



JLX25664G-580-BN 使用说明书

目 录

序号	内 容 标 题	页码
1	概述	2
2	字符型模块的特点	2
3	外形及接口引脚功能	3~5
4	基本原理	5~6
5	技术参数	6~7
6	时序特性	7~11
7	指令功能及硬件接口与编程案例	12~末页

1. 概述

晶联讯电子专注于液晶屏及液晶模块的研发、制造。所生产 JLX25664G-580 型液晶模块由于使用方便、显示清晰，广泛应用于各种人机交流面板。

JLX25664G-580 可以显示不大于 256×64 点阵单色图片，或显示 16 个 $\times 4$ 行=64 个的 16×16 点阵的汉字，或显示 32 个 $\times 4$ 行=128 个的 8×16 点阵的英文、数字、符号。或显示 42 个 $\times 8$ 行的 5×8 点阵的英文、数字、符号。

2. JLX25664G-580 图像型点阵液晶模块的特性

1.1 结构牢: LCD、背光;

1.2 IC 采用 ST75256, 功能强大, 稳定性好

1.3 功耗低: $1 - 100\text{mW}$ (不带背光 $1\text{mW} < 3.3\text{V}@0.3\text{mA}$), 带背光不大于 $100\text{mW} < 3.3\text{V}@30\text{mA}$);

1.4 显示内容:

- 256×64 点阵单色图片。

- 或显示 16 个 $\times 4$ 行=64 个的 16×16 点阵的汉字，按照 24×12 点阵汉字来计算可显示 10 字/行 $\times 5$ 行。

- 或显示 32 个 $\times 4$ 行=128 个的 8×16 点阵的英文、数字、符号。

- 或显示 42 个 $\times 8$ 行的 5×8 点阵的英文、数字、符号。;

- 可选用 16×16 点阵或其他点阵的图片来自编汉字也可配合晶联讯字库 IC (JLX-GB2312) 来显示汉字。

1.5 指令功能强;

1.6 接口简单方便: 可选 I^2C 总线、4 线 SPI 串口、6800 系列并口、8080 系列并口。

1.7 工作温度宽: $-20^\circ\text{C} - 70^\circ\text{C}$;

1.8 可靠性高。

3. 外形尺寸及接口引脚功能

3.1 外形尺寸图

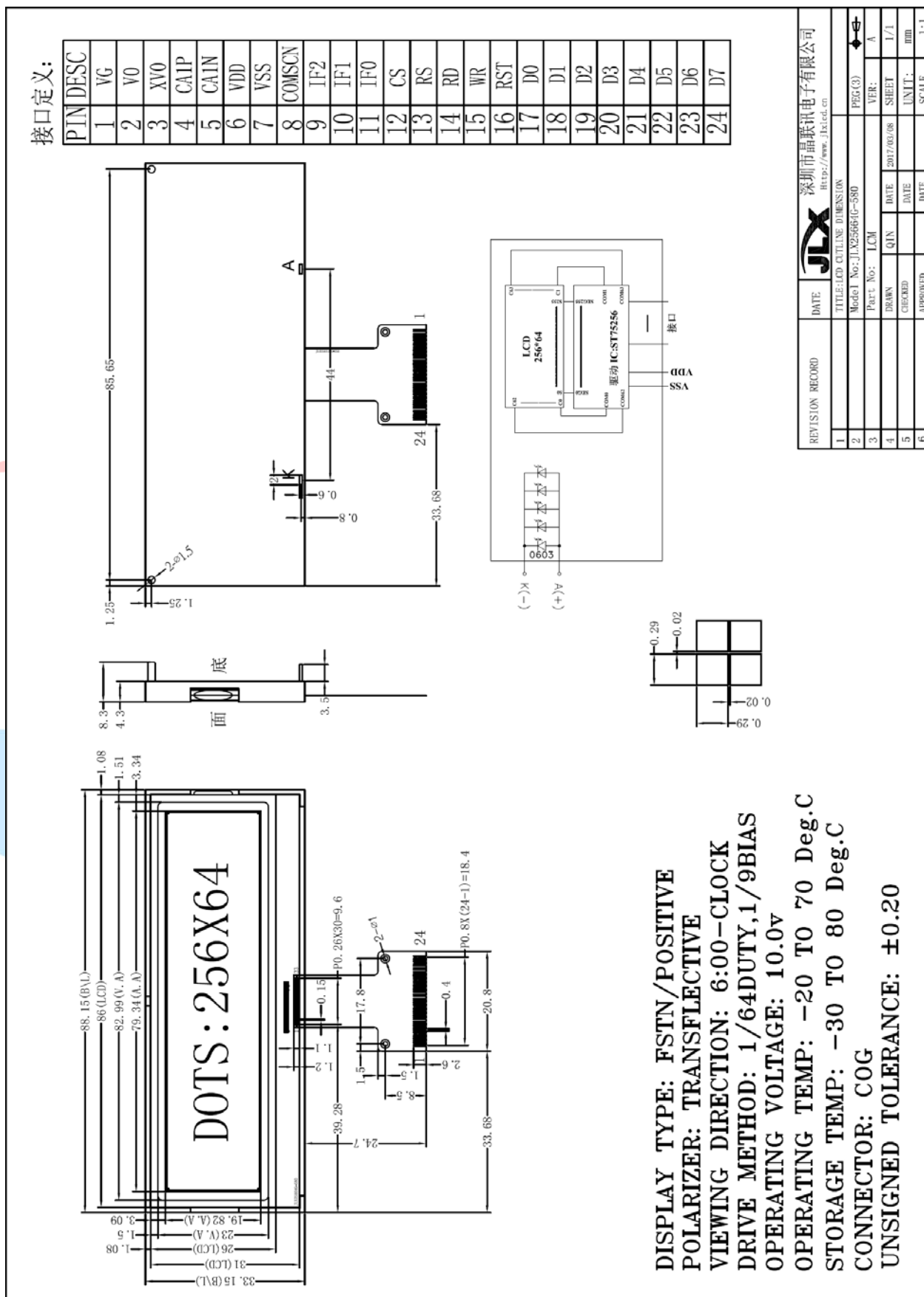


图 1. 外形尺寸

3.2 模块的接口引脚功能

3.2.1 并行时接口引脚功能

引 线 号	符 号	名 称	功 能
1	VG	偏压电路	LCD 偏置驱动电压, VG 与 VSS 之间接一个电容
2	V0	倍压电路	V0 与 XV0 之间接一个电容
3	XV0	倍压电路	
4	CA1P	倍压电路	CA1P 与 CA1N 之间接一个电容
5	CA1N	倍压电路	
6	VDD	电源电路	供电电源正极
7	VSS	接地	0V
8	COMSCN	镜像选择指令	VDD
9	IF2	I	L:接低电平
10	IF1	I	H:接高电平
11	IF0	I	L:接低电平
12	CS	片选	低电平片选
13	A0(RS)	寄存器选择信号	H:数据寄存器 0:指令寄存器 (IC 资料上所写为“CD”)
14	E (RD)	使能信号	6800 时序: 使能信号
15	RW(WR)	读/写	6800 时序: H:读数据 0:写数据
16	RST	复位	低电平复位, 复位完成后, 回到高电平, 液晶屏开始工作
17~24	DO~D7	I/O	并行接口时, 数据总线 DB0~DB7

表 1: 模块并行接口引脚功能

3.2.2 四线串行时接口引脚功能

引 线 号	符 号	名 称	功 能
1	VG	偏压电路	LCD 偏置驱动电压, VG 与 VSS 之间接一个电容
2	V0	倍压电路	V0 与 XV0 之间接一个电容
3	XV0	倍压电路	
4	CA1P	倍压电路	CA1P 与 CA1N 之间接一个电容
5	CA1N	倍压电路	
6	VDD	电源电路	供电电源正极
7	VSS	接地	0V
8	COMSCN	镜像选择指令	VDD
9	IF2	I	L:接低电平
10	IF1	I	L:接低电平
11	IF0	I	L:接低电平
12	CS	片选	低电平片选
13	A0(RS)	寄存器选择信号	H:数据寄存器 0:指令寄存器 (IC 资料上所写为“CD”)
14	E (RD)	使能信号	串行接口, RD 接高电平
15	RW(WR)	读、写	串行接口, RW 接高电平
16	RST	复位	低电平复位, 复位完成后, 回到高电平, 液晶屏开始工作
17	DO(SCK)	I/O	串行时钟
18~20	D1~D3 (SDA)	I/O	串行数据
21~24	D4~D7	I/O	串行接口, 请将 D4~D7 接 VDD

表 2: 4 线 SPI 串行接口引脚功能

3.2.3 I²C 总线时接口引脚功能

引 线 号	符 号	名 称	功 能
1	VG	偏压电路	LCD 偏置驱动电压, VG 与 VSS 之间接一个电容
2	V0	倍压电路	V0 与 XV0 之间接一个电容
3	XV0	倍压电路	
4	CA1P	倍压电路	CA1P 与 CA1N 之间接一个电容
5	CA1N	倍压电路	
6	VDD	电源电路	供电电源正极
7	VSS	接地	0V
8	COMSCN	镜像选择指令	VDD
9	IF2	I	L:接低电平
10	IF1	I	L:接低电平
11	IF0	I	H:接高电平
12	CS	片选	I ² C 接口, 不用, 此引脚接 VSS
13	A0(RS)	寄存器选择信号	I ² C 接口, 不用, 此引脚建议接高电平
14	E(RD)	使能信号	I ² C 接口, 不用, 此引脚建议接高电平
15	RW(WR)	读、写	I ² C 接口, 不用, 此引脚建议接高电平
16	RST	复位	低电平复位, 复位完成后, 回到高电平, 液晶屏开始工作
17	D0(SCK)	I/O	串行时钟
18~20	D1-D3(SDA)	I/O	串行数据
21	D4-D7	I/O	I2C 接口, 请将引脚接 VSS
22	D4-D7	I/O	I2C 接口, 请将引脚接 VSS
23	D4-D7	I/O	I2C 接口, 请将引脚接 VDD
24	D4-D7	I/O	I2C 接口, 请将引脚接 VDD

表 3: I²C 总线接口引脚功能

4. 基本原理+

4.1 液晶屏 (LCD)

在 LCD 上排列着 256×64 点阵, 256 个列信号与驱动 IC 相连, 64 个行信号也与驱动 IC 相连, IC 邦定在 LCD 玻璃上 (这种加工工艺叫 COG)。

4.2 工作电图:

图 1 是 JLX25664G-580 图像点阵型模块的电路框图, 它由驱动 IC ST75256 及几个电阻电容组成。

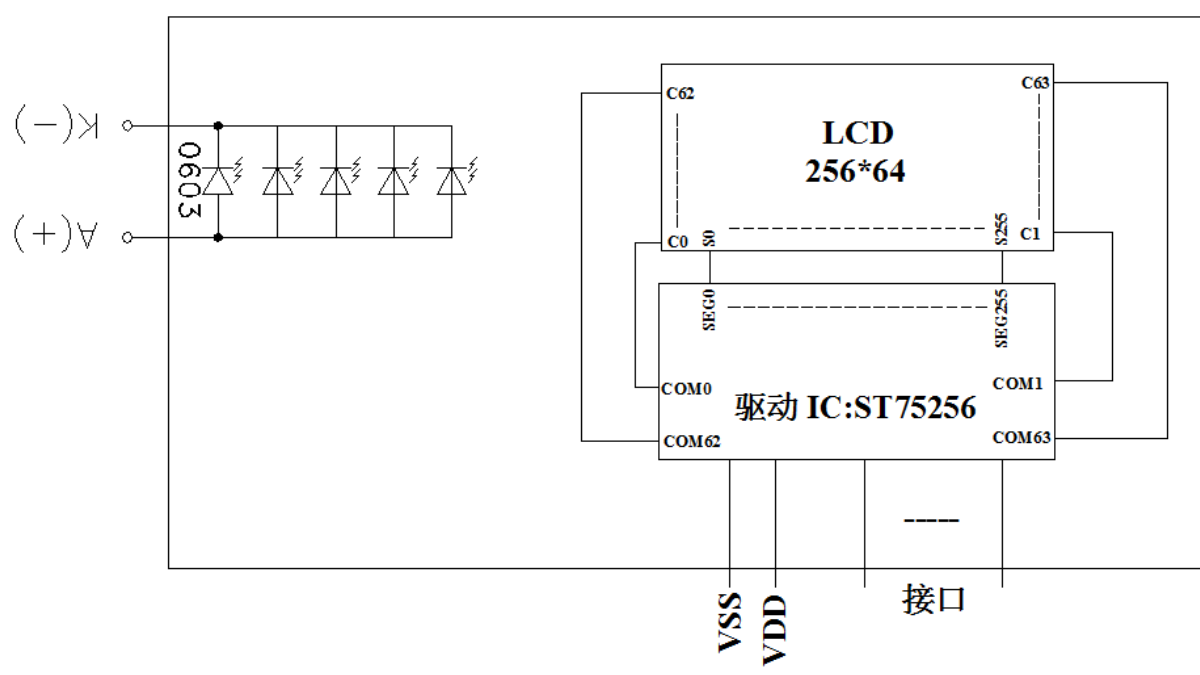


图 2: JLX25664G-580 图像点阵型液晶模块的电路框图

4.3 背光参数

该型号液晶模块带 LED 背光源。它的性能参数如下：
工作温度：-20~+70° C；
存储温度：-30~+80° C；
背光板选用白色；
正常工作电流为：40~100mA；
工作电压：3.0V

5. 技术参数

5.1 最大极限参数（超过极限参数则会损坏液晶模块）

名称	符号	标准值			单位
		最小	典型	最大	
电路电源	VDD - VSS	-0.3		7.0	V
LCD 驱动电压	VDD - V0	VDD - 13.5		VDD + 0.3	V
静电电压		-	-	100	V
工作温度		-20		+70	℃
储存温度		-30		+80	℃

表 5：最大极限参数

5.2 直流（DC）参数+

名 称	符 号	测 试 条 件	标 准 值			单位
			MIN	TYPE	MAX	
工作电压	VIN	3.3V 供电	1.7	3.3	3.4	V
输入高电平	VIH	—	2.2		VDD	V
输入低电平	VI0	—	-0.3		0.6	V
输出高电平	VOH	IOH = 0.2mA	2.4		—	V
输出低电平	V00	I00 = 1.2mA	—		0.4	V
模块工作电流	IDD	VDD = 3.3V	—		0.3	mA
背光工作电流	ILED	VLED=3.0V	40	65	100	mA

表 6：直流（DC）参数

6. 读写时序特性（AC 参数）

6.1 4 线 SPI 串行接口写时序特性（AC 参数）

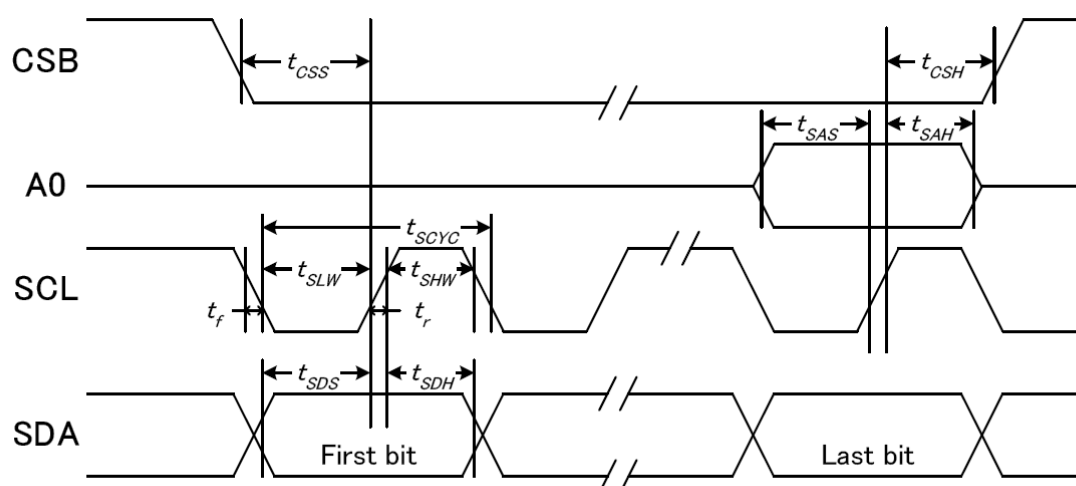


图 3. 从 CPU 写到 ST75256 (Writing Data from CPU to ST75256)

表 7. 写数据到 ST75256 的时序要求

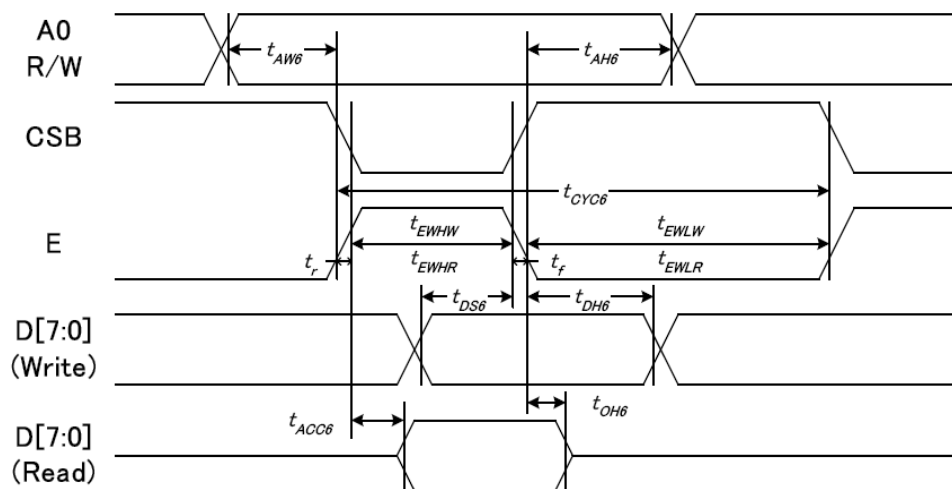
项 目	符 号	测试条件	极限值			单位
			MIN	TYPE	MAX	
4线 SPI串口时钟周期 (4-line SPI Clock Period)	tSCYC	引脚: SCL	80	--	--	ns
保持SCK高电平脉宽 (SCL "H" pulse width)	tSHW		30	--	--	ns
保持SCLK低电平脉宽 (SCL "L" pulse width)	tSLW		30	--	--	ns
地址建立时间 (Address setup time)	tSAS	引脚: A0	20	--	--	ns
地址保持时间 (Address hold time)	tSAH		20	--	--	ns
数据建立时间 (Data setup time)	tSDS	引脚: SID	20	--	--	ns
数据保持时间 (Data hold time)	tSDH		20	--	--	ns
片选信号建立时间 (CS-SCL time)	tCSS	引脚: CSB	20	--	--	ns
片选信号保持时间 (CS-SCL time)	tCSH		20	--	--	ns

VDD = 1.8~3.3V±5%, Ta = -30~85℃

输入信号的上升和下降时间 (TR, TF) 在 15 纳秒或更少的规定。

所有的时间, 用 20%和 80%作为标准规定的测定。

6.2 6800 时序并行接口的时序特性 (AC 参数)



1.

从 CPU 写到 ST75256 (Writing Data from CPU to ST75256)

图 4. 写数据到 ST75256 的时序要求 (6800 系列 MPU)

表 8. 读写数据的时序要求

项 目	符 号	名 称	极限值			单位
			MIN	TYPE	MAX	
地址保持时间	A0	tAH6	20		--	ns
地址建立时间		tAW6	0		--	ns
系统循环时间	E	tCYC6	160		--	ns
使能“低”脉冲宽度		tEWLW	70		--	ns
使能“高”脉冲宽度		tEWHW	70		--	ns
写数据建立时间	DB[7: 0]	tDS6	15		--	ns
写数据保持时间		tDH6	15		--	ns

VDD =1.8~3.3V±5%, Ta = -30~85℃

输入信号的上升时间和下降时间（TR，TF）是在 15 纳秒或更少的规定。当系统循环时间非常快，

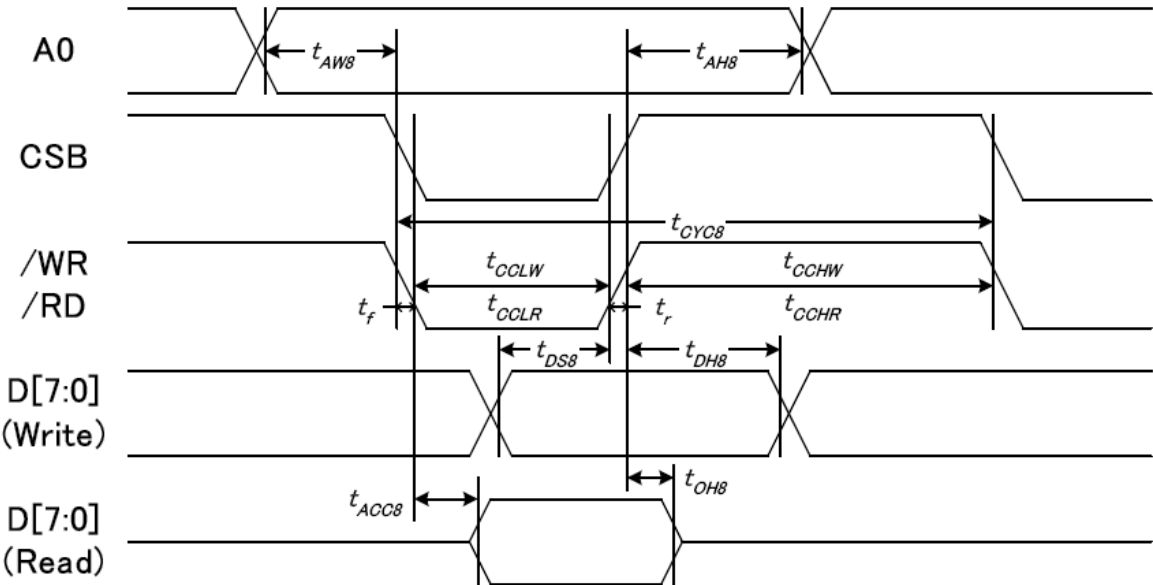
（TR + TF）≤（tcyc6 - tewlw - tewhw）指定。

所有的时间，用 20%和 80%作为参考指定的测定。

tewlw 指定为重叠的 CSB “H” 和 “L” 。

R / W 信号总是 “H”

6.3 8080 时序并行接口的时序特性（AC 参数）



从 CPU 写到 ST75256（Writing Data from CPU to ST75256）

图 4. 写数据到 ST75256 的时序要求（8080 系列 MPU）

表 8. 读写数据的时序要求

项 目	符 号	名 称	极限值	单位
-----	-----	-----	-----	----

			MIN	TYPE	MAX	
地址保持时间	A0	tAH8	20		--	ns
地址建立时间		tAW8	0		--	ns
系统循环时间	/WR	tCYC8	160		--	ns
使能“低”脉冲宽度		tCCLW	70		--	ns
使能“高”脉冲宽度		tCCHW	70		--	ns
写数据建立时间	DB	tDS8	15		--	ns
写数据保持时间		tDH8	15		--	ns

VDD =1.8~3.3V±5%, Ta = -30~85℃

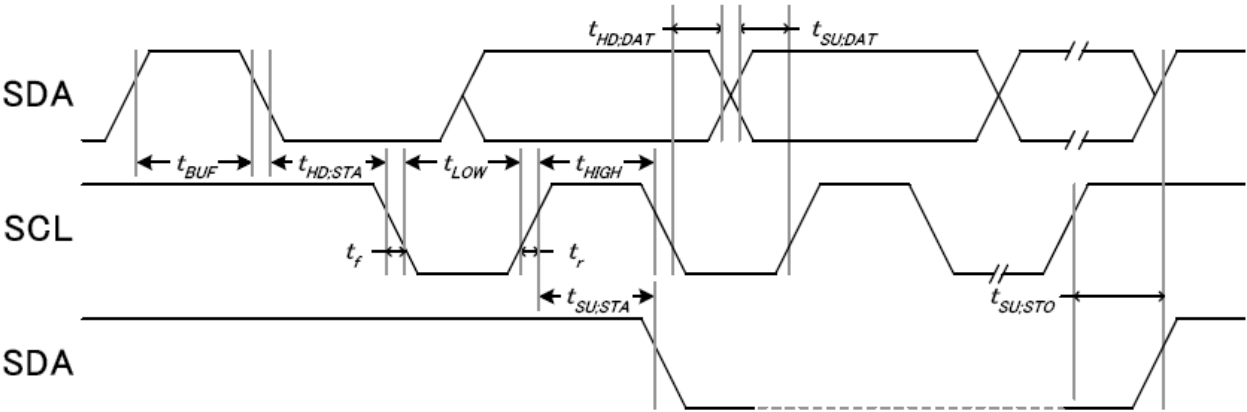
输入信号的上升时间和下降时间（TR，TF）是在 15 纳秒或更少的规定。当系统循环时间非常快，

（TR + TF）≤（tcyc8 - tcclw - tcchw）指定。

所有的时间，用 20%和 80%作为参考指定的测定。

tcclw 被指定为“L”之间的重叠 CSB 和/ WR 处于“L”级

6.3 I²C 接口的时序特性（AC 参数）



从 CPU 写到 ST75256（Writing Data from CPU to ST75256）

图 4. 写数据到 ST75256 的时序要求（I²C 系列 MPU）

表 8. 读写数据的时序要求

项 目	符 号	名 称	极限值			单位
			MIN	TYPE	MAX	
SCL时钟频率	CSL	FSCLK	--		400	kUZ
SCL时钟的低周期	CSL	TLOW	1.3		--	us
SCL时钟周期	CSL	THIGH	0.6		--	us
数据保持时间	SDA	TSU;Data	0.1		--	ns

数据建立时间	SDA	THD;Data	0		0.9	us
SCL，SDA 的上升时间	SCL	TR	20+0.1Cb		300	ns
SCL，SDA 下降时间	SCL	TF	20+0.1Cb		300	ns
每个总线为代表的电容性负载		Cb	--		400	pF
一个重复起始条件设置时间	SDA	TSU;SUA	0.6		--	us
启动条件的保持时间	SDA	THD;STA	0.6		--	us
为停止条件建立时间		TSU;STO	0.6		--	us
容许峰值宽度总线		TSW	--		50	ns
开始和停止条件之间的总线空闲时间	SCL	TBUF	0.1			us

所有的时间，用 20%和 80%作为标准规定的测定。

这是推荐的操作 I C 接口与 VDD1 高于 2.6V。

6.4 电源启动后复位的时序要求（RESET CONDITION AFTER POWER UP）：

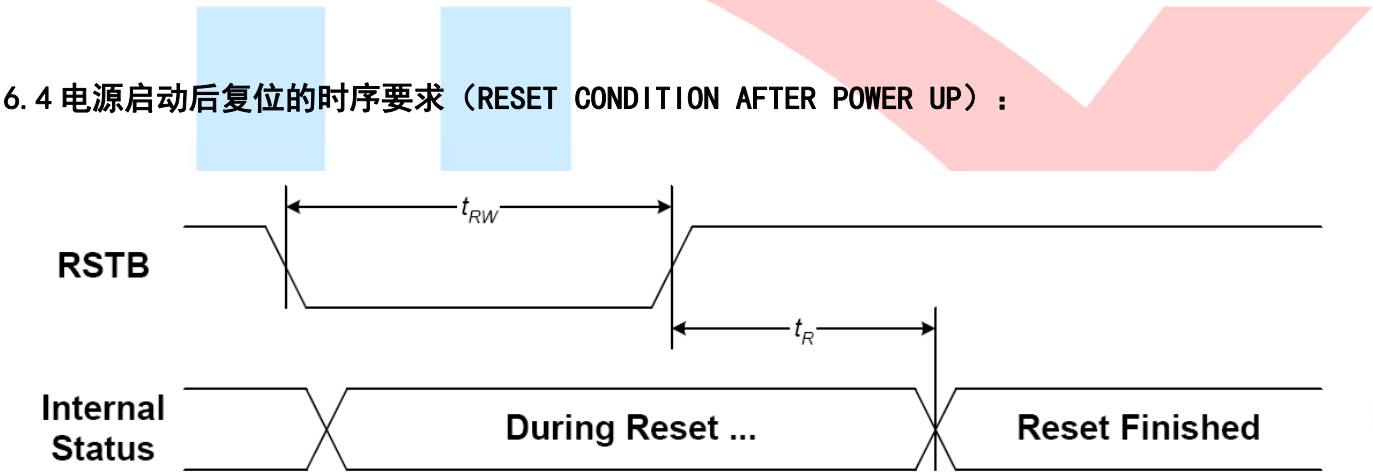


图 5：电源启动后复位的时序

表 6：电源启动后复位的时序要求

项 目	符 号	测试条件	极限值			单位
			MIN	TYPE	MAX	
复位时间	T _{RW}		--	--	1	us
复位保持低电平的时间	T _{RD}	引脚：RESET, WR	1	--	--	ms

7. 指令功能：

7.1 指令表

INSTRUCTION	A0	R/W	COMMAND BYTE								DESCRIPTION
			D7	D6	D5	D4	D3	D2	D1	D0	
1.Extension Command	0	0	0	0	1	1	EXT1	0	0	EXT0	Set extension instruction
Ext[1:0]=0,0 (Extension Command 1)											
2.Display ON/OFF	0	0	1	0	1	0	1	1	1	DSP	Set LCD display DSP=0: Display off DSP=1: Display on
3.Inverse Display	0	0	1	0	1	0	0	1	1	INV	Set inverse display INV=0: Normal display INV=1: Inverse display
4.All Pixel ON/OFF	0	0	0	0	1	0	0	0	1	AP	Set all pixel on mode AP=0: All pixel off mode AP=1: All pixel on mode
5.Display Control	0	0	1	1	0	0	1	0	1	0	Set display control CLD :Set CL dividing ratio DT[7:0] : Set the number of duty LF[4:0] : Set N-line inversion counter FI : Set the inversion type of frame at the end of common scan cycle
	1	0	0	0	0	0	0	CLD	0	0	
	1	0	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0	
	1	0	0	0	LF4	FI	LF3	LF2	LF1	LF0	
6.Power Save	0	0	1	0	0	1	0	1	0	SLP	Set power save mode SLP=0: Sleep out mode SLP=1: Sleep in mode
7.Set Page Address	0	0	0	1	1	1	0	1	0	1	Set page address Starting page address: 00h≤YS≤28h Ending page address: YS≤YE≤28h
	1	0	YS7	YS6	YS5	YS4	YS3	YS2	YS1	YS0	
	1	0	YE7	YE6	YE5	YE4	YE3	YE2	YE1	YE0	
8.Set Column Address	0	0	0	0	0	1	0	1	0	1	Set column address Starting column address: 00h≤XS≤FFh Ending column address: XS≤XE≤FFh
	1	0	XS7	XS6	XS5	XS4	XS3	XS2	XS1	XS0	
	1	0	XE7	XE6	XE5	XE4	XE3	XE2	XE1	XE0	
9.Data Scan Direction	0	0	1	0	1	1	1	1	0	0	Set normal/ inverse display of address and address scan direction
	1	0	0	0	0	0	0	MV	MX	MY	
10.Write Data	0	0	0	1	0	1	1	1	0	0	Write data to DDRAM
	1	0	D7	D6	D5	D4	D3	D2	D1	D0	
11.Read Data	0	0	0	1	0	1	1	1	0	1	Read data from DDRAM (Only for parallel interface and I ² C)
	1	1	D7	D6	D5	D4	D3	D2	D1	D0	
12.Partial In	0	0	1	0	1	0	1	0	0	0	Set partial area Starting partial display address: 00h≤PTS≤A1h Ending partial display address: 00h≤PTE≤A1h
	1	0	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0	
	1	0	PTE7	PTE6	PTE5	PTE4	PTE3	PTE2	PTE1	PTE0	

INSTRUCTION	A0	R/W	COMMAND BYTE								DESCRIPTION
			D7	D6	D5	D4	D3	D2	D1	D0	
13. Partial Out	0	0	1	0	1	0	1	0	0	1	Exit the partial mode
14. Read/Modify/Write In	0	0	1	1	1	0	0	0	0	0	Enable read modify write
15. Read/Modify/Write Out	0	0	1	1	1	0	1	1	1	0	Disable read modify write
16. Scroll Area	0	0	1	0	1	0	1	0	1	0	Set scroll area TL[7:0] : Set top line address BL[7:0] : Set bottom line address NSL[7:0] : Number of specified line SCM[1:0] : Area scroll mode
	1	0	TL7	TL6	TL5	TL4	TL3	TL2	TL1	TL0	
	1	0	BL7	BL6	BL5	BL4	BL3	BL2	BL1	BL0	
	1	0	NSL7	NSL6	NSL5	NSL4	NSL3	NSL2	NSL1	NSL0	
	1	0	0	0	0	0	0	0	SCM1	SCM0	
17. Set Start Line	0	0	1	0	1	0	1	0	1	1	Set scroll start address $00h \leq SL \leq A1h$
	1	0	SL7	SL6	SL5	SL4	SL3	SL2	SL1	SL0	
18. OSC ON	0	0	1	1	0	1	0	0	0	1	Turn on the internal oscillator
19. OSC OFF	0	0	1	1	0	1	0	0	1	0	Turn off the internal oscillator
20. Power Control	0	0	0	0	1	0	0	0	0	0	Power circuit operation VB=0: OFF, VB=1: ON VF=0: OFF, VF=1: ON VR=0: OFF, VR=1: ON
	1	0	0	0	0	0	VB	0	VF	VR	
21. Set Vop	0	0	1	0	0	0	0	0	0	1	Set Vop
	1	0	0	0	Vop5	Vop4	Vop3	Vop2	Vop1	Vop0	
	1	0	0	0	0	0	0	Vop8	Vop7	Vop6	
22. Vop Control	0	0	1	1	0	1	0	1	1	VOL	Control Vop VOL=0: Vop increase one step VOL=1: Vop decrease one step
23. Read Register Mode	0	0	0	1	1	1	1	1	0	REG	Set read register mode REG=0: read the register value of Vop[5:0] REG=1: read the register value of Vop[8:6]
24. Nop	0	0	0	0	1	0	0	1	0	1	No operation
25. Read Status (Parallel and I ² C)	0	1	D7	D6	D5	D4	D3	D2	D1	D0	Read status byte (Parallel and I ² C)
26. Read Status (4-Line and 3-Line SPI)	0	0	1	1	1	1	1	1	1	0	Read status byte (4-Line and 3-Line SPI)
	0	1	D7	D6	D5	D4	D3	D2	D1	D0	
27. Data Format Select	0	0	0	0	0	0	1	DO	0	0	DO=0: LSB on bottom (Default) DO=1: LSB on top
28. Display Mode	0	0	1	1	1	1	0	0	0	0	Set display mode DM=0 : Mono (Default) DM=1 : 4Gray Scale Mode
	1	0	0	0	0	1	0	0	0	DM	

INSTRUCTION	A0	R/W	COMMAND BYTE								DESCRIPTION
			D7	D6	D5	D4	D3	D2	D1	D0	
29.Set ICON	0	0	0	1	1	1	0	1	1	ICON	Enable/Disable ICON RAM ICON=1 ; Enable ICON RAM ICON=0 ; Disable ICON RAM
30.Set Master/Slave	0	0	0	1	1	0	1	1	1	MS	Select Master or Slave mode MS=0 ; CMD for Master (Default) MS=1 ; CMD for Slave
Ext[1:0]=0,1 (Extension Command 2)											
31.Set Gray Level	0	0	0	0	1	0	0	0	0	0	Set gray scale level GL[4:0]: Set Light Gray Level GD[4:0]: Set Dark Gray Level
	1	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	GL4	GL3	GL2	GL1	GL0	
	1	0	0	0	0	GL4	GL3	GL2	GL1	GL0	
	1	0	0	0	0	GL4	GL3	GL2	GL1	GL0	
	1	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	GD4	GD3	GD2	GD1	GD0	
	1	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	GD4	GD3	GD2	GD1	GD0	
	1	0	0	0	0	GD4	GD3	GD2	GD1	GD0	
	1	0	0	0	0	GD4	GD3	GD2	GD1	GD0	
	1	0	0	0	0	0	0	0	0	0	
32.Analog Circuit Set	0	0	0	0	1	1	0	0	1	0	Set analog circuit BE[1:0]: Booster efficiency set BS[2:0]: Set bias ratio
	1	0	0	0	0	0	0	0	0	0	
	1	0	0	0	0	0	0	0	BE1	BE0	
	1	0	0	0	0	0	0	BS2	BS1	BS0	
33.Booster Level	0	0	0	1	0	1	0	0	0	1	Set booster level BST=0 : X8 BST=1 : X10
	1	0	1	1	1	1	1	0	1	BST	
34. Driving Select	0	0	0	1	0	0	0	0	0	DS	Power type DS=0: Internal (Default) DS=1 :External
35.Auto Read Control	0	0	1	1	0	1	0	1	1	1	Set auto-read instruction XARD=0: Enable auto read XARD=1: Disable auto read
	1	0	1	0	0	XARD	1	1	1	1	
36.OTP WR/RD Control	0	0	1	1	1	0	0	0	0	0	OTP WR/RD control WR/RD=0: Enable OTP read WR/RD=1: Enable OTP write
	1	0	0	0	WR/RD	0	0	0	0	0	
37.OTP Control Out	0	0	1	1	1	0	0	0	0	1	OTP control out
38.OTP Write	0	0	1	1	1	0	0	0	1	0	OTP write

INSTRUCTION	A0	R/W	COMMAND BYTE								DESCRIPTION
			D7	D6	D5	D4	D3	D2	D1	D0	
39.OTP Read	0	0	1	1	1	0	0	0	1	1	OTP read
40.OTP Selection Control	0	0	1	1	1	0	0	1	0	0	OTP selection control Ctrl=1: Disable OTP Selection Ctrl=0: Enable OTP Selection
	1	0	1	Ctrl	0	1	1	0	0	1	
41.OTP Programming Setting	0	0	1	1	1	0	0	1	0	1	OTP programming setting
	1	0	0	0	0	0	1	1	1	1	
42.Frame Rate	0	0	1	1	1	1	0	0	0	0	Frame rate setting in different temperature range
	1	0	0	0	0	FRA4	FRA3	FRA2	FRA1	FRA0	
	1	0	0	0	0	FRB4	FRB3	FRB2	FRB1	FRB0	
	1	0	0	0	0	FRC4	FRC3	FRC2	FRC1	FRC0	
	1	0	0	0	0	FRD4	FRD3	FRD2	FRD1	FRD0	
43.Temperature Range	0	0	1	1	1	1	0	0	1	0	Temperature range setting
	1	0	0	TA6	TA5	TA4	TA3	TA2	TA1	TA0	
	1	0	0	TB6	TB5	TB4	TB3	TB2	TB1	TB0	
	1	0	0	TC6	TC5	TC4	TC3	TC2	TC1	TC0	
44.Temperature Gradient Compensation	0	0	1	1	1	1	0	1	0	0	Set temperature gradient compensation coefficient
	1	0	MT13	MT12	MT11	MT10	MT03	MT02	MT01	MT00	
	1	0	MT33	MT32	MT31	MT30	MT23	MT22	MT21	MT20	
	1	0	MT53	MT52	MT51	MT50	MT43	MT42	MT41	MT40	
	1	0	MT73	MT72	MT71	MT70	MT63	MT62	MT61	MT60	
	1	0	MT93	MT92	MT91	MT90	MT83	MT82	MT81	MT80	
	1	0	MTB3	MTB2	MTB1	MTB0	MTA3	MTA2	MTA1	MTA0	
	1	0	MTD3	MTD2	MTD1	MTD0	MTC3	MTC2	MTC1	MTC0	
	1	0	MTF3	MTF2	MTF1	MTF0	MTE3	MTE2	MTE1	MTE0	
Ext[1:0]=1,0(Extension Command 3)											
45.Set ID	0	0	1	1	0	1	0	1	0	1	Set ID
	1	0	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	
46 Read ID	0	0	0	1	1	1	1	1	1	RID	Read ID RID=1 ; Enable RID=0 ; Disable
Ext[1:0]=1,1(Extension Command 4)											
47.Enable OTP	0	0	1	1	0	1	0	1	1	0	Enable OTP EOTP =0 ; Disable (Default) EOTP =1 ; Enable
	1	0	0	0	0	EOTP	0	0	0	0	

表 8. 指令表

请详细参考 IC 资料”ST75256.PDF”。

7.2 点阵与 DD RAM 地址的对应关系

请留意页的定义: PAGE, 与平时所讲的“页”并不是一个意思, 在此表示 8 个行就是一个“页”, 一个 256*64 点阵的屏分为 8 个“页”, 从第 0“页”到第 7“页”。

DB7--DB0 的排列方向: 数据是从下向上排列的。最低位 D0 是在最上面, 最高位 D7 是在最下面。每一位 (bit) 数据对应一个点阵, 通常“1”代表点亮该点阵, “0”代表关掉该点阵。如下图所示:

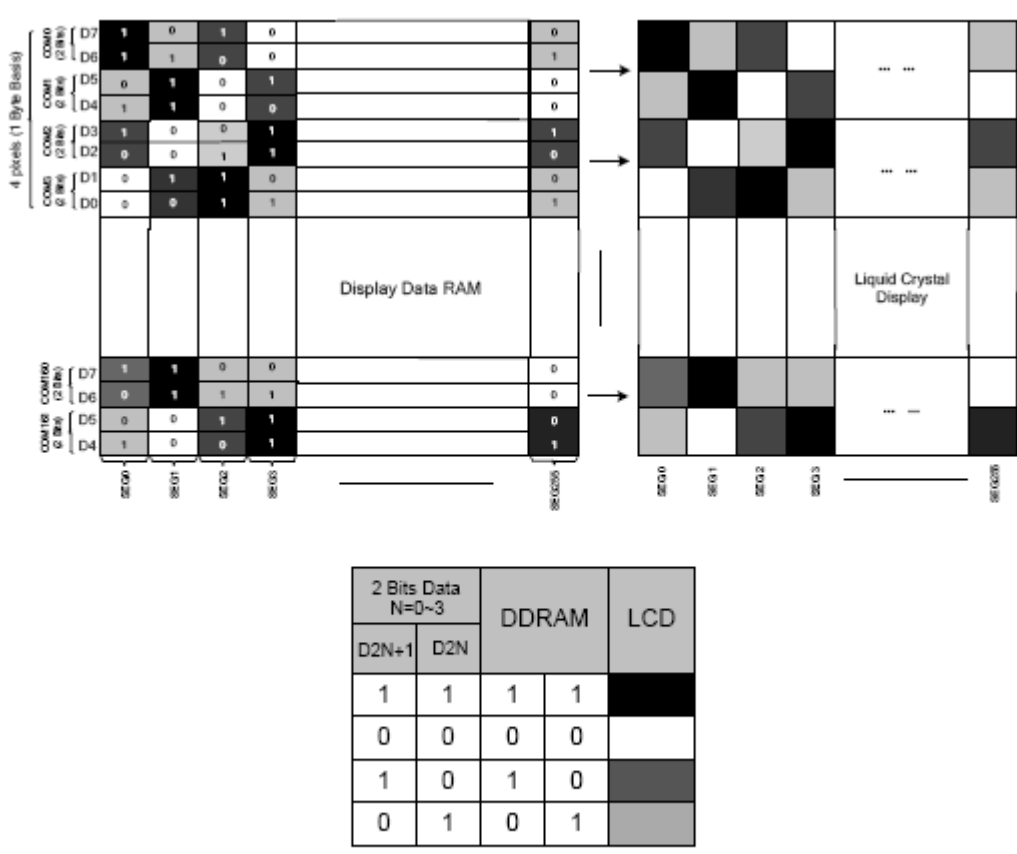


Figure 21 DDRAM Mapping (4-Level Gray Scale Mode)

下图摘自 ST75256 IC 资料，可通过“ST75256.PDF”之第 37 页获取最佳效果。

7.3 初始化方法

用户所编的显示程序, 开始必须进行初始化, 否则模块无法正常显示, 过程请参考程序

点亮液晶模块的步骤

硬件准备:
开发板（或专门设计的主板）、单片机、电源、连接线、仿真器或程序下载器（又名烧录器）

正确地接线
根据说明书正确地与开发板连接, 连接的线包括: 液晶模块电源线、背光电源线、IO端口（接口）
IO端口包括: 并口时: CS、RESET、RW、E、RS、D0--D7, 串口时: CS、SCLK、SDA、RESET、RS

编写软件
背光给合适的直流电可以点亮, 但液晶屏里面没有程序, 只给电不能让液晶屏显示（我们通常说“点亮”），程序须另外编写, 并烧录（下载）到单片机里液晶模块才能工作。

7.4 程序举例:

7.4.1 串行接口

液晶模块与 MPU(以 8051 系列单片机为例)接口图如下:

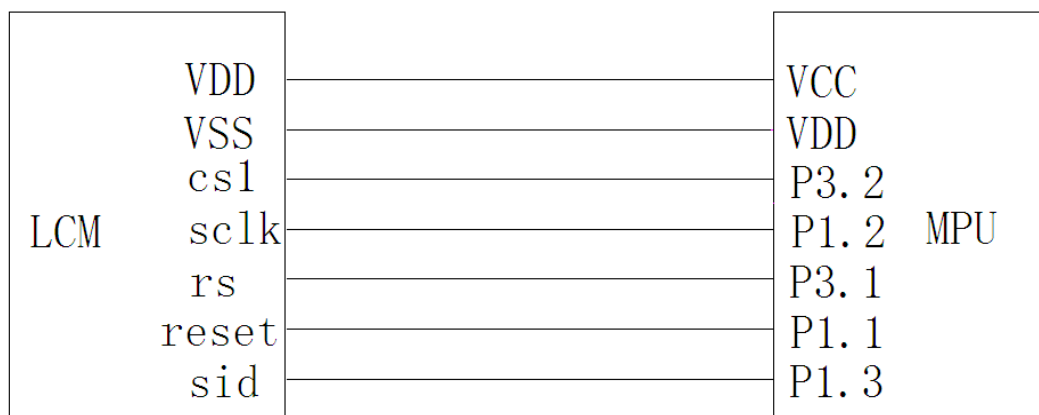
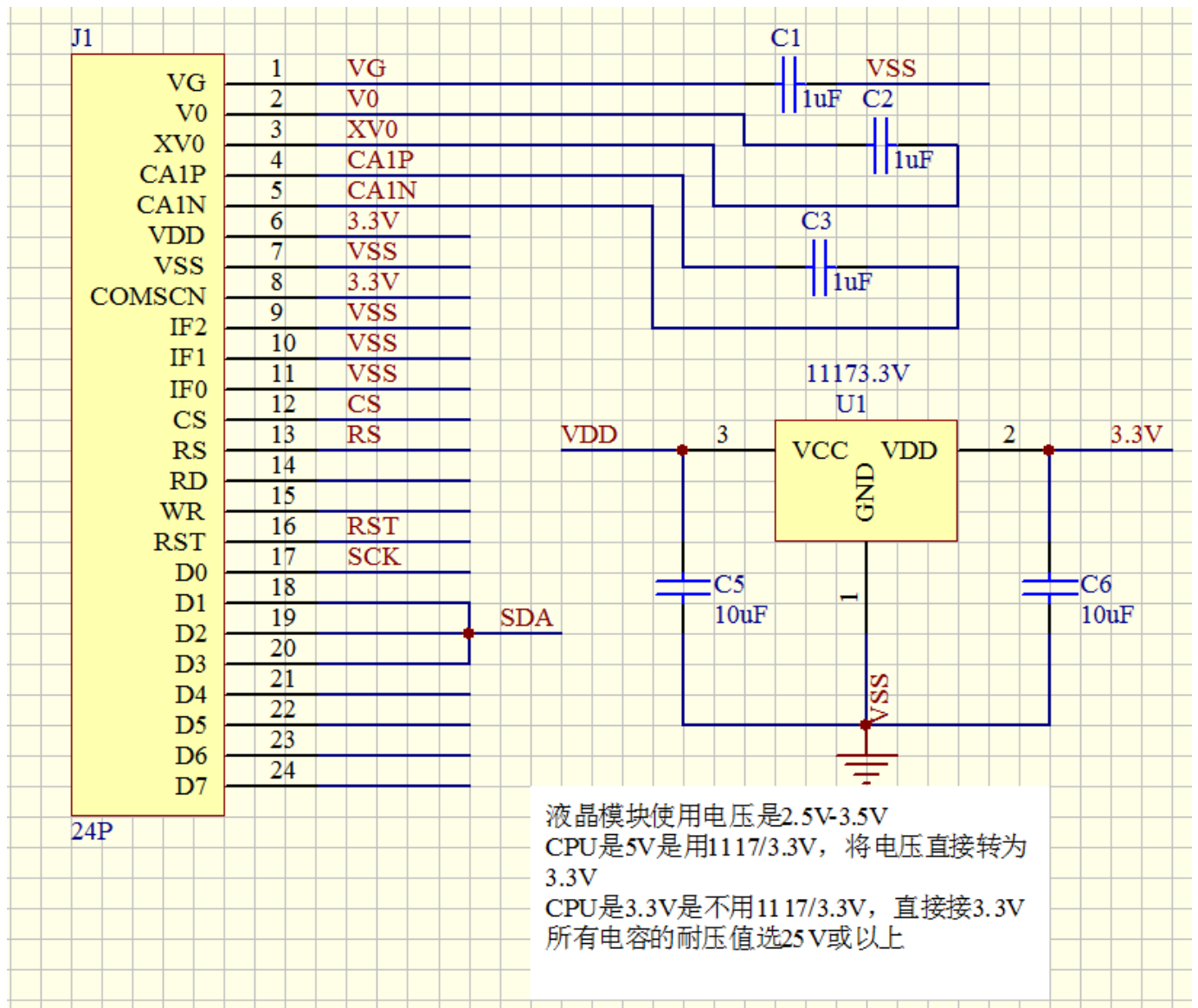


图 8. 串行接口



7.5.2 以下是串行接口例程序

```

/* 液晶模块型号：JLX25664G-580
   4线串行接口
   驱动 IC 是:ST75256
   版权所有：晶联讯电子：网址 http://www.jlxlcd.cn;
*/
#include <reg52.h>
#include <intrins.h>
#include <chinese_code.h>

sbit lcd_cs1 = P3^2;//CS
sbit lcd_reset= P1^1;//RST
sbit lcd_sclk = P1^2;//串行时钟
sbit lcd_rs = P3^1;//RS
sbit lcd_sid = P1^3;//串行数据
sbit key = P2^0; //按键

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

```
uchar code bmp10[];

/*延时：1 毫秒的 i 倍*/
void delay(int i)
{
    int j,k;
    for(j=0;j<i;j++)
        for(k=0;k<110;k++);
}

/*延时：1us 的 i 倍*/
void delay_us(int i)
{
    int j,k;
    for(j=0;j<i;j++)
        for(k=0;k<1;k++);
}

/*等待一个按键，我的主板是用 P2.0 与 GND 之间接一个按键*/
void waitkey()
{
    repeat:
        if (key==1) goto repeat;
        else delay(2000);
}

//写指令到 LCD 模块
void transfer_command_lcd(int data1)
{
    char i;
    lcd_cs1=0;
    lcd_rs=0;
    for(i=0;i<8;i++)
    {
        lcd_sclk=0;
        if(data1&0x80) lcd_sid=1;
        else lcd_sid=0;
        lcd_sclk=1;
        data1<<=1;
    }
    lcd_cs1=1;
}

//写数据到 LCD 模块
void transfer_data_lcd(int data1)
{
    char i;
    lcd_cs1=0;
    lcd_rs=1;
    for(i=0;i<8;i++)
    {
        lcd_sclk=0;
        if(data1&0x80) lcd_sid=1;
        else lcd_sid=0;
        lcd_sclk=1;
    }
}
```

```
    data1<<=1;
}
lcd_cs1=1;
}

void initial_lcd()
{
    reset=0;
    delay(100);
    reset=1;
    delay(100);

    transfer_command_lcd(0x30); //EXT=0
    transfer_command_lcd(0x94); //Sleep out
    transfer_command_lcd(0x31); //EXT=1
    transfer_command_lcd(0xD7); //Autoread disable
    transfer_data_lcd(0X9F); //

    transfer_command_lcd(0x32); //Analog SET
    transfer_data_lcd(0x00); //OSC Frequency adjustment
    transfer_data_lcd(0x01); //Frequency on booster capacitors->6KHz
    transfer_data_lcd(0x05); //Bias=1/9

    transfer_command_lcd(0x20); // Gray Level
    transfer_data_lcd(0x01);
    transfer_data_lcd(0x03);
    transfer_data_lcd(0x05);
    transfer_data_lcd(0x07);
    transfer_data_lcd(0x09);
    transfer_data_lcd(0x0b);
    transfer_data_lcd(0x0d);
    transfer_data_lcd(0x10);
    transfer_data_lcd(0x11);
    transfer_data_lcd(0x13);
    transfer_data_lcd(0x15);
    transfer_data_lcd(0x17);
    transfer_data_lcd(0x19);
    transfer_data_lcd(0x1b);
    transfer_data_lcd(0x1d);
    transfer_data_lcd(0x1f);

    transfer_command_lcd(0x30); //EXT1=0, EXT0=0, 表示选择了“扩展指令表 1”
    transfer_command_lcd(0x75); //页地址设置
    transfer_data_lcd(0X00); //起始页地址: YS=0X00
    transfer_data_lcd(0X1f); //结束页地址: YE=0x1F 每 4 行为一页, 第 0~3 行为第 0 页, 第 124~
127 行为第 31 页 (31=0x1f)
    transfer_command_lcd(0x15); //列地址设置
    transfer_data_lcd(0X00); //起始列地址: XS=0
    transfer_data_lcd(0Xff); // 结束列地址: XE=256 (0xff)

    transfer_command_lcd(0xBC); //Data scan direction
    transfer_data_lcd(0x00); //MX. MY=Normal
    transfer_data_lcd(0xA6);

    transfer_data_lcd(0x0c); //数据格式, 如果设为 0x0C: 表示选择 LSB (DB0) 在顶, 如果设为 0x08:
表示选择 LSB (DB0) 在底

    transfer_command_lcd(0xCA); //显示控制
    transfer_data_lcd(0X00); //设置 CL 驱动频率: CLD=0
    transfer_data_lcd(0X3F); //占空比: Duty=64
```

```
// transfer_data_lcd(0X7F);          //占空比: Duty=128
transfer_data_lcd(0X20);          //N 行反显: Nline=off

transfer_command_lcd(0xF0);      //显示模式
transfer_data_lcd(0X10);          //如果设为 0x11: 表示选择 4 灰度级模式, 如果设为 0x10:表示选择
黑白模式

transfer_command_lcd(0x81);      //设置对比度, “0x81” 不可改动, 紧跟着的 2 个数据是可改的, 但 “先
微调后粗调” 这个顺序别乱了
transfer_data_lcd(0x12);          //对比度微调, 可调范围 0x00~0x3f, 共 64 级
transfer_data_lcd(0x02);          //对比度粗调, 可调范围 0x00~0x07, 共 8 级

transfer_command_lcd(0x20);      //Power control
transfer_data_lcd(0x0B);          //D0=regulator ; D1=follower ; D3=booste,  on:1 off:0
delay_us(100);

transfer_command_lcd(0xAF);      //打开显示

}
```

/*写 LCD 行列地址: X 为起始的列地址, Y 为起始的行地址, x_total,y_total 分别为列地址及行地址的起点到终点的差值 */

```
void lcd_address(int x,int y,x_total,y_total)
{
    x=x-1;
    y=y-1;

    transfer_command_lcd(0x15); //Set Column Address
    transfer_data_lcd(x);
    transfer_data_lcd(x+x_total-1);

    transfer_command_lcd(0x75); //Set Page Address
    transfer_data_lcd(y);
    transfer_data_lcd(y+y_total-1);
    transfer_command_lcd(0x30);
    transfer_command_lcd(0x5c);
}
```

/*清屏*/

```
void clear_screen()
{
    int i,j;
    lcd_address(0,0,256,34);
    for(i=0;i<34;i++)
    {
        for(j=0;j<256;j++)
        {
            transfer_data_lcd(0x00);
        }
    }
}
```

```
void test(int x,int y)
{
    int i,j;
    lcd_address(x,y,256,16);
}
```

```
for(i=0;i<16;i++)
{
    for(j=0;j<256;j++)
    {
        transfer_data_lcd(0xff);
    }
}

//写入一组 16x16 点阵的汉字字符串（字符串表格中需含有此字）
//括号里的参数：（页，列，汉字字符串）
void display_string_16x16(uchar column, uchar page, uchar *text)
{
    uchar i, j, k;
    uint address;
    j=0;
    while(text[j] != '\0')
    {
        i=0;
        address=1;
        while(Chinese_text_16x16[i] > 0x7e)
        {
            if(Chinese_text_16x16[i] == text[j])
            {
                if(Chinese_text_16x16[i+1] == text[j+1])
                {
                    address=i*16;
                    break;
                }
            }
            i +=2;
        }
        if(column>255)
        {
            column=0;
            page+=2;
        }
        if(address !=1)
        {
            lcd_address(column, page, 16, 2);
            for(k=0;k<2;k++)
            {
                for(i=0;i<16;i++)
                {
                    transfer_data_lcd(Chinese_code_16x16[address]);
                    address++;
                }
            }
            j +=2;
        }
        else
        {
            lcd_address(column, page, 16, 2);
            for(k=0;k<2;k++)
            {
                for(i=0;i<16;i++)
                {
                    transfer_data_lcd(0x00);
                }
            }
        }
    }
}
```

```
    }  
    }  
    j++;  
}  
column+=16;  
}  
}
```

/*显示 32*32 点阵的汉字或等同于 32*32 点阵的图像*/
void disp_32x32(int x, int y, uchar *dp)

```
{  
    int i, j;  
    lcd_address(x, y, 32, 4);  
    for(i=0; i<4; i++)  
    {  
        for(j=0; j<32; j++)  
        {  
            transfer_data_lcd(*dp);  
            dp++;  
        }  
    }  
}
```

/*显示 172*104 点阵的图像*/

```
void disp_256x128(int x, int y, char *dp)  
{  
    int i, j;  
    lcd_address(x, y, 256, 8);  
    for(i=0; i<8; i++)  
    {  
        for(j=0; j<256; j++)  
        {  
            transfer_data_lcd(*dp);  
            dp++;  
        }  
    }  
}
```

//-----

void main ()

```
{  
    initial_lcd();                //对液晶模块进行初始化设置  
    while(1)  
    {  
        clear_screen();          //清屏  
        disp_256x128(1, 1, bmp8); //显示一幅 240*160 点阵的黑白图。  
        waitkey();  
        clear_screen();          //清屏  
        disp_256x128(1, 1, bmp1); //显示一幅 240*160 点阵的黑白图。  
        waitkey();  
        clear_screen();  
        display_string_16x16(32, 1, "深圳市晶联讯电子有限公司");  
        waitkey();  
        clear_screen();          //清屏  
        disp_32x32(48, 4, jing2);  
        disp_32x32((32*1+48), 4, lian2);  
        disp_32x32((32*2+48), 4, xun2);  
    }  
}
```

```
disp_32x32((32*3+48), 4, dian2);  
disp_32x32((32*4+48), 4, zi2);  
waitkey();  
}  
}
```

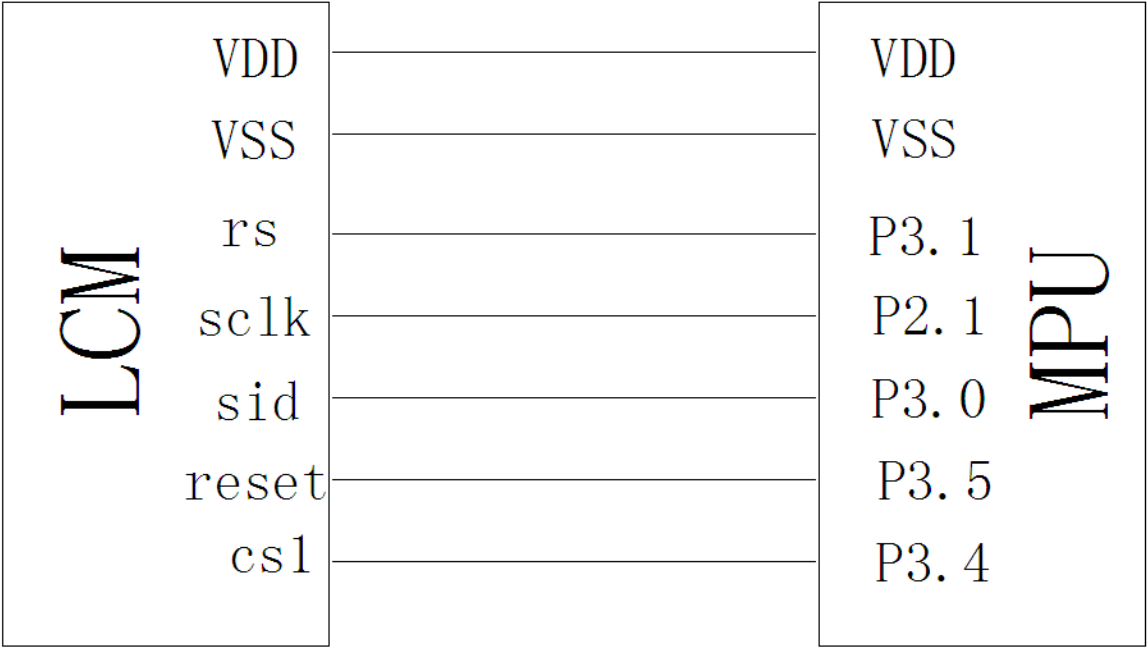
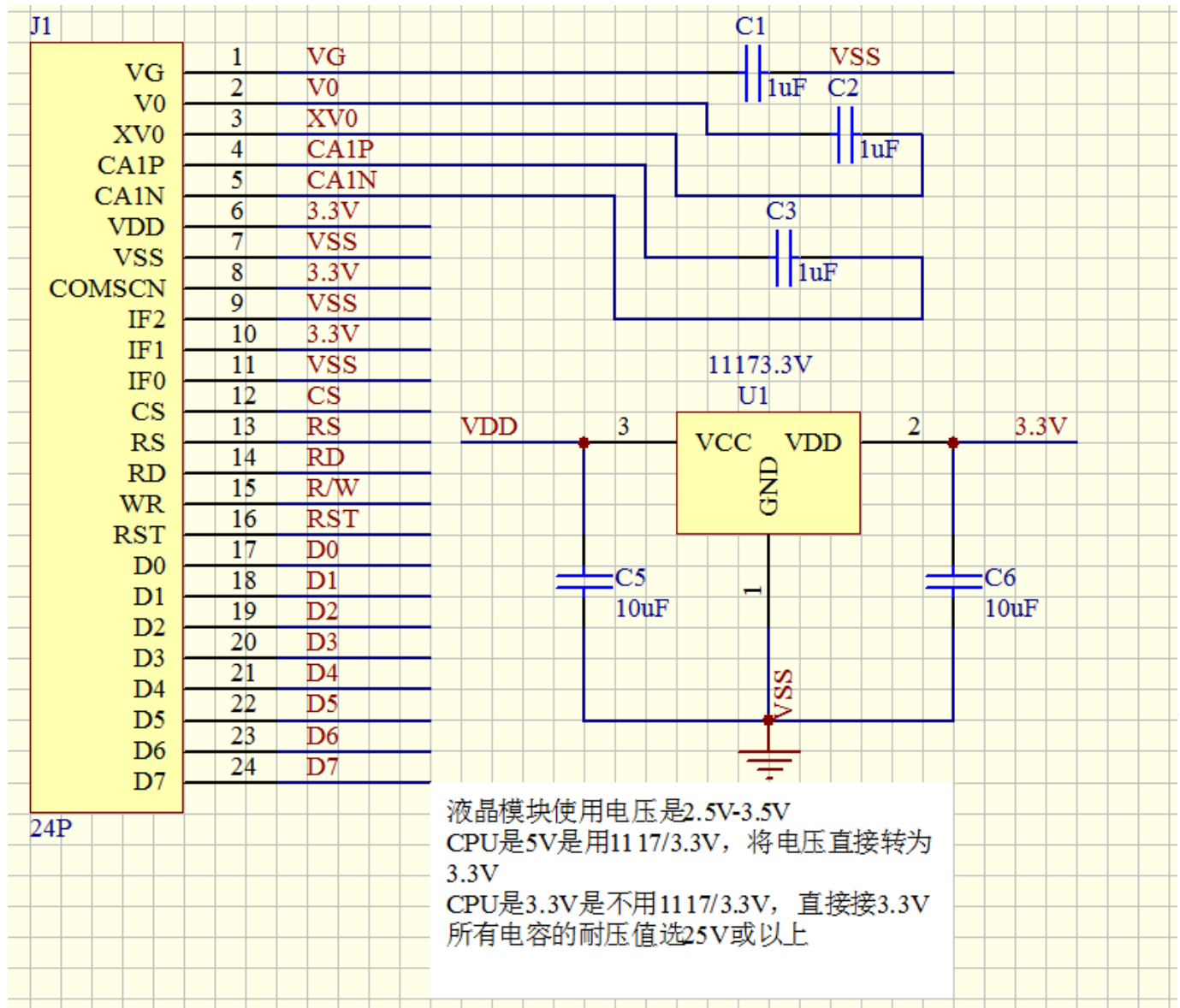


图 9. 并行接口



并行程序与串行只是接口定义、写数据和命令不一样，其它都一样

并行程序:

```
#include <reg52.H>
#include <intrins.h>
#include <chinese_code.h>

sbit lcd_rs=P3^1; /*接口定义:lcd_rs 就是 LCD 的 rs*/
sbit lcd_rd=P3^0; /*接口定义:lcd_e 就是 LCD 的 rd*/
sbit lcd_wr=P3^4; /*接口定义:lcd_rw 就是 LCD 的 wr*/
sbit lcd_reset=P3^5; /*接口定义:lcd_reset 就是 LCD 的 reset*/
sbit lcd_cs1=P3^2; /*接口定义:lcd_cs1 就是 LCD 的 cs1*/
sbit key = P2^0; //按键
```

//写指令到LCD 模块

```
void transfer_command_lcd(int data1)
{
    lcd_cs1=0;
    lcd_rs=0;
    lcd_rd=0;
    lcd_wr=0;
    P1=data1;
    lcd_rd=1;
    lcd_cs1=1;
    lcd_rd=0;
}
```

//写数据到LCD 模块

```
void transfer_data_lcd(int data1)
{
    lcd_cs1=0;
    lcd_rs=1;
    lcd_rd=0;
    lcd_wr=0;
    P1=data1;
    lcd_rd=1;
    lcd_cs1=1;
    lcd_rd=0;
}
```

IIC 接口:

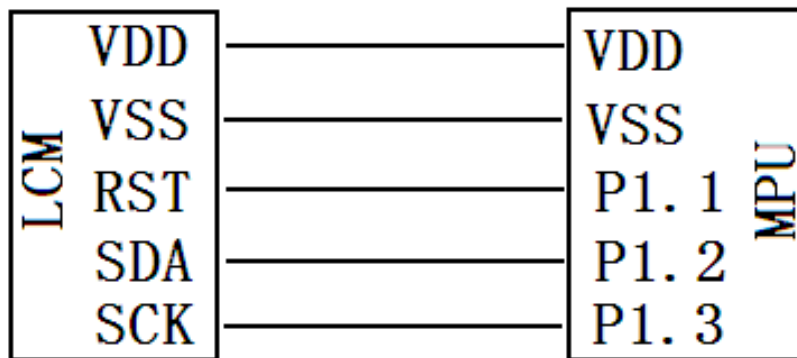
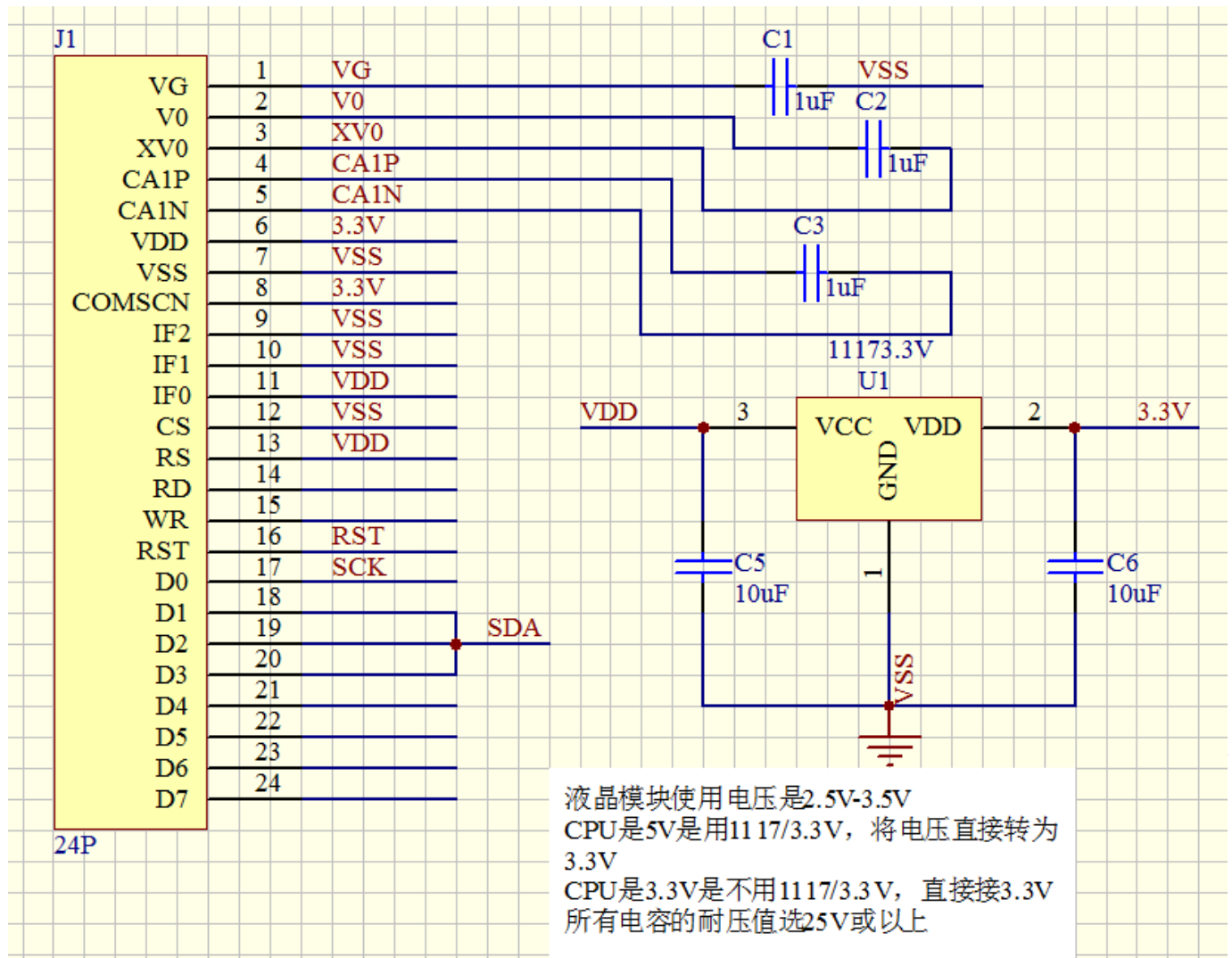


图 10. IIC 接口



IIC 接口与串行、并程序只是接口定义、写数据和命令不一样，其它都一样

7.5.4、以下为 IIC 接口方式范例程序

与串行方式相比较，只需改变接口顺序以及传送数据、传送命令这两个函数即可：

```
/* 液晶模块型号：JLX25664G-580
IIC 接口
驱动 IC 是:ST75256
版权所有：晶联讯电子：网址 http://www.jlxlcd.cn;
*/
#include <reg52.h>
#include <intrins.h>
#include <chinese_code.h>

sbit reset=P1^1;
sbit scl=P1^3;
sbit sda=P1^2;
sbit key=P2^0;
```

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

```
void transfer(int data1)
```

```
{
    int i;
    for(i=0;i<8;i++)
    {
        scl=0;
        if(data1&0x80) sda=1;
        else sda=0;
        scl=1;
        scl=0;
        data1=data1<<1;
    }

    sda=0;
    scl=1;
    scl=0;
}

void start_flag()
{
    scl=1;    /*START FLAG*/
    sda=1;    /*START FLAG*/
    sda=0;    /*START FLAG*/
}

void stop_flag()
{
    scl=1;    /*STOP FLAG*/
    sda=0;    /*STOP FLAG*/
    sda=1;    /*STOP FLAG*/
}

//写命令到液晶显示模块
void transfer_command_lcd (uchar com)
{
    start_flag();
    transfer(0x78);
    transfer(0x80);
    transfer(com);
    stop_flag();
}

//写数据到液晶显示模块
void transfer_data_lcd (uchar dat)
{
    start_flag();
    transfer(0x78);
    transfer(0xC0);
    transfer(dat);
    stop_flag();
}
```