

# Embedded System and Microcomputer Principle

LAB10 Film Transistor-Liquid Crystal Display (TFTLCD)

2022 Fall wangq9@mail.sustech.edu.cn



#### **CONTENTS**

- 1 TFTLCD Principle Description
- 2 TFTLCD Function Description
- How to Program
- Practice



01



- -- What is TFTLCD
- Film Transistor-Liquid Crystal Display,缩写TFTLCD,即薄膜晶体管液晶显示器
- TFT液晶为每个像素都设有一个半导体开关,每个像素都可以 通过点脉冲直接控制,因而每个节点都相对独立,并可以连续 控制,不仅提高了显示屏的反应速度,同时可以精确控制显示 色阶,所以TFT液晶的色彩更真
- 特点: 亮度好、对比度高、层次感强、颜色鲜艳等特点
- 目前最主流的LCD显示器
- 广泛应用于电视、手机、电脑、平板等各种电子产品

- -- ALINETEK TFTLCD
- ALINETEK 2.8寸 TFTLCD模块
- 分辨率: 240\*320
- 驱动IC: ILI9341
- 16位并口驱动
- 16位真彩显示 (65536色)
- 自带电阻触摸屏
- 自带背光电路
- 模块是3.3V供电的,不支持5V电压的MCU,如果是5V MCU,必须在信号线串接120R电阻使用





-- ATK-2.8-inch TFTLCD interface



• LCD\_WR: LCD写信号

• LCD\_RD: LCD读信号

• DB[17:1]: 16位双向数据线

• LCD\_RST: 硬复位LCD信号

• LCD\_RS: 命令/数据标志

(0:命令,1:数据)

• BL\_CTR: 背光控制信号

T\_MISO, T\_MOSI, T\_PEN,

T\_CS, T\_CLK: 触摸屏接口信号



#### 显示部分

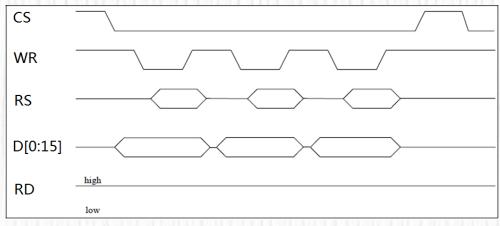
LCD CS 1 LCD WR 3 LCD RST 5 DB2 7 DB4 9 DB6 11 DB8 13 DB11 15 DB13 17 DB15 19 DB17 21 BL CTR3	LCD1 LCD_CS RS WR/CLK RD RST DB1 DB2 DB3 DB4 DB5 DB6 DB7 DB8 DB10 DB11 DB12 DB13 DB14 DB15 DB16 DB17 GND	2 LCD RS 4 LCD RD 6 DB1 8 DB3 10 DB5 12 DB7 14 DB10 16 DB12 18 DB14 20 DB16 22 GND 24 VCC3.3
T MISO 29 T PEN 31 T CS 33	VDD3.3 GND GND BL_VDD MISO MOSI T_PEN MO T CS CLK	26 GND 28 BL VDD 30 T MOSI 32 34 T CLK

TFT\_LCD

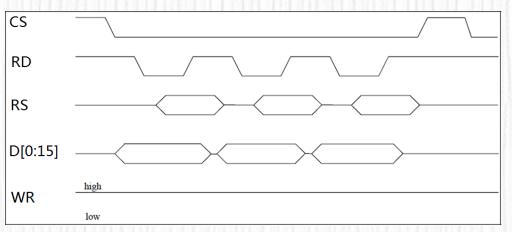
触屏控制部分

有方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- -- 8080 Parallel port principle
- 8080并口读/写的过程为:
- 先根据要数据类型,设置RS为高(数据)/低(命令),然后拉低片选,选中ILI9341,接着根据是读数据,还是要写数据置RD/WR为低,然后:
- 1.读数据:在RD的上升沿,读取数据线上的数据(D[15:0]);
- 2.写数据:在WR的上升沿,使数据写入到ILI9341里面



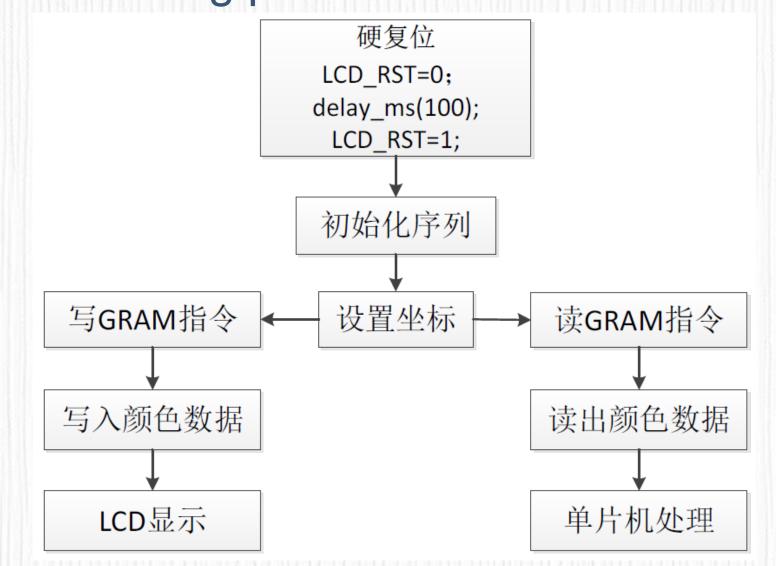
Writing timing diagram



Reading timing diagram

# 1. TFTLCD Principle Description-- TFTLCD driving process





- -- RGB565 format description
- 模块对外接口采用16位并口
- 颜色深度为16位
- 格式为RGB565

数据线	D15	D14	D13	D12	<b>D</b> 11	<b>D10</b>	D9	D8	D7	D6	<b>D</b> 5	D4	<b>D</b> 3	D2	D1	<b>D</b> 0
LCD GRAM	R[4]	R[3]	R[2]	R[1]	R[0]	G[5]	G[4]	G[3]	G[2]	G[1]	G[0]	B[4]	B[3]	B[2]	B[1]	B[0]



- -- ILI9341 instruction format description
- ILI9341所有的指令都

是8位的(高8位无效), 且参数除了读写GRAM (graphics RAM,图 像寄存器)的时候是16 位,其他操作参数,都 是8位的。

#### Regulative command set

D6 D5 D4 D3 D2 D1 D0 He

D/CX RDX WRX D17-8 D7

Command Function	D/CX	RDX	WRX	D17-8	D7	D6	D5	D4	D3	D2	D1	D0	Hex
No Operation	0	1	1	XX	0	0	0	0	0	0	0	0	00h
Software Reset	0	1	1	XX	0	0	0	0	0	0	0	1	01h
	0	1	1	XX	0	0	0	0	0	1	0	0	04h
Read Display Identification	1	1	1	XX	X	Χ	X	Χ	X	X	X	Χ	XX
Information	1	1	1	XX				ID1 [7	7:0]				XX
inomaton	1	1	1	XX				ID2 [7	7:0]				XX
	1	1	1	XX				ID3 [7	7:0]				XX
	0	1	1	XX	0	0	0	0	1	0	0	1	09h
	1	1	1	XX	X	Χ	X	X	X	X	X	X	XX
Read Display Status	1	1	1	XX			D	[31:25]				X	00
riead Display Status	1	1	1	XX	X	I	) [22:20]			D [19	):16]		61
	1	1	1	XX	X	X	X	X	X		D [10:8]		00
	1	1	1	XX	ı	D [7:5]		X	X	X	X	Χ	00
	0	1	1	XX	0	0	0	0	1	0	1	0	0Ah
Read Display Power Mode	1	1	1	XX	X	Χ	X	Χ	X	X	X	Χ	XX
	1	1	1	XX			D [7:	:2]			0	0	08
	0	1	1	XX	0	0	0	0	1	0	1	1	0Bh
Read Display MADCTL	1	1	1	XX	X	Χ	X	Χ	X	X	X	X	XX
	1	1	1	XX			D [7:	:2]			0	0	00
	0	1	1	XX	0	0	0	0	1	1	0	0	0Ch
Read Display Pixel Format	1	1	1	XX	X	Χ	X	X	X	X	X	X	XX
	1	1	1	XX	RIM	I	DPI [2:0]		X		DBI [2:0]		06
	0	1	1	XX	0	0	0	0	1	1	0	1	0Dh
Read Display Image Format	1	1	1	XX	X	Χ	X	X	X	X	X	Χ	XX
	1	1	1	XX	X	X	X	X	X		D [2:0]		00
	0	1	1	XX	0	0	0	0	1	1	1	0	0Eh
Read Display Signal Mode	1	1	1	XX	X	Χ	X	Χ	X	X	X	Χ	XX
	1	1	1	XX			D [7:	:2]			0	0	00
Read Display Self-Diagnostic	0	1	1	XX	0	0	0	0	1	1	1	1	0Fh
Result	1	1	1	XX	X	X	X	Χ	X	X	X	Χ	XX
Hoouit	1	1	1	XX	D [7	:6]	X	X	X	X	X	X	00



#### Regulative command set (continued)

Command Function	D/CX	RDX	WRX	D17-8	D7 .	D6	D5	D4	D3	D2	D1	D0	Hex	
Enter Sleep Mode	0	1	1	XX	0	0	0	1	0	0	0	0	101	
Sleep OUT	0	1	1	XX	0	0	0	1	0	0	0	1	110	
Partial Mode ON	0	1	1	XX	0	0	0	1	0	0	1	0	121	
Normal Display Mode ON	0	1	1	XX	0	0	0	1	0	0	1	1	131	
Display Inversion OFF	0	1	1	XX	0	0	1	0	0	0	0	0	201	
Display Inversion ON	0	1	1	XX	0	0	1	0	0	0	0	1	211	
Commo Cot	0	1	1	XX	0	0	1	0	0	1	1	0	261	
Gamma Set	1	1	1	XX				GC [7	7:0]				01	
Display OFF	0	1	1	XX	0	0	1	0	1	0	0	0	28	
Display ON	0	1	1	XX	0	0	1	0	1	0	0	1	291	
	0	1	1	XX	0	0	1	0	1	0	1	0	2Al	
	1	1	1	XX				SC [1	5:8]				XX	
Column Address Set	1	1	1	XX				SC [7	7:0]				XX	
	1	1	1	XX				EC [1	5:8]				XX	
	1	1	1	XX				EC [7	7:0]				XX	
	0	1	1	XX	0	0	1	0	1	0	1	1	2Bł	
	1	1	1	XX				SP [1	5:8]				XX	
Page Address Set	1	1	1	XX				SP [7	7:0]				XX	
	1	1	1	XX	EP [15:8]									
	1	1	1	XX				EP [7	7:0]				XX	

#### Regulative command set (continued)

Memory Write	0	1	1	XX	0	0	1	0	1	1	0	0	2Ch
Wemory write	1	1	1				[	0 [17:0]					XX
	0	1	1	XX	0	0	1	0	1	1	0	1	2Dł
	1	1	1	XX					R	00 [5:0]			XX
	1	1	1	XX					R	nn [5:0]			XX
	1	1	1	XX					R	31 [5:0]			XX
Color SET	1	1	1	XX					G	00 [5:0]			XX
Color SE I	1	1	1	XX					G	nn [5:0]			XX
	1	1	1	XX					G	64 [5:0]			XX
	1	1	1	XX					В	00 [5:0]			XX
	1	1	1	XX					В	nn [5:0]			XX
	1	1	1	XX					В	31 [5:0]			XX
	0	1	1	XX	0	0	1	0	1	1	1	0	2Eh
Memory Read	1	1	1	XX	X	X	X	X	X	X	X	X	XX
	1	1	1				[	0 [17:0]					XX
	0	1	1	XX	0	0	1	1	0	0	0	0	30h
	1	1	1	XX				SF	R [15:8]				00
Partial Area	1	1	1	XX				S	R [7:0]				00
	1	1	1	XX				EF	R [15:8]				01
	1	1	1	XX				Е	R [7:0]				3F
	0	1	1	XX	0	0	1	1	0	0	1	1	33h
	1	1	1	XX				TF	A [15:8]				00
	1	1	1	XX				TF	A [7:0]				00
Vertical Scrolling Definition	1	1	1	XX				VS	A [15:8]				01
	1	1	1	XX				VS	SA [7:0]				40
	1	1	1	XX					A [15:8]				00
	1	1	1	XX				BF	A [7:0]				00
Tearing Effect Line OFF	0	1	1	XX	0	0	1	1	0	1	0	0	34h
	0	1	1	XX	0	0	1	1	0	1	0	1	35h
Tearing Effect Line ON	1	1	<b>†</b>	XX	X	Х	Х	Х	X	X	Х	М	00



Regulative command set (continued)

Mamany Assess Control	0	1	1	XX	0	0	1	1	0	1	1	0	36h
Memory Access Control	1	1	1	XX	MY	MX	MV	ML	BGR	MH	X	X	00
	0	1	1	XX	0	0	1	1	0	1	1	1	37l
Vertical Scrolling Start Address	1	1	1	XX				VS	SP [15:8]				00
	1	1	1	XX				V	SP [7:0]				00
Idle Mode OFF	0	1	1	XX	0	0	1	1	1	0	0	0	381
Idle Mode ON	0	1	1	XX	0	0	1	1	1	0	0	1	391
Pixel Format Set	0	1	1	XX	0	0	1	1	1	0	1	0	3Al
Pixel Formal Set	1	1	1	XX	X		DPI [2:0	]	X		DBI [2:0	)]	66
Write Memory Continue	0	1	1	XX	0	0	1	1	1	1	0	0	3C
Write Memory Continue	1	1	1					[17:0]					XX
	0	1	1	XX	0	0	1	1	1	1	1	0	3E
Read Memory Continue	1	1	1	XX	X	Χ	X	X	X	X	X	X	XX
	1	1	1					[17:0]					XX
	0	1	1	XX	0	1	0	0	0	1	0	0	44
Set Tear Scanline	1	1	1	XX	X	Χ	X	X	X	X	X	STS [8]	00
	1	1	1	XX				S	TS [7:0]				00
	0	1	1	XX	0	1	0	0	0	1	0	1	45h
Cat Cappling	1	1	1	XX	X	X	Χ	X	X	Χ	X	X	XX
Get Scanline	1	1	1	XX	X	Χ	Χ	X	X	Χ	GTS	S [9:8]	00
	1	1	1	XX				G	TS [7:0]				00
Write Dienlau Brightnass	0	1	1	XX	0	1	0	1	0	0	0	1	511
Write Display Brightness	1	1	1	XX				DI	BV [7:0]				00

#### Regulative command set (continued)

	0	1	1	XX	0	1	0	1	0	0	1	0	52h
Read Display Brightness	1	1	1	XX	X	X	X	X	X	X	X	X	XX
	1	1	1	XX				DBV	[7:0]				00
Write CTRL Display	0	1	1	XX	0	1	0	1	0	0	1	1	53h
Willo OTTLE Display	1	1	1	XX	X	Χ	BCTRL	Χ	DD	BL	X	X	00
	0	1	1	XX	0	1	0	1	0	1	0	0	54h
Read CTRL Display	1	1	1	XX	X	X	X	X	X	X	X	X	XX
	1	1	1	XX	X	Χ	BCTRL	Χ	DD	BL	X	X	00
Write Content Adaptive	0	1	1	XX	0	1	0	1	0	1	0	1	55h
Brightness Control	1	1	1	XX	X	Χ	X	X	X	X	C[	1:0]	00
Dood Content Adentics	0	1	1	XX	0	1	0	1	0	1	1	0	56h
Read Content Adaptive Brightness Control	1	1	1	XX	X	X	X	X	X	X	X	X	XX
Brightiless Control	1	1	1	XX	X	Χ	X	X	X	X	C[	1:0]	00
Write CABC Minimum	0	1	1	XX	0	1	0	1	1	1	1	0	5EI
Brightness	1	1	1	XX				CME	[7:0]				00
Read CABC Minimum	0	1	1	XX	0	1	0	1	0	1	1	1	5FI
Brightness	1	1	1	XX	X	Χ	X	X	X	X	X	X	XX
Brighthoss	1	1	1	XX				CME	[7:0]				00
	0	1	1	XX	1	1	0	1	1	0	1	0	DA
Read ID1	1	1	1	XX	X	Χ	X	X	X	X	X	X	XX
	1	1	1	XX			Modu	le's Ma	nufacture	e [7:0]			XX
	0	1	1	XX	1	1	0	1	1	0	1	1	DB
Read ID2	1	1	1	XX	Х	Х	X	Х	X	Х	Х	Х	XX
	1	1	1	XX			LCD Mod	dule / D	river Vers	sion [7:0	]		XX
	0	1	1	XX	1	1	0	1	1	1	0	0	DC
Read ID3	1	1	1	XX	X	Х	X	Х	Χ	Χ	X	Х	XX
	1	1	1	XX			LCD N	/lodule /	Driver II	D [7:0]			XX



#### Extended command set (continued)

Command Function	D/CX	RDX	WRX	D17-8	D7	D6	DS	5 D4	D3	D2	D1	D0	Hex
RGB Interface	0	1	1	XX	1	0	1	1	0	0	0	0	B0h
Signal Control	1	1	1	XX	ByPass_MODI	RCI	M [1:0	] X	VSPL	HSPL	DPL	EPL	40
France October	0	1	1	XX	1	0	1	1	0	0	0	1	B1h
Frame Control	1	1	1	XX	X	X	X	X	Х	X	DIVA	[1:0]	00
(In Normal Mode)	1	1	1	XX	X	X	X			RTNA [4:	0]		1B
Frame Control	0	1	1	XX	1	0	1	1	0	0	1	0	B2h
(In Idle Mode)	1	1	1	XX	X	X	X	X	X	X	DIVE	8 [1:0]	00
(III Idle Mode)	1	1	1	XX	X	X	X			RTNB [4:	0]		1B
Frame Control	0	1	1	XX	1	0	1	1	0	0	1	1	B3h
(In Partial Mode)	1	1	1	XX	X	X	X	X	X	X	DIVO	[1:0]	00
(III Fartial Mode)	1	1	1	XX	X	X	X			RTNC [4:		1B	
Display Inversion Control	0	1	1	XX	1	0	1	1	0	1	0	0	B4h
Display Inversion Control	1	1	1	XX	X	X	X	X	X	NLA	NLB	NLC	02
	0	1	1	XX	1	0	1	1	0	1	0	1	B5h
	1	1	1	XX	0				VFP [6	6:0]			02
Blanking Porch Control	1	1	1	XX	0				VBP [6	6:0]			02
	1	1	1	XX	0	0	0			HFP [4:0	0]		0A
	1	1	1	XX	0	0	0			HBP [4:0	0]		14
	0	1	1	XX	1	0	1	1	0	1	1	0	B6h
	1	1	<b>↑</b>	XX	X	Χ	X	Χ	PTG	[1:0]	PT	[1:0]	0A
Display Function Control	1	1	1	XX	REV	GS	SS	SM		ISC	C [3:0]		82
	1	1	1	XX	X	Χ			N	IL [5:0]		·	27
	1	1	<b>↑</b>	XX	X	Χ			PC	DIV [5:0]			XX
Futus Mada Cat	0	1	1	XX	1	0	1	1	0	1	1	1	B7h
Entry Mode Set	1	1	1	XX	X	Χ	X	Χ	0	GON	DTE	GAS	07

#### Extended command set (continued)

	0	1	1	XX	1	0	1	1	1	0	0	0	B8h
Backlight Control 1	1	1	1	XX	X	X	X	X	X	X	X	X	XX
	1	1	1	XX	X	X	X	X		TH	_UI [3:0]		04
	0	1	1	XX	1	0	1	1	1	0	0	1	B9h
Backlight Control 2	1	1	1	XX	X	X	X	X	X	X	X	X	XX
	1	1	1	XX		TH_MV	[3:0]			TH	_ST [3:0]		B8
	0	1	1	XX	1	0	1	1	1	0	1	0	BAh
Backlight Control 3	1	1	1	XX	X	X	X	X	X	X	X	X	XX
	1	1	1	XX	X	X	X	X		DTH	1_UI [3:0]		04
	0	1	1	XX	1	0	1	1	1	0	1	1	BBh
Backlight Control 4	1	1	1	XX	X	X	X	X	X	X	X	X	XX
	1	1	1	XX		DTH_M\	V [3:0]			DTH	1_ST [3:0]		C9
	0	1	1	XX	1	0	1	1	1	1	0	0	BCh
Backlight Control 5	1	1	1	XX	X	X	X	X	X	X	X	X	XX
	1	1	1	XX		DIM2 [	[3:0]		X		DIM1 [2:	0]	44
Backlight Control 7	0	1	1	XX	1	0	1	1	1	1	1	0	BEh
Backlight Control /	1	1	1	XX				PWM	_DIV [7	<b>':0]</b>			0F
Backlight Control 8	0	1	1	XX	1	0	1	1	1	1	1	1	BFh
Dacklight Control o	1	1	1	XX	X	X	X	X	X	LEDONR	LEDONPOL	LEDPWMOPL	00
Power Control 1	0	1	1	XX	1	1	0	0	0	0	0	0	C0h
Fower Control i	1	1	1	XX	X	X			\	/RH [5:0]			26
Power Control 2	0	1	1	XX	1	1	0	0	0	0	0	1	C1h
Fower Control 2	1	1	1	XX	X	X	X	X	X		BT [2:0	0]	00
	0	1	1	XX	1	1	0	0	0	1	0	1	C5h
VCOM Control 1	1	1	1	XX	X		VMH [6:0]						31
	1	1	1	XX	X				VML	[6:0]			3C
VCOM Control 2	0	1	1	XX	1	1	0	0	0	1	1	1	C7h
VCOIVI CONTROL 2	1	1	1	XX	nVM				VMF	[6:0]			C0



#### Extended command set (continued)

			100								40.00			100
	0	1	1	XX		1	1	0	1	0	0	0	0	D0h
NV Memory Write	1	1	1	XX		X	X	X	X	X	F	PGM_AI	OR [2:0]	00
	1	1	1	XX					PGM_	DATA [	7:0]			XX
	0	1	1	XX		1	1	0	1	0	0	0	1	D1h
NV Memory Protection Key	1	1	1	XX					KE'	Y [23:16	]			55
14V Memory 1 Totection Key	1	1	1	XX					KE	Y [15:8]				AA
	1	1	1	XX					KI	EY [7:0]				66
	0	1	1	XX		1	1	0	1	0	0	1	0	D2l
NV Memory Status Read	1	1	1	XX		X	X	X	X	X	X	X	X	XX
TV Wellery States Fload	1	1	1	XX		X		D2_CNT		X		ID1_CN		XX
	1	1	1	XX	B	USY	VI	MF_CNT	[2:0]	X		ID3_CN	IT [2:0]	XX
	0	1	1	XX	1		1	0	1	0	0	1	1	D3h
	1	1	1	XX	X	)	X	X	X	X	X	X	X	XX
Read ID4	1	1	1	XX	0	(	0	0	0	0	0	0	0	00
	1	1	1	XX	1	(	0	0	1	0	0	1	1	93
	1	1	1	XX	0		1	0	0	0	0	0	1	41
	0	1	1	XX	1		1	1	0	0	0	0	0	E0h
	1	1	1	XX	X	)	X	X	X			0 [3:0]		08
	1	1	1	XX	X	)	X			VP1	[5:0]			0E
	1	1	1	XX	X	)	X			VP2	[5:0]			12
	1	1	1	XX	X	)	X	X	X		VP	4 [3:0]		05
	1	1	1	XX	X		X	X			VP6 [4			03
	1	1	1	XX	X	)	X	X	X		VP1	13 [3:0]		09
Positive Gamma	1	1	1	XX	X				VI	P20 [6:0]	]			47
Correction	1	1	1	XX		V	P36 [	3:0]				27 [3:0]		86
	1	1	1	XX	X					P43 [6:0]				2B
	1	1	1	XX	X		X	X	X			50 [3:0]		0B
	1	1	1	XX	X		X	X			VP57 [4			04
	1	1	1	XX	X		X	X	X			59 [3:0]		00
	1	1	1	XX	X		X				[5:0]			00
	1	1	1	XX	X		X			VP62				00
	1	1	1	XX	X	)	X	X	X		VP	63 [3:0]		00

#### Extended command set (continued)

	0	1	<b>↑</b>	XX	1	1	1	0	0	0	0	1	E1h				
	1	1	1	XX	X	X	X	X		VN	0 [3:0]		08				
	1	1	1	XX	X	X			VN1 [5	:0]			1A				
	1	1	1	XX	X	X			VN2 [5	:0]			20				
	1	1	1	XX	X	X	X	X		VN	4 [3:0]		07				
	1	1	<b>↑</b>	XX	X	X	X		VI	N6 [4:	:0]		0E				
	1	1	<b>†</b>	XX	X	X	X	X		VN1	3 [3:0]		05				
Negative Gamma	1	1	<b>↑</b>	XX	X			17	N20 [6:0]				ЗА				
Correction	1	1	<b></b>	XX		VN36 [	[3:0]			VN2	7 [3:0]		8A				
	1	1	<b>↑</b>	XX	X			1V	N43 [6:0]				40				
	1	1	<b>←</b>	XX	X	X	X	X		VN5	0 [3:0]		04				
	1	1	<b></b>	XX	X	X	X		VN	157 [4	:0]		18				
	1	1	1	XX	X	X	X	X		VN5		0F					
	1	1	<b>←</b>	XX	X	X			VN61 [5	VN61 [5:0]							
	1	1	<b>↑</b>	XX	X	X			VN62 [5	5:0]			3F				
	1	1	1	XX	X	X	X	X		VN6	3 [3:0]		0F				
Digital Gamma Control 1	0	1	1	XX	1	1	1	0	0	0	1	0	E2h				
1 <sup>st</sup> Parameter	1	1	1	XX		RCA0	[3:0]			BCA	0 [3:0]		XX				
:	1	1	1	XX		RCAx	[3:0]			BCA	x [3:0]		XX				
16 <sup>th</sup> Parameter	1	1	1	XX		RCA15	[3:0]			BCA	15 [3:0]		XX				
Digital Gamma Control 2	0	1	1	XX	1	1	1	0	0	0	1	1	E3h				
1 <sup>st</sup> Parameter	1	1	<b>↑</b>	XX		RFA0	[3:0]			BFA	0 [3:0]		XX				
:	1	1	<b>↑</b>	XX		RFAx	[3:0]			BFA	x [3:0]		XX				
64 <sup>th</sup> Parameter	1	1	1	XX		RFA63	[3:0]			BFA	63 [3:0]		XX				
	0	1	<b></b>	XX	1	1	1	1	0	1	1	0	F6h				
Interface Control	1	1	1	XX	MY_EOR	MX_EOR	MV_EOR	X	BGR_EOR	X	X	WEMODE	01				
interface Control	1	1	1	XX	X	X	EPF [	1:0]	X	X	MD	T [1:0]	00				
	1	1	1	XX	X	X	ENDIAN	X	DM [1:	0]	RM	RIM	00				



- -- 0xD3 instruction
- 该指令为<mark>读ID4指令</mark>,用于读取LCD控制器的ID。因此,同一个代码,可以根据ID的不同,执行不同的LCD驱动初始化,以兼容不同的LCD屏幕。

顺序		控制				名	位推	述					HEX
/i灰/丁*	RS	RD	WR	D15~D8	D7	D6	D5	D4	D3	D2	D1	D0	ПЕЛ
指令	0	1	<b>↑</b>	XX	1	1	0	1	0	0	1	1	D3H
参数 1	1	<b>↑</b>	1	XX	X	X	X	X	X	X	X	X	X
参数 2	1	<b>↑</b>	1	XX	0	0	0	0	0	0	0	0	00Н
参数 3	1	<b>↑</b>	1	XX	1	0	0	1	0	0	1	1	93H
参数 4	1	<b>↑</b>	1	XX	0	1	0	0	0	0	0	1	41H



- -- 0x36 instruction
- 该指令为<mark>存储访问控制指令</mark>,可以控制ILI9341存储器的读写方向,简单的说,就是在连续写GRAM的时候,<mark>可以控制GRAM</mark>指针的增长方向,从而控制显示方式(读GRAM也是一样)。

	顺序		控制					各位	描述					HEX
		RS	RD	WR	D15~D8	D7	D6	D5	D4	D3	D2	D1	D0	ПЕХ
	指令	0	1	<b>↑</b>	XX	0	0	1	1	0	1	1	0	36H
	参数	1	1	<b>↑</b>	XX	MY	MX	MV	ML	BGR	MH	0	0	0

Bit	Name	Description						
MY	Row Address Order							
MX	Column Address Order	These 3 bits control MCU to memory write/read direction.						
MV	Row / Column Exchange							
ML	Vertical Refresh Order	LCD vertical refresh direction control.						
BGR	RGB-BGR Order	Color selector switch control						
ban	Nab-Ban Older	(0=RGB color filter panel, 1=BGR color filter panel)						
MH	Horizontal Refresh ORDER	LCD horizontal refreshing direction control.						

找	空制位	Ĭ	效果							
MY	MX	MV	LCD 扫描方向 (GRAM 自增方式)							
0	0	0	从左到右,从上到下							
1	0	0	从左到右,从下到上							
0	1	0	从右到左,从上到下							
1	1	0	从右到左,从下到上							
0	0	1	从上到下,从左到右							
0	1	1	从上到下,从右到左							
1	0	1	从下到上, 从左到右							
1	1	1	从下到上,从右到左							



- -- 0x2A instruction
- 该指令是<mark>列地址设置指令</mark>,在从左到右,从上到下的扫描方式(默认)下面,该指令用于设置横坐标(x坐标)。
- 在默认扫描方式时,该指令用于设置x坐标,该指令带有4个参数,实际上是2个坐标值: SC和EC,即列地址的起始值和结束值,SC必须小于等于EC,且0≤SC/EC≤239(与分辨率相关)。一般在设置x坐标的时候,我们只需要带2个参数即可,也就是设置SC即可,因为如果EC没有变化,我们只需要设置一次即可(在初始化ILI9341的时候设置),从而提高速

度。

顺序	控制			各位描述									
州火力	RS	RD	WR	D15~D8	D7	D6	D5	D4	D3	D2	D1	D0	HEX
指令	0	1	<b>↑</b>	XX	0	0	1	0	1	0	1	0	2AH
参数 1	1	1	<b>↑</b>	XX	SC15	SC14	SC13	SC12	SC11	SC10	SC9	SC8	CC
参数 2	1	1	<b>↑</b>	XX	SC7	SC6	SC5	SC4	SC3	SC2	SC1	SC0	SC
参数 3	1	1	<b>↑</b>	XX	EC15	EC14	EC13	EC12	EC11	EC10	EC9	EC8	EC
参数 4	1	1	<b>↑</b>	XX	EC7	EC6	EC5	EC4	EC3	EC2	EC1	EC0	EC



- -- 0x2B instruction
- 该指令是页地址设置指令,在从左到右,从上到下的扫描方式(默认)下面,该指令用于设置纵坐标(y坐标)。
- 在默认扫描方式时,该指令用于设置y坐标,该指令带有4个参数,实际上是2个坐标值: SP和EP,即页地址的起始值和结束值,SP必须小于等于EP,且0≤SP/EP≤319 (与分辨率相关)。一般在设置y坐标的时候,我们只需要带2个参数即可,也就是设置SP即可,因为如果EP没有变化,我们只需要设置一次即可(在初始化ILI9341的时候设置),从而提高速

度。

顺序		控制					各位	描述					HEX				
顺灯	RS	RD	WR	D15~D8	D7	D6	D5	D4	D3	D2	D1	D0	ПЕЛ				
指令	0	1	<b>↑</b>	XX	0	0	1	0	1	0	1	0	2BH				
参数 1	1	1	<b>↑</b>	XX	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	CD				
参数 2	1	1	<b>↑</b>	XX	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SP				
参数 3	1	1	<b>↑</b>	XX	EP15	EP14	EP13	EP12	EP11	EP10	EP9	EP8	ED				
参数 4	1	1	1	XX	EP7	EP6	EP5	EP4	EP3	EP2	EP1	EP0	EP				



- -- 0x2C instruction
- 该指令是写GRAM指令,在发送该指令之后,我们便可以往LCD的GRAM里面写入颜色数据了,该指令支持连续写(地址自动递增)。
- 在收到指令0X2C之后,数据有效位宽变为16位,我们可以连续写入LCD GRAM值,而GRAM的地址将根据MY/MX/MV设置的扫描方向进行自增。例如:假设设置的是从左到右,从上到下的扫描方式,那么设置好起始坐标(通过SC,SP设置)后,每写入一个颜色值,GRAM地址将会自动自增1(SC++),如果碰到EC,则回到SC,同时SP++,一直到坐标:EC,EP结束,其间无需再次设置的坐标,从而大大提高写入速度。

顺序	控制			各位描述									HEX		
炒厂	RS	RD	WR	D15~D8	D7	D6	D5	D4	D3	D2	D1	D0	HEA		
指令	0	1	<b>↑</b>	XX	0	0	1	0	1	1	0	0	2CH		
参数 1	1	1	1			D	1[15:	0]					XX		
•••••	1	1	1		D2[15: 0]								XX		
参数 n	1	1	<b>↑</b>	Dn[15: 0]							XX				



- -- 0x2E instruction
- 该指令是读GRAM指令,用于读取ILI9341的显存(GRAM),同0X2C 指令一样,该指令支持连续读(地址自动递增)。
- ILI9341收到该指令后,第一次输出的是dummy数据(无效),第二次开始,读取到的才是有效的GRAM数据(从坐标:SC,SP开始),输出规律为:每个颜色分量占8bit,一次输出2个颜色分量。比如:第一次输出是R1G1,随后的规律为:

B1R2→G2B2→R3G3→B3R4→G4B4→R5G5... 依此类推。

顺序		控制			各位描述										HEV	
/ / / / / / / / / / / / / / / / / / /	RS	RD	WR	D15~D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	HEX
指令	0	1	<b>↑</b>		XX			0	0	1	0	1	1	1	0	2EH
参数 1	1	<b>↑</b>	1		XX							dummy				
参数 2	1	<b>↑</b>	1	R1[4:0]	XX				G1[5:0]					XX		R1G1
参数 3	1	<b>↑</b>	1	B1[4:0]		XX		R2[4:0]						XX		B1R2
参数 4	1	<b>↑</b>	1	G2[5:0]	XX		X	B2[4:0]				XX			G2B2	
参数 5	1	<b>↑</b>	1	R3[4:0]	XX		G3[5:0]						XX		R3G3	
参数 N	1	1	1	按以上规律输出												



02

TFTLCD Function Description

#### 2. TFTLCD Function Description



```
-- __lcd_dev structure
typedef struct {
   u16 width; //LCD 宽度
   u16 height; //LCD 高度
   u16 id;
                //LCD ID
                //横屏还是竖屏控制: 0, 竖屏; 1, 横屏。
   u8 dir;
   u16 wramcmd; //开始写GRAM指令
   u16 setxcmd; //设置x坐标指令
   u16 setycmd; //设置y坐标指令
}_lcd_dev;
//LCD参数 in lcd.h
extern lcd_dev lcddev;
```

//管理LCD重要参数



- -- Some functions
- LCD初始化函数: void LCD\_Init(void)
- 写寄存器值函数(通过 8080 并口向 LCD 模块写入寄存器命令):
   void LCD\_WR\_REG(u16 regval)
- 写数据函数(写入16位的数据): void LCD\_WR\_DATA(u16 data)
- 读数据函数: u16 LCD\_RD\_DATA(void)
- 写寄存器内容函数: void LCD\_WriteReg(u16 LCD\_Reg, u16 LCD\_RegValue)
- 读寄存器内容函数: u16 LCD\_ReadReg(u16 LCD\_Reg)
- 开始写GRAM函数: void LCD\_WriteRAM\_Prepare(void)
- 写GRAM函数: void LCD\_WriteRAM(u16 RGB\_Code)
- 坐标设置函数: void LCD\_SetCursor(u16 Xpos, u16 Ypos)
- 画点函数: void LCD\_DrawPoint(u16 x,u16 y)
- LCD读点函数: u16 LCD\_ReadPoint(u16 x, u16 y)



- -- Some functions(continued)
- LCD字符显示函数: void LCD\_ShowChar(u16 x, u16 y, u8 num, u8 size, u8 mode)
- void LCD\_Clear(uint16\_t Color); // Clear the screen with specific color
- void LCD\_DrawLine(uint16\_t x1, uint16\_t y1, uint16\_t x2, uint16\_t y2);
   // Draw a line
- void LCD\_DrawRectangle(uint16\_t x1, uint16\_t y1, uint16\_t x2, uint16\_t y2); // Draw a rectangle
- void LCD\_Fill(uint16\_t sx,uint16\_t sy,uint16\_t ex,uint16\_t ey,uint16\_t color); // Fill the area with color
- void LCD\_ShowNum(uint16\_t x,uint16\_t y,uint32\_t num,uint8\_t len,uint8\_t size); // Display number without the leading zeros
- void LCD\_ShowString(uint16\_t x, uint16\_t y, uint16\_t width, uint16\_t height, uint8\_t size, uint8\_t \*p); // Display a string



- -- Character code table
- //PC2LCD2002取模方式设置: 阴码+逐列式+顺向+C51格式
- //总共: 3个字符集(12\*12、16\*16和24\*24)in font.h
- //每个字符所占用的字节数为:(size/8+((size%8)?1:0))\*(size/2),其中size是字库生成时的点阵大小(12/16/24...)

```
//12*12 ASCII字符集点阵
```

#### //16\*16 ASCII字符集点阵

.....

 $\{0x00,0x00,0x60,0x00,0x80,0x00,0x80,0x00,0x40,0x00,0x40,0x00,0x20,0x00,0x20,0x00\}, /*"\sim",94*/\};$ 

//24\*24 ASICII字符集点阵 .....

# 1. TFTLCD Principle Description-- PCtoLCD2002





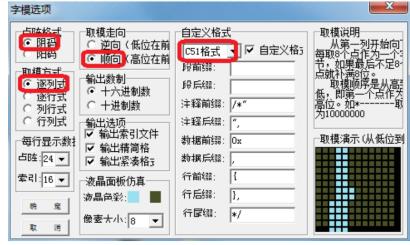
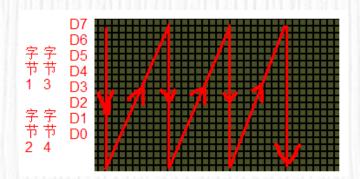


图 17.3.2 设置取模方式

上图设置的取模方式,在右上角的取模说明里面有,即:从第一列开始向下每取8个点作为一个字节,如果最后不足8个点就补满8位。取模顺序是从高到低,即第一个点作为最高位。如\*------取为10000000。其实就是按如图17.3.3 所示的这种方式:





03

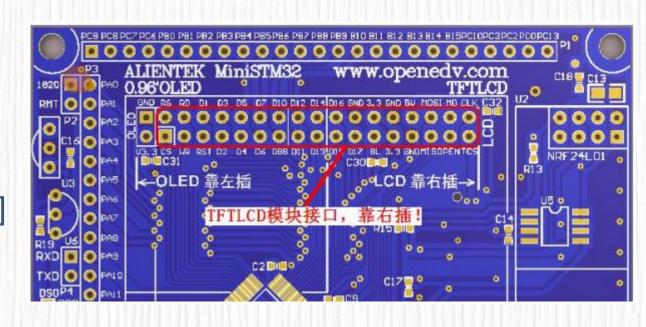
How to Program

- Our Goal
  - Show charactes on TFTLCD





- Hardware connection
  - MiniSTM32 开发板底板的 LCD 接口和 ALIENTEK TFTLCD 模块直接可以对插(靠右插!)
  - 多出的 2 个口是给 OLED 用的
  - TFTLCD 模块与 MiniSTM32 开发板的 IO口对应关系如下:
    - LCD\_LED 对应 PC10
    - LCD\_CS 对应 PC9
    - LCD \_RS 对应 PC8
    - LCD \_WR 对应 PC7
    - LCD \_RD 对应 PC6
    - LCD \_D[17:1]对应 PB[15:0]



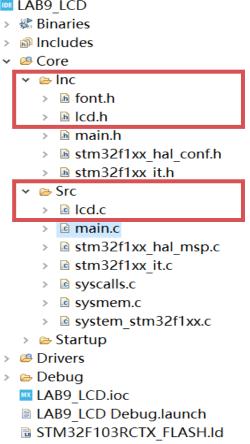


Download Icd.c, Icd.h and font.h from Sakai site

RES	OUR	CES				
Site Resou	urces	Trash	Transfer Files	Check Quota		
All sit	e files	<u>s</u> • /	CS301-2021	Fall Resource	s / <u>lab</u>	/ references
Move	Сор	у Мо	ove to Trash	Show Hide		
×* [	<u>Ti</u>	tle 🔨				
		refere	nces			
		<b>&gt;</b> <u>lab9</u>	LCD			
	)	□ <u>fo</u> ı	nt.h			
	)	<u>  lcc</u>	<u>l.c</u>			
	)	□ <u>lcc</u>	<u>l.h</u>			



 Add the files in your STM32Cube project. Source files should be in Src folder, while header files should be in Inc folder.





Add the following codes in main.c

```
/* USER CODE BEGIN Includes */
#include "lcd.h"
/* USER CODE END Includes */
int main(void)
         /* USER CODE BEGIN SysInit */
         LCD Init();
         /* USER CODE END SysInit */
         // .....
        /* Infinite loop */
         /* USER CODE BEGIN WHILE */
         uint8_t x = 0;
         while (1) {
         /* USER CODE END WHILE */
```

```
/* USER CODE BEGIN 3 */
switch (x) {
case 0: LCD_Clear(WHITE); BACK_COLOR = WHITE; break;
case 1: LCD_Clear(BLACK); BACK_COLOR = BLACK; break;
case 2: LCD Clear(BLUE); BACK COLOR = BLUE; break;
case 3: LCD Clear(RED); BACK COLOR = RED; break;
case 4: LCD_Clear(MAGENTA); BACK_COLOR = MAGENTA; break;
case 5: LCD_Clear(GREEN); BACK_COLOR = GREEN; break;
case 6: LCD_Clear(CYAN); BACK_COLOR = CYAN; break;
case 7: LCD Clear(YELLOW); BACK COLOR = YELLOW; break;
case 8: LCD_Clear(BRRED); BACK_COLOR = BRRED; break;
case 9: LCD_Clear(GRAY); BACK_COLOR = GRAY; break;
case 10: LCD_Clear(LGRAY); BACK_COLOR = LGRAY; break;
case 11: LCD Clear(BROWN); BACK COLOR = BROWN; break;
}//end of switch
```



Add the following codes in main.c

```
POINT COLOR = RED;
         LCD_ShowString(30, 40, 200, 24, 24, (uint8_t*) "Mini STM32 ^_^");
         LCD_ShowString(30, 70, 200, 16, 16, (uint8_t*) "TFTLCD TEST");
         POINT_COLOR = BLACK;
         LCD_DrawRectangle(30, 150, 210, 190);
         LCD_Fill(31, 151, 209, 189, YELLOW);
         X++;
         if (x == 12)
                 x = 0:
         HAL_Delay(2000);
         } //end of while(1)
} //end of main
```



04

Practice

#### 4. Practice

Run the demo on MiniSTM32 board

