**Arduino Racing Game**

Гоночная игра на Arduino

СОДЕРЖАНИЕ

1 Цель проекта 3

2 [Геймплей 3](#_Toc116889383)

[3 Необходимые компоненты 3](#_Toc116889384)

[4 Процесс сборки 4](#_Toc116889384)

[5 Программа 7](#_Toc116889384)

[6 Итоги 17](#_Toc116889384)

1. **Цель проекта**

Создать интересную и уникальную игру для развлечения, повысив свои навыки.

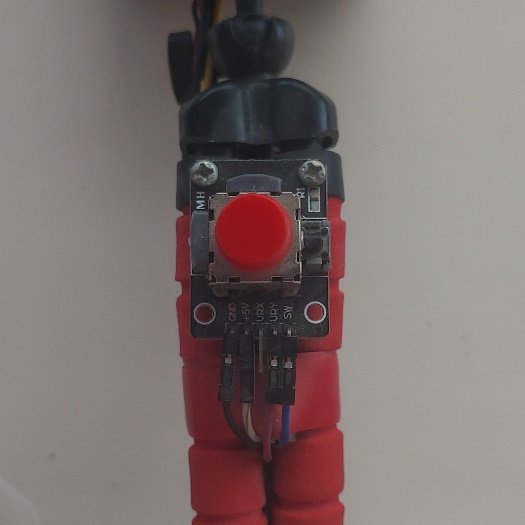
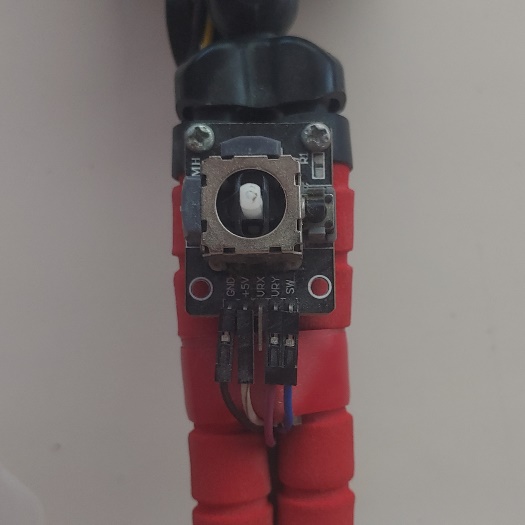
1. **Геймплей**

Игрок управляет машиной, на машину движутся препятствия, которые игроку необходимо объезжать. Есть несколько разных уровней сложности и возможность посмотреть рекорды. Видео геймплея: <https://www.youtube.com/watch?v=zKWT0bUBfUk>

**3. Необходимые компоненты**

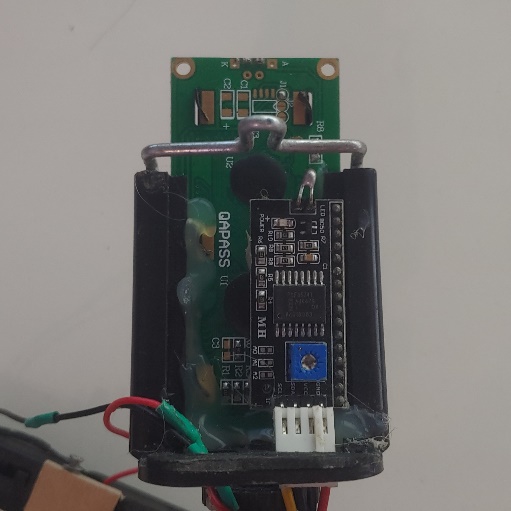
1. Arduino Nano.
2. Экран 16х2.
3. Джойстик.
4. Батарейный отсек.
5. 8 соединительных проводов.
6. Штатив для телефона.
7. Антистатический пакет.
8. 2 винта/самореза.
9. Изолента.
10. Термоклей.
11. **Процесс сборки**
12. Джойстик.

Берём штатив, и присоединяем к нему джойстик с помощью двух винтов или саморезов. Чтобы было проще, можно шуруповертом заранее сделать отверстия в пластике.



1. Экран.

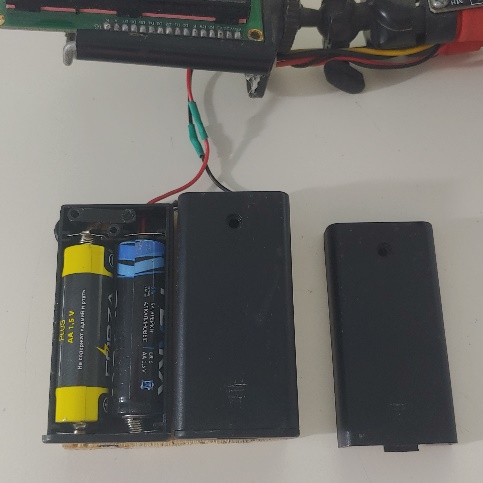
Разворачиваем прищепку штатива чтобы задняя часть смотрела на нас. Экран сюда походит по размерам как влитой, разве что может потребоваться немного подрезать пластик чтобы влезли контакты i2c модуля. Резать пластик удобно гравером, но можно использовать и пилу, паяльник, дрель или шуруповерт. Приклеиваем экран с обратной стороны на термоклей, чтобы не отвалился.



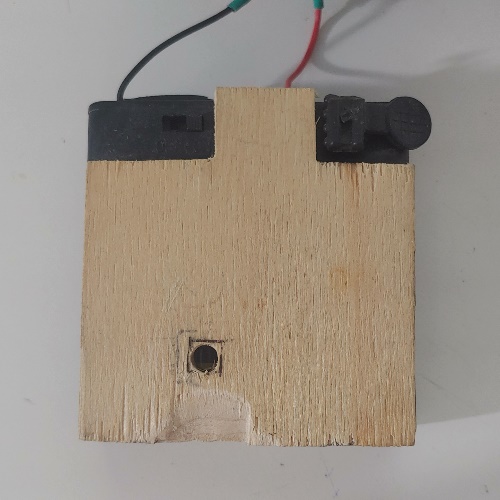
1. Батарейный отсек.

Лучше использовать батарейный отсек со встроенным переключателем. Если его нет – ничего страшного, можно будет выключать игру доставанием батарейки. Также лучше использовать отсек для батареек типа ААА, а не типа АА, так как они легче, но и менять их нужно чаще. Еще можно приклеить батарейный отсек к корпусу штатива, но я не увидел в этом необходимости.

В магазине не было отсека на 4 батарейки, поэтому мне пришлось купить два отсека на 2 батарейки АА и соединить их. Я приклеил их универсальным клеем к куску фанеры нужного размера и припаял плюс одного отсека к минусу другого. Теперь они как один отсек.

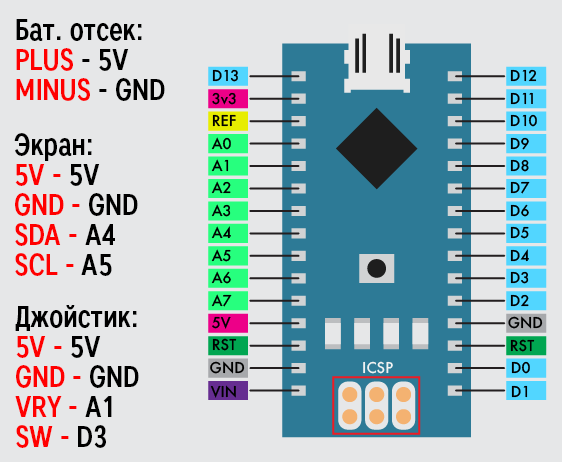


Также я сделал два выреза в фанере для переключателей. У штатива сверху была резиновая “пимпочка”, потянув за которую можно растягивать прищепку. Нам прищепку растягивать не требуется, поэтому я нашёл ей другое применение – приклеил её к одному из переключателей, чтобы было легче включать и выключать игру. Еще я сделал отверстие в фанере и батарейном отсеке чтобы можно было регулировать яркость экрана потенциометром, не снимая при этом батарейный отсек.



1. Arduino

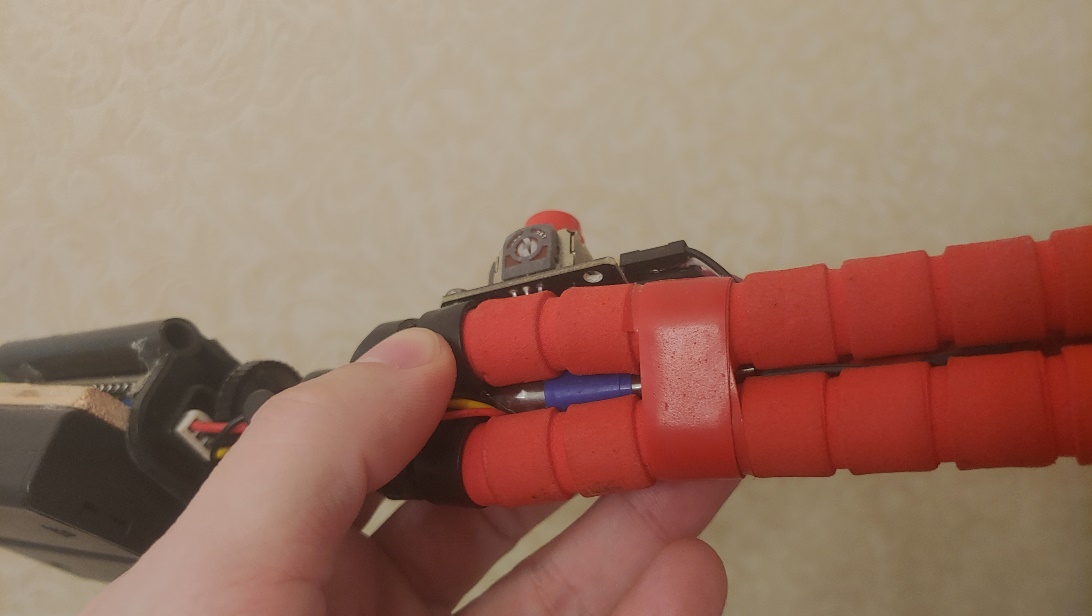
Берем Arduino и припаиваем провода к ней согласно схеме. Прошиваем программу в Arduino.



1. Финиш

Если всё работает корректно, то можем заканчивать сборку и наслаждаться результатом. Заворачиваем Arduino в отрезанный под её размер кусок антистатического пакета и заматываем изолентой.

Помещаем её между ножками штатива и заматываем изолентой ножки чтобы они не разъезжались в стороны. Ножки бывают разных цветов, и лучше использовать изоленту того же цвета. Игра готова.



**5. Программа**

Импорт библиотек и инициализация глобальных переменных. Также вам потребуется установить библиотеку LiquidCrystal\_I2C.

1. #include <EEPROM.h> // Библиотека для работы с памятью
2. #include <Wire.h> // Библиотека для работы с i2c
3. #include <LiquidCrystal\_I2C.h> // Библиотека для работы с дисплеем
4. LiquidCrystal\_I2C lcd(0x27, 16, 2); // Инициализация дисплея
5. uint8\_t b[8]={0x1f, 0x19, 0x15, 0x13, 0x19, 0x15, 0x13, 0x1f}; // Графическая модель препятствия
6. uint8\_t Coords[16]; // В этом массиве хранится информация о местонахождении препятствий, 0 – нет препятствия, 1 – есть слева, 2 – есть справа
7. int last=0; // Последнее положение препятствия, переменная нужна чтобы предотвратить спавн препятствия сразу после следующего чтобы дать игроку возможность проехать между ними
8. int last2=0; // Эта переменная означает количество свободного места
9. unsigned long int points=0; // Счетчик очков игрока
10. int i,j,c; // В переменной с храниться информация о том где заспавнится препятствие, 0 – нигде, 1 – слева, 2 -
11. uint8\_t pos=0; // Текущая позиция машины, 0 – слева, 1 - справа
12. uint8\_t car1[8]={0x18,0x1e,0x5,0x19,0x19,0x5,0x1e,0x18}; // Графическая модель передней части
13. uint8\_t car2[8]={0xc,0xf,0x14,0x13,0x13,0x14,0xf,0xc}; // Графическая модель задней части машины
14. unsigned long int e; // Последнее сохраненное значение переменной millis()
15. int velocity, sectors; // Эти параметры зависят от сложности, velocity – скорость, sectors – минимальное расстояние между двумя препятствиями
16. int dif; // Сложность

Функция, обновляющая положение машины на дисплее

1. void printcar(){
2. if (pos==0){
3. lcd.setCursor(1,0);
4. lcd.write(1);
5. lcd.setCursor(0,0);
6. lcd.write(2);
7. lcd.setCursor(0,1);
8. lcd.print(" ");
9. lcd.setCursor(1,1);
10. lcd.print(" ");
11. }
12. else{
13. lcd.setCursor(1,1);
14. lcd.write(1);
15. lcd.setCursor(0,1);
16. lcd.write(2);
17. lcd.setCursor(0,0);
18. lcd.print(" ");
19. lcd.setCursor(1,0);
20. lcd.print(" ");
21. }
22. }

Функция, срабатывающая при изменении положения джойстика. Она проверяет есть ли препятствие на полосе, на которую игрок хочет перестроится. Если есть, то возвращает 0 и игра окончена. Если нет, вызывает функцию printcar.

Функция, перемещающая все объекты на 1 клетку ближе к игроку

1. int turn(){
2. if (analogRead(A1)>640){
3. if (pos==0){
4. if (pos==0){
5. if ((Coords[0]==2)||(Coords[1]==2)){
6. return 0;
7. }
8. }
9. else{
10. if ((Coords[0]==1)||(Coords[1]==1)){
11. return 0;
12. }
13. }
14. pos=1;
15. printcar();
16. }
17. }
18. else if(analogRead(A1)<384){
19. if (pos==1){
20. if (pos==0){
21. if ((Coords[0]==2)||(Coords[1]==2)){
22. return 0;
23. }
24. }
25. else{
26. if ((Coords[0]==1)||(Coords[1]==1)){
27. return 0;
28. }
29. }
30. pos=0;
31. printcar();
32. }
33. }
34. return 1;
35. }

Эта функция вызывается каждый такт и перемещает все препятствия на 1 клетку

1. void moving(){
2. for (i=0;i<3;i++){
3. if (pos==0){
4. if (Coords[i]!=Coords[i+1]){
5. Coords[i]=Coords[i+1];
6. if (Coords[i]==1){
7. lcd.setCursor(i,1);
8. lcd.print(" ");
9. }
10. else if (Coords[i]==2){
11. lcd.setCursor(i,1);
12. lcd.write(0);
13. }
14. else{
15. lcd.setCursor(i,1);
16. lcd.print(" ");
17. }
18. }
20. }
21. else{
22. if (Coords[i]!=Coords[i+1]){
23. Coords[i]=Coords[i+1];
24. if (Coords[i]==1){
25. lcd.setCursor(i,0);
26. lcd.write(0);
27. }
28. else if (Coords[i]==2){
29. lcd.setCursor(i,0);
30. lcd.print(" ");
31. }
32. else{
33. lcd.setCursor(i,0);
34. lcd.print(" ");
35. }
36. }
37. }
38. }
39. for (i=2;i<15;i++){
40. if (Coords[i]!=Coords[i+1]){
41. Coords[i]=Coords[i+1];
42. if (Coords[i]==1){
43. lcd.setCursor(i,0);
44. lcd.write(0);
45. lcd.setCursor(i,1);
46. lcd.print(" ");
47. }
48. else if (Coords[i]==2){
49. lcd.setCursor(i,1);
50. lcd.write(0);
51. lcd.setCursor(i,0);
52. lcd.print(" ");
53. }
54. else{
55. lcd.setCursor(i,0);
56. lcd.print(" ");
57. lcd.setCursor(i,1);
58. lcd.print(" ");
59. }
60. }
61. }
62. }

Если препятствие заспавнится на другой стороне сразу после предыдущего препятствия, то получится ситуация в которой у игрока заблокируется проезд. В этой ситуации вызывается эта функция и передвигает все объекты вперед перед тем как отобразить следующий блок.

1. int movingwithoutrand(){
2. for (j=0;j<sectors;j++){
3. if (pos+1==Coords[2]){
4. return 0;
5. }
6. moving();
7. Coords[15]=0;
8. lcd.setCursor(15,0);
9. lcd.print(" ");
10. lcd.setCursor(15,1);
11. lcd.print(" ");
12. if (turn()==0){
13. return 0;
14. }
15. points++;
16. e=millis();
17. while(millis()-e<velocity){
18. if (turn()==0){
19. return 0;
20. }
21. }
23. }
24. return 1;
25. }

Функция изменения уровня сложности в меню показа максимальных набранных очков

1. void dispscore(){
2. byte val1 = EEPROM.read((dif-1)\*4+1);
3. byte val2 = EEPROM.read((dif-1)\*4+2);
4. byte val3 = EEPROM.read((dif-1)\*4+3);
5. byte val4 = EEPROM.read((dif-1)\*4+4);
6. unsigned long int val = val1\*pow(2,24)+val2\*pow(2,16)+val3\*(pow(2,8))+val4;
7. switch (dif){
8. case 1:
9. lcd.clear();
10. lcd.setCursor(0,0);
11. lcd.print(" TripMode > ");
12. break;
13. case 2:
14. lcd.clear();
15. lcd.setCursor(0,0);
16. lcd.print(" < Easy > ");
17. break;
18. case 3:
19. lcd.clear();
20. lcd.setCursor(0,0);
21. lcd.print(" < Normal > ");
22. break;
23. case 4:
24. lcd.clear();
25. lcd.setCursor(0,0);
26. lcd.print(" < Hard > ");
27. break;
28. case 5:
29. lcd.clear();
30. lcd.setCursor(0,0);
31. lcd.print("< ImpossibleFast");
32. break;
33. }

Функция интерфейса меню просмотра максимальных очков.

1. void maxscores(){ // "difficulty" procedure calls it when user picks up "Max Score" option on the menu.
2. dif=3;
3. lcd.clear();
4. lcd.setCursor(3,0);
5. lcd.print("< Normal >");
6. lcd.setCursor(0,1);
7. byte val1 = EEPROM.read((dif-1)\*4+1);
8. byte val2 = EEPROM.read((dif-1)\*4+2);
9. byte val3 = EEPROM.read((dif-1)\*4+3);
10. byte val4 = EEPROM.read((dif-1)\*4+4);
11. unsigned long int val = val1\*pow(2,24)+val2\*pow(2,16)+val3\*(pow(2,8))+val4;
12. lcd.print(val);
13. while (1){
14. if ((analogRead(A1)>640) && (dif<5)){
15. dif++;
16. dispscore();
17. while(analogRead(A1)>640){
18. delay(5);
19. }
20. }
21. else if ((analogRead(A1)<384) && (dif>1)){
22. dif--;
23. dispscore();
24. while(analogRead(A1)<384){
25. delay(5);
26. }
27. }
28. else if (digitalRead(3)==0){
29. while(digitalRead(3)==0){
30. delay(5);
31. }
32. break;
33. }
35. }
36. difficulty();
37. }

Функция меняет название уровня сложности на экране и устанавливает параметры для выбранного уровня.

1. void dispdif(){
2. switch (dif){
3. case 0:
4. lcd.setCursor(0,1);
5. lcd.print(" Max Scores > ");
6. break;
7. case 1:
8. lcd.setCursor(0,1);
9. lcd.print(" < TripMode > ");
10. velocity=200;
11. sectors=3;
12. break;
13. case 2:
14. lcd.setCursor(0,1);
15. lcd.print(" < Easy > ");
16. velocity=150;
17. sectors=3;
18. break;
19. case 3:
20. lcd.setCursor(0,1);
21. lcd.print(" < Normal > ");
22. velocity=100;
23. sectors=3;
24. break;
25. case 4:
26. lcd.setCursor(0,1);
27. lcd.print(" < Hard > ");
28. velocity=150;
29. sectors=2;
30. break;
31. case 5:
32. lcd.setCursor(0,1);
33. lcd.print("< ImpossibleFast");
34. velocity=100;
35. sectors=2;
36. break;
37. }
38. }

Функция интерфейса меню выбора сложности.

1. void difficulty(){ // It is called before race is started to let user set the difficulty
2. dif=3;
3. velocity=100;
4. sectors=3;
5. lcd.clear();
6. lcd.setCursor(0,0);
7. lcd.print("Difficulty");
8. lcd.setCursor(3,1);
9. lcd.print("< Normal >");
10. while (1){
11. if ((analogRead(A1)>640) && (dif<5)){
12. dif++;
13. dispdif();
14. while(analogRead(A1)>640){
15. delay(5);
16. }
17. }
18. else if ((analogRead(A1)<384) && (dif>0)){
19. dif--;
20. dispdif();
21. while(analogRead(A1)<384){
22. delay(5);
23. }
24. }
25. else if (digitalRead(3)==0){
26. while(digitalRead(3)==0){
27. delay(5);
28. }
29. if (dif>0){
30. break;
31. }
32. else{
33. maxscores();
34. break;
35. }
36. }
38. }
39. }

Функция setup.

1. void setup() {
2. lcd.begin();
3. lcd.backlight();
4. Serial.begin(9600);
5. lcd.createChar(0, b);
6. lcd.createChar(1,car1);
7. lcd.createChar(2,car2);
8. //lcd.createChar(1, c);
10. lcd.setCursor(0,0);
11. lcd.print("Racing game");
12. lcd.setCursor(0,1);
13. lcd.print("Press the button");
14. pinMode(3,INPUT\_PULLUP);
15. }

Функция loop. Она выполняется по кругу и управляет всем процессом игры.

1. void loop() {
2. if (digitalRead(3)==0){
3. while (1){
4. if (digitalRead(3)==1) break;
5. delay(5);
6. }
7. points=0;
8. difficulty();
9. randomSeed(millis() % 1000);
10. lcd.clear();
11. for (i=0;i<15;i++){
12. Coords[i]=0;
13. }
14. printcar();
15. while(1){
17. c=random(3);
18. if (c!=0 & c!=last & last!=0 & last2<4){
19. if (movingwithoutrand()==0){
20. break;
21. }
22. }
23. if (pos+1==Coords[2]){
24. break;
25. }
26. moving();
27. Coords[15]=c;
28. if (Coords[15]==1){
29. lcd.setCursor(15,0);
30. lcd.write(0);
31. lcd.setCursor(15,1);
32. lcd.print(" ");
33. }
34. else if (Coords[15]==2){
35. lcd.setCursor(15,1);
36. lcd.write(0);
37. lcd.setCursor(15,0);
38. lcd.print(" ");
39. }
40. else{
41. lcd.setCursor(15,0);
42. lcd.print(" ");
43. lcd.setCursor(15,1);
44. lcd.print(" ");
45. }
46. if (Coords[15]!=0){
47. last=Coords[15];
48. last2=0;
49. }
50. else{
51. last2++;
52. }
53. if (turn()==0){
54. break;
55. }
56. points++;
57. e=millis();
58. while(millis()-e<velocity){
59. if (turn()==0){
60. break;
61. }
62. }
63. }
64. lcd.clear();
65. lcd.setCursor(0,0);
66. lcd.print("Game over");
67. if (points<10000){
68. lcd.setCursor(0,1);
69. lcd.print("Your points:");
70. lcd.setCursor(12,1);
71. lcd.print(points);
72. }
73. else if (points<1000000){
74. lcd.setCursor(0,1);
75. lcd.print("Ur points:");
76. lcd.setCursor(10,1);
77. lcd.print(points);
78. }
79. else if (points<1000000000){
80. lcd.setCursor(0,1);
81. lcd.print("Points:");
82. lcd.setCursor(7,1);
83. lcd.print(points);
84. }
85. else{
86. lcd.setCursor(0,1);
87. lcd.print("Find a girlfriend");
88. }
89. byte val1 = EEPROM.read((dif-1)\*4+1);
90. byte val2 = EEPROM.read((dif-1)\*4+2);
91. byte val3 = EEPROM.read((dif-1)\*4+3);
92. byte val4 = EEPROM.read((dif-1)\*4+4);
93. unsigned long int power = 2;
94. unsigned long int val = val1\*pow(2,24)+val2\*pow(2,16)+val3\*(pow(2,8))+val4;
95. if (points>val){
96. if (points>=1000000000){
97. points=999999999;
98. }
99. val1=0;
100. val2=0;
101. val3=0;
102. val4=0;
103. val4=val4+bitRead(points, 0);
104. for (i=1;i<8;i++){
105. val4=val4+power\*bitRead(points, i);
106. power\*=2;
107. }
108. val3=val3+bitRead(points, 8);
109. power=2;
110. for (i=9;i<16;i++){
111. val3=val3+power\*bitRead(points, i);
112. power\*=2;
113. }
114. power=2;
115. val2=val2+bitRead(points, 16);
116. for (i=17;i<24;i++){
117. val2=val2+power\*bitRead(points, i);
118. power\*=2;
119. }
120. power=2;
121. val1=val1+bitRead(points, 24);
122. for (i=25;i<29;i++){
123. val1=val1+power\*bitRead(points, i);
124. power\*=2;
125. }
126. EEPROM.write((dif-1)\*4+1, val1);
127. EEPROM.write((dif-1)\*4+2, val2);
128. EEPROM.write((dif-1)\*4+3, val3);
129. EEPROM.write((dif-1)\*4+4, val4);
130. }
131. while (1){
132. if (digitalRead(3)==0){
133. while(digitalRead(3)==0){
134. delay(5);
135. }
136. break;
137. }
138. }
139. lcd.setCursor(0,0);
140. lcd.print("Racing game");
141. lcd.setCursor(0,1);
142. lcd.print("Press the button");
144. }
145. }

**6. Итоги**

Вы получили опыт взаимодействия со средой Arduino и интересную, уникальную, самодельную игру. Если что-то не работает – проверьте работоспособность каждого модуля по отдельности и качество пайки. Оценить мой результат можно по ссылке: <https://www.youtube.com/watch?v=zKWT0bUBfUk>