معرفی و راه‌اندازی ال‌سی‌دی لمسی با آردوینو

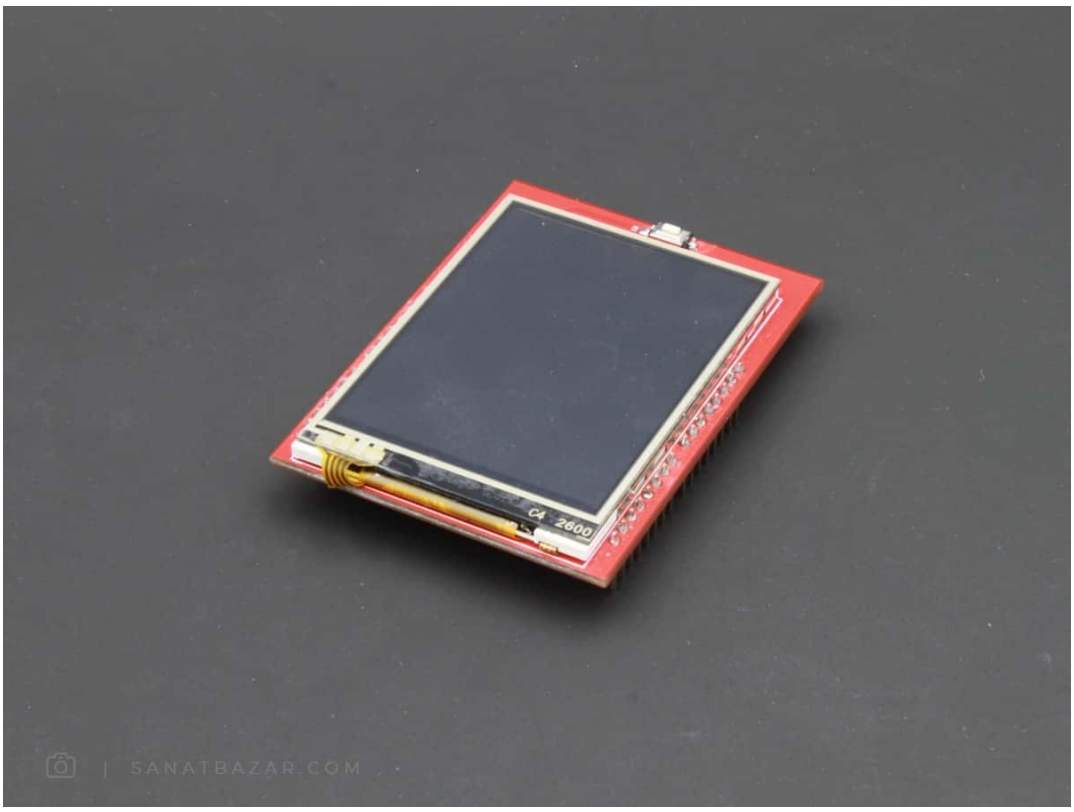
این مطلب بخش دهم از آموزش جامع آردوینو (مبتدی و پیشرفته) است. در این آموزش قصد داریم به معرفی و راه‌اندازی نمایشگر رنگی لمسی بپردازیم. اضافه کردن یک نمایشگر لمسی به پروژه‌تان می‌تواند جذابیت آن را چند برابر کند. به علاوه نمایشگر لمسی تمام مزایای یک نمایشگر مانند امکان مشاهده وضعیت سیستم و مزایای یک منبع ارسال فرمان به سیستم مانند دستور چرخیدن موتورها یا پخش صوت را یک جا دارد. در این آموزش علاوه بر کار با نمایشگر با مباحث بیشتری در کدنویسی و گرافیک کامپیوتر نیز آشنا خواهید شد. به دنیای نمایشگرها خوش آمدید!

قطعات مورد نیاز:

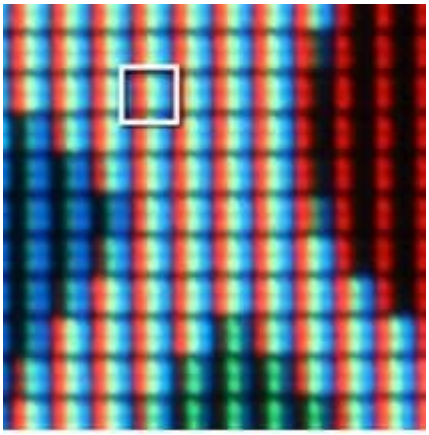
برد آردوینو و کابل رابط	1 عدد
شیلد نمایشگر لمسی "2.4	1 عدد

معرفی نمایشگر TFT

نمایشگر TFT LCD نوعی از نمایشگرهای کریستال مایع (LCD) است که در آن از یک لایه نازک ترانزیستور استفاده شده است. این تغییر باعث بهبود کارکرد و شفافیت تصویر آن شده است.

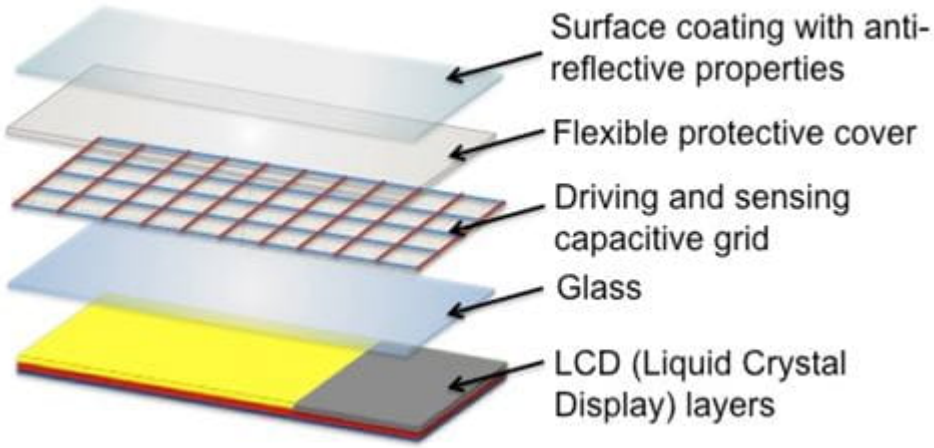


در LCD های معمولی آدرس‌دهی برای هر پیکسل به صورت مجزا صورت می‌گیرد. در صورتی که یک نمایشگر بزرگ با تعداد پیکسل‌های زیاد داشته باشید، این کار نیاز به میلیون‌ها اتصال الکترونیکی دارد که عملی نیست. در نمایشگر TFT از یک ماتریس سطر و ستون استفاده شده است و هر پیکسل به یک ترانزیستور متصل شده است. استفاده از این روش تعداد اتصالات را چند هزار بار کمتر می‌کند. هر پیکسل از سه المان نوری قرمز، سبز و آبی تشکیل شده است که ترکیب آنها رنگ دلخواه را تولید می‌کند.



معرفی نمایشگر لمسی

صفحه لمسی یک لایه حساس به تماس است که معمولا بر روی نمایشگرها نصب می‌شود و به عنوان ورودی عمل می‌کند. صفحه‌های لمسی در سال‌های گذشته پیشرفت چشمگیری داشته‌اند و فناوری‌های مختلفی برای ساخت آنها از جمله خازنی، مقاومتی و SAW ارائه شده است. زمانی که یک نقطه از صفحه لمس شود، یک سیگنال حاوی مختصات آن نقطه ارسال می‌گردد. امروزه از نمایشگرهای لمسی در موارد بسیاری از جمله گوشی‌های هوشمند، رایانه‌های شخصی و غیره استفاده می‌شود.



نحوه کار با نمایشگر لمسی

بسیاری از ماژول‌های الکترونیکی که با آردوینو استفاده می‌شوند، به دو صورت مجزا و شیلد ارائه شده‌اند. شیلدها بردهای الکترونیکی هستند که یک یا چند ماژول مختلف بر روی آن قرار دارد و با استفاده از تعدادی پین مشخص به یک برد کنترلی مانند آردوینو UNO وصل می‌شوند. مزیت شیلدها در این است که نیاز به اتصالات زیاد قطعات مختلف را از بین می‌برد و کافیست که آنرا مستقیما به آردوینو متصل کنید. علاوه بر این، استفاده از شیلد حجم اشغال شده توسط دستگاه‌تان را کم می‌کند. راه‌اندازی شیلدها معمولا تفاوتی با ماژول‌های مجزا ندارد و همان کد را بر روی آردوینو آپلود می‌کنید. در این آموزش از شیلد ماژول نمایشگر TFT لمسی استفاده شده است.

همان‌طور که می‌دانید آردوینو یک برد توسعه‌ای است که امکان راه‌اندازی بسیاری از تجهیزات الکتریکی و الکترونیک را به ساده‌ترین و سریع‌ترین شکل ممکن فراهم می‌کند. با این وجود نباید فراموش کنید که قدرت پردازشی و حافظه آردوینو محدود بوده و برای انجام کارهای سنگین محاسباتی چندان مناسب نیست. پردازش تصویر یکی از مواردی است که معمولا نیاز به قدرت محاسباتی و حافظه RAM زیادی دارد، بنابراین اگر می‌خواهید کار گرافیکی نسبتا سنگینی انجام دهید بهتر است به سراغ بردهای قوی‌ترین مانند رزبری پای بروید. با وجود این محدودیت‌ها به لطف کتابخانه‌های موجود و طراحی‌های خاص تراشه‌های موجود در نمایشگرها، آردوینو می‌تواند تصاویر نسبتا قابل قبولی تولید کند که برای خیلی از پروژه‌ها کافی است.

در این آموزش از شیلد نمایشگر لمسی TFT که مناسب آردوینو UNO ساخت شرکت MCUFriend استفاده شده است. برای کار با این شیلد نیاز به استفاده از سه کتابخانه وجود دارد. البته خودتان هم می‌توانید کدهای لازم برای استفاده از نمایشگر را بنویسید اما این کار نیاز به زمان بسیار زیاد برای نوشتن چند صد یا چند هزار خط دستور و داشتن دانش بیشتر در زمینه برنامه‌نویسی و گرافیک کامپیوتر دارد. بنابراین اگر کار اصلی شما برنامه‌نویسی برای سخت‌افزار نیست، خیلی راحت از خیر نوشتن برنامه بدون کتابخانه گذشته و از کتابخانه‌های موجود استفاده کنید!

کتابخانه Adafruit_GFX

اولین و مهمترین کتابخانه، Adafruit_GFX است. این کتابخانه معروف‌ترین کتابخانه آردوینو برای ایجاد تصاویر و نمایش خروجی گرافیکی است. این کتابخانه دارای مجموعه وسیعی از توابع گرافیکی بوده و با بسیاری از نمایشگرهای LCD، OLED و TFT همخوانی دارد. نصب کتابخانه با روشی که در آموزش دوم آردوینو توضیح داده شد به سادگی انجام می‌شود.

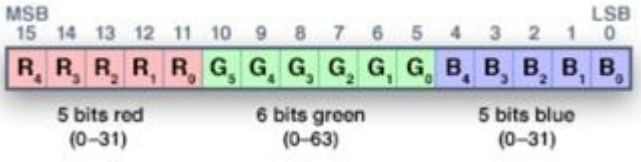
دانلود کتابخانه Adafruit_GFX

نمایشگر LED از ماتریسی از واحدهای بسیار کوچک نوری به نام پیکسل تشکیل شده است. در این صفحه هر پیکسل با مختصاتش مشخص می‌شود. مختصات پیکسل‌ها در یک دستگاه کارتزین (X, Y) تعریف شده است. محورهای این دستگاه بر خلاف حالت مرسوم، به سمت راست و پائین بوده و مبدا آن گوشه بالا و سمت چپ نمایشگر است. این یک روش متداول بوده و در اغلب سیستم‌های گرافیک کامپیوتر از آن استفاده می‌شود. برای اینکه جهت نمایش صفحه (عمودی یا افقی) تغییر کند باید این دستگاه را بچرخانید. اگر نقطه مبدا دستگاه مختصات در هر کدام از چهار گوشه صفحه قرار داشته باشد، یک جهت نمایش متفاوت ایجاد خواهد شد. چرخاندن صفحه نمایش به طور خاص وقتی کاربرد دارد که مجبور باشید نمایشگر را در جهت خاصی در پروژه‌تان نصب کنید.
طبق یک دسته‌بندی، می‌توان نمایشگرها را به سه گروه تقسیم کرد:

1. نمایشگر سیاه و سفید (Bitmap)؛ در این حالت هر پیکسل تنها دارای دو حالت سیاه(۰) و سفید(۱) است. نمونه یک عکس سیاه و سفید:

2. نمایشگر سیاه و سفید با طیف خاکستری (Grayscale)؛ در این نوع نمایشگرها، رنگ سیاه، سفید و مجموعه‌ای از رنگ‌های خاکستری بین سیاه و سفید قابل تولید است. هرچه تعداد بیت‌های بیشتری برای نمایش تصویر استفاده شود، تعداد رنگ‌های بیشتری نیز بین سیاه و سفید قابل تفکیک و ارائه خواهد بود. به تعداد بیت‌های استفاده شده برای یک رنگ، عمق رنگ (Color depth) گفته می‌شود. معمولا برای نمایش طیف رنگ از ۸ بیت (یک بایت) داده استفاده می‌شود که تفکیک 256 رنگ مجزا را ممکن می‌سازد. نمونه‌ای از یک تصویر سیاه و سفید با طیف خاکستری:

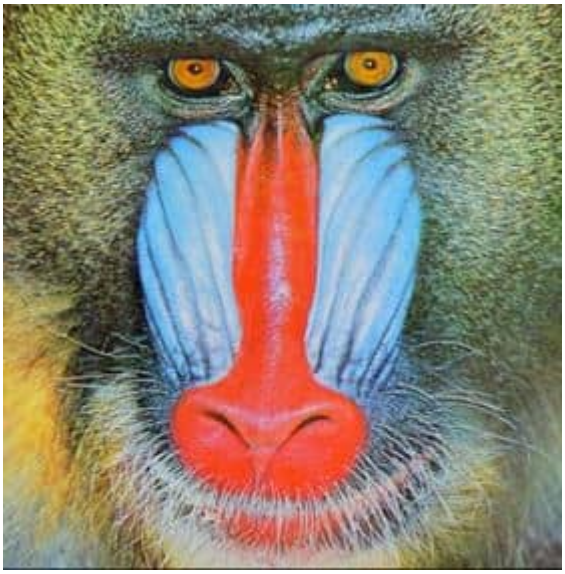
3. نمایشگرهای رنگی (Color)؛ در این نمایشگرها با استفاده از ترکیب چند رنگ پایه مانند قرمز (Red)، سبز (Green) و آبی (Blue) سایر رنگ‌ها ساخته می‌شود. این سه رنگ مبنای استاندارد نمایش RGB است که معمولا در سیستم‌های کامپیوتری استفاده می‌شود. اینکه برای هر کدام از این سه رنگ چه مقدار فضای داده در نظر گرفته شود متغیر است و هرچه بیشتر باشد، رنگ‌های بیشتر و زنده‌تری تولید شده و در نتیجه تصویر نهایی می‌تواند کیفیت بالاتری داشته باشد. البته کیفیت تصویر به عوامل دیگری مانند تراکم پیکسل‌ها در واحد سطح نیز وابسته است. معمول‌ترین سیستم ایجاد تصاویر، استفاده از ۸ بیت برای هر رنگ (Channel) است که مجموعا ۲۴ بیت برای ذخیره سازی هر پیکسل اشغال می‌شود. به این سیستم، bit-24 یا True Color گفته می‌شود و در آن مجموعا حدود ۱۶ میلیون رنگ متفاوت قابل تولید است. در تجهیزاتی که با محدودیت ذخیره‌سازی، پردازش یا انتقال اطلاعات مواجه‌اند یا نمایشگر کیفیت لازم برای نمایش این تعداد رنگ را ندارد، از سیستم bit-16 استفاده می‌شود. در این روش رنگ هر پیکسل در یک مجموعه ۱۶ بیتی ثبت شده که در آن ۵ بیت سمت چپ مربوط به رنگ قرمز، ٦ بیت وسط رنگ سبز و ۵ بیت سمت راست رنگ آبی است.



با اینکه در این حالت فقط حدود ٦۵ هزار رنگ قابل تولید است اما برای بسیاری از کاربردها کافی است. مخصوصا در آردوینو که محدودیت فضای RAM و قدرت پردازشی دارد، این موضوع مزیت بسیار مهمی به شمار می‌آید. کتابخانه Adafruit_GFX و نیز شیلد نمایشگر از سیستم bit-16 استفاده می‌کنند. نمونه یک عکس

 |

رنگی ۲۴ بیتی:



توابع اصلی کتابخانه Adafruit_GFX

در این قسمت بعضی از توابع مهم در این کتابخانه معرفی می‌شوند. پیش از آن باید با یک موضوع دیگر آشنا شوید. همان‌طور که می‌دانید انواع مختلفی از متغیرها وجود دارد، مانند عدد صحیح (Int)، عدد اعشاری (float) و متغیر منطقی (bool). هر کدام از این متغیرها نحوه تعریف و معنای مشخصی دارند اما حجم ذخیره‌سازی آنها می‌تواند به گونه‌های مختلفی در نظر گرفته شود؛ مثلا متغیر int می‌تواند به صورت ۸، ۱۶، ۳۲ و ۶۴ بیتی ذخیره شود. در این صورت برای تعریف متغیر به جایintاز عبارت‌هایی مانندint16_t،int8_tو استفاده می‌شود. همچنین متغیرها می‌توانند فقط شامل اعداد مثبت باشند که در این صورت بزرگترین عدد قابل تعریف برای آن متغیر با همان حجم ذخیره‌سازی، ۲ برابر خواهد شد. در این صورت تعریف متغیر مانند موارد زیر صورت می‌گیرد:

```
unsigned int Variable = number;
uint16_t Variable = number;
```

بنابراین در ورود اطلاعات به توابع، به نوع هر ورودی دقت کنید. توابع اصلی کتابخانه Adafruit_GFX شامل موارد زیر است. در تعاریف زیر نوع هر متغیر نیز مشخص شده است.

نمایش پیکسل:

```
drawPixel(uint16_t x, uint16_t y, uint16_t color);
```

متغیر اول مقدار x، متغیر دوم مقدار y و متغیر سوم رنگ پیکسل به صورت bit-16 است.

نمایش خط:

```
drawLine(uint16_t x1, uint16_t y1, uint16_t x2, uint16_t y2, uint16_t color);
```

متغیر اول و دوم مختصات نقطه شروع و متغیر سوم و چهارم مختصات نقطه انتهای خط و متغیر پنجم رنگ خط است.

نمایش خط افقی و عمودی:

```
drawFastVLine(uint16_t x1, uint16_t y1, uint16_t length, uint16_t color);
drawFastHLine(uint8_t x1, uint8_t y1, uint8_t length, uint16_t color);
```

متغیر اول و دوم مختصات نقطه شروع، متغیر سوم طول خط و متغیر چهارم رنگ آن است.

نمایش مستطیل:


```
drawRect(uint16_t x1, uint16_t y1, uint16_t w, uint16_t h, uint16_t color);
fillRect(uint16_t x1, uint16_t y1, uint16_t w, uint16_t h, uint16_t color);
```

تابع اول خط دور مستطیل و تابع دوم داخل آن را رسم می‌کند. در این توابع، متغیر اول و دوم مختصات گوشه بالا و سمت چپ مستطیل، متغیر سوم و چهارم طول و عرض مستطیل و متغیر پنجم رنگ آن است.

نمایش دایره:

```
drawCircle(uint16_t x1, uint16_t y1, uint16_t r, uint16_t color);
fillCircle(uint16_t x1, uint16_t y1, uint16_t r, uint16_t color);
```

رسم دایره مانند مستطیل است و ورودی آن مختصات مرکز دایره، شعاع و رنگ آن است.

نمایش متن:
برای نوشتن متن، باید چند خط کد اجرا شود. ابتدا باید محل شروع متن (گوشه سمت چپ و بالای متن) با دستورsetCursorتنظیم شود، سپس رنگ متن با دستورsetTextColorو اندازه آن با دستورsetTextSizeتعیین می‌شود. همچنین در صورتی که متن طولانی‌تر از صفحه نمایش باشد، ادامه آن در سطر بعد نشان داده خواهد شد؛ برای غیرفعال کردن این ویژگی از دستورsetTextWrap(false)استفاده کنید. توابع گفته شده به این صورت تعریف می‌شود:

```
setTextCursor(unint16_t x1, uint16_t y1);
setTextColor(uint16_t color);
setTextSize(uint8_t size);
setTextWrap(bool w);
```

تنظیم صفحه نمایش:
دستور زیر تمام صفحه را پاک کرده و به رنگ دلخواه در می‌آورد:

```
fillScreen(uint16_t color);
```

و دستور زیر صفحه نمایش را به میزان دلخواه (یکی از موارد ۹۰، ۱۸۰ یا ۲۷۰ درجه) می‌چرخاند.

```
setRotation(uint8_t angle);
```

دقت کنید که این دستور به چیزهایی که تاکنون نمایش داده‌اید اعمال نمی‌شود. همچنین توجه کنید که با چرخاندن صفحه، نقطه مبدا شما نیز تغییر می‌کند و در سایر دستورات باید این را در نظر داشته باشید. این دستور معمولا یک بار و در حلقه setup اجرا می‌شود.

دستور زیر نیز طول و عرض صفحه نمایش را برحسب پیکسل می‌دهد:

```
width();
height();
```

برای سادگی کار، در این کتابخانه یک کلاس به نامAdafruit_GFX_Buttonتعریف شده که با آن به راحتی می‌توانید یک کلید بر روی صفحه نمایش ایجاد کنید. در این کلاس دستورinitButtonکلید را در محل مورد نظر با رنگ دلخواه ایجاد کرده و یک متن روی آن می‌نویسد؛ دستورdrawButtonرنگ داخل کلید را پاک می‌کند؛ در صورتی که نقطه (x, y) درون کلید قرار داشته باشد دستورcontains(x, y)مقدار true را می‌دهد و برعکس؛ دستور press وضعیت کلید (لمس شده یا لمس نشده) را بررسی می‌کند؛ دستوراتjustPressed،isPressedوjustReleasedبه ترتیب در صورتی که کلید نگه‌داشته شده باشد، کلید به تازگی لمس شده باشد و کلید به تازگی رها شده باشد مقدار true را می‌دهند.

کتابخانه TouchScreen

کتابخانه دوم مورد استفاده، TouchScreen است. این کتابخانه برای خواندن داده‌های لمسی صفحه استفاده می‌شود. دستورات مهم این کتابخانه عبارتند از:

دریافت نقطه لمس‌شده:
تعیین نقطه لمس‌شده توسط دستور()getPointانجام می‌شود. برای استفاده از این دستور باید یک Structure از جنس TSPoint تعریف کنید. Structure ها شیء‌هایی هستند که می‌توانند چند متغیر داخلی داشته باشند. در این مورد، متغیرهای x, y مختصات نقطه لمس‌شده و z میزان فشار است. برای تعریف و دسترسی به Structure مانند زیر عمل کنید:

```
TSPoint touchPoint = touch.getPoint();
X = touchPoint.x;
Y = touchPoint.y;
Z = touchPoint.z;
```

دانلود کتابخانه Touchscreen

کتابخانه MCUFRIEND_kbv

این کتابخانه، یک کتابخانه مخصوص شیلد نمایشگر مورد استفاده در این آموزش است؛ معمولا هر نمایشگر، تعاریف خاص خود را داشته و کتابخانه مخصوص به خود را نیز دارد. پس اگر با نمایشگر دیگری کار می‌کنید از کتابخانه مخصوص آن استفاده کنید.

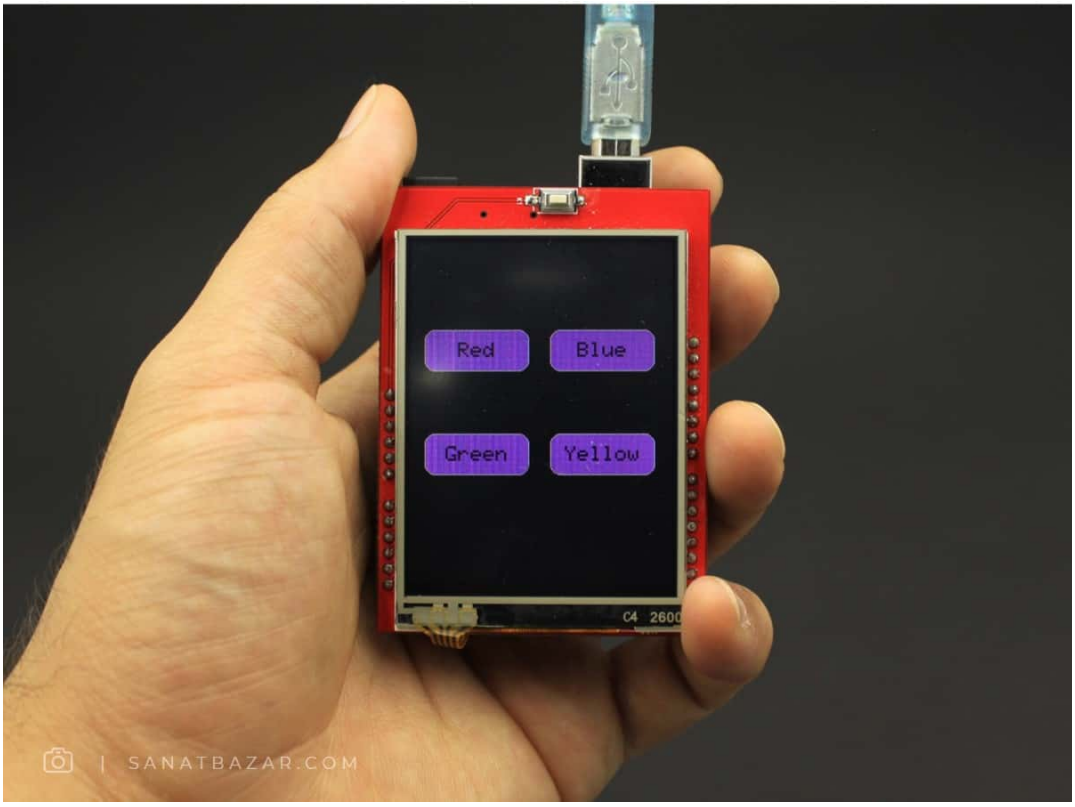
این کتابخانه عمدتا بر اساس همان کتابخانه Adafruit_GFX نوشته شده است و دستورات آن را می‌شناسد. دو دستور مهم مخصوص این کتابخانه به صورت زیر است:
برای تشخیص نوع نمایشگر از دستور()readIDو برای برقراری ارتباط با آن از دستورbeginاستفاده می‌شود. استفاده از این دستورات نیز مانند صفحه لمسی به صورت تعریف Structure است:

```
TFTID = TFT.readID();
TFT.begin(TFTID);
```

دانلود کتابخانه MCUFriend

شروع کار با نمایشگر لمسی

حالا وقت این است که اولین پروژه خودتان با نمایشگر لمسی را بنویسید. تعداد دستورات مورد استفاده در این وسیله زیاد است و ممکن است که کار با آنها را فراموش کنید؛ اما جای نگرانی نیست، با نوشتن اولین برنامه بسیاری از نکات را عملا یاد می‌گیرید. یادتان باشد که هرچه بیشتر با ماژولی کار کنید و چیزهای مختلف را با آن امتحان کنید، بیشتر به آن مسلط می‌شوید.
پیش از اینکه برنامه ما را نگاه کنید، با چیزهایی که تا الان یاد گرفتید سعی کنید خودتان برنامه‌ای بنویسید که یک صفحه سیاه را به همراه چهار کلید لمسی نمایش دهد. با لمس هر کدام از کلیدها پس‌زمینه نمایشگر به یک رنگ خاص در بیاید.



برنامه زیر این کار را انجام می‌دهد:

```
/*
SanatBazar
Arduino Tutorial Series
Author: Davood Dorostkar
Website: www.sanatbazar.com

*/

// TFT screen definitions
#include <Adafruit_GFX.h>
#include <MCUFRIEND_kbv.h>
#include <TouchScreen.h>
MCUFRIEND_kbv TFT;
Adafruit_GFX_Button Button[4]; //  redButton, blueButton, greenButton, yellowButton;

#define BLACK 0x0000
#define BLUE 0x001F
#define RED 0xF800
#define GREEN 0x07E0
#define CYAN 0x07FF
#define MAGENTA 0xF81F
#define YELLOW 0xFFE0
#define WHITE 0xFFFF
uint16_t color[4] = {RED, BLUE, GREEN, YELLOW};

// Touch definitions
#define minPressure 200
#define maxPressure 1000
const int XP = 6, XM = A2, YP = A1, YM = 7;
const int touchLeftLimit = 107, touchRightLimit = 910, touchUpLimit = 954,
touchButtomLimit = 100;
TouchScreen touch = TouchScreen(XP, YP, XM, YM, 300);
int touchX, touchY;

// Functions
bool touchSensed()
{
      TSPoint touchPoint = touch.getPoint();
      pinMode(YP, OUTPUT);
      pinMode(XM, OUTPUT);
      digitalWrite(YP, HIGH);
      digitalWrite(XM, HIGH);
      bool pressed = (touchPoint.z > minPressure && touchPoint.z < maxPressure);
```



```
      if (pressed)
      {
            touchX = map(touchPoint.x, touchLeftLimit, touchRightLimit, 0, TFT.width());
            touchY = map(touchPoint.y, touchUpLimit, touchButtomLimit, 0, TFT.height());
      }
      return pressed;
}

void makeIcons()
{
      Button[0].initButton(&TFT, 60, 100, 100, 40, WHITE, MAGENTA, BLACK, "Red", 2);
      Button[1].initButton(&TFT, 180, 100, 100, 40, WHITE, MAGENTA, BLACK, "Blue", 2);
      Button[2].initButton(&TFT, 60, 200, 100, 40, WHITE, MAGENTA, BLACK, "Green", 2);
      Button[3].initButton(&TFT, 180, 200, 100, 40, WHITE, MAGENTA, BLACK, "Yellow",
2);
      Button[0].drawButton();
      Button[1].drawButton();
      Button[2].drawButton();
      Button[3].drawButton();
}

void initStartPage()
{
      uint16_t TFTID = TFT.readID();
      TFT.begin(TFTID);
      TFT.fillScreen(BLACK);
      TFT.setRotation(0);
      TFT.setCursor(15, 140);
      TFT.setTextColor(RED);
      TFT.setTextSize(2);
      TFT.setTextWrap(false);
      TFT.print("www.sanatbazar.com");
}

void setBackColor(int color)
{
      TFT.fillScreen(color);
      makeIcons();
}

void checkButtonPressed()
{
      for (int i = 0; i < 4; i++)
      {
            Button[i].press(touchSensed() && Button[i].contains(touchX, touchY));
            if (Button[i].justReleased())
            {
                  Button[i].drawButton();
            }
            if (Button[i].justPressed())
            {
                  Button[i].drawButton(true);
                  setBackColor(color[i]);
            }
      }
}

void setup()
{
      initStartPage();
      delay(2000);
      TFT.fillScreen(BLACK);
      makeIcons();
}

void loop()
{
      checkButtonPressed();
}
```

در این برنامه ابتدا کتابخانه‌های ضروری فراخوانی شده و چند رنگ اصلی مورد استفاده تعریف شده‌اند. چهار کلید و رنگ‌های هر کدام به صورت آرایه‌ای از اشیا و رنگ‌ها تعریف شده‌اند. این کار باعث می‌شود بتوانیم به راحتی به آنها آدرس‌دهی کنیم و روند کدنویسی را راحت‌تر می‌کند:

```
Adafruit_GFX_Button Button[4]; // redButton, blueButton, greenButton, yellowButton;
```

```
uint16_t color[4] = {RED, BLUE, GREEN, YELLOW};
```

در ادامه تنظیمات صفحه لمسی قرار دارد. برای داشتن یک لمس دقیق باید محدوده قابل لمس برای برنامه تعریف شود. برای این کار ابتدا از مسیر File Examples MCUFRIEND_kbv مثال آماده TouchScreen_Calibr_native را اجرا کنید؛ نقاط مشخص شده روی نمایشگر را به ترتیب لمس کنید. در نهایت مقادیر چهار سمت نمایشگر نشان داده می‌شود. این مقادیر را در قسمت تنظیمات صفحه لمسی وارد کنید:

```
const int touchLeftLimit = 107, touchRightLimit = 910, touchUpLimit = 954,
touchButtomLimit = 100;
```

متغیرهای touchX و touchY مختصات نقطه لمس‌شده است که در ادامه مورد استفاده قرار می‌گیرد:

```
int touchX, touchY;
```

تابعtouchSensedنقطه لمس‌شده را ثبت کرده و در صورتی که میزان فشار آن مناسب باشد، ابتدا محل نقطه مورد نظر را با طول و عرض نمایشگر هماهنگ کرده و مقدار true را می‌دهد:

```
bool touchSensed()
{
      TSPoint touchPoint = touch.getPoint();
      pinMode(YP, OUTPUT);
      pinMode(XM, OUTPUT);
      digitalWrite(YP, HIGH);
      digitalWrite(XM, HIGH);
      bool pressed = (touchPoint.z > minPressure && touchPoint.z < maxPressure);
      if (pressed)
      {
            touchX = map(touchPoint.x, touchLeftLimit, touchRightLimit, 0, TFT.width());
            touchY = map(touchPoint.y, touchUpLimit, touchButtomLimit, 0, TFT.height());
      }
      return pressed;
}
```

تابعmakeIconsچهار کلیدی که قبلا تعریف شده بود را با مشخصات مورد نظر (اندازه، محل کلید، رنگ و عنوان) می‌سازد:

```
void makeIcons()
{
      Button[0].initButton(&TFT, 60, 100, 100, 40, WHITE, MAGENTA, BLACK, "Red", 2);
      Button[1].initButton(&TFT, 180, 100, 100, 40, WHITE, MAGENTA, BLACK, "Blue", 2);
      Button[2].initButton(&TFT, 60, 200, 100, 40, WHITE, MAGENTA, BLACK, "Green", 2);
      Button[3].initButton(&TFT, 180, 200, 100, 40, WHITE, MAGENTA, BLACK, "Yellow",
2);
      Button[0].drawButton();
      Button[1].drawButton();
      Button[2].drawButton();
      Button[3].drawButton();
}
```

تابعinitStartPageصفحه آغازین نمایشگر شامل آدرس سایت صنعت‌بازار با پس‌زمینه مشکی را نشان می‌دهد:

```
void initStartPage()
{
      uint16_t TFTID = TFT.readID();
      TFT.begin(TFTID);
      TFT.fillScreen(BLACK);
      TFT.setRotation(0);
      TFT.setCursor(15, 140);
      TFT.setTextColor(RED);
      TFT.setTextSize(2);
      TFT.setTextWrap(false);
      TFT.print("www.sanatbazar.com");
}
```

تابعsetBackColorیک رنگ را به عنوان ورودی گرفته و پس‌زمینه را به همان رنگ تغییر می‌دهد:

```
void setBackColor(int color)
{
      TFT.fillScreen(color);
      makeIcons();
}
```

تابعcheckButtonPressedتمام کلیدها را بررسی کرده و در صورتی که یکی از آنها لمس شده باشد، یک لحظه رنگ کلید را عوض کرده و رنگ پس‌زمینه را نیز تغییر می‌دهد. به محض رها کردن کلید، رنگ آن به حالت عادی بر می‌گردد. در این قسمت باید تشخیص دهیم که آیا کلیدی لمس شده یا نه. اگر جایی در صفحه نمایش لمس شده و همزمان مختصات آن درون یک کلید باشد، به معنی لمس آن کلید خواهد بود:

```
void checkButtonPressed()
{
      for (int i = 0; i < 4; i++)
      {
            Button[i].press(touchSensed() && Button[i].contains(touchX, touchY));
            if (Button[i].justReleased())
            {
                  Button[i].drawButton();
            }
            if (Button[i].justPressed())
            {
                  Button[i].drawButton(true);
                  setBackColor(color[i]);
            }
      }
}
```

برنامه اصلی به این صورت است که ابتدا صفحه آغاز نمایش داده شده و ۲ ثانیه مکث می‌کند، سپس کلیدها ساخته شده و در صورتی که یکی از آنها لمس شود، رنگ پس‌زمینه تغییر خواهد کرد.

```
void setup()
{
      initStartPage();
      delay(2000);
      TFT.fillScreen(BLACK);
      makeIcons();
}

void loop()
{
      checkButtonPressed();
}
```

نتیجه‌گیری

در این آموزش با مباحث پایه در تولید تصاویر گرافیکی در کامپیوتر و نحوه کار با نمایشگر لمسی TFT را یاد گرفتید. با نمایشگر لمسی می‌توانید پروژه‌هایتان را به وسیله‌های هوشمند تبدیل کنید.
در آموزش بعدی، نحوه راه‌اندازی و کار با موتور DC را خواهید آموخت.
نظرات شما باعث بهبود محتوای آموزشی ما می‌شود. اگر این آموزش را دوست داشتید، همین‌طور اگر سوالی در مورد آن دارید، از شنیدن نظراتتان خوشحال خواهیم شد.

