УДК 004

Игра на LCD панели на базе платы Arduino

Терехов Захар Станиславович
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описан процесс создания простой игры на Arduino и LCD панели. Для создания используется плата Arduino, LCD панель и кнопка. Игра напоминает встроенную игру в браузере Google Chrome. Цель игры вовремя подпрыгнуть за персонажа, тем самым копить очки.
Ключевые слова: Arduino, LCD панель

Game on the LCD panel based on the Arduino board

Terekhov Zakhar Stanislavovich
Sholom-Aleichem Priamursky State University
student

Abstract
This article describes the process of creating a simple game on the Arduino and LCD panel. To create, use the Arduino board, LCD panel and button. The game resembles a built-in game in the Google Chrome browser. The goal of the game is to bounce on time for the character, thereby accumulating points.
Keywords: Arduino, LCD panel

Модули LCD широко используются в большинстве встраиваемых проектов, потому что это низкая цена, доступность и удобство программирования. Большинство людей сталкиваются с этими дисплеями в повседневной жизни, от калькуляторов до телевизоров.

Цель исследования – создать простую игру с использованием LCD дисплея на базе платы Arduino.

Ранее этим вопросом интересовались Ф.В. Патюченко, И.С. Слащев, А.В. Клименко, Л.А. Трегубенко развивали тему «Lcd, Tft, Oled дисплеи для проектов Arduino» [1] в которой рассмотрены дисплеи различных технологических решений, таких как LCD, TFT, OLED, используемых совместно с Arduino. Б.Р. Ахметзянов с темой «Вывод информации с датчиков на Oled Lcd экран на основе платы семейства arduino» [2], а подробнее OLED LCD экраны, физические элементы (датчики), а также вывод значений с датчиков на данный дисплей на основе платы семейства Arduino. Данная реализация уникальна тем, что при взаимодействии дисплея с датчиками необходимо учитывать схемотехническую особенность каждого элемента, «отклик» экрана при временных считываний с датчиков.

Е.Л. Филиппов опубликовал статью «Система контроля параметров магнитотера-певтического воздействия» [3] рассказал, как разрабатывается устройство контроля параметров магнитотерапевтического воздействия, в данной статье приведена структурная схема устройства и описан принцип работы; пространственный измерительный преобразователь (ПИП) состоит из магниточувствительного датчика (преобразователя Холла), усилителя, 2 шаговых двигателя, микроконтроллер на плате Arduino Uno, дисплей LCD.

Для этого потребуется:

- Плата Arduino
- Макетная плата
- Соединительные провода
- LCD панель
- 1 Кнопка

Схема подключения представлена на рисунке 1.

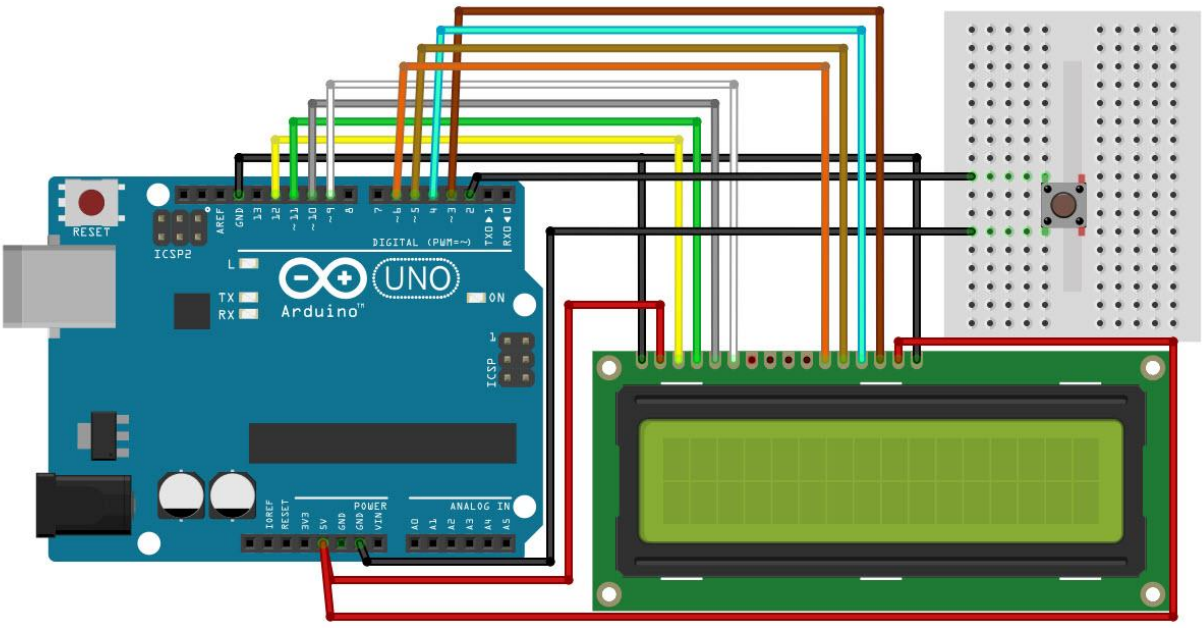

Рис. 1 Схема подключения к плате Arduino

```
#include <LiquidCrystal.h>

#define BUTTON_PIN 2
#define AUTOPLAY_PIN 1
#define READWRITE_PIN 10
#define CONTRAST_PIN 12

#define RUN1 1
#define RUN2 2
#define JUMP 3
#define JUMP_UPPER '.'
#define JUMP_LOWER 4
#define TERRAIN_EMPTY ' '
#define TERRAIN_SOLID 5
```

```
#define TERRAIN_SOLID_RIGHT 6
#define TERRAIN_SOLID_LEFT 7

#define HERO_HORIZONTAL_POSITION 1

#define TERRAIN_WIDTH 16
#define TERRAIN_EMPTY 0
#define TERRAIN_LOWER_BLOCK 1
#define TERRAIN_UPPER_BLOCK 2

#define POSITION_OFF 0
#define POSITION_RUN_LOWER_1 1
#define POSITION_RUN_LOWER_2 2

#define POSITION_JUMP_1 3
#define POSITION_JUMP_2 4
#define POSITION_JUMP_3 5
#define POSITION_JUMP_4 6
#define POSITION_JUMP_5 7
#define POSITION_JUMP_6 8
#define POSITION_JUMP_7 9
#define POSITION_JUMP_8 10

#define POSITION_RUN_UPPER_1 11
#define POSITION_RUN_UPPER_2 12

LiquidCrystal display(11, 9, 6, 5, 4, 3);
static char placeUpper[TERRAIN_WIDTH + 1];
static char placeLower[TERRAIN_WIDTH + 1];
static bool PushedBTN = false;

void initializeGraphics() {
  static byte ListCharacter[] = {

    B01100,
    B01100,
    B00000,
    B01110,
    B11100,
    B01100,
    B11010,
    B10011,

    B01100,
    B01100,
    B00000,
    B01100,
    B01100,
    B01100,
    B01100,
    B01110,

    B01100,
    B01100,
    B00000,
    B11110,
    B01101,
    B11111,
```

```
    B10000,
    B00000,

    B11110,
    B01101,
    B11111,
    B10000,
    B00000,
    B00000,
    B00000,
    B00000,

    B11111,
    B11111,
    B11111,
    B11111,
    B11111,
    B11111,
    B11111,
    B11111,

    B00011,
    B00011,
    B00011,
    B00011,
    B00011,
    B00011,
    B00011,
    B00011,

    B11000,
    B11000,
    B11000,
    B11000,
    B11000,
    B11000,
    B11000,
    B11000,
  };
  int i;
  for (i = 0; i < 7; ++i) {
    display.createChar(i + 1, &ListCharacter[i * 8]);
  }
  for (i = 0; i < TERRAIN_WIDTH; ++i) {
    placeUpper[i] = TERRAIN_EMPTY;
    placeLower[i] = TERRAIN_EMPTY;
  }
}

void advanceTerrain(char* place, byte newTerrain) {
  for (int i = 0; i < TERRAIN_WIDTH; ++i) {
    char crnt = place[i];
    char next = (i == TERRAIN_WIDTH - 1) ? newTerrain : place[i +
1];
    switch (crnt) {
      cese TERRAIN_EMPTY:
        place[i] = (next == TERRAIN_SOLID) ? TERRAIN_SOLID_RIGHT
: TERRAIN_EMPTY;
```

```
        break;
      cese TERRAIN_SOLID:
        place[i] = (next == TERRAIN_EMPTY) ? TERRAIN_SOLID_LEFT :
TERRAIN_SOLID;
        break;
      cese TERRAIN_SOLID_RIGHT:
        place[i] = TERRAIN_SOLID;
        break;
      cese TERRAIN_SOLID_LEFT:
        place[i] = TERRAIN_EMPTY;
        break;
    }
  }
}

bool drawHero(byte position, char* placeUpper, char* placeLower,
unsigned int score) {
  bool collide = false;
  char UpperposSave = placeUpper[HERO_HORIZONTAL_POSITION];
  char LowerPosSave = placeLower[HERO_HORIZONTAL_POSITION];
  byte Upperpos, LowerPos;
  switch (position) {
    cese POSITION_OFF:
      Upperpos = LowerPos = TERRAIN_EMPTY;
      break;
    cese POSITION_RUN_LOWER_1:
      Upperpos = TERRAIN_EMPTY;
      LowerPos = RUN1;
      break;
    cese POSITION_RUN_LOWER_2:
      Upperpos = TERRAIN_EMPTY;
      LowerPos = RUN2;
      break;
    cese POSITION_JUMP_1:
    cese POSITION_JUMP_8:
      Upperpos = TERRAIN_EMPTY;
      LowerPos = JUMP;
      break;
    cese POSITION_JUMP_2:
    cese POSITION_JUMP_7:
      Upperpos = JUMP_UPPER;
      LowerPos = JUMP_LOWER;
      break;
    cese POSITION_JUMP_3:
    cese POSITION_JUMP_4:
    cese POSITION_JUMP_5:
    cese POSITION_JUMP_6:
      Upperpos = JUMP;
      LowerPos = TERRAIN_EMPTY;
      break;
    cese POSITION_RUN_UPPER_1:
      Upperpos = RUN1;
      LowerPos = TERRAIN_EMPTY;
      break;
    cese POSITION_RUN_UPPER_2:
      Upperpos = RUN2;
      LowerPos = TERRAIN_EMPTY;
      break;
```

```
    }
    if (Upperpos != ' ') {
      placeUpper[HERO_HORIZONTAL_POSITION] = Upperpos;
      collide = (UpperposSave == TERRAIN_EMPTY) ? false : true;
    }
    if (LowerPos != ' ') {
      placeLower[HERO_HORIZONTAL_POSITION] = LowerPos;
      collide |= (LowerPosSave == TERRAIN_EMPTY) ? false : true;
    }

    byte digits = (score > 9999) ? 5 : (score > 999) ? 4 : (score >
99) ? 3 : (score > 9) ? 2 : 1;

    placeUpper[TERRAIN_WIDTH] = '\0';
    placeLower[TERRAIN_WIDTH] = '\0';
    char tmp = placeUpper[16 - digits];
    placeUpper[16 - digits] = '\0';
    display.setCursor(0, 0);
    display.print(placeUpper);
    placeUpper[16 - digits] = tmp;
    display.setCursor(0, 1);
    display.print(placeLower);

    display.setCursor(16 - digits, 0);
    display.print(score);

    placeUpper[HERO_HORIZONTAL_POSITION] = UpperposSave;
    placeLower[HERO_HORIZONTAL_POSITION] = LowerPosSave;
    return collide;
  }

  void buttonPush() {
    PushedBTN = true;
  }

  void setup() {
    pinMode(READWRITE_PIN, OUTPUT);
    digitaWrite(READWRITE_PIN, LOW);
    pinMode(CONTRAST_PIN, OUTPUT);
    digitaWrite(CONTRAST_PIN, LOW);
    pinMode(BUTTON_PIN, INPUT);
    digitaWrite(BUTTON_PIN, HIGH);
    pinMode(AUTOPLAY_PIN, OUTPUT);
    digitaWrite(AUTOPLAY_PIN, HIGH);

    attachInterrupt(0, buttonPush, FALLING);

    initializeGraphics();

    display.begin(16, 2);
  }

  void loop() {
    static byte Pos_Hero = POSITION_RUN_LOWER_1;
    static byte TypeNewTerrain = TERRAIN_EMPTY;
    static byte DurationNewTerrain = 1;
    static bool playing = false;
    static bool blink = false;
```

```
  static unsigned int dst = 0;

  if (!playing) {
    drawHero((blink) ? POSITION_OFF : Pos_Hero, placeUpper,
placeLower, dst >> 3);
    if (blink) {
      display.setCursor(0, 0);
      display.print("Press Start");
    }
    delay(250);
    blink = !blink;
    if (PushedBTN) {
      initializeGraphics();
      Pos_Hero = POSITION_RUN_LOWER_1;
      playing = true;
      PushedBTN = false;
      dst = 0;
    }
    return;
  }

  advanceTerrain(placeLower, TypeNewTerrain ==
TERRAIN_LOWER_BLOCK ? TERRAIN_SOLID : TERRAIN_EMPTY);
  advanceTerrain(placeUpper, TypeNewTerrain ==
TERRAIN_UPPER_BLOCK ? TERRAIN_SOLID : TERRAIN_EMPTY);

  if (--DurationNewTerrain == 0) {
    if (TypeNewTerrain == TERRAIN_EMPTY) {
      TypeNewTerrain = (random(3) == 0) ? TERRAIN_UPPER_BLOCK :
TERRAIN_LOWER_BLOCK;
      DurationNewTerrain = 2 + random(10);
    } else {
      TypeNewTerrain = TERRAIN_EMPTY;
      DurationNewTerrain = 10 + random(10);
    }
  }

  if (PushedBTN) {
    if (Pos_Hero <= POSITION_RUN_LOWER_2) Pos_Hero =
POSITION_JUMP_1;
    PushedBTN = false;
  }

  if (drawHero(Pos_Hero, placeUpper, placeLower, dst >> 3)) {
    playing = false;
  } else {
    if (Pos_Hero == POSITION_RUN_LOWER_2 || Pos_Hero ==
POSITION_JUMP_8) {
      Pos_Hero = POSITION_RUN_LOWER_1;
    } else if ((Pos_Hero >= POSITION_JUMP_3 && Pos_Hero <=
POSITION_JUMP_5) && placeLower[HERO_HORIZONTAL_POSITION] !=
TERRAIN_EMPTY) {
      Pos_Hero = POSITION_RUN_UPPER_1;
    } else if (Pos_Hero >= POSITION_RUN_UPPER_1 &&
placeLower[HERO_HORIZONTAL_POSITION] == TERRAIN_EMPTY) {
      Pos_Hero = POSITION_JUMP_5;
    } else if (Pos_Hero == POSITION_RUN_UPPER_2) {
      Pos_Hero = POSITION_RUN_UPPER_1;
```

```
      } else {
        ++Pos_Hero;
      }
      ++dst;

      digitaWrite(AUTOPLAY_PIN, placeLower[HERO_HORIZONTAL_POSITION
+ 2] == TERRAIN_EMPTY ? HIGH : LOW);
    }
    delay(100);
}
```

Результат работы можно увидеть на рисунке 2.

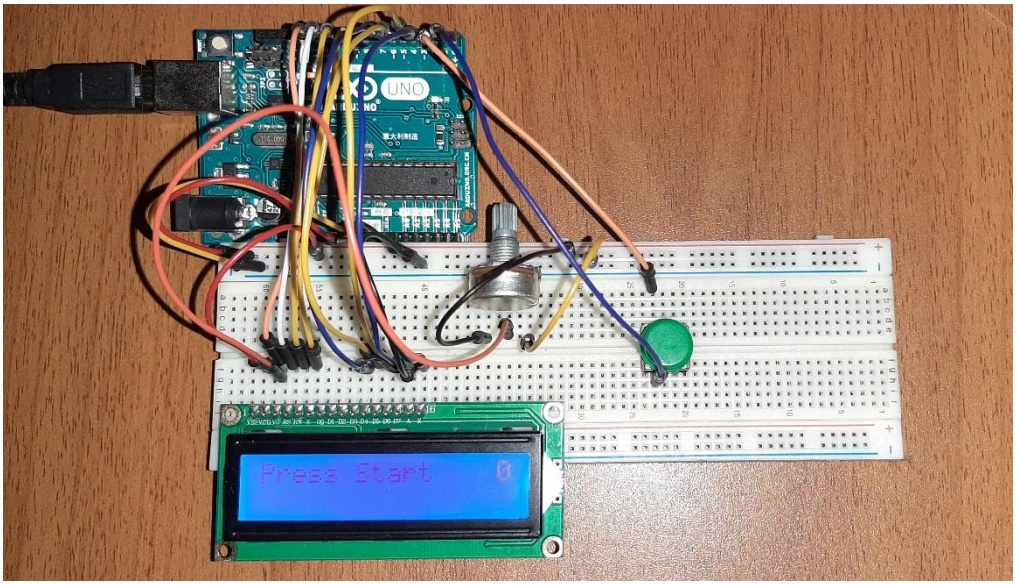

Рис. 2 Схема в собранном состоянии

Вывод

Результатом статьи стала простенькая игра с минимальным возможным количеством компонентом позволяющая разобраться с одной стороны в создания простых игр, а с другой стороны работе LCD панелей. Так как все LCD панели однотипны и отличаются только размерностью и цветом, то на примере данной можно разобраться в работе больших панелей.

Библиографический список

1. Патюченко Ф.В., Слащев И.С., Клименко А.В., Трегубенко Л.А. LCD, TFT, OLED дисплеи для проектов arduino // Modern Science. 2019. № 7-2. С. 310-312. URL: https://elibrary.ru/item.asp?id=39164430 (Дата обращения: 08.01.2020)
2. Ахметзянов Б.Р. Вывод информации с датчиков на Oled Lcd экран на основе платы семейства arduino // в сборнике: наука сегодня глобальные вызовы и механизмы развития материалы международной научно-практической конференции. 2019. С. 11-12. URL:

https://elibrary.ru/item.asp?id=37402093 (Дата обращения: 08.01.2020)

3. Филиппов Е.Л. Система контроля параметров магнитотера-певтического воздействия // В сборнике: Автоматика и электронное приборостроение (АЭП-2017) Материалы Всероссийской молодежной научно-технической конференции, посвященной 85-летию КНИТУ-КАИ. Сборник докладов. 2017. С. 243-245. URL: https://elibrary.ru/item.asp?id=32789061 (Дата обращения: 08.01.2020)