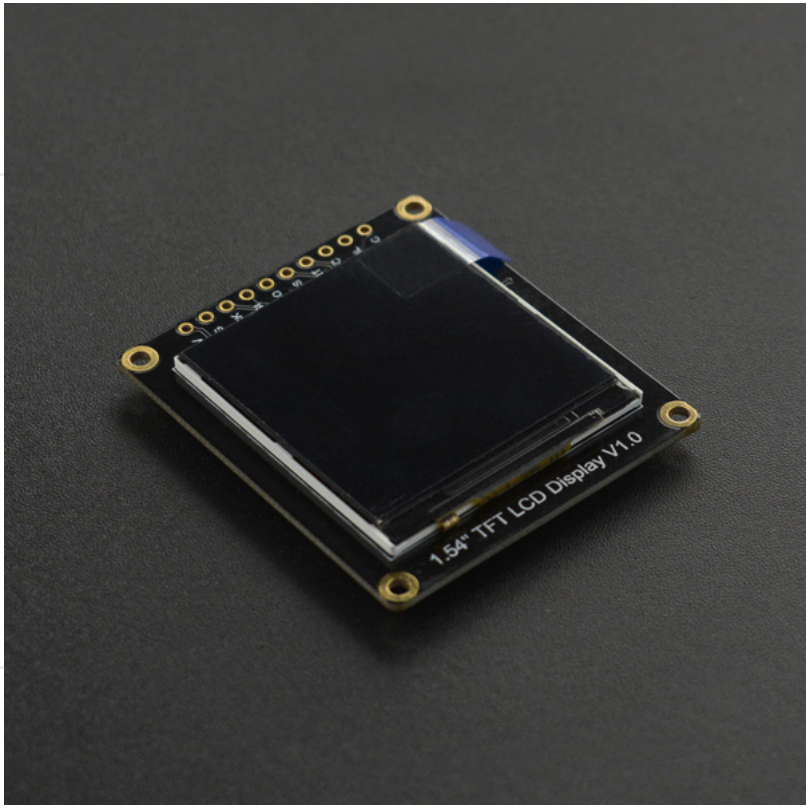
SKU:DFR0649 (https://www.dfrobot.com/product-2072.html)

(https://www.dfrobot.com/product-2072.html)



Introduction

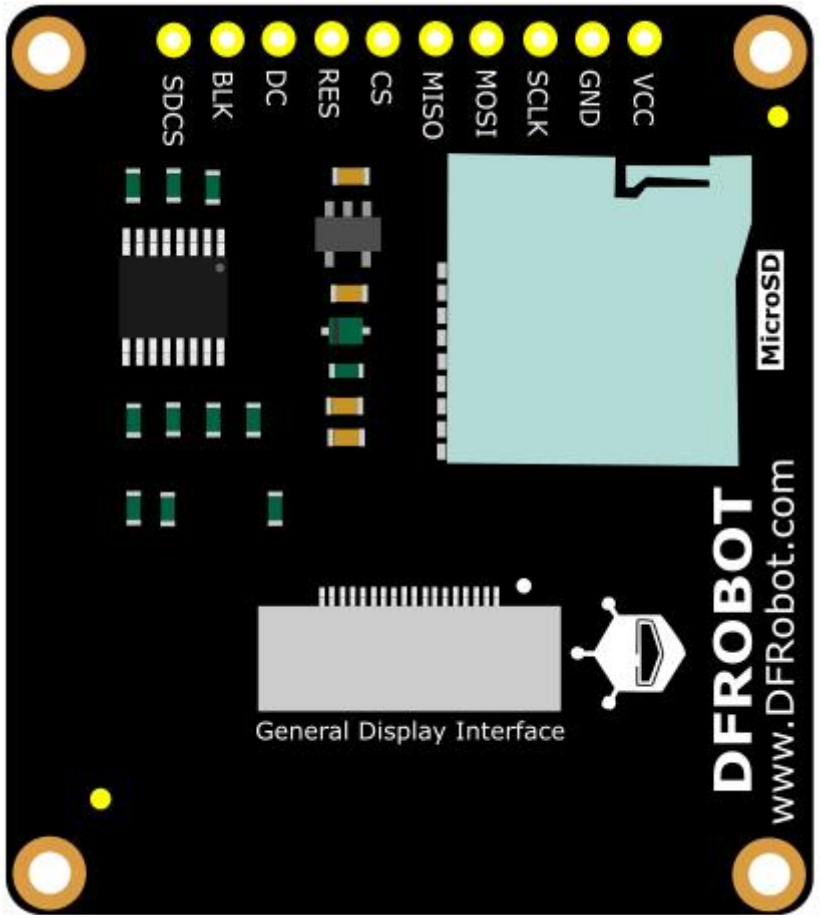
This display module features high resolution, low power consumption, wide angle and easy wiring. It employs IPS display with a small size of 1.54 inches, offering 240×240 resolution. The module adopts SPI and GDI interface(work with maincontrollers with GDI port). This LCD display can be powered by 3.3V~5V, and the maximum is power consumption is 24Ma. This product can be used in many display applications: waveform monitor display, electronic gift box, electronic weather decorations, etc.

Specification

- Operating Voltage: 3.3V~5V
- IPS Angle of View: 80/80/80/80
- Color Depth: 16-bit (RGB565)
- Pixels: 240 × 240
- Connection Port: SPI
- Driver Chip: ST7789
- Brightness: 250 (Typ) cd/m2

- Brightness: 250 (Typ) cd/m2
- Full-screen Power Consumption: about 17mA (3.3V) 17mA (5V) (Typ)
- Operating Temperature: -30℃~+70℃
- Display Area: 27.72×27.72mm
- Mount Hole Diameter: 2mm
- Dimension: 44.00x39.00mm
- Weight: g

Board Overview



Num	Label	Description
1	VCC	Positive
2	GND	Negative
3	SCLK	Clock
4	MOSI	Data (Master send; Slave receive)
5	MISO	Data (Master receive; Slave send)
6	CS	Screen Chip Select
7	RES	Reset
8	DC	Data/Command
9	BL	Backlight The backlight has been set to a default value, and can be turned on without connecting the backlight pin. When the backlight pin is connected, input High level(1) to turn the backlight brightness to maximum; Input Low level to turn off backlight.
10	SDCS	SD card chip select

Dimension Diagram

- Dimension：44.00x39.00mm/1.73x1.54”
- Mount Hole Pitch：40mm/35mm
- Mount Hole Size：2.0mm

Tutorial

The product is a Breakout module. It adopts SPI communication and has onboard GDI interface, which reduces the complexity of wiring and can easily display the contents read from SD card.

⚠ **NOTE**

1. GDI interface could only work well with maincontrollers with GDI.
2. It is recommended to use Arduino 1.8.10 and above.
3. If the SD card is not in good contact, it may fail to initialize, and then please try to replug it.

Requirements

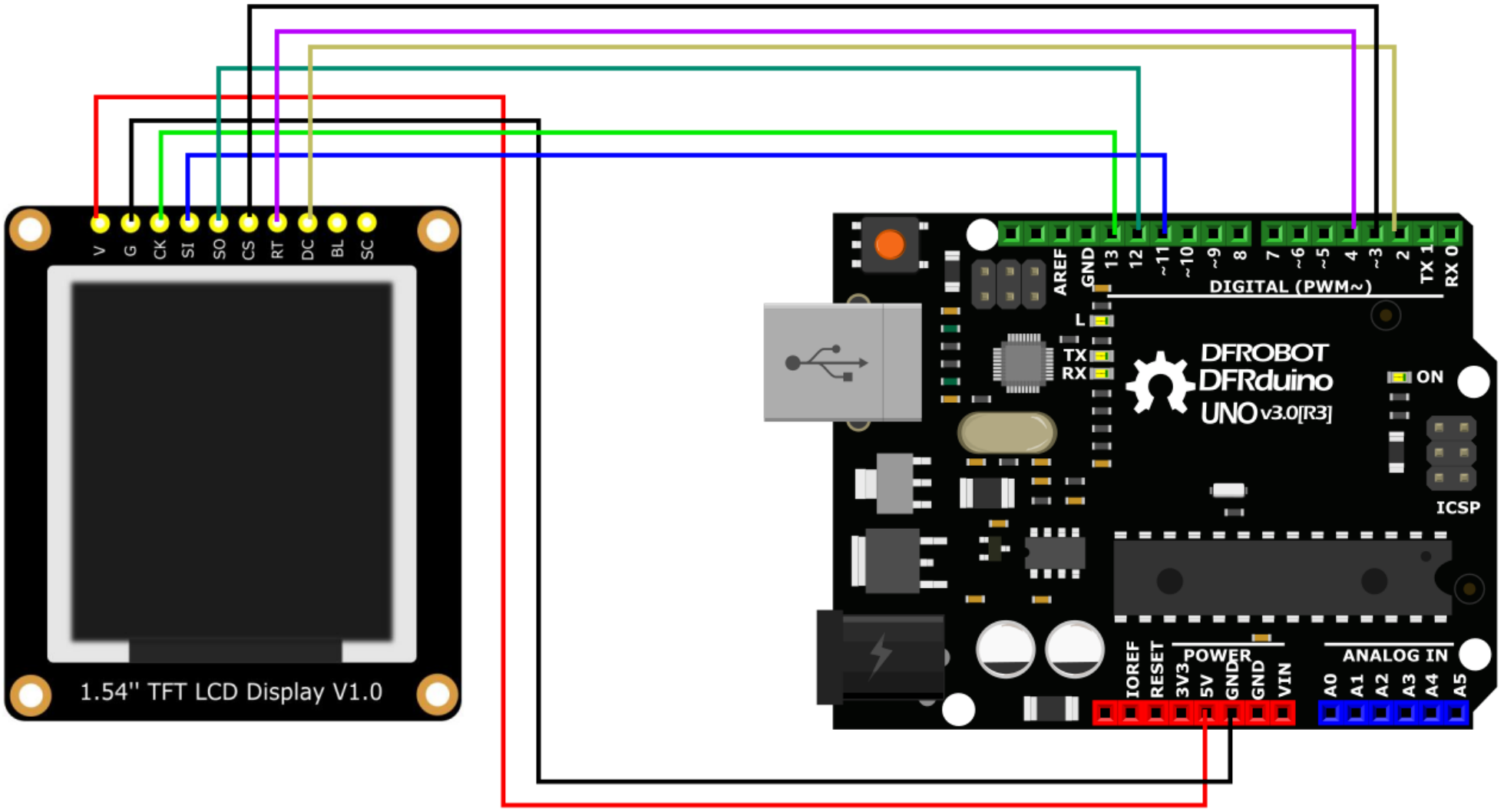
- **Hardware**
  - DFRduino UNO R3 (https://www.dfrobot.com/product-838.html) (or similar) x 1
  - 1.54" 240x240 LCD Module x 1
  - M-M/F-M/F-F Jumper wires
  - Wires
- **Software**
  - Arduino IDE (https://www.arduino.cc/en/Main/Software)
  - Download and install the **DFRobot_GDL Library** (https://codeload.github.com/DFRobot/DFRobot_GDL/zip/master) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))
  - DFRobot_GDL API Function, click to find more detailed information (https://github.com/DFRobot/DFRobot_GDL/wiki/English-WIKI)

⚠ **NOTE**

1. All the demo files of this product are stored in file DFRobot_GDL->example->basic.

2. Please open the corresponding constructed functions (DFRobot_ST7789_240x240_HW_SPI) before burning demo into your device.

Connection Diagram



Sample Code 1

This is a basic sample to show you how to draw point, circle, line, and rectangle on the module.

```
v/*!
 * @file basicTest.ino
 * @brief Demonstrate various graphic painting effects
 * @n This demo supports Arduino Uno, Leonardo, Mega2560, FireBeetle-ESP32, FireBeetle-ESP8266, and FireBeetle-M0.
 * @copyright Copyright (c) 2010 DFRobot Co. Ltd (http://www.dfrobot.com)
 * @licence The MIT License (MIT)
 * @author [LuoYufeng] (yufeng.luo@dfrobot.com)
 * @version V0.1
 * @date 2020-01-07
 * @url https://github.com/DFRobot/DFRobot_GDL
 */
#include "DFRobot_GDL.h"
/*M0*/
#if defined ARDUINO_SAM_ZERO
#define TFT_DC  7
#define TFT_CS  5
#define TFT_RST 6
/*ESP32 and ESP8266*/
#elif defined(ESP32) || defined(ESP8266)
#define TFT_DC  D3
#define TFT_CS  D4
#define TFT_RST D5
/*AVR series mainboard*/
#else
#define TFT_DC  2
#define TFT_CS  3
#define TFT_RST 4
#endif

/**
 * @brief Constructor Constructor of hardware SPI communication
 * @param dc Command/data line pin for SPI communication
```

```
 * @param dc Command/data line pin for SPI communication
 * @param cs Chip select pin for SPI communication
 * @param rst reset pin of the screen
 */
DFRobot_ST7789_240x240_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);

//DFRobot_ST7789_240x320_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9341_240x320_HW_SPI  screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9488_320x480_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
/* M0 mainboard DMA transfer */
//DFRobot_ST7789_240x240_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ST7789_240x320_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9341_240x320_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9488_320x480_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);



/*
 *User-selectable macro definition color
 *COLOR_RGB565_BLACK   COLOR_RGB565_NAVY    COLOR_RGB565_DGREEN   COLOR_RGB565_DCYAN
 *COLOR_RGB565_MAROON  COLOR_RGB565_PURPLE  COLOR_RGB565_OLIVE    COLOR_RGB565_LGRAY
 *COLOR_RGB565_DGRAY   COLOR_RGB565_BLUE    COLOR_RGB565_GREEN    COLOR_RGB565_CYAN
 *COLOR_RGB565_RED     COLOR_RGB565_MAGENTA COLOR_RGB565_YELLOW   COLOR_RGB565_ORANGE
 *COLOR_RGB565_WHITE
 */

void setup() {
  Serial.begin(115200);
  screen.begin();
}

void loop(){
    testDrawPixel();
    testLine();
    testFastLines(COLOR_RGB565_PURPLE,COLOR_RGB565_YELLOW);
    testRects(COLOR_RGB565_BLACK,COLOR_RGB565_WHITE);
    testRoundRects();
    testCircles(24,COLOR_RGB565_BLUE);
    testTriangles(COLOR_RGB565_YELLOW);
```

```
    testPrint();

}

/* Test to draw a pixel*/

void testDrawPixel() {
  //Clear screen
  screen.fillScreen(COLOR_RGB565_BLACK);
  int x = 0;
  int y = screen.height();
  for(int i = 0; i <= screen.width()/2; i += 10){
    for (x = screen.width() - i; x >= i; x-=10 ){
      /*
       * @ brief draw a pixel
       * @ param x coordinate
       *         y coordinate
       * c pixel color
       */
      screen.drawPixel(x, y, COLOR_RGB565_ORANGE);
      delay(10);
    }

    for (y = screen.height() - i; y >= i; y-=10){
      screen.drawPixel(x, y, COLOR_RGB565_ORANGE);
      delay(10);
    }

    for (x = i; x <= screen.width() - i + 1; x+=10 ){
      screen.drawPixel(x, y, COLOR_RGB565_ORANGE);
      delay(10);
    }

    for (y = i; y <= screen.height() - i + 1; y+=10){
      screen.drawPixel(x, y, COLOR_RGB565_ORANGE);
      delay(10);
    }
  }
```

```
}

/* Test to draw a line*/
void testLine(){
// 0x00FF is the color data in the format of RGB565

  uint16_t color = 0x00FF;
  screen.fillScreen(COLOR_RGB565_BLACK);
  for (int16_t x=0; x < screen.width(); x+=6) {
    /*
     * @ brief draw a line
     * @ param x0 The x-coordinate of the first vertex
     *         y0 The y-coordinate of the first vertex
     *         x1 The x-coordinate of the second vertex
     *         y1 The y-coordinate of the second vertex
     *         c line color
     */
    screen.drawLine(/*x0=*/screen.width()/*Screen width*//2, /*y0=*/screen.height()/*Screen height*//2, /*x1=*/x, /*y1=*/0, /*c=*/colo
  }
  for (int16_t y=0; y < screen.height(); y+=6) {
    screen.drawLine(screen.width()/2, screen.height()/2, screen.width(), y, color+=0x0700);
  }

  for (int16_t x = screen.width(); x >= 0; x-=6) {
    screen.drawLine(screen.width()/2, screen.height()/2, x,screen.height(), color+=0x0700);
  }

  for (int16_t y = screen.height(); y >= 0; y-=6) {
    screen.drawLine(screen.width()/2, screen.height()/2, 0, y, color+=0x0700);
  }
}

/* Test to fast draw line(need to set delay), only horizontal line and vertical line */
void testFastLines(uint16_t color1, uint16_t color2) {
  for (int16_t y=0; y < screen.height(); y+=4) {
    /*
     * @ brief draw a line
     * @ param x The x-coordinate of the first vertex
```

```
     *         y The y-coordinate of the first vertex
     *         w Length of line segment
     *         c line color
     */
    screen.drawFastHLine(/*x=*/0, /*y=*/y, /*w=*/screen.width(),/*c=*/color2);

    delay(10);
  }

  for(int16_t x=0; x < screen.width(); x+=3) {
    /*
     * @ brief draw a line
     * @ param x The x-coordinate of the first vertex
     *         y The y-coordinate of the first vertex
     *         h length of line segment
     *         c line color
     */
    screen.drawFastVLine(/*x=*/x, /*y=*/0, /*h=*/screen.height(), /*c=*/color1);
    delay(10);
  }
}

/* Test to draw a rectangle*/
void testRects(uint16_t color1, uint16_t color2) {
    screen.fillScreen(COLOR_RGB565_BLACK);
    int16_t x=screen.width()-12;
    for (; x > 100; x-=screen.width()/40) {
      /*
       * @ brief draw a hollow rectangle
       * @ param x The x-coordinate of the vertex
       * @ param y The y-coordinate of the vertex
       * @ param w horizontal side length
       * @ param h longitudinal side length
       * @ param color Fill color, RGB color with 565 structure
       */
      screen.drawRect(/*x=*/screen.width()/2 -x/2, /*y=*/screen.height()/2 -x/2 , /*w=*/x, /*h=*/x, /*color=*/color2+=0x0F00);
      delay(100);
    }
```

```
    /*
     * @brief draw a filled rectangle
     * @param x The x-coordinate of the vertex
     * @param y The y-coordinate of the vertex
     * @param w horizontal side length
     * @param h longitudinal side length
     * @param color Fill color, RGB color with 565 structure
     */
    screen.fillRect(/*x=*/screen.width()/2 -x/2, /*y=*/screen.height()/2 -x/2 , /*w=*/x, /*h=*/x, /*color=*/color2);
    delay(100);
    for(; x > 6; x-=screen.width()/40){
      screen.drawRect(screen.width()/2 -x/2, screen.height()/2 -x/2 , x, x, color1);
      delay(100);
    }
}

/* Test to draw a rounded rectangle */
void testRoundRects() {
  screen.fillScreen(COLOR_RGB565_BLACK);
// 0xF00F is the color data in the format of RGB565
  int color = 0xF00F;
  int i;
  int x = 0;
  int y = 0;
  int w = screen.width()-3;
  int h = screen.height()-3;
  for(i = 0 ; i <= 16; i+=2) {
    /*
     * @brief Draw a hollow rounded rectangle
     * @param x0 The x-coordinate of the start vertex
     * @param y0 The y-coordinate of the start vertex
     * @param w horizontal side length
     * @param h longitudinal side length
     * @param radius Round corner radius
     * @param color border color, 565 structure RGB color
     */
```

```
    screen.drawRoundRect(/*x0=*/x, /*y0=*/y, /*w=*/w, /*h=*/h, /*radius=*/20, /*color=*/color);
    x+=5;
    y+=5;
    w-=10;
    h-=10;

    color+=0x0100;
    delay(50);
  }
  for(i = 0 ; i <= 16; i+=2) {
    /*
     * @ brief Draw a filled and rounded rectangle
     * @ param x0 The x-coordinate of the start vertex
     * @ param y0 The y-coordinate of the start vertex
     * @ param w horizontal side length
     * @ param h longitudinal side length
     * @ param radius Round corner radius
     * @ param color Fill color, RGB color with 565 structure
     */
    screen.fillRoundRect(/*x0=*/x, /*y0=*/y, /*w=*/w, /*h=*/h, /*radius=*/10, /*color=*/color);
    x+=5;
    y+=5;
    w-=10;
    h-=10;
    color+=0x0500;
    delay(50);
  }
}

/* Test to draw a circle */
void testCircles(uint8_t radius, uint16_t color) {
  screen.fillScreen(COLOR_RGB565_BLACK);
  for (int16_t x=radius; x <=screen.width()-radius; x+=radius*2) {
    for (int16_t y=radius; y <=screen.height()-radius; y+=radius*2) {
      /*
       * @ brief Draw a hollow circle
       * @ param x0 The x-coordinate of the center point
       * @ param y0 The y-coordinate of the center point
```

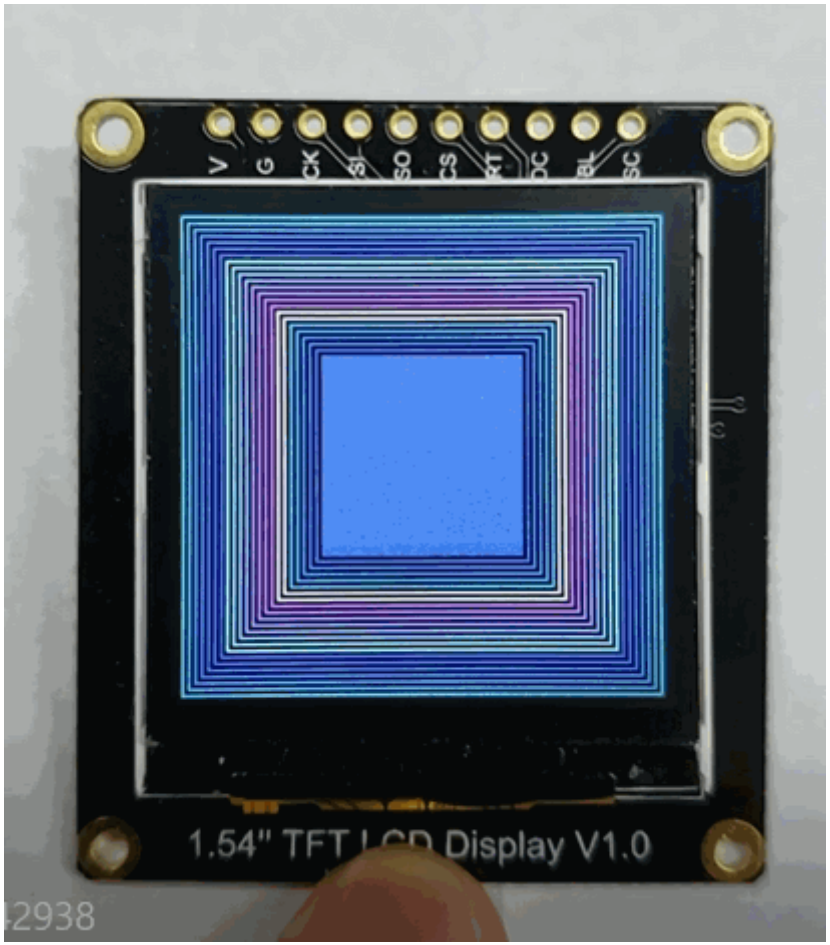
```
       * @ param r radius
       * @ param color Circle color, RGB color with 565 structure
       */
      screen.drawCircle(/*x0=*/x, /*y0=*/y, /*r=*/radius, /*color=*/color);
        if(x == y ||x == -y ||x == y + 2*radius)

          /*
           * @ brief Draw a filled circle
           * @ param x0 The x-coordinate of the center point
           * @ param y0 The y-coordinate of the center point
           * @ param r radius
           * @ param color Fill color, RGB color with 565 structure
           */
          screen.fillCircle(/*x0=*/x, /*y0=*/y, /*r=*/radius, /*color=*/color);
       color += 800;
       delay(100);
    }
  }
}

/* Test to draw a triangle */
void testTriangles(uint16_t color){
  screen.fillScreen(COLOR_RGB565_BLACK);

  for (int16_t i=0; i <=screen.width(); i+=24)
    /*
     * @ brief Draw a hollow triangle
     * @ param x0 The x-coordinate of the start vertex
     * @ param y0 The y-coordinate of the start vertex
     * @ param x1 The x-coordinate of the second vertex
     * @ param y1 The y-coordinate of the second vertex
     * @ param x2 The x-coordinate of the third vertex
     * @ param y2 The y-coordinate of the third vertex
     * @ param color border color, 565 structure RGB color
     */
    screen.drawTriangle(/*x0=*/i,/*y0=*/0,/*x1=*/0,/*y1=*/screen.height()-i,/*x2=*/screen.width()-i,/*y2=*/screen.height(), /*color=*/

  for (int16_t i=0; i <screen.width(); i+=24)
```

```
    screen.drawTriangle(screen.width(),i*4/3,0,screen.height()-i*4/3,i,0, color);

  for (int16_t i=0; i <screen.width(); i+=24)
    screen.drawTriangle(screen.width(),i*4/3,i,0,screen.width()-i,screen.height(), color);


  color = COLOR_RGB565_RED;
  for (int16_t i=0; i <=screen.width(); i+=24)
     /*
      * @ brief Draw a filled triangle
      * @ param x0 The x-coordinate of the start vertex
      * @ param y0 The y-coordinate of the start vertex
      * @ param x1 The x-coordinate of the second vertex
      * @ param y1 The y-coordinate of the second vertex
      * @ param x2 The x-coordinate of the third vertex
      * @ param y2 The y-coordinate of the third vertex
      * @ param color Fill color, RGB color with 565 structure
      */
    screen.fillTriangle(/*x0=*/i,/*y0=*/0,/*x1=*/0,/*y1=*/screen.height()-i,/*x2=*/screen.width()-i,/*y2=*/screen.height(), /*color=*/

  for (int16_t i=0; i <screen.width(); i+=24)
    screen.fillTriangle(screen.width(),i*4/3,0,screen.height()-i*4/3,i,0, color+=100);

  for (int16_t i=0; i <screen.width(); i+=24)
    screen.fillTriangle(screen.width(),i*4/3,i,0,screen.width()-i,screen.height(), color+=100);
}

void testPrint() {
  // 0x00FF is the color data in the format of RGB565
  int16_t color = 0x00FF;
   // Set text wrapping mode
   // true = Text word wrap, false = No word wrap
  screen.setTextWrap(false);
  //Fill color, RGB color with 565 structure
  screen.fillScreen(COLOR_RGB565_BLACK);

  //Set the coordinate position x = 0, y = 50
  screen.setCursor(0, 50);
```

```
//Set the text color; this is a changeable value
screen.setTextColor(color+=0x3000);
//Set text size to 0
screen.setTextSize(0);
//Output text

screen.println("Hello World!");

screen.setTextColor(color+=0x3000);
//Set text size to 1
screen.setTextSize(1);
screen.println("Hello World!");

screen.setTextColor(color+=0x3000);
//Set text size to 2
screen.setTextSize(2);
screen.println("Hello World!");

screen.setTextColor(color+=0x3000);
//Set text size to 3
screen.setTextSize(3);
screen.println("Hello World!");

screen.setTextColor(color+=0x3000);
//Set text size to 4
screen.setTextSize(4);
screen.println("Hello!");
//Set text size to 5
screen.setTextSize(5);
screen.print("Hello!");
delay(2000);

//Set coordinate position x = 0, y = 0
screen.setCursor(0, 0);
//Fill color, RGB color with 565 structure
screen.fillScreen(COLOR_RGB565_BLACK);
screen.setTextSize(2);
screen.setTextColor(color+=0x3000);
```

```
  screen.print("a = ");

  screen.setTextColor(color+=0x3000);
  int a = 1234;
  screen.println(a, 1);

  screen.setTextColor(color+=0x3000);
  screen.print(8675309, HEX);
  screen.println("this is HEX!");
  screen.println("");

  screen.setTextColor(color+=0x0F00);
  screen.println("running for: ");
  screen.setTextColor(color+=0x0F00);
  //Output time in millisecond
  screen.print(millis());
  screen.setTextColor(color+=0x0F00);
  screen.println("/1000 seconds.");

  char *text = "Hi DFRobot!";
  screen.setTextColor(color+=0x0F00);
  screen.setTextWrap(true);
  screen.setTextSize(3);
  screen.println(text);
  //screen.setFonts((const gdl_Font_t *)SIMKAIFont18ptBitmaps);
  screen.println(text);
  delay(2000);
}
```

Result 1



Sample Code 2-icon

This is an example of commonly-used icons. 1. We use GIMP2 (https://www.gimp.org/downloads/) to convert these icons into codes for better display. 2. We provide some icons for you, Click here to find more (https://github.com/DFRobot/DFRobot_Icon/archive/master.zip).

```
v/*!
 * @file icon.ino
 * @brief Display some small icons on the screen
 * @n The demo supports Arduino Uno, Leonardo, Mega2560, FireBeetle-ESP32, FireBeetle-ESP8266, FireBeetle-M0
 * @copyright Copyright (c) 2010 DFRobot Co. Ltd (http://www.dfrobot.com)
 * @licence The MIT License (MIT)
 * @author [YeHangYu] (hangyu.ye@dfrobot.com)
 * @version V0.1
 * @date 2020-01-07
 * @url https://github.com/DFRobot/DFRobot_GDL
 */
#include "DFRobot_GDL.h"
#include "Icon.h"
//Custom communication pins
/*M0*/
#if defined ARDUINO_SAM_ZERO
#define TFT_DC  7
#define TFT_CS  5
#define TFT_RST 6
/*ESP32 and ESP8266*/
#elif defined(ESP32) || defined(ESP8266)
#define TFT_DC  D3
#define TFT_CS  D4
#define TFT_RST D5
/* AVR series mainboard */
#else
#define TFT_DC  2
#define TFT_CS  3
#define TFT_RST 4
#endif

/**
```

```
/**
 * @brief Constructor Constructor of hardware SPI communication
 * @param dc Command/data line pin for SPI communication
 * @param cs Chip select pin for SPI communication
 * @param rst Reset pin of the screen

 */
DFRobot_ST7789_240x240_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ST7789_240x320_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9341_240x320_HW_SPI  screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9488_320x480_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
/* M0 mainboard DMA transfer */
//DFRobot_ST7789_240x240_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ST7789_240x320_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9341_240x320_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9488_320x480_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);



void setup() {
  Serial.begin(115200);
  screen.begin();
}

void loop() {
  int w = screen.width();
  int h = screen.height();
  int a = millis()/1000;//Get Time
     //0x00FF is color data in the format RGB565
  uint16_t color = 0x00FF;

  screen.fillScreen(COLOR_RGB565_WHITE);

  while(1) {
    for(int i = 0;i < 12; i++){
      //Display time with font, program execution time in second
      screen.fillRect(16,16,w-16*2,35, COLOR_RGB565_WHITE);
      screen.setTextWrap(false);
```

```
 //0x30FF is the color data in the format of RGB565
screen.setTextColor(0x30FF);
screen.setTextSize(3);
screen.setCursor(30, 30);
screen.println("Time:");

 //0x00FF is color data in RGB565 format
screen.setTextColor(0x00FF);
screen.setTextSize(3);
screen.setCursor(120, 30);
a = millis()/1000;
screen.println(a, 1);
/**
 * @brief Fill a rounded rectangle
 * @param x The x-coordinates of the starting point of the rectangle
 * @param y The y-coordinates of the starting point of the rectangle
 * @param w Rectangle width
 * @param h Rectangle height
 * @param r Fillet radius
 * @param color Rectangle color
 */
 screen.fillRoundRect(w/2-48-12, h/2-16-8, 32*3+12*2, 32+8*2, 20, 0x0000);
/**
 * @brief Draw horizontal and vertical lines to draw frame
 * @param x The x-coordinate of the starting point of the line
 * @param y The y-coordinate of the starting point of the line
 * @param h Line length
 * @param color Line color
 */
for(int x = 0; x<16 ;x++)
screen.drawFastVLine(/*x=*/x,/*y=*/0,/*h=*/h,/*color=*/color);
for(int y = 0; y<16 ;y++)
screen.drawFastHLine(/*x=*/16,/*y=*/y,/*w=*/w-16*2,/*color=*/color);
for(int x = w-1; x>=w-16 ;x--)
screen.drawFastVLine(x,0,h, color);
for(int y = h-1; y>=h-16 ;y--)
screen.drawFastHLine(16,y,w-16*2,color);
/**
```

```
       * @brief Draw monochrome pictures with different drawing orders
       * @param x 0 Set the starting point to be at the upper left of the screen, near the left side
       * @param y 0 Near the upper side
       * @param bitmap gImage_XBitmap The array in the header file XBitmap, the array elements are single bytes
       * @param w 240 Picture width

       * @param h 240 Picture height
       * @param color Set color
       */
      screen.drawXBitmap(/*x=*/w/2-48,/*y=*/h/2-16,/*bitmap gImage_Bitmap=*/gImage[i],/*w=*/32,/*h=*/32,color+=0x0700);
      //Delay 1 second
      delay(1000);

      screen.drawXBitmap(/*x=*/w/2-16,/*y=*/h/2-16,/*bitmap gImage_Bitmap=*/gImage[i+1],/*w=*/32,/*h=*/32,color+=0x0700);
      delay(1000);

      screen.drawXBitmap(/*x=*/w/2+16,/*y=*/h/2-16,/*bitmap gImage_Bitmap=*/gImage[i+2],/*w=*/32,/*h=*/32,color+=0x0700);
      delay(1000);

    }
  }


}
```

Result 2

Sample Code 3-UI control

This example presents you how to load three different controls.

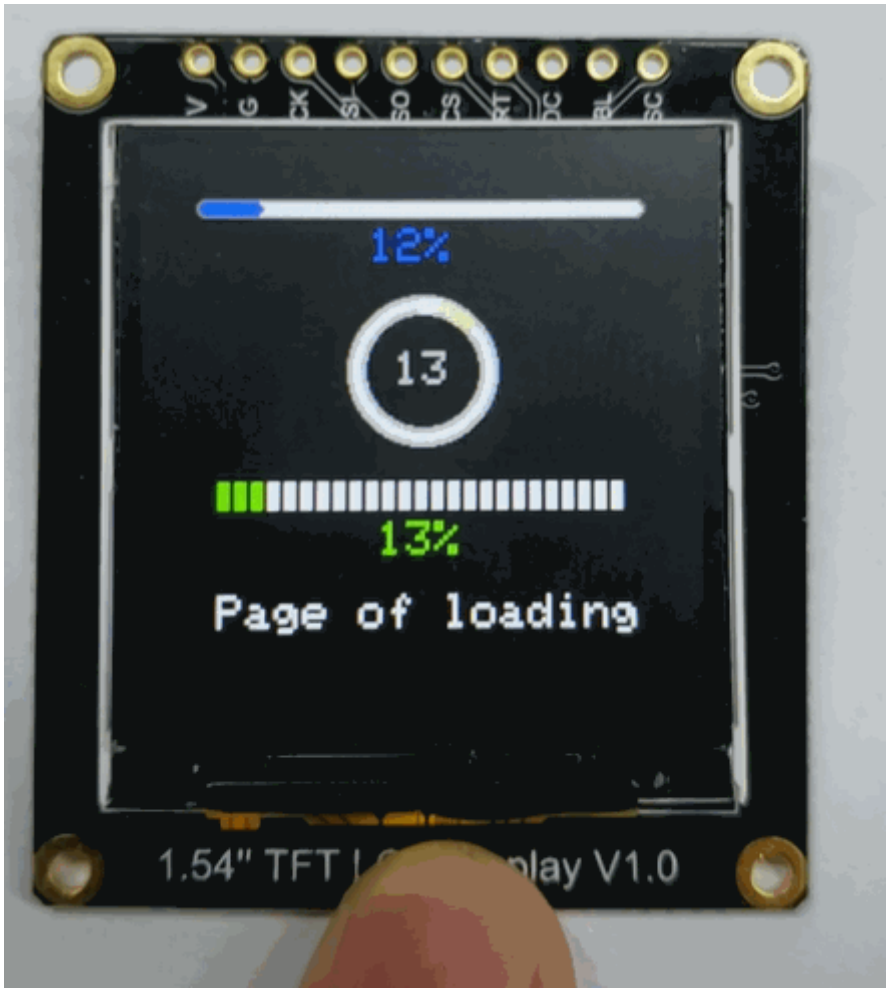
```
v/*!
 * @file UI_bar.ino
 * @brief Create a progress bar control on the screen.
 * @n Users can customize the parameters of the progress bar or use the default parameters.
 * @n Users can control the value of the progress bar through the callback function of the progress bar.
 * @n The example supports Arduino Uno, Leonardo, Mega2560, FireBeetle-ESP32, FireBeetle-ESP8266, FireBeetle-M0.
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @licence     The MIT License (MIT)
 * @author [fengli](li.feng@dfrobot.com)
 * @version  V1.0
 * @date  2019-12-6
 * @get from https://www.dfrobot.com
 * @url https://github.com/DFRobot/DFRobot_GDL/src/DFRpbot_UI
*/


#include "DFRobot_UI.h"
 #include "DFRobot_GDL.h"

/*M0*/
#if defined ARDUINO_SAM_ZERO
#define TFT_DC  7
#define TFT_CS  5
#define TFT_RST 6
/*ESP32 and ESP8266*/
#elif defined(ESP32) || defined(ESP8266)
#define TFT_DC  D3
#define TFT_CS  D4
#define TFT_RST D5
/*AVR series mainboard*/
#else
#define TFT_DC  2
```

```
#define TFT_DC  2
#define TFT_CS  3
#define TFT_RST 4
#endif


/**
 * @brief Constructor  Constructors for hardware SPI communication
 * @param dc  Command pin or data line pin of SPI communication
 * @param cs  Chip select pin for SPI communication
 * @param rst Reset pin of the screen
 * @param bl  Screen backlight pin
 */
DFRobot_ST7789_240x240_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ST7789_240x320_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9341_240x320_HW_SPI  screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9488_320x480_HW_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
/* M0 mainboard DMA transfer */
//DFRobot_ST7789_240x240_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ST7789_240x320_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9341_240x320_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);
//DFRobot_ILI9488_320x480_DMA_SPI screen(/*dc=*/TFT_DC,/*cs=*/TFT_CS,/*rst=*/TFT_RST);


/**
 * @brief Construct a function
 * @param gdl Screen object
 * @param touch Touch object
 */
DFRobot_UI ui(&screen, NULL);


uint8_t value1 = 0;
uint8_t value2 = 0;
uint8_t value3 = 0;
//Callback function of progress bar1
void barCallback1(DFRobot_UI:: sBar_t &obj){
    //Enable the progress bar plus 1 in each time, it enters the callback function.
```

```
   delay(50);
    obj.setValue(value1);
    if(value1 < 100) value1++;
}
//Callback function of progress bar2

void barCallback2(DFRobot_UI:: sBar_t &obj){
    //Enable the progress bar plus 1 in each time, it enters the callback function.
   delay(50);
    delay(50);
    obj.setValue(value2);
    if(value2 < 100) value2++;

}
//Callback function of progress bar3
void barCallback3(DFRobot_UI:: sBar_t &obj){
    //Enable the progress bar plus 1 in each time, it enters the callback function.
   delay(50);
    delay(50);
    obj.setValue(value3);
    if(value3 < 100) value3++;
}
void setup()
{

  Serial.begin(9600);
  //Initialize UI
  ui.begin();
  ui.setTheme(DFRobot_UI::MODERN);

  //Display a string on the screen
  ui.drawString(/*x=*/33,/*y=*/screen.height()/5*4,"Page of loading",COLOR_RGB565_WHITE,ui.bgColor,/*fontsize =*/2,/*Invert=*/0);
  //Create a progress bar control
  DFRobot_UI::sBar_t &bar1 = ui.creatBar();
  /** User-defined progress bar parameter **/
  bar1.setStyle(DFRobot_UI::COLUMN);
  bar1.fgColor = COLOR_RGB565_GREEN;
  bar1.setCallback(barCallback1);
```

```
  ui.draw(&bar1,/*x=*/33,/*y=*/screen.height()/5*3);

  DFRobot_UI::sBar_t &bar2 = ui.creatBar();
  /**User-defined progress bar parameter*/
  bar2.setStyle(DFRobot_UI::CIRCULAR);

  bar2.setCallback(barCallback2);
  ui.draw(&bar2,/*x=*/120,/*y=*/screen.height()/5*2);

  DFRobot_UI::sBar_t &bar3 = ui.creatBar();
  /**User-defined progress bar parameter*/
  bar3.fgColor = COLOR_RGB565_BLUE;
  bar3.setStyle(DFRobot_UI::BAR);
  bar3.setCallback(barCallback3);
  ui.draw(&bar3,/*x=*/(screen.width()-bar3.width)/2,/*y=*/screen.height()/10);
}


void loop()
{
  //Refresh
  ui.refresh();
}
```

Result 3



Compatiblity Test

MCU	OK	Failed	Untest	Remark
FireBeetle-ESP32	√			

MCU	OK	Failed	Untest	Remark
FireBeetle-ESP8266	√			
Arduino Uno	√			
Leonardo	√			
Mega2560	√			
Arduino M0	√			

FAQ

Any questions to ask or interesting applications to share, please visist our forum (https://www.dfrobot.com/forum) or community (https://community.dfrobot.com/).

More Documents

- Schematics (https://github.com/DFRobot/Wiki/raw/master/DFR0649/240x240-LCD-Schematic.pdf)
- Dimension Diagram (https://github.com/DFRobot/Wiki/raw/master/DFR0649/DFR0649-Dimensions.pdf)
- Chip Datasheet (https://github.com/DFRobot/Wiki/raw/master/DFR0649/SPEC%20ZJY154T-PG04%20VER%20A.pdf)

Get **1.54" 240x240 IPS TFT LCD Display with MicroSD Card Breakout** (https://www.dfrobot.com/product-2072.html) from DFRobot Store or **DFRobot Distributor**. (https://www.dfrobot.com/index.php?route=information/distributorslogo)

Turn to the Top