Display - ST7735/ST7789 TFT

.

Plugin details

Type: Display

Name: ST7735/ST7789 TFT

Status: DISPLAY

GitHub: _P116_ST77xx.ino (https://github.com/letscontrolit/ESPEasy/blob/mega/src/_P116_ST77xx.ino)

Maintainer: *TD-er, tonhuisman, (WDS)*

Used libraries: *Adafruit GFX, Adafruit ST77xx*

Description

The ST7735 and ST7789 chip family drives color TFT displays in various resolutions, and usually supports 65535 colors (RGB565 coded).

This plugin supports these display models:

- **ST7735** with resolutions 128 x 128, 128 x 160 and 80 x 160 pixels
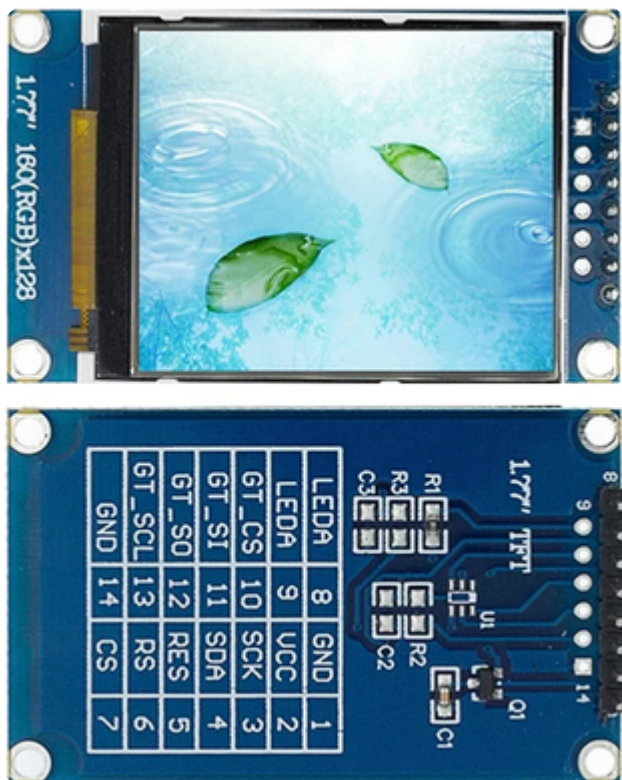- **ST7789** with resolutions 240 x 320, 240 x 240, 240 x 280 and 135 x 240 pixels

The plugin can be configured to show predefined text, a reservation for up to 24 lines is made, but only the lines that can fit on-screen will be shown, or via commands, sent via http or from rules, text or graphical elements like lines, rectangles, circles, pixels, etc., can be shown.
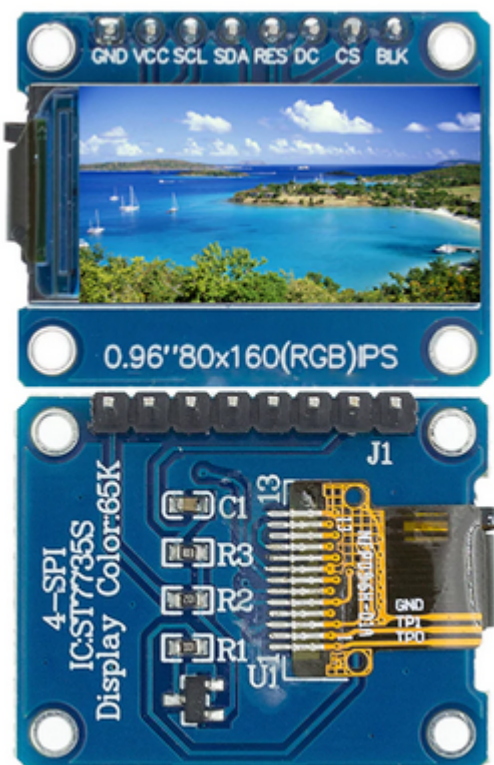
Hardware

A few images, grabbed from suppliers on Aliexpress, showing available & supported displays:
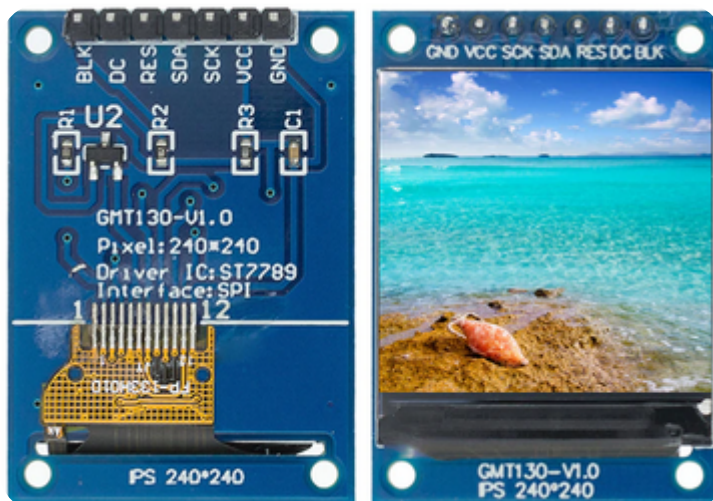
ST7735 128 x 128 pixels

ST7735 128 x 160 pixels

ST7735 80 x 160 pixels

ST7789 240 x 320 pixels

ST7789 240 x 240 pixels (This model has no `CS` pin available)

ST7789 240 x 280 pixels (some have rounded corners, in that case corner-pixels are addressable but not visible)

ST7789 135 x 240 pixels (separate and integrated modules)

Connections

These displays are connected using the SPI interface, minimally with 3 signal wires, though most also support the `CS` signal, so 4-wire SPI can be used, to allow multiple SPI-connected devices.

The text on most displays is somewhat confusing, as not the usual SPI names are used:

```
ESP         TFT
--------    ------
3V3     --- VCC (most displays only support 3.3V)
GND     --- GND
MOSI    --> SDA
CLK     --> SCL/SCK
(gpio)  --> DC
(gpio)  --> CS (optional, not available on all displays)
(gpio)  --> RES (optional)
(gpio)  --> BLK (optional)


(gpio) = configurable GPIO pin.
```

Device

Task Settings

Device:	Display - ST7735/ST7789 TFT [DEVELOPMENT] ❓ ℹ️
Name:	st77xx
Enabled:	☐

Actuator

GPIO → CS (optional):	GPIO-0 (D3) ⚠
GPIO → DC:	GPIO-4 (D2)
GPIO → RES (optional):	- None -
GPIO → Backlight (optional):	- None -
Backlight percentage:	100 [1-100%]
Display button:	- None -
Inversed Logic:	☐
Display Timeout:	0
TFT display model:	ST7735 128 x 128px

Layout

Rotation:	Normal
Text print Mode:	Truncate exceeding message
Font scaling:	1 [1x..10x]
Clear display on exit:	☐
Write Command trigger:	st77xx

Note: Select the command that is used to handle commands for this display.

Wake display on receiving text:	☑

Note: When checked, the display wakes up at receiving remote updates.

Text Coordinates in col/row:	☐

Note: Unchecked: Coordinates in pixels. Applies only to 'txp', 'txz' and 'txtfull' subcommands.

Background-fill for text:	☑

Note: Fill entire line-height with background color.

Content

Foreground color:	white
Background color:	black

Note: Use Color name, '#RGB565' (# + 1..4 hex nibbles) or '#RRGGBB' (# + 6 hex nibbles RGB color).

Note: NB: Colors stored as RGB565 value!

Line 1:	
Line 2:	
Line 3:	
Line 4:	
Line 5:	
Line 6:	
Line 7:	
Line 8:	
Line 9:	
Line 10:	
Line 11:	
Line 12:	
Line 13:	
Line 14:	

Line 20:		
Line 21:		
Line 22:		
Line 23:		
Line 24:		[Remaining: 1000]

Interval: `0` [sec] (Optional for this Device)

Values

#	Name
1	CursorX
2	CursorY

[Close] [Submit] [Delete]

- **Name** A unique name should be entered here.
- **Enabled** The device can be disabled or enabled. When not enabled the device should not use any resources.

Actuator

- **GPIO -> CS (optional)** Select the GPIO pin to use for the `CS` connection. If the display doesn't have a `CS` connection it can be set to *None*.
- **GPIO -> DC** The GPIO pin to use for the `DC` connection (Data/Command).
- **GPIO -> RES (optional)** Select the GPIO pin to use for the `RES` (reset) connection. If the display doesn't have a `RES` (or RST) connection, or no free pin is available, it can be set to *None*. If it is set to None, for proper operation it may need too be wired to the Reset connection on the ESP, so the device is initialized correctly.
- **GPIO -> Backlicht (optional)** Select the GPIO pin to use for controlling the backlight. To save power, the backlight can be dimmed, or turned off if the display is turned off. If set to *None*, usually the max. brightness is used for the backlight.
- **Backlight percentage** The backlight can be controlled via PWM modulation on the Backlight (BLK) pin of the display. This is set as a percentage between 1 and 100.
- **Display button** A GPIO pin can be configured to wake the display on demand. This, combined with the **Display Timeout** setting, can preserve the lifetime of the display, and save some power.
- **Inversed Logic** When checked, reverses the pin-state action of the **Display button** gpio. This allows an external circuit, f.e. an IR sensor, that may provide a *high* signal when activated, to wake the display.
- **Display Timeout** Select the timeout in seconds to turn off the display after the last update or wake-up. Only used if the **Display button** is *also* configured.
- **TFT Display model** Select the hardware model that is connected. Currently there are only preset resolutions available.

Available options:

- *ST7735 128 x 128px* Allows about 12 lines of text in the smallest font scaling setting (the 13th line will be distorted, as the bottom 2 pixellines aren't available)
- *ST7735 128 x 160px* Allows 16 lines of text in the smallest font scaling setting.
- *ST7735 80 x 160px* Allows 16 lines of text in the smallest font scaling setting.
- *ST7789 240 x 320px* Allows 32 lines of text in the smallest font scaling setting. Predefined text only goes to 24, extra lines can be displayed from rules or external commands.
- *ST7789 240 x 240px* Allows 24 lines of text in the smallest font scaling setting.
- *ST7789 240 x 280px* Allows 28 lines of text in the smallest font scaling setting. Predefined text only goes to 24, extra lines can be displayed from rules or external commands.
- *ST7789 135 x 240px* Allows about 13 lines of text in the smallest font scaling setting (the 14th line may be distorted because of a lack of pixellines).

Layout

- **Rotation** Depending on how the display is to be mounted/installed, it may be needed to rotate the content, or with a non-square resolution, to use the display in *Landscape* layout instead of the default *Portrait*.

Available options:

The available options rotate the display in a clock-wise direction.

- **Text print Mode** Defines the handling of text when predefined text or text via the `txt` or `txtfull` subcommands (see below) is displayed.

Available options:

- *Continue to next line* Simply 'prints' all text to the display, with automatic overflow to the next line(s) if the text doesn't fit on the current line.
- *Truncate exceeding message* Display the provided message, but any excess at the right border of the display is discarded.
- *Clear then truncate exceeding message* Clear from the current cursor position to the end of the display, then display the provided message, but any excess at the right border of the display is discarded.
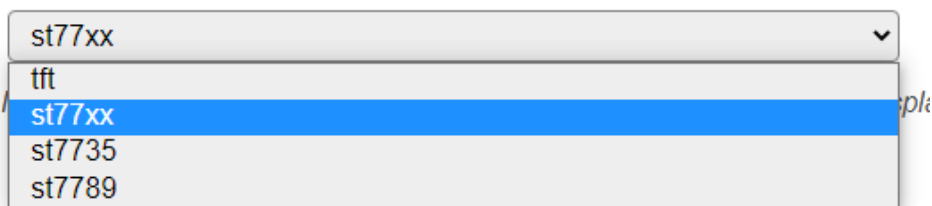
Default setting is *Truncate exceeding message*.

- **Font scaling** The scaling factor for the currently active font. Select a factor between 1 and 10.
- **Clear display on exit** When checked, will clear the display when the task is disabled, either from settings or via

the `TaskDisable` command. The screen will be turned off, and when a backlight pin is configured, also the backlight is turned off.
- **Write Command trigger** The command to handle any commands for this device can be selected here. This can make the commands compatible with other (tft) displays, using the same command structure via the ESPEasy Adafruit Graphics helper class.

Available options:

- *tft*
- *st77xx*
- *st7735*
- *st7789*

The command is handled non-case sensitive. See below for available commands and subcommands.

- **Wake display on receiving text** When checked, the display will be enabled once any content is written to the screen, either triggered by the Interval, or from a command. Default checked.
- **Text Coordinates in col/row** When checked, the coordinates for the `txp`, `txz` and `txtfull` subcommands will be handled in cursor columns & rows, instead of pixels. Column and row are calculated from the current font size and font scaling settings.
- **Background-fill for text** When checked, for any text-line sent to the screen, the entire background (including top and bottom lines) will have the provided background color, *unless* transparent is used (Background color == Foreground color). Default checked.

Content

- **Foreground color**
- **Background color**

These are the default colors, used to display the content as configured below (if any). The background color is also used as the Clear screen color.

Colors can be specified in 3 ways:

- *Color name* Some named colors are recognized: White, Black, Red, Green, Blue, Yellow, Orange, Navy, Darkgreen, Cyan, Darkcyan, Maroon, Purple, Olive, Magenta, Lightgrey, Darkgrey, Pink and Greenyellow. If any of these colors is recognized, the name will be shown.
- *#RGB565* A `#` prefixed, 1 to 4 digit, hex-coded color value in RGB565 format (5 bits red, 6 bits green, 5 bits blue) giving the max. number of 65535 colors. This is the value that is shown after the settings are saved, and it is not one of the known color names.
- *#RRGGBB* A `#` prefixed, 6 digit, hex coded RGB color value. Will be transformed to an RGB565 value!

If the Foreground and Background colors are the same, the background color will become `transparent`. If the Forground color is empty, as a default `white` will be set.

- **Line 1..24** Predefined content can be specified. The number of lines used depends on the size of the display, the font used and the font scaling that is set. On a 240x240 pixel display, with the default font, and font scling set ot 1, all 24 lines will be displayed, but when using font scaling 2, then only the first 12 lines are displayed. If more lines are needed, then they should be set from the rules or an external source, like commands via http.

The usual variables, like `[Taskname#Valuename]`, or `%v1%`, system variables, formulas and functions can be used.

Input length is limited to 60 characters per line. If a longer calculated text is needed for a line, then it should be set from the rules or from an external source via http commands.

Next to Line 24, the remaining capacity in characters is displayed.

The total combination of lines * input length can not exceed 1000 characters (sized dynamically), as there is limited storage per task available for these settings. An error message will be shown after (trying to) save the settings, **any excess content will be discarded!**

Example:

Error: Not all strings fit in custom settings.

- **Interval** By default, Interval will be set to 0. If set to a non-zero value, the pre-configured content will be updated automatically using that interval (seconds). Depending on the **Text print Mode** setting, content that may have been draw from rules or external commands, may be erased.

Values

The current text-cursor position is available in 2 variables **CursorX** and **CursorY**, that will be updated on every change of text or received command. **No** events are generated, and these values will never be sent to controllers.

The values can be used in rules, if desired. They will follow the **Text Coordinates in col/row** setting.

Commands

Command Syntax	Extra information
This plugin allows dynamic configuration of the command-trigger. Available triggers are: • `tft` • `st77xx` • `st7735` • `st7789` The default trigger is `st77xx`	Some commands are postfixed with `cmd`, indicating device specific commands. Not device-specific commands are genericly applicable to all *Adafruit Graphics* enabled devices, using the ESPEasy `AdafruitGFX_helper` module.
`st77xxcmd,on`	Switch the display on.
`st77xxcmd,off`	Switch the display off. If the Backlight is connected and configured, that will also be turned off.
`st77xxcmd,clear`	Clear the display, using the default background color.
`st77xxcmd,backlight,<percentage>`	Set the level of backlight brightness, from 1..100%. Only applicable if a backlight pin is configured.
Generic commands: `<trigger>` is the command-trigger, documented above.	Generic notes: • If a text has comma's or spaces, then it should be 'wrapped' in either double quotes `"`, single quotes `'` or back-ticks `` ` ``. • For `color` arguments, see how colors can be defined in the `txc` subcommand description.
`<trigger>,clear[,<color>]`	Clear the screen using last set background color, or a specified color. Background color is either from configuration or `txc` subcommand.

`<trigger>,rot,<rotation>`	Rotate the display orientation clock-wise from the initial position, where rotation can be: (current display content will stay as-is!) • 0 : No rotation • 1 : 90 degrees • 2 : 180 degrees • 3 : 270 degrees
`<trigger>,tpm,<TextPrintMode>`	Select the Text print Mode • 0 : Continue to next line (wrap text onto the next line) • 1 : Truncate exceeding message (cut-off text that won't fit on the screen width) • 2 : Clear then truncate exceeding message (Clear to width of screen, then print the message)
`<trigger>,txt,<text>`	Write simple text (entire rest of the text provided), use last position, color and size. Color is either from configuration or `txc` subcommand.
`<trigger>,txp,<x>,<y>`	Set text position (move the cursor). Depending on the setting **Text Coordinates in col/row**, these coordinates are pixels (default) or column/rows.
`<trigger>,txz,<x>,<y>,<text>`	Set text position (move the cursor) and print the text. Combines the `txp` and `txt` commands in 1. Depending on the setting **Text Coordinates in col/row**, these coordinates are pixels (default) or column/rows.

`<trigger>,txc,<foreColor>`
`[,<backgroundColor>]`
	Set text color (background is transparent if not provided, or the same as the foreground color).
	
	Colors can be specified in 3 ways:
	
	• *Color name* **Some named colors are recognized:**
	  ○ Full color display: `White`, `Black`, `Red`, `Green`, `Blue`, `Yellow`, `Orange`, `Navy`, `Darkgreen`, `Cyan`, `Darkcyan`, `Maroon`, `Purple`, `Olive`, `Magenta`, `Lightgrey`, `Darkgrey`, `Pink` and `Greenyellow`.
	  ○ 7-color (eInk) displays: `White`, `Black`, `Red`, `Green`, `Blue`, `Yellow` and `Orange`
	  ○ Monochrome (eInk, 1 add. color, and 1 or 2-tone greyscale) displays: `White`, `Black`, `Inverse`, `Red`, `Light` and `Dark`
	
	• *#RGB565* A `#` prefixed, 1 to 4 digit, hex-coded color value in RGB565 format (5 bits red, 6 bits green, 5 bits blue) giving the max. number of 65535 colors.
	• *#RRGGBB* A `#` prefixed, 6 digit, hex coded RGB color value. Will be transformed to RGB565!

`<trigger>,txs,<size>`	Set font scaling option. Only values from 0 to 10 are accepted. 0 assumes 1 by default.

`<trigger>,txtfull,<x>,<y>,<size=1>,`
`<foreColor=white>,`
`<backColor=transparent>,<text>`
	Write text at position X/Y with all options. Depending on the setting **Text Coordinates in col/row**, these coordinates are pixels (default) or column/rows.
	
	All options that have a default specified (size, foreColor, backColor) can be left out, but text is expected. ForeColor and BackColor use the last set Foreground and Background colors (that will by default be white and transparent if not provided).

`<trigger>,asciitable[,<offset>,` `<fontscaling>]`	Show an ascii table on the display, starting at position 0,0. Display is not cleared before drawing the table. `offset` is in steps of 32 characters (0x20), and by default starts at 0x80, showing the 'high ascii' characters for the current font. Applicable range is -4..3, allowing to view the entire supported character set. The CR (0x0D) and LF (0x0A) characters are replaced by a space to not disturb the table on screen. `fontscaling` can be changed, from the default value 2, to enlarge the characters (possibly rendering less characters of the table, hence the offset can be applied). Does not change the current fontscaling setting. This command is not available in builds that use the `LIMIT_BUILD_SIZE` compile-time option, as it is intended for debugging/development use.
`<trigger>,font,<fontname>`	Switch to font. Also applies font settings to recalculate the max. column and row settings. Depending on compile-time settings, none or multiple fonts are available. • *default* The default 6 * 10 font, includes most of the special characters like degrees centigrade and a few valuta symbols. Enabled by default, but disabled in memory-challenged builds: (most extra fonts do *not* include special symbol characters) • *sevenseg24* A rather large 7-segment 21 * 48 font • *sevenseg18* A somewhat less large 7-segment 16 * 34 font • *freesans* A sans-serif 10 * 21 font Usually disabled fonts: (can be enabled in a Custom build, default included in the MAX builds) • *angelina8prop* A proportional, handwriting, 6 * 16 font • *novamono8pt* A modern 9 * 16 font • *unispace8pt* A modern 12 * 24 font • *unispaceitalic8pt* A modern italic (slanted) 12 * 24 font • *whiterabbit8pt* A modern 12 * 24 font • *angelina12prop* A proportional, handwriting, 8 * 24 font • *novamono12pt* A modern 13 * 34 font • *repetitionscrolling12pt* A modern 13 * 24 font • *unispace12pt* A modern 13 * 18 font • *unispaceitalic12pt* A modern italic (slanted) 13 * 18 font • *whiterabbit12pt* A modern 13 * 18 font • *amerikasans16pt* A proportional 17 * 20 font • *whiterabbit16pt* A modern 16 * 20 font • *whiterabbit18pt* A modern 18 * 22 font • *whiterabbit20pt* A modern 20 * 24 font

`<trigger>,l,<x1>,<y1>,<x2>,<y2>,<color>`	Draw a simple line between point x1,y1 and x2,y2, using the (required) color, 1 pixel wide.
`<trigger>,lh,<y>,<width>,<color>`	Draw a horizontal line (width = Line 'width' in pixels (positive = right of first point, negative = left of first point).
`<trigger>,lv,<x>,<height>,<color>`	Draw a vertical line (height= Line 'height' in pixels (positive = below first point, negative = above first point).
`<trigger>,r,<x>,<y>,<width>,<height>,<color>`	Draw a rectangle.
`<trigger>,rf,<x>,<y>,<width>,<height>,<bordercolor>,<innercolor>`	Draw a filled rectangle.
`<trigger>,c,<x>,<y>,<radius>,<color>`	Draw a circle.
`<trigger>,cf,<x>,<y>,<radius>,<bordercolor>,<innercolor>`	Draw a filled circle.
`<trigger>,t,<x1>,<y1>,<x2>,<y2>,<x3>,<y3>,<color>`	Draw a triangle.
`<trigger>,tf,<x1>,<y1>,<x2>,<y2>,<x3>,<y3>,<bordercolor>,<innercolor>`	Draw a filled triangle.
`<trigger>,rr,<x>,<y>,<width>,<height>,<corner_radius>,<color>`	Draw a rounded rectangle.
`<trigger>,rrf,<x>,<y>,<width>,<height>,<corner_radius>,<bordercolor>,<innercolor>`	Draw a filled rounded rectangle.
`<trigger>,px,<x>,<y>,<color>`	Print a single pixel.
`<trigger>,pxh,<x>,<y>,<color1>[,<color2>,...]`	Print a horizontal chain of colored pixels (left to right, starting at position x,y). Just supply as many colors as pixels that need to be painted.
`<trigger>,pxv,<x>,<y>,<color1>[,<color2>,...]`	Print a vertical chain of colored pixels (top to bottom, starting at position x,y). Just supply as many colors as pixels that need to be painted.

Change log

Changed in version 2.0: …

added   2021-08 Moved from an external forum to ESPEasy.