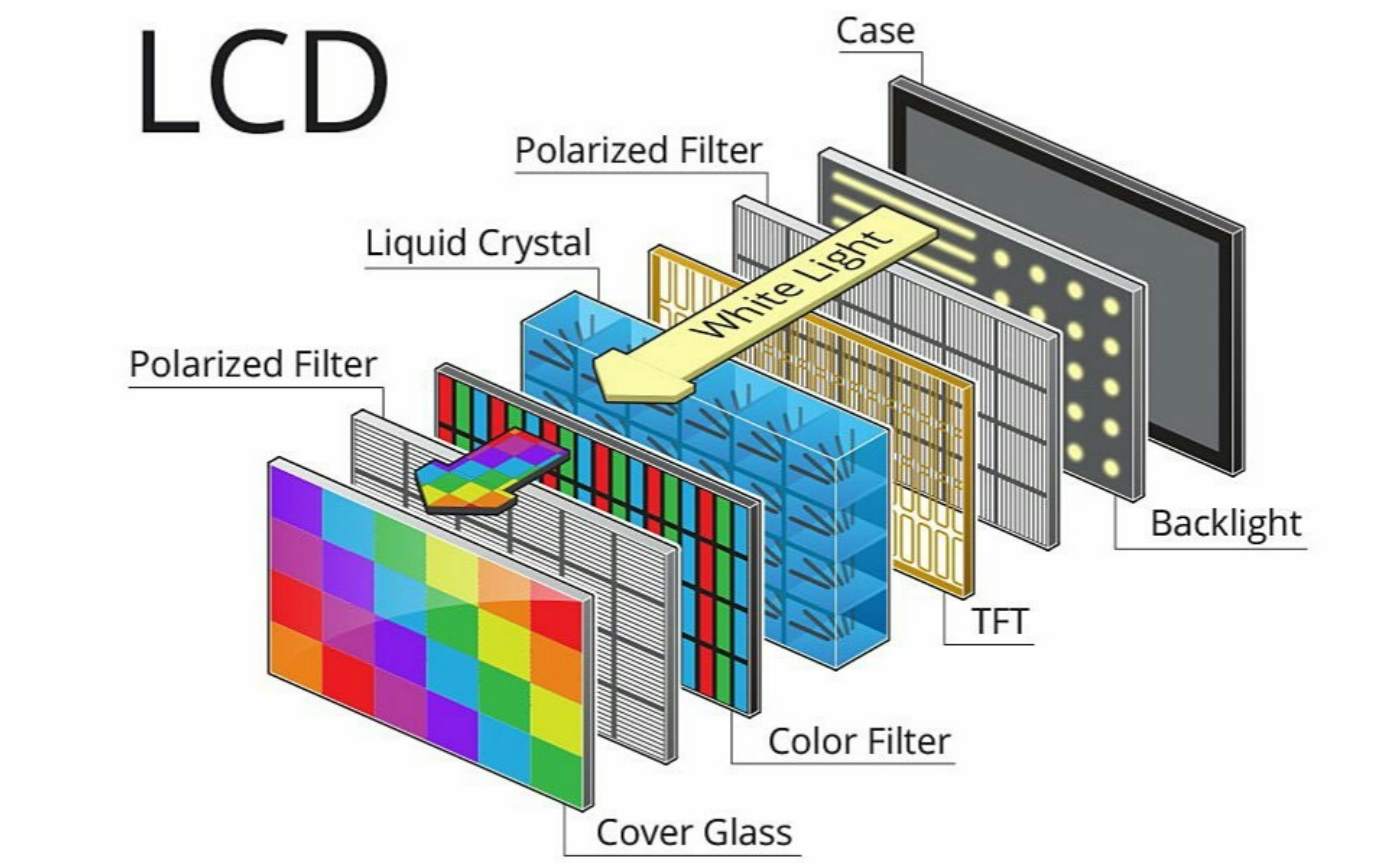
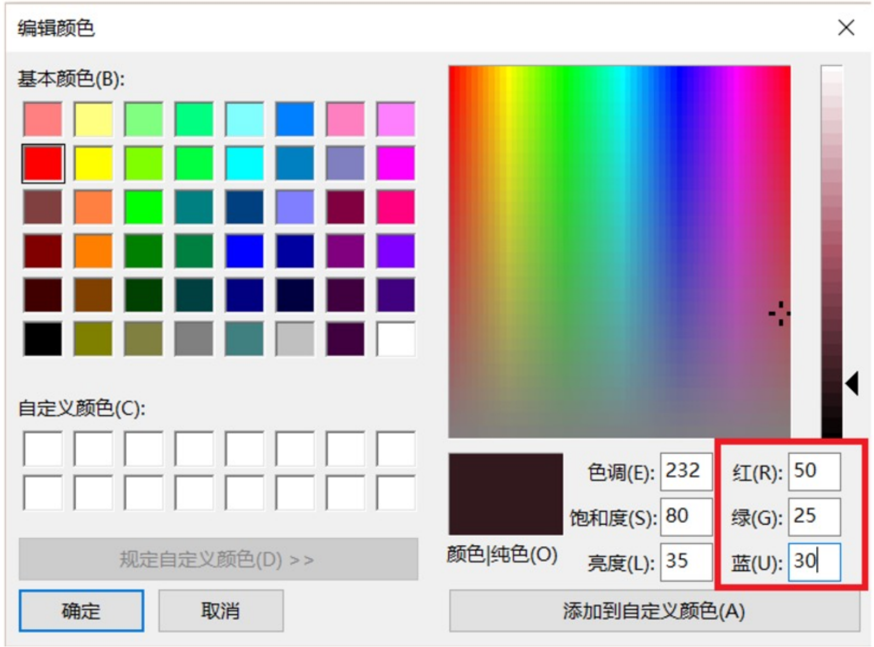
Final Project – LCD & AC motor

Liquid Crystal Display



Basic parameters

- Pixel
- Resolution
- Size
- Color depth



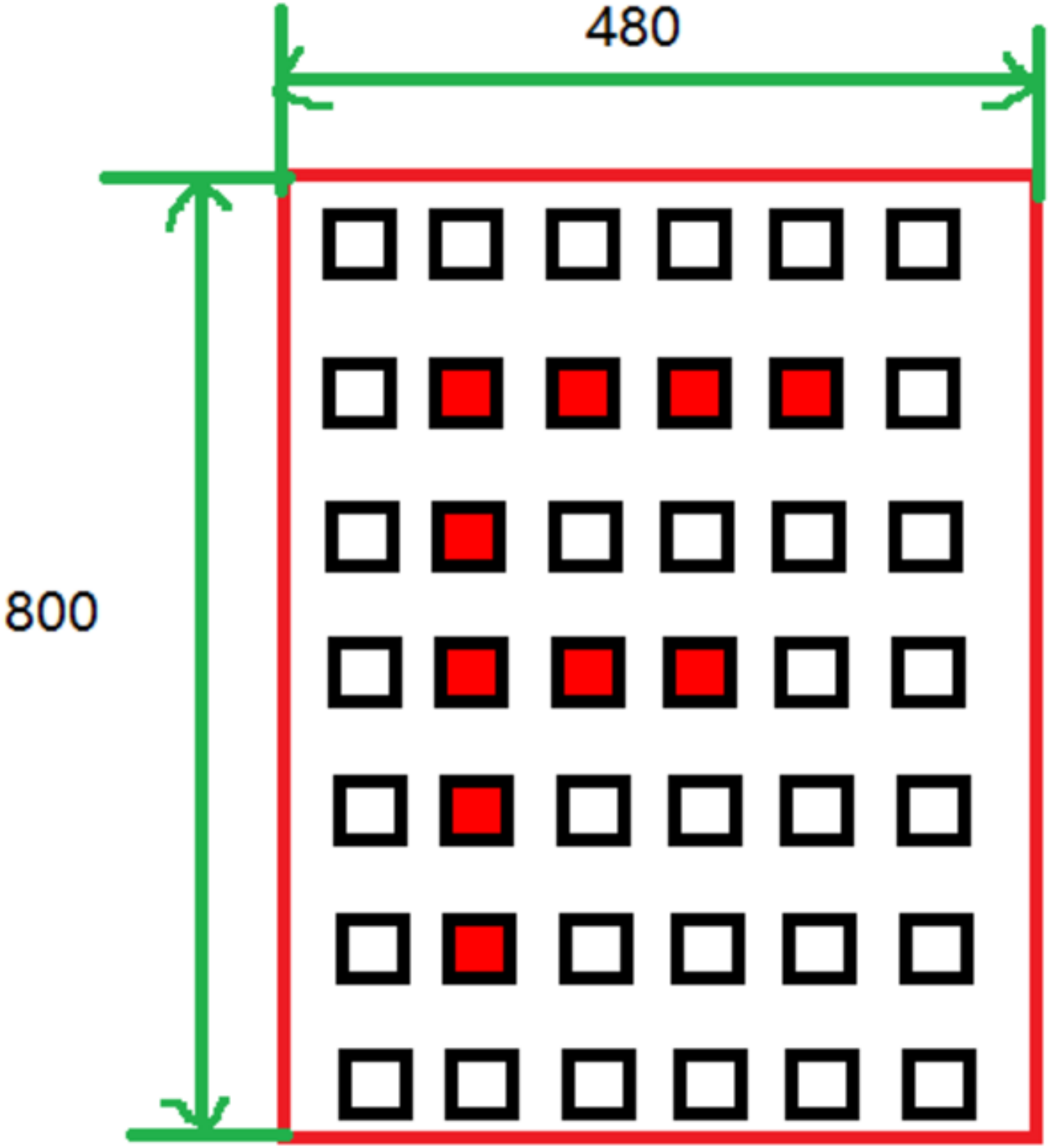
编辑颜色

基本颜色(B):

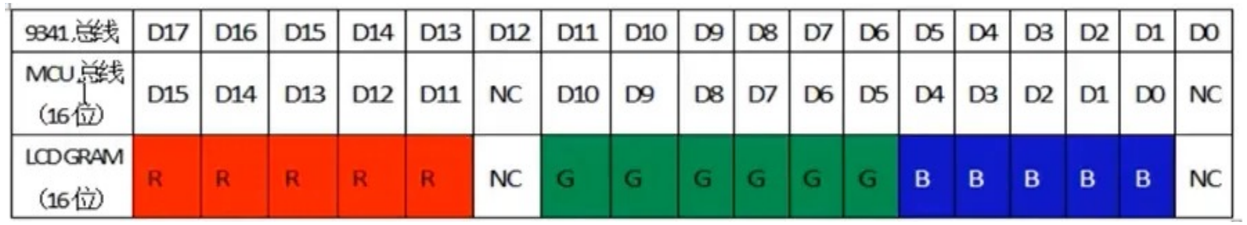
自定义颜色(C):

色调(E): 232
饱和度(S): 80
亮度(L): 35

红(R): 50
绿(G): 25
蓝(U): 30



Memory:

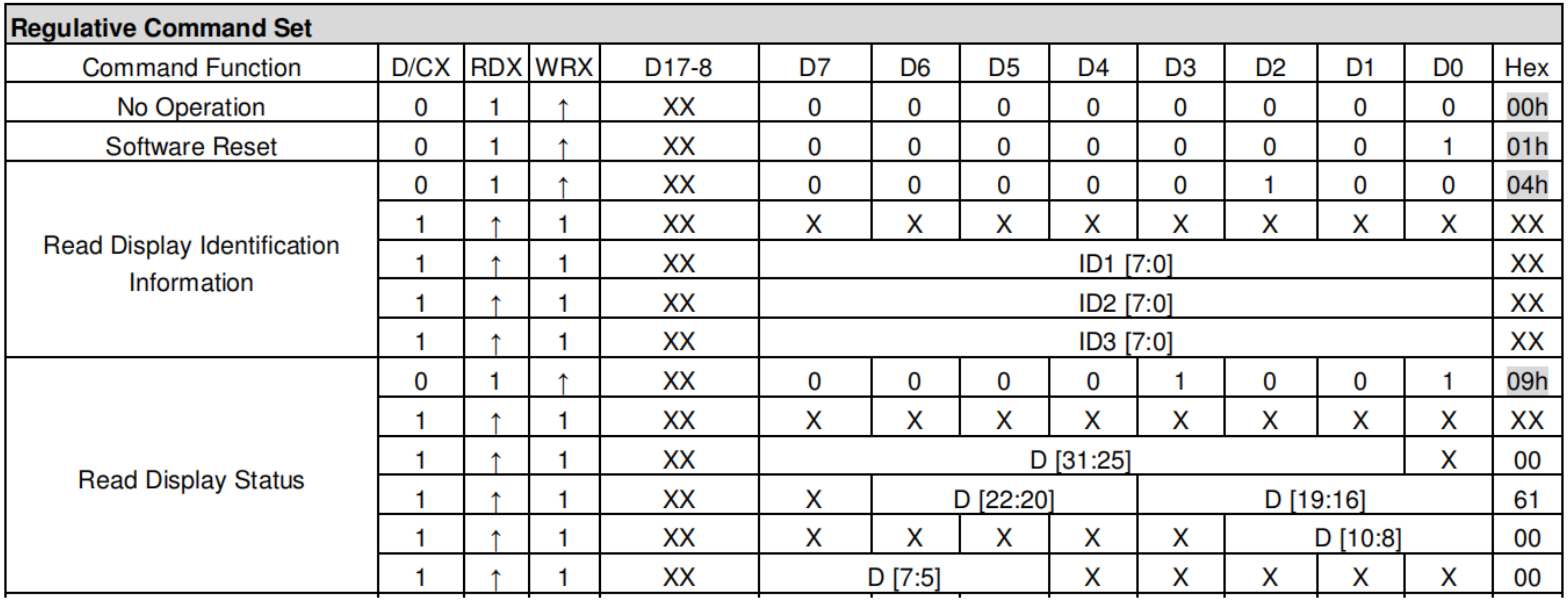


9341总线	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
MCU总线 （16位）	D15	D14	D13	D12	D11	NC	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	NC
LCD GRAM （16位）	R	R	R	R	R	NC	G	G	G	G	G	G	B	B	B	B	B	NC

- Each pixel in the LCD screen is data, in practical applications need to cache the data of each pixel, and then transmitted to the LCD screen, generally use SRAM or SDRAM nature of the memory, and these are specifically used to store the display data memory, is known as the video memory.
- Memory should generally be able to store at least one frame of the LCD screen (one page, that is, 480 * 800 pixels) display data, such as resolution of 800x480 LCD screen, using the RGB565 format display, the size of a frame of the display data is: 2x800x480 = 768000 bytes.
- Generally speaking, external LCD controllers come with their own memory, while chips with integrated LCD controllers such as the STM32F429 can use internal SRAM or external SDRAM for the memory space.

Command of SSD1963

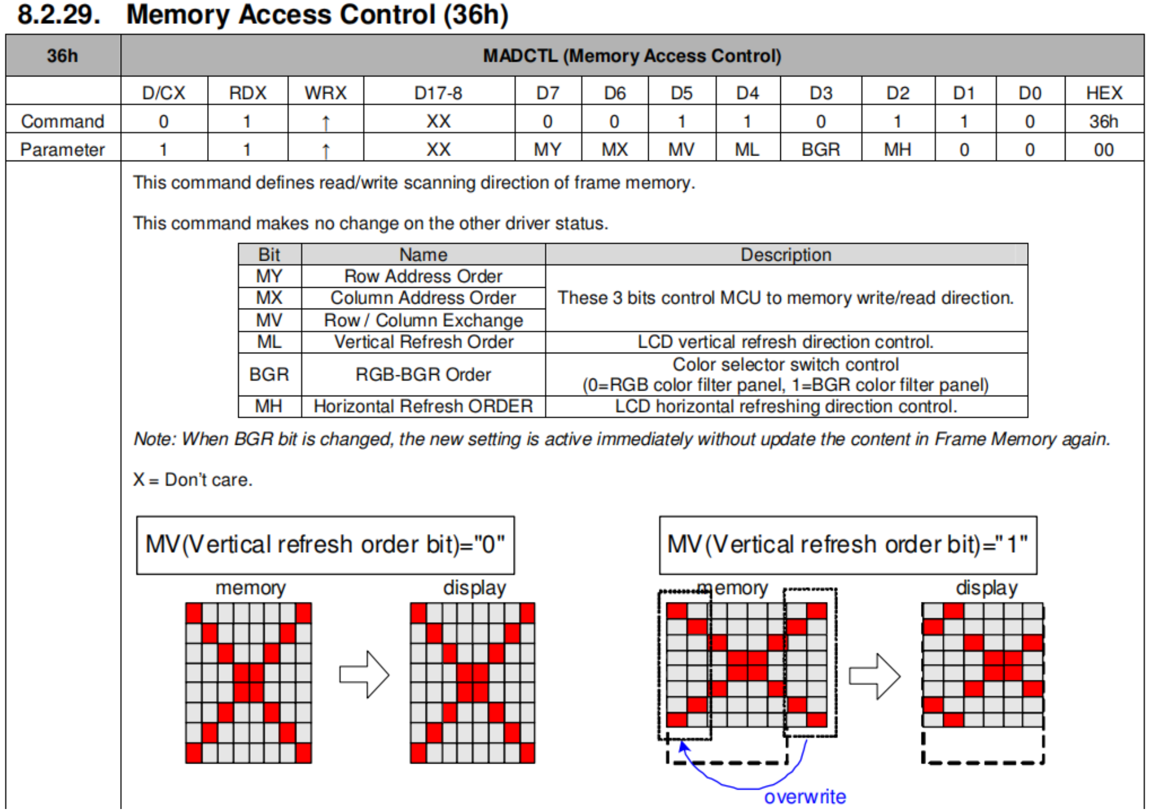
- Only accept 8-bit command
- 0xD3, 0x36, 0x2A, 0x2B, 0x2C, 0x2E



Regulative Command Set													
Command Function	D/CX	RDX	WRX	D17-8	D7	D6	D5	D4	D3	D2	D1	D0	Hex
No Operation	0	1	↑	XX	0	0	0	0	0	0	0	0	00h
Software Reset	0	1	↑	XX	0	0	0	0	0	0	0	1	01h
Read Display Identification Information	0	1	↑	XX	0	0	0	0	0	1	0	0	04h
	1	↑	1	XX	X	X	X	X	X	X	X	X	XX
	1	↑	1	XX	ID1 [7:0]								XX
	1	↑	1	XX	ID2 [7:0]								XX
	1	↑	1	XX	ID3 [7:0]								XX
Read Display Status	0	1	↑	XX	0	0	0	0	1	0	0	1	09h
	1	↑	1	XX	X	X	X	X	X	X	X	X	XX
	1	↑	1	XX	D [31:25]							X	00
	1	↑	1	XX	X	D [22:20]			D [19:16]				61
	1	↑	1	XX	X	X	X	X	X	D [10:8]			00
	1	↑	1	XX	D [7:5]			X	X	X	X	X	00

Command of SSD1963

- 0xD3: read LCD ID
- 0x36: scanning direction
- 0x2A, 0x2B: set the area of accessible frame memory in x and y direction
- 0x2C: write GRAM
- 0x2E: read GRAM
- 0x29, 0x28: display on/off



8.2.29. Memory Access Control (36h)

36h	MADCTL (Memory Access Control)												
	D/CX	RDX	WRX	D17-8	D7	D6	D5	D4	D3	D2	D1	D0	HEX
Command	0	1	↑	XX	0	0	1	1	0	1	1	0	36h
Parameter	1	1	↑	XX	MY	MX	MV	ML	BGR	MH	0	0	00

This command defines read/write scanning direction of frame memory.

This command makes no change on the other driver status.

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

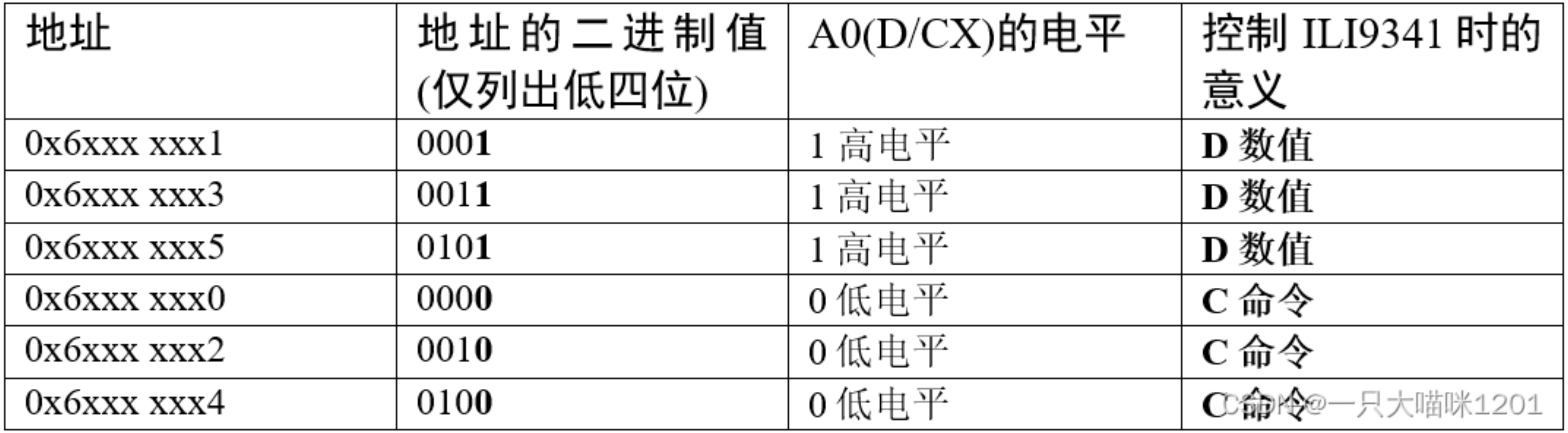
Note: When BGR bit is changed, the new setting is active immediately without update the content in Frame Memory again.

X = Don't care.

MV(Vertical refresh order bit)="0"

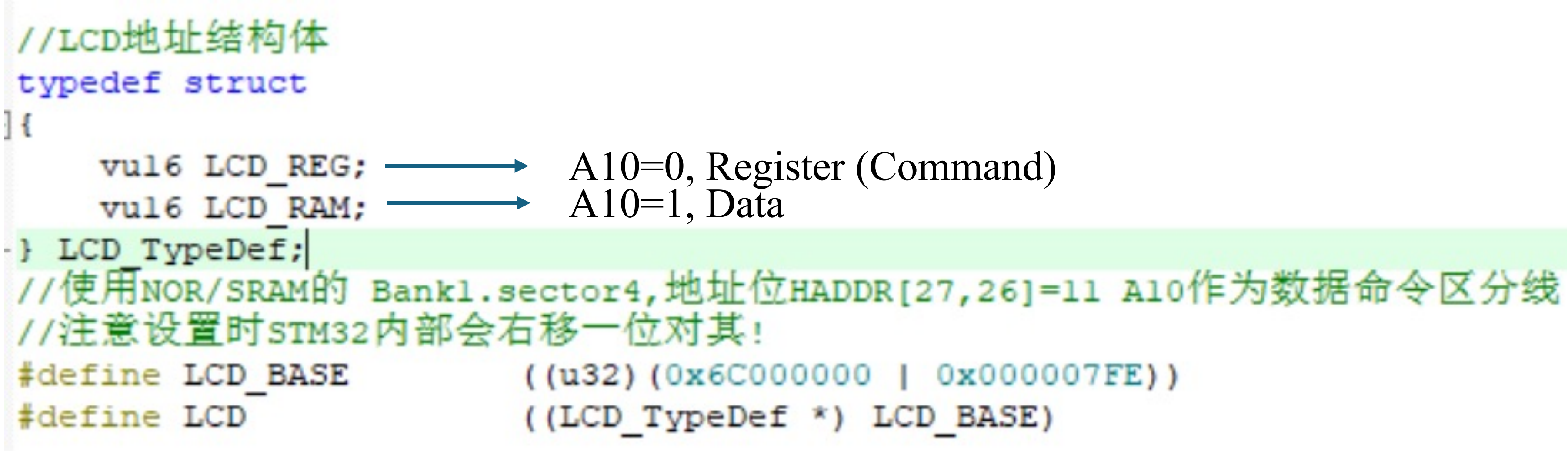
MV(Vertical refresh order bit)="1"

Flexible static memory controller (FSMC)



地址	地址的二进制值(仅列出低四位)	A0(D/CX)的电平	控制 ILI9341 时的意义
0x6xxx xxx1	000**1**	1 高电平	**D** 数值
0x6xxx xxx3	001**1**	1 高电平	**D** 数值
0x6xxx xxx5	010**1**	1 高电平	**D** 数值
0x6xxx xxx0	000**0**	0 低电平	**C** 命令
0x6xxx xxx2	001**0**	0 低电平	**C** 命令
0x6xxx xxx4	010**0**	0 低电平	**C** 命令

LCD Address

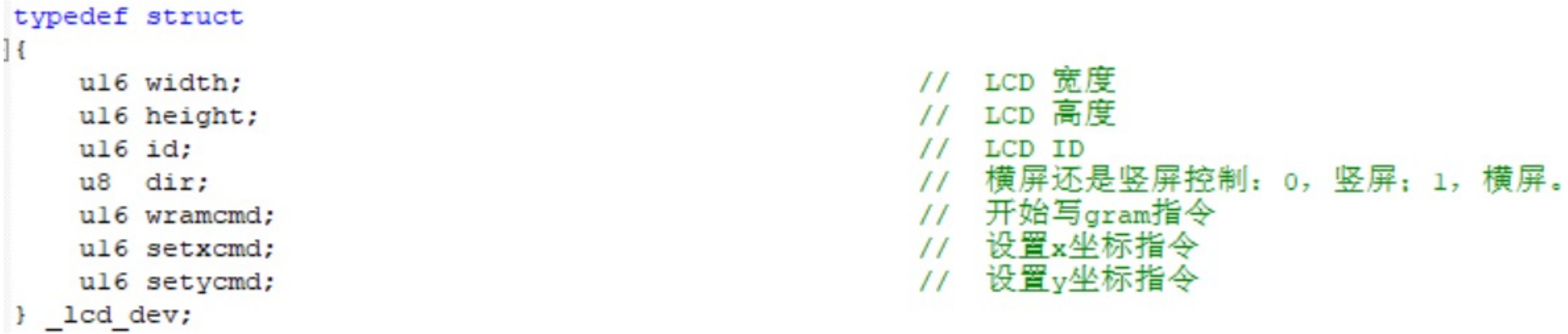


```
//LCD地址结构体
typedef struct
{
	vu16 LCD_REG;
	vu16 LCD_RAM;
} LCD_TypeDef;
//使用NOR/SRAM的 Bank1.sector4,地址位HADDR[27,26]=11 A10作为数据命令区分线
//注意设置时STM32内部会右移一位对其!
#define LCD_BASE        ((u32)(0x6C000000 | 0x000007FE))
#define LCD             ((LCD_TypeDef *) LCD_BASE)
```

A10=0, Register (Command)

A10=1, Data

LCD Parameters



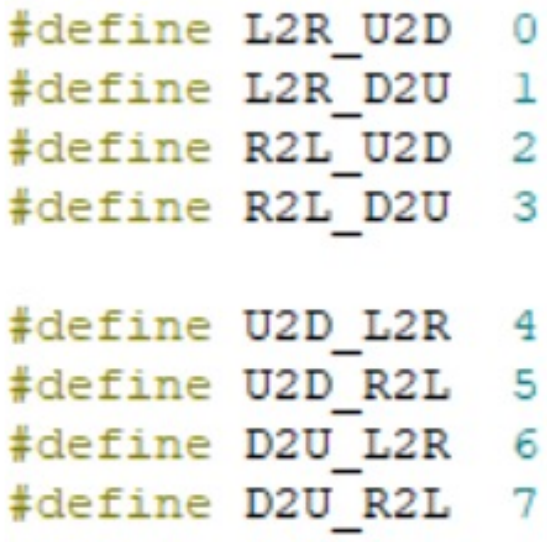
```
typedef struct
{
	u16 width;                          //  LCD 宽度
	u16 height;                         //  LCD 高度
	u16 id;                             //  LCD ID
	u8  dir;                            //  横屏还是竖屏控制：0，竖屏；1，横屏。
	u16 wramcmd;                        //  开始写gram指令
	u16 setxcmd;                        //  设置x坐标指令
	u16 setycmd;                        //  设置y坐标指令
} _lcd_dev;
```

Basic functions

- void LCD_WR_REG(u16 regval);   # write register, i.e. command
- void LCD_WR_DATA(u16 data);   # write data
- u16 LCD_RD_DATA(void);   # read data
- void LCD_WriteReg(u16 LCD_Reg, u16 LCD_RegValue);   # write register and the value
- u16 LCD_ReadReg(u16 LCD_Reg);   # read the value of register
- void LCD_WriteRAM_Prepare(void);   # configure to prepare to write GRAM (graphic RAM)
- void LCD_WriteRAM(u16 RGB_Code);   # write GRAM

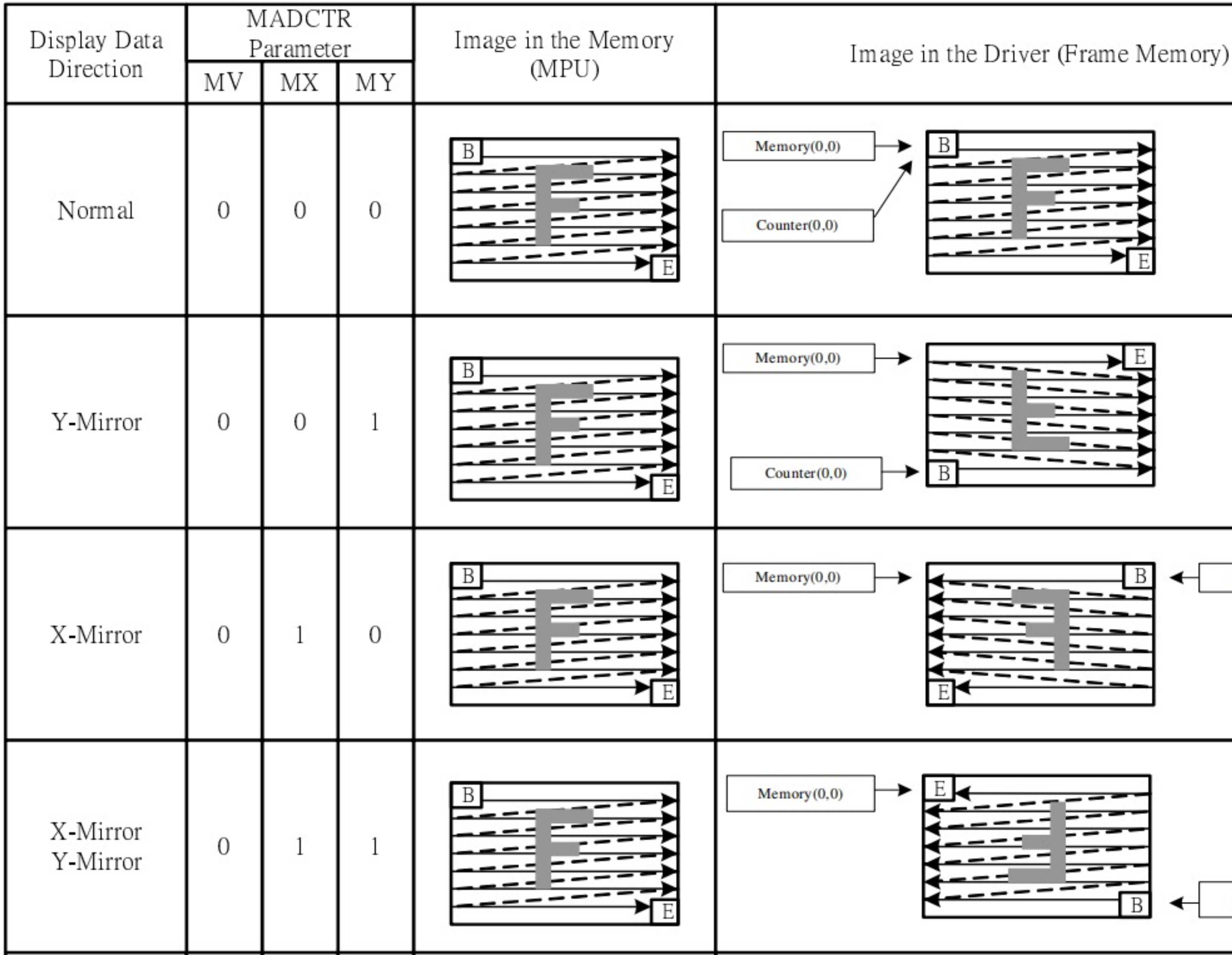
Initialization

1. Enable Clock
2. Initialize GPIO
3. Initialize FSMC
4. Read LCD ID
5. Initialize LCD based on its ID
6. Set parameters   //display_dir(0)
7. LCD_LED=1   //light the background
8. Clear LCD



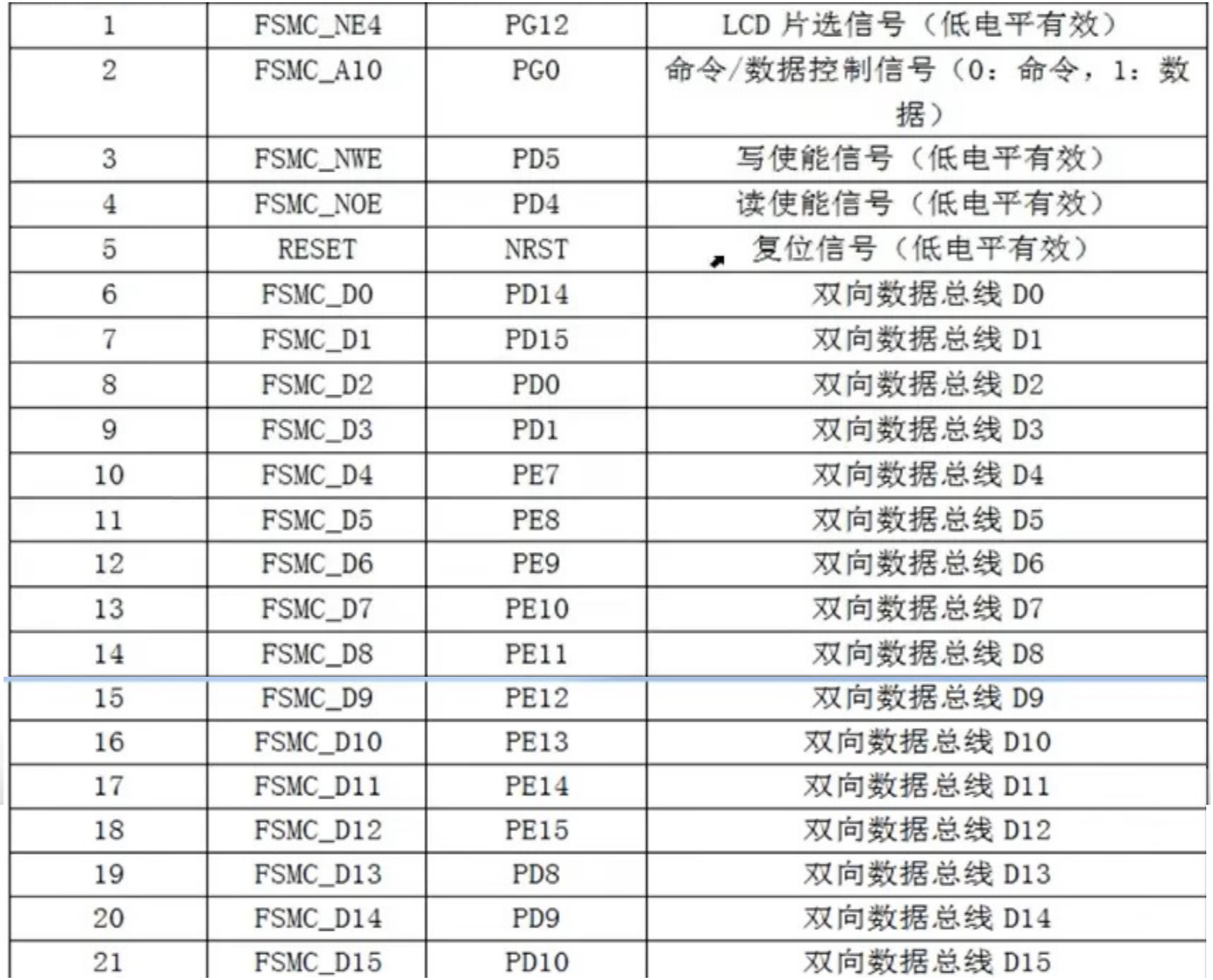
```
#define L2R_U2D  0
#define L2R_D2U  1
#define R2L_U2D  2
#define R2L_D2U  3

#define U2D_L2R  4
#define U2D_R2L  5
#define D2U_L2R  6
#define D2U_R2L  7
```



Display Data Direction	MADCTR Parameter			Image in the Memory (MPU)	Image in the Driver (Frame Memory)
	MV	MX	MY		
Normal	0	0	0		
Y-Mirror	0	0	1		
X-Mirror	0	1	0		
X-Mirror Y-Mirror	0	1	1		

B
E
Memory(0,0)
Counter(0,0)
B
E
Memory(0,0)
E
Counter(0,0)
B
B
E
Memory(0,0)
B
E
B
E
Memory(0,0)
E
B



1	FSMC_NE4	PG12	LCD 片选信号（低电平有效）
2	FSMC_A10	PG0	命令/数据控制信号（0：命令，1：数据）
3	FSMC_NWE	PD5	写使能信号（低电平有效）
4	FSMC_NOE	PD4	读使能信号（低电平有效）
5	RESET	NRST	复位信号（低电平有效）
6	FSMC_D0	PD14	双向数据总线 D0
7	FSMC_D1	PD15	双向数据总线 D1
8	FSMC_D2	PD0	双向数据总线 D2
9	FSMC_D3	PD1	双向数据总线 D3
10	FSMC_D4	PE7	双向数据总线 D4
11	FSMC_D5	PE8	双向数据总线 D5
12	FSMC_D6	PE9	双向数据总线 D6
13	FSMC_D7	PE10	双向数据总线 D7
14	FSMC_D8	PE11	双向数据总线 D8
15	FSMC_D9	PE12	双向数据总线 D9
16	FSMC_D10	PE13	双向数据总线 D10
17	FSMC_D11	PE14	双向数据总线 D11
18	FSMC_D12	PE15	双向数据总线 D12
19	FSMC_D13	PD8	双向数据总线 D13
20	FSMC_D14	PD9	双向数据总线 D14
21	FSMC_D15	PD10	双向数据总线 D15

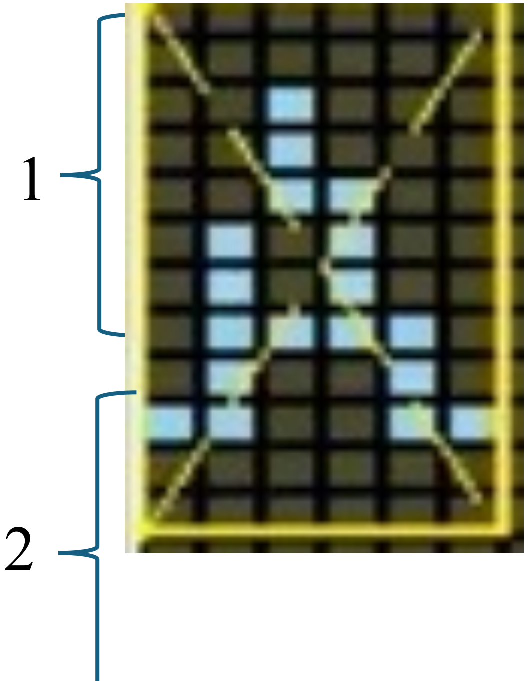
How to draw a point?

1. Setcursor
2. Write GRAM command
3. Write color data

   LCD->LCD_RAM

How to draw a character?

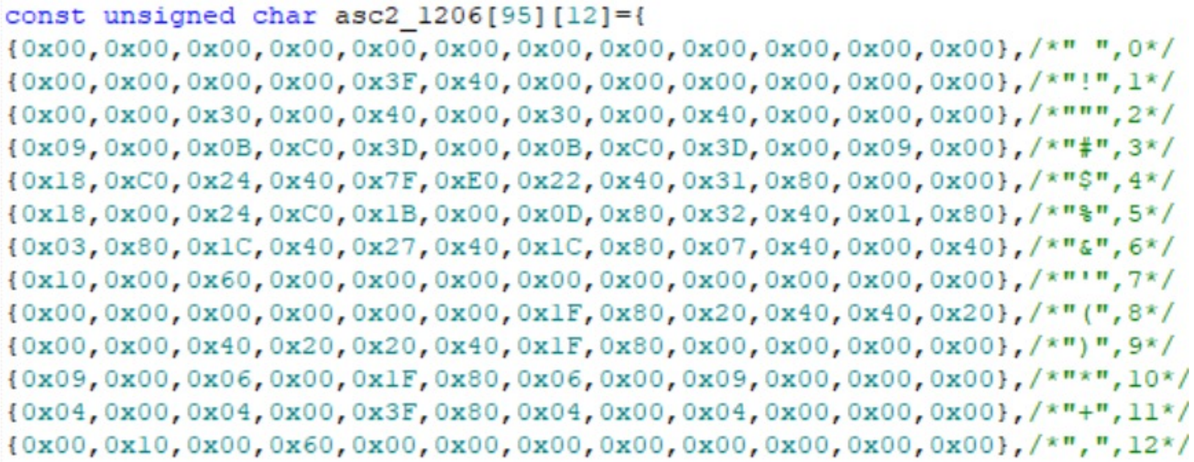
1. By the size of the characters to be displayed, the number of bytes occupied by a single character model is calculated using the formula.
2. Call the corresponding size (font size) of the font, here made three sizes of font, 12*12, 16*16, and 24 * 24
3. Through the loop to determine the font in a single byte in each bit of the font, if ==1, then fill the pixel



asc2_1206

asc2_1608

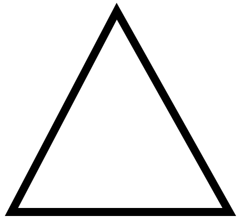
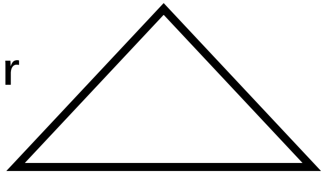
asc2_2412



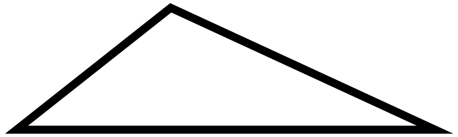
```
const unsigned char asc2_1206[95][12]={
{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00},/*" ",0*/
{0x00,0x00,0x00,0x00,0x3F,0x40,0x00,0x00,0x00,0x00,0x00,0x00},/*"!",1*/
{0x00,0x00,0x30,0x00,0x40,0x00,0x30,0x00,0x40,0x00,0x00,0x00},/*""",2*/
{0x09,0x00,0x0B,0xC0,0x3D,0x00,0x0B,0xC0,0x3D,0x00,0x09,0x00},/*"#",3*/
{0x18,0xC0,0x24,0x40,0x7F,0xE0,0x22,0x40,0x31,0x80,0x00,0x00},/*"$",4*/
{0x18,0x00,0x24,0xC0,0x1B,0x00,0x0D,0x80,0x32,0x40,0x01,0x80},/*"%",5*/
{0x03,0x80,0x1C,0x40,0x27,0x40,0x1C,0x80,0x07,0x40,0x00,0x40},/*"&",6*/
{0x10,0x00,0x60,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00},/*"'",7*/
{0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0x80,0x20,0x40,0x40,0x20},/*"(",8*/
{0x00,0x00,0x40,0x20,0x20,0x40,0x1F,0x80,0x00,0x00,0x00,0x00},/*")",9*/
{0x09,0x00,0x06,0x00,0x1F,0x80,0x06,0x00,0x09,0x00,0x00,0x00},/*"*",10*/
{0x04,0x00,0x04,0x00,0x3F,0x80,0x04,0x00,0x04,0x00,0x00,0x00},/*"+",11*/
{0x00,0x10,0x00,0x60,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00},/*",",12*/
```

Experiment 1: Use LCD to draw

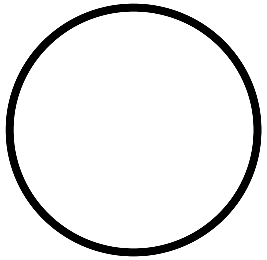
1.

2.  or 

isosceles triangle right triangle

3. 

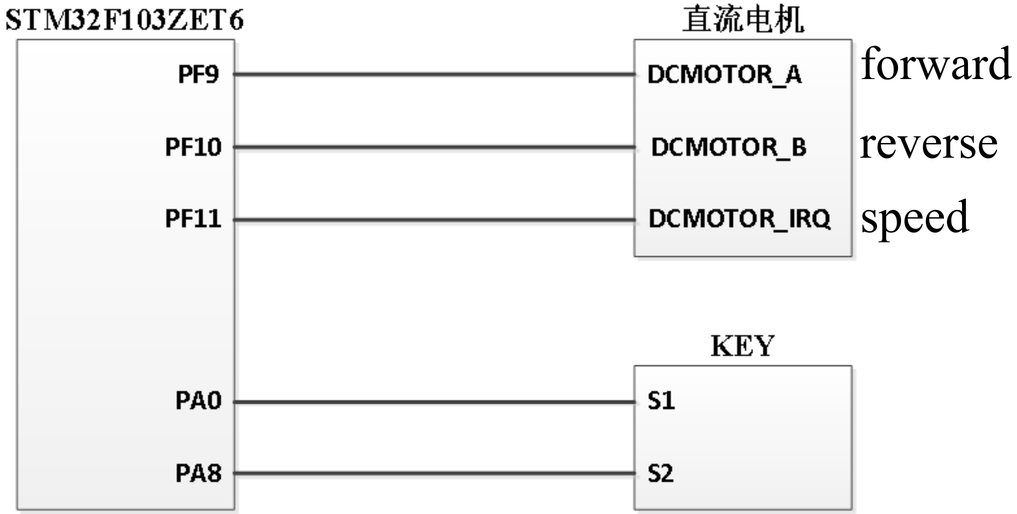
arbitrary triangle

4. 

And their corresponding filled patterns

Experiment 2: Use LCD to display

1. The names of the group members
2. Timer, count from 00:00 to 59:59
3. Use AD to control the rotation speed of the DC motor, and display it on the LCD screen


STM32F103ZET6
直流电机
PF9
DCMOTOR_A
forward
PF10
DCMOTOR_B
reverse
PF11
DCMOTOR_IRQ
speed
KEY
PA0
S1
PA8
S2