Генератор персонажа Dangeous & Dragons на платформе Arduino

Отчет по научно-исследовательской работе

Выполнил студент 595 группы: Лаптев А. В.
Научный руководитель: Шмаков И.А.

27 июня 2022 г.

Цель и задачи работы

Целью работы: разработка программного продукта под микроконтроллер одного из рассмотренных семейств (AVR, ARM), с использованием выбранной программной и аппаратной платформ.

Задачи работы:

1. Рассмотрение имеющихся аппаратных и программных средств для выбора наиболее подходящей платформы для разработки под AVR и ARM микроконтроллеры.
2. Разработка собственного программного продукта на выбранной аппаратно-программной платформе.

Генерация персонажа DnD

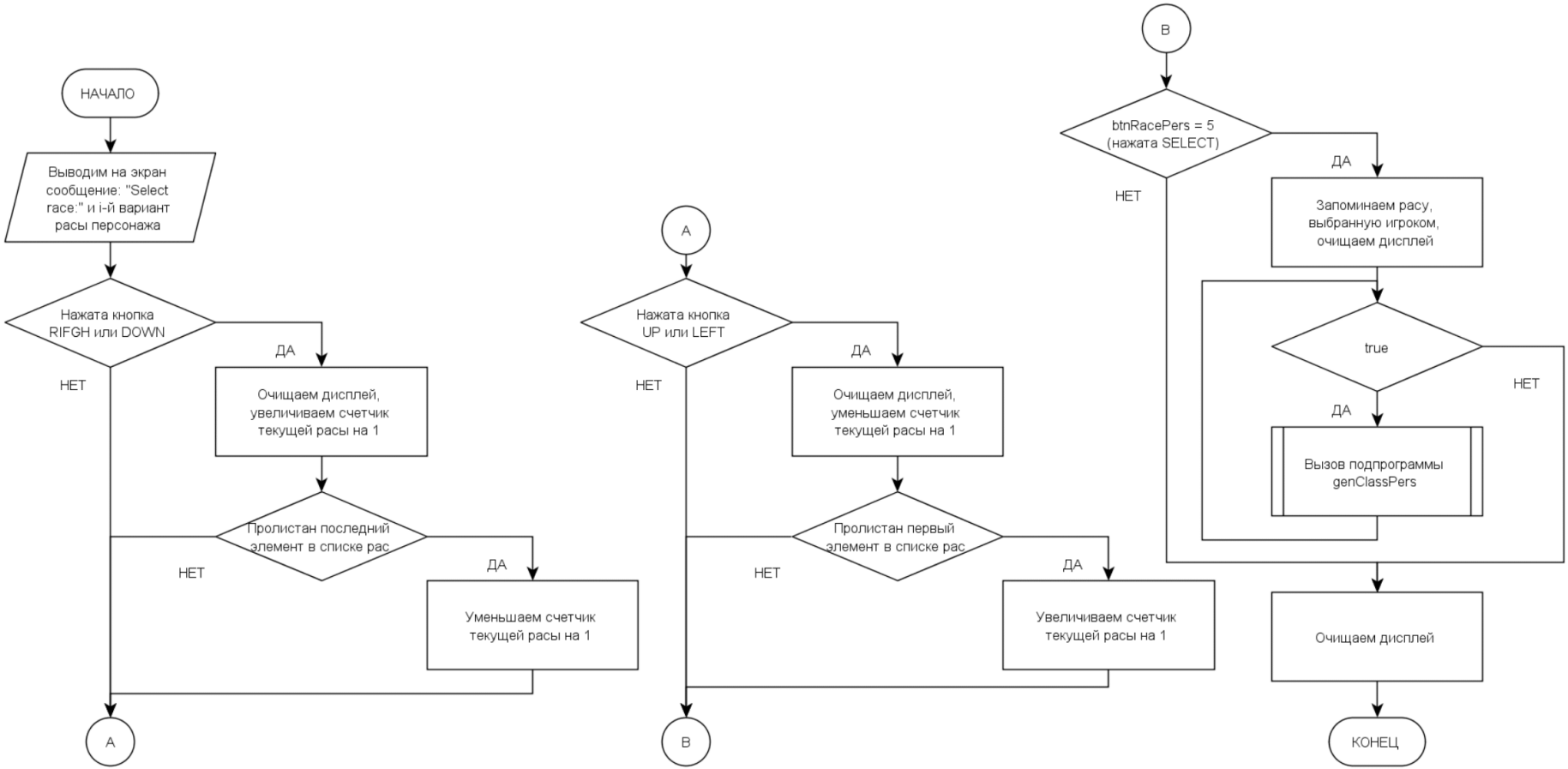
Генерация персонажа представляет собой совокупность параметров персонажа, которые пользователь может выбрать из предложенных вариантов, а также характеристик, которые одинаковы для всех персонажей, но значения которых генерируются случайным образом.

Пользователю предоставляется возможность выбрать то, кем он, непосредственно, хочет быть в игре — раса и класс персонажа. После чего происходит генерация значений для характеристик персонажа. В число которых входят: Сила, Телосложение, Ловкость, Интеллект, Мудрость и Харизма. Помимо этих характеристик, также могут генерироваться класс защиты и хит-поинты.

Реализация генератора персонажа DnD
Подпрограмма mainText

```
lcd.setCursor(0, 1);
lcd.print("Press the 'SELECT' button");
lcd.home();
lcd.print("Character generator DnD");
delay(500);

for (int j = 0; j < 9; j++) {
  btnSel = clickButton();
  if (btnSel == 5)
    break;
  lcd.scrollDisplayLeft();
  delay(500);
}
switch (btnSel){
  case BTN_S:
    randomSeed(millis());
    for (int i = 0; i < 6; i++) {
      roll = 6;
      characteristic = 0;
      for (int j = 0; j < 4; j++) {
        randRoll = random(1, 7);
        if (randRoll < roll) {
          roll = randRoll;
          characteristic += randRoll;
        }
        else
        {
          characteristic += randRoll;
        }
      }
      characteristics[i] = characteristic - roll;
    }
    delay(500);
    lcd.clear();
    while(true) {
      lcd.home();
      genRacePers();
    }
    break;
  default:
    break;
}
lcd.clear();
```

Реализация генератора персонажа DnD
Подпрограмма genRacePers



Реализация генератора персонажа DnD
Подпрограмма genClassPers

```
if(j == 0) {
  lcd.print("Select class:");
  lcd.setCursor(0, 1);
  lcd.print(classPers[j]);
  lcd.createChar(2, arrowDown);
  lcd.setCursor(15, 1);
  lcd.print(char(2));
}
else if(j == 12) {
  lcd.print("Select class:");
  lcd.setCursor(0, 1);
  lcd.print(classPers[j]);
  lcd.createChar(1, arrowUp);
  lcd.setCursor(15, 0);
  lcd.print(char(1));
}
else {
  lcd.print("Select class:");
  lcd.setCursor(0, 1);
  lcd.print(classPers[j]);
  lcd.createChar(1, arrowUp);
  lcd.setCursor(15, 0);
  lcd.print(char(1));
  lcd.createChar(2, arrowDown);
  lcd.setCursor(15, 1);
  lcd.print(char(2));
}
delay(150);

int btnClassPers = clickButton();
switch (btnClassPers){
  case BTN_R:
    lcd.clear();
    j++;
    if (j > 12)
      j--;
    break;
  case BTN_U:
    lcd.clear();
    j--;
    if (j < 0)
      j++;
    break;
  case BTN_D:
    lcd.clear();
    j++;
    if (j > 12)
      j--;
    break;
  case BTN_L:
    lcd.clear();
    j--;
    if (j < 0)
      j++;
    break;
  case BTN_S:
    pers[1] = classPers[j];
    lcd.clear();
    while(true) {
      lcd.home();
      charactPers();
    }
    break;
  default:
    break;
}
lcd.clear();
```

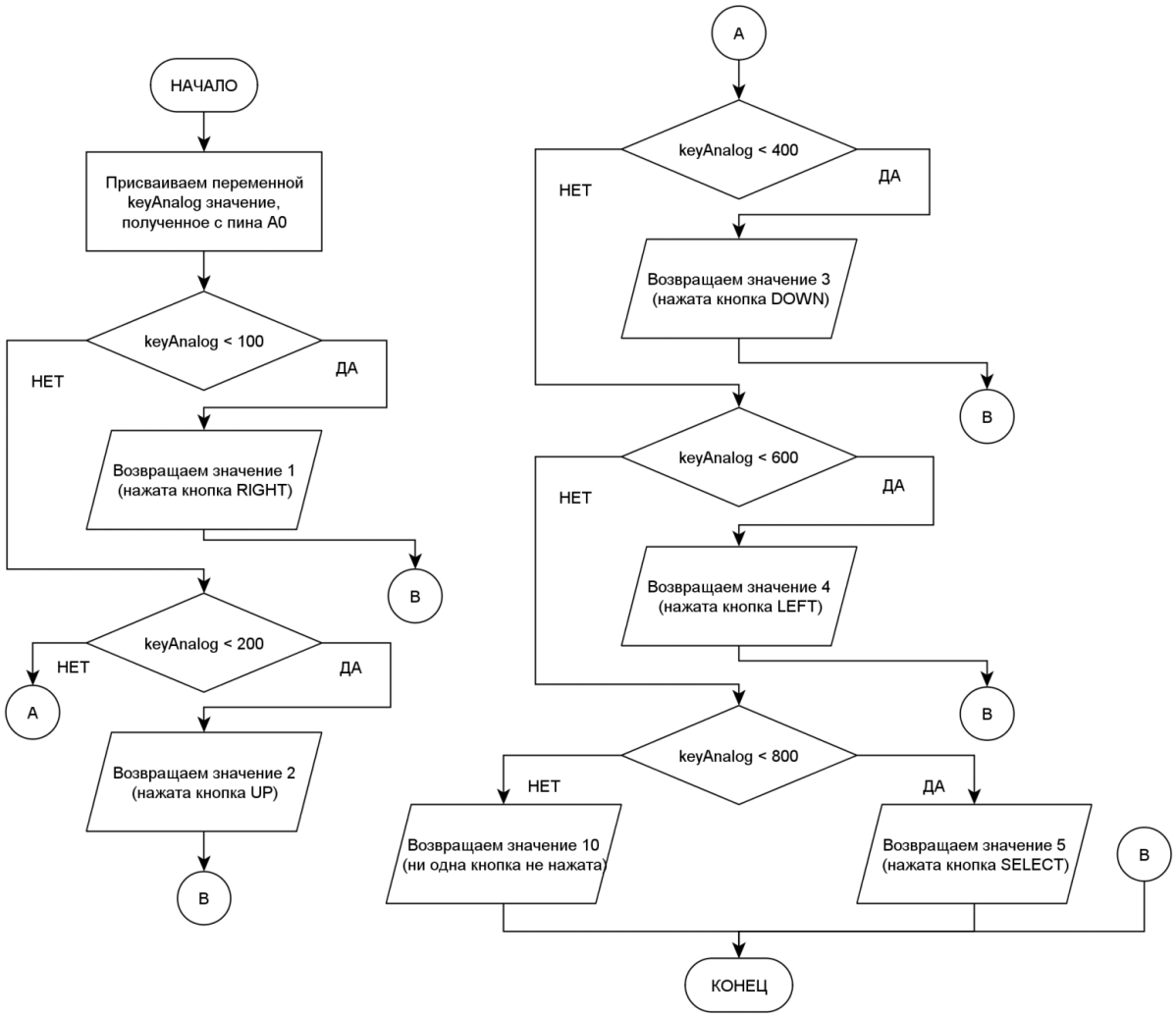
Реализация генератора персонажа DnD
Подпрограмма charactPers

```
if ((pers[1] == classPers[0]) || (pers[1] == classPers[4])
|| (pers[1] == classPers[5]) || (pers[1] == classPers[6])
|| (pers[1] == classPers[7]) || (pers[1] == classPers[8])
|| (pers[1] == classPers[10])) {
  HP = 8 + floor((characteristics[1] - 10) / 2);
}
else if (pers[1] == classPers[1]) {
  HP = 12 + floor((characteristics[1] - 10) / 2);
}
else if ((pers[1] == classPers[2]) || (pers[1] == classPers[9])
|| (pers[1] == classPers[11])) {
  HP = 10 + floor((characteristics[1] - 10) / 2);
}
else {
  HP = 6 + floor((characteristics[1] - 10) / 2);
}
characteristics[6] = HP;

AC = 10 + floor((characteristics[2] - 10) / 2);
characteristics[7] = AC;

if(k == 0) {
  lcd.print("Characteristics:");
  lcd.setCursor(0, 1);
  lcd.print(characteristicsPers[k]);
  lcd.print(pers[k]);
  lcd.createChar(2, arrowDown);
  lcd.setCursor(15, 1);
  lcd.print(char(2));
}
```

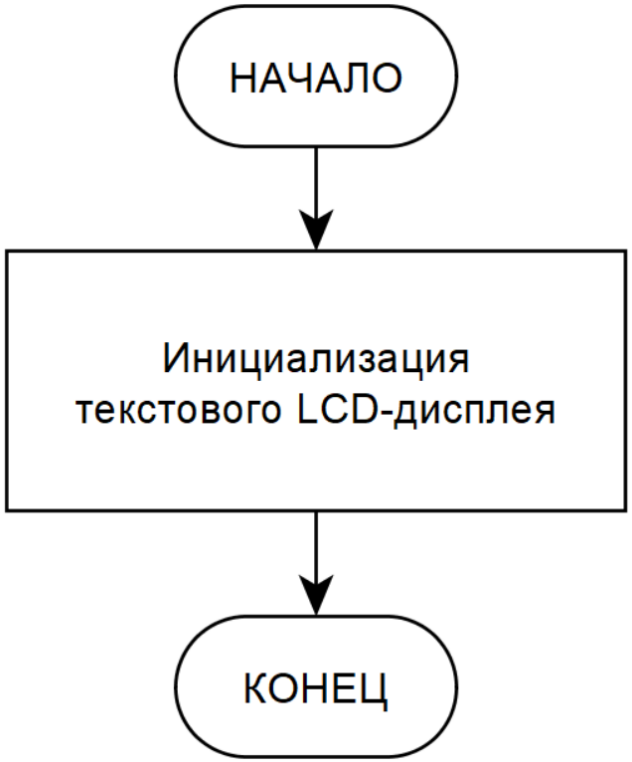
```
else if(k == 1) {
  lcd.print("Characteristics:");
  lcd.setCursor(0, 1);
  lcd.print(characteristicsPers[k]);
  lcd.print(pers[k]);
  lcd.createChar(1, arrowUp);
  lcd.setCursor(15, 0);
  lcd.print(char(1));
  lcd.createChar(2, arrowDown);
  lcd.setCursor(15, 1);
  lcd.print(char(2));
}
else if(k == 9) {
  lcd.print("Characteristics:");
  lcd.setCursor(0, 1);
  lcd.print(characteristicsPers[k]);
  lcd.print(characteristics[k - 2]);
  lcd.createChar(1, arrowUp);
  lcd.setCursor(15, 0);
  lcd.print(char(1));
}
else {
  lcd.print("Characteristics:");
  lcd.setCursor(0, 1);
  lcd.print(characteristicsPers[k]);
  lcd.print(characteristics[k - 2]);
  lcd.createChar(1, arrowUp);
  lcd.setCursor(15, 0);
  lcd.print(char(1));
  lcd.createChar(2, arrowDown);
  lcd.setCursor(15, 1);
  lcd.print(char(2));
}
delay(150);
```

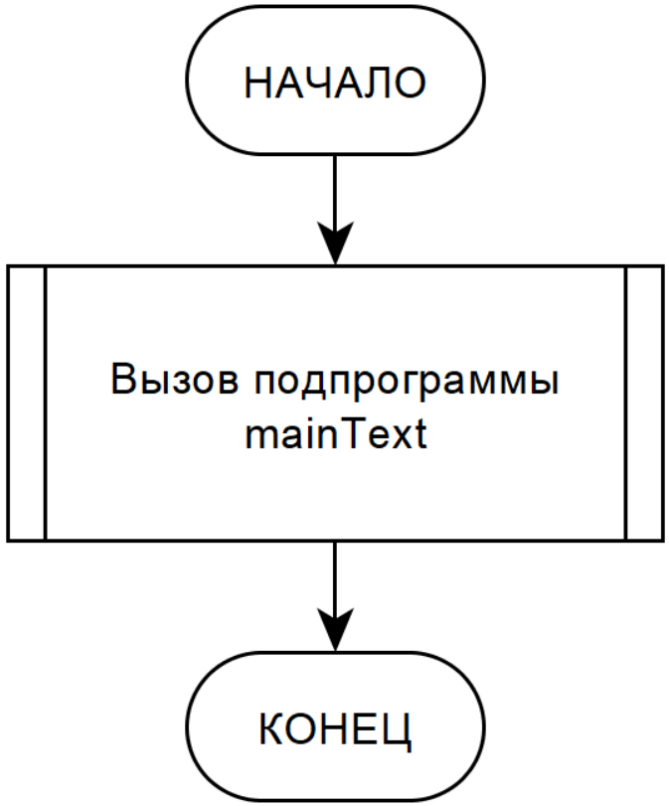
```
int btnCharPers = clickButton();
switch (btnCharPers){
  case BTN_R:
    lcd.clear();
    k++;
    if (k > 9)
      k--;
    break;
  case BTN_U:
    lcd.clear();
    k--;
    if (k < 0)
      k++;
    break;
  case BTN_D:
    lcd.clear();
    k++;
    if (k > 9)
      k--;
    break;
  case BTN_L:
    lcd.clear();
    k--;
    if (k < 0)
      k++;
    break;
  default:
    break;
}
lcd.clear();
```

Реализация генератора персонажа DnD
Подпрограмма clickButton



Реализация генератора персонажа DnD
Подпрограммы setup и loop





Недостатки реализации генератора персонажа DnD

Встроенные функции в Arduino IDE зачастую имеют в своем составе ряд дополнительных проверок, чтобы минимизировать количество возможных ошибок, которые могут появиться при компиляции.

Также, довольно большое количество глобальных переменных, а также их не оптимальная типизация, из-за которой они могут занимать в памяти больше места, чем им требуется в действительности, также увеличивают количество потребляемых ресурсов микроконтроллера.

Помиом этого, на работу программы негативное влияние оказывает использование встроенной функции задержки delay, поскольку, при использовании этой функции, приостанавливается работа всей программы.

Заключение

В ходе выполнения научно-исследовательской работы было проведено знакомство с различными платформами для разработки под микроконтроллеры, проведен обзор различных семейств микроконтроллеров и выбрана платформа для разработки собственного программного продукта — генератора персонажа DnD.

Был реализован основной функционал:

1. наличие возможности выбора расы и класса персонажа;
2. генерация случайных значений для характеристик персонажа, согласно правилам DnD;
3. расчет некоторых дополнительных характеристик персонажа с использованием сгенерированных для основных характеристик значений.

В результате, был создан вариант генератора персонажа DnD.