

型号: LX12864L-1

版本: 2.0

地址:	深圳市宝安区西乡三围航空路 30 号 B 栋 4 楼
电话:	+86(755)29582963
传真:	+86(755)22144273
技术:	18948703963
Email:	lcd@lcdlcd.cn
网站:	www.lcdlcd.cn

版本变更历史记录

版本	修订日期	修改内容	修订人
1.0	2013-11-1	初版发行	Huang
2.0	2022-11-24	修正规格书内容	he

一、LCD 基本参数

1. 产品简介:

我司所生产 LX-12864L-1 型液晶模块由于小巧轻便、使用方便、显示清晰, 广泛应用于各种人机交流面板。

此款可以显示 128 列*64 行

点阵单色图片, 或显示 8 个/行*4 行 16*16 点阵的汉字, 或显示 16 个/行*8 行 8*8 点阵的英文、数字、符号。输入指令强, 可组合成各种输入、显示、位移方式以满足不同的要求

可广泛应用于各种仪器仪表、PM2.5 检测仪, POS 刷卡机, 考勤系统、门禁系统等。

2. 模块的特性:

2.1. 产品薄、轻、结构牢、FPC、插接工艺。

2.2. COG 工艺, IC 采用 ST7567, 功能强大, 稳定性好。

2.3. 显示内容:

●128*64 点阵单色图片;

●可选用 16*16 点阵或其他点阵的图片来自编汉字, 按照 16*16 点阵汉字来计算可显示 8 字/行*4 行。

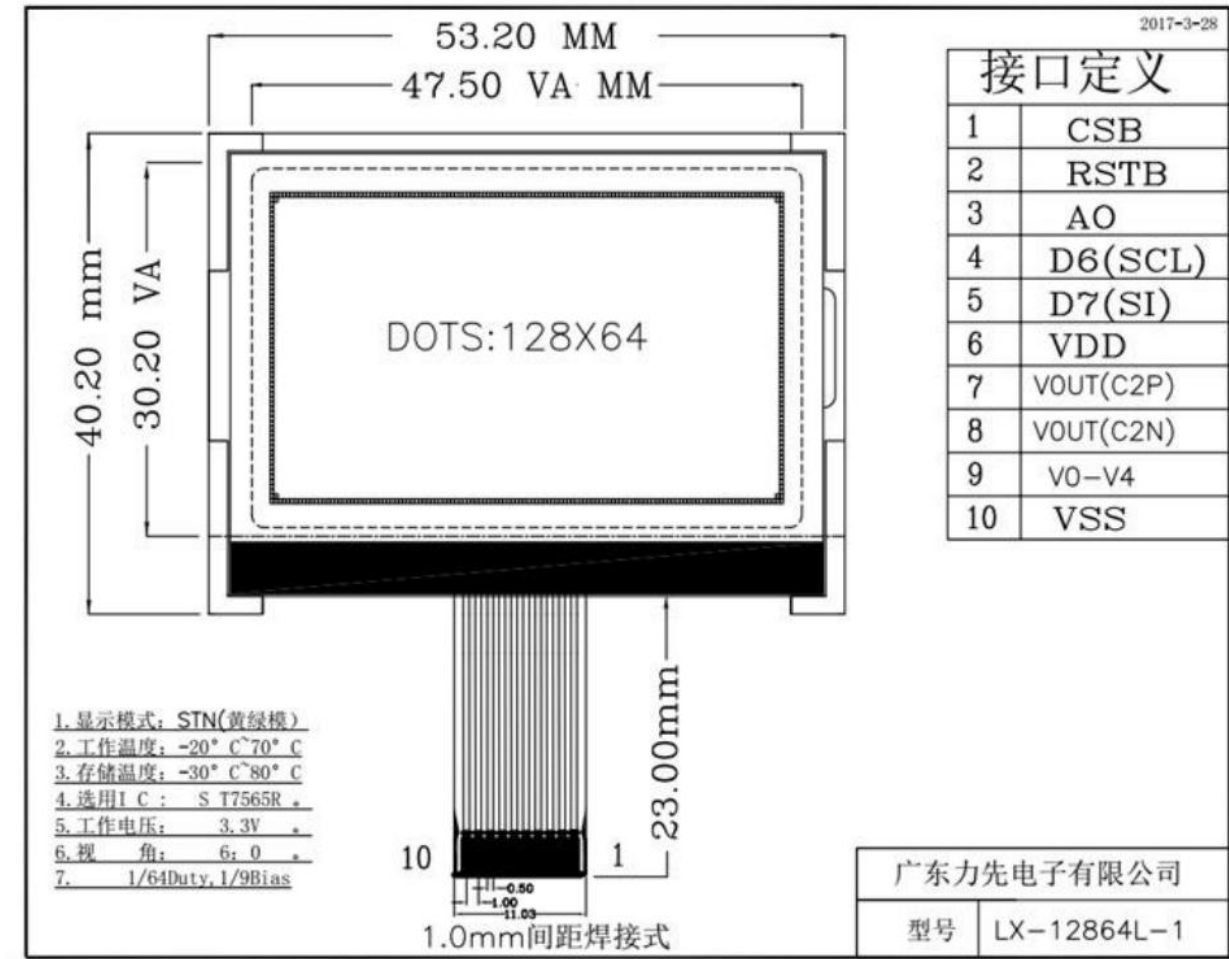
2.4. 指令功能强: 可组合成各种输入、显示、移位方式以满足不同的要求;

2.5. 接口简单方便: 串行接口。

3. 显示屏基本参数:

项目	规格描述	单位
显示模式	STN/黄绿底黑字/正显半透	---
LCD 偏压比	1/64duty, 1/9bias	---
逻辑电源 (VDD)	3.3	V
视角	6	o'clock
外形尺寸	49.6×36.8×32.5	mm
V A 区域	46×30	mm
A A 区域	42.93×26	mm
驱动点阵数	128 × 64	dots
工作温度	-10 ~ +60	°C
储存温度	-20 ~ +70	°C

4. 显示屏尺寸图:



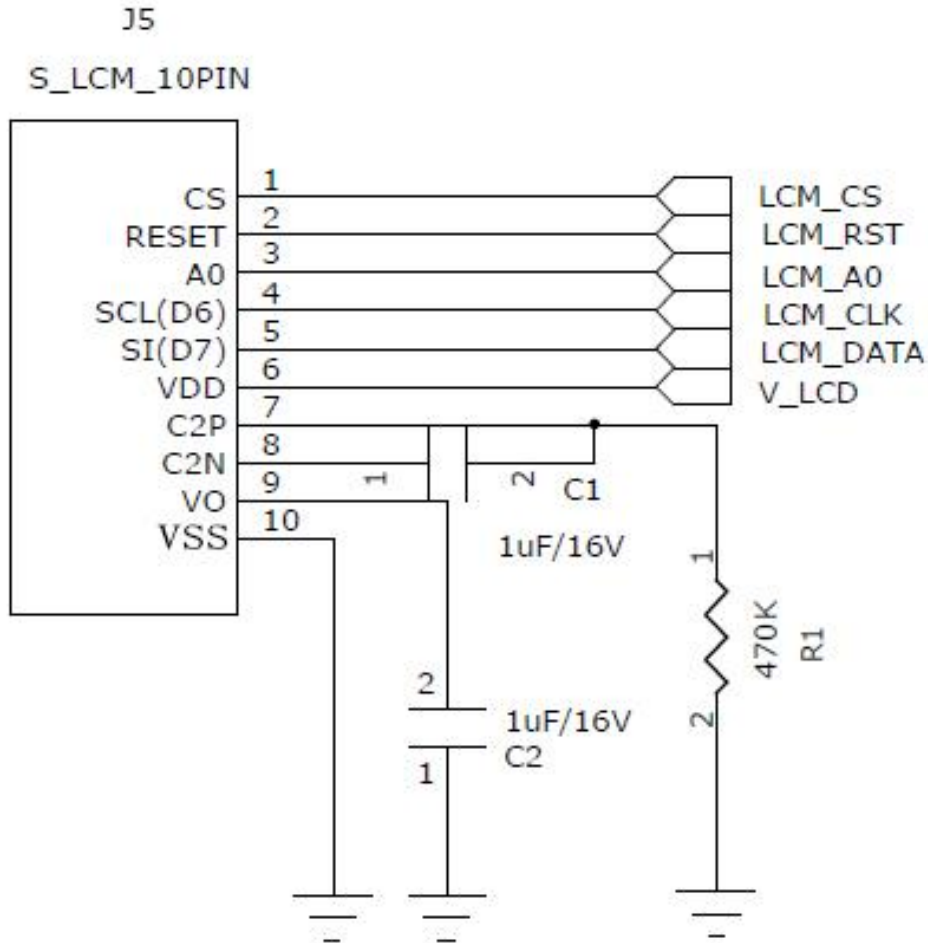
5. 显示屏接口定义:

5.1

序号	名称		功能
1	CS	H/L	片选输入引脚, 低电平使能
2	RES	H/L	硬件复位输入引脚, 当为低电平时, 电路复位
3	A0	H/L	指令和数据选择端口 A0=H: 为显示数据 A0=L: 为控制指令
4	D6 (SCL)		串行时钟输入
5	D7 (SDA)		串行数据输入
6	VDD	---	3.3V
7	VO	---	倍压电路, 两线之间接电容. (105P)
8	XVO	---	

9	V0-V4		LCD倍压输出, 与VSS这间接一个电容 (105P)
10	VSS	---	接地

5.2 外围连接图:



二、芯片参数

6. 技术参数

6.1 极限参数

除非另有规定, $T_{amb}=25^{\circ}C$, $VSS=0V$

参数名称	符号	额定值	单位
数字电源电压	VDD	-0.3 ~ +3.6	V
模拟电源电压	VDD2	-0.3 ~ +3.6	V
LCD 电源电压	VO _{UT} 、V ₀	-0.3 ~ +13.5	V
LCD 偏置电压	V ₁ 、V ₂ 、V ₃ 、V ₄	-0.3 ~ V ₀	V
逻辑输入电压	V _{IN}	-0.3 ~ V _{DD} + 0.3	V
工作环境温度	T _{amb}	-10~+60	°C
贮存温度	T _{stg}	-20~+70	°C

注: 1、V₀, VDD2, V_G, V_M, VSS 和 X_{V0} 的匹配关系: V₀ ≥ VDD2 > V_G > V_M > VSS ≥ X_{V0}

6.2 直流参数 1

除非另有规定, T_a=-30°C~+80°C, VSS_c=0V

参数名称	符号	测试条件	规范值			单位	对应端口	
			最小	典型	最大			
工作电压 (1)	VDD ₁		1.7	-	3.3	V	VDD1	
工作电压 (2)	VDD ₂		2.4	-	3.3	V	VDD2	
工作电压 (3)	VDD ₃		2.4	-	3.3	V	VDD3	
输入高电平电压	V _{IHC}		0.7VDD ₁	-	VDD1	V	MPU 接口	
输入低电平电压	V _{ILC}		VSS1	-	0.3VDD ₁	V	MPU 接口	
输出高电平电压	V _{OHC}	I _{OUT} =1mA, VDD1=1.8V	0.8VDD ₁	-	VDD1	V	D[7: 0]	
输出低电平电压	V _{OLC}	I _{OUT} =-1mA, VDD1=1.8V	VSS1	-	0.2VDD ₁	V	D[7: 0]	
输入漏电流	I _{LI}		-1.0	-	1.0	μA	MPU 接口	
输出漏电流	I _{LO}		-3.0	-	3.0	μA	MPU 接口	
液晶驱动导通电阻	R _{ON}	T _a =25°C	V _{OP} =8.5V, ΔV=0.85V	-	0.6	0.8	KΩ	COMX
			V _G =1.9V, ΔV=0.19V	-	1.3	1.5	KΩ	SEGX
帧频	FR	Duty=1/65, OP=8.5V, T _a =25°C	70	75	80	Hz		

6.2.1 直流参数 2

电流损耗：输出显示，内部电源工作，整个裸芯片的电流损耗

工作状态	符号	测试条件	规范值			单位
			最小	典型	最大	
显示： SNOW（静态）	ISS	VDD1=VDD2=VDD3=3.0V, 倍压 X5 , VOP=8.5 V, Bias=1/9, Ta=25°C	-	150	300	μ A
显示关	ISS	VDD1=VDD2=VDD3=3.0V, 倍压 X5 , VOP=8.5 V, Bias=1/9, Ta=25°C	-	95	190	μ A
掉电	ISS	VDD1=VDD2=VDD3=3.0V, Ta=25°C	-	8	16	μ A

7.读写时序特性

串行 4 线接口时序参数

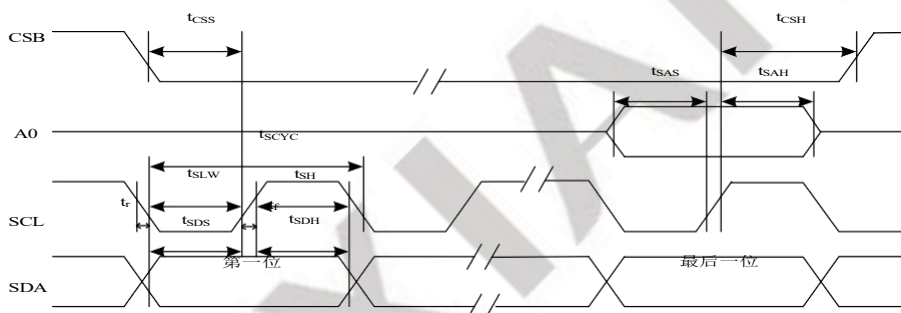


图 6、交流参数 3

(VDD1 = 3.3V , Ta =25°C)

参数名称	对应 端口	符号	测试条件	规范值		单位
				最小	最大	
串行时钟周期	SCLK	tSCY C		50	—	ns
SCLK H 脉冲宽度		tSHW		25	—	
SCLK L 脉冲宽度		tSLW		25	—	
地址建立时间	A 0	tSAS		20	—	
地址保持时间		tSAH		10	—	
数据建立时间	SDA	tSDS		20	—	
数据保持时间		tSDH		10	—	
CSB 到 SCLK 时 间	CSB	tCSS		20	—	
CSB 到 SCLK 时间		tCSH		40	—	

(VDD1 = 2.8V , Ta =25°C)

参数名称	对应 端口	符号	测试条件	规范值		单位
				最小	最大	

串行时钟周期	SCLK	tSCY C	100	—	ns
SCLK H 脉冲宽度		tSHW	50	—	
SCLK L 脉冲宽度		tSLW	50	—	
地址建立时间	A 0	tSAS	30	—	
地址保持时间		tSAH	20	—	
数据建立时间	SDA	tSDS	30	—	
数据保持时间		tSDH	20	—	
CSB 到 SCLK 时间	CSB	tCSS	30	—	
CSB 到 SCLK 时间		tCSH	60	—	

(VDD1 = 1.8V, Ta = 25°C)

参数名称	对应端口	符号	测试条件	规范值		单位
				最小	最大	
串行时钟周期	SCLK	tSCY C		200	—	ns
SCLK H 脉冲宽度		tSHW		80	—	
SCLK L 脉冲宽度		tSLW		80	—	
地址建立时间	A 0	tSAS		60	—	
地址保持时间		tSAH		30	—	
数据建立时间	SDA	tSDS		60	—	
数据保持时间		tSDH		30	—	
CSB 到 SCLK 时间	CSB	tCSS		40	—	
CSB 到 SCLK 时间		tCSH		100	—	

注：1、输入信号的上升下降时间(t_r , t_f)要 ≤ 15 ns。

2、所有时序测试的参考电压为 20% VDD1 到 80% VDD1。

7.1 硬件复位时序参数

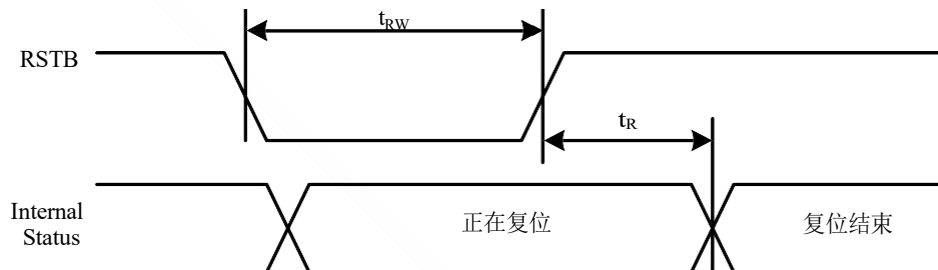


图 7、交流参数 4

(VDD1 = 3.3V , Ta =25°C)

参数名称	符号	测试条件	规范值		单位
			最小	最大	
复位时间	tR		—	1.0	us
RESET L 脉冲宽度	tRW		1.0	—	

(VDD1 = 2.8V , Ta =25°C)

参数名称	符号	测试条件	规范值		单位
			最小	最大	
复位时间	tR		—	2.0	us
RESET L 脉冲宽度	tRW		2.0	—	

(VDD1 = 1.8V , Ta =25°C)

参数名称	符号	测试条件	规范值		单位
			最小	最大	
复位时间	tR		—	3.0	us
RESET L 脉冲宽度	tRW		3.0	—	

7.2

SPI4 线串行通讯 (PSB 为高电平, C86 为高电平或低电平)

设置串行接口

通讯模式	PS B	C86	CS B	A0	ER D	RWR	D[7:0]
SPI 4 线串行通讯	L	X	CS B	A0	--	--	SDA,SCLK,--,--,--,--,--,--

注: 1、被标注为“--”的引脚必须要短接到高电位, VDD1、VDDH 2、C86 被标注为“×”, 可以接高电位也可以接低位

当 CSB 为低电平时, 芯片可以进行通信, 串行数据 (SDA) 和串行时钟 (SCLK) 开始工作。当 CSB 为高电平时, ST7567 无法进行通信, 内部的 8 位移位寄存器和 3 位的计数器被复位。当电路处于串行模式时, SDA 上面的数据在 SCLK 的上升沿被存储在移位寄存器中, 在第八个时钟的上升沿时将 A0 端口的信号存储, 同时产生一个脉冲信号, 将串行数据转换为并行数据, 之后数据的处理与并行信号完全一致。在 DDRAM 存取每个字节之后, DDRAM column 地址指针会自动加一。(注意) SCLK 的抗干扰性是非常重要的, 外部的噪声会导致其有异常的数据或命令出现。

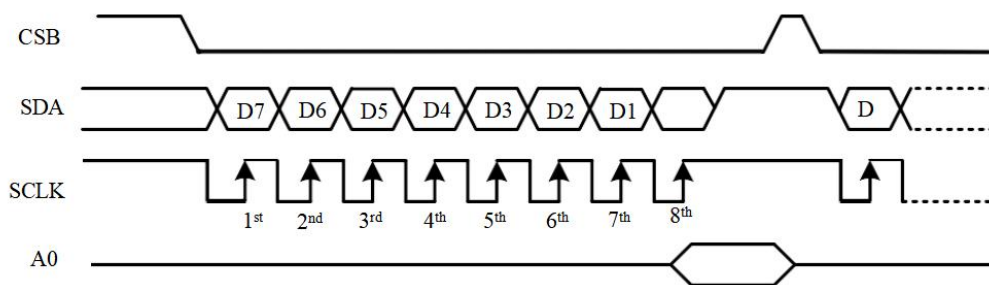


图8、4 线SPI 存取

注: 1、当处于省电模式或硬件复位后, 一些微处理器往往会处于高阻抗状态。而当芯片的 VDD1 端导通时, 这是不允许的, 因为这会导致电路的浮空输入端出现异常状态。

7.3、数据传输

ST7567 使用总线锁存和内部数据总线进行接口数据传输。在从 MPU 向 DDRAM 写数据时, 数据自动从总线锁存传输至 DDRAM, 如图 4 当从片内 DDRAM 读数据到 MPU 时, 第一个读周期读取总线锁存的内容(空读), 在下一个读周期才输出 MPU 应该读取的数据, 如图 5, 这表示, 设置完目标地址后, 接下来的读操作之前需要有一个空闲的读周期。因此, 一些要求精确的数据无法在设置完目标地址的第一个读周期读取, 但是可以在第二个读周期读取。

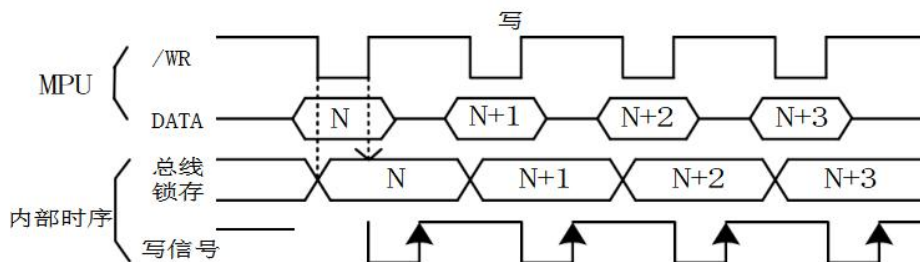


图9、数据传输：写

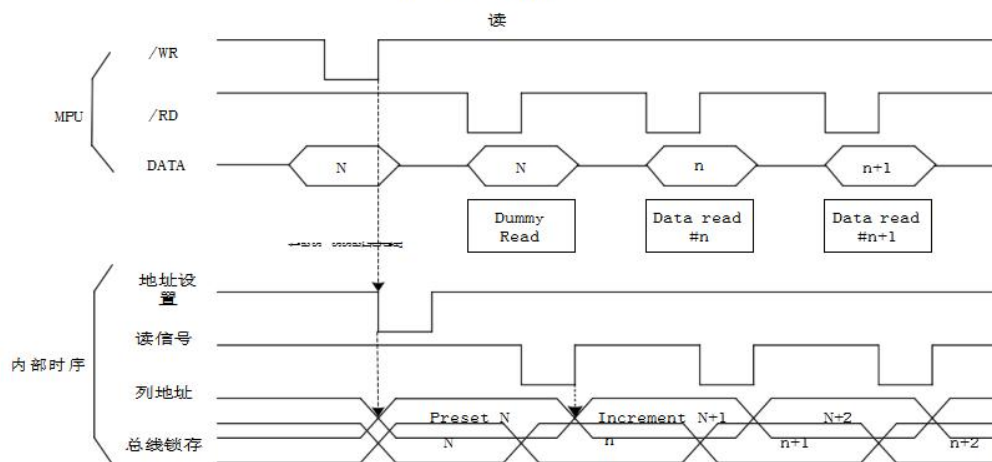


图10、数据传输：读

7.4、显示数据

ST7567 电路内建有 65*132bit 的 DDRAM 用于存储显示数据，显示数据 RAM (DDRAM) 存储了 LCD 的点数据，可以通过设定 132 列和 65 行来控制显示。DDRAM 与 LCD 屏及通讯地址的对应关系如图 6。当处于 MPU 通讯模式时，DDRAM 被 X、Y 地址分割成 9 行，132 列，每列 8bit 数据，当处于 LCD 显示模式时，DDRAM 被分为 65 行，每行 132bit 数据，其中行又以页来划分，Page0~Page7 每页有 8 行(对应 COM0~63)，Page8 仅有一行(对应 COMS，用作图像显示)显示数据(D7 ~D0)对应 LCD 的 COM 行方向，D0 在首位。除图像页外，其余所有页都可以通过 D7 ~D0 直接存取。图像 RAM 只需使用数据总线的 D0 这一位。见图 7。MPU 可以通过 I/O 总线来进行读写操作。由于 LCD 驱动器可独立操作，数据进行显示时可以同步写入数据，不会导致 LCD 闪烁或者数据冲突。

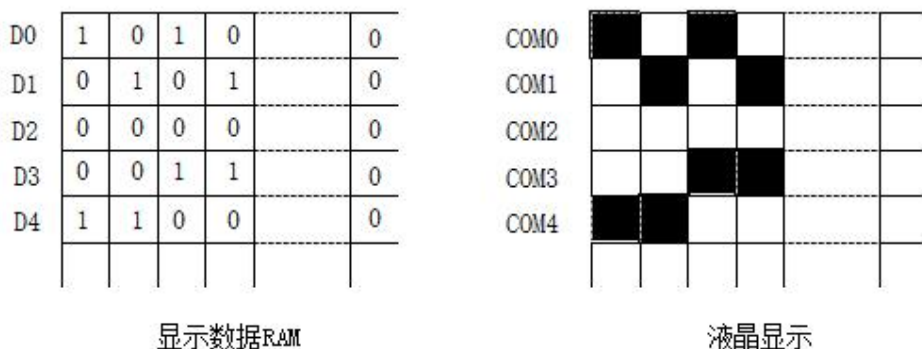


图 11、 DDRAM 的数据与显示屏的对应关系

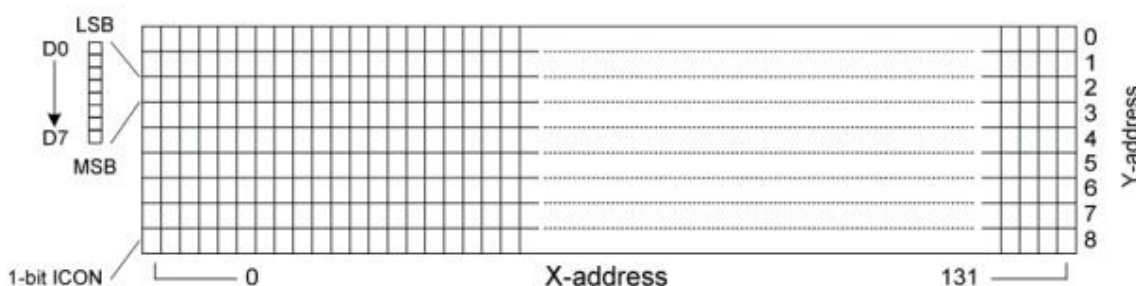


图 12、 读写地址与 DDRAM 的对应关系

注：串行模式下只能写不能读

7.5 、寻址

ST7567 的显示数据 RAM 为132bit*65，地址范围为：X=0~131（列地址）,Y=0~8（页地址）在该范围之外的数据是无效的。

7.6 、页地址电路

此电路由一个 4 位页地址寄存器组成，只能通过“PAGEADDRESSSET”指令进行修改，能提供 DDRAM 的页面数据。页地址必须在存取 DDRAM 内容前进行设置，页地址 8 是一个用于图像显示的特殊 RAM 区，只有一个合法操作位：D0。

7.7 、列地址电路

DDRAM 的列地址是由“COLUMN ADDRESS SET”指令来设置的。列地址在每次显示数据存取（读/写）后会加 1，因此 MPU 可连续存取 DDRAM 的内容，但由于页地址电路和列地址电路是独立的，此特性只能执行至每一页页尾（列地址“83H”）。例如，从（页-0，列-83H）到（页-1，列-0），需要对页地址和列地址均重新赋值来改变 DDRAM 指针。

此外，寄存器 MX 和 MY 可以颠倒 DDRAM 与输出（COM/SEG）的关系，在改变 MX 的设置后，必须重新将显示数据写入到 DDRAM 里。

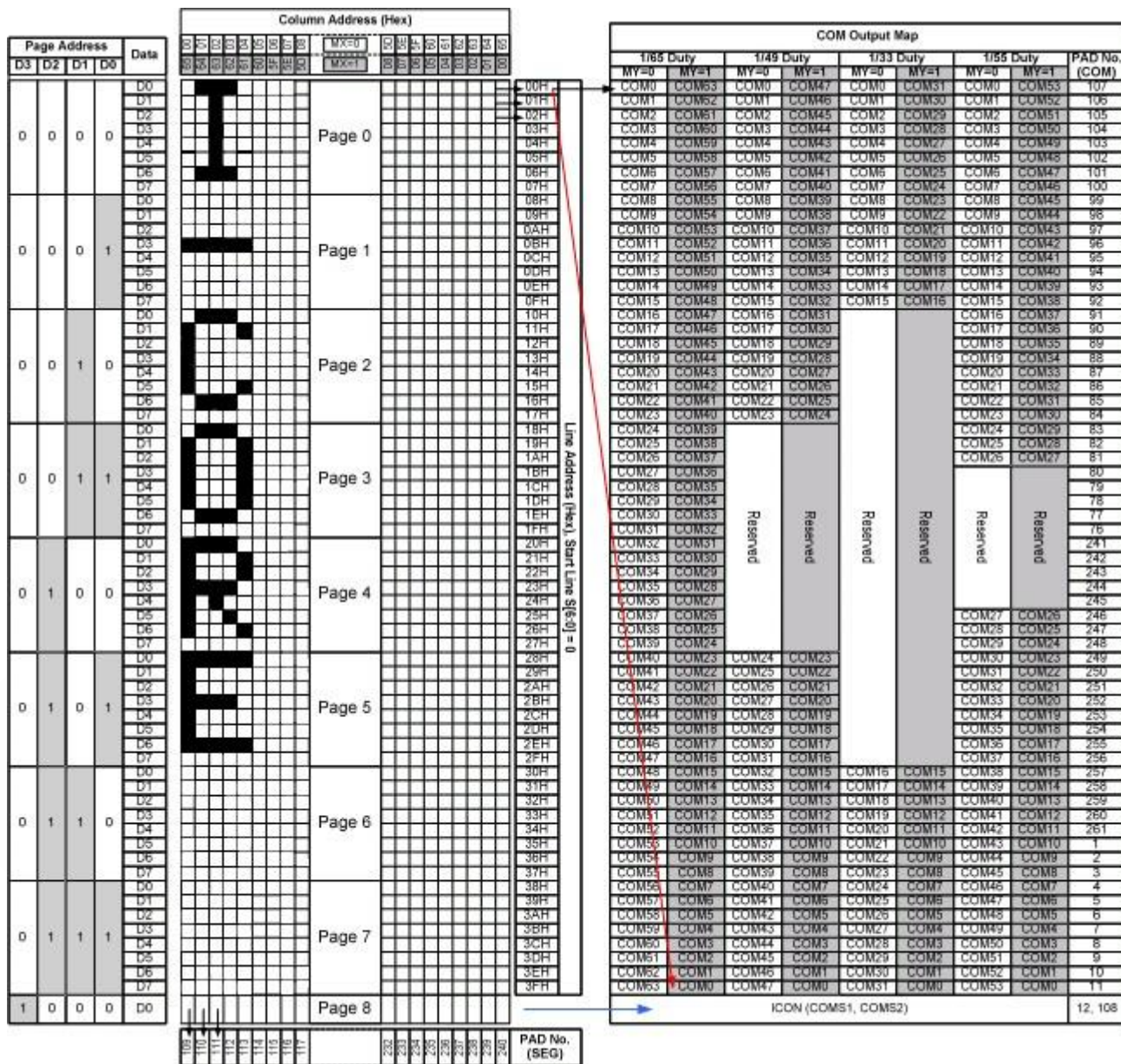


图13、 DDRAM 的数据与 SEG、COM 的对应关系

7.8 、行地址电路

行地址电路由一个计数器和一个行地址寄存器组成，行地址寄存器由“DISPLAY STARTLINESET”指令来设置。此电路赋值给 DDRAM 一个行地址，作为显示的第一行（COM0）。因此，通过重复设置行地址，ST7567 可以不改变 DDRAM 内容来实现屏幕的滚动。如图 9 所示。最后一个行始终是 COMS（用作图像的行输出）即图像不会和通常的显示数据一起滚动。

8.0 、振荡电路

ST7567 内建振荡电路来产生液晶驱动电路需要的系统时钟。在 ST7567 初始化后振荡电路被激活。为降低电源损耗，时钟不会被输出。

8.1 、液晶驱动电源电路

ST7567 内建电源电路来产生驱动液晶的电压。电路采用最少的外围元件以降低电源损耗。内建的电源电路包括电压倍压器、电压调整器和电压跟随电路。在 ST7567 断电前需要一个电源关闭程序（参考操作流程部分）

8.2 、电源电路的外围元件

推荐的电源外围元件只有两个电容。这两个电容的具体值由屏的尺寸和负载决定。

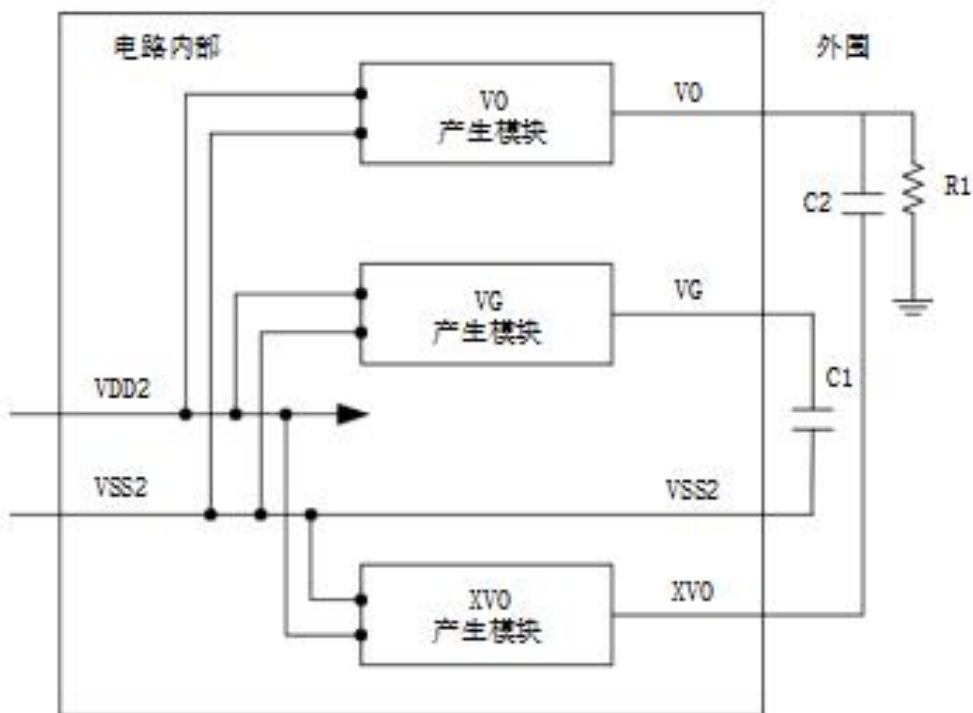


图 15、电源电路

8.3 、调整器电路

ST7567 内建高精度调整器电路，共 8 种调节比率（regulation ratio—RR），每种 RR 有 64 个 EV 电平进行电压调节。无需额外的外围元件，输出电压可以通过“Regulation Ratio”和“Set EV”指令进行改变。指令描述部分有详细的设置方法。

8.4 、复位电路

ST7567 由 RSTB 端口置低对电路内部进行初始化。当 RSTB 置低时除读状态指令有效，其余均无效。操作前需要通过 RSTB 脚对电路进行初始化。硬件复位与软件复位不同，当 RSTB 变为低，硬件复位程序就会启动；当执行 RESET 指令后，软件复位程序就会启动。

寄存器	RSTB 硬件复位值	RESET 软件复位值
显示关闭: D=0, 所有 SEG 和 COM 输出均为低	V	X
正常显示: INV=0, AP=0	V	X
SEG 正常显示	V	X
串行计数器和移位寄存器清零 (若使用了串行接口)	V	X
偏置选择: BS=0	V	X
倍压幅度: BL=0	V	X
退出节电模式	V	X
关闭电源控制: VB=0, VR=0, VF=0	V	X

退出读写修正模式	V	V
起始行: S[5]=0	V	V
行地址: X[7:0]=0	V	V
页地址: Y[3:0]=0	V	V
COM 正常显示方式: MY=0	V	V
V0 调整率: RR[2:0]=(1,0,0)	V	V
EV[5:0]=(1,0,0,0,0,0)	V	V
退出测试模式	V	V

上电后，RAM 数据未定义，显示状态为“显示关”。在显示打开之前最好初始化整个 DDRAM (如：填写全 00h 或写显示图案)。此外，电源刚打开时不稳定，当电源稳定之后需要进行硬件复位对内部寄存器进行初始化。

9. 指令描述

9.1、通用指令表

序号	指令	A0	R/W (RW R)	指令位								描述
				D7	D6	D5	D4	D3	D2	D1	D0	
1	显示开/关 (display on/off)	0	0	1	0	1	0	1	1	1	D	D=1, 显示开 D=0, 显示关
2	设置起始行 (set start line)	0	0	0	1	S5	S4	S3	S2	S1	S0	设置显示的起始行
3	设置页地址 (set pageaddress)	0	0	1	0	1	1	Y3	Y2	Y1	Y0	设置页地址
4	设置列地址 (set column address)	0	0	0	0	0	1	X7	X6	X5	X4	设置列地址高位 (MSB)
		0	0	0	0	0	0	X3	X2	X1	X0	设置列地址地位 (LSB)

5	读状态 (read status)	0	1	0	M X	D	RST	0	0	0	0	读取 IC 的状态
6	写数据 (write data)	1	0	D 7	D 6	D5	D4	D3	D2	D1	D0	对DDRAM 写数据
7	读数据 (read data)	1	1	D 7	D 6	D5	D4	D3	D2	D1	D0	读取 DDRAM 的数据
8	SEG 显示方式 (seg direction)	0	0	1	0	1	0	0	0	0	MX	设置 SEG 的扫描方向 MX=1, 显示左右颠倒 MX=0, 普通显示
9	反显 (inverse display)	0	0	1	0	1	0	0	1	1	IN V	INV=1, 反显 INV=0, 普通显示
10	屏全亮 (all pixel on)	0	0	1	0	1	0	0	1	0	AP	AP=1, 屏全部点亮 AP=0, 普通显示
11	偏置选择 (biasselect)	0	0	1	0	1	0	0	0	1	BS	偏置选择 0=1/9; 1=1/7(1/65 占空比)
12	read-modify-write	0	0	1	1	1	0	0	0	0	0	行地址增量: 读: +0, 写: +1
13	END	0	0	1	1	1	0	1	1	1	0	退出

												read-modify-write 模式
14	复位(RESET)	0	0	1	1	1	0	0	0	1	0	软件复位
15	COM 扫描方式 (com direction)	0	0	1	1	0	0	MY	-	-	-	设置 COM 的扫描方向 MY=1, 上下颠倒 MY=0, 普通显示
16	电源控制 (power control)	0	0	0	0	1	0	1	VB	VR	VF	设置内置电源 管理电路的工作
17	RR 设置 (regulation ratio)	0	0	0	0	1	0	0	RR2	RR 1	RR 0	选择 RR 电阻范围
18	EV 设置(set EV)	0	0	1	0	0	0	0	0	0	1	双行指令设 置 EV 等级
		0	0	0	0	EV 5	EV 4	EV3	EV2	EV 1	EV 0	
19	设置倍压 (set booster)	0	0	1	1	1	1	1	0	0	0	双行指令 设置倍压等级: BL=0: 4 倍 BL=1: 5 倍
		0	0	0	0	0	0	0	0	0	BL	
20	省电模式 (power save)	0	0	复用指令							display off + all pixel on	
21	空操作(nop)	0	0	1	1	1	0	0	0	1	1	不执行操作
22	测试(test)	0	0	1	1	1	1	1	1	1	-	测试指令

注: “-”可接“H”或“L”

9.2 、显示开/关(display on/off)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	0	1	0	1	1	1	D

D=1, 显示开

D=0, 显示关, 所有的 SEG、COM 端口被置为 0 电平

9.3 、设置起始行(set start line)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	1	S5	S4	S3	S2	S1	S0

设置起始行的作用是选择 DDRAM 中被 S[5:0]指定的显示数据在 COM0 上面进行显示, 其余的数据按照地址自加进行循环, 用于设置画面的滚动效果。

S5	S4	S3	S2	S1	S0	显示地址
0	0	0	0	0	0	0
0	0	0	0	0	1	1
0	0	0	0	1	0	2
0	0	0	0	1	1	3
.
1	1	1	1	0	1	61
1	1	1	1	1	0	62
1	1	1	1	1	1	63

9.4、设置页地址(set page address)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	0	1	1	Y3	Y2	Y1	Y0

Y3	Y2	Y1	Y0	页地址	数据的有效位
0	0	0	0	page0	D7~D0
0	0	0	1	page1	D7~D0
0	0	1	0	page2	D7~D0
.
0	1	1	0	page6	D7~D0
0	1	1	1	page7	D7~D0
1	0	0	0	page8 (icon page)	D0

9.5、设置列地址(set column address)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	1	X7	X6	X5	X4
0	0	0	0	0	0	X3	X2	X1	X0

行地址可选择的范围为 0~131，需要采用两条指令才能够完全设置完成。

X7	X6	X5	X4	X3	X2	X1	X0	行地址
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1
:	:	:	:	:	:	:	:	:
1	0	0	0	0	0	0	1	129
1	0	0	0	0	0	1	0	130
1	0	0	0	0	0	1	1	131

9.6、写数据 (write data)

A0	R/W(RWR)	D6	D5	D4	D3	D2	D1	D0
1	0	D6	D5	D4	D3	D2	D1	D0

当地址设置完成之后，MPU 可以连续的对 DDRAM 进行写数据操作，但当一行写完之后，必须重新设置 X、Y 地址才可以进行下一行数据的写操作，否则在 X 地址溢出之后将会覆盖原输入数据。

9.7、SEG 显示方式 (seg direction)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	0	1	0	0	0	0	MX

MX=0: 普通显示模式 (SEG0->SEG131)

MX=1 : 左右颠倒显示方式
(SEG131~SEG0)

9.8、反显 (inverse display)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	0	1	0	0	1	1	INV

INV=0: 普通显示模式

INV=1: 反转显示模式

9.9、屏全亮 (all pixel on)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	0	1	0	0	1	0	AP

AP=0: 普通显示模式

AP=1: 屏全亮显示模式

10、偏置选择 (bias select)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	0	1	0	0	0	1	BS

10.1、复位 (RESET)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	1	1	0	0	0	1	0

执行这条指令之后，电路进入软件复位状态，各寄存器值详见复位状态寄存器表。

10.2、COM 扫描方式 (com direction)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	1	0	0	MY	-	-	-

MY=0: 普通扫描显示模式 (COM0~COM63)

MY=1: 上下颠倒扫描显示模式 (COM63~COM0)

10.3、省电模式 (power save)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	0	1	0	1	1	1	0
0	0	1	0	1	0	0	1	0	1

ST7567 电路的省电模式是通过两条指令联合使用来实现的，第一条指令为设置显示关 (D=0) 第二条指令为设置屏全亮 (AP=1)，之后电路进入省电模式，进入省电模式时电路的工作状态：

- 1、RC 时钟关闭
- 2、内置的电源管理电路关闭
- 3、LCD 的时序发生关闭，所有的 COM、SEG 端口被置为 0 电位



当FD=0 时，电路内部按照如下时序工作

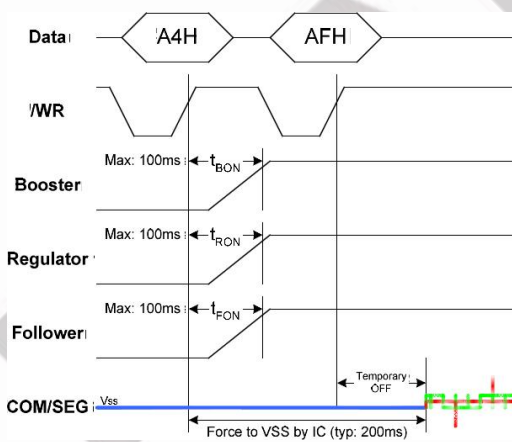


图19、工作时序

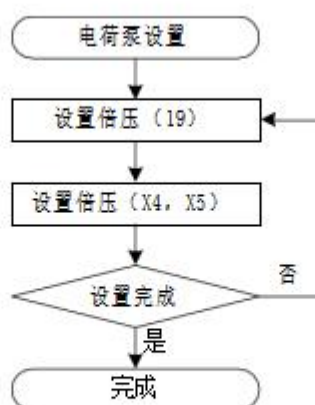
退出省电模式时方向执行上面两条指令，退出省电模式后，电路回复到省电模式前的配置状态

10.4、设置倍压 (set booster)

A0	R/W(RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	BL

BL=0: 4 倍压

BL=1: 5 倍压



11. 、NOP

A0	R/W (RWR)	D7	D6	D5	D4	D3	D2	D1	D0
0	0	1	1	1	0	0	0	1	1

当设置为这条指令时，电路不执行任何操作

11.1、工作时序

11.2、电路上电

工作流程：

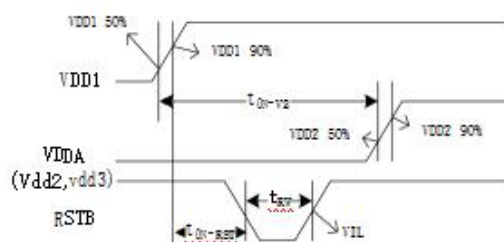
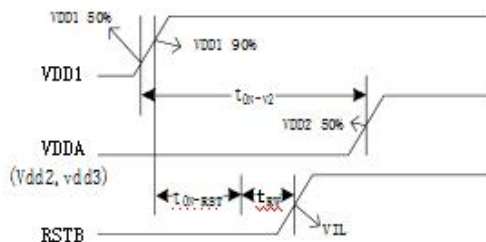
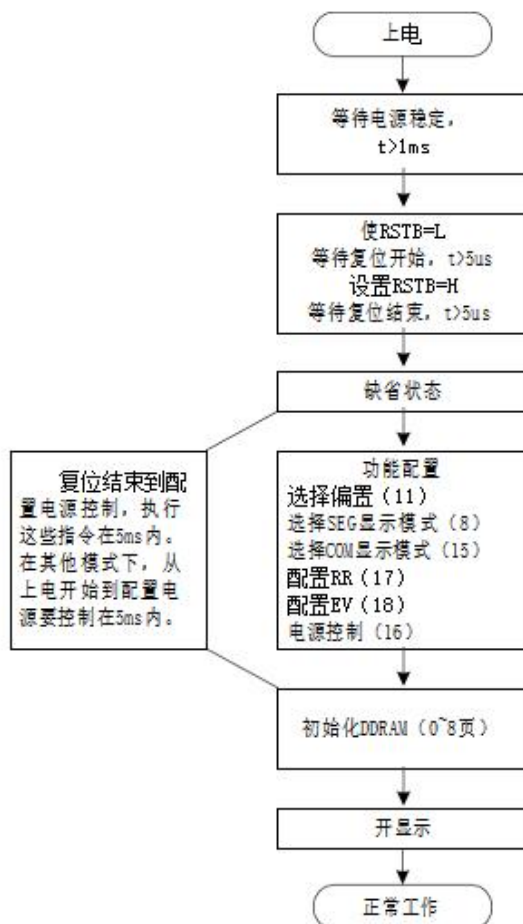


图 21、上电时序

注：下面表格有参数的详细描述

- 1、 t_{RW} 和 t_{R} 请参考时序参数指标
- 2、RESET 请参考4.11.6节说明
- 3、5ms 是为了符合LCD屏的规格和电源部分外接器件的要求。可根据实际使用的器件来检测
- 4、INSTRUCTION 功能的详细描述见4.11节说明
- 5、VDDI 或者 VDDA电压上升到预定值的90%时，被视为电源的稳定态。

11.3、时序要求:

参数	符号	条件	备注
VDDA 电源延时	t_{ON-V2}	$0 \leq t_{ON-V2}$	VDDI 和 VDDA 在任何情况下都不会损坏电路。
RSTB 输入时间	t_{ON-RST}	没有限制	<ul style="list-style-type: none"> 在上电期间, 如果RSTB 为低电平、高电平或者补丁态, RSTB有效的外部复位应该是在VDDI 电压稳定后。 电源电压稳定后, 在任何时候都可以使RSTB置为低电平。 t_{RW} 和 t_{R} 必须符合RSTB的时序要求。 防止损坏显示, 推荐的时序是: $0 \leq t_{ON-RST} \leq 30 \text{ ms.}$

注: 表中给出的时序要求是为了防止损坏LCD模组

11.4、显示数据



图 22、显示数据流程

注: 参考项目

- 1、INSTRUCTION 功能的详细描述见4.11节分说明
- 2、在显示打开之前, 推荐要写入显示数据, 即初始化DDRAM

11.5、刷新

推荐在固定的间隔时间刷新时序

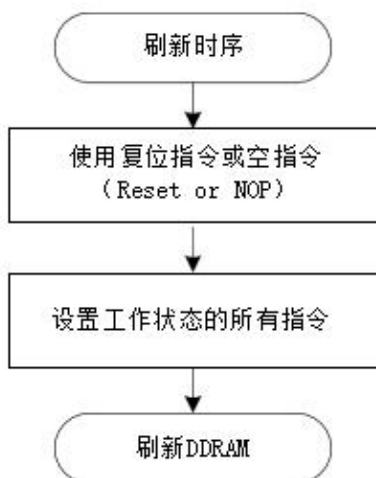


图23、刷新流程

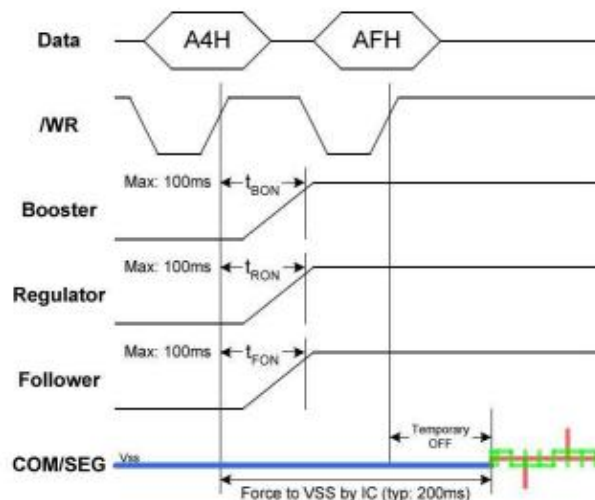


图24、刷新时序

注:

- 1、电源稳定时间取决于加载的LCD屏。
- 2、上图中给出的电源稳定时间的条件是: LCD屏尺寸=1.4", $C1=1\mu F$, $C2=1\mu F$, $VDD=2.7V$, $Vop=9V$ 。

11.6、电路掉电时序及流程

电路在省电模式时, LCD输出端拉到VSS, 模拟输出端处于放电状态, 电源电压关断。下面给出的两种方式可以触发电路进入省电模式。

使用省电模式

掉电流程:

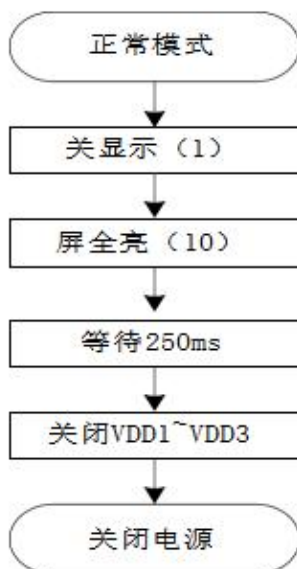


图25、掉电流程

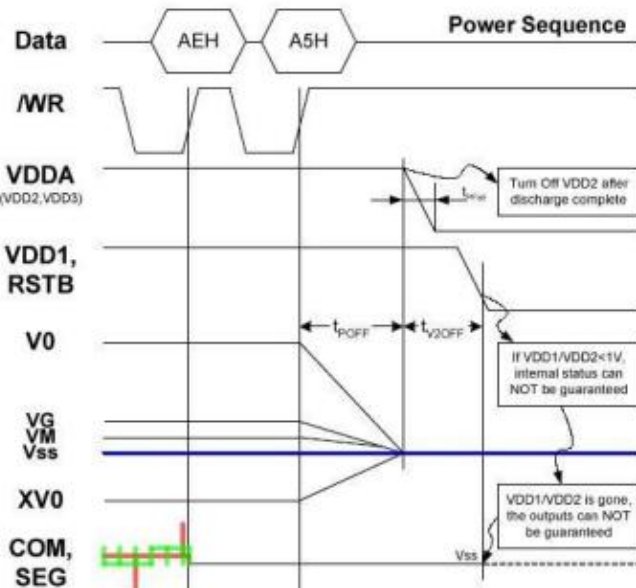


图26、工作时序:

在内置电源电路关断和完全放电之后, VDD1和VDDA电压被移掉。

11.7、硬件复位功能: 掉电流程:

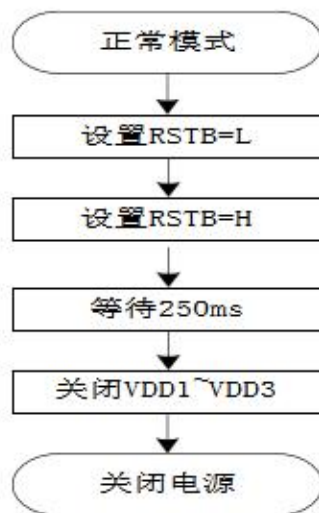


图27、掉电流程

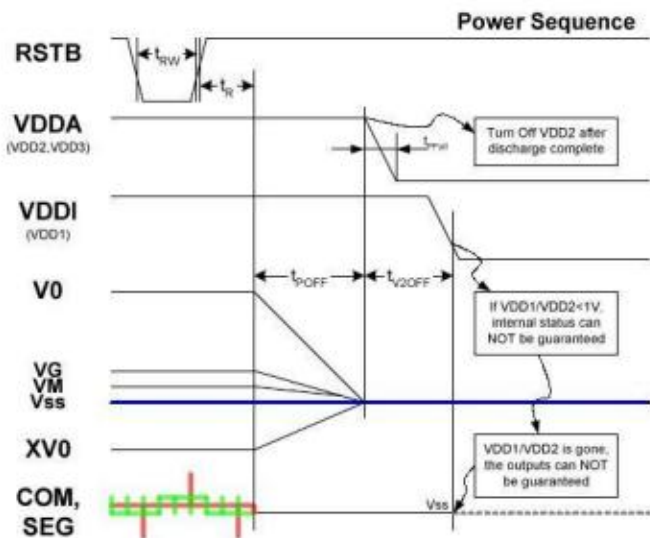


图28、工作时序

在内置电源电路关断和完全放电之后，VDD1和VDDA电压被移除。

注:

- 1、 t_{POFF} : 内部电源放电时间 $\geq 250ms$ (最大)
- 2、 t_{V2OFF} : VDDI 和 VDDA关断时间 $\geq 0ms$ (最小)
- 3、不建议在VDDA关断前,先关断VDDI。关断了VDDI,电路内部状态不稳定,可能会停止放电。未被放掉的电压可能会流入COM/SEG输出端,及极化LCD屏。
- 4、VDDI 和 VDDA 不同时供电,不会损坏电路
- 5、时序与负载屏和外接电容有关
- 6、上图中的时序测试条件: LCD P屏尺寸 = 1.4" , $C1=1\mu F$, $C2=1\mu F$
- 7、VDDA关断时,下降时间要满足如下要求: $20ms \leq t_{PFall} \leq 0.2s$

12、示例参考程序:

```
/**
// project    : 221-3059-1428
// driver IC  : st7567
// LCD        : 1/65 duty, 1/9bias, 8.8V vop
// interface  : SPI
// ver        : 00
// date       : -07
// other      : VDD 3.1v
**/

#include "reg51.h"
/*
    sbit RS  = P3^7;  AO
    sbit RES = P3^3;  复位
    sbit CS  = P3^4;  偏选
    sbit SCL = P1^1;
    sbit SDI = P1^0;
*/
    sbit CS  = P1^0;
    sbit RES = P1^1;
    sbit RS  = P1^2;
    sbit SCL = P1^3;
    sbit SDI = P1^4;

//  sbit SCL = P1^6;
//  sbit SDI = P1^7;
    sbit  key1=P2^1;
    sbit  key2=P2^2;
    sbit  pause=P2^0;
void writec(uchar);
void stop(void);
void writed(uchar);

#define uchar  unsigned char
#define uint   unsigned int
```

```
uchar code chara2[]=  
{  
/* Image size : 128 x 64 pixels */  
/*-- 调入了一幅图像: C:\Users\Administrator\Desktop\诗 1.bmp --*/  
/*-- 宽度 x 高度=128x64 --*/  
0xFF,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,  
0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,  
0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x81,0x41,0x21,0x81,0x41,0x01,0x01,  
0x01,0x01,0x41,0x41,0x41,0x41,0x41,0xE1,0x41,0x41,0x41,0x41,0x01,0x01,0x81,  
0x81,0x81,0x81,0x01,0x01,0x01,0xE1,0x01,0x81,0x81,0x81,0x01,0x01,0x01,0x41,  
0x81,0x21,0x41,0x81,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,  
0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,  
0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0xFF,  
0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x00,0x08,0x14,0x22,0x49,0x94,0x22,0x41,0x80,0x00,0x00,0x00,  
0x00,0x24,0x24,0x15,0xED,0xA5,0xA7,0xA5,0xA5,0xA5,0xED,0x15,0x24,0x24,0x00,0xFF,  
0x88,0x88,0xFF,0x00,0x51,0x51,0xC9,0x4B,0xC4,0x4A,0x50,0x1C,0x00,0x00,0x00,0x00,  
0x80,0x41,0x22,0x94,0x49,0x22,0x14,0x08,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,  
0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x02,0x00,0x01,0x00,0x00,  
0x00,0x00,0x00,0x00,0x07,0x02,0x02,0x02,0x02,0x02,0x07,0x00,0x00,0x00,0x00,0x01,  
0x00,0x00,0x01,0x04,0x04,0x02,0x01,0x00,0x03,0x04,0x04,0x06,0x00,0x00,0x00,0x01,  
0x00,0x02,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,  
0xFF,0x00,0x00,0x00,0x24,0xAC,0xEC,0xEC,0xBC,0xAA,0xAC,0xEC,0xAC,0x24,0x00,0xFE,  
0x52,0xFE,0x00,0xFE,0x52,0x72,0xD2,0x52,0x5E,0x00,0x00,0x82,0x82,0x42,0x32,0xFA,  
0x06,0x12,0x62,0xC2,0x82,0x00,0x00,0x38,0xCA,0x2C,0x28,0xAE,0x28,0x28,0xEE,0x0A,  
0x18,0x00,0xFC,0x24,0xFC,0x80,0xA4,0xA7,0x9C,0xB4,0xCA,0xE2,0x00,0x00,0x00,0x00,  
0x00,0x00,0x00,0x00,0x60,0x3C,0xCA,0x08,0xF8,0x00,0xFE,0x00,0x10,0x60,0x00,0x00,  
0x60,0x3C,0xCA,0x08,0xF8,0x00,0xFE,0x00,0x10,0x60,0x00,0x00,0xF8,0x02,0x00,0xF0,
```



```
};  
void delay1(unsigned int t)  
{  
  while(t>0)  
  {  
    t--;          //TT-  
    pause=1;  
    if(pause==0)stop();  
  }  
}  
void flash(unsigned int t)  
{  
  while(t>0)  
  {  
    t--;          //TT-  
  }  
}  
//-----  
void stop()  
{  
  flash(100);  
  while(pause==0)  
  {  
    pause=1;  
    key1=1;  
    key2=1;  
    if(key1==0)  
    {  
      flash(200);  
      if(key1==0)  
      {  
        while(key1==0);  
        flash(100);  
        if(vop<63)  
        {  
          vop++;  
        }  
      }  
    }  
  }  
}
```

```
        writec(0x81);
        writec(vop);}
    }
}

else if(key2==0)
{
    flash(100);
if(key2==0)
{
    while(key2==0);
    flash(100);
    if(vop>0)
    {
        vop--;
        writec(0x81);
        writec(vop);
    }
}
}
}

void writec(uchar com)
{ unsigned char i ;
  CS=0;
  RS=0;

  for(i=0;i<8;i++)
  { com=com<<1;
    SDI=CX;
    SCL=1;
    SCL=0;
  }
  CS=1;
  RS=1;
}
```

```
void writed(uchar dat)
{
    unsigned char i;
    CS=0;
    RS=1;
    for(i=0;i<8;i++)
    {
        dat=dat<<1;
        SDI=CX;
        SCL=1;
        SCL=0;
    }

    CS=1;
    RS=1;
}
void init ()
{
    uchar col;
    RES=1;
    flash(1000);
    RES=0;
    flash(2000);
    RES=1;
    flash(1000);

    writec(0xe3); // reset signal
    writec(0xa3); //(0xa2 1/9 bias,1/65 duty )
    writec(0xa0); // ADC select
    writec(0xc8); // command output select
    writec(0x2f); // power control
    writec(0x21); // select resistor ratio Rb/Ra
    writec(0x81); // select volume
    writec(55); // vop
    writec(0xf8); // x4
    writec(0x08); // x4
    writec(0xb0);//set page address
```

```
writec(0x10);//set column address
writec(0x00);
for(col=0;col<128;col++)
{
    writed(0x00);
}
writec(0xaf); //display on
}
void display(uchar dat1,uchar dat2)
{
    uchar row,col;

    for (row=0xb0; row<0xb8; row++) //0XB0 0XB8
    {
        writec(row);//set page address
        writec(0x10);//set column address
        writec(0x00);
        for(col=0;col<128;col++)
        {
            writed(dat1);
            writed(dat2);
        }
    }
    delay1(50000);
}
void displaychar(uchar *p)
{
    uchar row,col;
    for (row=0xb0; row<0xb8; row++)
    {
        writec(row);//set page address
        writec(0x10);//set column address
        writec(0x00);
        for(col=0;col<128;col++)
            writed(*p++);
    }
}
```

```
    delay1(50000);  
}  
void main(void)  
{  
    delay1(1000);  
  
    vop=0x29; //vop=9.1V  
//vop=0x15; //vop=7.1V  
    init();  
    while (1)  
    {  
        display(0xff,0xff);  
        display(0x00,0x00);  
        display(0x55,0xaa);  
        display(0xaa,0x55);  
        displaychar(chara1);  
        displaychar(chara2);//vop_test()
```

LIXIAN