



OZDISAN ELECTRONIC A.S.

3.5” 8 bit TFT Digital Driver Board Specification

TDDB-SSD-3.5-54P-8B-V2

Doc.Version : 1.0



Ozdisan Electronic R&D and Technical Support Department

Email: ts@ozdisan.com Tel: +90 2164201882

www.ozdisan.com



Product Pictures



TDDB-SSD-3.5-54P-8B-V2

22 Pins 8Bit User MPU interface, 40 pins TFT connection, 20mA Backlight Driver

Main Parameters

Board supply voltage	3.3V
Backlight supply voltage	2.7V ~ 5.5V
Working current	< 1mA
Working temperature scope	-20°C ~ +60°C
Storage temperature scope	-40°C ~ +70°C

Controller Information

Built-in SSD1963

SSD1963 is a display controller of 1215K byte frame buffer to support up to 864x480x24bit graphics content. It also equips parallel MCU interfaces in a different bus width to receive graphics data and commands from MCU. Its display interface supports common RAM-less LCD driver of color depth up to 24 bit-per pixel.

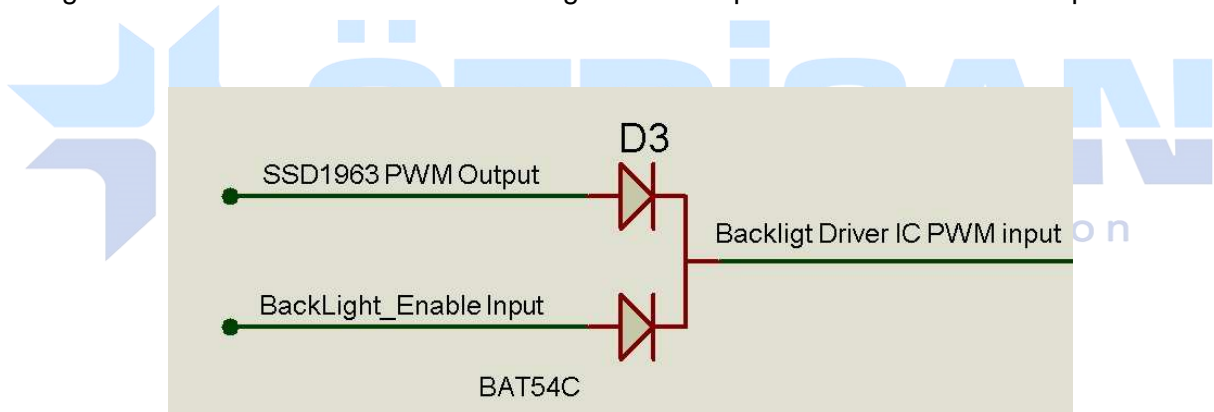


Data Format

Interface	Cycle	D[23]	D[22]	D[21]	D[20]	D[19]	D[18]	D[17]	D[16]	D[15]	D[14]	D[13]	D[12]	D[11]	D[10]	D[9]	D[8]	D[7]	D[6]	D[5]	D[4]	D[3]	D[2]	D[1]	D[0]
24 bits	1 st	R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4	G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0
18 bits	1 st							R5	R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B5	B4	B3	B2	B1	B0
16 bits (565 format)	1 st									R5	R4	R3	R2	R1	G5	G4	G3	G2	G1	G0	B5	B4	B3	B2	B1
16 bits	1 st									R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4	G3	G2	G1	G0
	2 nd									B7	B6	B5	B4	B3	B2	B1	B0	R7	R6	R5	R4	R3	R2	R1	R0
	3 rd									G7	G6	G5	G4	G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0
12 bits	1 st													R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4
	2 nd											G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0		
9 bits	1 st																R5	R4	R3	R2	R1	R0	G5	G4	G3
	2 nd															G2	G1	G0	B5	B4	B3	B2	B1	B0	
8 bits	1 st																	R7	R6	R5	R4	R3	R2	R1	R0
	2 nd																	G7	G6	G5	G4	G3	G2	G1	G0
	3 rd																	B7	B6	B5	B4	B3	B2	B1	B0

Backlight

The driver board has 20mA constant current Backlight Driver circuit with maximum 24V output. The backlight can be controlled from MCU Backlight Enable input or SSD1963 PWM output.



Backlight Enable Input	SSD1963 PWM Output	Backlight
0	Off	Off
0	On	Brightness Dimming
1	Off	%100
1	On	%100



Pin description for CN2: 22Pin 8 bit input from User's MCU

Pin No.	Symbol	Function Description
1	GND	GND
2	Vbl	Backlight Supply 2.7V-5.5V
3	Vcc	Board and TFT Supply 3.3V
4	BLen	Backlight Enable
5	WR	Write Signal-Active Low
6	RD	Read Signal-Active Low
7	RS	Register select (Data Or command)
8	CS	Chip Select Signal- Active Low (Enables data or command sending)
9	TE	Tear Effect
10	RESET	Reset Signal- Active Low
11-18	D0-D7	8 Bit Bidirectional Data bus
19	TP_R	Touch Panel Right
20	TP_B	Touch Panel Bottom
21	TP_L	Touch Panel Left
22	TP_T	Touch Panel Top

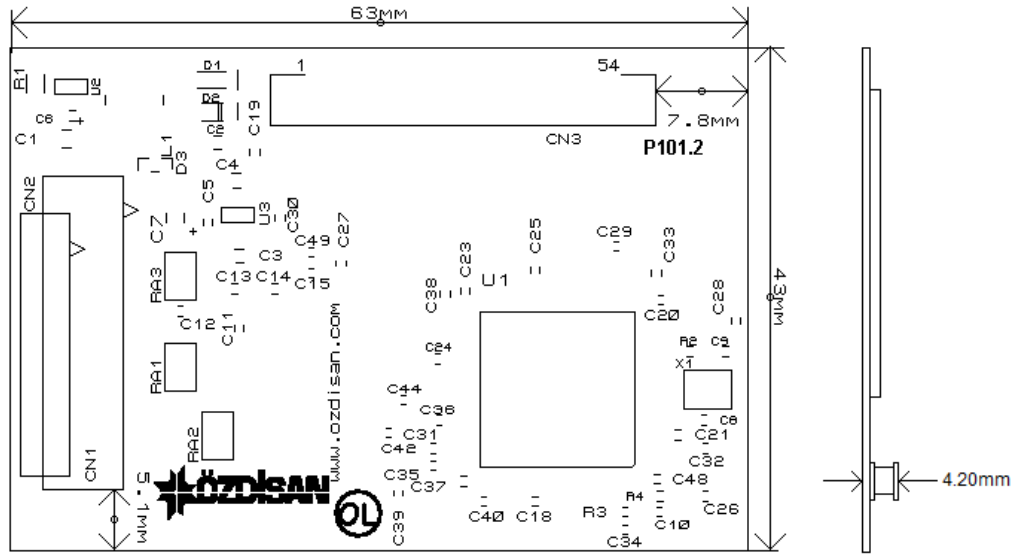
Pin Description for CN3: Pin Connections to TFT Panel

Pin No.	Symbol	Function Description
1	LED-	LED Cathode
2	LED-	LED Cathode
3	LED+	LED Anode
4	LED+	LED Anode
5-7	NC	
8	RESET	Reset
9	SPENA	Serial Port Data Enable Signal
10	SPCK	SPI Serial Clock
11	SPDA	SPI Serial Data Input
12-19	B0-B7	Blue Data 0-7
20-27	G0-G7	Green Data 0-7
28-35	R0-R7	Red Data 0-7
36	HSYNC	Horizontal Sync
37	VSYNC	Vertical Sync
38	DCLK	Data Clock
39-40	NC	
41-42	VDD	Power Supply (3.3V)
43-51	NC	
52	DEN	Data Enabling Signal
53-54	GND	Ground

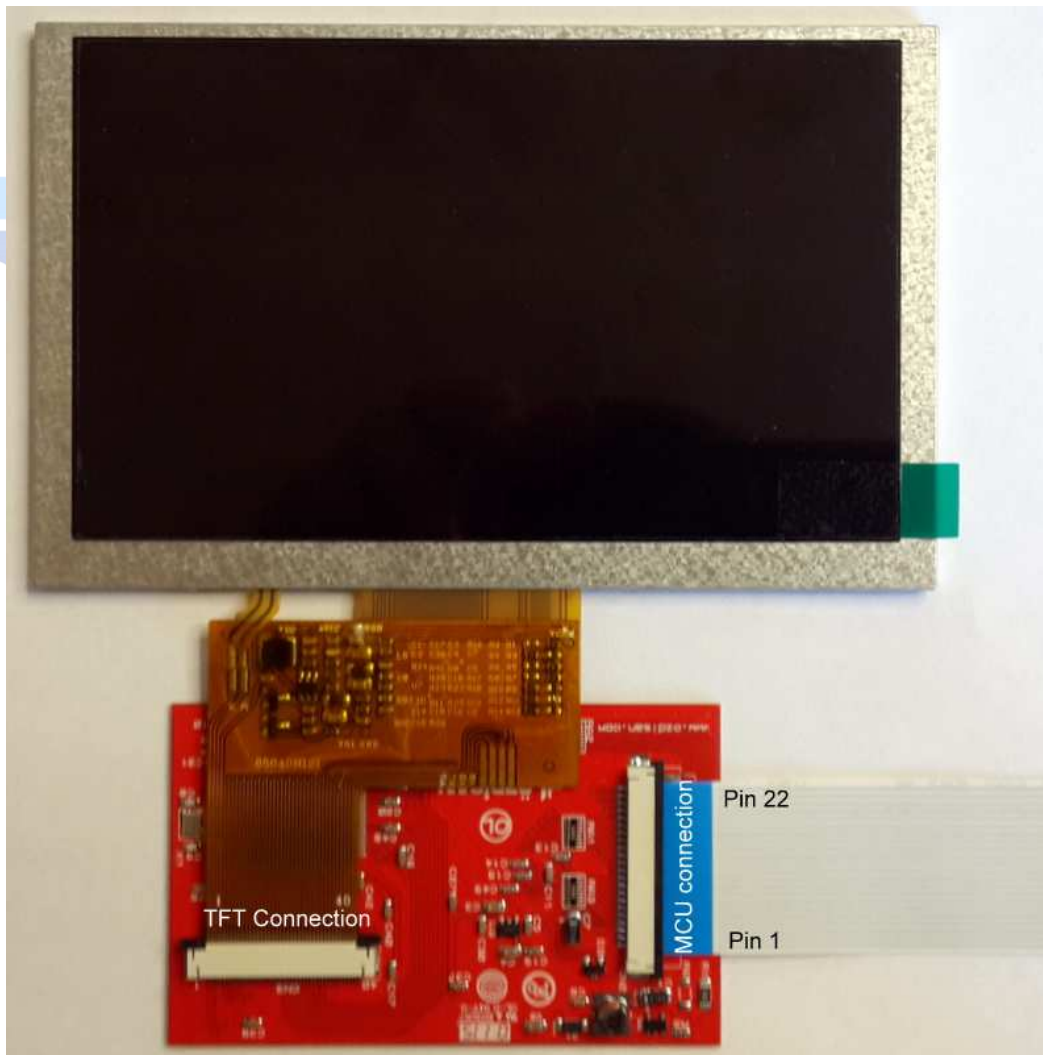
*SPI inputs are used just for TFT controller register settings



Outline Dimensions



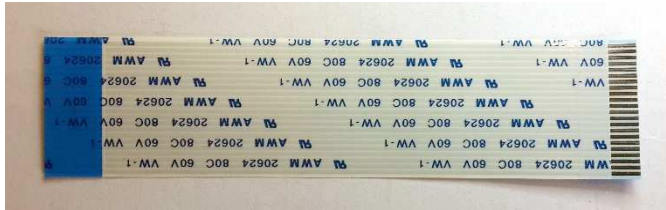
Board Connections





Other Tools used with the boards

DS1057-03-1E22W5L10E1B: Flat cable with 22 pins 10cm



FPC3AMR6-22TNBT-U: Connector with 22 pins



3.5" TFT Panel

SAT035TM54DMR1-A0-01

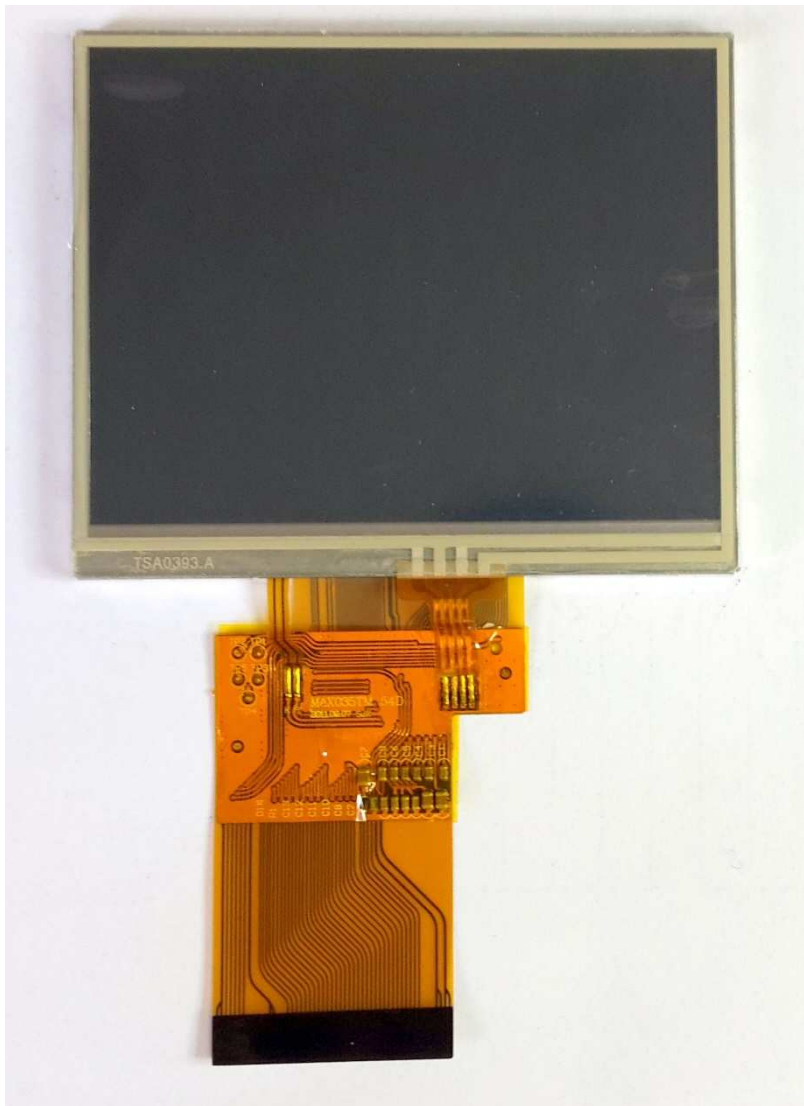


hizdisan
Way of Distribution



3.5" TFT Panel with Touch Panel

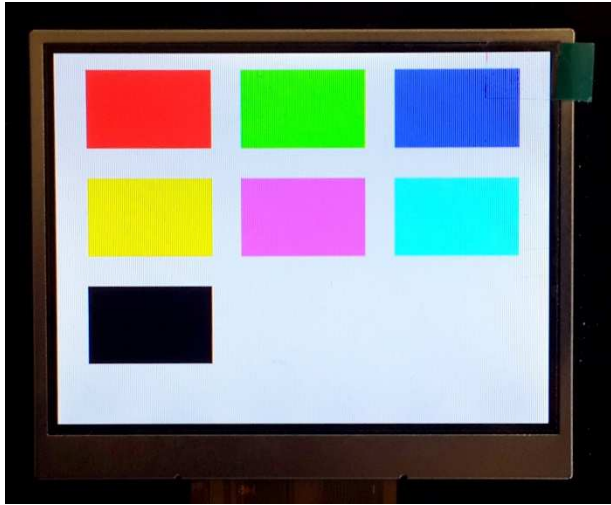
SAT035TM54DMR1-A0-01-TP



SAN
Distribution



Code Sample



TFT screen which is working with this software

```
//pin defines for STM32 in mikroc compiler
unsigned int TFT_DataPort at GPIOE_ODR;
sbit TFT_RST at GPIOB_ODR.B1;
sbit TFT_RS at GPIOC_ODR.B4;
sbit TFT_CS at GPIOC_ODR.B5;
sbit TFT_RD at GPIOA_ODR.B7;
sbit TFT_WR at GPIOA_ODR.B6;
sbit TFT_BLED at GPIOA_ODR.B5;
```

```
void Send_TFT_Command(char index)
```

```
{
  TFT_CS = 0;
  TFT_RS = 0;
  TFT_DataPort = index;
  TFT_WR = 0;
  asm nop;
  TFT_WR = 1;
  TFT_CS = 1;
}
```

```
void Send_TFT_Data_8(unsigned char index)
```

```
{
```




```
TFT_CS = 0;
TFT_RS = 1;
TFT_DataPort = index;
TFT_WR = 0;
asm nop;
TFT_WR = 1;
TFT_CS = 1;
}

void Send_TFT_Data_16(int index)
{
TFT_CS = 0;
TFT_RS = 1;
TFT_DataPort = index;
/*
if data port has 8 bit output
TFT_DataH = index>>8;
TFT_DataL = index&0x00FF;
*/
TFT_WR = 0;
asm nop;
TFT_WR = 1;
TFT_CS = 1;
}

void init_SSD1963(void)
{
TFT_RST = 0; // Hold in reset
TFT_RS = 1; // Enable data access
TFT_CS = 1; // Disable LCD
TFT_RD = 1;
TFT_WR = 1;

// Release from reset
Delay_ms(100);
TFT_RST = 1;
```



Delay_ms(100);

Send_TFT_Command(0x01); //Software Reset

Send_TFT_Command(0x01); //Software Reset

Send_TFT_Command(0x01); //Software Reset

Delay_ms(100);

Send_TFT_Command(0xE2); //SET PLL freq=110MHz

Send_TFT_Data_8(0x21); //N=33, 33X10=330Mhz

Send_TFT_Data_8(0x02); //M=3 330/3=110Mhz

Send_TFT_Data_8(0x54);

Delay_ms(100);

Send_TFT_Command(0xE0); //PLL settings

Send_TFT_Data_8(0x01); //START PLL

delay_ms(1); // Wait to let the PLL stable

Send_TFT_Command(0xE0); //PLL settings

Send_TFT_Data_8(0x03); //LOCK PLL

delay_ms(1);

Send_TFT_Command(0xB0); //LCD MODE Settings

Send_TFT_Data_8(0x20); //TFT panel data width 24bit, FRC, dithering disable

Send_TFT_Data_8(0x00); //hsync+Vsync+DEN

Send_TFT_Data_8(0x01); //horizontal panel size(horizontal lines) HightByte

Send_TFT_Data_8(0x3F); //LowByte

Send_TFT_Data_8(0x00); //vertical panel size(vertical lines) HightByte

Send_TFT_Data_8(0xEF); //SET vertical size LowByte

Send_TFT_Data_8(0x00); //avaible if serial RGB mode is selected.

delay_ms(1);

Send_TFT_Command(0xF0);

Send_TFT_Data_8(0x00); //SSD1963 data input format data 8 bit

delay_ms(1);

Send_TFT_Command(0x3A); //Pixel format

Send_TFT_Data_8(0x60);

delay_ms(1);

Send_TFT_Command(0xE6); //SET PCLK freq=10MHz = 110MHz * LCDC_FPR / 2^20

Send_TFT_Data_8(0x01);

Send_TFT_Data_8(0x45);

Send_TFT_Data_8(0x47);



```

delay_ms(1);
Send_TFT_Command(0xB4);
Send_TFT_Data_8(0x01); //horizontal total period (display + non-display)-1 highbyte
Send_TFT_Data_8(0x97); //low byte
Send_TFT_Data_8(0x00); //Horizontal Pulse Width + Horizontal Back Porch highbyte
Send_TFT_Data_8(0x46); //low byte
Send_TFT_Data_8(0x00); //Horizontal Sync Pulse Width
Send_TFT_Data_8(0x00); //Hsync pulse start position
Send_TFT_Data_8(0x00); //lowbyte
Send_TFT_Data_8(0x00); //for serial RGB mode
delay_ms(1);

```

```

Send_TFT_Command(0xB6);
Send_TFT_Data_8(0x01); //vertical total period (display + non-display)-1 highbyte
Send_TFT_Data_8(0x06); //low byte
Send_TFT_Data_8(0x00); //vertical Pulse Width + vertical Back Porch highbyte
Send_TFT_Data_8(0x0D); //low byte
Send_TFT_Data_8(0x00); //vertical Sync Pulse Width
Send_TFT_Data_8(0x00); //Vsync pulse start position
Send_TFT_Data_8(0x00); //lowbyte
delay_ms(1);

```

////////////////////////////////////

```

Send_TFT_Command(0x36); // Address Mode
Send_TFT_Data_8(0x00);
delay_ms(1);
Send_TFT_Command(0x29); //SET display on

```

//backlight PWM setting.

```

Send_TFT_Command(0xBE);
Send_TFT_Data_8(0x01); //PLL clock / (256 * (PWMF[7:0] + 1)) / 256
Send_TFT_Data_8(0x64); //PWM duty cycle
Send_TFT_Data_8(0x01); //PWM, DBC enable/disable setting.
Send_TFT_Data_8(0x00); //DBC manual brightness
Send_TFT_Data_8(0x00); //DBC minimum brightness
Send_TFT_Data_8(0x00); //Brightness prescaler

```

```

}

```



```

void draw_rectagle(unsigned int X1,unsigned int X2,unsigned int Y1,unsigned int Y2,unsigned char red,unsigned char green,unsigned char blue)
{
    unsigned char X_point_1_1;
    unsigned char X_point_1_2;
    unsigned char X_point_2_1;
    unsigned char X_point_2_2;
    unsigned char Y_point_1_1;
    unsigned char Y_point_1_2;
    unsigned char Y_point_2_1;
    unsigned char Y_point_2_2;

    unsigned long temp1;
    unsigned long temp2;
    unsigned long frame_pixel;
    unsigned long pixel_CNT;

    temp1= X2-X1+1;
    temp2= Y2-Y1+1;
    frame_pixel=temp1*temp2;
    X_point_1_1=X1>>8;
    X_point_1_2=X1&0x00FF;

    X_point_2_1=X2>>8;
    X_point_2_2=X2&0x00FF;

    Y_point_1_1=Y1>>8;
    Y_point_1_2=Y1&0x00FF;

    Y_point_2_1=Y2>>8;
    Y_point_2_2=Y2&0x00FF;

    Send_TFT_Command(0x2A);//Setup the frame buffer vertical addressing range
    Send_TFT_Data_8(X_point_1_1);
    Send_TFT_Data_8(X_point_1_2);
    Send_TFT_Data_8(X_point_2_1);
    Send_TFT_Data_8(X_point_2_2);

```



```
Send_TFT_Command(0x2B);// Setup the frame buffer horizontal address range
```

```
Send_TFT_Data_8(Y_point_1_1);
```

```
Send_TFT_Data_8(Y_point_1_2);
```

```
Send_TFT_Data_8(Y_point_2_1);
```

```
Send_TFT_Data_8(Y_point_2_2);
```

```
Send_TFT_Command(0x2C);
```

```
pixel_CNT=0;
```

```
while(pixel_CNT<=frame_pixel)
```

```
{
```

```
Send_TFT_Data_8(red);
```

```
Send_TFT_Data_8(green);
```

```
Send_TFT_Data_8(blue);
```

```
pixel_CNT++;
```

```
}
```

```
}
```

```
void main()
```

```
{
```

```
///pin configurations for STM32 in MikroC compiler
```

```
GPIO_Digital_Output(&GPIOA_BASE, _GPIO_PINMASK_ALL); // Set PORTB as digital output
```

```
GPIO_Digital_Output(&GPIOB_BASE, _GPIO_PINMASK_ALL); // Set PORTB as digital output
```

```
GPIO_Digital_Output(&GPIOC_BASE, _GPIO_PINMASK_ALL); // Set PORTB as digital output
```

```
GPIO_Digital_Output(&GIOD_BASE, _GPIO_PINMASK_ALL); // Set PORTB as digital output
```

```
GPIO_Digital_Output(&GPIOE_BASE, _GPIO_PINMASK_ALL); // Set PORTB as digital output
```

```
GPIOA_ODR = 0;
```

```
GPIOB_ODR = 0;
```

```
GPIOC_ODR = 0;
```

```
GIOD_ODR = 0;
```

```
GPIOE_ODR = 0;
```

```
GPIOA_OSPEEDR=0xFFFFFFFF;
```

```
GPIOB_OSPEEDR=0xFFFFFFFF;
```

```
GPIOC_OSPEEDR=0xFFFFFFFF;
```

```
GIOD_OSPEEDR=0xFFFFFFFF;
```

```
GPIOE_OSPEEDR=0xFFFFFFFF;
```





```
TFT_BLED=1;
/*
TFT_BLED=1 -> backlight % 100
TFT_BLED=0 and SSD1963 PWM on-> backlight=PWM
TFT_BLED=0 and SSD1963 PWM off-> backlight=off
*/

init_SSD1963();

draw_rectagle(0,319,0,239,255,255,255);

while(1)
{
draw_rectagle(20,100,10,60,255,0,0);
draw_rectagle(120,200,10,60,0,255,0);
draw_rectagle(220,300,10,60,0,0,255);
draw_rectagle(20,100,80,130,255,255,0);
draw_rectagle(120,200,80,130,255,0,255);
draw_rectagle(220,300,80,130,0,255,255);
draw_rectagle(20,100,150,200,0,0,0);
}
}
```

