

Phụ lục

LỜI NÓI ĐẦU.....	1
I. HƯỚNG DẪN KẾT NỐI MÀN HÌNH HOMEPHONE VỚI LCD	3
II. CÁC HÀM KHỞI ĐỘNG VÀ KHAI BÁO SỬ DỤNG MÀN HÌNH	3
III. CÁC HÀM VẼ CƠ BẢN.....	4
• Vẽ một điểm ảnh.....	4
• Vẽ một đường thẳng.....	5
• Vẽ một hình vuông hay hình chữ nhật.....	5
• Tô màu hình vuông hay hình chữ nhật.....	5
• Vẽ hình tròn	5
• Tô màu hình tròn.....	5
• Vẽ hình tam giác	6
• Viết chữ.....	6
• Vẽ hình ảnh.....	6

LỜI NÓI ĐẦU.

Như chúng ta đã biết, để tận dụng màn hình trong các máy điện thoại homephone của viettel cũ hỏng, chúng ta có thể mua lại từ những hàng đồng nát giá rẻ (tính theo KG) nhưng phải rình mò mới có. Hoặc mua sẵn do các bạn đã cất công đi mua rồi bán lại (giá tầm 15 đến 20k/chiếc). và để sử dụng chúng tiện lợi hơn thì thư viện cho Arduino điều khiển homephone.

Trước hết xin giới thiệu đôi chút qua.

Thư viện này đã được chỉnh sửa một chút và được điều chỉnh cho phù hợp với màn hình homephone.

Tài liệu này được viết hoàn toàn bằng tiếng việt, các câu lệnh bằng tiếng anh và các chân kết nối chuẩn SPI bằng tiếng anh.

Trước mỗi câu lệnh sẽ là dấu “#”.

Thư viện này cũng được chỉnh sửa dựa theo thư viện GFX adafruit,

Một số thuật toán cũng được tham khảo và dựa theo để viết lại.

Sẽ không chịu trách nhiệm với bất kì sự hỏng hóc nào với thiết bị của bạn, tuy nhiên, thư viện đã được thử nhiều lần trong môi trường kiểm thử và không có hỏng hóc gì với các thiết bị,

Các hàm trong thư viện cũng được điều chỉnh gần giống so với thư viện điều khiển cho màn hình PCD8544 GFX cho Arduino.

Thư viện này bao gồm các hàm khởi động và vẽ được liệt kê dưới đây:

- *Khởi động màn hình,*
- *Cài đặt độ tương phản*
- *Hiển thị màn hình*
- *Xóa màn hình.*
- *Cài đặt con trỏ để chuẩn bị viết chữ*

- *Cài đặt*
- *Vẽ điểm ảnh,*
- *Vẽ đường thẳng*
- *Vẽ hình vuông*
- *Tô màu hình vuông*
- *Vẽ hình tròn,*
- *Tô màu hình tròn*
- *Vẽ hình tam giác*
- *Tô màu hình tam giác (đang phát triển thêm)*
- *Viết chữ*
- *Đặt màu chữ*
- *Vẽ bitmap (đang phát triển thêm)*

Thư viện này có sẽ:

- Lấy mất khoảng hơn 1KB cho việc buffer lcd.
- Sử dụng SPI mềm. thuật tiện hơn cho việc kết nối chân, không phải sử dụng chân cố định trên Arduino.
 - Ưu điểm: thuật tiện cho thiết kế
 - Nhược điểm: chậm hơn so với SPI cứng.
- Có khả năng cập nhật từng phần của màn hình. Để làm giảm thiểu sự chậm trễ của việc dùng SPI mềm, cập nhật màn hình từng phần giúp cho khả năng hiển thị của màn hình nhanh hơn nếu dùng trong hình động. Thay vì mỗi lần cập nhật lại toàn bộ màn hình, thư viện chỉ cập nhật một phần màn hình, những gì đã thay đổi.

I. HƯỚNG DẪN KẾT NỐI MÀN HÌNH HOMEPHONE VỚI LCD

Chúng ta có thể sử dụng 5 chân của Arduino kết nối với màn hình homophone.

Ví dụ ở đây sẽ là

9 *sdin*

8 *slk*

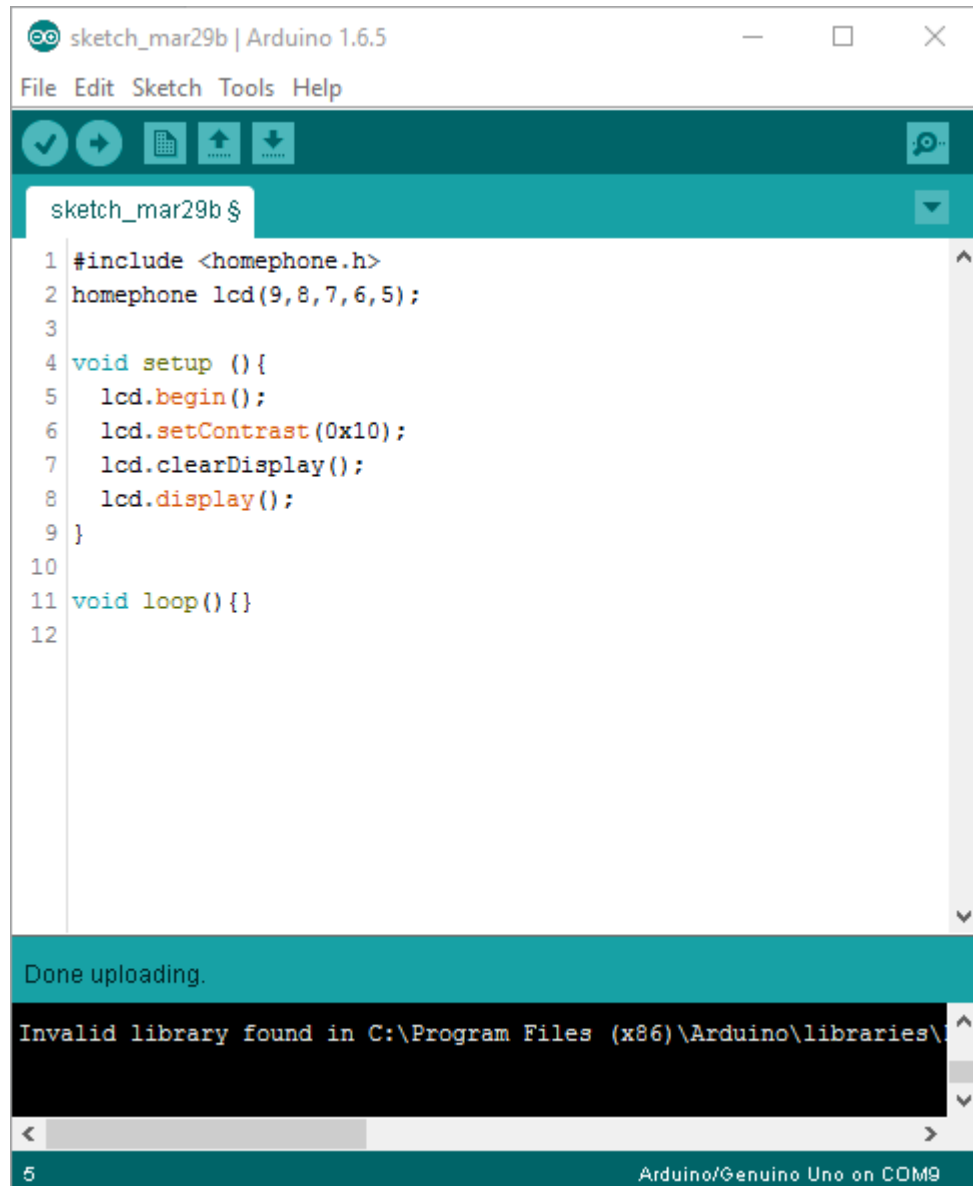
7 *a*

6 *reset*

5 *CS*

Cũng có thể kết nối bằng cách sử dụng các chân khác trên Arduino để kết nối đến màn hình.

II. CÁC HÀM KHỞI ĐỘNG VÀ KHAI BÁO SỬ DỤNG MÀN HÌNH



```
sketch_mar29b | Arduino 1.6.5
File Edit Sketch Tools Help

sketch_mar29b $
1 #include <homophone.h>
2 homophone lcd(9,8,7,6,5);
3
4 void setup () {
5   lcd.begin();
6   lcd.setContrast(0x10);
7   lcd.clearDisplay();
8   lcd.display();
9 }
10
11 void loop() {}
12

Done uploading.
Invalid library found in C:\Program Files (x86)\Arduino\libraries\

5 Arduino/Genuino Uno on COM9
```

Dòng 1: thêm thư viện

Dòng 2: chỉ định chân của Arduino kết nối đến homophone lcd như theo thứ tự đã nêu ở mục trước.

Tiếp đó vào chương trình chạy một lần

Dòng 5: lcd.begin khởi động lcd.

Dòng 6: cài đặt độ tương phản cho màn hình lcd

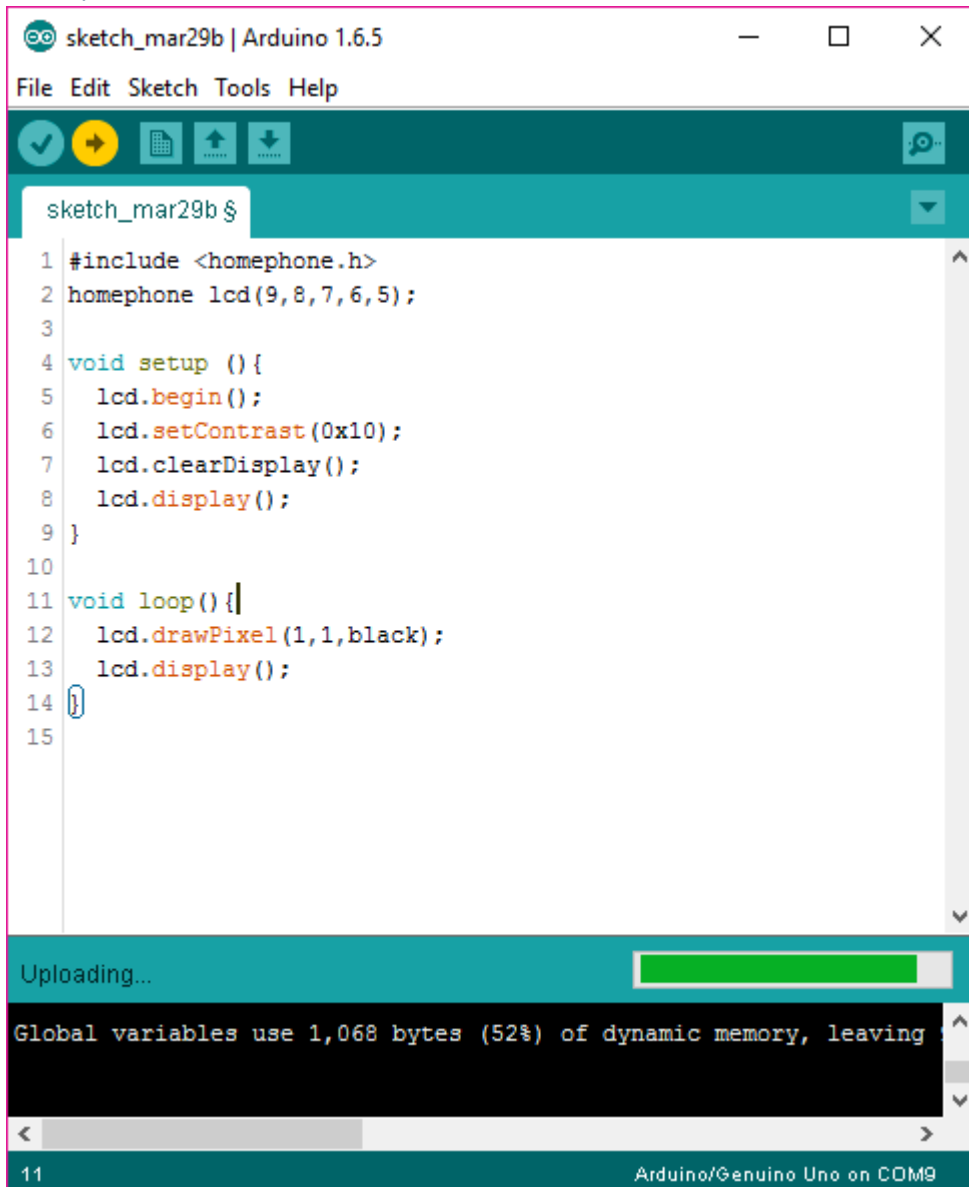
Dòng 7: xóa màn hình.

Dòng 8: hiển thị màn hình. Hàm này sẽ được sử dụng để hiển thị thông tin sau khi sử dụng các hàm vẽ khác.

Ghi chú: nên xóa màn hình trước khi sử dụng nếu màn hình có hiện các điểm ảnh không mong muốn.

III. CÁC HÀM VẼ CƠ BẢN

- Vẽ một điểm ảnh



lcd.drawPixel(x,y,color)

hàm này sử dụng để vẽ một điểm ảnh tại tọa độ x, y và màu sắc do các bạn định nghĩa (black, white).

Ví dụ:

- Vẽ một điểm đen, tọa độ 1,1 trên màn hình, sau đó dùng hàm `lcd.display()` để hiển thị điểm ảnh đó trên màn hình.

lcd.drawPixel(1,1,black);

lcd.display();

- Vẽ một điểm trắng tọa độ 2,2 trên màn hình tương tự sử dụng hàm `lcd.display()` để hiển thị trên màn hình.

```
lcd.drawPixel(2,2,white);  
lcd.display();
```

- **Vẽ một đường thẳng**

```
lcd.drawLine(x0,y0, x1,y1, color);
```

Hàm này để vẽ một đoạn thẳng đi nối hai điểm (x0,y0) và (x1,y1), và màu sắc của đường thẳng do các bạn định nghĩa (black, white)

Ví dụ:

- Vẽ một đoạn thẳng nối hai điểm (2,5) và (4,3) đường màu đen

```
lcd.drawLine(2,5,4,3,black);  
lcd.display();
```
- Tương tự mới một đoạn thẳng màu trắng

```
lcd.drawLine(2,5,4,3,white);  
lcd.display();
```

- **Vẽ một hình vuông hay hình chữ nhật**

```
lcd.drawRect(x0,y0,w,h,color);
```

Hàm này để vẽ một hình vuông, chữ nhật trong đó điểm (x0,y0) là tọa độ đặt điểm đầu tiên của hình chữ nhật, w chiều dài, h chiều rộng và màu sắc

Ví dụ:

- Vẽ hình chữ nhật có điểm bắt đầu từ tọa độ (1,1) chiều dài 15, chiều rộng 5 màu đen

```
lcd.drawRect(1,1,15,5,black);  
lcd.display();
```
- Hoàn toàn có thể cho 2 chiều bằng nhau để tạo hình vuông.

- **Tô màu hình vuông hay hình chữ nhật**

```
lcd.fillRect(x0,y0,w,h,color);
```

Tương tự như trên ví dụ

- Tô màu hình chữ nhật (toàn màu đen) có điểm bắt đầu từ tọa độ (1,1) chiều dài 15, chiều rộng 5

```
lcd.fillRect(1,1,15,5,black);  
lcd.display();
```
- Với màu trắng

```
lcd.fillRect(1,1,15,5,white);  
lcd.display();
```

- **Vẽ hình tròn**

```
lcd.drawCircle(x0,y0,r,color)
```

Hàm này cho phép vẽ đường tròn có tâm (x0,y0) và đường kính r, màu sắc của đường tròn,

Ví dụ:

- Vẽ đường tròn tâm A (15,15) đường kính 10, màu đen

```
lcd.drawCircle(15,15,10,black);  
lcd.display();
```
- Tương tự với màu trắng chúng ta chỉ việc thay black = white

- **Tô màu hình tròn**

```
lcd.fillCircle(x0,y0,r,color)
```

hàm này cho phép tô toàn bộ màu trong hình tròn

Ví dụ:

- Tô màu đường tròn tâm A (15,15) đường kính 10, màu đen

```
lcd.fillCircle(x0,y0,r,color);  
lcd.display();
```

- **Vẽ hình tam giác**

`lcd.drawTriangle(x0,y0, x1,y1, x2,y2,color);`

hàm này cho phép vẽ một hình tam giác nối 3 đỉnh tại 3 tọa độ A(x0,y0), B(x1,y1), C(x2,y2)
Ví dụ:

- Vẽ đường tam giác qua 3 điểm A(15,20); B (7,9); C(8,15);
`lcd.drawTriangle(15,20,7,9,8,15,black);`
`lcd.display();`
- Tương tự có thể vẽ với màu trắng, thay black = white

- **Viết chữ**

`lcd.print("chữ mình cần đưa lên màn hình");`

hàm này cho phép mình đưa lên màn hình dòng chữ mà mình muốn hiển thị tại bất cứ tọa độ nào trên màn hình. Các câu lệnh để viết chữ như sau:

Ví dụ:

- Viết chữ hello tại tọa độ (15,15) chữ đen nền trắng
`lcd.setCursor(15,15); //đặt con trỏ vào tọa độ (15,15)`
`lcd.setTextCursor(black,white); //chữ đen nền trắng, nếu muốn chữ trắng nền đen thì chỉ việc đổi lại lcd.setTextCursor(white,black);`
`lcd.print("hello world!");`
`lcd.display();`

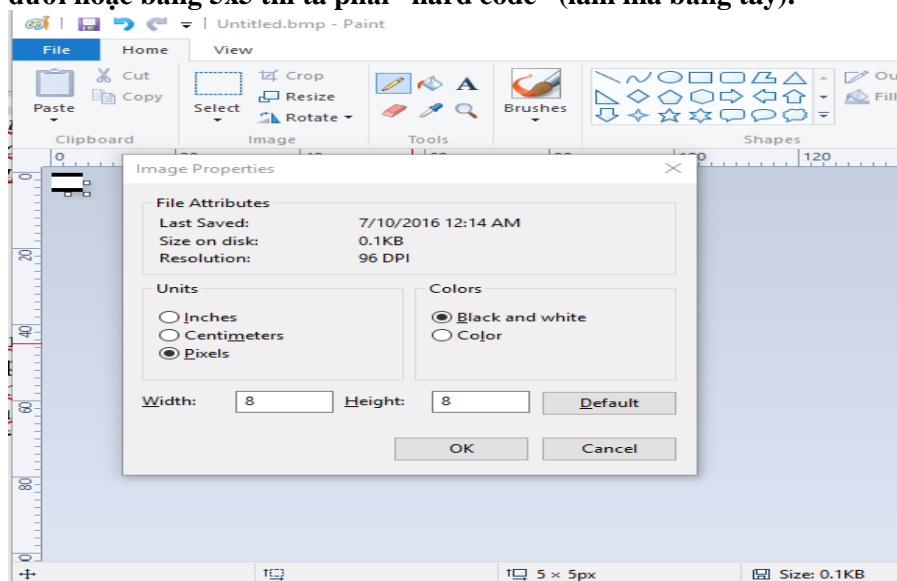
- **Vẽ hình ảnh.**

`lcd.drawBitmap(x,y,array,w,h,color,backgroundColor);`

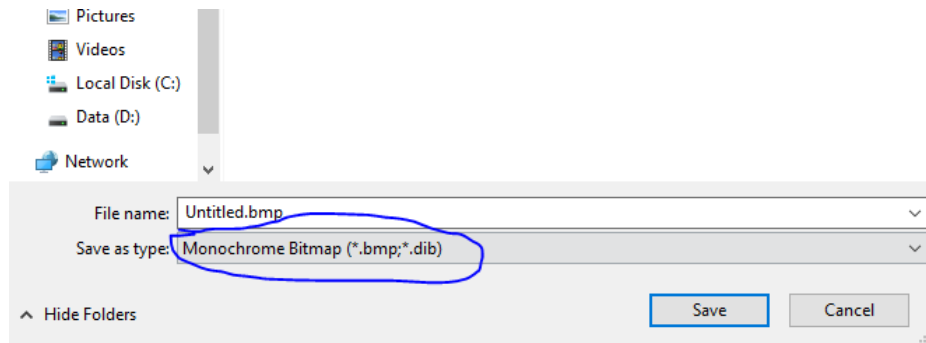
hàm này cho phép thể hiện các điểm ảnh từ trong mảng. trong đó x, y là tọa độ cần đặt hình ảnh đó, array là dữ liệu chứa hình ảnh đó, w là độ dài hình ảnh, h chiều cao hình ảnh, color màu nền của hình. Có thể là màu đen (black) hoặc trắng (white), BackgroundColor: là màu nền của hình đó. Có thể là màu đen (black) hoặc trắng (white)

Ví dụ:

- Để tạo được mảng chứa hình ảnh ta cần công cụ vẽ tranh bất cứ nào. Trong tài liệu này sẽ sử dụng paint để làm ví dụ.
Mở Paint ra, bấm nút Ctrl + e để set độ phân giải. **Chú ý: Độ phân giải tối đa của hình ảnh sẽ là 128x64, và độ phân giải tối thiểu là 6x6. Nếu đặt độ phân giải dưới hoặc bằng 5x5 thì ta phải "hard code" (làm mã bằng tay).**



Sau đó bấm OK và vẽ vào ảnh vừa tạo sau đó save lại dưới dạng monochrome

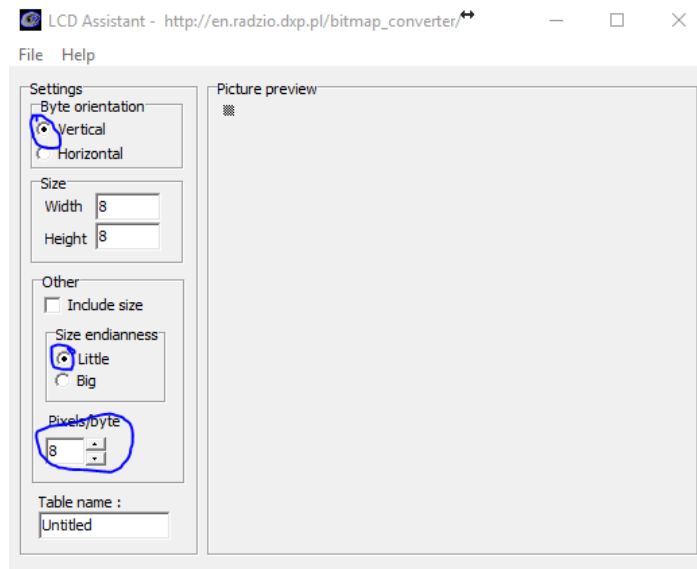


Tiếp tục dùng phần mềm đính kèm có tên: “LCDAssistant.exe”

Vào mục file chọn load image và trở vào file mình cần làm.

Trong mục “Byte Orientation” chọn “Vertical”

Tất cả còn lại để mặc định như hình dưới:



Tiếp tục vào file chọn save output dưới dạng: tên file.txt

Vào file .txt vừa tạo và copy mảng xong dán vào đoạn code bên trên và nạp vào MCU để hiện thị hình ảnh.

- Vẽ hình có độ phân giải 8x8 tại tọa độ (0,0):

```
#include "homephone.h"
homephone lcd (9,8,7,6,5);
static const unsigned char PROGMEM test [] = {0x83, 0xC6, 0x4C, 0x68, 0x38,
0x78, 0xCE, 0x83, };
void setup (){
  lcd.begin();
  lcd.setContrast(0x10);
  lcd.clearDisplay();
  lcd.drawBitmap (0,0,test,8,8,black,white);
  lcd.display();
}

void loop(){
}
```