SPECIAL CAR

Документация

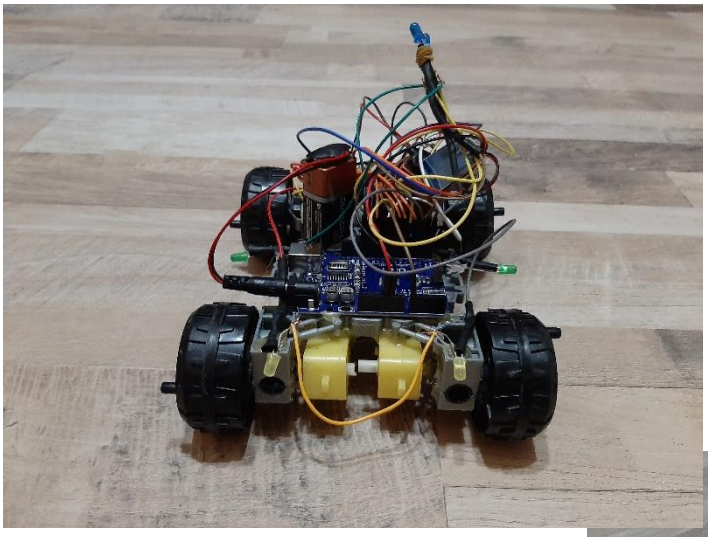
Изготвили:
Цветилин Цветилов,
Стоян Златев,
Павлин Маринов

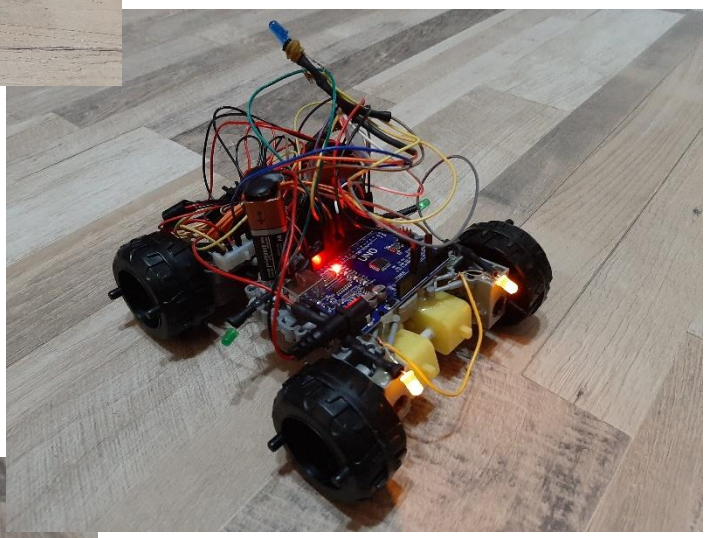
Съдържание

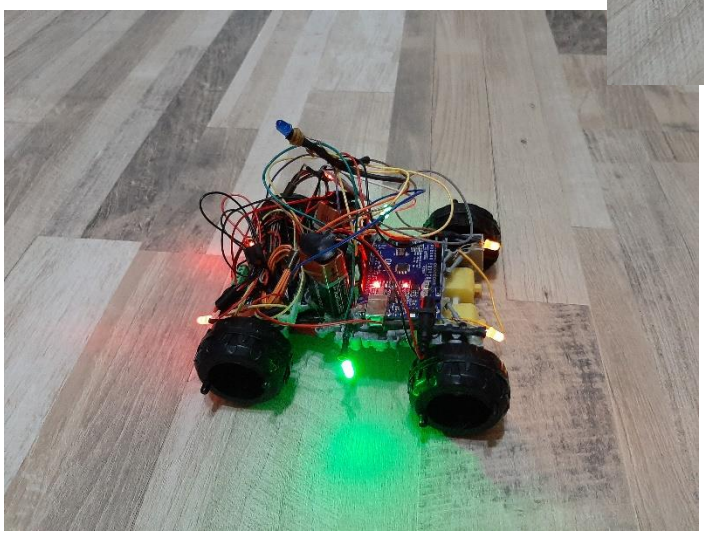
Описание на проекта

Special Car е количка, управлявана от смартфон чрез bluetooth връзка, осъществена чрез HC-06 модул. Количката се задвижва от два 1.5 волтови DC мотора. Те се управляват от L293D модул, а цялата система се упрвлява с “Arduino Uno” платка. Включването и изключването на захранването се осъществява чрез Rocker Switch.
Количката разполага с 8 LED светлини: 2 за фарове, 2 за мигачи, 2 за стопове и 2 за специални светлини.

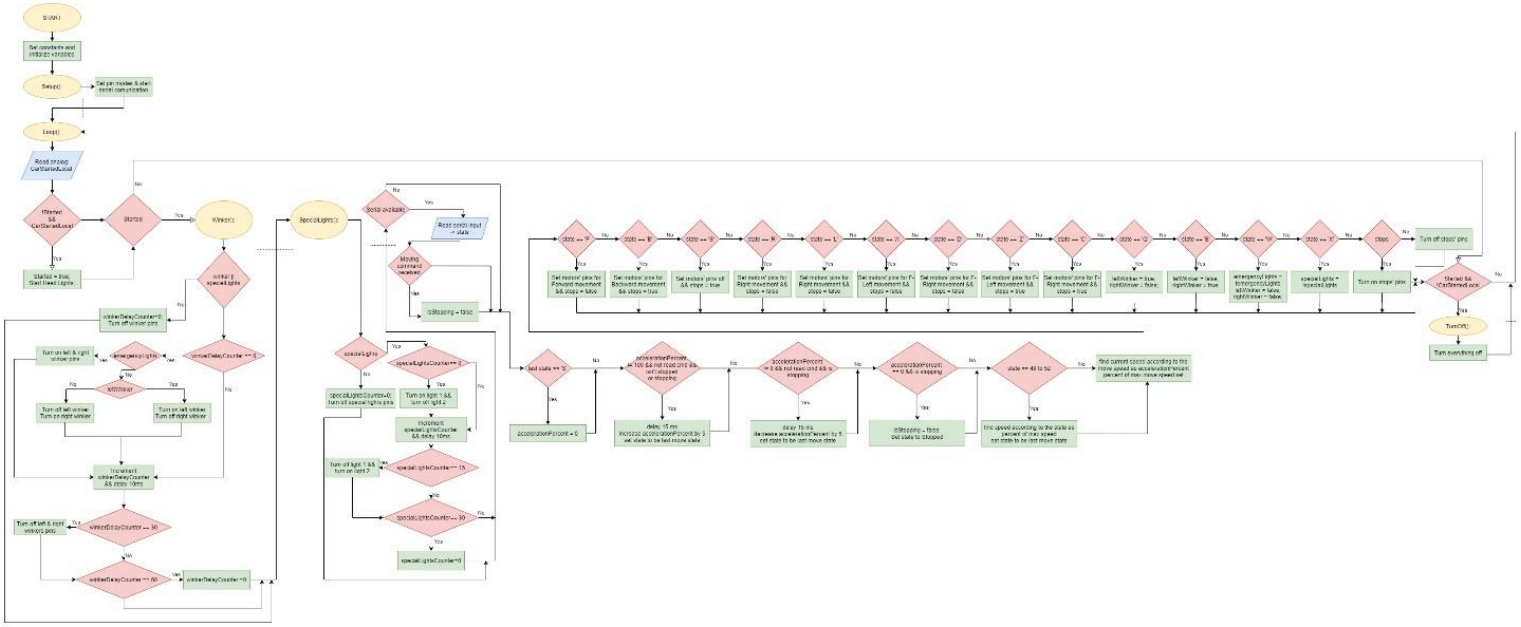
При стартиране на захранването фаровете светват мигновено, а при маневра назад или при аварийно спиране стоповете автоматично се активират до приключване на маневрата. При потегляне управляващият може да регулира скоростта, с която да се движи количката от мобилното приложение като има 4 опции - 25%, 50%, 75% и 100% от максималният скоростен потенциал на количката. При тръгване колата ускорява плавно до достигане на максимума на избраната скорост. При отпускане на подадената команда за потегляне (напр. при отпускане на бутон Forward) колата постепенно намаля скоростта докато не е налице пълно спиране или не е подадена друга команда. При натискане на бутон СТОП моторите незабавно прекратяват движението си.





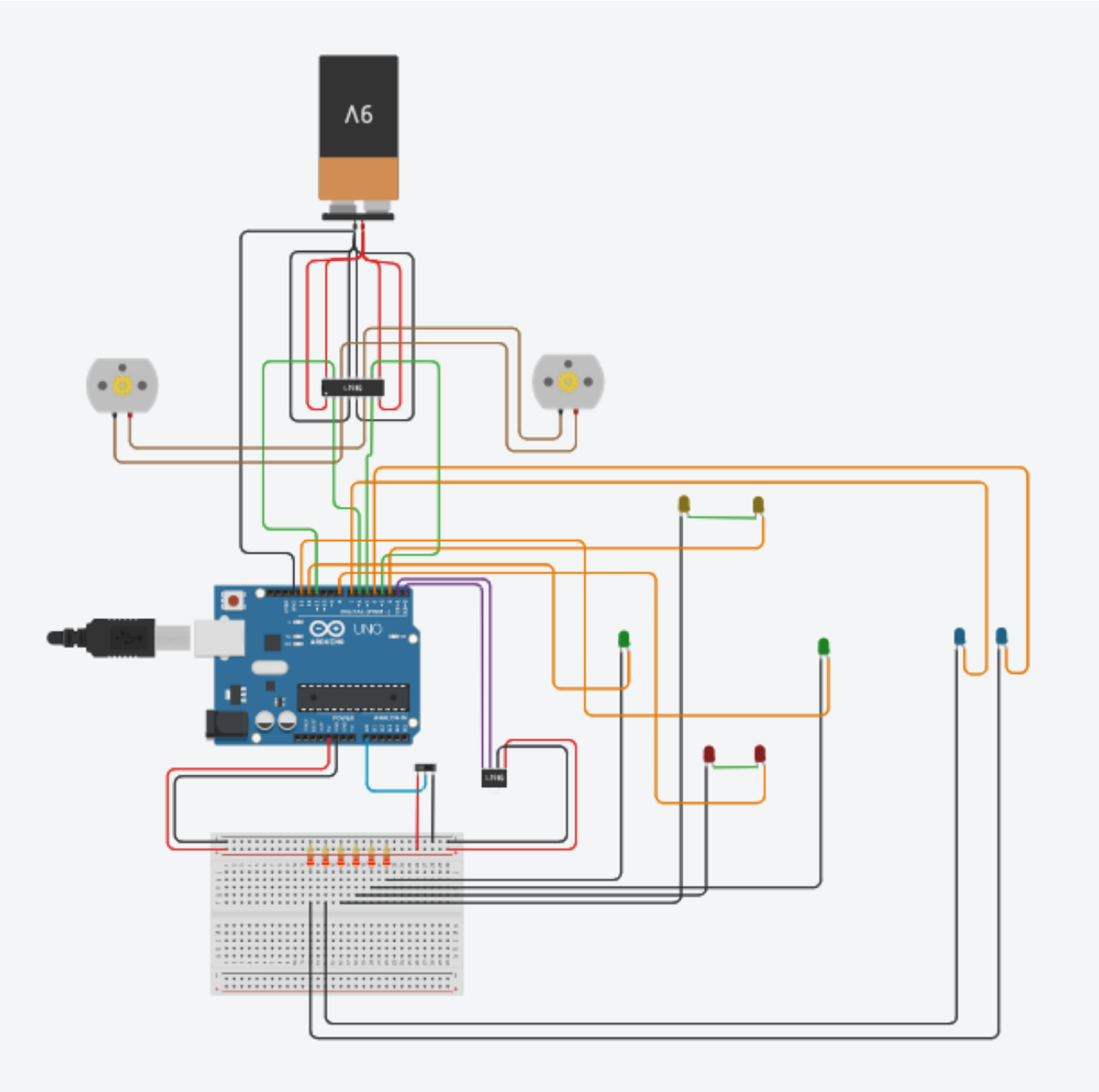


Блокова схема



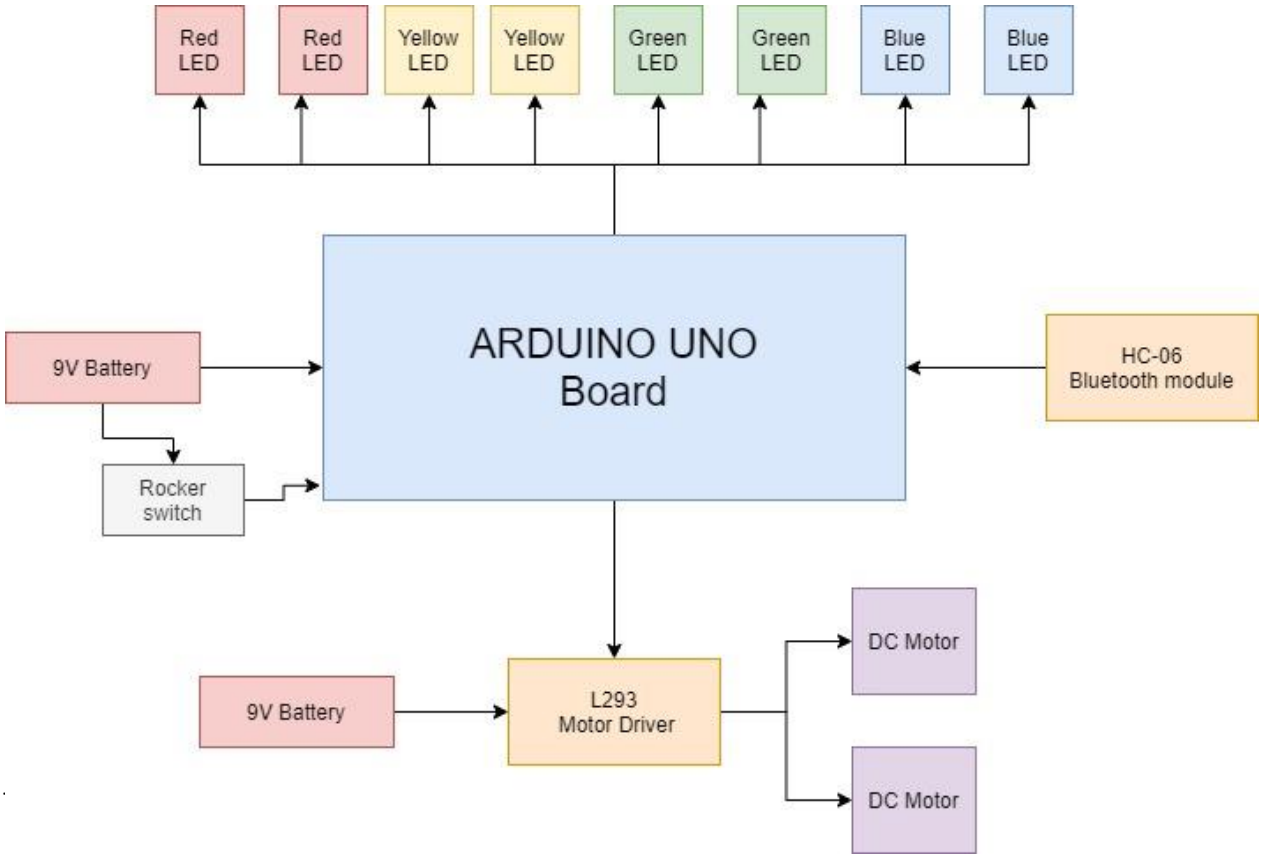
- Функцията setup() се изпълнява еднократно в началото и задава режима на пиновете и инициира серийна комуникация.
- Функцията loop() се изпълнява непрекъснато – до спирането на захранването. В нея се извършват всички промени по състоянието на робота.
- Функциите SpecialLights() и Winkers() управляват пътните показатели на прототипа, посредством своеобразен брояч.

Електрическа схема



Списък съставни части

1. Arduino Uno;
2. L293D Motor Driver;
3. 2x 9V Battery;
4. Bluetooth module HC-06;
5. 2x DC Motor;
6. Rocker Switch;
7. 8x LED(2xGreen, 2xYellow, 2xRed, 2xBlue);
8. Copper Jumper Wires;
9. 8x Resistors(220Ω);
10. Breadboard;
11. 4x Tyres;
12. Chassis



Сорс код - описание на функционалността

За менажиране на състоянието на робота запазваме в променливи съответно дали колата работи, дали мигачите или аварийните светлини са пуснати, дали стоповете трябва да светят, както и специалните светлини, указващи специалния режим на движение на нашето "МПС".

За допълнителна прецизност и симулация на реален модел използваме постепенно ускорение и намаляне преди спиране, както и регулиране на скоростта на движение. Запазваме последната команда за движение.

За определяне на периода на премигване на светлините, чиято нормална функция го изисква, използваме два брояча - за пътепоказателите и за специалните светлини.

```
bool started = false;
bool leftWinker = false;
bool rightWinker = false;
bool emergencyLight = false;
bool stops = false;
bool specialLights = false;

int moveSpeed = 125;
int lastState = -1;
int percentAcceleration = 0;
bool isStopping = false;

int winkerDelayCounter = 0;
int specialLightsCounter  = 0;
```

За по-удобна употреба на номерата на пиновете, те са записани в константи

```
const int motor1Pin1 = 5;
const int motor1Pin2 = 6;
const int motor2Pin1 = 11;
const int motor2Pin2 = 3;
const int switchPin = A0;
const int leftWinkerPin= 12;
const int headLightsPin = 2;
const int backLightsPin = 8;
```

```
const int rightWinkerPin= 13;
const int specialLights1Pin = 4;
const int specialLights2Pin = 7;
```

При захранването на платката с ток еднократно задаваме вида на комуникация на пина: пиновете за комуникация с блутут модула - RX и TX - не се обозначават; аналоговия вход - A0, четящ състоянието на бутона; PWM пиновете, контролиращи двигателите чрез драйвера; другите пинове, контролиращи LED-овете. Инициира се серийна комуникация с блутут модула със скорост на предаване 9600 бита в секунда.

```
void setup()
{
   pinMode(motor1Pin1, OUTPUT);
   pinMode(motor1Pin2, OUTPUT);
   pinMode(motor2Pin1, OUTPUT);
   pinMode(motor2Pin2, OUTPUT);
   pinMode(leftWinkerPin, OUTPUT);
   pinMode(headLightsPin, OUTPUT);
   pinMode(backLightsPin, OUTPUT);
   pinMode(rightWinkerPin, OUTPUT);
   pinMode(specialLights1Pin, OUTPUT);
   pinMode(specialLights2Pin, OUTPUT);

   pinMode(switchPin, INPUT);

   Serial.begin(9600);
}
```

В непрекъснатия цикъл се извършват всички манипулации по робота, като командите се отчитат от серийната комуникация.

```
void loop()
{
```

Локалното състояние на итерацията(state - подадена команда; startedLocal - индикатор дали бутона е във включено състояние)

```
 int state = 0;
 int startedLocal = digitalRead(switchPin);
```

Ако бутонът е включен, но колата не е стартирана, се стартира и се вкюлчват предните светлини

```
 if(startedLocal == 1 && !started)
 {
  digitalWrite(headLightsPin, HIGH);
  started = true;
 }
```

Ако колата е стартирана, се извършват нужните манипулации по състоянието ѝ.

```
 if(started)
 {
```

Менажира се състоянието на пътепоказателите и специалните светлини.

```
  Winker();
  SpecialLights();
```

Ако е налична информация в буфера, четем подадената команда и ако тя е команда за движение, колата не трябва да губи ускорение (isStopping приема стойност истина когато колата трябва да намали усорението си)

```
if(Serial.available() > 0)
  {
   state = Serial.read();
   if(state== 'F' || state== 'B' ||state== 'L' ||state== 'R' ||
      state== 'A' ||state== 'D' ||state== 'Z' ||state== 'C')
   {
    isStopping = false;
   }
   Serial.println(state);
  }
```

Ако колата е напълно спряла (състояние lastState : 'S') нейното ускорение е 0

```
   if(lastState == 'S')
   {
    percentAcceleration = 0;
   }
```

Ако колата се движи, но не е стигнала максималното си ускорение, го увеличаваме

```
    if(percentAcceleration != 100 && state == 0 && lastState != 'S' && lastState != -1 &&
!isStopping)
    {
      delay(15);
      percentAcceleration+=5;
      state = lastState;
    }
```

Ако колата спира и все още е налично ускорение, то се намаля

```
    else if(isStopping && percentAcceleration != 0 && state == 0)
    {
      delay(15);
      percentAcceleration-=5;
      state = lastState;
    }
```

Ако колата е напълно спряла, вече не е в процес на спиране (isStopping : false), се променя състоянието ѝ на спряна.

```
    else if(isStopping && percentAcceleration == 0)
    {
      isStopping = false;
      lastState = 'S';
    }
```

При подадена команда за регулиране на скоростта, скоростта се изчислява и командата се променя на последно зададената команда за движение, за да се пренапишат стойностите на работа на моторите чрез PWM.

```
    if(state == 49 || state == 50 || state ==51 || state == 52)
    {
      moveSpeed = 255 * (state-48) / 4;
      Serial.println("SPEED");
      Serial.println(moveSpeed);
      state = lastState;
    }
```

За съответната итерация смятаме скоростта, с която трябва да се движи колата, според ускорението ѝ.

```
    int currentSpeed = moveSpeed * percentAcceleration / 100;
```

Според подадената команда се извършват петте основни действия като се подават сигнали към драйвера, управляващ моторите. Съответно: F-напред, B-назад, S-стоп, L-наляво, R-надясно, както и команда за постепенно намаляване на скоростта до пълно спиране - J

```
if (state == 'F')
    {
       analogWrite(motor1Pin1, currentSpeed );
       digitalWrite(motor1Pin2, LOW);
       digitalWrite(motor2Pin1, LOW);
       analogWrite(motor2Pin2, currentSpeed);
       Serial.println("FORWARD");
       stops = false;
       lastState = state;
    }

    else if (state == 'B')
    {
       digitalWrite(motor1Pin1, LOW);
       analogWrite(motor1Pin2, currentSpeed);
       analogWrite(motor2Pin1, currentSpeed);
       digitalWrite(motor2Pin2, LOW);
       Serial.println("BACKWARD");
       stops = true;
       lastState = state;
    }

    else if (state == 'S')
    {
       digitalWrite(motor1Pin1, LOW);
       digitalWrite(motor1Pin2, LOW);
       digitalWrite(motor2Pin1, LOW);
       digitalWrite(motor2Pin2, LOW);
       Serial.println("STOP");
       stops = true;
       lastState = state;
    }

    else if (state == 'J')
    {
      isStopping = true;
      stops = true;
    }
```

```
  else if (state == 'R')
  {
    analogWrite(motor1Pin1, currentSpeed);
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor2Pin1, LOW);
    digitalWrite(motor2Pin2, LOW);
    Serial.println("RIGHT");
    stops = false;
    lastState = state;
  }

  else if (state == 'L')
  {
    digitalWrite(motor1Pin1, LOW);
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor2Pin1, LOW);
    analogWrite(motor2Pin2, currentSpeed);
    Serial.println("LEFT");
    stops = false;
    lastState = state;
  }
```

Отчитат се и междинните състояние за постепенно завиване чрез забавяне на скоростта на единия мотор чрез PWM - 100/255 дължина на импулса.
Съответно: A-напред и наляво, D-напред и надясно, Z-назад и наляво,C-назад и надясно.

```
  else if (state == 'A')
  {
    analogWrite(motor1Pin1, 100 * percentAcceleration / 100);
    digitalWrite(motor1Pin2, LOW);
    digitalWrite(motor2Pin1, LOW);
    analogWrite(motor2Pin2, 255 * percentAcceleration / 100);
    Serial.println("FRONT LEFT");
    stops = false;
    lastState = state;
  }
```

```
else if (state == 'D')
    {
       analogWrite(motor1Pin1, 255 * percentAcceleration / 100);
       digitalWrite(motor1Pin2, LOW);
       digitalWrite(motor2Pin1, LOW);
       analogWrite(motor2Pin2, 100 * percentAcceleration / 100);
       Serial.println("FRONT RIGHT");
       stops = false;
       lastState = state;
    }

    else if (state == 'Z')
    {
       digitalWrite(motor1Pin1, LOW);
       analogWrite(motor1Pin2, 100 * percentAcceleration / 100);
       analogWrite(motor2Pin1, 255 * percentAcceleration / 100);
       digitalWrite(motor2Pin2, LOW);
       Serial.println("BACK LEFT");
       stops = true;
       lastState = state;
    }

    else if (state == 'C')
    {
       digitalWrite(motor1Pin1, LOW);
       analogWrite(motor1Pin2, 255 * percentAcceleration / 100);
       analogWrite(motor2Pin1, 100 * percentAcceleration / 100);
       digitalWrite(motor2Pin2, LOW);
       Serial.println("BACK RIGHT");
       stops = true;
       lastState = state;
    }
```

При промяна на състоянието на някой мигач се спира другия, тъй като е сигурно, че той няма как да работи едновременно с другия. Съответно: Q-ляв мигач, E-десен мигач, W - аварийни светлини, X- специални светлини.

```
    else if(state == 'Q')
    {
       rightWinker = false;
       leftWinker=!leftWinker;
       Serial.println("LEFT WINKER");
    }
```

```
    else if(state == 'E')
    {
       leftWinker = false;
       rightWinker=!rightWinker;
       Serial.println("RIGHT WINKER");
    }
```

При промяна на състоянието на аварийните светлини двата мигача се изгасят, тъй като работата на аварийните светлини е с по-висок приоритет от мигачите

```
    else if(state == 'W')
    {
       emergencyLight = !emergencyLight;
       leftWinker = rightWinker = false;
       Serial.println("EMERGENCY LIGHTS");
    }
```

Промяна на състоянието на светлините, указващи специалния режим на движение

```
     else if(state == 'X')
    {
       specialLights=!specialLights;
       Serial.println("SPECIAL LIGHTS");
    }
  }
```

При спиране и движение назад стоповете светят, което се отразява тук

```
  if(stops)
  {
   digitalWrite(backLightsPin, HIGH);
  }
  else
  {
   digitalWrite(backLightsPin, LOW);
  }
 }
```

Ако колата е стартирала, но бутонът се изключи, се изключва всичко работещо

```
 if(startedLocal == 0 && started)
 {
   TurnOff();
 }
}
```

Изключване на цялото захранване към елементите на колата и промяна на състоянието на спряно.

```
void TurnOff()
{
   started = false;
   digitalWrite(motor1Pin1, LOW);
   digitalWrite(motor1Pin2, LOW);
   digitalWrite(motor2Pin1, LOW);
   digitalWrite(motor2Pin2, LOW);
   digitalWrite(leftWinkerPin, LOW);
   digitalWrite(headLightsPin, LOW);
   digitalWrite(backLightsPin, LOW);
   digitalWrite(rightWinkerPin, LOW);
   digitalWrite(specialLights1Pin, LOW);
   digitalWrite(specialLights2Pin, LOW);
   emergencyLight = leftWinker = rightWinker = false;
   specialLights = false;
   lastState = 'S';
   Serial.println("STOP");
}
```

Менажиране на пътепоказателите

```
void Winker()
{
```

Ако състоянието на някоя от светлините е активно се извършва действие според таймера, иначе се изключват светлините и таймерът се занулява

```
 if(leftWinker || rightWinker || emergencyLight)
   {
```

При първоначалната стойност на таймера - ако са включени аварийните светлини се пускат двата пътепоказателя, а ако е пуснат само един мигач - съответно той светва, а другият угасва.

```
      if(winkerDelayCounter == 0)
      {
       if(emergencyLight)
       {
         digitalWrite(leftWinkerPin, HIGH);
         digitalWrite(rightWinkerPin, HIGH);
       }
       else
       {
```

```
          if(leftWinker)
        {
         digitalWrite(leftWinkerPin, HIGH);
         digitalWrite(rightWinkerPin, LOW);
        }
        else
        {
         digitalWrite(leftWinkerPin, LOW);
         digitalWrite(rightWinkerPin, HIGH);
        }
       }
      }
```

Увеличава се стойността на брояча и се изчакват 10 милисекунди.

```
winkerDelayCounter++;
delay(10);
```

При достигане на следваща стойност за промяна на състоянието, независимо в кой режим на работа са светлините, те винаги угасват, защото имат еднакъв период на работа

```
if(winkerDelayCounter == 30)
{
  digitalWrite(leftWinkerPin, LOW);
  digitalWrite(rightWinkerPin, LOW);
}
```

При достигане на максималната стойност на брояча, той се занулява и запомва нов цикъл на изчакване на промяната на състоянието на светлините.

```
else if(winkerDelayCounter == 60)
{
  winkerDelayCounter=0;
}
}
else
{
  winkerDelayCounter = 0;
  digitalWrite(leftWinkerPin,LOW);
  digitalWrite(rightWinkerPin,LOW);
}
}
```

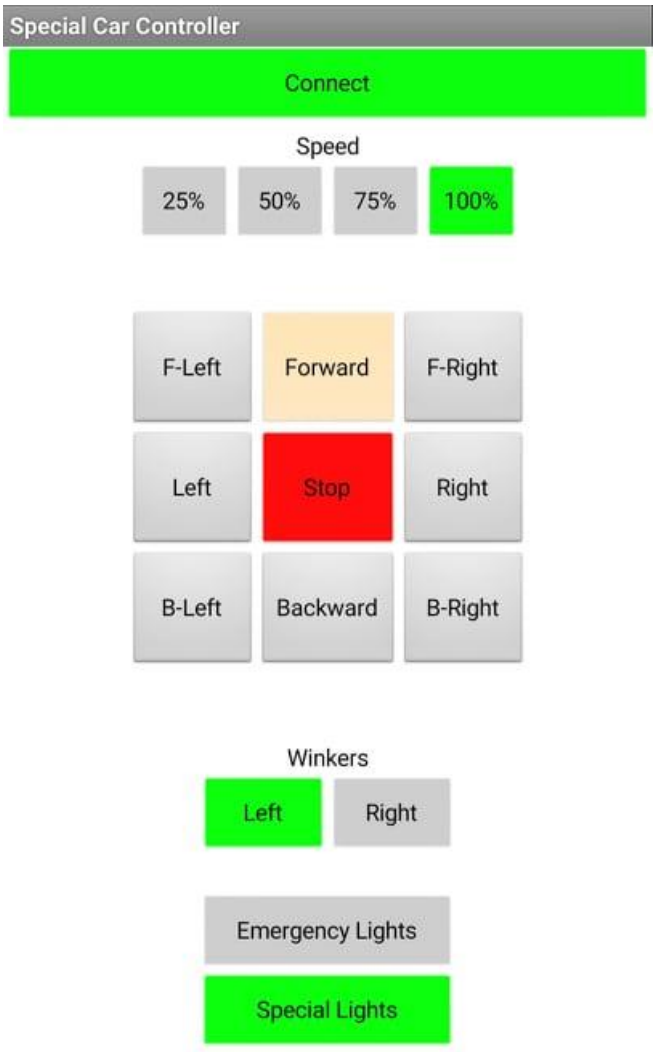
Аналогично работят и специалните светлини - ако са пуснати в зависимост от брояча светва едната, а другата угасва, а когато са изключени и двете не светят.

```
void SpecialLights()
{
 if(specialLights)
   {
      if(specialLightsCounter == 0)
      {
         digitalWrite(specialLights1Pin, HIGH);
         digitalWrite(specialLights2Pin, LOW);
      }

       specialLightsCounter ++;
       delay(10);

      if(specialLightsCounter == 20)
      {
         digitalWrite(specialLights1Pin, LOW);
         digitalWrite(specialLights2Pin, HIGH);
      }
      else if(specialLightsCounter == 40)
      {
         specialLightsCounter = 0;
      }
   }
   else
   {
      specialLightsCounter  = 0;
      digitalWrite(specialLights1Pin, LOW);
      digitalWrite(specialLights2Pin, LOW);
   }
}
```

Особеността, поради която използваме такъв своеобразен брояч (delay counter) в постоянния цикъл (void loop()), е че при нужда от изчакване на по-голям интервал от време, например 3 секунди, програмата забива докато нужното време изтече, и така забавянето в реакцията на подадените команди ще е 3000ms, докато в нашия случай е само 35ms, т.е. почти 85 пъти по-бързо ще е времето за реакция на командите, и то единствено при условие че едновременно колата спира и аварийните и специалните светлини са пуснати.

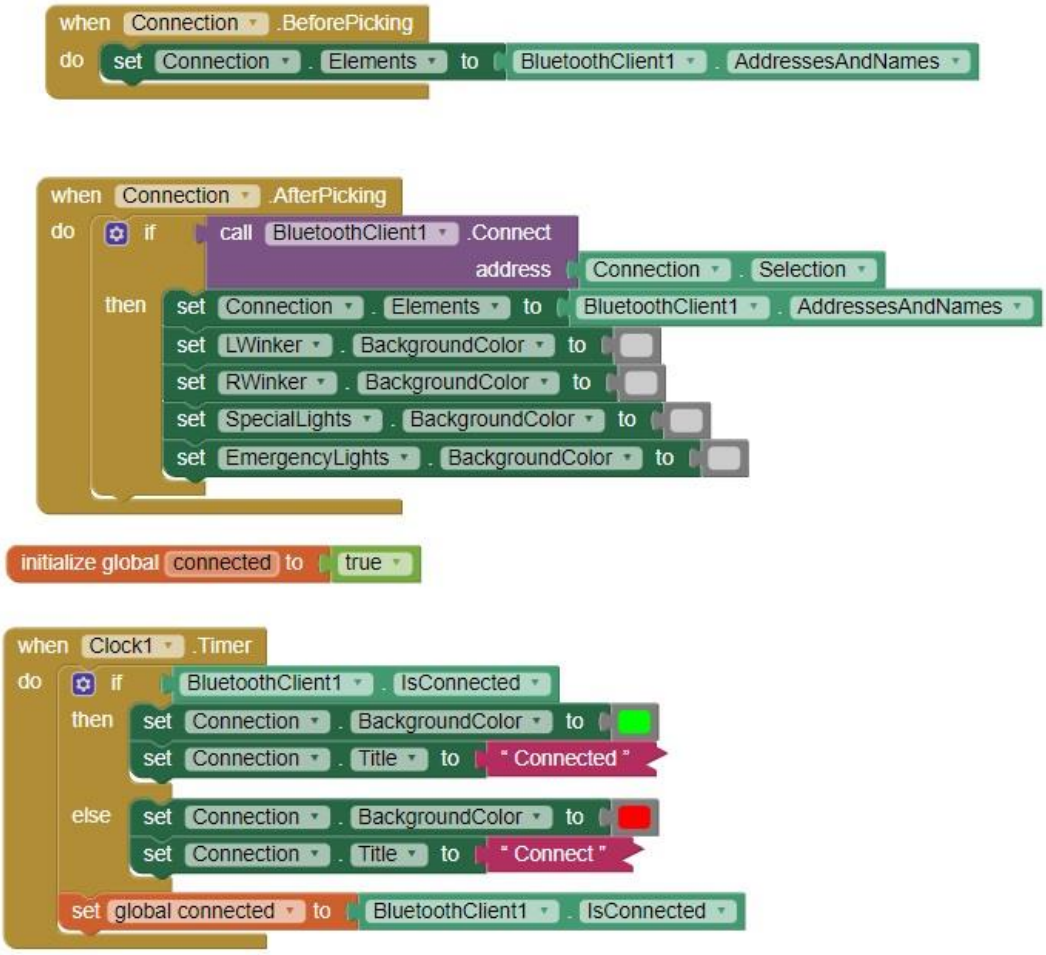
Мобилното приложение

Мобилното приложение за Android е изработено с помощта на MIT App Inventor Има бутон за свързване с блутут модула, който също показва състоянието на свързаност. Има 4 бутона за контрол на скоростта, 8 бутона за промяна на посоката, 1 за стоп, 2 за пътепоказателите, 1 за аварийните светлини и 1 за специалните светлини. При натискане на бутон за движение се изпраща неговото състояние, а при отпускането му се изпраща команда за плавно спиране. Бутонът СТОП изпраща команда за незабавно спиране (аварийна спирачка). Бутоните за светлините и тези за скоростта изпращат еднократно сигнал за промяна на състоянието и го отчитат в приложението чрез промяна на цвета на бутона.

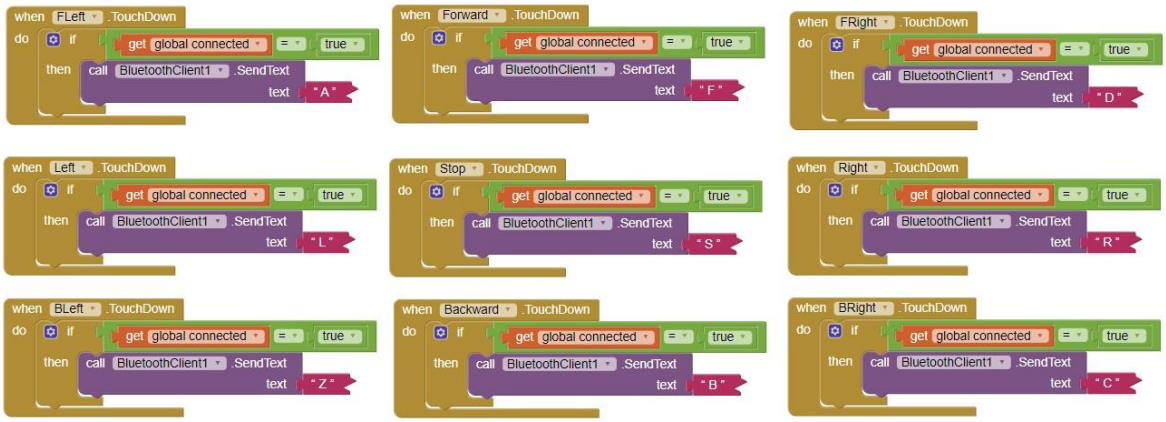


Кодовите схеми на мобилното приложение:

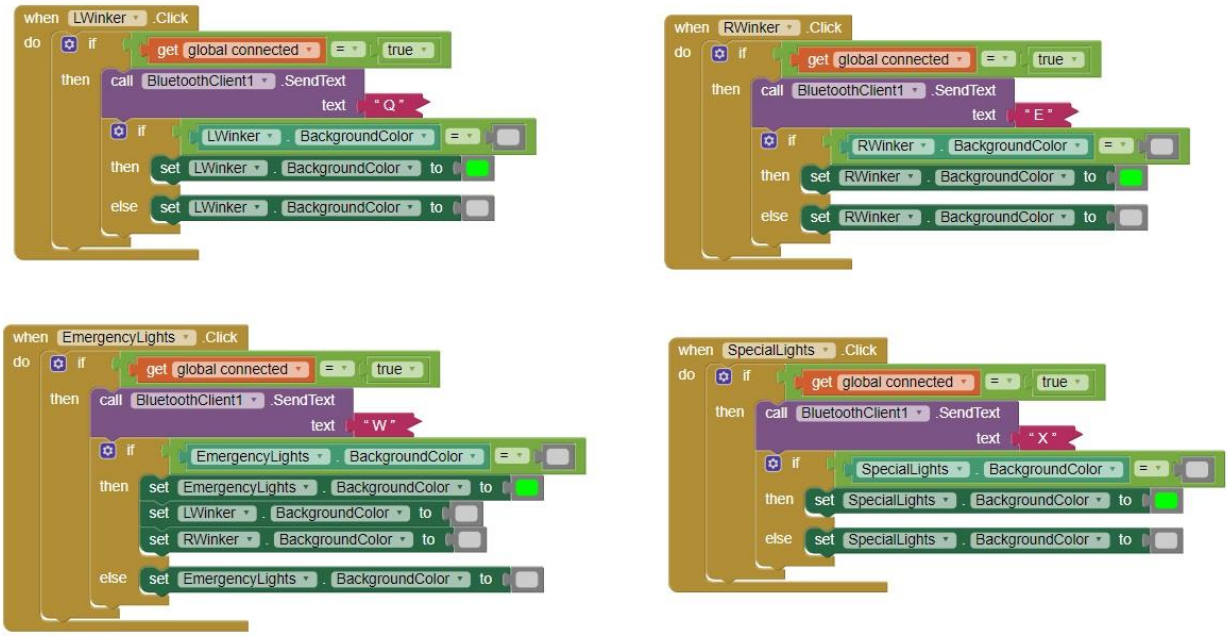
Свързване на приложението с блутут модула на количката и отчитане на състоянието на свързаност чрез цветови кодове.



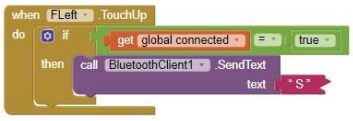
Командите за движение или внезапно спиране при натискане на съответния бутон.

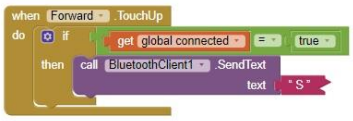


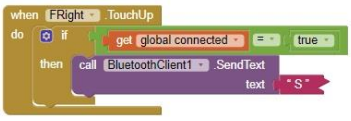
Команди за пътепоказателите и специалните светлини, като при натискане и изпращане на команда се променя цвета на бутона в интерфейса.

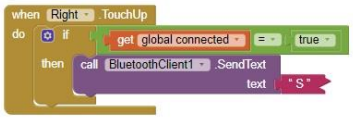


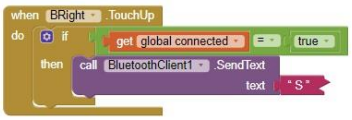
Команда за плавно спиране при отпускане на бутоните за движение.



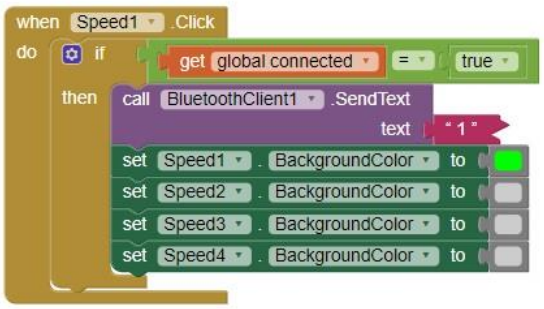


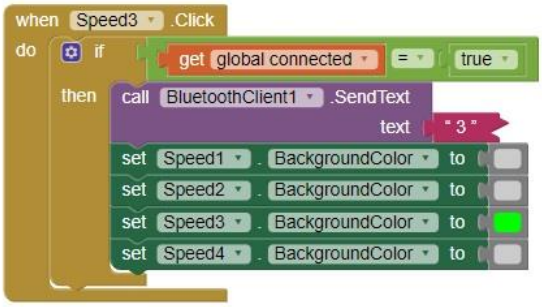


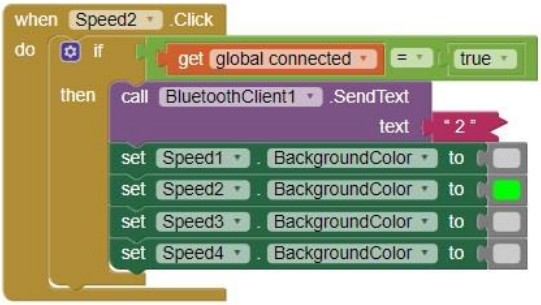


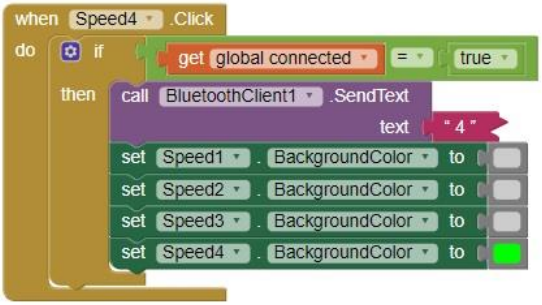


Команда за регулиране на скоростта на движение, като избраната скорост се обозначава с промяна на цвета на бутона.









Заключение

Изработеният робот представлява съчетание между софтуер и хардуер, което определя фунционалността на количката.

+ С предоставените материали максимално реалистично изготвихме работещ прототип на кола, като движението се осъществява плавно и точно във всички желани посоки и разполага с характерните за всеки автомобил светлини и пътепоказатели, както и допълнителни специални светлини.

+ Прототипът е компактен, изработен с подръчни материали, като е вложено много старание и желание за успех.

- Батериите се изтощават много бързо и при тестването бяха необходими общо 10 9-волтови батерии

- Единият мотор беше дефектен и имаше нужда от подмяна, но въпреки това съумяхме да изработим функциониращ прототип.

Въпреки негативните емоции, добрите моменти надделяха и се забавлявахме изключително много с изработването на този проект. :)