Graphics using Nokia 3310/5110 LCD Display

http://www.adafruit.com/products/338
https://learn.adafruit.com/nokia-5110-3310-monochrome-lcd
http://www.thaieasyelec.com/LCD-Display/Module/Graphic-LCD-84-48-Nokia-5110.html
http://playground.arduino.cc/Code/PCD8544
http://nds2.nokia.com/files/support/apac/phones/guides/Nokia_5110_APAC_UG__EN.pdf

CONNECTION: From LCD to Arduino Pins

pin 7	Serial clock out (SCLK)
pin 6	Serial data out (DIN)
pin 5	Data/Command select (D/C)
pin 4	LCD chip select (CS)
pin 3	LCD reset (RST)
LCD_CMD	13 (LIGHT) Digital Pin 13 else control LCD backlight with potentiometer using PWM pins
VCC	3.3V (DO NOT PLUG IT TO 5V)
GRD	GROUND

HARDWARE REQUIRED

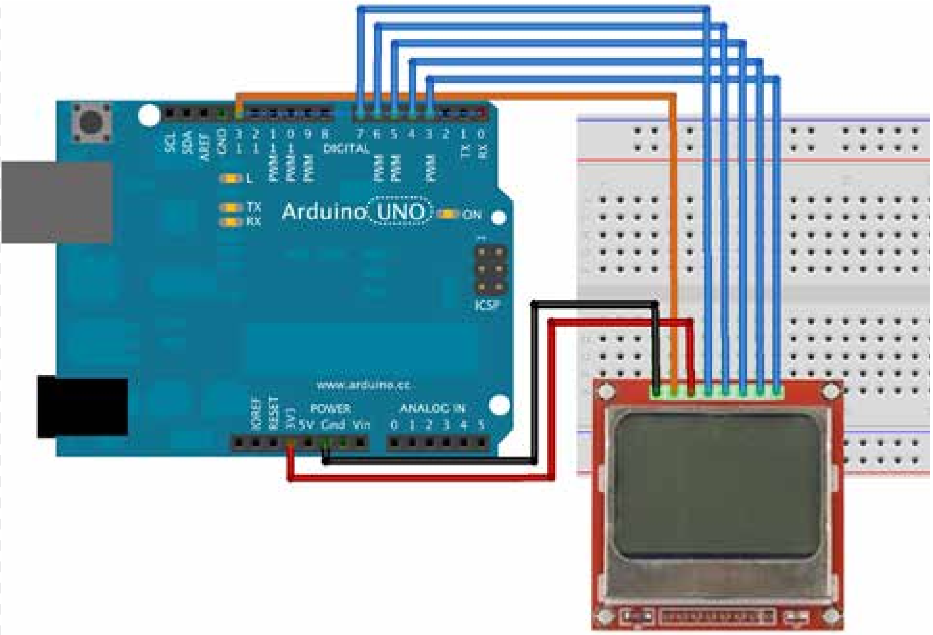
Arduino Board
LCD Display NOKIA type 3310 or 5110

CIRCUIT

SOME PROJECTS REQUIRE LIBRARIES TO BE INSTALLED
These displays are small, only about 1.5» diameter, but very readable due and comes with a backlight. This display is made of 84x48 individual pixels, so you can use it for graphics, text or bitmaps. These displays are inexpensive, easy to use, require only a few digital I/O pins and are fairly low power as well.

To drive the display, you will need 3 to 5 digital output pins (depending on whether you want to manually control the chip select and reset lines). Another pin can be used to control (via on/off or PWM) the backlight. To make things easy for you, we've written a nice graphics library that can print text, pixels, rectangles, circles and lines! The library is written for the Arduino but can easily be ported to your favorite microcontroller.

IMAGE



CODE

```
/****************************
****************************
**************
This is an example sketch for our
Monochrome Nokia 5110 LCD
Displays

  Pick one up today in the adafruit
shop!
  ------> http://www.adafruit.com/
products/338

These displays use SPI to commu-
nicate, 4 or 5 pins are required to
interface

Adafruit invests time and re-
sources providing this open source
code,
please support Adafruit and open-
source hardware by purchasing
products from Adafruit!

Written by Limor Fried/Ladyada
for Adafruit Industries.
BSD license, check license.txt for
more information
All text above, and the splash
screen must be included in any
redistribution
****************************
****************************
*************/
```

```
// libraries
#include <SPI.h>
#include <Adafruit_GFX.h>
#include <Adafruit_PCD8544.h>

// Software SPI (slower updates, more flexible pin options):
// pin 7 - Serial clock out (SCLK)
// pin 6 - Serial data out (DIN)
// pin 5 - Data/Command select (D/C)
// pin 4 - LCD chip select (CS)
// pin 3 - LCD reset (RST)

Adafruit_PCD8544 display = Adafruit_PCD8544(7, 6, 5, 4, 3);

// Hardware SPI (faster, but must use certain hardware pins):
// SCK is LCD serial clock (SCLK) - this is pin 13 on Arduino Uno
// MOSI is LCD DIN - this is pin 11 on an Arduino Uno
// pin 5 - Data/Command select (D/C)
// pin 4 - LCD chip select (CS)
// pin 3 - LCD reset (RST)
// Adafruit_PCD8544 display = Adafruit_PCD8544(5, 4, 3);
// Note with hardware SPI MISO and SS pins aren't used but will still be read
// and written to during SPI transfer.  Be careful sharing these pins!

#define NUMFLAKES 10
#define XPOS 0
#define YPOS 1
#define DELTAY 2

#define LOGO16_GLCD_HEIGHT 16
#define LOGO16_GLCD_WIDTH  16

static unsigned char PROGMEM logo16_glcd_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };

void setup()   {
  Serial.begin(9600);
  pinMode(13,OUTPUT); // backlight connection

display.begin();
  // init done

  // you can change the contrast around to adapt the display
  // for the best viewing!
  display.setContrast(50);

  display.display(); // show splashscreen
  delay(2000);
  display.clearDisplay();   // clears the screen and buffer

  // draw a single pixel
  display.drawPixel(10, 10, BLACK);
  display.display();
  delay(2000);
  display.clearDisplay();
```

CODE

```
// draw many lines
testdrawline();
display.display();
delay(2000);
display.clearDisplay();

// draw rectangles
testdrawrect();
display.display();
delay(2000);
display.clearDisplay();

// draw multiple rectangles
testfillrect();
display.display();
delay(2000);
display.clearDisplay();

// draw mulitple circles
testdrawcircle();
display.display();
delay(2000);
display.clearDisplay();

// draw a circle on the center, 10 pixel radius
display.fillCircle(display.width()/2, display.height()/2, 10, BLACK);
display.display();
delay(2000);
display.clearDisplay();

testdrawroundrect();
delay(2000);
display.clearDisplay();

testfillroundrect();
delay(2000);
display.clearDisplay();

testdrawtriangle();
delay(2000);
display.clearDisplay();

testfilltriangle();
delay(2000);
display.clearDisplay();

// draw the first ~12 characters in the font
testdrawchar();
display.display();
delay(2000);
display.clearDisplay();

// text display tests
display.setTextSize(1);
display.setTextColor(BLACK);
display.setCursor(0,0);
display.println(“Hello, world!”);
display.setTextColor(WHITE, BLACK); // ‘inverted’ text
display.println(3.141592);
display.setTextSize(2);
display.setTextColor(BLACK);
display.print(“0x”); display.println(0xDEADBEEF, HEX);
display.display();
delay(2000);

// miniature bitmap display
display.clearDisplay();
display.drawBitmap(30, 16,  logo16_glcd_bmp, 16, 16, 1);
display.display();
```

CODE

```
  // invert the display
  display.invertDisplay(true);
  delay(1000);
  display.invertDisplay(false);
  delay(1000);

  // draw a bitmap icon and ‘animate’ movement
  testdrawbitmap(logo16_glcd_bmp, LOGO16_GLCD_HEIGHT, LOGO16_
GLCD_WIDTH);
}


void loop() {
  digitalWrite(13,HIGH);
}


void testdrawbitmap(const uint8_t *bitmap, uint8_t w, uint8_t h) {
  uint8_t icons[NUMFLAKES][3];
  srandom(666);     // whatever seed

  // initialize
  for (uint8_t f=0; f< NUMFLAKES; f++) {
    icons[f][XPOS] = random() % display.width();
    icons[f][YPOS] = 0;
    icons[f][DELTAY] = random() % 5 + 1;

    Serial.print(“x: “);
    Serial.print(icons[f][XPOS], DEC);
    Serial.print(“ y: “);
    Serial.print(icons[f][YPOS], DEC);
    Serial.print(“ dy: “);
    Serial.println(icons[f][DELTAY], DEC);
  }

  while (1) {
    // draw each icon
    for (uint8_t f=0; f< NUMFLAKES; f++) {
      display.drawBitmap(icons[f][XPOS], icons[f][YPOS], logo16_glcd_
bmp, w, h, BLACK);
    }
    display.display();
    delay(200);

    // then erase it + move it
    for (uint8_t f=0; f< NUMFLAKES; f++) {
      display.drawBitmap(icons[f][XPOS], icons[f][YPOS],  logo16_glcd_
bmp, w, h, WHITE);
      // move it
      icons[f][YPOS] += icons[f][DELTAY];
      // if its gone, reinit
      if (icons[f][YPOS] > display.height()) {
           icons[f][XPOS] = random() % display.width();
           icons[f][YPOS] = 0;
           icons[f][DELTAY] = random() % 5 + 1;
      }
    }
   }
}


void testdrawchar(void) {
  display.setTextSize(1);
  display.setTextColor(BLACK);
  display.setCursor(0,0);

  for (uint8_t i=0; i < 168; i++) {
    if (i == ‘\n’) continue;
    display.write(i);
```

CODE

```
    //if ((i > 0) && (i % 14 == 0))
      //display.println();
  }
  display.display();
}

void testdrawcircle(void) {
  for (int16_t i=0; i<display.height(); i+=2) {
    display.drawCircle(display.width()/2, display.height()/2, i, BLACK);
    display.display();
  }
}

void testfillrect(void) {
  uint8_t color = 1;
  for (int16_t i=0; i<display.height()/2; i+=3) {
    // alternate colors
    display.fillRect(i, i, display.width()-i*2, display.height()-i*2, color%2);
    display.display();
    color++;
  }
}

void testdrawtriangle(void) {
  for (int16_t i=0; i<min(display.width(),display.height())/2; i+=5) {
    display.drawTriangle(display.width()/2, display.height()/2-i,
                     display.width()/2-i, display.height()/2+i,
                     display.width()/2+i, display.height()/2+i, BLACK);
    display.display();
  }
}

void testfilltriangle(void) {
  uint8_t color = BLACK;
  for (int16_t i=min(display.width(),display.height())/2; i>0; i-=5) {
    display.fillTriangle(display.width()/2, display.height()/2-i,
                     display.width()/2-i, display.height()/2+i,
                     display.width()/2+i, display.height()/2+i, color);
    if (color == WHITE) color = BLACK;
    else color = WHITE;
    display.display();
  }
}

void testdrawroundrect(void) {
  for (int16_t i=0; i<display.height()/2-2; i+=2) {
    display.drawRoundRect(i, i, display.width()-2*i, display.height()-2*i,
display.height()/4, BLACK);
    display.display();
  }
}

void testfillroundrect(void) {
  uint8_t color = BLACK;
  for (int16_t i=0; i<display.height()/2-2; i+=2) {
    display.fillRoundRect(i, i, display.width()-2*i, display.height()-2*i,
display.height()/4, color);
    if (color == WHITE) color = BLACK;
    else color = WHITE;
    display.display();
  }
}

void testdrawrect(void) {
  for (int16_t i=0; i<display.height()/2; i+=2) {
    display.drawRect(i, i, display.width()-2*i, display.height()-2*i, BLACK);
    display.display();
  }
}
```

CODE

```
void testdrawline() {
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(0, 0, i, display.height()-1, BLACK);
    display.display();
  }
  for (int16_t i=0; i<display.height(); i+=4) {
    display.drawLine(0, 0, display.width()-1, i, BLACK);
    display.display();
  }
  delay(250);

  display.clearDisplay();
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(0, display.height()-1, i, 0, BLACK);
    display.display();
  }
  for (int8_t i=display.height()-1; i>=0; i-=4) {
    display.drawLine(0, display.height()-1, display.width()-1, i, BLACK);
    display.display();
  }
  delay(250);

  display.clearDisplay();
  for (int16_t i=display.width()-1; i>=0; i-=4) {
    display.drawLine(display.width()-1, display.height()-1, i, 0, BLACK);
    display.display();
  }
  for (int16_t i=display.height()-1; i>=0; i-=4) {
    display.drawLine(display.width()-1, display.height()-1, 0, i, BLACK);
    display.display();
  }
  delay(250);

  display.clearDisplay();
  for (int16_t i=0; i<display.height(); i+=4) {
    display.drawLine(display.width()-1, 0, 0, i, BLACK);
    display.display();
  }
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(display.width()-1, 0, i, display.height()-1, BLACK);
    display.display();
  }
  delay(250);
}
```