JLX160160G-162-PN 使用说明书

目 录

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 JLX160160G-162 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。

JLX160160G-162 可以显示 160 列*160 行点阵单色图片，或显示 10 个×10 行=100 个 16*16 点阵的汉字，或显示 20 个×10 行=200 个 8*16 点阵的英文、数字、符号，或显示 32 个×20 行的点阵的英文、数字、符号

2．JLX160160G-162 图像型点阵液晶模块的特性

2.1 结构轻、薄、带背光、带黑铁框、焊接式 FPC。

2.2 IC 采用 ST75161,功能强大，稳定性好

2.3 功耗低:当电压为 3.3V 时，功耗低：不带背光 1.29mW（3.3V*（0.39mA 测试最大值）），带背光不大于 270mW（3.3V*45mA）；

2.4 显示内容:

（1）160*160 点阵单色图片，或其它小于 160*160 点阵的单色图片;

（2）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 13 字*13 行；

（3）按照 8*16 点阵汉字来计算可显示 20 字*10 行；

（4）按照 5*8 点阵汉字来计算可显示 32 字*20 行；

（5）可选用 16*16 点阵或其它点阵的图片自编汉字，也可配合晶联讯字库 IC（JLX-GB2312-3207）来显示汉字；

2.5 指令功能强:可软件调对比度、正显/反显转换、行列扫描方向可改（可旋转 180 度使用）。

2.6 接口简单方便:并行（6800 时序或 8080 时序可选）、串行、IIC 接口；

2.7 工作温度宽:-20℃ – +70℃;

2.8 储存温度宽:-30℃ – +80℃;

3. 外形尺寸及接口引脚功能

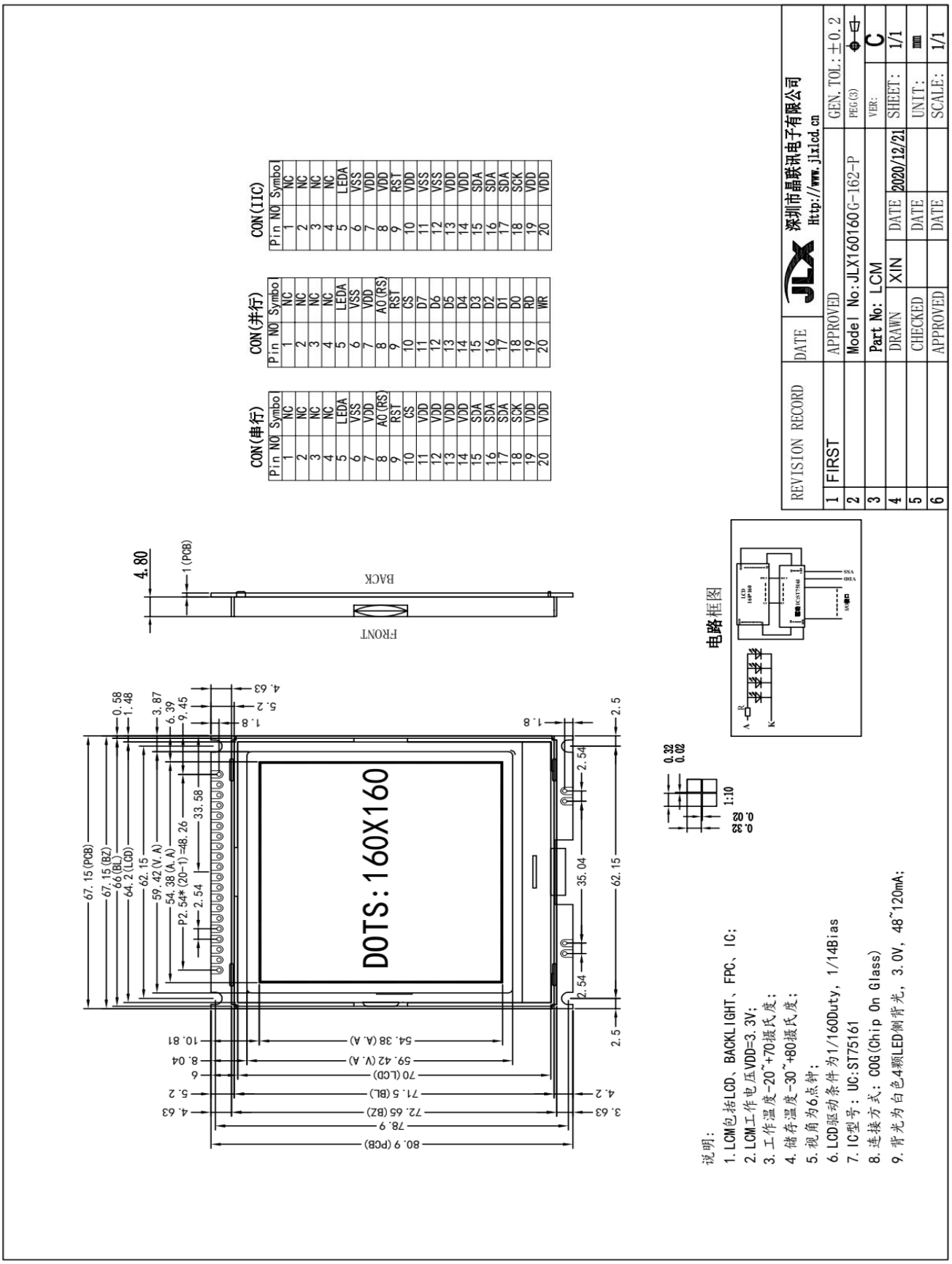


图 1. 外形尺寸

模块的接口引脚功能：

引 线 号	符 号	名 称	功 能
1	NC	NC	空脚
2	NC	NC	空脚
3	NC	NC	空脚
4	NC	NC	空脚
5	LEDA	背光电源	供电电源正极(同 VDD 电压)
6	VSS	接地	0V
7	VDD	电路电源	供电电源正极接(5V 或 3.3V 购买时须选择 3.3V 还是 5.0 供电)
8	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”） IIC 接口时：接 VDD
9	RST	复位	低电平复位，复位完成后，回到高电平，液晶模块开始工作
10	CS	片选	低电平片选 IIC 接口时：接 VDD
11	D7	I/O	并行接口时：数据总线 DB7 串行接口时：接高电平 IIC 接口时：接低电平
12	D6	I/O	并行接口时：数据总线 DB6 串行接口时：接高电平 IIC 接口时：接低电平
13	D5	I/O	并行接口时：数据总线 DB5 IIC/串行接口时：接 VDD
14	D4	I/O	并行接口时：数据总线 DB4 IIC/串行接口时：接 VDD
15	D3	I/O	并行接口时：数据总线 DB3 IIC/串行接口时：为 SDA 串行数据（D1、D2、D3 短接）
16	D2	I/O	并行接口时：数据总线 DB2 IIC/串行接口时：为 SDA 串行数据（D1、D2、D3 短接）
17	D1	I/O	并行接口时：数据总线 DB1 IIC/串行接口时：为 SDA 串行数据（D1、D2、D3 短接）
18	D0	I/O	并行接口时：数据总线 DB0 IIC/串行接口时：为 SCK 串行时钟
19	E(/RD)	6800 时序:使能 8080 时序：读	并行接口时并且选择 6800 时序时：使能信号,高电平有效. 并行接口时并且选择 8080 时序时：读数据,低电平有效. IIC/串行接口时：接 VDD
20	R/W(/WR)	6800 时序:读/写 8080 时序：写	并行接口时并且选择 6800 时序时：H:读数据 L:写数据 并行接口时并且选择 8080 时序时：写数据,低电平有效. IIC/串行接口时：接 VDD

表 1：模块的接口引脚功能

4. 基本原理

4.1 液晶屏（LCD）

在 LCD 上排列着 160*160 点阵,160 个列信号与驱动 IC 相连，160 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：

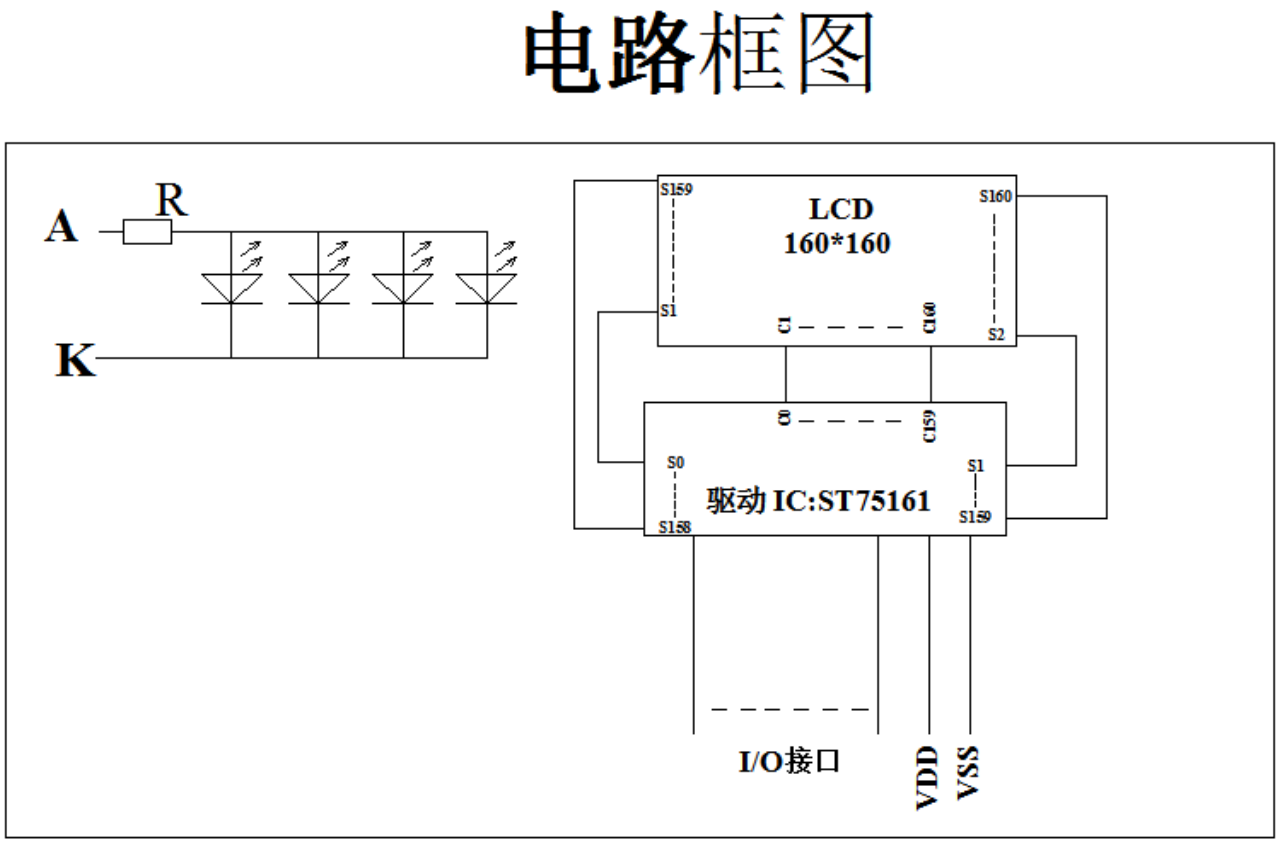


图 2:JLX160160G-162 图像点阵型液晶模块的电路框图

4.2 背光参数

该型号液晶模块带 LED 背光源。它的性能参数如下:
背光板可选择白色。
正常工作电流为：48∽120mA（LED 灯数共 4 颗）;
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	3.3	3.6	V
LCD 驱动电路	V0-XV0	-0.3	15.5	19	V
工作温度		-20		+70	℃
储存温度		-30		+80	℃

表 2：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	VDD		2.6	3.3	3.6	V
背光工作电压	VLED		2.9	3.0	3.1	V
输入高电平	VIH	-	0.8xVDD		VDD	V
输入低电平	VIO	-	VSS		0.2	V
输出高电平	VOH	IOH = 0.2mA	0.8xVDD		VDD	V
输出低电平	VOO	IOO = 1.2mA	VSS		0.2xVDD	V
模块工作电流	IDD	VDD = 3.0V	-		0.3	mA
背光工作电流	ILED	VLED=3.0V	24	60	90	mA

表 3：直流（DC）参数

6. 读写时序特性

6.1 串行接口：

从 CPU 写到 ST75161（Writing Data from CPU to ST75161）

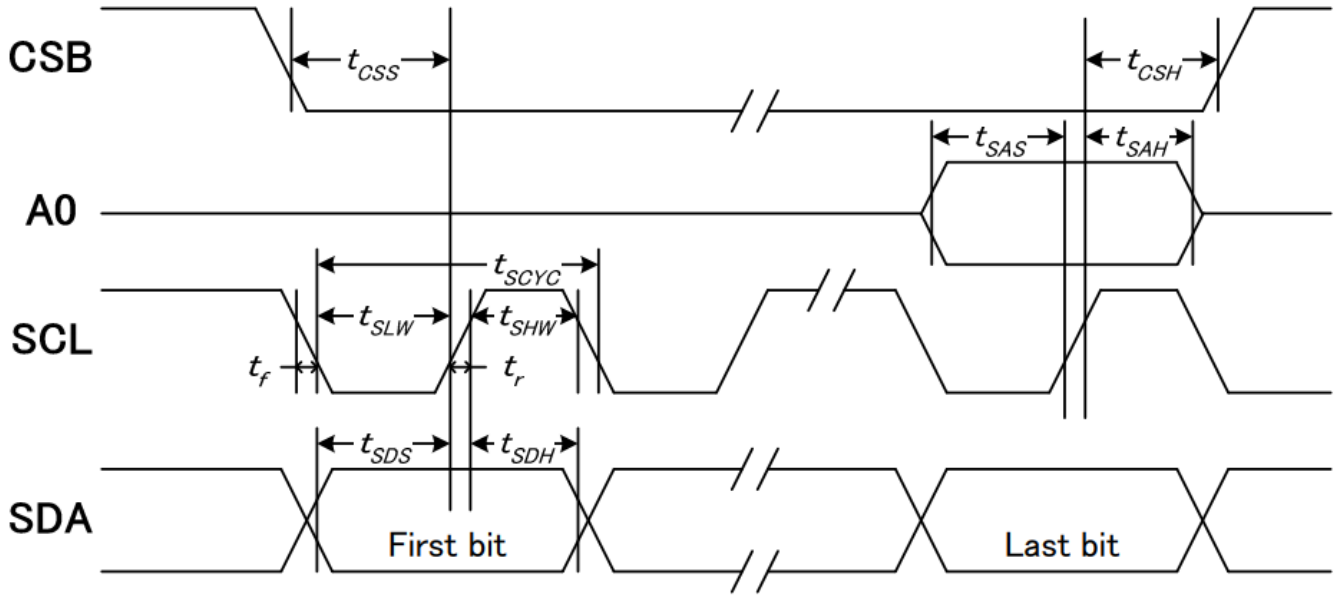


图 3. 从 CPU 写到 ST75161（Writing Data from CPU to ST75161）

6.2 串行接口：时序要求（AC 参数）：

写数据到 ST75161 的时序要求：

VDD1 = 1.8~3.3V, Ta = –45~85 ℃

Item	Signal	Symbol	Condition	Min.	Max.	Unit
Serial clock period	SCLK	tSCYC		80	—	ns
SCLK “H” pulse width		tSHW		30	—	
SCLK “L” pulse width		tSLW		30	—	
Address setup time	A0	tSAS		20	—	
Address hold time		tSAH		20	—	
Data setup time	SDA	tSDS		20	—	
Data hold time		tSDH		20	—	
CSB-SCLK time	CSB	tCSS		20	—	
CSB-SCLK time		tCSH		20	—	
CS “H” pulse width		tCHW		0	—	

Note:

1. The input signal rise and fall time (tr, tf) are specified at 15 ns or less.
2. All timing is specified using 20% and 80% of VDD1 as the standard.

表 4

6.3 并行接口：（8080）

从 CPU 写到 ST75161（Writing Data from CPU to ST75161）

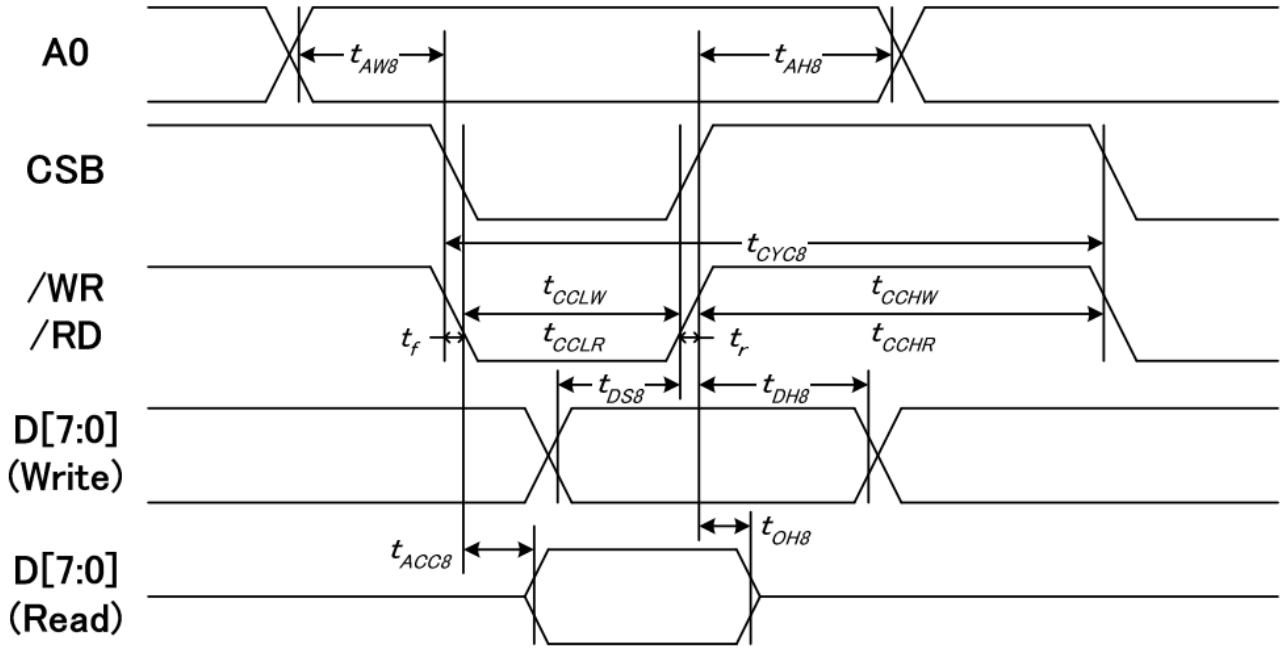


图 4. 从 CPU 写到 ST75161（Writing Data from CPU to ST75161）

6.4 并行接口：时序要求（AC 参数）：

写数据到 ST75161 的时序要求：（8080 系列 MPU）

VDD1 = 1.8~3.3V, Ta = –45~85 ℃

Item	Signal	Symbol	Condition	Min.	Max.	Unit
Address setup time	A0	tAW8		20	—	ns
Address hold time		tAH8		0	—	
System cycle time (WRITE)	/WR	tCYC8		160	—	
/WR L pulse width (WRITE)		tCCLW		70	—	
/WR H pulse width (WRITE)		tCCHW		70	—	
System cycle time (READ)	RD	tCYC8		400	—	
/RD L pulse width (READ)		tCCLR		180	—	
/RD H pulse width (READ)		tCCHR		180	—	
WRITE Data setup time	D[7:0]	tDS8		15	—	
WRITE Data hold time		tDH8		15	—	
READ access time		tACC8	CL = 30 pF	—	100	
READ Output disable time		tOH8	CL = 30 pF	10	110	

Note:

1. The input signal rise time and fall time (tr, tf) is specified at 15 ns or less. When the system cycle time is extremely fast, (tr + tf) ≦ (tCYC8 – tCCLW – tCCHW) for (tr + tf) ≦ (tCYC8 – tCCLR – tCCHR) are specified.
2. All timing is specified using 20% and 80% of VDD1 as the reference.

表 5

6.5 并行接口：（6800）

从 CPU 写到 ST75161（Writing Data from CPU to ST75161）

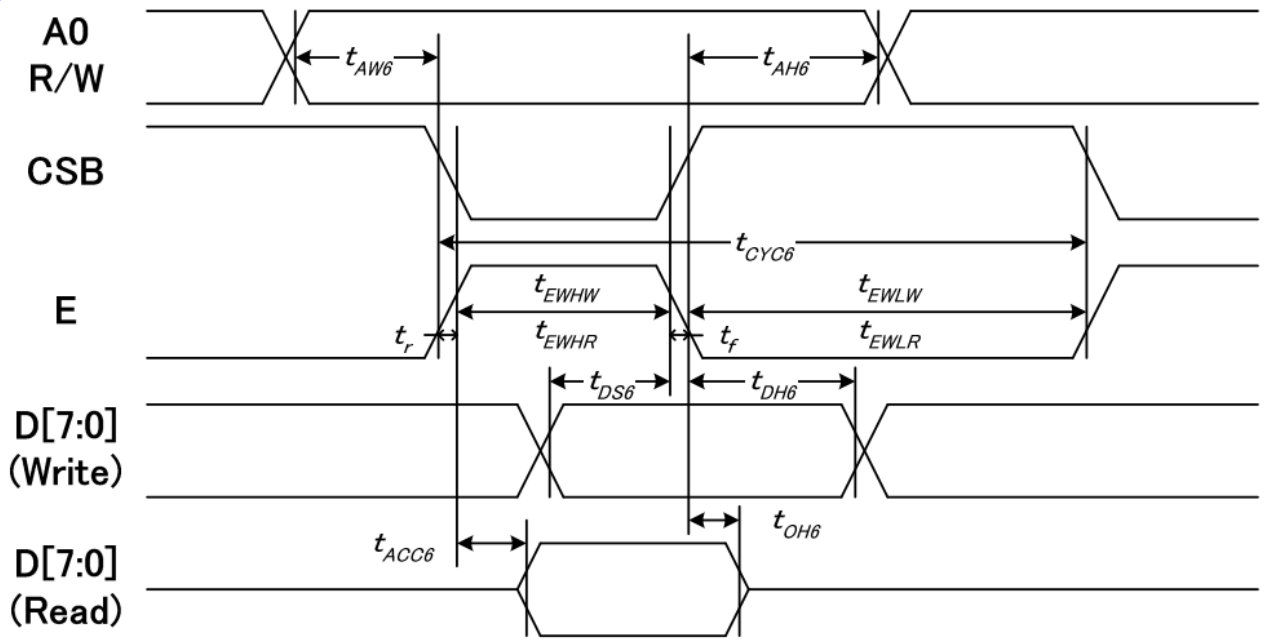


图 5. 从 CPU 写到 ST75161（Writing Data from CPU to ST75161）

6.6 并行接口：时序要求（AC 参数）：
写数据到 ST75161 的时序要求：（6800 系列 MPU）

VDD1 = 1.8~3.3V, Ta = –45~85 ℃

Item	Signal	Symbol	Condition	Min.	Max.	Unit
Address setup time	A0	tAW6		20	—	ns
Address hold time		tAH6		0	—	
System cycle time (WRITE)	E	tCYC6		160	—	
Enable L pulse width (WRITE)		tEWLW		70	—	
Enable H pulse width (WRITE)		tEWHW		70	—	
System cycle time (READ)		tCYC6		400	—	
Enable L pulse width (READ)		tEWLR		180	—	
Enable H pulse width (READ)		tEWHR		180	—	
Write data setup time	D[7:0]	tDS6		15	—	
Write data hold time		tDH6		15	—	
Read data access time		tACC6	CL = 30 pF	—	100	
Read data output disable time		tOH6	CL = 30 pF	10	110	

Note:

1. The input signal rise time and fall time (tr, tf) is specified at 15 ns or less. When the system cycle time is extremely fast, (tr + tf) ≦ (tCYC6 – tEWLW – tEWHW) for (tr + tf) ≦ (tCYC6 – tEWLR – tEWHR) are specified.
2. All timing is specified using 20% and 80% of VDD1 as the reference.

表 6

6.7 IIC 接口：
从 CPU 写到 ST75161（Writing Data from CPU to ST75161）

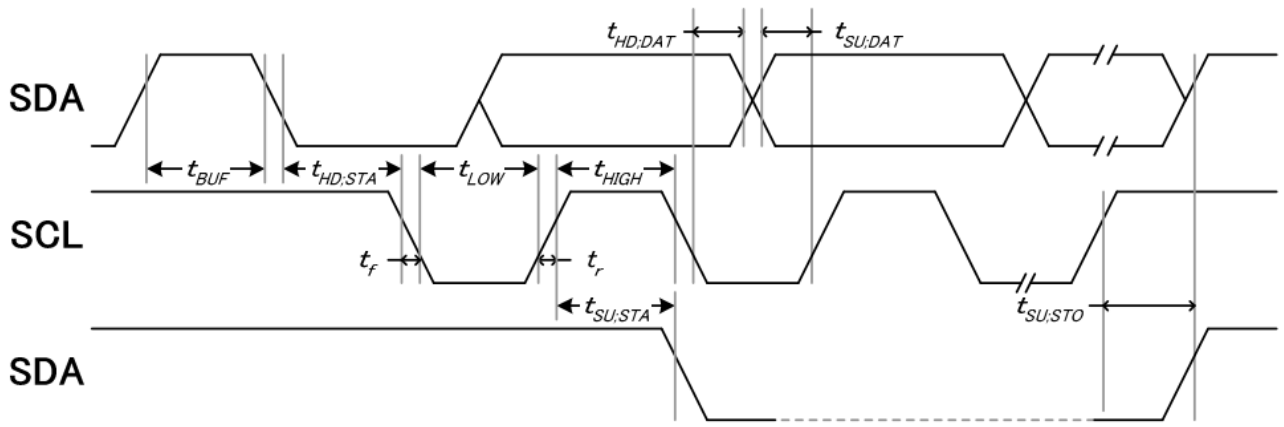


图 6. **从 CPU 写到 ST75161（Writing Data from CPU to ST75161）**

6.8 IIC 接口：时序要求（AC 参数）：

写数据到 ST75161 的时序要求：

VDD1 = 1.8~3.3V, Ta = –45~85 ℃

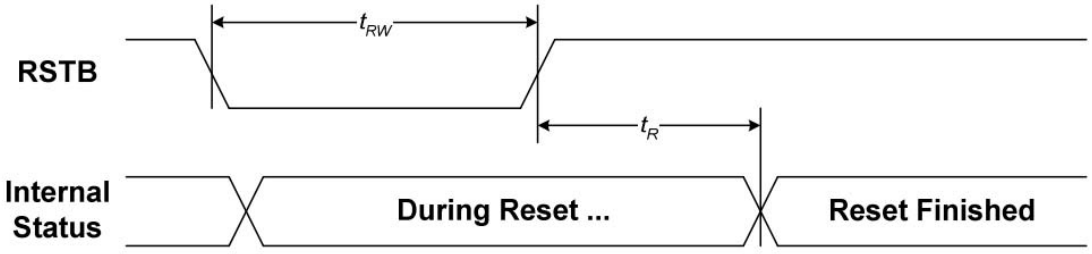
Item	Signal	Symbol	Condition	Rating		Unit
				Min.	Max.	
SCL clock frequency	SCL	fSCL		—	400	KHZ
SCL clock low period		tLOW		1.3	—	us
SCL clock high period		tHIGH		0.6	—	
Data set-up time	SDA	tSU;Data		0.1	—	
Data hold time		tHD;Data		0	0.9	
Setup time for a repeated START condition		tSU;STA		0.6	—	
Start condition hold time		tHD;STA		0.6	—	
Setup time for STOP condition		tSU;STO		0.6	—	
Bus free time between a STOP and START		tBUF		0.1	—	
Signal rise time	SCL SDA	tr		20+0.1Cb	300	ns
Signal fall time		tf		20+0.1Cb	300	
Capacitive load represented by each bus line		Cb		—	400	pF
Tolerable spike width on bus		tSW		—	50	ns

Note:

1. All timing is specified using 20% and 80% of VDD1 as the standard.

表 7

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



VDD1 = 1.8~3.3V, Ta = –45~85 ℃

Item	Symbol	Condition	Min.	Max.	Unit
Reset time	tR		—	1	ms
Reset “L” pulse width	tRW		1	—	ms

图 7：**电源启动后复位的时序**

表 8

Item	Symbol	Condition	Min.	Max.	Unit
Reset time	tR		-	1	ms
Reset “L” pulse width	tRW		1	-	

7．指令功能：

7.1 指令表

下表是“ST75161”IC 支持的指令：

CD:0:指令； 1:数据 W/R: 0:写； 1:读 D7~D0:有用的数据位； -：不必理会的

表 9.

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≦27h Ending page address: YS≦YE≦27h
	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≦9Fh Ending column address: XS≦XE≦9Fh
	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	0	
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^2C)
	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≦9Fh Ending partial display address: 00h≦PTE≦9Fh
	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≦SL≦9Fh
	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
Ext[1:0]=0,1 (Extension Command 2)											
30. 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	
	1	0	0	0	0	0	0	0	0	0	
31.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	
32.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	
33. Driving Select	0	0	0	1	0	0	0	0	0	DS	Power type DS=0: Internal (Default) DS=1 :External
34.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	
35.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	
36.OTP Control Out	0	0	1	1	1	0	0	0	0	1	OTP control out
37.OTP Write	0	0	1	1	1	0	0	0	1	0	OTP write
38.OTP Read	0	0	1	1	1	0	0	0	1	1	OTP read

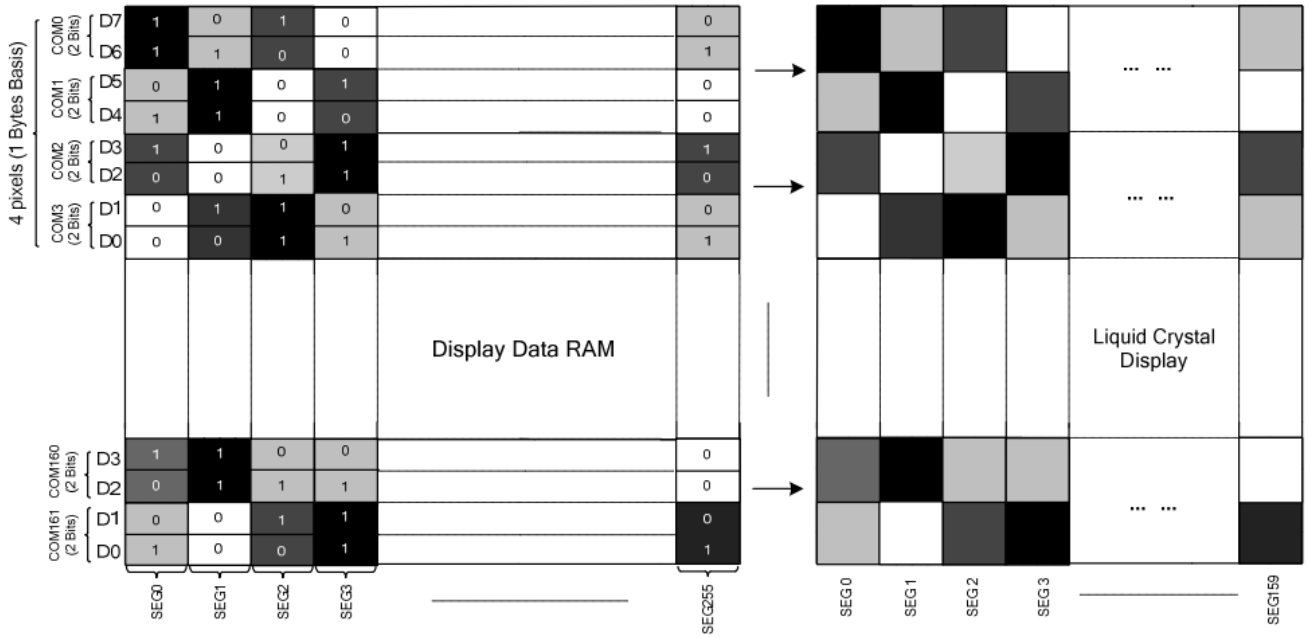
INSTRUCTION	A0	R/W	COMMAND BYTE								DESCRIPTION
			D7	D6	D5	D4	D3	D2	D1	D0	
39.OTP Selection Control	0	0	1	1	1	0	0	1	0	0	OTP selection control Ctrl=1: Disable OTP Selection Ctrl=0: Enable OTP Selection
	1	0	1	Ctrl	0	1	1	0	0	1	
40.OTP Programming Setting	0	0	1	1	1	0	0	1	0	1	OTP programming setting
	1	0	0	0	0	0	1	1	1	1	
41.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	
42.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	
43.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]=1,1(Extension Command 4)											
44.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	

请详细参考 IC 资料”ST75161-G2A_V1.2a PDF”的第 55～77 页。

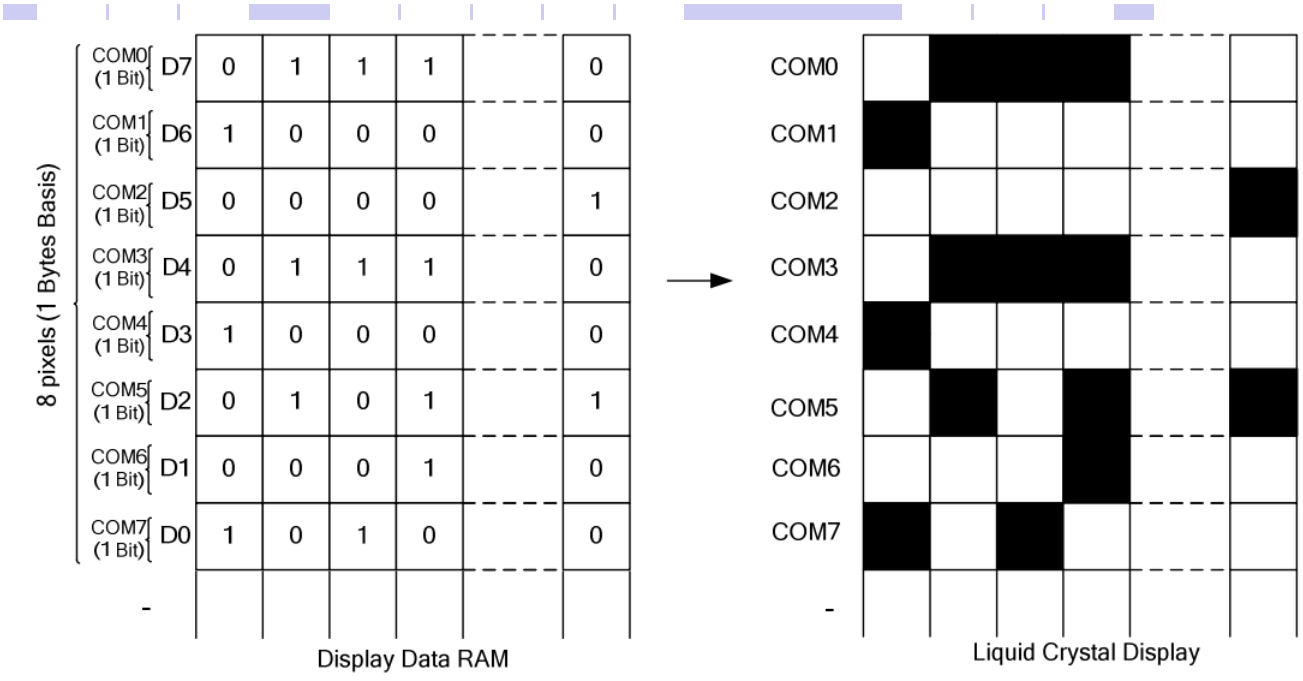
7.3 点阵与 DD RAM(显示数据存储器)地址的对应关系

ST75161 包含 160 x161x2 位静态 RAM 存储显示数据。数据显示 RAM (DDRAM)

商店液晶显示器的像素数据。内置的 DDRAM 与 160 列可寻址内存数组 161 行。ST75161 提供 2 种显示模式(4-Gray /单色规模模式)和一个 fast-addressing 模式快速更新显示数据。每个页面地址代表不同 sub-COMs 不同的显示模式。例如,在 4-Gray /单色规模模式设置页面地址“00 h”意味着即将到来的 8 位数据寻址 COM0 ~ COM3 / COM0 ~ COM7。列地址到赛格输出数量直接相关。LCD 控制器读取像素 DDRAM 中的数据,然后输出 COM/SEG 垫。而液晶控制器独立运作,显示数据可以写入 DDRAM 同时数据也被显示在 LCD 面板不会引起异常显示。如下图所示：



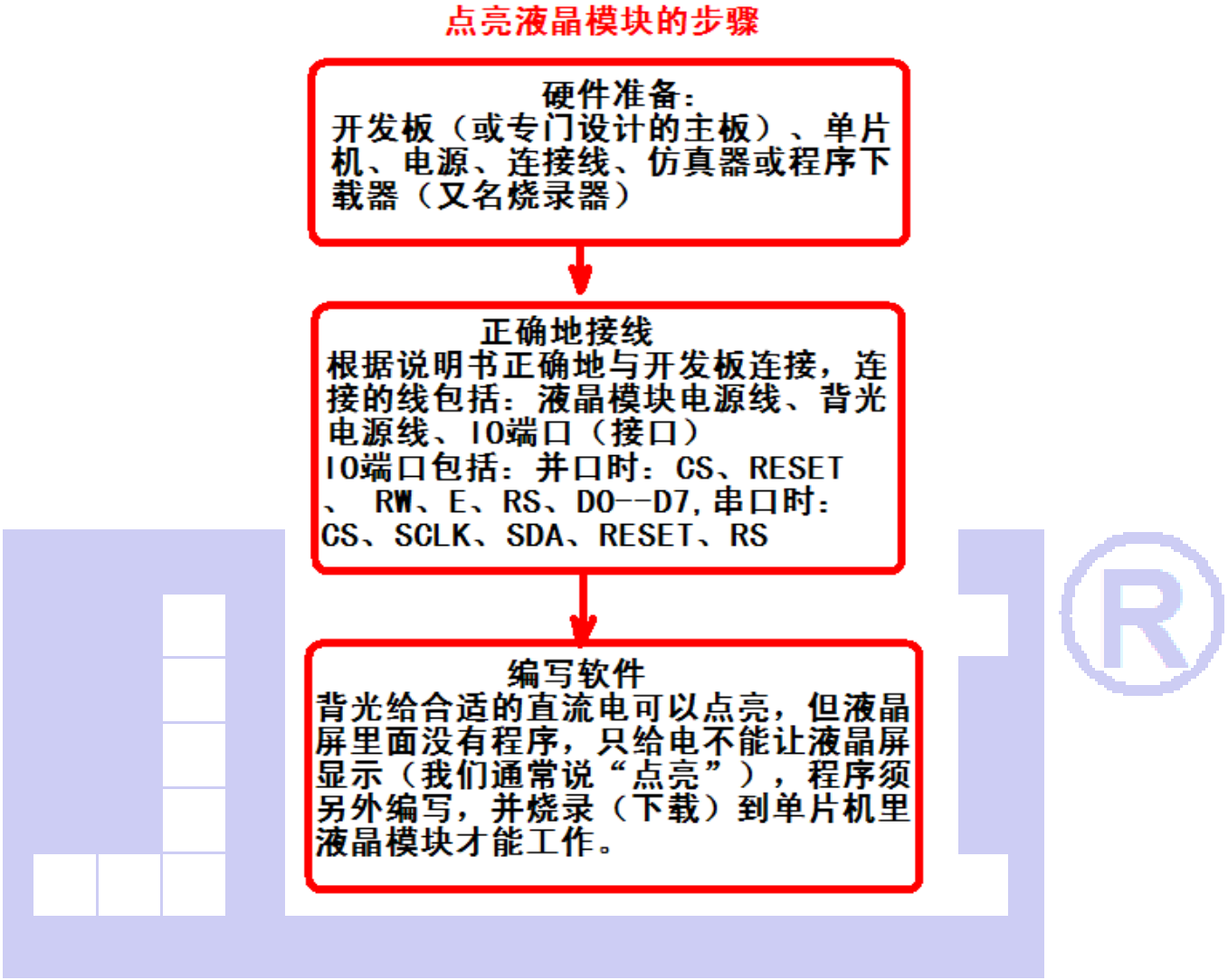
2 Bits Data N=0~3		DDRAM		LCD
D2N+1	D2N			
1	1	1	1	
0	0	0	0	
1	0	1	0	
0	1	0	1	



下图摘自 ST75161 IC 资料，可通过“ST75161-G2A_V1.2a pdf”第 38～49 页。。

7.4 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



7.5 程序举例:

液晶模块与 MCU(以 8051 系列单片机为例)接口图如下:

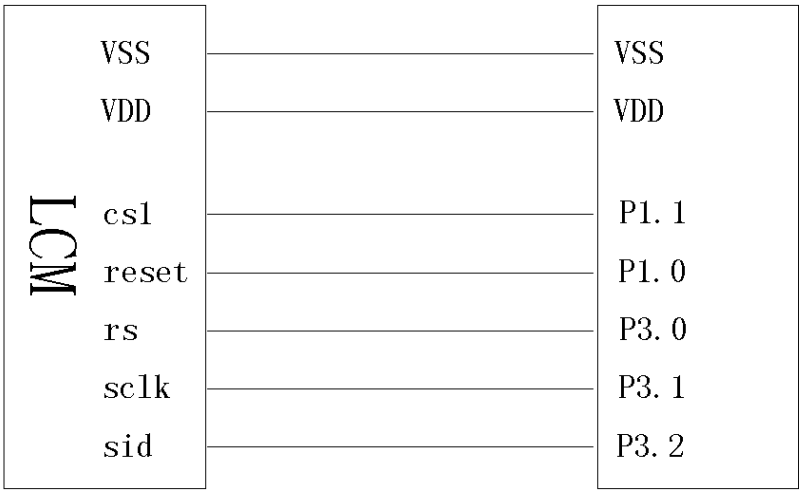


图 8. 串行接口

7.5.1 程序

```
/*  液晶模块型号：JLX256160G-162          串行接口
    驱动 IC 是:ST75161
    版权所有：晶联讯电子：网址  http://www.jlxlcd.cn;
*/
#include <STC15F2K60S2.H>
#include <intrins.h>
#include <chinese_code.h> //此文件购买后联系销售人员索要

sbit lcd_cs1  =  P3^4;//CS
sbit lcd_reset=  P3^5;//RST
sbit lcd_sclk =  P1^0;//串行时钟
sbit lcd_rs   =  P3^3;//RS
sbit lcd_sid  =  P1^1;//串行数据

sbit key=P2^0;       /*按键接口，P2.0 口与 GND 之间接一个按键*/

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

//写指令到 LCD 模块
void transfer_command(int data1)
{
    char i;
    cs1=0;
    rs=0;
    for(i=0;i<8;i++)
    {
        sclk=0;
        if(data1&0x80) sid=1;
        else sid=0;
        sclk=1;
        data1=data1<<=1;
    }
    cs1=1;
}

//写数据到 LCD 模块
void transfer_data(int data1)
{
```

```
    char i;
    cs1=0;
    rs=1;
    for(i=0;i<8;i++)
    {
        sclk=0;
        if(data1&0x80) sid=1;
        else sid=0;
        sclk=1;
        data1=data1<<=1;
    }
    cs1=1;
}

void waitkey()
{
repeat:
    if(key==1)goto repeat;
    else delay(2500);
}

//------对比度值设置,粗调 0x05,微调 0x23------//
void initial_lcd()
{
    lcd_reset=0;
    delay(100);
    lcd_reset=1;
    delay(500);
    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(0x00);      //Bias=1/14
    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(0x31);   //EXT=1
    transfer_command_lcd(0xf0);   //此指令比较重要,不加此指令升压会慢 0.5s
    transfer_data_lcd(0x0f);
    transfer_data_lcd(0x0f);
    transfer_data_lcd(0x0f);
    transfer_data_lcd(0x0f);

    transfer_command_lcd(0x30);   //EXT=0
    transfer_command_lcd(0x75);   //Page Address setting
    transfer_data_lcd(0X00);      // XS=0
    transfer_data_lcd(0X28);      // XE=159 0x28
    transfer_command_lcd(0x15);   //Clumn Address setting
    transfer_data_lcd(0X00);      // XS=0
    transfer_data_lcd(0Xff);      // XE=256
    transfer_command_lcd(0xBC);   //Data scan direction
    transfer_data_lcd(0x00);      //MX.MY=Normal
    transfer_data_lcd(0xA6);
    transfer_command_lcd(0xCA);   //Display Control
    transfer_data_lcd(0X00);       //
    transfer_data_lcd(0X9F);       //Duty=160
    transfer_data_lcd(0X20);       //Nline=off
    transfer_command_lcd(0xF0);   //Display Mode
    transfer_data_lcd(0X10);       //10=Monochrome Mode,11=4Gray
    transfer_command_lcd(0x81);   //EV control
    transfer_data_lcd(0x23);       //微调对比度的值，可设置范围 0x00～0x3f
    transfer_data_lcd(0x05);       //粗调对比度，可设置范围 0x00～0x07
    transfer_command_lcd(0x20);   //Power control
    transfer_data_lcd(0x0B);       //D0=regulator ; D1=follower ; D3=booste,  on:1 off:0
    delay(20);
    transfer_command_lcd(0xAF);   //Display on
}
```

/*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;
    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(0,1,160,20);
    for(i=0;i<20;i++)
    {
        for(j=0;j<160;j++)
        {
            transfer_data_lcd(0x00);
        }
    }
}
```

```
void test(uchar data1,uchar data2)
{
    int i,j;
    lcd_address(0,1,160,20);
    for(i=0;i<20;i++)
    {
        for(j=0;j<160;j++)
        {
            transfer_data_lcd(data1);
            transfer_data_lcd(data2);
        }
    }
}
```

```
//显示 8x16 的点阵的字符串，括号里的参数分别为（页,列，字符串指针）
void display_string_8x16(uint page,uint column,uchar reverse,uchar *text)
{

    uint i=0,j,k,n;
    if(column>248)
    {
        column=1;
        page+=2;
    }
    while(text[i]>0x00)
    {
        if((text[i]>=0x20)&&(text[i]<=0x7e))
        {
            j=text[i]-0x20;
            for(n=0;n<2;n++)
            {
                lcd_address(column,page+n,256,8);
                for(k=0;k<8;k++)
                {
            if(reverse==1)
                    {
             transfer_data_lcd(~ascii_table_8x16[j][k+8*n]);    //写数据到 LCD,每写完 1 字节的数据后列地址自动加 1
                    }
                else
                    {
                        transfer_data_lcd(ascii_table_8x16[j][k+8*n]);  //写数据到LCD,每写完 1 字节的数据后列地址自动加 1
                    }
                }
            }
            i++;
            column+=8;
        }
        else
        i++;
    }
}

//显示 5x8 的点阵的字符串，括号里的参数分别为（页,列，字符串指针）
void display_string_5x8(uint page,uint column,uchar reverse,uchar *text)
{
    uint i=0,j,k,disp_data;
    while(text[i]>0x00)
    {
```

```
        if((text[i]>=0x20)&&(text[i]<=0x7e))
        {
            j=text[i]-0x20;
            lcd_address(column,page,256,8);
            for(k=0;k<5;k++)
            {
                if(reverse==1)
                {
                    disp_data=~ascii_table_5x8[j][k];
                }
                else
                {
                    disp_data=ascii_table_5x8[j][k];
                }

                transfer_data_lcd(disp_data);   //写数据到 LCD,每写完 1 字节的数据后列地址自动加 1
            }
            if(reverse==1)  transfer_data_lcd(0xff);    //写入一列空白列，使得 5x8 的字符与字符之间
有一列间隔，更美观
            else    transfer_data_lcd(0x00);        //写入一列空白列，使得 5x8 的字符与字符之间
有一列间隔，更美观
            i++;
            column+=6;
            if(column>251)
            {
                column=1;
                page++;
            }
        }
        else
        i++;
    }
}


//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 256*160 点阵的图像*/
void disp_160x160(int x,int y,char *dp)
{
    int i,j;
    lcd_address(x,y,160,20);
    for(i=0;i<20;i++)
    {
        for(j=0;j<160;j++)
        {
            transfer_data_lcd(*dp);
            dp++;
        }
    }
}


//-----------------------------
void main()
{
  P1M1=0x00;
    P1M0=0x00;   //P1 配置为准双向
    P2M1=0x00;
    P2M0=0x00;   //P2 配置为准双向
    P3M1=0x00;
    P3M0=0x00;   //P3 配置为准双向
    initial_lcd();          //对液晶模块进行初始化设置
    while(1)
    {
        clear_screen();             //清屏
```

```
        transfer_command_lcd(0x0c);   //低位在前
        disp_160x160(0,1,bmp2);       //显示一幅 160*160 点阵的黑白图。
        waitkey();
        clear_screen();               //清屏
        disp_160x160(0,1,bmp1);       //显示一幅 160*160 点阵的黑白图。
        waitkey();

        clear_screen();
        transfer_command_lcd(0x08);   //高位在前
        display_string_16x16(16,4,"深圳市晶联讯电子");
        disp_32x32(0,8,jing2);
        disp_32x32((32*1+0),8,lian2);
        disp_32x32((32*2+0),8,xun2);
        disp_32x32((32*3+0),8,dian2);
        disp_32x32((32*4+0),8,zi2);
        waitkey();
        clear_screen();               //清屏
        disp_160x160(0,1,bmp3);       //显示一幅 160*160 点阵的黑白图。
        waitkey();
    }
}
```

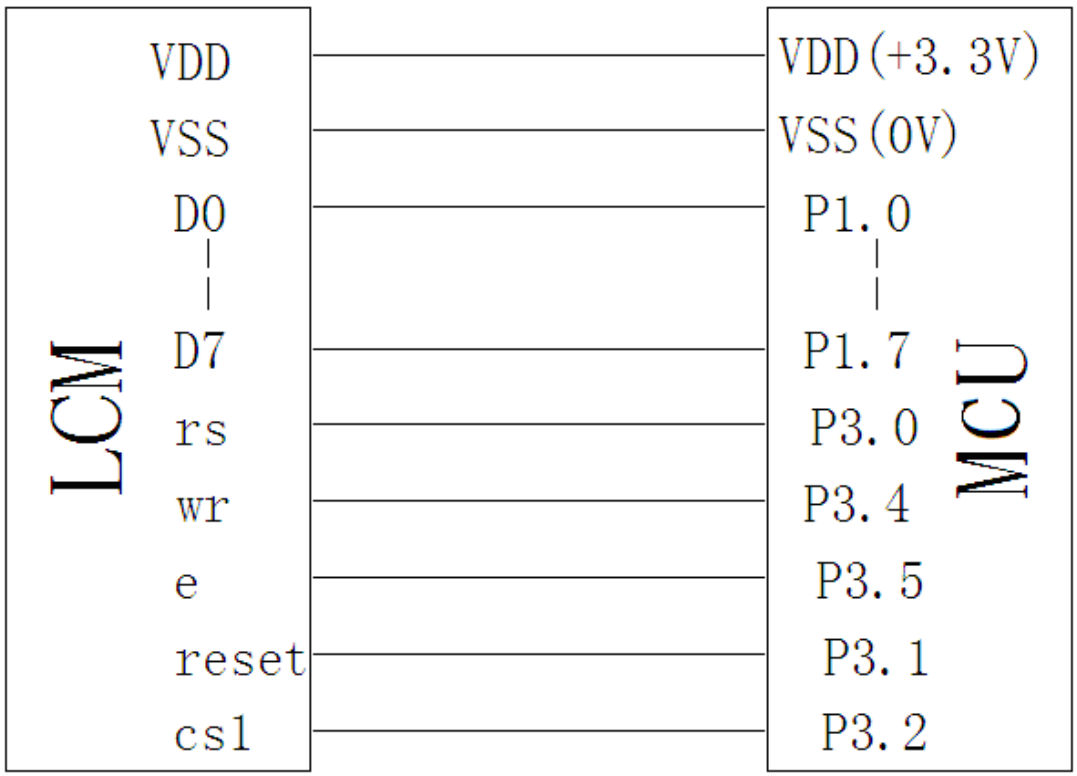


图 9. 并行接口

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：

```
/*  液晶模块型号：JLX256160G-162
```

```
    并行接口 6800 时序
    驱动 IC 是:ST75161
    版权所有：晶联讯电子：网址  http://www.jlxlcd.cn;
*/
#include <STC15F2K60S2.H>
#include <intrins.h>
#include <chinese_code.h>

sbit wr=P2^1;    //接口定义:lcd_rw 就是 LCD 的 wr,有些图纸上写“WR”
sbit rd=P3^0;     //接口定义:lcd_e 就是 LCD 的 rd
sbit rs=P3^3;    //接口定义:lcd_rs 就是 LCD 的 rs,有些图纸上写“A0”\“DC”\"DC0", 都是它
sbit cs1=P3^4;   //接口定义:lcd_cs1 就是 LCD 的 cs1
sbit reset=P3^5; //接口定义:lcd_reset 就是 LCD 的 reset
sbit key=P2^0;       //P2.0 口与 GND 之间接一个按键

//=============transfer command to LCM===============
void transfer_command_lcd(int data1)
{
    cs1=0;
    rs=0;
    rd=0;
    wr=0;
    P1=data1;
    rd=1;
    delay_us(1);
    rd=0;
    cs1=1;
}

//-----------transfer data to LCM----------------
void transfer_data_lcd(int data1)
{
    cs1=0;
    rs=1;
    rd=0;
    wr=0;
    P1=data1;
    rd=1;
    delay_us(1);
    rd=0;
    cs1=1;
}
/*  液晶模块型号：JLX256160G-162
    并行接口     8080 时序
    驱动 IC 是:ST75161
    版权所有：晶联讯电子：网址  http://www.jlxlcd.cn;
```

```
*/
#include <STC15F2K60S2.H>
#include <intrins.h>
#include <chinese_code.h>


sbit CS=P3^4;          /*对应 LCD 的 CS 引脚*/
sbit RST=P3^5;     /*对应 LCD 的 RST 引脚*/
sbit RS=P3^3;            /*对应 LCD 的 RS 引脚*/
sbit E=P3^0;             /*对应 LCD 的 E(RD)引脚*/
sbit RW=P2^1;            /*对应 LCD 的 RW(WR)引脚。另外 P1.0~1.7 对应 DB0~DB7*/
sbit key=P2^0;       /*按键接口，P2.0 口与 GND 之间接一个按键*/

#define uchar unsigned char
#define uint unsigned int


/*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);
}


//===============8080 时序============================
//=============transfer command to LCM===============
```

```
void transfer_command_lcd(int data1)
{
    CS=0;
    RS=0;
    E=1;
    delay_us(1);
    RW=0;
    P1=data1;
    RW=1;
    delay_us(1);
    RS=1;
    CS=1;
    P1=0;
}

//-----------transfer data to LCM---------------
void transfer_data_lcd(int data1)
{
    CS=0;
    RS=1;
    E=1;
    delay_us(1);
    RW=0;
    P1=data1;
    RW=1;
    delay_us(1);
    RS=1;
    CS=1;
    P1=0;
}
```

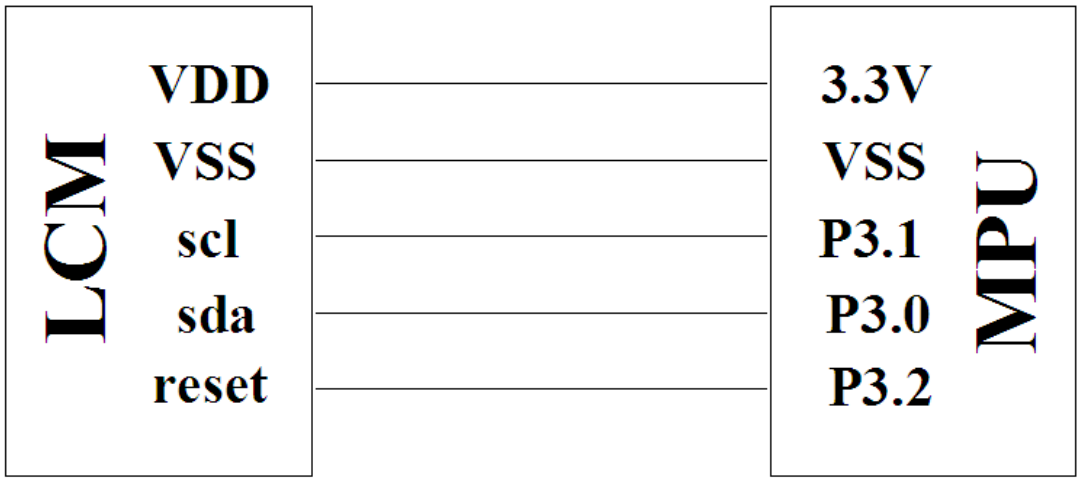


图 10. IIC 接口

IIC 程序与串、并行接口定义、写数据和命令不一样，取模代码是一样的

IIC 程序：

```
/*  液晶模块型号：JLX256160G-162        IIC 行接口
    驱动 IC 是:ST75161
    版权所有：晶联讯电子：网址  http://www.jlxlcd.cn;
*/
#include <STC15F2K60S2.H>
#include <intrins.h>
#include <chinese_code.h>


//================ I2C============================
sbit  lcd_reset=P1^1;
sbit    lcd_scl=P1^3;
sbit    sda=P1^2;
sbit  key=P2^0;
//=============================================================

#define uchar unsigned char
#define uint unsigned int



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

//延时 2
void delay_us(int i)
{
 int j,k;
 for(j=0;j<i;j++)
 for(k=0;k<10;k++);
```

```
}

void waitkey()
{
repeat:
     if(key==1)goto repeat;
     else delay(400);
}

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(uchar com)
{
     start_flag();
     transfer(0x7c);
     transfer(com);
     stop_flag();
}

//写数据到液晶显示模块
void transfer_data(uchar dat)
{
     start_flag();
     transfer(0x7e);
     transfer(dat);
     stop_flag();
}
```

-END-