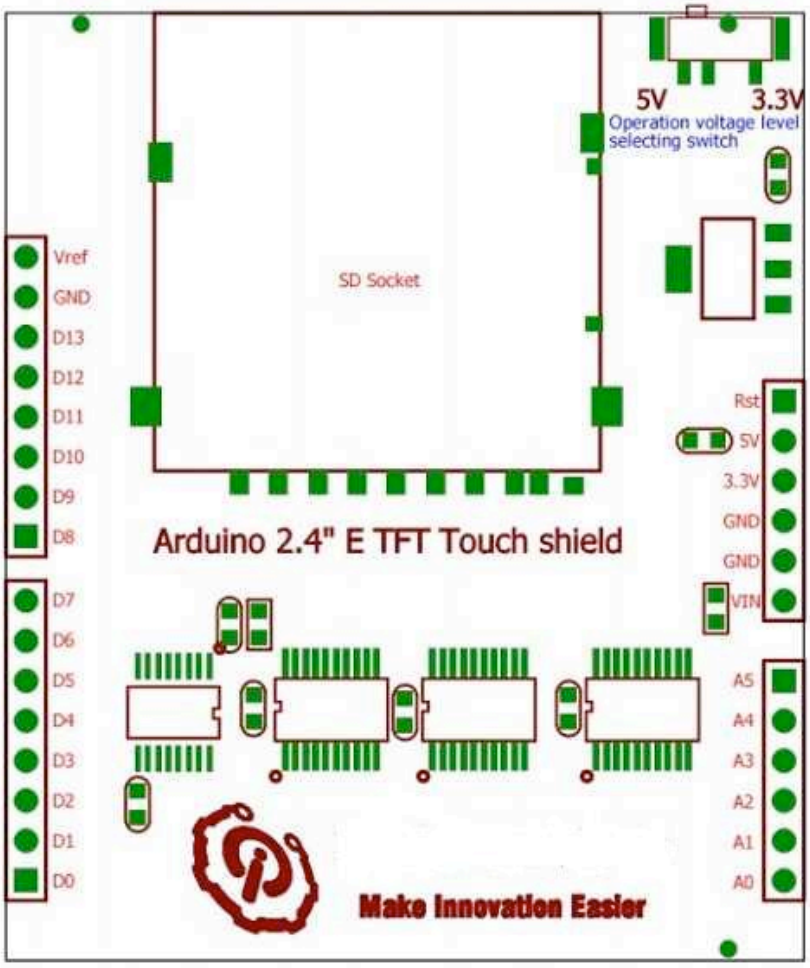
Arduino 2.4” Colored TFT Touch LCD Shield

Description

Arduino 2.4" TFT LCD Touch shield is an Arduino UNO/ Mega compatible multicolored TFT display with touch-screen and SD card socket. It is available in an Arduino shield compatible pinout for attachment. The TFT driver is based on S6D112 with 8bit data and 4bit control interface.

Features

- Compatible with 3.3/5V operation voltage level
- Compatible with UTFT library
- With SD Card Socket



Arduino PIN	Description
D0	DB8
D1	DB9
D2	DB10
D3	DB11
D4	DB12
D5	DB13
D6	DB14
D7	DB15
D8	Touch_Dout
D9	Touch_IRQ

D10	SD_CS
D11	SD_MOSI
D12	SD_MISO
D13	SD_SCK
A0	Touch_Din
A1	Touch_CLK
A2	-
A3	TFT_CS
A4	TFT_WR
A5	TFT_RS

The Arduino 2.4“E TFT Touch shield uses the S6D1121 controller , it support 8bit data interface. The touch IC is TSC2046.

Operation voltage level setting switch

When using the Arduino 2.4”E TFT Touch shield with 5V operation level development board – like the Arduino UNO, Arduino MEGA and so on, set the
operation voltage level switch to 5V side.

When using the Arduino 2.4“E TFT Touch shield with 3.3V operation level development board – like the Iteaduino BT, leaf maple, chipKit UNO and so on,
set the operation voltage level switch to 3.3V side.

Software

This shield is compatible with UTFT graphic library for Arduino
http://henningkarlsen.com/electronics/download.php?f=UTouch.rar

For touch function library
http://henningkarlsen.com/electronics/download.php?f=UTouch.rar

Arduino Mega Code

Pay an attention!

This code is designed for Arduino Mega board.

```
**********************************************/
#include <SdFat.h>
Sd2Card card;
/*********************************************
Define zone
**********************************************/
#define RS 19
#define WR 18
#define CS 17
#define RST 16



#define T_CLK 15
#define T_CS 20
#define T_DIN 14
#define T_DOUT 8
#define T_IRQ 9

#define X_CONST 240
#define Y_CONST 320

#define PREC_TOUCH_CONST 10

#define PixSizeX 13.78
#define PixOffsX 411

#define PixSizeY 11.01
#define PixOffsY 378
```

```
#define WINDOW_XADDR_START        0x0050 // Horizontal Start Address
Set
#define WINDOW_XADDR_END   0x0051 // Horizontal End Address Set
#define WINDOW_YADDR_START        0x0052 // Vertical Start Address
Set
#define WINDOW_YADDR_END   0x0053 // Vertical End Address Set
#define GRAM_XADDR                0x0020 // GRAM Horizontal Address
Set
#define GRAM_YADDR                0x0021 // GRAM Vertical Address Set
#define GRAMWR                    0x0022 // memory write

/* LCD color */
#define White          0xFFFF
#define Black          0x0000
#define Blue           0x001F
#define Blue2          0x051F
#define Red            0xF800
#define Magenta        0xF81F
#define Green          0x07E0
#define Cyan           0x7FFF
#define Yellow         0xFFE0

/*********************************************
Val Zone
*********************************************/
int TP_X,TP_Y;


/*********************************************
Standard C functions zone
*********************************************/

void Write_Command(unsigned int c)
{
   digitalWrite(RS,LOW);//LCD_RS=0;
       digitalWrite(CS,LOW);//LCD_CS =0;
       PORTD = c>>8; //LCD_DataPortH=DH>>8;
       digitalWrite(WR,LOW);//LCD_WR=0;
```

```
        digitalWrite(WR,HIGH);//LCD_WR=1;
        PORTD = c;//LCD_DataPortH=DH;
        digitalWrite(WR,LOW);//LCD_WR=0;
        digitalWrite(WR,HIGH);//LCD_WR=1;
        digitalWrite(CS,HIGH);//LCD_CS =0;
}


void Write_Data(unsigned int c)
{
   digitalWrite(RS,HIGH);//LCD_RS=0;
        digitalWrite(CS,LOW);//LCD_CS =0;
        PORTD = c>>8; //LCD_DataPortH=DH>>8;
        digitalWrite(WR,LOW);//LCD_WR=0;
        digitalWrite(WR,HIGH);//LCD_WR=1;
        PORTD = c;//LCD_DataPortH=DH;
        digitalWrite(WR,LOW);//LCD_WR=0;
        digitalWrite(WR,HIGH);//LCD_WR=1;
        digitalWrite(CS,HIGH);//LCD_CS =0;
}

void Write_Command_Data(unsigned int cmd,unsigned int dat)
{
        Write_Command(cmd);
        Write_Data(dat);
}

void Lcd_Init()
{
        pinMode(RS,OUTPUT);
        pinMode(WR,OUTPUT);
        pinMode(CS,OUTPUT);
        pinMode(RST,OUTPUT);

        DDRD = 0xFF;

        digitalWrite(RST,HIGH);
   delay(1);
```

```
        digitalWrite(RST,LOW);
        delay(1);

        digitalWrite(RST,HIGH);
        digitalWrite(CS,HIGH);
        digitalWrite(WR,HIGH);
        delay(20);


        Write_Command_Data(0x0011,0x2004);
    Write_Command_Data(0x0013,0xCC00);
    Write_Command_Data(0x0015,0x2600);
        Write_Command_Data(0x0014,0x252A);
//      Write_Command_Data(0x14,0x002A);
        Write_Command_Data(0x0012,0x0033);
        Write_Command_Data(0x0013,0xCC04);
        //delayms(1);
        Write_Command_Data(0x0013,0xCC06);
        //delayms(1);
        Write_Command_Data(0x0013,0xCC4F);
        //delayms(1);
        Write_Command_Data(0x0013,0x674F);
        Write_Command_Data(0x0011,0x2003);
        //delayms(1);
        Write_Command_Data(0x0030,0x2609);
        Write_Command_Data(0x0031,0x242C);
        Write_Command_Data(0x0032,0x1F23);
        Write_Command_Data(0x0033,0x2425);
        Write_Command_Data(0x0034,0x2226);
        Write_Command_Data(0x0035,0x2523);
        Write_Command_Data(0x0036,0x1C1A);
        Write_Command_Data(0x0037,0x131D);
        Write_Command_Data(0x0038,0x0B11);
        Write_Command_Data(0x0039,0x1210);
        Write_Command_Data(0x003A,0x1315);
        Write_Command_Data(0x003B,0x3619);
        Write_Command_Data(0x003C,0x0D00);
        Write_Command_Data(0x003D,0x000D);
```

```
        Write_Command_Data(0x0016,0x0007);
        Write_Command_Data(0x0002,0x0013);
        Write_Command_Data(0x0003,0x0003);
        Write_Command_Data(0x0001,0x0127);
        //delayms(1);
        Write_Command_Data(0x0008,0x0303);
        Write_Command_Data(0x000A,0x000B);
        Write_Command_Data(0x000B,0x0003);
        Write_Command_Data(0x000C,0x0000);
        Write_Command_Data(0x0041,0x0000);
        Write_Command_Data(0x0050,0x0000);
        Write_Command_Data(0x0060,0x0005);
    Write_Command_Data(0x0070,0x000B);
        Write_Command_Data(0x0071,0x0000);
        Write_Command_Data(0x0078,0x0000);
        Write_Command_Data(0x007A,0x0000);
        Write_Command_Data(0x0079,0x0007);
        Write_Command_Data(0x0007,0x0051);
        //delayms(1);
        Write_Command_Data(0x0007,0x0053);
        Write_Command_Data(0x0079,0x0000);

        Write_Command(0x0022);

}
void SetXY(unsigned int x0,unsigned int x1,unsigned int y0,unsigned int y1)
{
  Write_Command_Data(0x0046,(x1 << 8)| x0);
  //Write_Command_Data(0x0047,x1);
  Write_Command_Data(0x0047,y1);
  Write_Command_Data(0x0048,y0);
  Write_Command_Data(0x0020,x0);
  Write_Command_Data(0x0021,y0);
  Write_Command (0x0022);//LCD_WriteCMD(GRAMWR);
}
void Pant(unsigned int color)
{
        int i,j;
```

```
        SetXY(0,239,0,319);

    for(i=0;i<320;i++)
         {
          for (j=0;j<240;j++)
               {
        Write_Data(color);
           }

         }
}
void LCD_clear()
{
    unsigned int i,j;
        SetXY(0,239,0,319);
        for(i=0;i<X_CONST;i++)
        {
           for(j=0;j<Y_CONST;j++)
               {
               Write_Data(0x0000);
               }
        }
}

void Touch_Init(void)
{
        pinMode(T_CLK,  OUTPUT);
    pinMode(T_CS,   OUTPUT);
    pinMode(T_DIN,  OUTPUT);
    pinMode(T_DOUT, INPUT);
    pinMode(T_IRQ,  INPUT);

        digitalWrite(T_CS,  HIGH);
        digitalWrite(T_CLK, HIGH);
        digitalWrite(T_DIN, HIGH);
        digitalWrite(T_CLK, HIGH);
}
```

```
void Touch_WriteData(unsigned char data)
{
	unsigned char temp;
	unsigned char nop;
	unsigned char count;

	temp=data;
	digitalWrite(T_CLK,LOW);

	for(count=0; count<8; count++)
	{
		if(temp & 0x80)
			digitalWrite(T_DIN, HIGH);
		else
			digitalWrite(T_DIN, LOW);
		temp = temp << 1;
		digitalWrite(T_CLK, LOW);
		nop++;
		digitalWrite(T_CLK, HIGH);
		nop++;
	}
}

unsigned int Touch_ReadData()
{
	unsigned char nop;
	unsigned int data = 0;
	unsigned char count;
	for(count=0; count<12; count++)
	{
		data <<= 1;
		digitalWrite(T_CLK, HIGH);
		nop++;
		digitalWrite(T_CLK, LOW);
		nop++;
		if (digitalRead(T_DOUT))
			data++;
	}
```

```
        return(data);
}
void Touch_Read()
{
        unsigned long tx=0;
        unsigned long ty=0;

        digitalWrite(T_CS,LOW);

        for (int i=0; i<PREC_TOUCH_CONST; i++)
        {
                Touch_WriteData(0x90);
                digitalWrite(T_CLK,HIGH);
                digitalWrite(T_CLK,LOW);
                ty+=Touch_ReadData();

                Touch_WriteData(0xD0);
                digitalWrite(T_CLK,HIGH);
                digitalWrite(T_CLK,LOW);
                tx+=Touch_ReadData();
        }

        digitalWrite(T_CS,HIGH);

        TP_X=tx/PREC_TOUCH_CONST;
        TP_Y=ty/PREC_TOUCH_CONST;
}

char Touch_DataAvailable()
{
  char avail;
  avail = !digitalRead(T_IRQ);
  return avail;
}

int Touch_GetX()
{
        int value;
```

```
        value = ((TP_X-PixOffsX)/PixSizeX);
        if (value < 0)
                value = 0;
        return value;
}
int Touch_GetY()
{
        int value;
        value = ((TP_Y-PixOffsY)/PixSizeY);
        if (value < 0)
                value = 0;
        return value;
}
/*********************************************
Arduino functions zone
*********************************************/
void setup()
{

  Lcd_Init();
  if (!card.init(SPI_EIGHTH_SPEED, 10))
  {
    while(1)
    {
      Pant(0xffff);
      Pant(0x0000);
    }
  }
  Touch_Init();
  LCD_clear();
        Pant(0xf800);
        Pant(0x07e0);
        Pant(0x001f);
        Pant(0xffff);
        Pant(0x0000);
}

void loop()
```

```
{
   unsigned int  i,j;
       while(Touch_DataAvailable() == 1)
       {
              Touch_Read();
              i = Touch_GetX();
              j = Touch_GetY();
              SetXY(i,i,j,j);
              Write_Data(0xFFFF);
       }


}
```