# Shoot the bullet на Arduino

Автор: Долженков Сергей Дмитриевич
Группа: 18.Б12-пу

03.12.2019

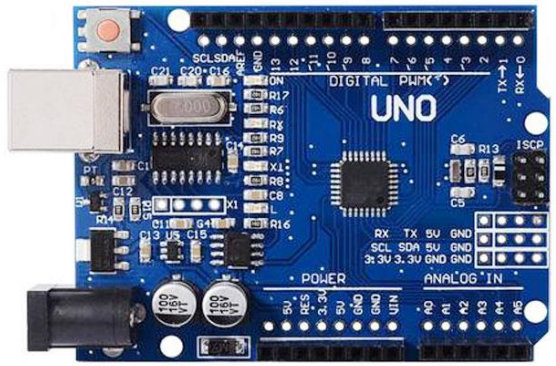
# Постановка задачи

Требуется полностью повторить геймплей оригинальной игры Shoot the bullet на микроконтроллере Arduino Uno используя соответствующую периферию.

# Требования

1. Финальная версия проекта должна являться законченным продуктом с полноценной документацией.
2. Код проекта должен содержать подробные комментарии, поясняющие основные моменты логики работы.
3. Styleguide не конкретизирован, однако необходимо придерживаться единого стиля во всем проекте.
4. Необходимо максимально разделить три основных компонента: обработку входных данных, вывод на дисплей и логику работы программы.
5. Реализовать пункты: таблица результатов и настройка(сложность и размер). И сохранять значения при отключении питания. А также реализовать ввод имени игрока.

# Использованные устройства

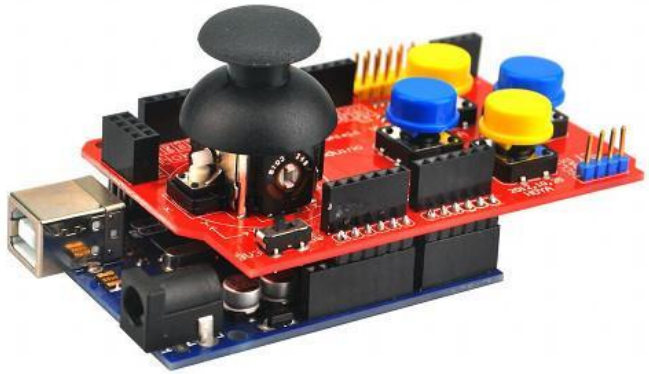



1. Arduino Uno

2. Joystick Shield

3. LCD 5110

# Подключение Joystick Shield



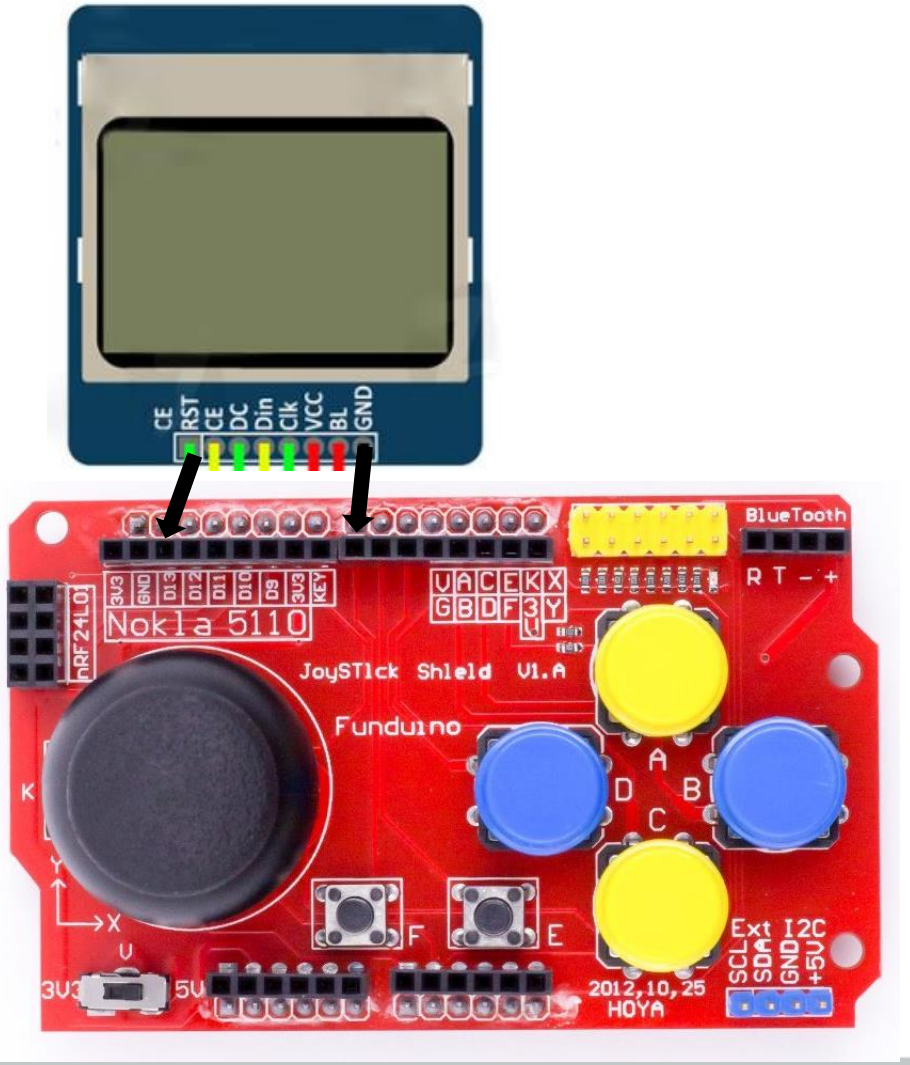
```
 9 #define UP_PIN 2
10 #define RIGHT_PIN 3
11 #define DOWN_PIN 4
12 #define LEFT_PIN 5

31 void setup() {

35   pinMode(UP_PIN, INPUT_PULLUP);
36   digitalWrite(UP_PIN, HIGH);
37   pinMode(RIGHT_PIN, INPUT_PULLUP);
38   digitalWrite(RIGHT_PIN, HIGH);
39   pinMode(DOWN_PIN, INPUT_PULLUP);
40   digitalWrite(DOWN_PIN, HIGH);
41   pinMode(LEFT_PIN, INPUT_PULLUP);
42   digitalWrite(LEFT_PIN, HIGH);
```

Режим чтения

Включаем подтягивающий резистор

| **Arduino** | A0 | A1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 |
|---|---|---|---|---|---|---|---|---|---|
| **Shield** | Ось X | Ось Y | A | B | C | D | E | F | K |

# Подключение LCD 5110




```
15 Adafruit_PCD8544 display = Adafruit_PCD8544(9, 10, 11, 13, 12);

31 void setup() {
32   pinMode(7, OUTPUT);
33   digitalWrite(7, LOW);
```

| LCD | JS |
|---|---|
| GND | -> D7 |
| BL | -> KEY |
| VCC | -> 3V3 |
| Clk | -> D9 |
| Din | -> D10 |
| DC | -> D11 |
| CE | -> D12 |
| RST | -> D13 |

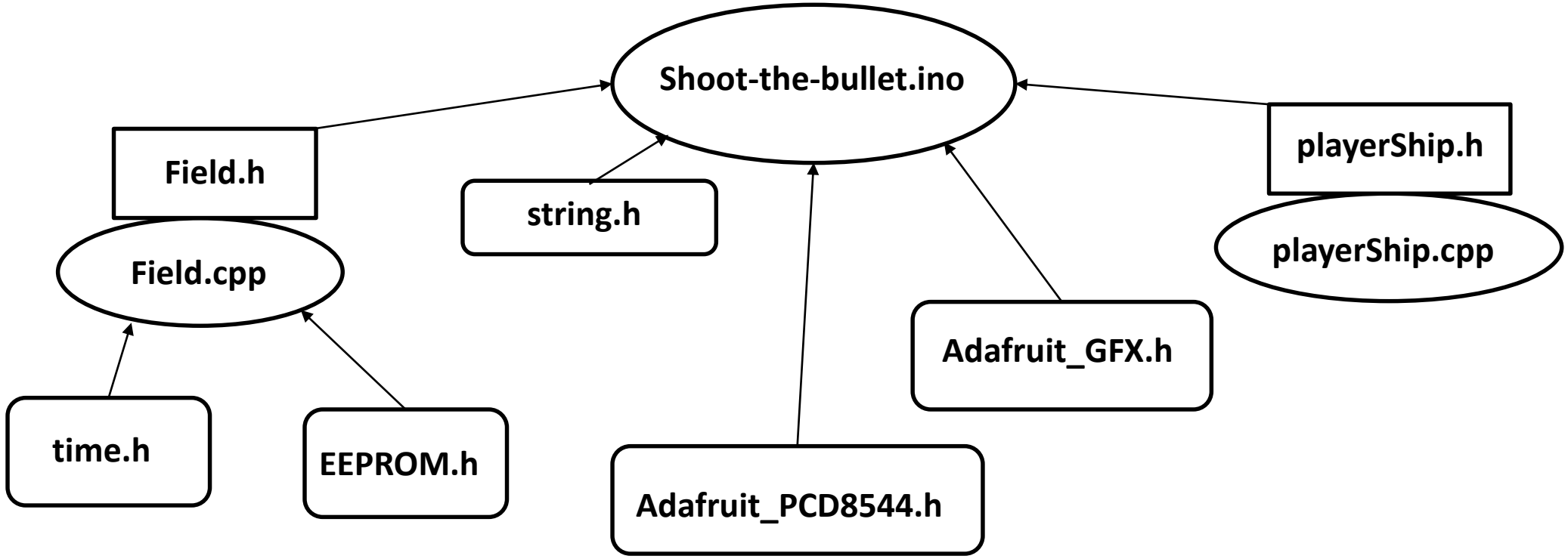
# Вид игры


Меню
• New game
• Continue
• Difficulty и значение
• Size и значение
• Highcores
Игра
• Игровое поле
• Очки
Таблица результатов
• Лучшие результаты игроков(имя и очки)
Окно проигрыша
• Сообщение о проигрыше
• Количество набранных очков
• Имя пользователя

# Использованные библиотеки

1. Adafruit_GFX(синтаксис и графические функции для LCD дисплеев)
2. Adafruit_PCD8544(дополнение к Adafruit_GFX для работы с LCD 5110)
3. string(для работы со строками)
4. time(для генерации случайных строк)
5. EEPROM(основные функции для работы с EEPROM(энергонезависимая память))

# Файловая архитектура проекта



# Описание файла Shoot-the-bullet.ino

```
53 void loop() {
54 //в соответсвии с состоянием игры в
55   if (Field.getMenuTrigger()) {
56     delay(200);
57     for (int i = 2; i < 6; i++) {
58       if (digitalRead(i) == LOW) {
59         input(i);
60       }
61     }
62     output();
63     delay(50);
64   }
```

```
67   else if (Field.getGameTrigger()) {
68     for (int h = 0; h < 50 - Field.getDifficulty() * 5; h++) {
69       output();
70       for (int i = 2; i < 6; i++) {
71         if (digitalRead(i) == LOW) {
72           input(i);
73           if (t) {
74             break;
75           }
76         }
77       }
78       if (t) {
79         break;
80       }
81       delay(200);
82       Field.flightShells();
83     }
84     if (!t) {
85       Field.fallingRow();
86     }
87     else {
88       t = false;
89     }
90   }
```

# Описание класса Field

```
class Field {
public:
  Field();
  ~Field();
  void createShell(int _x);//создаём с
  void deleteShell(int pos);//удляем с
  void createRow();//генерируем неполн
  void fallingRow();//логика падения р
  void deleteRow(int _y);//очищаем стр
  bool checkCollision(int _x, int _y);
  void flightShells();//логика полёта
  void changeRow(int _x, int _y);//зап
  bool checkRow(int _y);//проверяем не
  bool checkLose();//проверяе не дошли
  void lose();//меняем состояние игры
  void newGame();//перемещаем все элим
  void rewriteRows(int _y);
  void toMenu();
  void minusMenuState();
  void plusMenuState();
  void toGame();
  void plusSizeField();
  void plusDifficulty();
  void plusLoseState();
  void plusName(int i);
  void minusName(int i);
  void toTab();
  void writeTab();
  void rewriteTab(int pos);
  void minusLoseState();
  void saveTab();
  void reset();
  void saveDifficulty();
  void saveSize();

  const int getDataShellsX(int _i);
  const int getDataShellsY(int _i);
  const bool getDataRows(int _x, int _y);
  const int getPoints();
  const bool getLoseTrigger();
  const bool getMenuTrigger();
  const bool getGameTrigger();
  const bool getTabTrigger();
  const int getSizeField();
  const int getDifficulty();
  const int getMenuState();
  const bool getGameState();
  const int getLoseState();
  const char getName(int i);
  const char getTabName1(int i);
  const char getTabName2(int i);
  const char getTabName3(int i);
  const int getTabPoints(int i);
private:
  const static int width = 13;
  const static int height = 20;

  int dataShells[height - 2][2];//масси
  bool dataRows[height - 2][width - 2];

  int points;//очки ирока

  bool loseTrigger;//состояние игры
  bool menuTrigger;
  bool gameTrigger;
  bool tabTrigger;
  bool gameState;

  int sizeField;
  int difficulty;

  int menuState;
  int loseState;

  char Name[3];

  char tabName[5][3];
  int tabPoints[5];
```

# Описание класса PlayerShip

```
3  class playerShip {
4  public:
5    playerShip();
6    ~playerShip();
7    void moveLeft();
8    void moveRight();
9    void newGame();//перемеща
10
11   const int getX();
12   const int getY();
13
14 private:
15   int x;
16   const static int y = 17;
17 };
```

# Инструкция по эксплуатации

При подключении питания к Arduino Uno с соответствующей периферией, на экране отобразится меню игры в котором: Кнопки A и C осуществляют перемещение курсора между пунктами меню вверх и вниз соответственно. Кнопка B осуществляет выбор соответствующего пункта меню. Пункты меню: New Game - переход к новой игре. Continue - переход к игре, если есть приостановленная игра. Difficulty - выбор сложности игры(от 1 до 3). Чем больше значение, тем быстрее падают ряды. Size - выбор размера игрового поля(от 1 до 3). Значение соответствуют размеру стороны одного блока в пикселях. Highcores - переход к таблице лучших результатов.

Во время игры отображается игровое поле и количество очков. Кнопки отвечают: A и C - движение модели игрока вправо и влево на один пиксель соответственно. D - выстрел блока. B - переход в меню с остановкой текущей игры.

В случае поражения отображается сообщение о проигрыше, количество очков, накопленных игроком за последнею игру, и возможность выбора имя при помощи кнопок: B и D - передвижение курсора вправо и влево соответственно, если курсор стоит на третей букве и нажать B, то случится переход в меню. A и C - изменение соответствующей буквы в имени вверх и вниз по алфавиту соответственно.

В таблице с результатами отображаются лучшие 5 результатов. Кнопки отвечают: A, B, C - переход в меню. D - сброс таблицы результатов на изначальные. При первом запуске следует провести эту процедуру для корректного отображения результатов.

# Окончание