OPEN-SMART

Name: Serial TFT User manual

Version: v1.0

Date: 2018.05.20

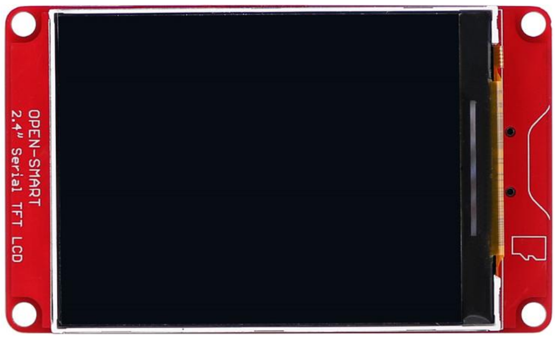
CONTENT

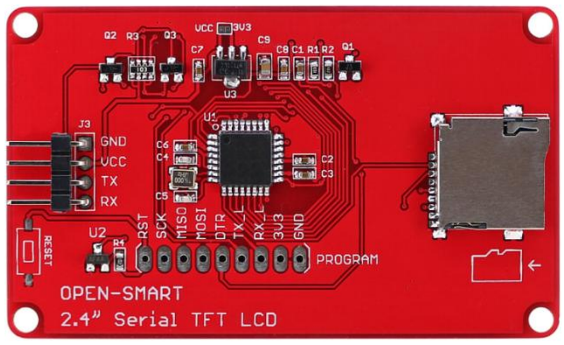
0 update record

Version	Content	Date
V1.0	**Initial Release**	**2018.05.20**

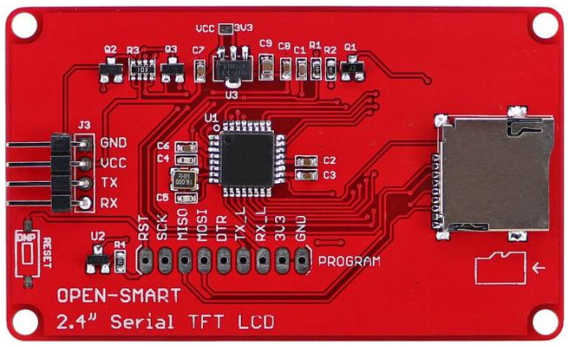
1 Description

This is the one without touch screen.





This is the one with touch screen.



This is a 2.4 inch TFT LCD expansion board using standard serial UART interface and it has good compatibility. It integrates TF card holder, level conversion circuit, and the secondary development is less difficult.

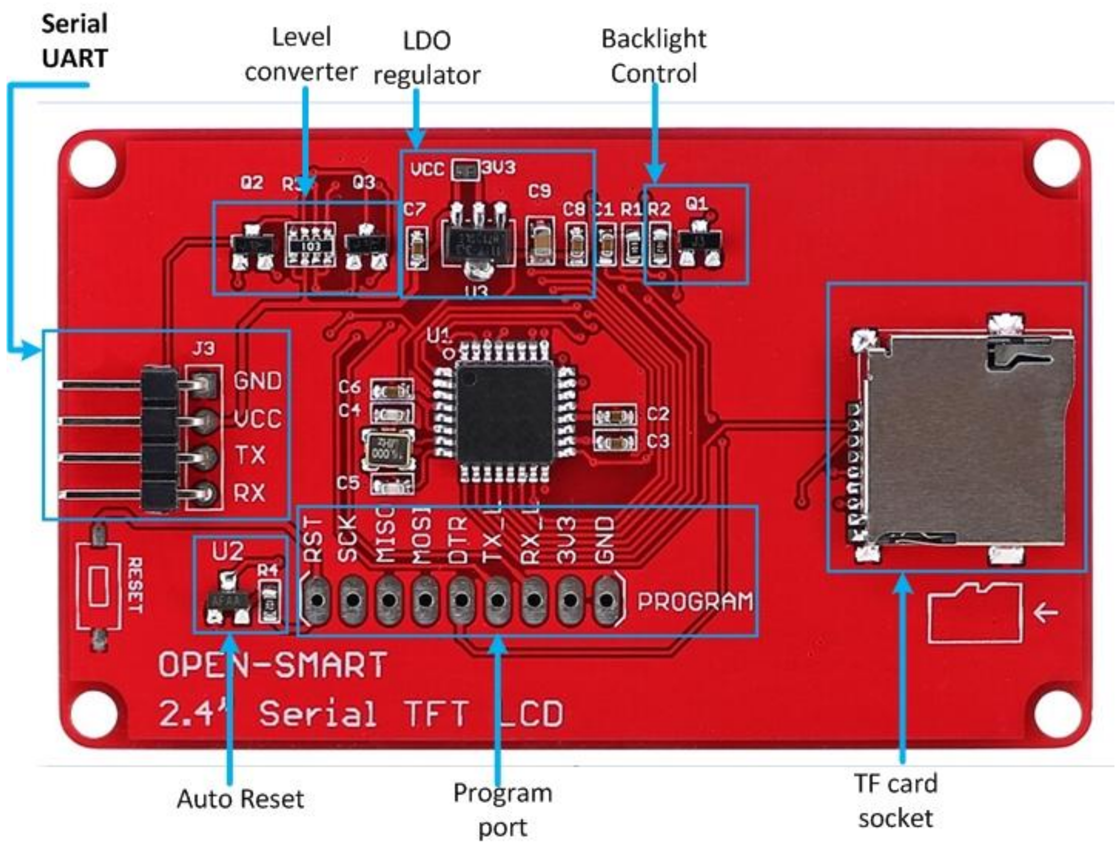
Features:

- Compatibility: compatible with 3.3V / 5V MCU, such as Arduino UNO R3, OPEN-SMART UNO R3, STM32, 51MCU, RPi
- Built-in functions: Only a few commands are needed to display the pictures in the TF card, displaying characters, numbers, graphics.
- Working voltage: 4.5V - 5.5V (On-board 3.3v LDO Regulator),if short the pad VCC to 3V3, you can use 3.1-3.4V.
- Interface logic voltage: 3.3V / 5V compatible(On-board level shift circuit)
- Working current: 90mA(MAX)
- Serial port baud rate: 9600 (default), can use command to modify it to be other values: 19200, 38400, 57600, 115200
- Serial communication format: 8N1
- You can debug the module with the USB to TTL module, such as FT232RL, CH340G module.
- Onboard ATmega328 Series MCU
- **Touch screen type: resistive touchscreen.(If it has touch screen)**
- **Touch Pen: length is 9cm; (If it has touch screen)**
- Resolution: 240X320;
- Display size: 2.4 inch;
- Onboard Micro SD slot, support Micro SD / TF Card;
- Great for DIY

2 Specification

Item	Min	Typical	Max	Unit
Power Supply(VCC)	4.5	**5**	5.5	V
Power Supply(VCC) If VCC short to 3V3	3.1	**3.3**	3.4	V
Current（@VCC=5V）	/	/	90	mA
Logic interface	3.3V / 5V TTL			/
Supported Card Type	Micro SD card(<=2G); Mirco SDHC card(<=32G)			/
File system format	Fat16 / Fat32			/
Uart baud rate	9600(default), 19200, 38400, 57600, 115200			bps

3 Interface



Serial UART: communication port between external MCU and the Serial TFT. There is on-board level shift circuit so that logic level of RX / TX is 3.3V / 5V compatible. The communication format is 8N1 and 9600bps(default).

Level converter: convert the level of RX / TX pins to be 3.3V for the onboard MCU.

LDO regulator: Regulate the input voltage VCC to 3.3V for the MCU and the TFT, also if you short the VCC to 3V3, you can supply 3.3V for the VCC pin directly.

Backlight Control: use a Transistor to turn on or turn off the backlight, and it is connected to PWM pin of the MCU, so that the backlight can be controlled by duty cycle.

Auto Reset: auto reset IC make it easy to hardware reset the MCU and the TFT when it is power on.

Program port: we can use OPEN-SMART 10P Test Fixture to upload firmware for Arduino to onboard ATmega328PB to design your own individual Serial TFT. Board type in IDE: Arduino/Genuino Uno

TF card socket: plug the TF card into it and it is easy to display the picture in bmp format.

4 Usage

4.1 Command Review

4.1.1 Asynchronous serial port control play mode:

Command bytes: StartByte Len CMD data EndByte **Note: the number of whole command bytes should not more than 64**		
Mark	**Byte**	**Byte description**
StartByte	0X7E	Every command should start with $(0x7E)
Len	0Xxx	The number of bytes following by the Len byte
CMD	0Xxx	Such as SET_READ_CURSOR and SET_TEXTCOLOR and so on
data	0Xxx…	The length of the data is limit to be no more than 60
EndByte	0XEF	Ending byte of the command

4.1.2 Commonly Command bytes Review

Details go to 4.2

Command	Command bytes (HEX)	Remark
TEST	7E 02 **00** EF	Test command
SET_READ_CURSOR	7E 02 **01** EF	Read the current cursor (x, y)
	7E 06 **01** xH xL yH yL EF	setCursor(int16_t x, int16_t y)
	7E 06 **01** 00 10 00 20 EF	Set the cursor to (0x10, 0x20)
SET_TEXTCOLOR	7E 04 **02** cH cL EF	setTextColor(uint16_t c)
	7E 04 **02** F8 00 EF	Set text color to be RED, the 16bit is RGB565 format.
SET_TEXTSIZE	7E 03 **03** size EF	setTextSize(uint8_t size)
SET_ROTATION	7E 03 **04** rota EF	setRotation(uint8_t rota) rota = 0, 1, 2, 3
RESET	7E 02 **05** EF	Soft reset for the whole serial TFT
SET_BACKLIGHT	7E 03 **06** bightness EF	setBacklight(uint8_t bightness) brightness: 0 - 255
	7E 03 **06** FF EF	Set Backlight to max,0xFF = 255
PRINTLN	7E 02 **10** EF	A new line
PRINT_CHAR_ARRAY	7E 07 **11** 48 65 6F 6C 6F EF	Print a string “Hello” whose ASCII code is 48 65 6F 6C 6F
PRINT_INT_8	7E 05 **12** Sign Format data EF	Print 8-bit data
	7E 05 **12** 00 0A F0 EF	00 tells that is unsigned number; 0x0A = 10, which tells that is Decimal number; 0xF0 = 240, so this will print **240** on the TFT
	7E 05 **12** 01 0A F0 EF	01 tells that is signed number; 0x0A

		= 10, which tells that is Decimal number; 0xF0 with sign is -16, so this will print **-16** on the TFT
PRINT_INT_16	7E 06 **13** Sign Format dH dL EF	Print 16-bit data
PRINT_INT_32	7E 08 **14** Sign Format dHH dHL dLH dLL EF	Print 32-bit data
FILL_SREEN	7E 04 **20** cH cL EF	Fill the screen with the color you set
DRAW_PIXEL	7E 08 **21** xH xL yH yL cH cL EF	Draw a pixel with the color and coordinate (x,y)
DRAW_FASTVLINE	7E 0A **22** xH xL yH yL hH hL cH cL EF	Draw a vertical line, with coordinate (x,y), **h** is height, **c** is color
DRAW_FASTHLINE	7E 0A **23** xH xL yH yL wH wL cH cL EF	Draw a horizontal line, with coordinate (x,y), **w** is width, **c** is color
DRAW_LINE	7E 0C **24** x0H x0L y0H y0L x1H x1L y1H y1L cH cL EF	Draw a line from the start point(**x0,y0**) to the end point(**x1,y1**)
DRAW_RECT	7E 0C **25** xH xL yH yL wH wL hH hL cH cL EF	Draw rectangle with (x,y), width, height, color
FILL_RECT	7E 0C **26** xH xL yH yL wH wL hH hL cH cL EF	Fill rectangle with (x,y), width, height, color
DRAW_CIRCLE	7E 0A **27** x0H x0L y0H y0L rH rL cH cL EF	Draw a circle with center coordinates (x0,y0) and radius, color
FILL_CIRCLE	7E 0A **28** x0H x0L y0H y0L rH rL cH cL EF	Fill a circle with center coordinates (x0,y0) and radius, color
DRAW_TRIANGLE	7E 10 **29** x0H x0L y0H y0L x1H x1L y1H y1L x2H x2L y2H y2L cH cL EF	Draw a triangle with three vertex coordinates
FILL_TRIANGLE	7E 10 **2A** x0H x0L y0H y0L x1H x1L y1H y1L x2H x2L y2H y2L cH cL EF	Fill a triangle with three vertex coordinates
DRAW_ROUNDRECT	7E 0E **2B** xH xL yH yL wH wL hH hL rH rL cH cL EF	Draw a rectangle with rounded corners
FILL_ROUNDRECT	7E 0E **2C** xH xL yH yL wH wL hH hL rH rL cH cL EF	Fill a rectangle with rounded corners
DRAW_BMP	7E 0A **30** 6D 69 6E 69 77 6F 6F 66 EF	Draw **miniwoof.bmp in the TF card, ASIIC code of miniwoof is** 6D 69 6E 69 77 6F 6F 66
WRITE_READ_BAUD	7E 02 **40** EF	Read baud rate of the TFT
	7E 03 **40** Baud EF	Set the baud rate of the TFT
READ_VERSION	7E 02 **41** EF	Read the version of the firmware
READ_DRIVER_ID	7E 02 **42** EF	Read the driver ID of the TFT LCD
READ_RESOLUTION	7E 02 **43** EF	Read the resolution of the TFT

4.1.3 TFT Feedback

Details go to 4.2

Feedback bytes (HEX)	Remark
7E 03 6F 6B EF	ASIIC code of "ok" is 6F 6B, it will send this while the TFT initialization is done. Also when the command you sent is executed, it will send this.
7E 03 65 31 EF	ASIIC code of "e1" is 65 31, tells that the TF card failed to initialize, maybe you have not plug the TF card.
7E 03 65 32 EF	ASIIC code of "e2" is 65 32, tells that can not open the .bmp file you want, maybe the file is not exist or maybe there is something wrong with the bmp.
7E 06 **01** xH xL yH yL EF	Tell that the cursor is at (x, y) and x = xH*256+xL, y = yH*256+yL
7E 06 **07** xH xL yH yL EF	Tell the touch value x and y, so that you can judge which point is touched, and x = xH*256+xL, y = yH*256+yL
7E 03 **40** Baud EF	Tell the baud rate it use now. Baud: 0 = 9600 1 = 19200 2 = 38400 3 = 57600 4 = 115200
7E 03 **41** Version EF	If Version is 0x10, that tells the firmware version is v1.0
7E 04 **42** idH idL EF	Tell the driver IC of the TFT itself. If id is ILI9325, idH = 0X93, idL = 0X25.
7E 06 **43** xH xL yH yL EF	Tell the resolution of the TFT, and x = xH*256+xL, y = yH*256+yL

4.2 Command and feedback description

4.2.00 TEST

Command bytes: 7E 02 **00** EF

Return: 7E 03 6F 6B EF

You can send it after setting the baud rate to check whether the Serial TFT is ready.

4.2.01 SET_READ_CURSOR

When you want to know the current cursor, you can send 7E 02 **01** EF

Return: 7E 06 **01** xH xL yH yL EF 7E 03 6F 6B EF

Tell that the cursor is at (x, y) and x = xH*256+xL, y = yH*256+yL

When you want to set the current cursor to display bmp picture or some charactors, you can send: 7E 06 **01** xH xL yH yL EF

Return: 7E 03 6F 6B EF

eg:

If you want to set the cursor to (0x0010, 0x0020),

xH = 0x0010>>8;

xL = 0x0010 & 0xff;

yH = 0x0020>>8;

yL = 0x0020 & 0xff;

4.2.02 SET_TEXTCOLOR

Before you display text, you can set the color of the text. It will keep this setting until you change it again.

All color should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits.

You can send: 7E 04 **02** cH cL EF

Return: 7E 03 6F 6B EF

eg:

If you want to set pure red color, 5bits red part should be 0b11111, blue part should be 0b**000000**, green part should be 0b00000.

So,

cH = 0b11111**000** = 0XF8

cL = 0b**000**00000 = 0x00

4.2.03 SET_TEXTSIZE

Note: Minimum size of one character is 5x7 pixel and there is one pixel between two characters.

Before you display text, you can set the size of the text. It will keep this setting until you change it again. If size is 2, that tells one character is (5x2)(7x2) pixel.

You can send: 7E 03 **03** size EF

Return: 7E 03 6F 6B EF

4.2.04 SET_ROTATION

Before display picture or text, graphics, you can set the rotation of the screen.
You can send: 7E 03 **04** rota EF
Return: 7E 03 6F 6B EF
Note: rota is 0, 1, 2, 3, and it tells 4 kind of different directions for displaying.

4.2.05 RESET

This command will use software to reset the MCU.
You can send: 7E 02 **05** EF
Return: 7E 03 6F 6B EF which tells that reset is done
or Return: 7E 03 65 31 EF 7E 03 6F 6B EF which tells that reset is done but the TF card failed to initialize, maybe you have not plug the TF card.

4.2.06 SET_BACKLIGHT

It can set the brightness of the backlight of the TFT, and it takes effect immediately.
In fact it controls the duty cycle of the PWM to control the brightness. brightness can be 0~0xff (0~255). Recommend you to set it to be 0XC8 (200). Of course if you want to save more power, you can set to be smaller value.
You can send: 7E 03 **06** bightness EF
Return: 7E 03 6F 6B EF

4.2.07 PRINTLN

It is used to make a new line to display other characters or pictures.
You can send: 7E 02 **10** EF
Return: 7E 03 6F 6B EF

4.2.08 PRINT_CHAR_ARRAY

As the whole command bytes can not be more that 64bytes, so you can only print up to 60 characters each time. And you should send ASCII code of the characters.
eg:
You can send: 7E **07** **11** 48 65 6F 6C 6F EF
Return: 7E 03 6F 6B EF
This command print a string “Hello” whose ASCII code is 48 65 6F 6C 6F, **07** in the command bytes tells that there is 7 bytes following by **07**.

4.2.09 PRINT_INT_8

This command will display 8 bit number. You should also tell the sign and format for the number.
If it is signed number, sign should be 1, otherwise 0.
You can send: 7E 05 **12** Sign Format data EF
Return: 7E 03 6F 6B EF
About the format which can only be the following number:
//#define DEC 0x0A
//#define HEX 0x10
//#define OCT 0x08
//#define BIN 0x02
eg: When you want to display unsigned int 8 bit data in HEX format, you can send
7E 05 **12** 00 10 data EF
When you want to display signed int 8bit data (can only in DEC format), you can send
7E 05 **12** 01 0A data EF

4.2.10 PRINT_INT_16

This command will display 16 bit number. You should also tell the sign and format for the number.
If it is signed number, sign should be 1, otherwise 0.
You can send: 7E 06 **13** Sign Format dH dL EF
Return: 7E 03 6F 6B EF
About the format which can only be the following number:
//#define DEC 0x0A
//#define HEX 0x10
//#define OCT 0x08
//#define BIN 0x02
eg: When you want to display unsigned int 16 bit data (0XABCD)in HEX format,
you should know:
Sign = 0x00
Format = 0x10
dH = 0XAB (high 8 bit of the data)
dL = 0XCD(low 8bit of the data)
So you can send:
7E 06 **13** 00 10 AB CD EF
When you want to display signed int 16bit data -300(can only in DEC format)
you should know:
-300(DEC format) = 0XFED4, if you do not know why, please search google
Of course generally you do not have to know this, because the program compiler or IDE have done that. Then
Sign = 0x01
Format = 0x0A
dH = 0XFE (high 8 bit of the data)
dL = 0XD4(low 8bit of the data)
So you can send
7E 06 **13** 01 0A FE D4 EF

4.2.11 PRINT_INT_32

This command will display 32 bit number. You should also tell the sign and format for the number.
If it is signed number, sign should be 1, otherwise 0.
You can send: 7E 08 **14** Sign Format dHH dHL dLH dLL EF
Return: 7E 03 6F 6B EF
About the format which can only be the following number:
//#define DEC 0x0A
//#define HEX 0x10
//#define OCT 0x08
//#define BIN 0x02
eg: When you want to display unsigned int 16 bit data (0X12345678)in HEX format,
you should know:
Sign = 0x00
Format = 0x10
dHH = 0X12 (high 8 bit of high part of the data)
dHL = 0X34(low 8bit of high part of the data)
dLH = 0X56 (high 8 bit of low part of the data)
dLL = 0X78(low 8bit of low part of the data)
So you can send:
7E 08 **14** 00 10 12 34 56 78 EF
When you want to display signed int 32bit data -300(can only in DEC format)
you should know:
-300(DEC format) = 0XFFFFFED4, if you do not know why, please search google
Of course generally you do not have to know this, because the program compiler or IDE have done that. Then
Sign = 0x01
Format = 0x0A
dHH = 0XFF (high 8 bit of high part of the data)
dHL = 0XFF(low 8bit of high part of the data)
dLH = 0XFE (high 8 bit of low part of the data)
dLL = 0XD4(low 8bit of low part of the data)
So you can send:
7E 08 **14** 01 0A FF FF FE D4 EF

4.2.12 FILL_SREEN

This command will fill the whole screen with the color you set. Generally you can clear the screen by filling the screen with black color.

You can send: 7E 04 **20** cH cL EF

Return: 7E 03 6F 6B EF

Note: H is high 8bits of 16 bits data. L is low 8bits of it.

All color should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits.

4.2.13 DRAW_PIXEL

This command can draw one pixel with coordinate (x,y) and color.

You can send: 7E 08 **21** xH xL yH yL cH cL EF

Return: 7E 03 6F 6B EF

Note: H is high 8bits of 16 bits data. L is low 8bits of it.

All color should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits.

4.2.14 DRAW_FASTVLINE

Draw a vertical line, with coordinate (x,y), **h** is height, **c** is color.

All color should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits.

You can send: 7E 0A **22** xH xL yH yL hH hL cH cL EF

Return: 7E 03 6F 6B EF

Note: H is high 8bits of 16 bits data. L is low 8bits of it.

eg: If you want to draw a vertical line at (0x30,0x40) whose height is 0x20 pixels and color is red.

xH = 0x00

xL = 0x30

yH = 0x00

yL = 0x40

hH = 0x00

hL = 0x20

cH = 0xF8

cL = 0x00

So you can send: 7E 0A **22** 00 30 00 40 00 20 F8 00 EF

4.2.15 DRAW_FASTHLINE

Draw a horizontal line, with coordinate (x,y), **w** is width, **c** is color

You can send: 7E 0A **23** xH xL yH yL wH wL cH cL EF

Return: 7E 03 6F 6B EF

you can refer to 4.2.14

4.2.16 DRAW_LINE

Draw a line from the start point(**x0,y0**) to the end point(**x1,y1**).

You can send: 7E 0C **24** x0H x0L y0H y0L x1H x1L y1H y1L cH cL EF

Return: 7E 03 6F 6B EF

4.2.17 DRAW_RECT

Draw rectangle start at (x,y) with width, height, color.

You can send: 7E 0C **25** xH xL yH yL wH wL hH hL cH cL EF

Return: 7E 03 6F 6B EF

You can refer to 4.2.14

4.2.18 FILL_RECT

Fill rectangle start at (x,y) with width, height, color.

You can send: 7E 0C **26** xH xL yH yL wH wL hH hL cH cL EF

Return: 7E 03 6F 6B EF

You can refer to 4.2.14

4.2.19 DRAW_CIRCLE

Draw a circle with center coordinates (x0,y0) and radius, color

You can send: 7E 0A **27** x0H x0L y0H y0L rH rL cH cL EF

Return: 7E 03 6F 6B EF

You can refer to 4.2.14

4.2.20 FILL_CIRCLE

Fill a circle with center coordinates (x0,y0) and radius, color

You can send: 7E 0A **28** x0H x0L y0H y0L rH rL cH cL EF

Return: 7E 03 6F 6B EF

You can refer to 4.2.14

4.2.21 DRAW_TRIANGLE

Draw a triangle with its three vertex coordinates

You can send: 7E 10 **29** x0H x0L y0H y0L x1H x1L y1H y1L x2H x2L y2H y2L cH cL EF

Return: 7E 03 6F 6B EF

You can refer to 4.2.14

4.2.22 FILL_TRIANGLE

Fill a triangle with its three vertex coordinates

You can send: 7E 10 **2A** x0H x0L y0H y0L x1H x1L y1H y1L x2H x2L y2H y2L cH cL EF

Return: 7E 03 6F 6B EF

You can refer to 4.2.14

4.2.23 DRAW_ROUNDRECT

Draw a rectangle with rounded corners.

You can send: 7E 0E **2B** xH xL yH yL wH wL hH hL rH rL cH cL EF

Return: 7E 03 6F 6B EF

First it will draw a normal rectangle at start point (x,y) with height (h), width (w), color (c).

And then turn the 4 corners into an arc with a radius (r)

You can refer to 4.2.14

4.2.24 FILL_ROUNDRECT

Fill a rectangle with rounded corners.

You can send: 7E 0E **2C** xH xL yH yL wH wL hH hL rH rL cH cL EF

Return: 7E 03 6F 6B EF

First it will draw a normal rectangle at start point (x,y) with height (h), width (w), color (c).

And then turn the 4 corners into an arc with a radius (r)

And then fill it with color (c)

You can refer to 4.2.14

4.2.25 DRAW_BMP

Draw a bmp picture according to the name you send.

The number of the characters of the picture name should not be more than 8.

In general, you can set cursor and rotation before draw a bmp.

eg:

if you want to draw miniwoof.bmp in TF card, **ASIIC code of miniwoof is** 6D 69 6E 69 77 6F 6F 66, usually you do not have to know this, because the program compiler or IDE have done that.

You can send: 7E 0A **30** 6D 69 6E 69 77 6F 6F 66 EF

Return: 7E 03 6F 6B EF

Note:

When power on, if you get

7E 03 65 31 EF

ASIIC code of "e1" is 65 31, tells that the TF card failed to initialize, maybe you have not plug the TF card.

When you want to draw bmp, if you get

7E 03 65 32 EF

ASIIC code of "e2" is 65 32, tells that can not open the .bmp file you want, maybe the file is not exist or maybe there is something wrong with the bmp.

4.2.26 WRITE_READ_BAUD

This command can set baud rate and get to know the baud rate of serial TFT.

You can send: 7E 02 **40** EF

Return: 7E 03 **40** Baud EF 7E 03 6F 6B EF

Baud:

0 = 9600

1 = 19200

2 = 38400

3 = 57600

4 = 115200

Also you can set the baud rate of the serial TFT,

You can send: 7E 03 **40** Baud EF

Baud:

0 = 9600

1 = 19200

2 = 38400

3 = 57600

4 = 115200

As the baud rate is changed, so ignore the meaningless return bytes.

About 500ms later, you can send TEST command to check if it is ok.

4.2.27 READ_VERSION

This command helps to know the version of the firmware.

You can send: 7E 02 **41** EF

Return: 7E 03 **41** Version EF 7E 03 6F 6B EF

If Version is 0x10, that tells the firmware version is v1.0

4.2.28 READ_DRIVER_ID

This command helps to know the driver IC of the TFT itself.

You can send: 7E 02 **42** EF

Return: 7E 04 **42** idH idL EF 7E 03 6F 6B EF

If idH = 0X93, idL = 0x25, that the driver IC is ILI9325.

4.2.29 READ_RESOLUTION

This command tells the resolution of the TFT.

You can send: 7E 02 **43** EF

Return: 7E 06 **43** xH xL yH yL EF 7E 03 6F 6B EF

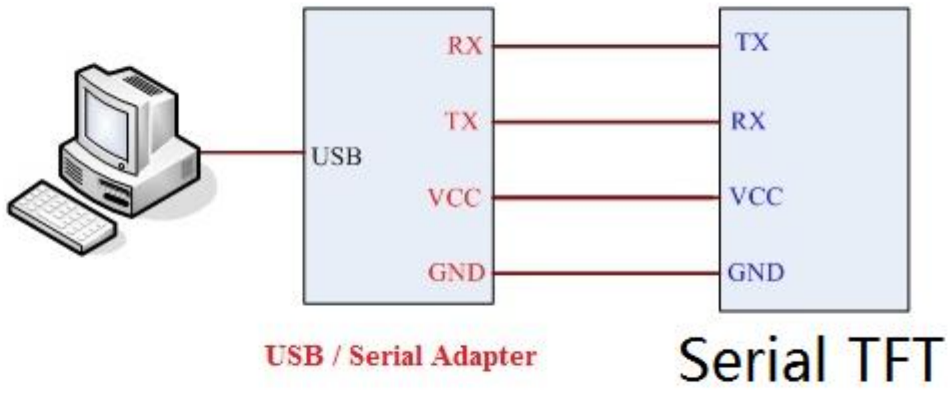
and you can calculate the width (x) and height (y) of the serial TFT.

x = xH*256+xL

y = yH*256+yL

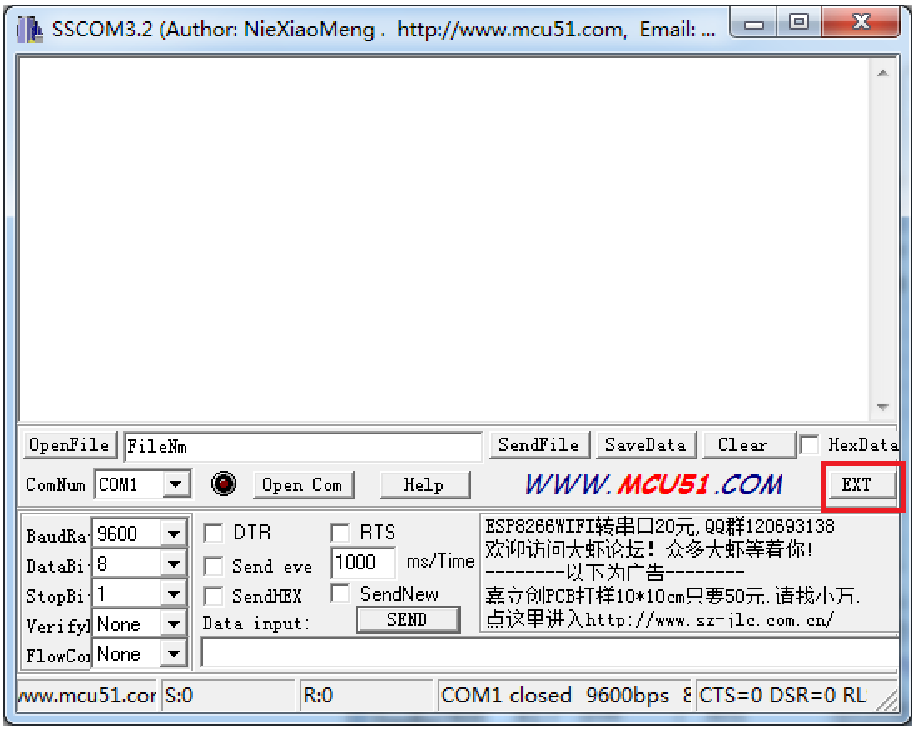
4.3 Use USB to Uart TTL module

(1) You need a **USB to Uart TTL module** (such as USB/Serial Adapter) to connect **Serial MP3 Player** to PC. The hardware installation as show below:



(2) After the connection is completed, open SSCOM32 Serial Tool for Serial TFT that you can down load from OPEN-SMART net disk to send commands. About the specific commands, please refer to 4.1.2 and 4.2 part.

(3) Click the EXT button and then you can manage the commands to be sent.

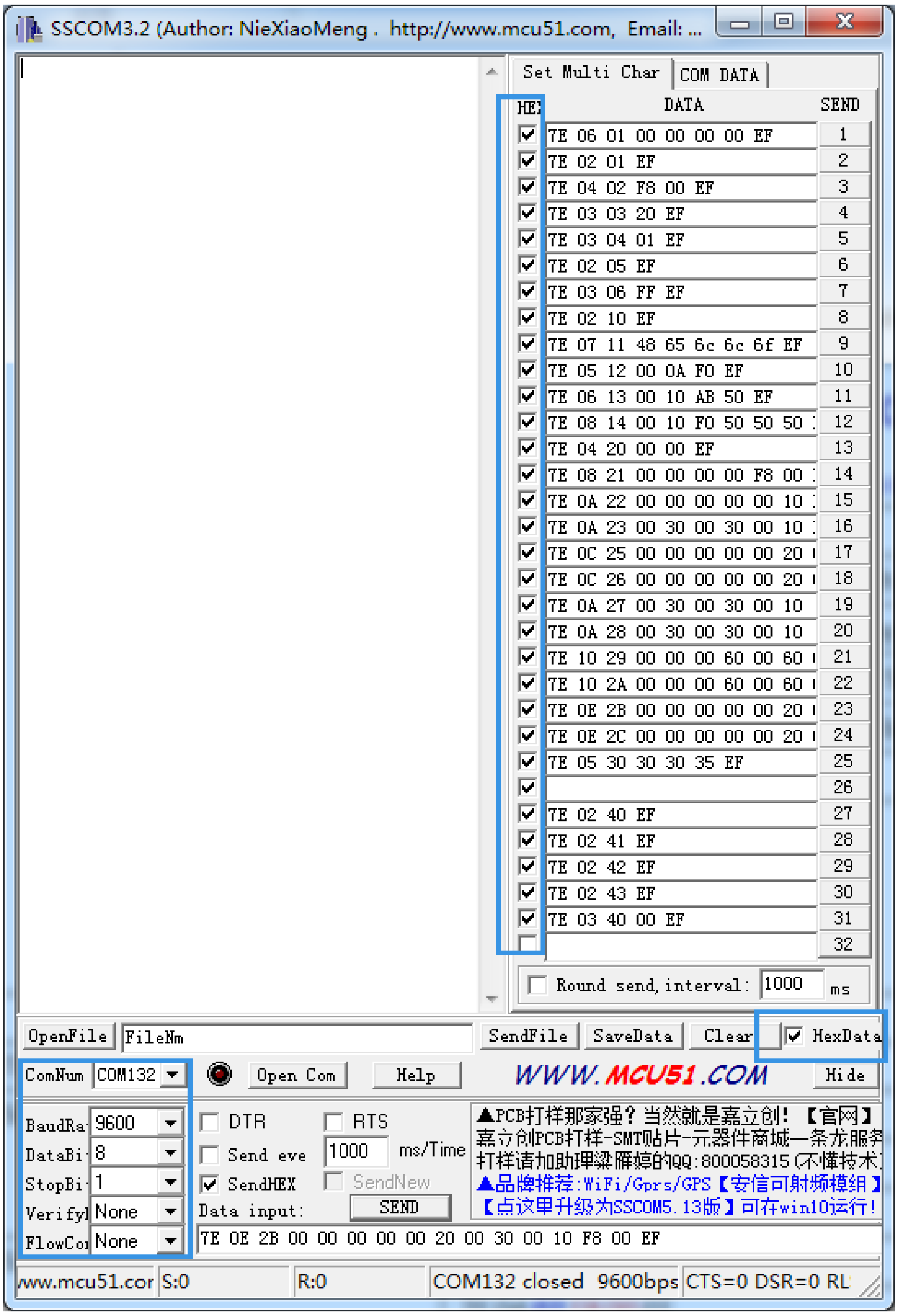


ComNum COM1
BaudRa 9600
DataBi 8
StopBi 1
Verify None
FlowCo None
Send eve 1000 ms/Time

ESP8266WIFI转串口20元,QQ群120693138
欢迎访问大虾论坛! 众多大虾等着你!
--------以下为广告--------
嘉立创PCB打样10*10cm只要50元.请找小万.
点这里进入http://www.sz-jlc.com.cn/

www.mcu51.com S:0 R:0 COM1 closed 9600bps 8 CTS=0 DSR=0 RL

(3) Baud rate should be 9600. Tick HEX and HexData so that the command can be received by the Serial MP3 Player and you can see the feedback information in the blank of the window. Before sending commands, you should select the [ComNum] and click [Open Com].


SSCOM3.2 (Author: NieXiaoMeng . http://www.mcu51.com, Email: ...

Set Multi Char | COM DATA

HE)	DATA	SEND
✓	7E 06 01 00 00 00 00 EF	1
✓	7E 02 01 EF	2
✓	7E 04 02 F8 00 EF	3
✓	7E 03 03 20 EF	4
✓	7E 03 04 01 EF	5
✓	7E 02 05 EF	6
✓	7E 03 06 FF EF	7
✓	7E 02 10 EF	8
✓	7E 07 11 48 65 6c 6c 6f EF	9
✓	7E 05 12 00 0A F0 EF	10
✓	7E 06 13 00 10 AB 50 EF	11
✓	7E 08 14 00 10 F0 50 50 50	12
✓	7E 04 20 00 00 EF	13
✓	7E 08 21 00 00 00 00 F8 00	14
✓	7E 0A 22 00 00 00 00 00 10	15
✓	7E 0A 23 00 30 00 30 00 10	16
✓	7E 0C 25 00 00 00 00 00 20	17
✓	7E 0C 26 00 00 00 00 00 20	18
✓	7E 0A 27 00 30 00 30 00 10	19
✓	7E 0A 28 00 30 00 30 00 10	20
✓	7E 10 29 00 00 00 60 00 60	21
✓	7E 10 2A 00 00 00 60 00 60	22
✓	7E 0E 2B 00 00 00 00 00 20	23
✓	7E 0E 2C 00 00 00 00 00 20	24
✓	7E 05 30 30 30 35 EF	25
✓		26
✓	7E 02 40 EF	27
✓	7E 02 41 EF	28
✓	7E 02 42 EF	29
✓	7E 02 43 EF	30
✓	7E 03 40 00 EF	31
		32

☐ Round send, interval: 1000 ms

OpenFile | FileNm | SendFile | SaveData | Clear | ✓ HexData

ComNum: COM132 | Open Com | Help | WWW.MCU51.COM | Hide

BaudRa	9600
DataBi	8
StopBi	1
Verify	None
FlowCo	None

☐ DTR ☐ RTS
☐ Send eve 1000 ms/Time
✓ SendHEX ☐ SendNew
Data input: SEND

▲PCB打样那家强？当然就是嘉立创！【官网】
嘉立创PCB打样-SMT贴片-元器件商城一条龙服务
打样请加助理梁雁婷的QQ:800058315 (不懂技术
▲品牌推荐:WiFi/Gprs/GPS【安信可射频模组】
【点这里升级为SSCOM5.13版】可在win10运行!

7E 0E 2B 00 00 00 00 00 20 00 30 00 10 F8 00 EF

www.mcu51.com | S:0 | R:0 | COM132 closed 9600bps | CTS=0 DSR=0 RL

(4) Make sure your micro sd card is formatted as FAT16 or FAT32 and you should put the photos that you download from Google Drive into the TF card.

After power up, you should send the some commands to test the TFT.

More operations ? Please refer to 4.1.2 and 4.2 part.

If you have any questions, please let us know: catalex_inc@163.com

4.4 Functions for Arduino Reference

Limitations

The library uses the software serial library that has the following known limitations:
If using multiple software serial ports, only **one** can receive data at a time.
Not all pins on the Mega and Mega 2560 support change interrupts, so only the following can be used for RX: 10, 11, 12, 13, 14, 15, 50, 51, 52, 53, A8 (62), A9 (63), A10 (64), A11 (65), A12 (66), A13 (67), A14 (68), A15 (69).
Not all pins on the Leonardo and Micro support change interrupts, so only the following can be used for RX: 8, 9, 10, 11, 14 (MISO), 15 (SCK), 16 (MOSI).
On Arduino or Genuino 101 the current maximum RX speed is **57600**bps
On Arduino or Genuino 101 RX doesn't work on Pin 13

Using the library

```
#include <SoftwareSerial.h>      //Software Serial Port
#include <OS_SerialTFT.h>      //Software Serial Port

#define TFT_RX 4//RX of Serial TFT module connect to D4 of Arduino / OPEN-SMART UNO
#define TFT_TX 2//TX of Serial TFT to D2 this pin is RX of the software serial port whose limitations refer to Limitations
SerialTFT myTFT(TFT_RX, TFT_TX);
```

SerialTFT Functions

- begin()
- test()
- setCursor()
- readCursor()
- setTextColor()
- setTextSize()
- setRotation()
- reset()
- setBacklight()
- print()
- println()
- fillScreen()
- drawPixel()
- drawFastHLine()
- drawFastVLine()
- drawLine()
- drawRect()
- fillRect()
- drawCircle()
- fillCircle()
- drawTriangle()
- fillTriangle()
- drawRoundRect()
- fillRoundRect()
- bmpDraw()
- color565()
- touch()

void begin(long speed)

Description

Sets the speed (baud rate) for the serial communication. Supported baud rates are 9600, 19200, 38400, 57600, and 115200.
You should call this first to initialize the Serial Port.

Parameters

speed: the baud rate (long)

Returns

none

uint8_t test()

Description

Test whether it is ok after you have set the baud rate for the Serial TFT.

Parameters

none

Returns

uint8_t: returns whether it is ok, if it is, returns 1; otherwise returns 0.

void setCursor(int16_t x, int16_t y)

Description

Before display picture or text, graphics, you can set the current cursor.

Syntax

myTFT.setCursor(x, y);

Parameters

x: the location on the x-axis you want to start drawing text to the screen

y: the location on the y-axis you want to start drawing text to the screen

Returns

none

void readCursor(int16_t &x, int16_t &y)

Description

You can use this function to know the current cursor.

Syntax

readCursor(x, y);

Parameters

x: it will save the location on the x-axis of the screen

y: it will save the location on the y-axis of the screen

Returns

none

Example

```
#include <SoftwareSerial.h>      //Software Serial Port
#include <OS_SerialTFT.h>      //Software Serial Port

#define TFT_RX 4//RX of Serial TFT module connect to D4 of Arduino / OPEN-SMART UNO
#define TFT_TX 2//TX of Serial TFT to D2 this pin is RX of the software serial port whose
limitations refer to Limitations
SerialTFT myTFT(TFT_RX, TFT_TX);
void setup(){
	Serial.begin(9600);
	myTFT.begin(9600);
	myTFT.reset();
	int x, y;
	myTFT. readCursor ();
	Serial.print(" x = ");
	Serial.print(x);
	Serial.print("    y = ");
	Serial.println(" y = "); //print the current cursor on serial monitor of Arduino IDE

}
void loop()
{}
```

void setTextColor(uint16_t color)

Description

Before you display text, you can set the color of the text. It will keep this setting until you change it again.

Syntax

myTFT. setTextColor(color);

myTFT. setTextColor(myTFT.color565(r, g ,b));

Parameters

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void setTextSize(uint8_t size)

Description

Note: Minimum size of one character is 5x7 pixel and there is one pixel between two characters.

Before you display text, you can set the size of the text. It will keep this setting until you change it again. If size is 2, that tells one character is (5x2)(7x2) pixel.

Syntax

myTFT. setTextSize(size);

Parameters

size: can be 1 or larger.

Returns

none

void setRotation(uint8_t rota)

Description

Before display picture or text, graphics, you can set the rotation of the screen.

Syntax

myTFT. setRotation(rata);

Parameters

rata: it is 0, 1, 2, 3, and it tells 4 kind of different directions for displaying.

Returns

none

uint8_t reset()

Description

This will use software to reset the MCU.

Syntax

myTFT. reset();

Parameters

none

Returns

uint8_t:

Returns 0x6F: TF card is Initialized and reset is done. So that you can draw bmp from the TF card.

Returns 0x65: TF card is not online or not recognized and reset is done.

Returns 0x00: reset is failed

void setBacklight(uint8_t bightness)

Description

Before display picture or text, graphics, you can set the rotation of the screen.

Syntax

myTFT. setRotation(rata);

Parameters

rata: it is 0, 1, 2, 3, and it tells 4 kind of different directions for displaying.

Returns

none

void print()

Description

Prints data on the TFT as human-readable ASCII text. This command can take many forms. Numbers are printed using an ASCII character for each digit. Bytes are sent as a single character. Characters and strings are sent as is. For example:

- myTFT.print(78) displays "78"
- myTFT.print("Hello world.") displays "Hello world."

An optional second parameter specifies the base (format) to use; permitted values are BIN (binary, or base 2), OCT (octal, or base 8), DEC (decimal, or base 10), HEX (hexadecimal, or base 16).

- myTFT.print(78, BIN) displays "1001110"
- myTFT.print(78, OCT) displays "116"
- myTFT.print(78, DEC) displays "78"
- myTFT.print(78, HEX) displays "4E"
- myTFT.print(0x1234,HEX) displays 16bits data "1234"
- myTFT.print(0x12345678,HEX) displays 32bits data "12345678"

Syntax

myTFT.print(val)

myTFT.print(val, base)

Parameters

val: the value to print - any data type, if it is unsigned number, its base is DEC so you can only call print(val)

base: specifies the number base (for integral data types) or number of decimal places (for floating point types)

Returns

none

void println()

Description

Prints data on the TFT as human-readable ASCII text followed by a carriage return character (ASCII 13, or '\r') and a newline character (ASCII 10, or '\n'). This command takes the same forms as myTFT.print(). Please refer to print().

Syntax

myTFT.println(val)

myTFT.println(val, base)

Parameters

val: the value to print - any data type, if it is unsigned number, its base is DEC so you can only call print(val)

base: specifies the number base (for integral data types) or number of decimal places (for floating point types)

Returns

none

void fillScreen(uint16_t color)

Description

Fill the whole screen with the color you set. Generally you can clear the screen by filling the screen with black color.

Syntax

myTFT. fillScreen(color)

Parameters

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void drawPixel(int16_t x, int16_t y, uint16_t color)

Description

Draw one pixel with coordinate (x,y) and color.

Syntax

myTFT. drawPixel(x, y, color)

Parameters

x: the location on the x-axis you want to draw to the screen

y: the location on the y-axis you want to draw to the screen

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void drawFastHLine(int16_t x0, int16_t y0, int16_t w, uint16_t color)

Description

Draw a horizontal line, with start point (x,y), **w** is width, **c** is color

Syntax

myTFT. drawFastHLine(x0, y0, w, color)

Parameters

x0: the location on the x-axis you want to start drawing to the screen

y0: the location on the y-axis you want to start drawing to the screen

w: width of the line (number of pixels)

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void drawFastVLine(int16_t x0, int16_t y0, int16_t h, uint16_t color)

Description

Draw a vertical line, with coordinate (x,y), **h** is height, **c** is color.

Syntax

myTFT. drawFastVLine(x0, y0, h, color)

Parameters

x0: the location on the x-axis you want to start drawing to the screen

y0: the location on the y-axis you want to start drawing to the screen

h: height of the line (number of pixels)

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void drawLine(int16_t x0, int16_t y0, int16_t x1, int16_t y1, uint16_t color)

Description

Draw a line from the start point(**x0,y0**) to the end point(**x1,y1**).

Syntax

myTFT. drawLine(x0, y0, x1, y1, color)

Parameters

x0: the location on the x-axis of the start point of the line

y0: the location on the y-axis of the start point of the line

x1: the location on the x-axis of the end point of the line

y1: the location on the y-axis of the end point of the line

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void drawRect(int16_t x, int16_t y, int16_t w, int16_t h, uint16_t color)

Description

Draw rectangle start at (x,y) with width, height, color.

Syntax

myTFT. drawRect(x, y, w, h, color)

Parameters

x: the location on the x-axis of the start point

y: the location on the y-axis of the start point

w: width of the line (number of pixels)

h: height of the line (number of pixels)

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void fillRect(int16_t x, int16_t y, int16_t w, int16_t h, uint16_t color)

Description

Fill rectangle start at (x,y) with width, height, color.

Syntax

myTFT. fillRect(x, y, w, h, color)

Parameters

x: the location on the x-axis of the start point

y: the location on the y-axis of the start point

w: width of the line (number of pixels)

h: height of the line (number of pixels)

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void drawCircle(int16_t x0, int16_t y0, int16_t r, uint16_t color)

Description

Draw a circle with center coordinates (x0,y0) and radius, color

Syntax

myTFT. drawCircle(x0, y0, r, color)

Parameters

x0: the location on the x-axis of center point of the circle

y0: the location on the y-axis of center point of the circle

r: radius of the circle

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void fillCircle(int16_t x0, int16_t y0, int16_t r, uint16_t color)

Description

Fill a circle with center coordinates (x0,y0) and radius, color

Syntax

myTFT. fillCircle (x0, y0, r, color)

Parameters

x0: the location on the x-axis of center point of the circle

y0: the location on the y-axis of center point of the circle

r: radius of the circle

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void drawTriangle(int16_t x0, int16_t y0, int16_t x1, int16_t y1,

int16_t x2, int16_t y2, uint16_t color)

Description

Draw a triangle with its three vertex coordinates

Syntax

myTFT. drawTriangle(x0, y0, x1, y1, x2, y2, color)

Parameters

x0: the location on the x-axis of first vertex point of the triangle

y0: the location on the y-axis of first vertex point of the triangle

x1: the location on the x-axis of second vertex point of the triangle

y1: the location on the y-axis of second vertex point of the triangle

x2: the location on the x-axis of third vertex point of the triangle

y2: the location on the y-axis of third vertex point of the triangle

r: radius of the circle

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void fillTriangle(int16_t x0, int16_t y0, int16_t x1, int16_t y1,

int16_t x2, int16_t y2, uint16_t color)

Description

Fill a triangle with its three vertex coordinates

Syntax

myTFT. fillTriangle(x0, y0, x1, y1, x2, y2, color)

Parameters

x0: the location on the x-axis of first vertex point of the triangle

y0: the location on the y-axis of first vertex point of the triangle

x1: the location on the x-axis of second vertex point of the triangle

y1: the location on the y-axis of second vertex point of the triangle

x2: the location on the x-axis of third vertex point of the triangle

y2: the location on the y-axis of third vertex point of the triangle

r: radius of the circle

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void drawRoundRect(int16_t x0, int16_t y0, int16_t w, int16_t h,

int16_t r, uint16_t color)

Description

Draw a rectangle with rounded corners. First it will draw a normal rectangle at start point (x0,y0) with height (h), width (w), color

And then turn the 4 corners into an arc with a radius (r)

Syntax

myTFT. drawRoundRect(x0, y0, w, h, r, color)

Parameters

x0: the location on the x-axis of start point

y0: the location on the y-axis of start point

w: width of the line (number of pixels)

h: height of the line (number of pixels)

r: radius of the corner

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void fillRoundRect(int16_t x0, int16_t y0, int16_t w, int16_t h,

int16_t r, uint16_t color)

Description

Fill a rectangle with rounded corners. First it will draw a normal rectangle at start point (x0,y0) with height (h), width (w), color

And then turn the 4 corners into an arc with a radius (r)

And then fill it with color

Syntax

myTFT. fillRoundRect(x0, y0, w, h, r, color)

Parameters

x0: the location on the x-axis of start point

y0: the location on the y-axis of start point

w: width of the line (number of pixels)

h: height of the line (number of pixels)

r: radius of the corner

color: All color in this user manual should match RGB565 format, color is 16bits, and red part is high 5bits, blue part is middle 6bits, green part is low 5bits. Also you can use color565(uint8_t r, uint8_t g, uint8_t b) to get the 16bit color.

Returns

none

void bmpDraw(char *filename)

Description

Draw a bmp picture according to the name you send.

The number of the characters of the picture name should not be more than 8.

In general, you can set cursor and rotation before draw a bmp.

when you want to display the picture miniwoof.bmp, you can call

- myTFT. bmpDraw (“miniwoof”)

Syntax

myTFT. bmpDraw (char *filename)

Parameters

filename: the name of the bmp picture in the root directory of TF card

Returns

none

uint16_t color565(uint8_t r, uint8_t g, uint8_t b)

Description

Change color format to be RGB565 format 16bits color that can be parameter for many functions of the SerialTFT library.

eg:

- myTFT. setTextColor(myTFT.color565(255, 0,0)) this set the text color to be pure red
- myTFT. setTextColor(myTFT.color565(0, 255,0)) this set the text color to be pure green
- myTFT. setTextColor(myTFT.color565(0, 0,255)) this set the text color to be pure blue

Syntax

myTFT. color565(r, g, b)

Parameters

r: 0-255

g: 0-255

b: 0-255

Returns

uint16_t: the RGB565 format 16bits color

uint8_t touch()

Description

Tell the touch value x and y, so that you can judge which point is touched.

In general, you should touch with the pen to find the minimum and maximum value of x and y.

TS_MINX

TS_MINY

TS_MAXX

TS_MAXY

So that you can calculate the exact point the pen touch. Please refer to the demo code tftpaint.ino.

Syntax

myTFT. touch()

myTFT. touchX

myTFT. touchY

Parameters

touchX: after calling touch(), if there is someone touching, it will save x value

touchY: after calling touch(), if there is someone touching, it will save y value

Returns

uint8_t: 0 or 1. if someone touch, it will return 1. otherwise return 0;

4.5 Use Arduino UNO R3

4.5.1 Project1: Simple test for the Serial TFT

You can also watch the video from our Youtube.

Step1: Material preparation

1 x OPEN-SMART UNO R3

1 x Serial TFT without Touch Screen

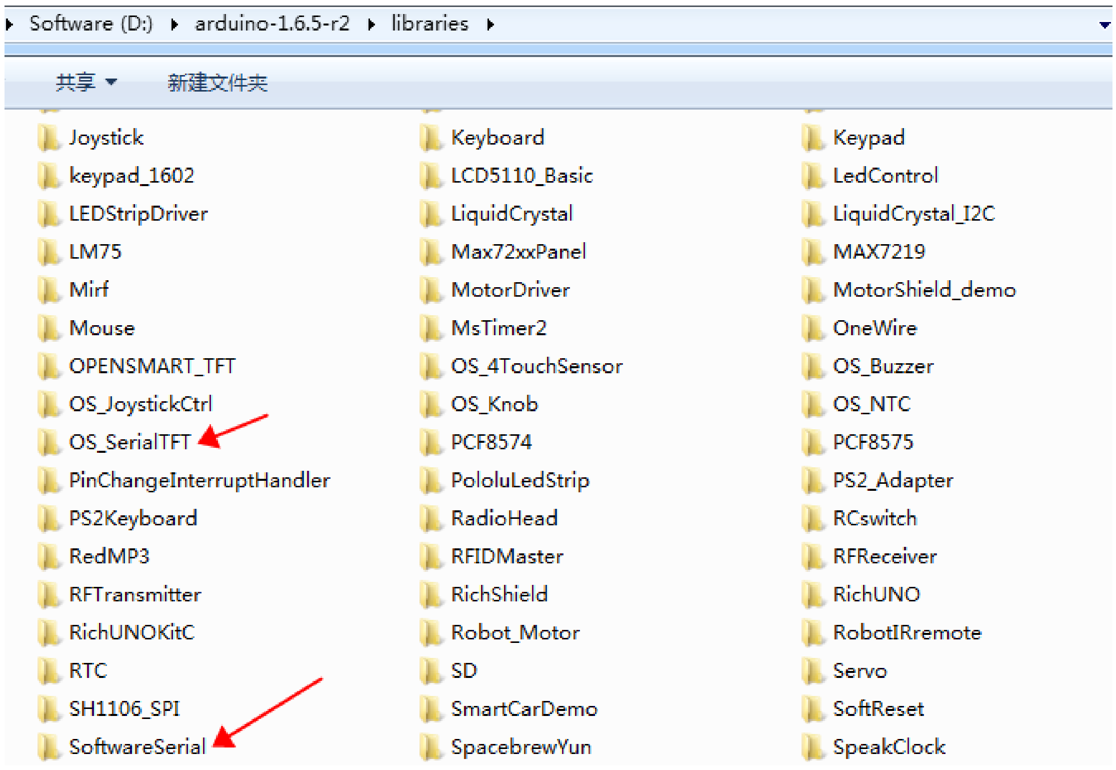
1 x Serial TFT with Touch Screen

1x IO Expansion Shield

4x Female to Female Dupont cables

Step2: Download library for Arduino

Download library for Arduino from Documents download link and then you should put the 2 library into the libraries directory of Arduino IDE.



Software (D:) ▸ arduino-1.6.5-r2 ▸ libraries ▸

共享 ▾ 新建文件夹

Joystick	Keyboard	Keypad
keypad_1602	LCD5110_Basic	LedControl
LEDStripDriver	LiquidCrystal	LiquidCrystal_I2C
LM75	Max72xxPanel	MAX7219
Mirf	MotorDriver	MotorShield_demo
Mouse	MsTimer2	OneWire
OPENSMART_TFT	OS_4TouchSensor	OS_Buzzer
OS_JoystickCtrl	OS_Knob	OS_NTC
OS_SerialTFT	PCF8574	PCF8575
PinChangeInterruptHandler	PololuLedStrip	PS2_Adapter
PS2Keyboard	RadioHead	RCswitch
RedMP3	RFIDMaster	RFReceiver
RFTransmitter	RichShield	RichUNO
RichUNOKitC	Robot_Motor	RobotIRremote
RTC	SD	Servo
SH1106_SPI	SmartCarDemo	SoftReset
SoftwareSerial	SpacebrewYun	SpeakClock

And the example codes are in directory of D:\arduino-1.6.5-r2\libraries\OS_SerialTFT\examples

Step3: Hardware install

Plug the IO Expansion Shield which is just the I/O expansion board to OPEN-SMART UNO R3.

Connect the serial TFT to IO Expansion Shield with the cables

		IO Expansion Shield
Serial TFT	RX	D2
	TX	D4
	VCC	VCC
	GND	GND

Make sure your micro sd card is formatted as FAT16 or FAT32 and you should put the photos that you download from Google Drive into the TF card.

Step4: Power on

Use the USB cable to connect the OPEN-SMART UNO R3 and PC.

NOTE: Do not touch the metal frame of the TFT LCD.

Step5: Learn to display characters, number, graphics

Upload the demo code in the directory of

D:\arduino-1.6.5-r2\libraries\OS_SerialTFT\examples\graphicstest

Then you can see it displays some words, numbers, lines, circles, etc.

Step6: Learn to display bmp picture

Download **pictures used in showbmp demo code.rar**

And then unzip in your PC and copy all the pictures to root directory of your TF card.

Plug the TF card to the Serial TFT.

Upload the demo code in the directory of

D:\arduino-1.6.5-r2\libraries\OS_SerialTFT\examples\showBMP

Then you can see it display pictures and its brief introduction.

Step7: Paint something (If it has touch screen)

Upload the demo code in the directory of

D:\arduino-1.6.5-r2\libraries\OS_SerialTFT\examples\tftpaint

Then you can paint some words or drawing.

If you have any questions, please let us know: catalex_inc@163.com

5 Part List

1x Serial TFT Module

1 x Touch Pen (If the TFT has touch screen)

Documents download link:

https://drive.google.com/drive/folders/1ReWai0mdEofMkcXbM3ROxDt2Ya159-DG?usp=sharing

SSCOM32 Serial Tool for Serial TFT：

https://drive.google.com/drive/folders/1mdHFJRP_ii62utroczuZTO49IKsGS23_?usp=sharing

Buy from OPEN-SMART Official Store:

Serial TFT without Touch Screen

Serial TFT with Touch Screen

Technical Support: catalex_inc@163.com