MODULES

1420491028

VÝROBNÍ ČÍSLO

LCD display Nokia 5110 modré podsvícení

1. POPIS

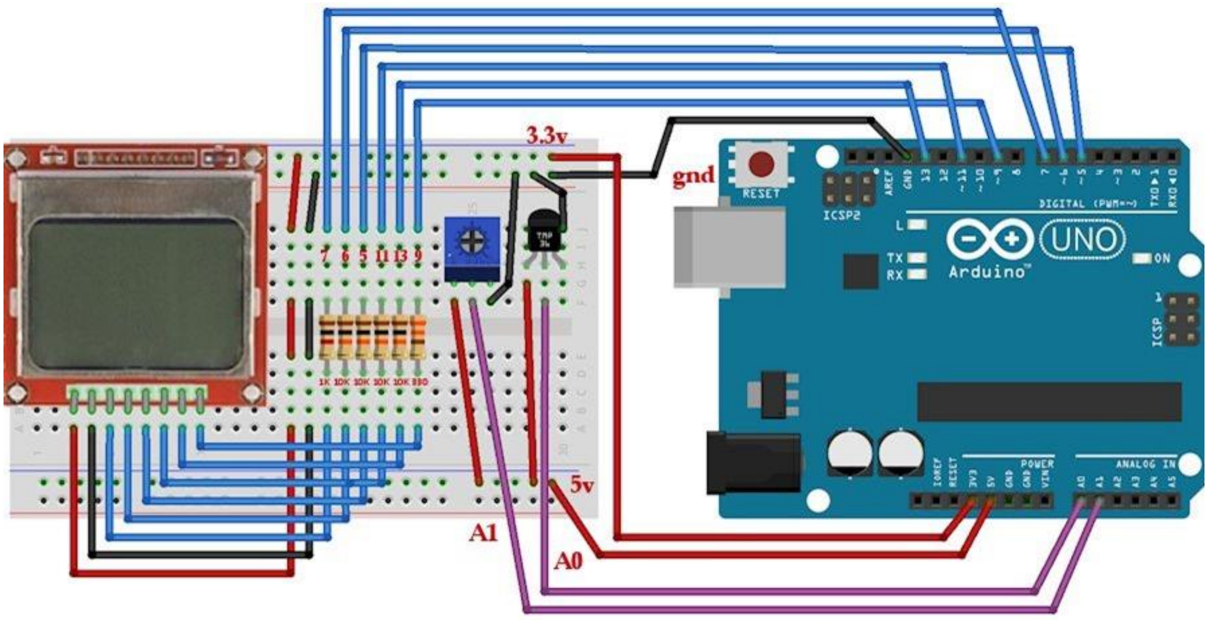
LCD display z populárního telefonu Nokia 5110 je přizpůsoben ke snadnému použití s vývojovými kity. Výhodou displeje je velmi nízká spotřeba (displej umožňuje také power down mód). Komunikace probíhá přes rozhraní SPI (Serial Peripheral Interface) rychlostí až 4 Mbps.

2. SPECIFIKACE DISPLEJE

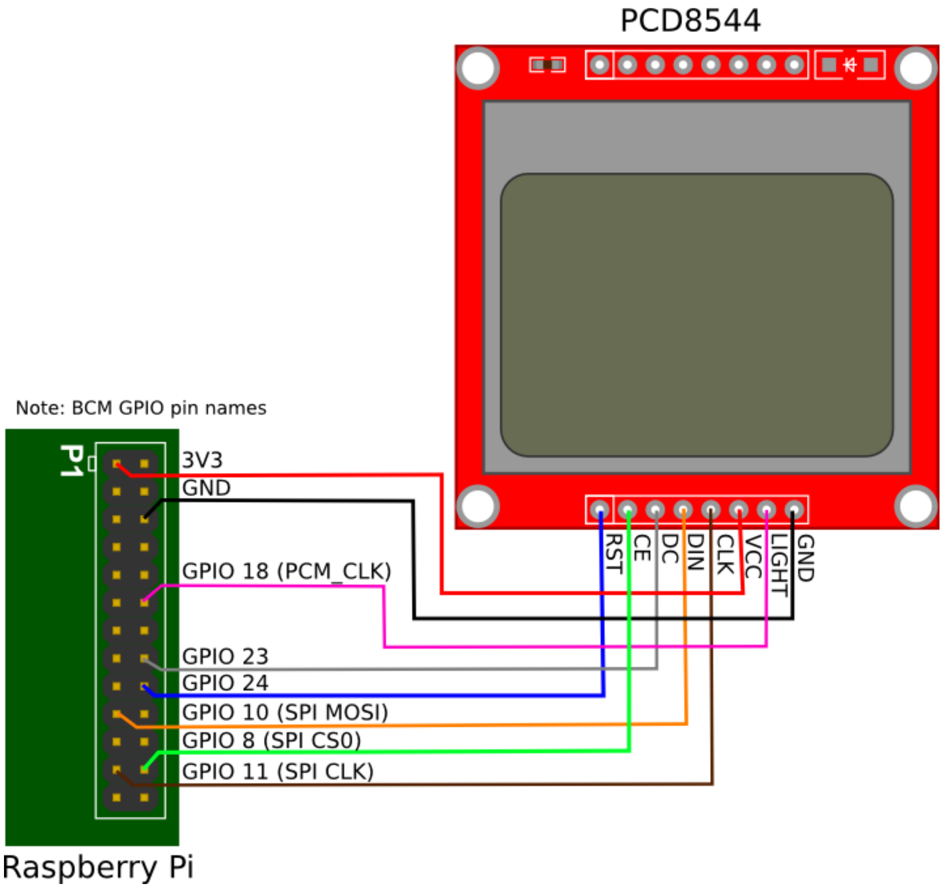
Napájení	2,7–3,3 V	**Spotřeba**	< 200 uA
Rozlišení	84 x 48 px	**Podsvícení**	Modré
Komunikační rychlost SPI	až 4 MBps	**Rozměry**	43,6 x 43,1 mm

ECLIPSERA s.r.o. Distributor pro ČR.

3. Zapojení



3.3v
gnd
7 6 5 11 13 9
1K 10K 10K 10K 10K 330
5v
A1
A0
RESET
ICSP2
DIGITAL (PWM~)
UNO
Arduino
POWER
ANALOG IN



PCD8544
Note: BCM GPIO pin names
P1
3V3
GND
GPIO 18 (PCM_CLK)
GPIO 23
GPIO 24
GPIO 10 (SPI MOSI)
GPIO 8 (SPI CS0)
GPIO 11 (SPI CLK)
RST
CE
DC
DIN
CLK
VCC
LIGHT
GND
Raspberry Pi

4. Ukázka programu

Pro správnou funkci programu je nutné nainstalovat knihovnu „Adafruit PCD8544 Nokia 5110 LCD Library“. Po instalaci je v příkladech k dispozici kód uv edený níže.

```
/*********************************************************************
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 communicate, 4 or 5 pins are required to
interface

Adafruit invests time and resources 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
*********************************************************************/

#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 const 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);

  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();

  // 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, 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);

  // rotation example
  display.clearDisplay();
  display.setRotation(1);  // rotate 90 degrees counter clockwise, can also
use values of 2 and 3 to go further.
  display.setTextSize(1);
  display.setTextColor(BLACK);
  display.setCursor(0,0);
  display.println("Rotation");
  display.setTextSize(2);
  display.println("Example!");
  display.display();
  delay(2000);

  // revert back to no rotation
  display.setRotation(0);

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

  // 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_WIDTH,
LOGO16_GLCD_HEIGHT);
}


void loop() {

}


void testdrawbitmap(const uint8_t *bitmap, uint8_t w, uint8_t h) {
  uint8_t icons[NUMFLAKES][3];
  randomSeed(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);
    //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();
  }
}

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);
}
```