СИСТЕМА РАСПОЗНАВАНИЯ ДОРОЖНЫХ ЗНАКОВ «ИТС»

Авторы:
Пешков Иван
Молотильников Семён
Руководитель: Кошелева Л.А.

2022 год

ОГЛАВЛЕНИЕ

Введение

Проблема безопасности на дорогах актуальна для всех государств. Правила дорожного движения соблюдаются, к сожалению, не всеми участниками. Водители устают, их внимание притупляется. Всё это способствует появлению аварий. В настоящее время эта проблема решается внедрением беспилотных автомобилей с искусственным интеллектом и умных дорог. Умные дороги способны увеличить пропускную способность дорожного трафика, повысить безопасность дорожного движения.

В настоящее время дорожная инфраструктура стран с северным климатом не подготовлена к повсеместному внедрению полностью автономных беспилотных автомобилей, поэтому я рассматриваю систему, которая поможет водителю осуществлять лучший контроль в наших климатических условиях. Управление автомобилем по-прежнему производит человек, однако автоматические системы управления автомобилем, в том числе системы распознавания дорожных знаков позволят ему точно соблюдать правила дорожного движения и обеспечат безопасность на дорогах.

Сегодня «умная дорога» – это система, которая включает в себя датчики движения транспортных средств и пешеходов, видеокамеры, модули управления светофорами и многое другое. Все элементы работают на базе единой платформы. Важной частью системы умной дороги являются умные дорожные знаки. Умная дорожная разметка также является значительным элементом системы дорожных знаков. Распознавание дорожных знаков – важная и сложная задача, которая требует комплексного решения.

Объект исследования: система дорожных знаков.

Предмет исследования: система дорожных знаков, нанесённых в виде дорожной разметки.

Цель: разработать систему распознавания дорожных знаков, нанесённых в виде дорожной разметки, адаптированную к разным климатическим условиям.

Задачи:

- провести анализ существующих решений;

- провести поиск решений проблемы распознавания дорожных знаков;
- разработать модель устройства распознавания дорожных знаков;
- провести испытания и корректировку модели распознавания дорожных знаков.

Актуальность: снижение аварийных ситуаций на дорогах – стратегически важная задача, которая позволит сохранить жизнь людям, а также снизит экономические потери. А также создание комфортной среды пользования дорогой, где все участники исполняют предписания дорожных знаков.

Методы исследования: анализ и абстрагирование, метод мозгового штурма, метод проб и ошибок, моделирование.

Исследование является прикладным. Результаты, полученные в работе, могут быть использованы при проектировании «умной» дорожной сети, а также дорог для беспилотных автомобилей, внутренних магистралей (внутри предприятия).

Для написания работы использовались различные книги и интернет-источники: правила дорожного движения, книги по разработке алгоритмов, консультации специалистов.

Идея «умных дорог» достаточно новая, поэтому русскоязычной литературы на эту тему недостаточно. Мы изучали статьи, переведенные с английского языка, и интернет-источники, содержащие новостные и аналитические ресурсы по теме разработки беспилотных автомобилей [7]. [8], [9], а также сообщества разработчиков Arduino.

Исследование проблемы

Анализ существующих решений

Концепция «умной дороги» предполагает взаимодействие автомобиля и всех элементов дорожной системы. Для этого сам автомобиль должен уметь анализировать дорожную ситуацию.

Одним из вариантов является загрузка подробной дорожной карты с указанием расположения перекрёстков, дорожных знаков и т.д. Недостатком этого решения может быть несоответствие загруженной карты реальной обстановке, а также необходимость позиционирования автомобиля на дороге в каждый конкретный момент. Позиционирование автомобиля осуществляется посредством камеры, установленной на автомобиле. Она распознаёт окружающие объекты, дорожные знаки и позиционирует автомобиль на карте. Но такое решение неэффективно, если объекты сместились или частично утратили видимость.

Помимо карт используются лидары и камеры на автомобиле, которые фиксируют дорожную разметку и знаки. Однако такие решения предполагают идеальные дорожные условия: хорошо читаемую разметку на всей протяженности дороги, качественное дорожное покрытие и хорошие климатические и погодные условия.

Например, автомобиль компании Tesla в 2015 году, попав в зону стертой дорожной разметки, прочитал линию соседней полосы как свою собственную, сместился к ней и в результате врезался в бетонное ограждение. Водитель погиб.

Зимой погодные условия создают для «умной дороги» множество помех: дорожные знаки и разметка могут быть покрыты снегом, и датчики не могут их прочитать. Грязь и снег покрывают и сами датчики. Для решения похожей проблемы инженерам фирмы Waymo пришлось разработать систему защиты лидара от продуктов жизнедеятельности птиц.

Таким образом, основной проблемой современных разработки беспилотных транспортных средств является создание автомобилей, не зависящих от погодных условий. Одно из решений - использовать дополнительные лидары. Но

это очень дорого, один лидар может стоит несколько десятков тысяч долларов. Поэтому, например, компания Tesla принципиально не использует лидары в своих автомобилях.

Выводы:

Таким образом, были обнаружены следующие проблемы существующих решений, чтения знаков и разметки с помощью камер:

1) Знаки могут быть намеренно повреждены, даже незначительное повреждение делает его нечитаемым.

Рис. 1. Повреждённый дорожный знак

2) Чтение знаков в зимних климатических условиях затруднено, знаки могут быть занесены снегом, грязью. Сильный снегопад затрудняет их чтение.

3) Чтение знаков и разметки требует от беспилотного автомобиля дорогостоящей аппаратуры.

Нужно новое решение.

Разработка решения

Было решено отказаться от идеи цифрового «прочтения» машиной обычных дорожных знаков, установленных на стойках и опорах, поскольку такие знаки информативны для человека, но не машины. Машина делает лишнюю работу: прочитывает знак или разметку, переводит в информативных для себя цифровой формат, и потом анализирует. Мы решили минимизировать, упро-

стить этот процесс и создать для машины умную среду, где знаки сразу «говорят» с машиной на ее языке.

Первое решение - оснащение знаков дорожного движения дополнительным Bluetooth-модулем, передающим беспилотным автомобилям цифровое значение знака.

Мы разработали специальную программу, которая включает в алгоритм движение автомобиля полученный им Bluetooth сигнал от знака «Пешеходный переход».

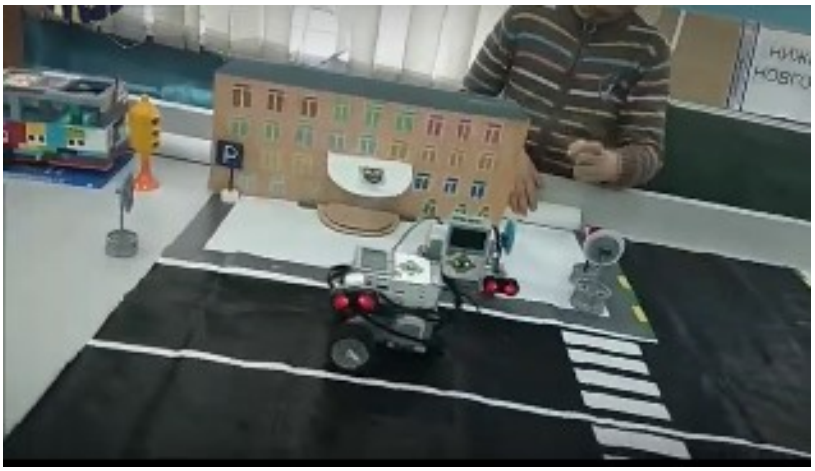


Рис. 2. Первое решение. Знаки, оснащенные Bluetooth-сигналом

Плюсы

1) Высокая информативность знака для машины.

2) Информативность знака не зависит от погодных и иных условий

Минусы

1) Экономическая неэффективность, дорогостоящим оборудованием необходимо обеспечить множество знаков.

2) Знак посылает сигнал не избирательно, всем участникам в зоне его действия, даже тем, кому он не нужен.

Второе решение – штрих-кодовая разметка на месте действия дорожных знаков. Цвет полосы может задавать действие для определенной группы знаков. Первая полоса начинает действие знака. Время и расстояние, затраченные машиной до второй полосы, задает указанное действие знака. Третья полоса завершает действие знака.

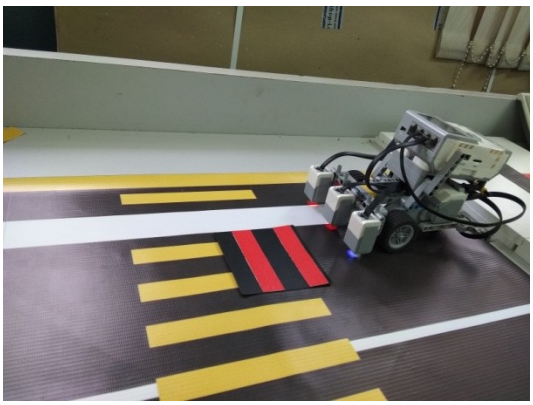


Рис. 3. Второе решение. Штрих-кодовая разметка

Плюсы

1) Высокая информативность знака для машины.

2) Знак посылает сигнал избирательно, только в зоне его действия.

3) Дешевизна, не требует специальной инфраструктуры. Только датчик цвета в машине и краска на дороге.

Минусы

1) Информативность знака зависит от погодных условий

2) Эффективен только в южных климатических зонах

Третье решение – использовать RFID технологию для создания знаков, не зависящих от погодных условий, локального действия.

Разметка и знаки представляют собой RFID-метку, нанесенную на дорожное полотно. Проезжающий автомобиль считывает информацию, которая сообщает ему его местоположение, информацию о знаке, действия которого нужно выполнить, сигнализирует о крае обочины или встречной полосе.

Мы предлагаем использование RFID-меток, в которые загружены геоданные, позволяющие автомобилю точно позиционировать себя на местности. Это нужно для машин, выполняющих движение, и действия знаков по загруженным в них картам.

Плюсы

1) Высокая информативность знака для машины.

3) Информативность знака не зависит от погодных и иных условий

4) Экономическая эффективность, пассивные метки стоят недорого.

5) Знак посылает сигнал локально, только в зоне его действия.

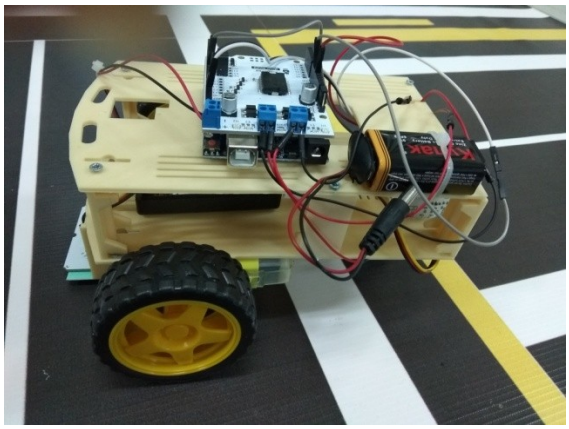


Рис. 4. Третье решение. Считывание автомобилем RFID-меток

Методы исследования

В ходе исследования использовались методы анализа источников информации и абстрагирование с целью найти новое решение. Изучив информацию по задаче распознавания дорожных знаков и известным решениям, обнаружены следующие проблемы:

- не все участники дорожного движения соблюдают правила, регулируемые дорожными знаками;
- дорожные знаки не всегда отражают реальную ситуацию;
- усталость водителей влияет на безопасность дорожного движения;
- беспилотные автомобили позволяют исключить человеческий фактор, однако они остаются плохо управляемыми в нестандартных ситуациях.

В процессе поиска решения поставленных проблем мы использовали метод «мозговой штурм» и «метод контрольных вопросов».

При отладке программы используется «метод проб и ошибок». Это единственно возможный в данной ситуации способ настроить программу.

Также было выполнено моделирование предложенной системы с использованием Lego Mindstorms Ev3 и Arduino.

Описание системы распознавания дорожных знаков

Система распознавания дорожных знаков «ИТС» («Интеллектуальная транспортная система») включает:

- □ дорожное покрытие со специальной разметкой;
- □ дорожные знаки в виде RFID-меток;
- □ «умный» автомобиль.

Дорожные знаки нанесены прямо на дорожное покрытие в виде RFID-меток. Знаки расположены на некотором расстоянии до начала зоны действия, чтобы автомобиль успел среагировать.

Система дорожных знаков в виде RFID-меток

Radio Frequency IDentification, радиочастотная идентификация - способ автоматической идентификации объектов, в котором посредством радиосигналов считываются или записываются данные, хранящиеся в так называемых транспондерах, или RFID-метках *(Википедия)*.

RFID-метки сегодня широко используются в торговле, промышленности и даже в транспортных картах.

Для считывания RFID-меток мы применяем модуль Motor Shield. Программируем его с использованием функций библиотеки CRC.

Код, загруженный в метку, представляет собой набор команд, записанных двоичным кодом. Сканер считывает команды и передаёт их непосредственно на устройства машины, которые выполняют загруженную в код программу.

Алгоритм подтверждения целостности знака: из незащищенного кода получается crc сумма, которая шифруется алгоритмом AES128. Полученное значение сравнивается с эталонным, записанным в коде на последней странице памяти. 4. Если значения не совпали, выполняется защищенный код – машина переходит на ручное управление.

Мы используем RFID-метки частотой 13.56 МГц, на которые записали защищенный код.

Мы разработали знаки – «Стоп», «Движение без остановки запрещено» и «Поворот».

Автомобиль

Корпус автомобиля мы изготовили на 3d-принтере из качественного abs-пластика. За основу мы взяли готовую модель, которую модифицировали в среде 3d-моделирования TinkerCad.

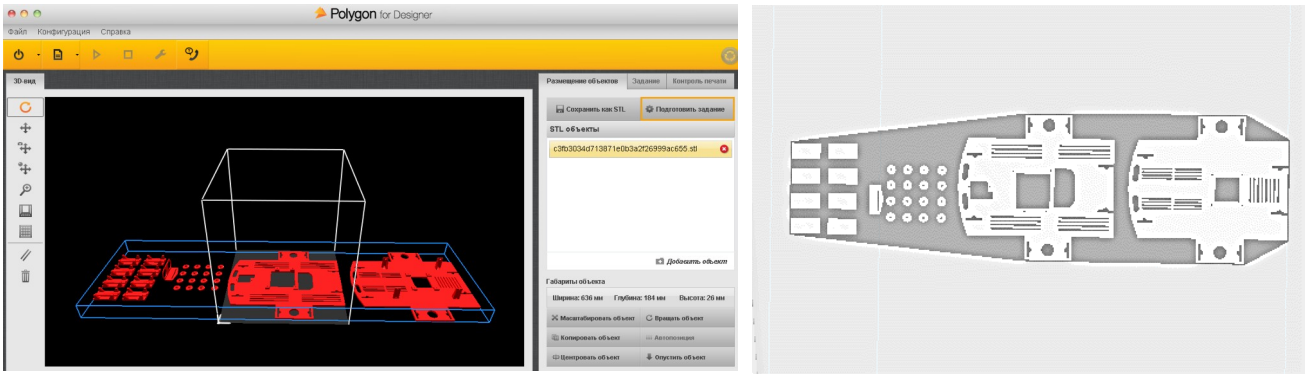


Рис. 5. 3d-модель корпуса автомобиля

Управление происходит при помощи платы Arduino Uno. Также на машину мы установили плату расширения для управления моторами (Motor Shield), электромоторы с зубчатой передачей, сканер для чтения RFID-меток частотой 13.56 МГц, два блока электропитания для платы Arduino Uno (9 В) и электромоторов (4 батарейки по 1,5 В).

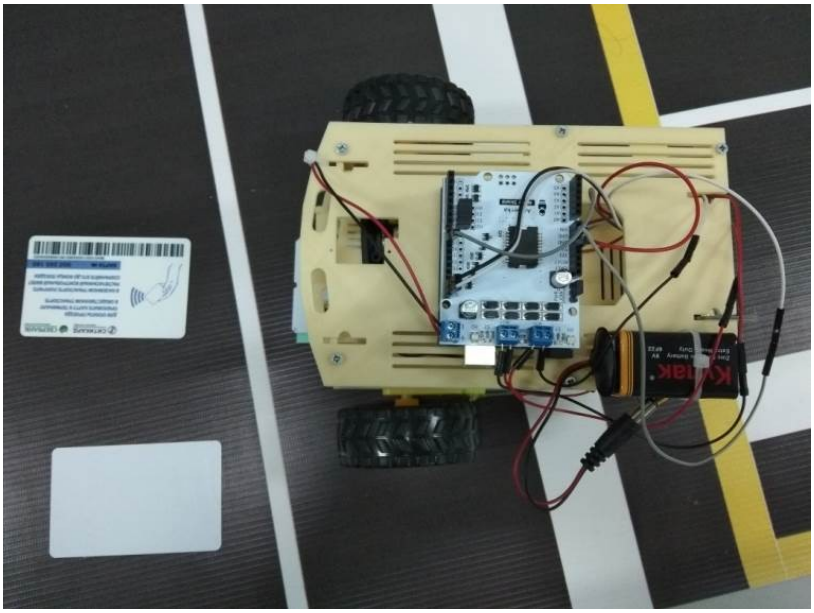


Рис. 6. Модель автомобиля

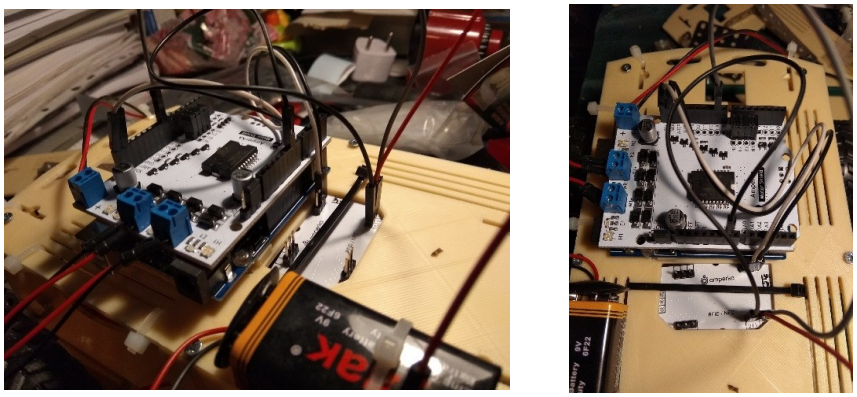
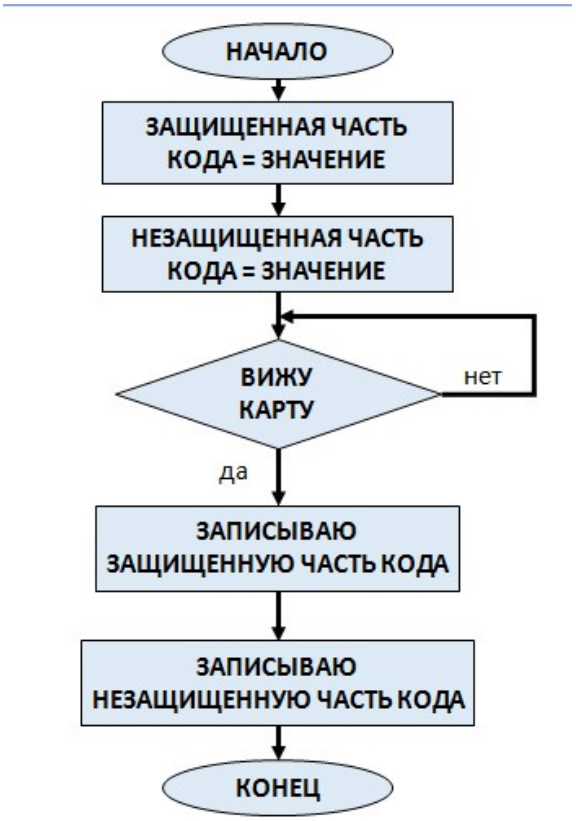


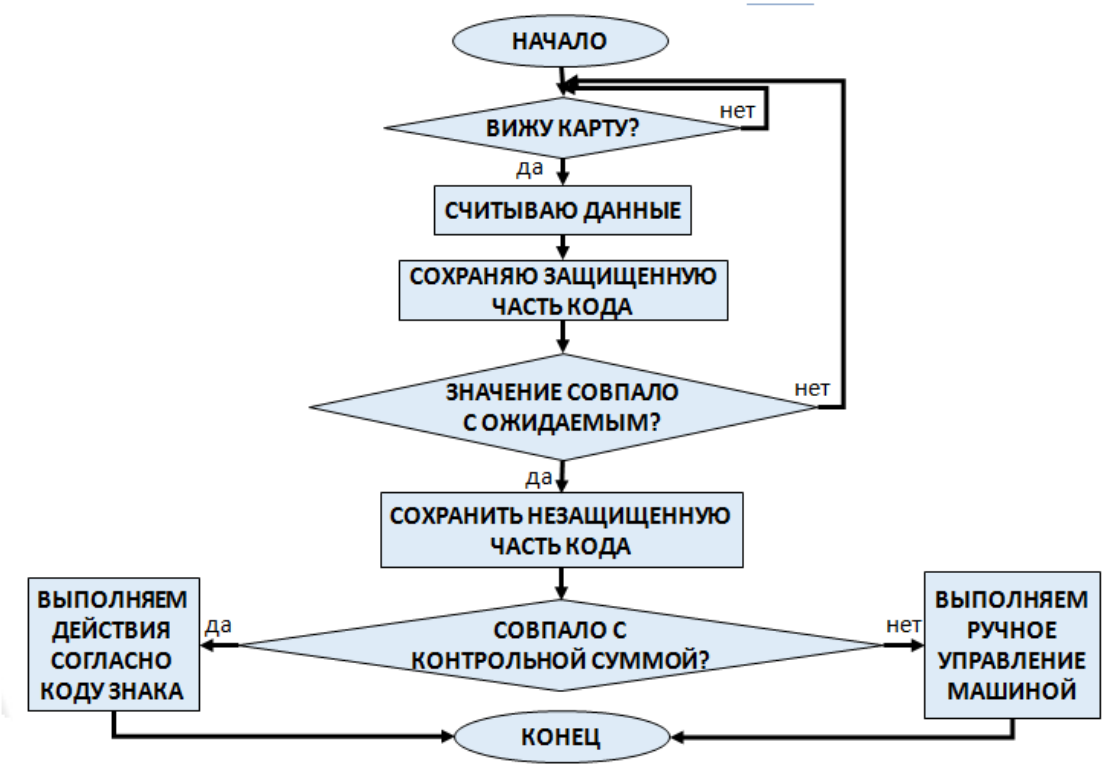
Рис. 7. Чип с контроллером для чтения RFID-меток.

Алгоритмы

Алгоритм записи CRC-кода на RFID-метку



Алгоритм считывания машиной crc-кода с rfid-метки



Программа

Скетч для записи дорожного знака на RFID-метку:

```
#include <Crypto.h>
#include <AES.h>
#include <Arduino_CRC32.h>
#include <Adafruit_PN532.h>
#define IRQ 9
Adafruit_PN532 nfc(IRQ,100);
uint8_t protected_programm[13][4];
uint8_t unprotected_programm[16][4];
Arduino_CRC32 crc;
AESTiny128 aes128;
const uint8_t enc_key[] PROGMEM = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07,0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F};
const uint8_t unprot_magic[] PROGMEM = {0xFF,0xAA,0xFF,0xAA};
const uint8_t unprot PROGMEM = 0;

void setup() {
  Serial.begin(9600);
  Serial.println(F("Initialization start"));
  nfc.begin();
  uint32_t versiondata = nfc.getFirmwareVersion();
 if (!versiondata) {
   Serial.println(F("Didn't find PN53x board"));
   while (1); // halt
  }
  nfc.SAMConfig();
  protected_programm[0][0] = 0xAA;
  protected_programm[0][1] = 0x03;
  protected_programm[0][2] = 0xFF;
```

```
  protected_programm[0][3] = 0x00;
}

void loop() {
   if(!Serial.available()){
    return;
   }else{
    while(Serial.available()){
     Serial.read();
    }
    Serial.println("Starting writing");
   }

   uint8_t success;
   uint8_t uidLength;
   uint8_t uid[] = { 0, 0, 0, 0, 0, 0, 0 };
   uint8_t zero[] = {0, 0, 0, 0};
    success = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A, uid, &uidLength);
   if(success && uidLength == 7){
    for(uint8_t i=0;i<13;i++){
     Serial.print("Writing ");
     Serial.print(i,DEC);
     Serial.println("/13 protected page");
     if(!nfc.ntag2xx_WritePage(4+i,protected_programm[i])){
      Serial.println("Write error");
     }
    }
    if(unprot){
     nfc.ntag2xx_WritePage(17,unprot_magic);
```

```
          for(uint8_t i=18;i<35;i++){
            Serial.print("Writing ");
            Serial.print(i-18,DEC);
            Serial.println("/16 unprotected page");
            if(!nfc.ntag2xx_WritePage(i,(uint8_t*)unprotected_programm[i-18])){
              Serial.println("Write error");
            }
          }
                        uint32_t checksum = crc.calc(&unprotected_programm[0]
[0],16*4*sizeof(uint8_t));
          uint32_t pad = 0xDEADBEEF;
          uint32_t enc_buffer[] = {checksum,pad,pad,pad};
          uint8_t encrypted[16];
          crypto_feed_watchdog();
          aes128.setKey(enc_key,aes128.keySize());
          aes128.encryptBlock(encrypted,(uint8_t*)&enc_buffer[0]);
          for(uint8_t i=0;i<4;i++){
              nfc.ntag2xx_WritePage(35+i,&encrypted[i*4]);
          }
          Serial.println("Written protection");
        }else{
          nfc.ntag2xx_WritePage(17,zero);
        }
      }
    }
```

Скетч для управления машиной на платформе Arduino:

```
#include <Arduino_CRC32.h>
```

//Protected code - минимальная программа для защищенной памяти – гарантирует, что машина будет стоять на месте

```
//0xAA 0x00 0x00 0xF0
//0x00 0xFF 0x00 0x00
//Example 1 - движение вперед до останавливающего знака
//0xAA 0x01 0xFF 0x00
//Example 2 - поворот
//0xAB 0x78 0xAA 0x03
//0xAA 0x01 0x00 0xFF
//Stop_Sign - останавливающий знак
//0xAA 0x00 0x00 0xFF
#include <Crypto.h>
#include <AES.h>
#include <Adafruit_PN532.h>
#define OPFLAG_NOARG 1
#define EXECFLAG_PROGSET 1
#define EXECFLAG_CMP     2
#define SPEED_1      5
#define DIR_1        4
#define SPEED_2      6
#define DIR_2        7
#define IRQ 9
Adafruit_PN532 nfc(IRQ, 100);
typedef struct{
  uint8_t flags;
  uint8_t op;
  uint8_t (*handler)(uint8_t arg);
}op_t;

op_t* ops[32];
uint32_t op_c;
```

```
uint8_t reg_exec_flags;

uint8_t reg_dat[16];

uint8_t reg_sel;
uint8_t reg_cmp;
uint8_t reg_ang;

uint32_t reg_slp;
uint32_t reg_eip;

uint8_t c_programm[16][4];
op_t* reg_cop;
uint8_t reg_carg;

Arduino_CRC32 crc;
AESTiny128 aes128;

const uint8_t enc_key[] PROGMEM = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05,
0x06, 0x07,0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F};
const uint8_t unprot_magic[] PROGMEM = {0xFF,0xAA,0xFF,0xAA};

void clean_registers(){
  reg_eip = 0;
  memset(reg_dat,0,16*sizeof(uint8_t));
  reg_cop =  0;
  reg_carg = 0;
  reg_sel = 0;
  reg_cmp = 0;
  reg_ang = 0;
```

```
  for(uint8_t i = 0;i < 16; i++){
    memset(&c_programm[i][0],0,4);
  }
}
uint8_t echo_handlr(uint8_t arg){
  Serial.println(arg,HEX);
  return 0;
}
uint8_t null_handlr(uint8_t arg){
  return 0;
}
uint8_t move_handlr(uint8_t arg){
  //arg = {1;0;2;3;4} - direction
  switch(arg){
    case 0: //STOP
     analogWrite(SPEED_1, 0);
     analogWrite(SPEED_2, 0);
     return 0;
    case 1://FORWARD
     digitalWrite(DIR_1, LOW);
     digitalWrite(DIR_2, LOW);
     analogWrite(SPEED_1, 210);
     analogWrite(SPEED_2, 140);
     return 0;
    case 2://BACKWARD
     digitalWrite(DIR_1, HIGH);
     digitalWrite(DIR_2, HIGH);
     analogWrite(SPEED_1, 210);
     analogWrite(SPEED_2, 140);
     return 0;
```

```
    case 3://LEFT
     digitalWrite(DIR_1, HIGH);
     digitalWrite(DIR_2, LOW);
     analogWrite(SPEED_1, 0);
     analogWrite(SPEED_2, 140);
     return 0;
    case 4://RIGHT
     digitalWrite(DIR_1, LOW);
     digitalWrite(DIR_2, HIGH);
     analogWrite(SPEED_1, 140);
     analogWrite(SPEED_2, 0);
     return 0;
    default:
     return 1;
  }
}
uint8_t sav_handlr(uint8_t arg){
  reg_dat[reg_sel] = arg;
  return 0;
}
uint8_t inc_handlr(uint8_t arg){
  reg_dat[reg_sel] += arg;
  return 0;
}
uint8_t dec_handlr(uint8_t arg){
  reg_dat[reg_sel] -= arg;
  return 0;
}
uint8_t sleep_handlr(uint8_t arg){
  reg_slp = arg;
```

```
  return 0;
}
uint8_t end_handlr(uint8_t arg){
  reg_exec_flags &= ~EXECFLAG_PROGSET;
  return 0;
}
uint8_t jmp_handlr(uint8_t arg){
  if(arg >= 64){
    return 1;
  }
  reg_eip = arg;
  return 0;
}
uint8_t jmpif_handlr(uint8_t arg){
  if(reg_exec_flags & EXECFLAG_CMP){
    reg_exec_flags &= ~EXECFLAG_CMP;
    return jmp_handlr(arg);
  }
  return 0;
}
uint8_t cmp_handlr(uint8_t arg){
 switch(reg_cmp){
    case 0:
      if(reg_dat[arg] < reg_cmp){
        reg_exec_flags |= EXECFLAG_CMP;
      }
    break;
    case 1:
      if(reg_dat[arg] <= reg_cmp){
        reg_exec_flags |= EXECFLAG_CMP;
```

```
      }
    break;
    case 2:
      if(reg_dat[arg] > reg_cmp){
        reg_exec_flags |= EXECFLAG_CMP;
      }
    break;
    case 3:
      if(reg_dat[arg] >= reg_cmp){
        reg_exec_flags |= EXECFLAG_CMP;
      }
    break;
    case 4:
      if(reg_dat[arg] == reg_cmp){
        reg_exec_flags |= EXECFLAG_CMP;
      }
    break;
  }
 return 0;
}
uint8_t setcmp_handlr(uint8_t arg){
  reg_cmp = arg;
  return 0;
}
uint8_t sel_handlr(uint8_t arg){
  reg_sel = arg;
}
void register_op(uint8_t op,uint8_t (*handler)(uint8_t arg),uint8_t need_arg){
  //Serial.println(op,HEX);
  if(find_op(op)){
```

```
    Serial.print(F("Failed to register: Operator 0x"));
    Serial.print(op,HEX);
    Serial.println(F(" already registered!"));
    return;
  }
  if(op_c >= 32){
    Serial.println(F("Failed to register: Operator limit exceeded!"));
    return;
  }
  op_t* op_s = (op_t*)malloc(sizeof(op_t));
  op_s->op = op;
  op_s->handler = handler;
  op_s->flags = 0;
  if(!need_arg){
    op_s->flags |= OPFLAG_NOARG;
  }
  ops[op_c] = op_s;
  op_c++;
}
op_t* find_op(uint8_t op){
  for(uint32_t i=0;i<op_c;i++){
    if(ops[i]->op == op){
      return ops[i];
    }
  }
  return 0;
}
uint8_t process_op(op_t* op){
  Serial.print(F("Processing "));
  Serial.print(op->op,HEX);
```

```
  Serial.print("(");
  Serial.print(reg_carg);
  Serial.println(")");
  return op->handler(reg_carg);
 }
 void setup() {
  Serial.begin(9600);
  Serial.println(F("Initialization start"));
  nfc.begin();
  uint32_t versiondata = nfc.getFirmwareVersion();
 if (!versiondata) {
   Serial.println(F("Can't find PN53x board"));
   while (1); // halt
  }
  for (int i = 4; i < 8; i++) {
   pinMode(i, OUTPUT);
  }
  nfc.SAMConfig();
  op_c = 0;
  clean_registers();
  memset(ops,0,32*sizeof(op_t*));
  register_op(0x00,null_handlr ,0); // NOP
  register_op(0x0D,echo_handlr ,1); // Отладочный вывод
  register_op(0xAA,move_handlr ,1); // Движение
   register_op(0xA0,sav_handlr  ,1); // Сохранение значения в один их регистров общего назначения
  register_op(0xA1,inc_handlr  ,1); // Увеличение значения регистра
  register_op(0xA2,dec_handlr  ,1); // Уменьшение значения регистра
  register_op(0xCC,cmp_handlr  ,1); // Сравнение
  register_op(0xCA,setcmp_handlr,1);// Операнд сравнения
```

```
  register_op(0xDD,sleep_handlr,1); // Задержка
  register_op(0xC0,sel_handlr,  1); // Выбор регистра
  register_op(0xFE,jmpif_handlr,1); // Условный переход
  register_op(0xF0,jmp_handlr  ,1); // Безусловный переход
  register_op(0xFF,end_handlr  ,0); // Выход из программы
  Serial.println(F("Initialization end"));
}
void loop() {
  if(!(reg_exec_flags & EXECFLAG_PROGSET)){
    uint8_t success;
    uint8_t uidLength;
    uint8_t uid[] = { 0, 0, 0, 0, 0, 0, 0 };
     success = nfc.readPassiveTargetID(PN532_MIFARE_ISO14443A, uid, &ui-
dLength);
    if(success && uidLength == 7){
      clean_registers();
      uint8_t   protected_programm[13][4];
      uint8_t unprotected_programm[16][4];
      for(uint8_t i=0;i<13;i++){
        nfc.ntag2xx_ReadPage(4+i,&protected_programm[i][0]);
      }
      uint8_t magic[4];
      nfc.ntag2xx_ReadPage(17,magic);
      uint8_t res = 1;
      for(uint8_t i=0;i<4;i++){
        if(magic[i] != unprot_magic[i]){
          res = 0;
          break;
        }
      }
```

```
          if(res){
           Serial.println(F("Found unprotected code"));
           // Serial.println(magic,HEX);
           for(uint8_t i=18;i<35;i++){
             nfc.ntag2xx_ReadPage(i,unprotected_programm[i-18]);
           }
           Serial.println(F("Verifying..."));
                        uint32_t checksum = crc.calc(&unprotected_programm[0]
[0],16*4*sizeof(uint8_t));
           uint32_t pad = 0xDEADBEEF;
           uint32_t enc_buffer[] = {checksum,pad,pad,pad};
           uint8_t encrypted[16];
           crypto_feed_watchdog();
           aes128.setKey(enc_key,aes128.keySize());
           aes128.encryptBlock(encrypted,(uint8_t*)&enc_buffer[0]);
           uint8_t eta[16];
           for(uint8_t i=0;i<4;i++){
               nfc.ntag2xx_ReadPage(35+i,&eta[i*4]);
           }
           if(memcmp(eta,encrypted,16)){
                 Serial.println(F("Failed to verify code integrity, executing protected
code!"));
             for(int i=0;i<13;i++){
               memcpy(c_programm[i],&protected_programm[i][0],4);
             }
           }else{
             for(int i=0;i<16;i++){
               memcpy(c_programm[i],&unprotected_programm[i][0],4);
             }
          }
```

```
    }else{
      Serial.println(F("Unprotected code not found"));
      for(int i=0;i<13;i++){
          memcpy(c_programm[i],&protected_programm[i][0],4);
      }
    }
    reg_exec_flags |= EXECFLAG_PROGSET;
    Serial.println(F("Starting execution"));
  }
 }else{
  if(reg_slp){
    reg_slp--;
    return;
  }
  uint8_t  c_page = reg_eip / 4;
  uint8_t  c_word = reg_eip % 4;
  if(!reg_cop){
    op_t* op = find_op(c_programm[c_page][c_word]);
    if(!op){
      Serial.print(F("Failed to exec: Invalid operator 0x"));
      Serial.print(c_programm[c_page][c_word],HEX);
      Serial.println(F("!"));
      end_handlr(0);
      return;
    }
    if(op->flags & OPFLAG_NOARG){
      if(process_op(op)){
        Serial.println(F("Failed to exec: Operator failure!"));
        end_handlr(0);
        return;
```

```
      }
    }else{
      reg_cop = op;
    }
  }else{
    reg_carg = c_programm[c_page][c_word];
    if(process_op(reg_cop)){
      Serial.println(F("Failed to exec: Operator failure!"));
      end_handlr(0);
    }
    reg_cop = 0;
    reg_carg = 0;
  }
  reg_eip++;
  if(reg_eip >= 64){
    end_handlr(0);
  }
 }
}
}
```

Экономическая часть

1) Bluetooth-модуль 1 класса большой дальности – средняя цена 500 рублей

2) Штрих код-знак – средняя цена 150 рублей п.м. (цена дорожной разметки в г. Нижний Новгород: от 20 до 150 р./п.м.)

Датчик цвета – средняя цена 200 рублей

3) RFID метка. Средняя цена – 80 рублей

RFID сканер. Средняя цена – 800 рублей

Таким образом, смета проекта «ИТС» для 1 км дорог составляет:

Наименование	Стоимость 1 единицы	Количество	Сумма
Дорога			
Штрих код-знак	150 р.	1000	150000 р.
RFID-метка	80 р.	1000	80000 р.
Автомобиль			
RFID-сканер	800 р.	1	800 р.

Итого: 230 800 р.

Выводы

Плюсы использования дорожных знаков в виде rfid-разметки в системе «ИТС»:

1) Высокая информативность знака для машины.

3) Информативность знака не зависит от погодных и иных условий

4) Экономическая эффективность, пассивные RFID-метки стоят недорого.

5) Знак посылает сигнал локально, только в зоне его действия.

Мы считаем, что наше решение может служить основой эффективного организации дорожного движения. Это подход будущего, который нужно осваивать уже сегодня!

Список литературы

1. Правила дорожного движения 2019. Официальный текст с иллюстрациями. - СПб: Питер, 2019

2. Валк Л. Большая книга LEGO MINDSTORMS EV3. - М.: Эксмо, 2017

3. Овсяницкая Л.Ю. Алгоритмы и программы движения по линии робота Lego Mindstorms EV3. / Л.Ю. Овсяницкая, Д.Н, Овсяницкий, А.Д. Овсяницкий - М.: "Перо", 2015

Интернет-ресурсы

4. RFID/NFC-сканер. Режим доступа: http://wiki.amperka.ru/%D0%BF%D1%80%D0%BE%D0%B4%D1%83%D0%BA%D1%82%D1%8B:nfc Дата обращения: 21.01.2020

5. Урок 6. Arduino считываем метки (RFID-модуль RC522). Режим доступа: https://lesson.iarduino.ru/page/urok-6-arduino-schityvaem-metki-rfid-modul-rc522 Дата обращения: 21.01.2020

6. RFID. Режим доступа: https://ru.wikipedia.org/wiki/RFID Дата обращения: 21.01.2020

7. Проверяем системы распознавания знаков в деле. Режим доступа: https://www.zr.ru/content/articles/444117-proverajem_sistemy_raspoznavanija_znakov_v_dele/ Дата обращения: 21.01.2020

8. Как работает система распознавания дорожных знаков в деле. Режим доступа: https://techautoport.ru/sistemy-bezopasnosti/aktivnaya/sistema-raspoznavaniya-dorozhnyh-znakov.html Дата обращения: 21.01.2020

9. Простой алгоритм распознавания дорожной разметки. Режим доступа: https://www.reg.ru/blog/simple-algorithm-for-road-marking-detection/ Дата обращения: 21.01.2020