BMduino-Shield
2.8" TFT-LCD Display

BMD58T280
Arduino Library V1.0.3 Description

Revision: V1.30 Date: March 19, 2024

www.bestmodulescorp.com

Contents

Introduction **3**
Arduino Lib Functions **3**
Arduino Lib Download and Installation **12**
Arduino Example **13**
Example 1: TFTBitmapScroll 13
Example 2: TFTColorPicker 16
Example 3: TFTDisplayText 18
Example 4: TFTEtchASketch 20
Example 5: TFTEtchASketchAddIcon 22
Example 6: TFTGraph 25
Example 7: TFTLogoVideo 27

Introduction

The Best Modules BMD58T280 is a shield board for 2.8” TFT-LCD display function, which uses the SPI and EBI communication methods. This document provides the description of the BMD58T280 Arduino Lib functions and how to install the Arduino Lib. The example demonstrates the function of TFT display.

Arduino Lib Functions

Arduino Lib Name: BMD58T280			Lib Version: V1.0.3
Class Name: BMD58T280			
Constructors & Initialisation			
1	BMD58T280()		
	Description	Constructor, configure the EBI interface	
	Parameter	—	
	Return Value	—	
	Note	Arduino UNO does not support this interface	
2	BMD58T280(SPIClass *spiClass)		
	Description	Constructor, configure the SPI interface	
	Parameter	*spiClass: SPI parameter	
	Return Value	—	
	Note	—	
3	void begin(uint32_t freq=0)		
	Description	Module initialisation, configure the SPI frequency	
	Parameter	freq: SPI frequency. When freq=0, the SPI frequency is f_{cpu}/2Hz (default).	
	Return Value	void	
	Note	—	
Performance Functions			
4	void image(ImageInf & img, uint16_t x, uint16_t y)		
	Description	Display the BMP image file which is stored in SD	
	Parameter	& img: file name x: displayed starting position x coordinate y: displayed starting position y coordinate	
	Return Value	void	
	Note	—	
5	void scrollTo(uint16_t y)		
	Description	Scroll image	
	Parameter	y: move y pixels to the left	
	Return Value	void	
	Note	Move the image horizontally to the left by y pixels along the y axis	

6	void drawPixels(int16_t x, int16_t y, int16_t w, int16_t h, const uint8_t *pColor)	
	Description	Draw multiple pixels pColor in the window within the starting position (x, y) and width range (w, h) settings
	Parameter	x: the starting x-coordinate of the pixel y: the starting y-coordinate of the pixel w: width h: height *pColor: pixel color arrays. A pixel color is composed of 2 bytes, which can be written up to w×h pixels
	Return Value	void
	Note	A pixel color: Bit 11~Bit 8: RED color parameter Bit 7~Bit 4: GREEN color parameter Bit 3~Bit 0: BLUE color parameter When the window is without color filled in, default no color
7	int16_t height(void)	
	Description	Get the screen height
	Parameter	void
	Return Value	Screen height
	Note	The shield board screen height can be 320 or240 depending on the rotation position
8	int16_t width(void)	
	Description	Get the screen width
	Parameter	void
	Return Value	Screen width
	Note	The shield board screen width can be 240 or 320 depending on the rotation position
9	uint8_t getRotation(void)	
	Description	Get the LCD screen coordinate clockwise rotational angle(2)
	Parameter	void
	Return Value	r: clockwise rotation angle 0: 0 degrees 1: 90 degrees 2: 180 degrees 3: 270 degrees
	Note	Refer to Note 2 for coordinate change details
10	void text(const char *text, int16_t x, int16_t y)	
	Description	Display the string at (x, y) position
	Parameter	*text: string x: x-coordinate y: y-coordinate
	Return Value	void
	Note	Character size: set by the setTextSize function Character color: set by the stroke function
11	void text(const char * text, int16_t x, int16_t y, uint16_t textcolor, uint16_t textbgcolor)	
	Description	Display the string at (x, y) position
	Parameter	*text: string x: x-coordinate y: y-coordinate textcolor: Character color(1) textbgcolor: Character background color(1)
	Return Value	void
	Note	The character color and character background color are set directly, independent of the stroke function and the background function

12	void textWrap(const char *text, int16_t x, int16_t y)	
	Description	Display the string at (x, y) position, line wrapping
	Parameter	*text: string x: x-coordinate y: y-coordinate
	Return Value	void
	Note	Character size: set by the setTextSize function Character color: set by the stroke function
13	void textWrap(const char * text, int16_t x, int16_t y, uint16_t textcolor, uint16_t textbgcolor)	
	Description	Displays the string at (x, y) position and wraps automatically
	Parameter	*text: string x: x-coordinate y: y-coordinate textcolor: Character color(1) textbgcolor: Character background color(1)
	Return Value	void
	Note	The character color and character background color are set directly, independent of the stroke function and the background function
14	void circle(int16_t x, int16_t y, int16_t r)	
	Description	Draw a circle
	Parameter	x: the center x coordinate of a circle y: the center y coordinate of a circle r: radius
	Return Value	void
	Note	Circle color: set by the stroke function Circle interior color: set by the fill function
15	void point(int16_t x, int16_t y)	
	Description	Draw a point
	Parameter	x: x-coordinate of the point y: y-coordinate of the point
	Return Value	—
	Note	Point color: set by the stroke function
16	void line(int16_t x1, int16_t y1, int16_t x2, int16_t y2)	
	Description	Draw a line
	Parameter	x1: the starting x-coordinate of the point y1: the starting y-coordinate of the point x2: the ending x-coordinate of the point y2: the ending y-coordinate of the point
	Return Value	void
	Note	Draw a line from (x1, y1) to (x2, y2) Line color: set by the stroke function
17	void rect(int16_t x, int16_t y, int16_t width, int16_t height)	
	Description	Draw a rectangle
	Parameter	x: the starting x-coordinate of the retangle y: the starting y-coordinate of the retangle width: width height: height
	Return Value	void
	Note	Rectangle color: set by the stroke function Rectangle interior color: set by the fill function

18	void rect(int16_t x, int16_t y, int16_t width, int16_t height, int16_t radius)	
	Description	Draw a rounded rectangle
	Parameter	x: the starting x-coordinate y: the starting y-coordinate width: width height: height radius: fillet radius
	Return Value	void
	Note	Rectangle color: set by the stroke function Rectangle interior color: set by the fill function
19	void triangle(int16_t x1, int16_t y1, int16_t x2, int16_t y2, int16_t x3, int16_t y3)	
	Description	Draw a triangle
	Parameter	x1: point 1 x-coordinate y1: point 1 y-coordinate x2: point 2 x-coordinate y2: point 2 y-coordinate x3: point 3 x-coordinate y3: point 3 y-coordinate
	Return Value	void
	Note	Triangle color: set by the stroke function Triangle interior color: set by the fill function
Parameter Configuration		
20	void setScrollMargins(uint16_t top, uint16_t bottom)	
	Description	Set the scroll margins
	Parameter	top: top bottom: bottom
	Return Value	void
	Note	—
21	void setAddrWindow(uint16_t x, uint16_t y, uint16_t w, uint16_t h)	
	Description	Set the window
	Parameter	x: the starting x-coordinate y: the starting y-coordinate w: width h: height
	Return Value	void
	Note	Select a specific range on the screen, to insert a BMP file or C-array Insert a BMP: image function Insert C-array: drawPixels function
22	void setRotation(uint8_t r)	
	Description	Set the LCD coordinate for clockwise rotation(2)
	Parameter	r: clockwise rotation angle 0: 0 degrees 1: 90 degrees 2: 180 degrees 3: 270 degrees
	Return Value	void
	Note	Refer to Note 2 for coordinate change details
23	void fill(uint8_t red, uint8_t green, uint8_t blue)	
	Description	Set the fill color, which can fill with circle, triangle and other closed graphics
	Parameter	red: R color green: G color blue: B color
	Return Value	void
	Note	RGB color is composed of R, G and B colors

24	void fill(uint16_t c)	
	Description	Set the fill color
	Parameter	c: color constant. Refer to the “Color Constants” table(1)
	Return Value	void
	Note	—
25	void noFill()	
	Description	Set no fill effect
	Parameter	—
	Return Value	void
	Note	Transparent
26	void stroke(uint8_t red, uint8_t green, uint8_t blue)	
	Description	Set the font and frame color
	Parameter	red: R color green: G color blue: B color
	Return Value	void
	Note	RGB color is composed of R, G and B colors
27	void stroke(uint16_t c)	
	Description	Set the font and frame color
	Parameter	c: color constant. Refer to the “Color Constants” table(1)
	Return Value	void
	Note	—
28	void noStroke()	
	Description	Set no font, no frame
	Parameter	—
	Return Value	void
	Note	—
29	void background(uint8_t red, uint8_t green, uint8_t blue)	
	Description	Set the background color
	Parameter	red: R color green: G color blue: B color
	Return Value	void
	Note	RGB color is composed of R, G and B colors
30	void background(uint16_t c)	
	Description	Set the background color
	Parameter	c: color constant (see table of color constants(1))
	Return Value	void
	Note	—
31	void setTextSize(uint8_t s)	
	Description	Set the font size
	Parameter	s: font size, range from 1 to 255. The default value is 1
	Return Value	void
	Note	—
Class Name: BmpImage		
Constructors & Initialisation		

1	BmpImage()	
	Description	BMP file, constructor
	Parameter	—
	Return Value	—
	Note	—
Performance Functions		
2	operator bool()	
	Description	Judge whether the file exists
	Parameter	—
	Return Value	Existence condition: true: file existence false: file inexistence
	Note	—
3	int width()	
	Description	Obtain the image width
	Parameter	—
	Return Value	Picture width
	Note	—
4	int height()	
	Description	Obtain the image height
	Parameter	—
	Return Value	Picture height
	Note	—
Parameter Configuration		
5	bool loadImage(const char *fileName, HardwareSerial *pSerial)	
	Description	Load the BMP file from SD
	Parameter	*fileName: BMP *pSerial: Default NULL. The SD connection information will not be displayed when the pSerial is NULL
	Return Value	Whether the file exists: true: file existence false: file inexistence
	Note	—
Class Name: BM_XPT2046		
Constructors & Initialisation		
1	BM_XPT2046(uint8_t cspin cspin=18, uint8_t tirq=19)	
	Description	Constructor, configure CS and TP IRQ pins
	Parameter	cspin: CS pin tirq: TP IRQ pin
	Return Value	—
	Note	—
2	void begin(SPIClass &wspi)	
	Description	Touch IC initialisation
	Parameter	&wspi: configure the SPI interface
	Return Value	void
	Note	—
Performance Functions		

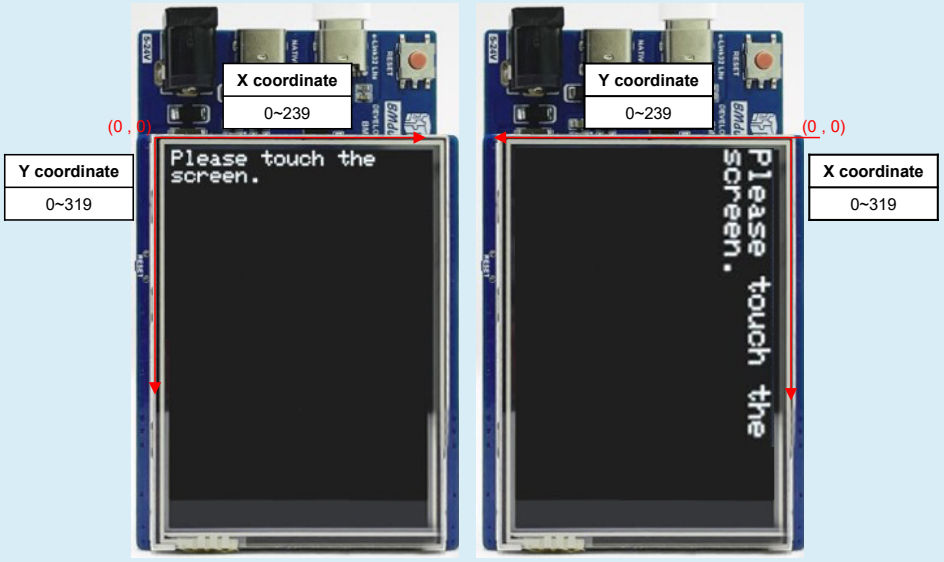
3	bool operator==(TS_Point p)	
	Description	Judge whether the coordinates and finger pressure points are the same
	Parameter	—
	Return Value	Execution result: false: Same true: Different
	Note	—
4	bool operator!=(TS_Point p)	
	Description	Judge whether the coordinates and finger pressure points are different
	Parameter	—
	Return Value	Execution result: false: Different true: Same
	Note	—
5	TS_Point getPoint()	
	Description	Return finger pressure point coordinates
	Parameter	—
	Return Value	_xraw: x-coordinate _yraw: y-coordinate _zraw: z-coordinate
	Note	The returned value is stored in the TS_Point class. Call the class TS_Point (int16_tx, int16_ty, int16_tz) to get the pressure point coordinates. Refer to the example for details.
6	bool touched()	
	Description	Judge whether the LCD is touched
	Parameter	—
	Return Value	Whether the LCD is touched: true: LCD is touched flase: LCD is not touched
	Note	—
Parameter Configuration		
7	void setRotation(uint8_t n)	
	Description	Set the touch screen coordinate for clockwise rotation(2)
	Parameter	n: clockwise rotation angle 0: 0 degrees 1: 90 degrees 2: 180 degrees 3: 270 degrees
	Return Value	void
	Note	
Class 名称：SDReadRawData		
Constructors & Initialisation		
1	SDReadRawData(void)	
	Description	Create structure function
	Parameter	—
	Return Value	—
	Note	Arduino UNO does not support

2	uint8_t init(uint8_t chipSelectPin)	
	Description	SD card initialisation
	Parameter	chipSelectPin: SPI CS pin
	Return Value	Execution result: true: Succeeded false: Failed
	Note	Arduino UNO does not support
Performance Functions		
3	uint8_t readBlock(uint32_t block, uint16_t count, uint8_t *dst)	
	Description	Read the SD block data
	Parameter	block: block data index count: data length *dst: destination
	Return Value	—
	Note	Arduino UNO does not support
4	void waitDmaFinish()	
	Description	Wait for the data reading complete
	Parameter	—
	Return Value	void
	Note	Arduino UNO does not support

Note 1: Color Constants Table

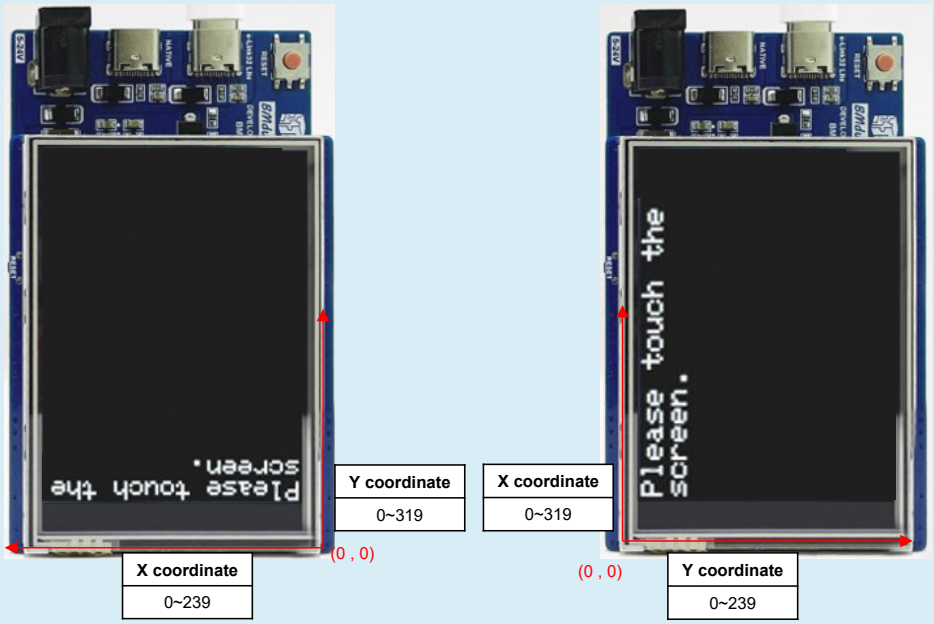
BM_ILI9341::NAVY
BM_ILI9341::BLACK
BM_ILI9341::DARKGREEN
BM_ILI9341::DARKCYAN
BM_ILI9341::MAROON
BM_ILI9341::PURPLE
BM_ILI9341::OLIVE
BM_ILI9341::LIGHTGREY
BM_ILI9341::DARKGREY
BM_ILI9341::BLUE
BM_ILI9341::GREEN
BM_ILI9341::CYAN
BM_ILI9341::RED
BM_ILI9341::MAGENTA
BM_ILI9341::YELLOW
BM_ILI9341::WHITE
BM_ILI9341::ORANGE
BM_ILI9341::GREENYELLOW
BM_ILI9341::PINK

Note 2: The TFT display uses x-axis and y-axis coordinates



A. Screen Coordinate without Rotation

B. Coordinates Rotated Clockwise by 90 Degrees



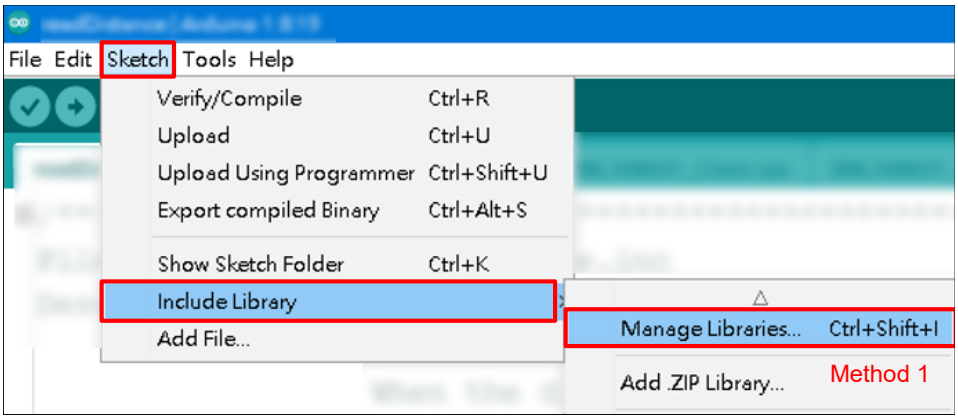
C. Coordinates Rotated Clockwise by 180 Degrees D. Coordinates Rotated Clockwise by 270 Degrees

Arduino Lib Download and Installation

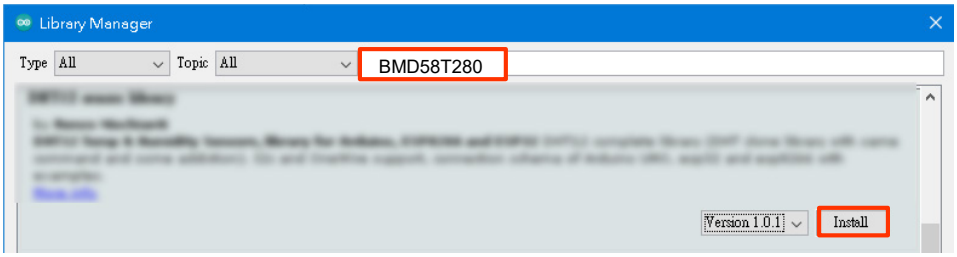
BMD58T280 Library: Refer to the following two methods to install the BMD58T280 Arduino Library.

Method 1: Search for installation

Arduino IDE→Sketch→Include Library→Manage Libraries...→Search BMD58T280→Install



Search for Installation Step 1

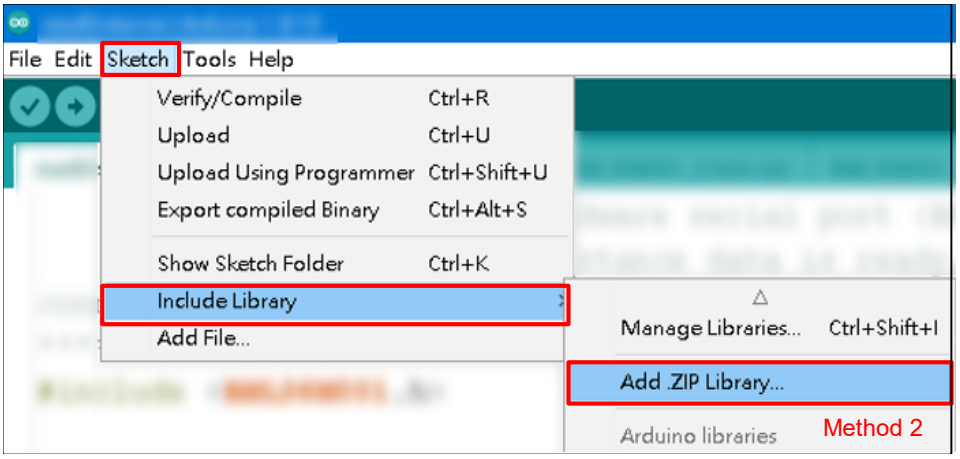


Search for Installation Step 2

Method 2: Download the .ZIP library before adding it

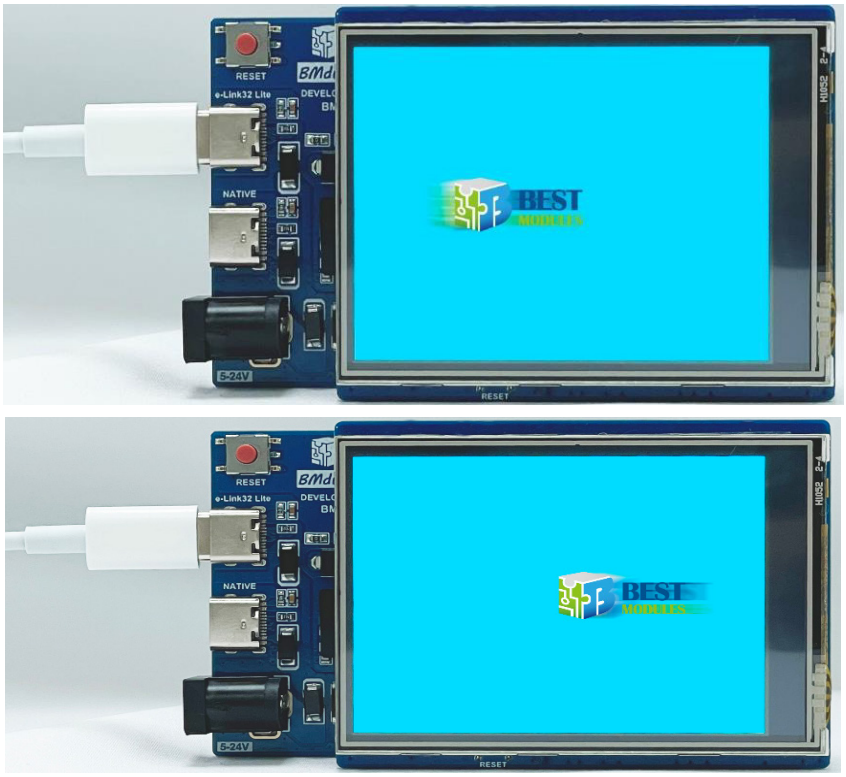
Download the Arduino example (BMD28T280 Library) under the DOCUMENTS menu from the Best Modules website (https://www.bestmodulescorp.com/bmd58t280.html).

Add .ZIP library: Arduino IDE→Sketch→Include Library→Add .ZIP Library...



Arduino Example

Example 1: TFTBitmapScroll



Physical Connection Diagram

Function: Display the bmp file which is stored in SD card on the LCD, move the picture horizontally through the scrolling effect to show the background, print, refresh the BMP file and scrollTo functions.

1. Open the example:

   Arduino IDE→File→Examples→Select Lib→Select example (TFTBitmapScroll)

2. Environment Setup:

   - Prepare the micro SD card and set the file system format to be FAT32.
   - Copy the “Logo.bmp” from the TFTBitmapScroll example folder to the SD card through PC.
   - Insert SD into the BMD58T280.

3. Example Description:

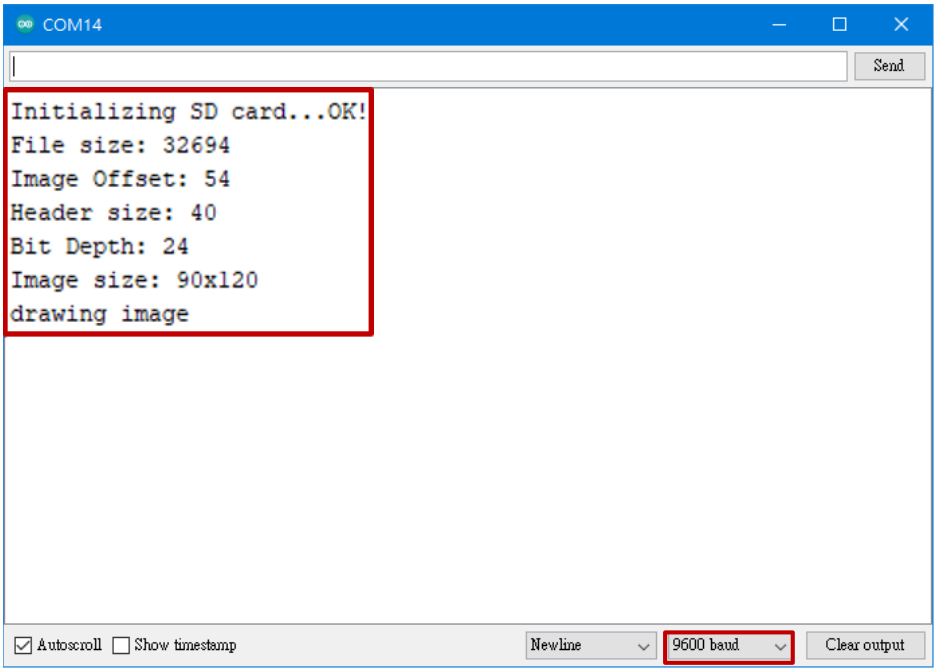
a. Create object & initialise object

```
#include <SD.h>
#include <BMD58T280.h>
#include <SPI.h>
// Select the communication interface
#if !defined(ARDUINO_AVR_UNO)  // BMduino
BMD58T280 TFTscreen;           // Create an LCD object
#else                          // Arduino
BMD58T280 TFTscreen(&SPI);     // Create an LCD object
#endif
#define sd_cs  10              // SD pin definition
BmpImage image1; // This variable represents the image to be drawn on
                 // the screen
void setup()
{
  // Initialise LCD and display the message
  TFTscreen.begin();
  TFTscreen.background(255, 255, 255);
  TFTscreen.setTextSize(1);
  TFTscreen.stroke(0, 0, 255);
  TFTscreen.println();
  TFTscreen.println(F("BMduino TFT Bitmap Example And Scroll"));
  TFTscreen.stroke(0, 0, 0);
  TFTscreen.println(F("Open serial monitor"));
  TFTscreen.println(F("to run the sketch"));
  delay(3000);
  Serial.begin(9600);          // Serial port initialisation
  while (!Serial)
  {
    // Wait for the serial port connection. Only the native USB port
    // is required
  }
  TFTscreen.background(0, 0xd8, 0xFF); // Clear the GLCD screen before
  // starting Attempt to access the SD card. If fails, for example, no
  // card exists, the setup process will stop.
  Serial.print(F("Initializing SD card..."));
  if (!SD.begin(sd_cs))
  {
  Serial.println(F("failed!"));
  return;
  }
  Serial.println(F("OK!"));
  // The SD card can be accessed, attempt to load the image file
  image1.loadImage("Logo.bmp", &Serial);
  if (!image1)
  {
    Serial.println(F("error while loading Logo.bmp"));
  }
  Serial.println(F("drawing image"));
  TFTscreen.image(image1, (TFTWIDTH - image1.width())/2, 0);// Draw
  // the image to the screen, the image position is at the center of
  // the x-axis
}
```

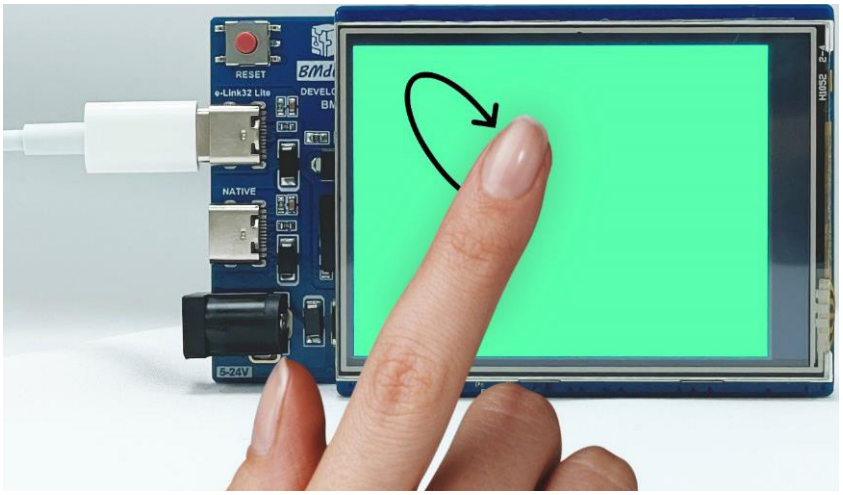
b.loop

```
void loop()
{
  // The image moves from left to right
  for(int i=TFTscreen.height();i>image1.height();i--)
  {
    TFTscreen.scrollTo(i);
    delay(4);
  }
  // The image moves from right to left
  for(int i=image1.height();i<TFTscreen.height();i++)
  {
    TFTscreen.scrollTo(i);
    delay(4);
  }
}
```

4. Open the serial monitor and set the baud rate to be 9600. The serial monitor will display as follows.



Example 2: TFTColorPicker



Physical Connection Diagram

Function: Display the touch screen function by the touched x, y, and z values as the LCD RGB.

1. Open the example:

   Arduino IDE→File→Examples→Select Lib→Select example (TFTColorPicker)

2. Interactive method: Slide the screen arbitrarily, the screen color changes constantly.

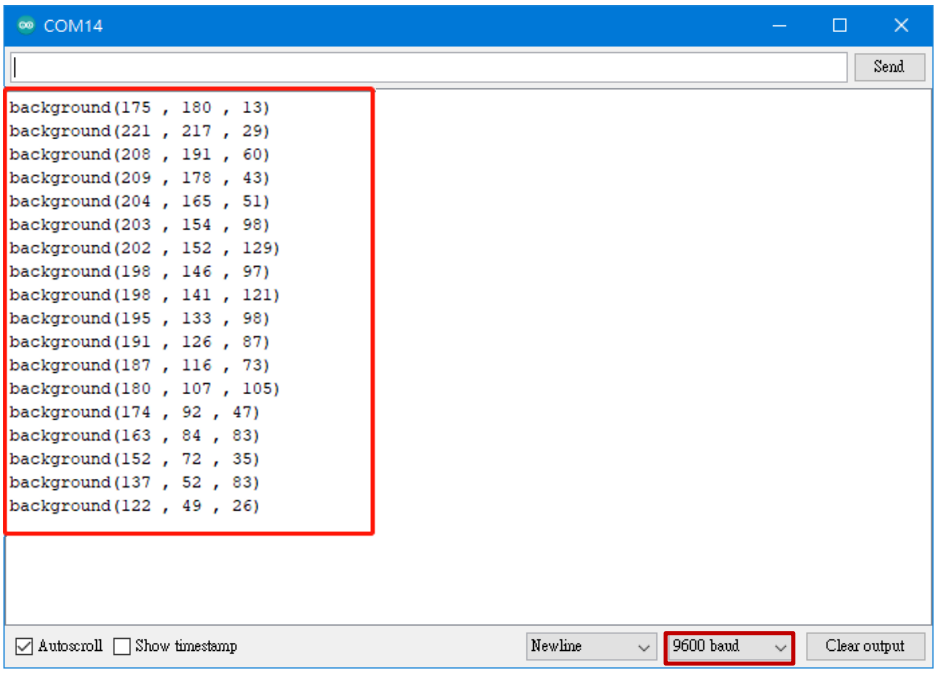
3. Example Description:

   a. Create object & initialise object

```
#include <BMD58T280.h>
#include <SPI.h>
// Select the communication interface
#if !defined(ARDUINO_AVR_UNO) // For BMduino
BMD58T280 TFTscreen;          // Create an LCD object
#else                         // For Arduino UNO
BMD58T280 TFTscreen(&SPI);    // Create an LCD object
#endif
BM_XPT2046 touchScreen;       // Create a touch screen object
void setup()
{
  // Serial port initialisation
  Serial.begin(9600);
  // Initialisation
  TFTscreen.begin();
  touchScreen.begin();
  // Set the background to white
  TFTscreen.background(255, 255, 255);
}
void loop()
{
  if (touchScreen.touched())
  {
    // Touch the screen, read the touch screen (x, y, z)
    BM_XPT2046::TS_Point p = touchScreen.getPoint();
    p.x = map(p.x, 400, 3600, 0, 255);   if(p.x<0) p.x = 0;
    p.y = map(p.y, 400, 3600, 0, 255);   if(p.y<0) p.y = 0;
```

```
    p.z = map(p.z, 400, 2800, 0, 255);
    // Draw the background according to the map value
    TFTscreen.background(p.x, p.y, p.z);
    // Send the value to the serial monitor
    Serial.print("background(");
    Serial.print(p.x);
    Serial.print(" , ");
    Serial.print(p.y);
    Serial.print(" , ");
    Serial.print(p.z);
    Serial.println(")");
  }
  delay(20);
}
```

4. Open the serial monitor and set the baud rate to be 9600. The serial monitor will display as follows.



Example 3: TFTDisplayText

Physical Connection Diagram

Function: The LCD displays the x and y coordinates of the touch screen, and displays the functions of the touch screen with text and setTextSize.

1. Open the example:

   Arduino IDE→File→Examples→Select Lib→Select example (TFTDisplayText)

2. Interactive method: Slide the screen arbitrarily and observe the change of the coordinates.

3. Example Description:

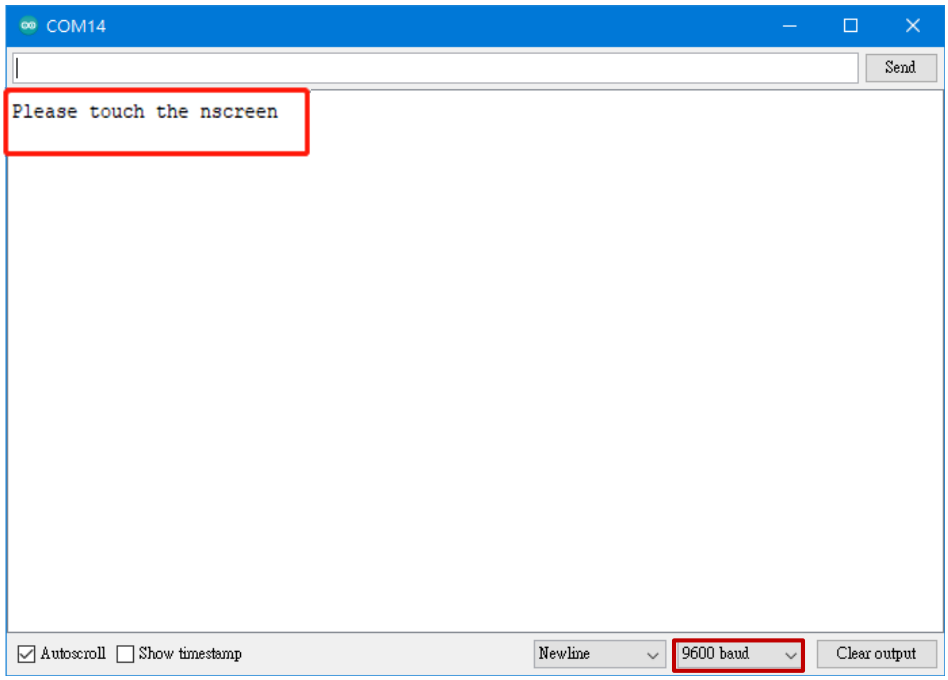
   a. Create object & initialise object

```
#include <BMD58T280.h>
#include <SPI.h>
// Select the communication interface
#if !defined(ARDUINO_AVR_UNO) // For BMduino
BMD58T280 TFTscreen;          // Create an LCD object
#else                         // For Arduino UNO
BMD58T280 TFTscreen(&SPI);    // Create an LCD object
#endif
BM_XPT2046 touchScreen;       // Create a touch screen object
char AxisPrintout[20]; // Character array to be printed to the screen
void setup()
{
  TFTscreen.begin();
  touchScreen.begin();              // Touch screen initialisation
  Serial.begin(9600);
  TFTscreen.background(0, 0, 0);    // Clear screen
  // Write static text to the screen
  TFTscreen.stroke(255, 255, 255); // Set the font color to white
  TFTscreen.setTextSize(2);         // Set font size
  TFTscreen.text("Please touch the\nscreen.", 0, 0); // Write the text
  // to the top left corner of the screen
  TFTscreen.setTextSize(5);         // Set the large font for loop
  Serial.println("Please touch the nscreen"); // Serial port output
}
```

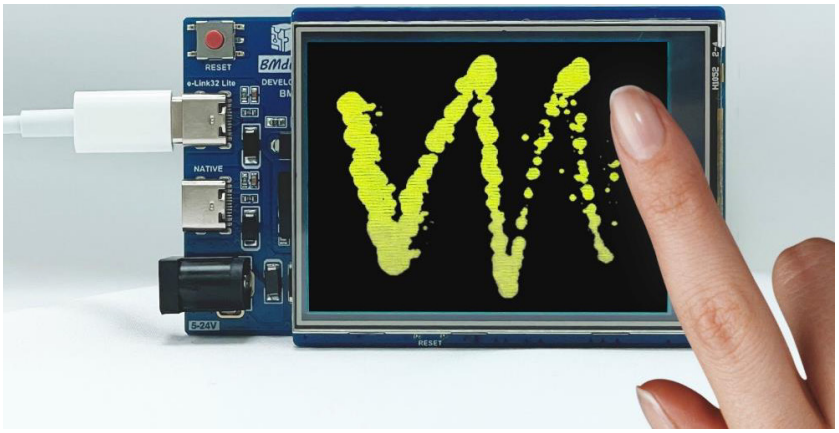
b. loop

```
void loop()
{
  if (touchScreen.touched())
  {
    // Touch the screen, read x and y coordinates
    BM_XPT2046::TS_Point p = touchScreen.getPoint();
    p.x = map(p.x, 400, 3600, 0, TFTscreen.width()); if(p.x<0) p.x = 0;
    p.y = map(p.y, 400, 3600, 0, TFTscreen.height());if(p.y<0) p.y = 0;
    String strXaxis = String(p.x);
    String strYaxis = String(p.y);
    String strResult = strXaxis + "," + strYaxis+ "    ";
    strResult.toCharArray(AxisPrintout, 20);// Convert value to
                                            // character array
    TFTscreen.text(AxisPrintout, 0, 40 , BM_ILI9341::WHITE,
    BM_ILI9341::BLACK);     // Print the sensor value
    delay(250);             // Wait a while
  }
  else
  {
    TFTscreen.text("                    ", 0, 40 , BM_ILI9341::WHITE,
    BM_ILI9341::BLACK);  // Erase text
  }
}
```

4. Open the serial monitor and set the baud rate to be 9600. The serial monitor will display as follows.



Example 4: TFTEtchASketch



Physical Connection Diagram

Function: Implement the drawing board function and display the functions of touch screen with fill and circle.

1. Open the example:

   Arduino IDE→File→Examples→Select Lib→Select example (TFTEtchASketch)

2. Interactive method: Draw pictures on the screen

3. Example Description:

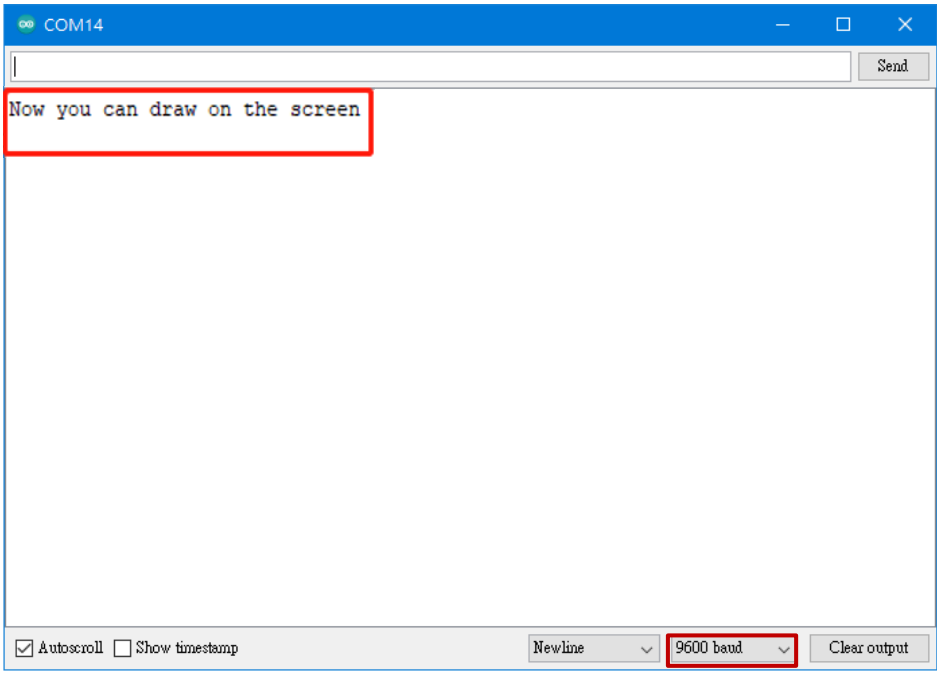
   a. Create object & initialise object

```
#include <BMD58T280.h>
#include <SPI.h>
// Select the communication interface
#if !defined(ARDUINO_AVR_UNO) // For BMduino
BMD58T280 TFTscreen;
#else                         // For UNO
BMD58T280 TFTscreen(&SPI);    // Create an LCD object
#endif
BM_XPT2046 touchScreen;       // Create an LCD object
void setup()
{
  // Screen initialisation
  TFTscreen.begin();
  touchScreen.begin();
  Serial.begin(9600);
  TFTscreen.background(0, 0, 0);      // Set the background to black
  TFTscreen.fill(BM_ILI9341::YELLOW); // Set the fill to yellow
  Serial.println("Now you can draw on the screen");   // Serial port
                                                      // output
}
```

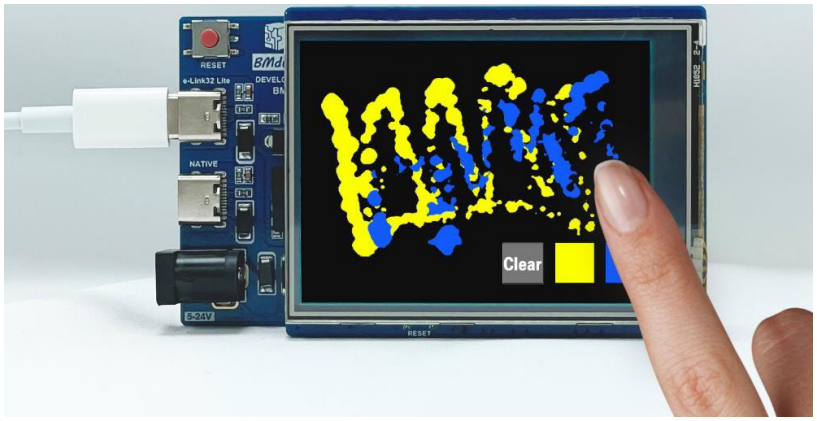
b.loop

```
void loop()
{
  if (touchScreen.touched())
  {
    BM_XPT2046::TS_Point p = touchScreen.getPoint();  // Touch the
    // screen, read the touch screen (x, y, z)
    p.x = map(p.x, 400, 3600, 0, TFTscreen.width());
    p.y = map(p.y, 400, 3600, 0, TFTscreen.height());
    p.z = map(p.z, 400, 2800, 0, 20);
    TFTscreen.circle(p.x, p.y, p.z);  // Draw a circle
  }
}
```

4. Open the serial monitor and set the baud rate to be 9600. The serial monitor will display as follows.



Example 5: TFTEtchASketchAddIcon



Physical Connection Diagram

Function: Add color selection and clear screen, display the touch screen with fill, circle, rect, drawPixels functions based on TFTEtchASketch.

1. Open the example:

   Arduino IDE→File→Examples→Select Lib→Select example (TFTEtchASketchAddIcon)

2. Interactive method: Draw pictures on the screen

3. Example Description:

   a. Create object & initialise object

```
#include <BMD58T280.h>
#include "icon.h"            // The raw data of the "Clear" icon
#include <SPI.h>
// Select the communication interface
#if !defined(ARDUINO_AVR_UNO) // For BMduino
BMD58T280 TFTscreen;         // Create an LCD object
#else                        // For UNO
BMD58T280 TFTscreen(&SPI);   // Create an LCD object
#endif
BM_XPT2046 touchScreen;      // Create a touch screen object
#define X_SATART         12  // Icon x-axis start position
#define Y_SATART         195 // Icon y-axis start position
#define ICON_W           35  // Icon width
#define ICON_H           36  // Icon length
#define ICON_OFFSET      10  // The distance of each icon
uint16_t color = (uint16_t)BM_ILI9341::YELLOW;
void setup()
{
  // Screen initialisation
  TFTscreen.begin();
  touchScreen.begin();
  Serial.begin(9600);
  TFTscreen.background(0, 0, 0);      // Set the background to black
  TFTscreen.fill(BM_ILI9341::YELLOW); // Set the fill to yellow
  TFTscreen.rect(X_SATART, Y_SATART + ICON_W + ICON_OFFSET, ICON_
                 W,ICON_H)            // Draw a yellow icon
  TFTscreen.fill(BM_ILI9341::BLUE);   // Set the fill to blue
  TFTscreen.rect(X_SATART, Y_SATART + ICON_W + ICON_OFFSET + ICON_W +
                 ICON_OFFSET,ICON_W,ICON_H);  // Draw a yellow icon
```

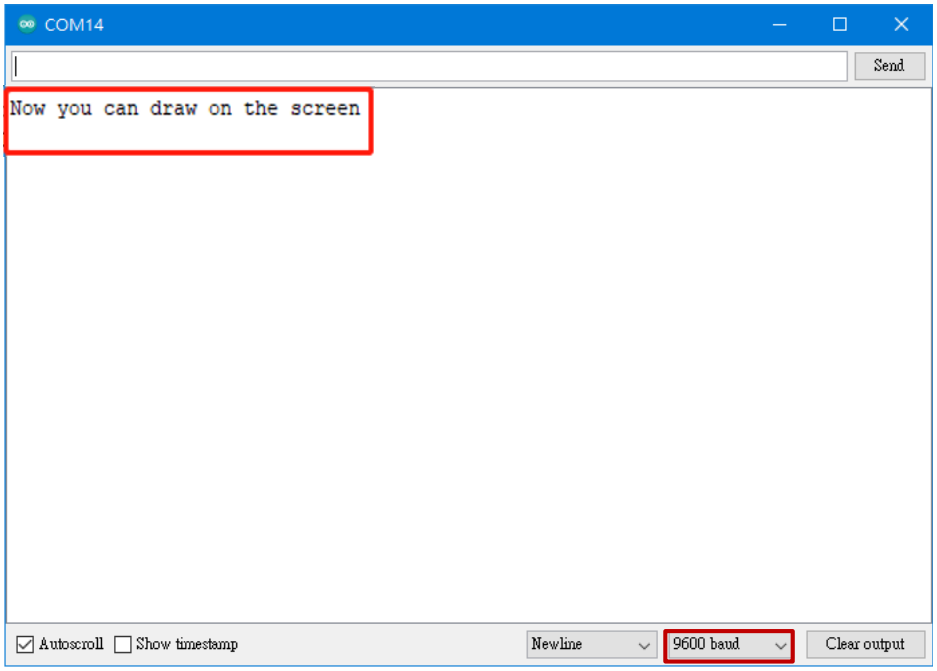
```
  TFTscreen.drawPixels(X_SATART, Y_SATART, ICON_W, ICON_H,
                       clearIcon);    // Draw a "Clear" icon
  Serial.println("Now you can draw on the screen");  // Serial port
                                                     // output
}
```

b.loop

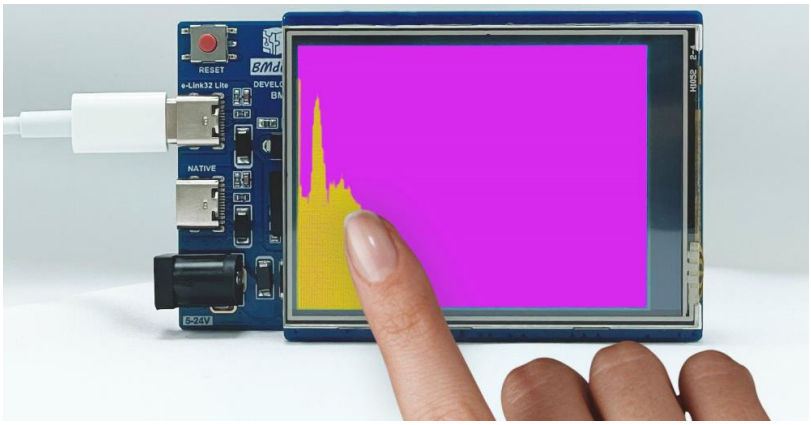
```
void loop()
{
  uint16_t isTouchIcon;
  if (touchScreen.touched())
  {
    BM_XPT2046::TS_Point p = touchScreen.getPoint(); // Touch the
    // screen, read the touch screen (x, y, z)
    p.x = map(p.x, 400, 3600, 0, TFTWIDTH);
    p.y = map(p.y, 400, 3600, 0, TFTHEIGHT);
    p.z = map(p.z, 400, 2800, 0, 20);
    isTouchIcon = 0;
    if(p.x>X_SATART & p.x<(ICON_H + X_SATART))
    {
      if((p.y>Y_SATART &&
      p.y <(Y_SATART +ICON_W) ))
      {
        // Users touch the "Clear" icon. Need to reset the screen
        // arrangement
        isTouchIcon = 1;
        TFTscreen.background(0, 0, 0); // Set the background to black
        TFTscreen.fill(BM_ILI9341::YELLOW);// Set the fill to yellow
        TFTscreen.rect(X_SATART,Y_SATART + ICON_W + ICON_OFFSET,ICON_
                       W,ICON_H);          // Draw a yellow icon
        TFTscreen.fill(BM_ILI9341::BLUE);  // Set the fill to blue
        TFTscreen.rect(X_SATART,Y_SATART+ICON_W+ICON_OFFSET+ICON_W+
                       ICON_OFFSET,ICON_W,ICON_H);// Draw a yellow
                                                  // icon
        TFTscreen.drawPixels(X_SATART,Y_SATART,ICON_W,ICON_H,
                             clearIcon);  // Draw a "Clear" icon
        TFTscreen.fill(color);            // Set brush color
      }
      else if((p.y>(Y_SATART + ICON_W + ICON_OFFSET) &&
      p.y<(Y_SATART + ICON_W + ICON_OFFSET + ICON_W) ))
      {
        // Users touch the 'YELLOW' icon. Redraw the yellow icon
        isTouchIcon = 1;
        TFTscreen.fill(BM_ILI9341::YELLOW);// Set the fill to yellow
        TFTscreen.rect(X_SATART,Y_SATART + ICON_W + ICON_OFFSET,ICON_
                       W,ICON_H);          // Draw a yellow icon
        color = (uint16_t)BM_ILI9341::YELLOW;
        TFTscreen.fill(color); // Set brush color
      }
      else if((p.y>(Y_SATART + ICON_W + ICON_OFFSET + ICON_W + ICON_
              OFFSET) &&
      p.y<(Y_SATART + ICON_W + ICON_OFFSET + ICON_W + ICON_OFFSET
           +ICON_W) ))
      {
        // Users touch the 'BLUE' icon. Redraw the blue icon
        isTouchIcon = 1;
        TFTscreen.fill(BM_ILI9341::BLUE);  // Set the fill to blue
```

```
        TFTscreen.rect(X_SATART,Y_SATART+ICON_W+ICON_OFFSET+ICON_W+
                       ICON_OFFSET,ICON_W,ICON_H); // Draw a blue icon
        color = (uint16_t)BM_ILI9341::BLUE;
        TFTscreen.fill(color);          // Set brush color
      }
    }
    if(isTouchIcon == 0)
    {
      TFTscreen.circle(p.x, p.y, p.z); // Without touching icon, start
                                       // drawing the circle
    }
  }
}
```

4. Open the serial monitor and set the baud rate to be 9600. The serial monitor will display as follows.



Example 6: TFTGraph



Physical Connection Diagram

Function: Change graph of the X-axis, display the functions of touch screen with stroke, line, and setRotation.

1. Open the example:

   Arduino IDE→File→Examples→Select Lib→Select example (TFTGraph)

2. Interactive method: Slide up and down the screen and observe the curve change.

3. Example Description:

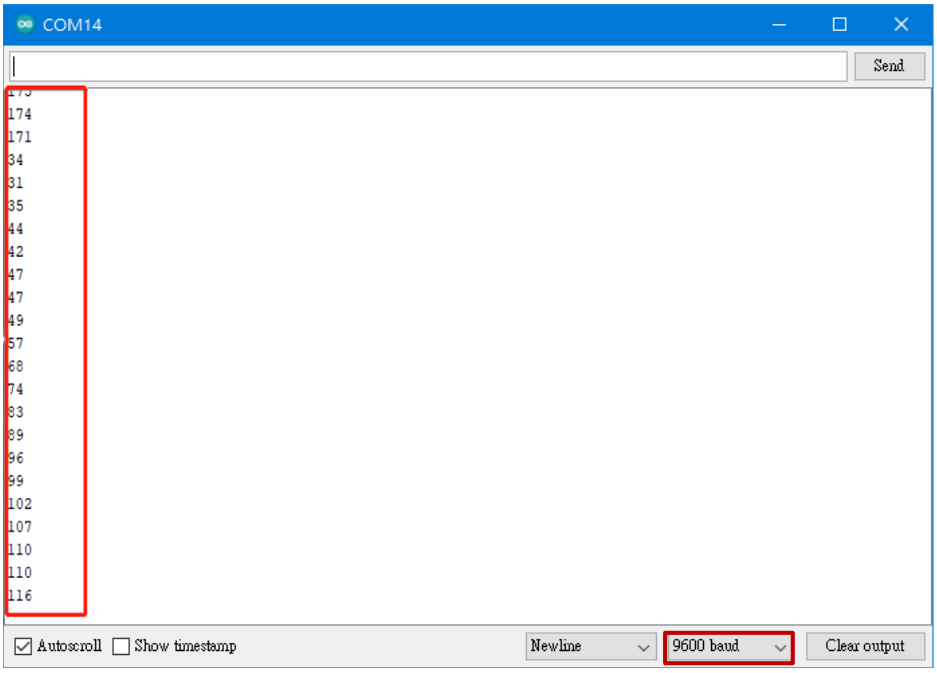
   a. Create object & initialise object

```
#include <BMD58T280.h>
#include <SPI.h>
// Select the communication interface
#if !defined(ARDUINO_AVR_UNO) // For BMduino
BMD58T280 TFTscreen;          // Create an LCD object
#else                         // For UNO
BMD58T280 TFTscreen(&SPI);    // Create an LCD object
#endif
BM_XPT2046 touchScreen;       // Create a touch screen object
int xPos = 0;                 // Line position
void setup()
{
  Serial.begin(9600);         // Serial port initialisation
  // Display initialisation
  TFTscreen.begin();
  touchScreen.begin();
  TFTscreen.setRotation(1);
  // Clear the screen with a specific color
  TFTscreen.background(250, 16, 200);
}
```

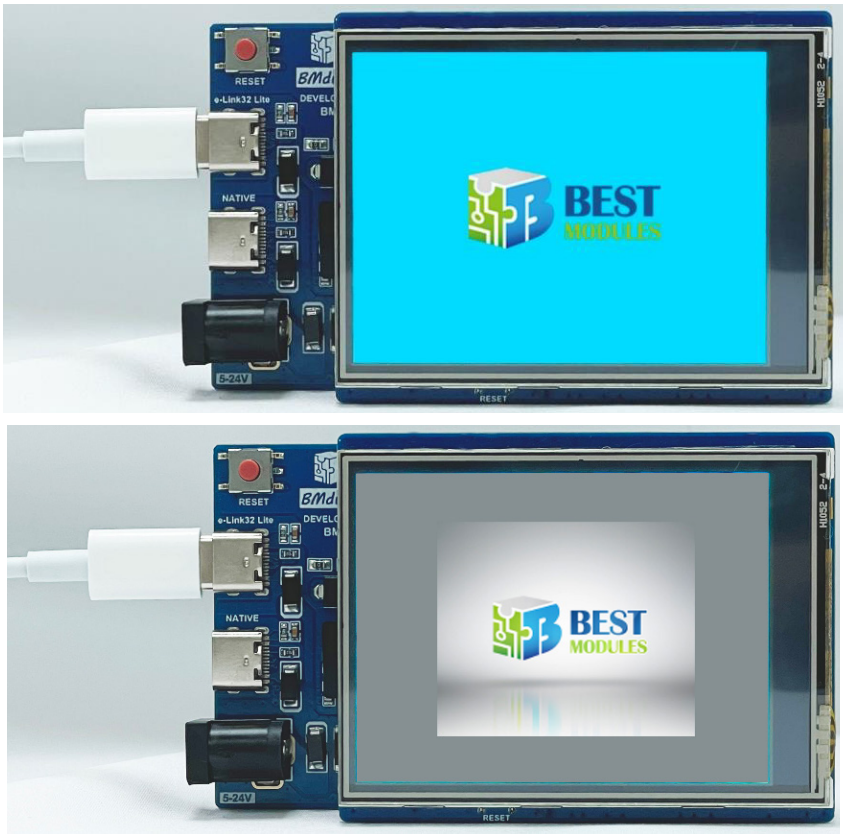
b. loop

```
void loop()
{
  // Read the transferred data and map it to the screen height
  if (touchScreen.touched())
  {
    // Touch the screen, read the x coordinate
    BM_XPT2046::TS_Point p = touchScreen.getPoint();
    int drawHeight = map(p.x, 400, 3600, 0, TFTscreen.height());
    Serial.println(drawHeight);  // Print the height to the serial
                                 // monitor
    // Draw a line
    TFTscreen.stroke(250, 180, 10);
    TFTscreen.line(xPos, TFTscreen.height() - drawHeight, xPos,
                   TFTscreen.height());
    // If the graph has reached the screen edge, erase the screen and
    // start again
    if (xPos >= 360)
    {
      xPos = 0;
      TFTscreen.background(250, 16, 200);
    }
    else
    {
      xPos++;                    // Increase horizontal position
      delay(16);
    }
  }
}
```

4. Open the serial monitor and set the baud rate to be 9600. The serial monitor will display as follows.



Example 7: TFTLogoVideo



Physical Connection Diagram

Function: Read the raw data in the SD card and through the bottom of the control to achieve the rapid refresh function to reach the animation effect, to achieve the drawing board function.

1. Open the example:

   Arduino IDE→File→Examples→Select Lib→Select example (TFTLogoVideo)

2. Environment Setting:

   - Prepare a micro SD card.
   - Download win32-disk-imager (https://win32diskimager.download/download-win32-disk-imager/), refer to the above link directly for the download and use method.
   - Write the “Video.img” in the TFTLogoVideo example folder to the SD card by the win32-disk-imager.
   - Insert SD into the BMD58T280.

3. To generate a new Video.img (160×192 pixels), refer directly to the instructions in the file: how_to_create_imgfile. The file path is TFTLogoVideo Image_src\how_to_create_img_file.txt.

Note: This example only supports the BMduino EBI method.

4. Example Description:

a. Create object & initialise object

```
#include <BMD58T280.h>
#include <SD.h>
#include <SDReadRawData.h>
SDReadRawData sdcard;  // Create an SD object
BMD58T280 TFTscreen;   // Create an SD object
#define sd_cs  10       // SD pin definition
#define IMAGE_HEIGHT       160
#define IMAGE_WIDTH        192
#if defined(ARDUINO_AVR_UNO)
#ERROR "This example don't support UNO. Please use the BMduino
        development board instead."
#endif
void setup()
{
  Serial.begin(9600);  // Serial port initialisation
  // Attempt to access the SD card. If fails, for example, no card
  // exists, the setup process will stop.
  if (!sdcard.init(sd_cs)
  {
    Serial.println(F("failed!"));
    return;
  }
  Serial.println(F("succeeded!"));
  TFTscreen.begin();   // LCD initialisation
}
```

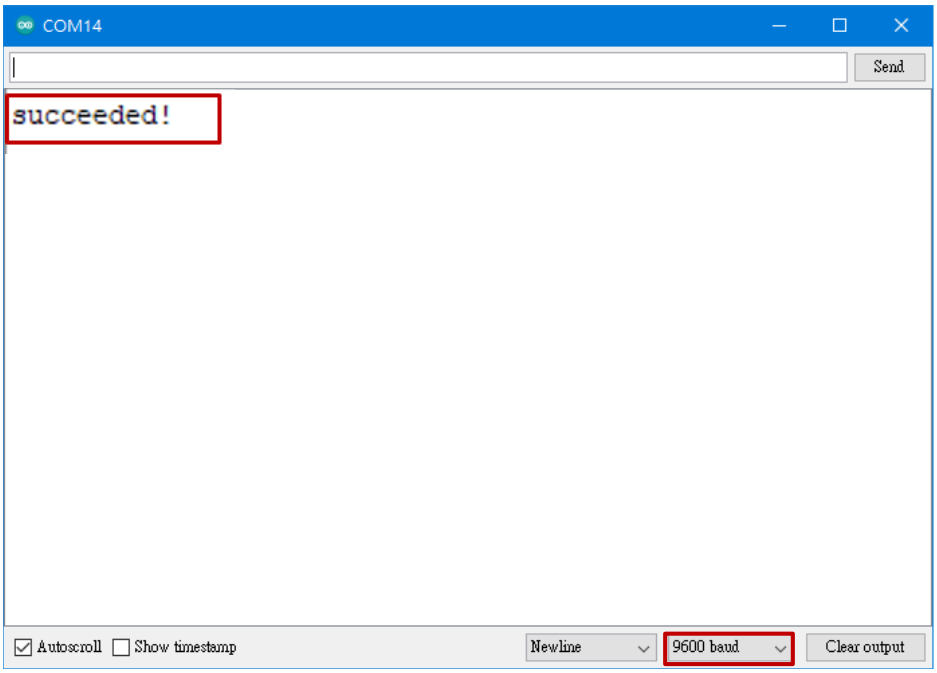
b. loop

```
void loop()
{
  // Show the first video with blue background, clear the GLCD screen
  // before starting
  TFTscreen.background(0, 216, 255);   // Blue background
  // Display 50 images on the LCD
  for(int i =0;i<50;i++)
  {
    // Display an image of 160×192 pixels, starting at (40, 69)
    isplayImage(40,69,IMAGE_HEIGHT,IMAGE_WIDTH,IMAGE_HEIGHT*IMAGE_
                WIDTH*2*i);
  }
  // Display the second video with grey background
  // Clear the GLCD screen before starting
  TFTscreen.background(135, 144, 146); // Grey background
  // Display 25 images on the LCD
  for(int i =50;i<75;i++)
  {
    // Display an image of 160×192 pixels, starting at (40, 69)
    isplayImage(40,69,IMAGE_HEIGHT,IMAGE_WIDTH,IMAGE_HEIGHT*IMAGE_
                WIDTH*2*i);
  }
}
#define SD_BLOCK_BYTE_SIZE              512
#define SD_BLOCK_PIXEL_SIZE             (SD_BLOCK_BYTE_SIZE / 2)
#define SD_BLOCK_BYTE_SIZE_WITH_CRC     (SD_BLOCK_BYTE_SIZE + 2)
#define HALF_ADDRESS_INDEX              516
```

```
// Image buffer
u8 dataBuffer[1032];
// Image buffer start address
u8 *pStartDatabuffer = &(dataBuffer[0]);
// Image buffer half address
u8 *pHalfDatabuffer = &(dataBuffer[HALF_ADDRESS_INDEX]);
void displayImage(uint16_t x, uint16_t y, uint16_t w, uint16_t h,
                  uint32_t SDOffset)
{
  SDOffset = SDOffset/SD_BLOCK_BYTE_SIZE;// Obtain the SD block index
  TFTscreen.setAddrWindow(x, y, w, h);// Set the starting position,
                                      // width and length of the image
/* To speed up, read the 514 bytes of the SD card in the first. Read
   the block data of the SD card, 512-byte (data)+2-byte(CRC)=514-byte.
   Store data to the buffer beginning */
  sdcard.readBlock(SDOffset, SD_BLOCK_BYTE_SIZE_WITH_CRC,
                   pStartDatabuffer);
  sdcard.waitDmaFinish();  // Wait for the 514 bytes reading complete
                           // (Data at the buffer beginning)
  for(int i=1;i<(w*h/SD_BLOCK_PIXEL_SIZE);i++)
  {
/* Read the block data of the SD card, 512-byte(data)+2-byte(CRC)=514-
   byte. store the data to the half buffer*/
    sdcard.readBlock(SDOffset + i, SD_BLOCK_BYTE_SIZE_WITH_CRC,
                     pHalfDatabuffer);
    // Display 512 bytes of data on the LCD, skip the 2-byte CRC. (Data
    // at the buffer beginning)
    for(int j=0;j<SD_BLOCK_BYTE_SIZE;j+=2)
    {
      // Convert the raw data to color
      u16 _color = (u16)((dataBuffer[j+1]<<8) | dataBuffer[j]);
      // Speed up data writing through the basic EBI interface
      TFTscreen.lowLevel->writeData16(_color);
    }
    sdcard.waitDmaFinish();// Wait for the 514 bytes reading complete.
                           // (Half buffer data)
    i++;
    // Confirm that the image data is read completely from SD
    if(i < (w*h/SD_BLOCK_PIXEL_SIZE))
    {
/* Read the data of the SD block, 512-byte(data)+2-byte(CRC)=514-byte.
   Store data to the buffer beginning. (Data at the buffer beginning) */
      sdcard.readBlock(SDOffset + i, SD_BLOCK_BYTE_SIZE_WITH_CRC,
                       pStartDatabuffer);
    }
/* The LCD displays 512 bytes data, skip the 2-byte CRC (Half buffer
   data) */
    for(int j=0;j<SD_BLOCK_BYTE_SIZE;j+=2)
    {
      u16 _color = (u16)((dataBuffer[j+HALF_ADDRESS_INDEX+1]<<8) |
                         dataBuffer[j+HALF_ADDRESS_INDEX]);
      TFTscreen.lowLevel->writeData16(_color);
    }
    sdcard.waitDmaFinish(); // Wait for the 514 bytes reading complete
  }
}
```

5. Open the serial monitor and set the baud rate to be 9600. The serial monitor will display as

follows.


COM14
Send
succeeded!
Autoscroll
Show timestamp
Newline
9600 baud
Clear output