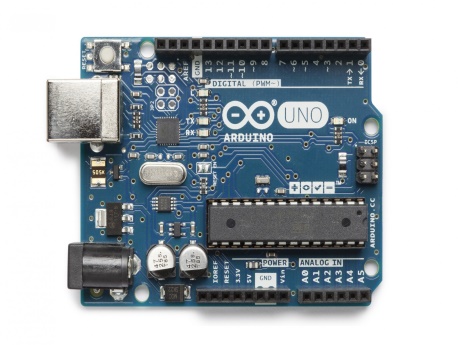
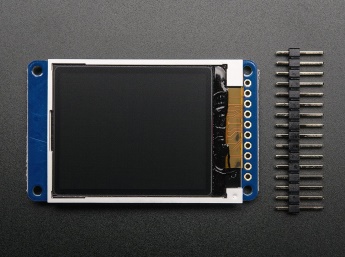
In this project, you needed these parts (***Dear visitors. You can support our project buy clicking on the links of parts and buying them or donate us to keep this website alive. Thank you***):

1.[Arduino Uno R3](http://s.click.aliexpress.com/e/jm2vJaM) (you can also use the other version of Arduino)



2. [1.8" SPI TFT LCD](http://s.click.aliexpress.com/e/ZzVjMbU)128x160 module (there are quite a lot of different modules but all of them have the same working principle) 1pc



3.Arduino IDE ( you can download it from [here](https://www.arduino.cc/en/Main/Software) )

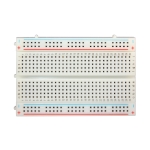
4.[Jumper cables](http://s.click.aliexpress.com/e/NjQJayJ) F-M, M-M



5.[Resistor](http://s.click.aliexpress.com/e/Eyf2Vju) 5 pcs (1 - 2KOhm 4 pcs, 40 - 470 Ohm 1 pc)



6.[Breadboard](http://s.click.aliexpress.com/e/7mAAe62) half size or small size 1 pc



*General*

We will learn how to connect 1.8" SPI TFT LCD 128x160 module to Arduino board and use it.

*Understanding the 1.8" SPI TFT LCD 128x160 module*

This lovely little display breakout is the best way to add a small, colorful and bright display to any project. Since the display uses 4-wire SPI to communicate and has its own pixel-addressable frame buffer, it can be used with every kind of microcontroller. Even a very small one with low memory and few pins available.

The 1.8" display has 128x160 color pixels. Unlike the low cost "Nokia 6110" and similar LCD displays, which are CSTN type and thus have poor color and slow refresh, this display is a true TFT! The TFT driver (ST7735R, ST7735S, ST7735B) can display full 18-bit color (262K shades). And the LCD will always come with the same driver chip so there's no worries that your code will not work from one to the other.

The breakout has the TFT display soldered on (it uses a delicate flex-circuit connector) as well as a ultra-low-dropout 3.3V regulator and a 3/5V level shifter so you can use it with 3.3V or 5V power and logic. We also had a little space so we placed a microSD card holder so you can easily load full color bitmaps from a FAT16/FAT32 formatted microSD card.

**Specifications:**

* 1.8" diagonal LCD TFT display
* 128x160 resolution, 18-bit (262,144) color
* 4 or 5 wire SPI digital interface
* Built-in microSD slot - uses 2 more digital lines
* 5V compatible! Use with 3.3V or 5V logic
* Onboard 3.3V @ 150mA LDO regulator
* 2 white LED backlight, transistor connected so you can PWM dim the backlight
* 1x10 header for easy breadboarding
* 4 x 0.9"/2mm mounting holes in corners
* Overall dimensions: 1.35" x 2.2" x 0.25" (34mm x 56mm x 6.5mm)
* Current draw is based on LED backlight usage: with full backlight draw is ~50mA

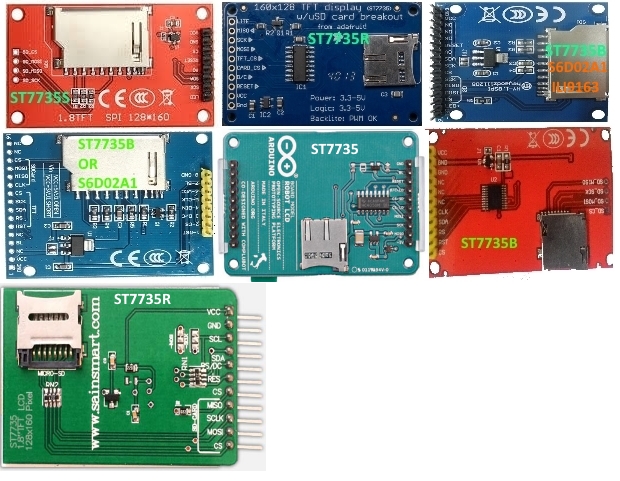
You can find the 1.8 TFT SPI 128\*160 V1.1 schematic and datasheet [here](http://thaieasyelec.com/downloads/EFDV366/Datasheet.zip). It is using driver ST7735S.

You can find datasheet of ST7735R [here](https://www.google.lt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0ahUKEwjhhJTa9NXYAhVHkSwKHfWvBBgQFggnMAA&url=https%3A%2F%2Fcdn-shop.adafruit.com%2Fdatasheets%2FST7735R_V0.2.pdf&usg=AOvVaw0XTiXEWpAgJdZOoJHr3HLn).

You can find specification and datasheet of ST7735B/ST7735 [here](https://forum.arduino.cc/index.php?action=dlattach;topic=424447.0;attach=180779) and [here](https://www.google.lt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=9&cad=rja&uact=8&ved=0ahUKEwiEyfbh9NXYAhWGESwKHVxfB-AQFghHMAg&url=http%3A%2F%2Fwww.displayfuture.com%2FDisplay%2Fdatasheet%2Fcontroller%2FST7735.pdf&usg=AOvVaw3c7tGIh_iby2ALaWludqqW).

You can find datasheet of S6D02A1 [here](https://www.google.lt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0ahUKEwjVvtP359XYAhVFiiwKHfc7DGUQFgg3MAI&url=http%3A%2F%2Fwww.vctec.co.kr%2Fweb%2Fproduct%2FBAE%2Fimg%2F103769%2FS6D02A1A01.pdf&usg=AOvVaw0wJUYxTJ64e2mhGWTBetjQ).

*Signals and connections of the 1.8" SPI TFT LCD 128x160 module*

**

This color display uses SPI to receive image data. That means you need at least 4 pins - CLOCK, DATA IN, TFT CS and D/C. If you'd like to have SD card usage too, add another 2 pins - DATA OUT and card CS.

Note: depending on the module you’re using, the pins may be in a different order.

**LITE** - this is the PWM input for the backlight control. Connect to 3-5VDC to turn on the backlight. Connect to ground to turn it off. Or, you can PWM at any frequency.

**MISO**(or **SD\_MISO**or**SDO**) (Master In Slave Out) - this is the SPI Master In Slave Out pin, its used for the SD card. It isn't used for the TFT display which is write-only

**SCLK** (or **SD\_SCK**or **SCK**or**CLK**or**SCL**) (Serial Clock) - The clock pulses which synchronize data transmission generated by the master. This is the SPI clock input pin.

**MOSI** (or **DIN** or **SD\_MOSI**or**SDA**) (Master Out Slave In) - this is the SPI Master Out Slave In pin, it is used to send data from the microcontroller to the SD card and/or TFT

**TFT\_CS** (Chip Select or Slave Select) - the pin on each device that the master can use to enable and disable specific devices. This is the TFT SPI chip select pin

**Card\_CS** (or **SD\_CS**) (Chip Select or Slave Select) - the pin on each device that the master can use to enable and disable specific devices. Used if you want to read from the SD card.

**D/C** (or **A0** or **DC**or**RS**) - this is the TFT SPI data or command selector pin

**RST** (or **RESET**or**RES**) - this is the TFT reset pin. Connect to ground to reset the TFT! Its best to have this pin controlled by the library so the display is reset cleanly, but you can also connect it to the Arduino Reset pin, which works for most cases.

**CS** (or **CE**or**SS**) (Chip Select or Slave Select) - the pin on each device that the master can use to enable and disable specific devices.

**VCC** - this is the power pin. Can be connected to +5VDC or +3.3VDC pin of Arduino board.

**GND** - ground. Connected to Arduino board GND pin.

**BL** (or **LED+**) - this is the input for the backlight control. Connect to 3.3V or 5V DC (with resistor 40-150 Ohm) to turn on the backlight.

**LED** - 3.3V IO and Power Supply pin

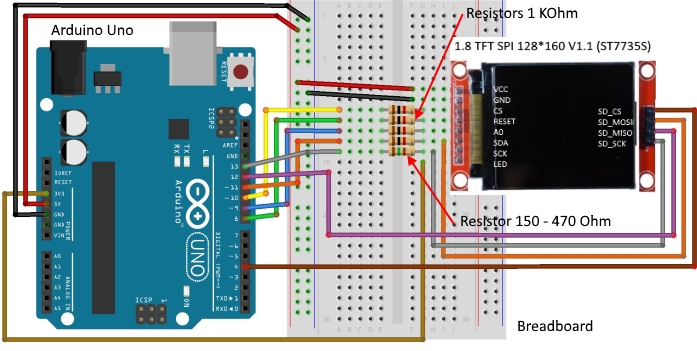
**LED-** - this is the backlight control ground pin. Connect to GND pin of Arduino board.

**NC**- Not connected. This pin is not in use.

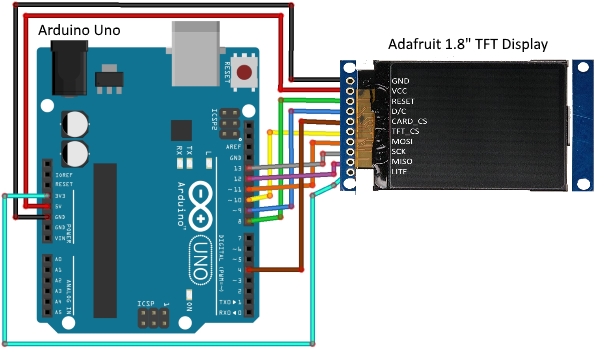
*Wiring*

There are two ways to wire up these displays - one is a more flexible method Software SPI (you can use any pins on the Arduino) and the other Hardware SPI is much faster (4-8x faster, but you are required to use the hardware SPI pins)

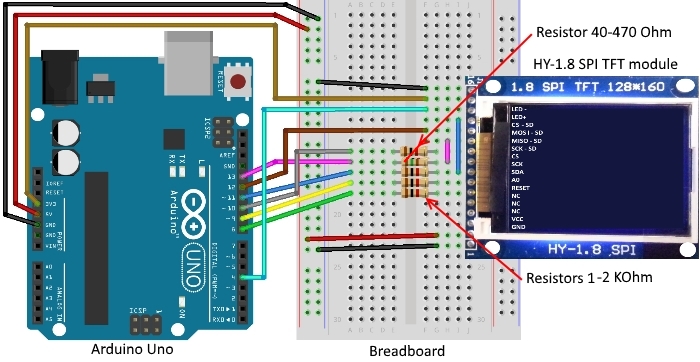
**1. 1.8" TFT SPI display 128\*160 v1.1 module with ST7735S IC**



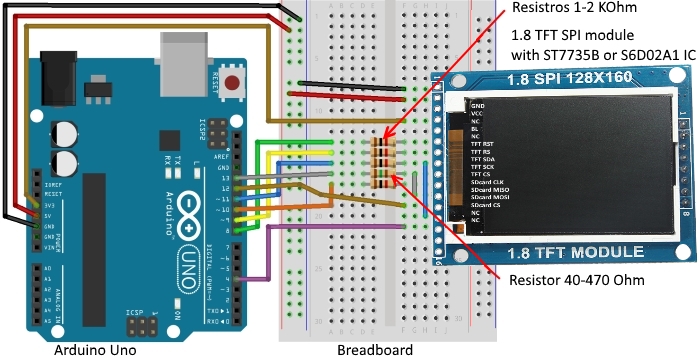
**2. Adafruit 1.8" TFT SPI display 128\*160 module with ST7735R IC**



**3. HY-1.8 SPI - 1.8" TFT SPI display 128\*160 module with ST7735B IC**

****

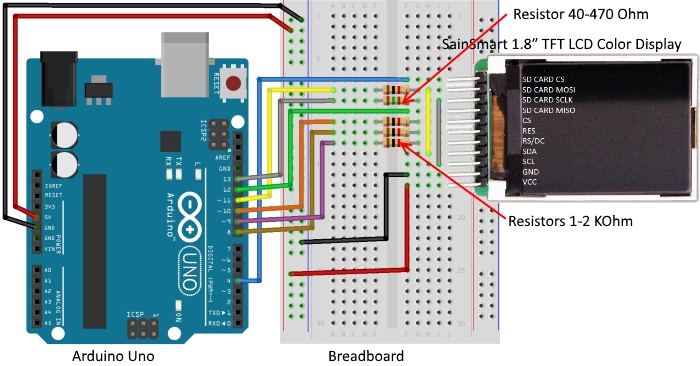
**4. 1.8" TFT SPI display 128\*160 module with ST7735B IC or S6D02A1 IC**



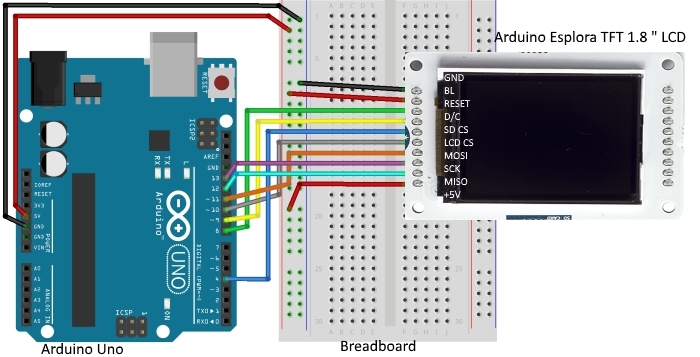
**5. KMR-1.8 SPI 1.8" TFT SPI display 128\*160 module with ST7735R IC or S6D02A1 IC or ILI9163 IC or ST7735S IC**

Wiring is the same as for HY-1.8 SPI - see above.

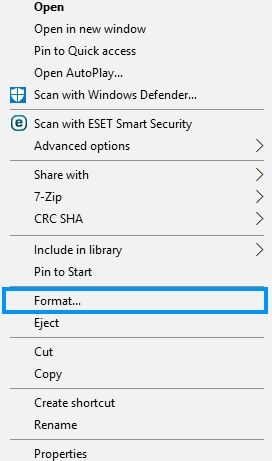
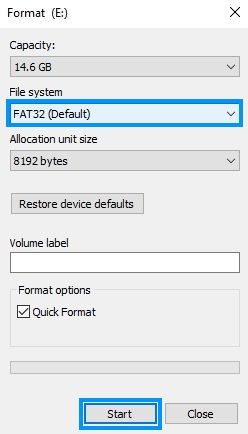
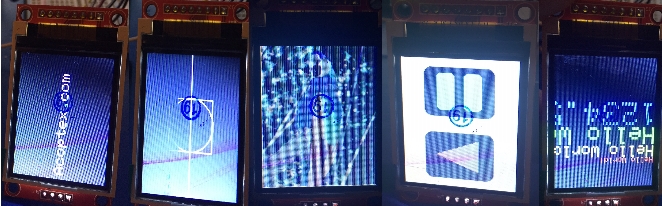
**6. SainSmart 1.8″ Color TFT LCD Display with ST7735R IC**

****

**7. Arduino Esplora 1.8″ Color TFT LCD Display with ST7735R IC**

****

*Step by Step instruction*

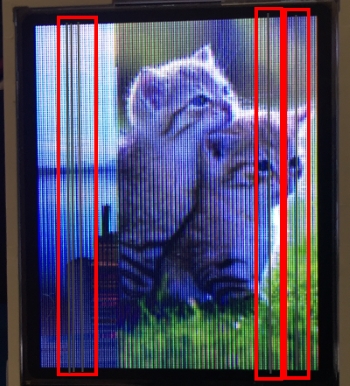
1. Solder four header pins for the SD card.
2. Most SD cards work right out of the box, but it's possible you have one that was used in a computer or camera and it cannot be read by the SD library. Formatting the card will create a file system that the Arduino can read and write to. It's not desirable to format SD cards frequently, as it shortens their life span. You’ll need a SD reader and computer to format your card. The library supports the FAT16 and FAT32 filesystems, but use FAT16 when possible. See additional info [here](https://www.arduino.cc/en/Reference/SDCardNotes).
3. Format the SD card as**FAT16** or **FAT32**. Insert the SD card in your computer. Go to My Computer and right click on the SD card. Select **Format...**
4. A new window pops up. Select **FAT32**, press Start to initialize the formatting process and follow the onscreen instructions.
5. Copy the image to the SD card or Micro SD card and insert it on the SD card/Micro SD card slot at the back of the display.
6. 
7. Do wiring. Both the display and the SD card work with SPI communication, so you’ll have pins on the Arduino with two connections.
8. Open Arduino IDE.
9. Plug your Adruino Uno board into your PC and select the correct board and com port
10. Verify and upload the sketch to your Adruino Uno.
11. If you are not sure which LCD driver your 1.8" TFT LCD has we suggest to contact your seller or try all sketches/libraries to find out.
12. 

*Summary*

We learnt how to connect 1.8" SPI TFT LCD 128x160 module to Arduino board and use it.

**Flickering streaks**

You have got one of these really cheap 1.8" TFT SPI LCD module. Great value for the price. You have got it up and running in no time. Just one little problem... flickering streaks - horizontal, vertical or combined. You did a search on the Web and can't find right answer. We will help you. It's very easy.



The root cause of what often described as flicker, to me it is more like flickering lines when loading data into display. Actualy, if you have more than one device on SPI - any SPI activity will cause that.

Most Arduino's running at 5V. Present display powered internally, including IOs, at 3.3v. Atmel's IOs drive current is pretty high. Display's inputs have clamps to VCC. Most signals idle high. When signals are 5V - current from signals through clamp diodes back feeding VCC and actually raising voltage on 3.3V regulator on display board to ~3.9V. During intense communications this voltage drops a bit and display's analog circuitry thrown a out of wack causing streaks on display.

So, how to eliminate it?

Solutions, options:

What I did, is to insert resistive dividers into each signal from Atmel to TFT. It consists of one in series of 180 Ohm and one 330 Ohm parallel to the inputs to ground, for each input.

 (you don't need to do anything to MISO, haha like soup)

Some more expensive versions of this display do have onboard level shifters, 5 to 3.3. You may prefer to add one like this.

Convert your Atmel board and display to 3.3V. Many ways to do that for Arduino. On display board you have to put solder blob across JP1.

There are some solutions:

1. Reduce the Arduino I/Os current with resistors (for SDA,SCL,CS,A0,RESET). You can see that on the first picture in Wiring section.
2. Insert resistive dividers into each signal from Arduino board to TFT display. It consists of one in series of 180 Ohm and one 330 Ohm parallel to the inputs to ground, for each input.
3. Convert your Arduino board and TFT display to 3.3V. Many ways to do that for Arduino. For example, on display board you have to put solder blob across JP1.

*Code*

1.TFTDisplayText sketch. Displays text in the middle of the screen.

2.TFTshapes sketch. Displays different shapes.

3.TFTBitmapLogo sketch. Displays your picture.The display can load images bigger or smaller than the display size (160 x 128 px), but for better results, edit your image size to 160 x 128 px.The image should be in **.bmp** format. To do that, you can use a photo editing software and save the image as**.bmp**format. If you want to later use your own image, use an image editing tool and crop your image to no larger than 128 pixels high and 160 pixels wide. Save it as a 24-bit color BMP file - it must be 24-bit color format to work, even if it was originally a 16-bit color image - becaue of the way BMPs are stored and displayed!You can download example [here](http://acoptex.com/uploads/parrot.bmp).

4. graphicstest sketch. Test of graphics.

5. rotationtest sketch. Test of rotation.

6.spitftbitmap sketch. The display can load images bigger or smaller than the display size (160 x 128 px), but for better results, edit your image size to 160 x 128 px.The image should be in **.bmp** format. To do that, you can use a photo editing software and save the image as**.bmp**format. If you want to later use your own image, use an image editing tool and crop your image to no larger than 160 pixels high and 128 pixels wide. Save it as a 24-bit color BMP file - it must be 24-bit color format to work, even if it was originally a 16-bit color image - becaue of the way BMPs are stored and displayed!You can download example [here](http://learn.adafruit.com/system/assets/assets/000/010/143/original/parrot.bmp).

7. ST7735\_SD  sketch. Scrolls you pictures like Photo frame. The display can load images bigger or smaller than the display size (160 x 128 px), but for better results, edit your image size to 160 x 128 px.The image should be in **.bmp** format. To do that, you can use a photo editing software and save the image as**.bmp**format. If you want to later use your own image, use an image editing tool and crop your image to no larger than 160 pixels high and 128 pixels wide. Save it as a 24-bit color BMP file - it must be 24-bit color format to work, even if it was originally a 16-bit color image - becaue of the way BMPs are stored and displayed! You can change the rotation of pictures to landscape or vertical. You can find examples of pictures used [here](http://acoptex.com/uploads/picturesexamples18tftlcd.zip).

8. UTFT\_Demo\_160x128\_Serial sketch. Graphics test.

9. graphicstest\_QDTech sketch. Test of graphics for Adafruit\_QDTech library. Supports S6D02A1 chip

10.tft\_s6d02a1\_graphicstest sketch. Test of Graphics for TFT\_S6D02A1 ibrary. Library has quite a lot examples. Supports S6D02A1 chip

11. TFT\_graphicstest\_small sketch for TFT\_ILI9163 library. Supports ILI9163 chip. Do not forget to check the User\_Setup.h file configuration in library folder.