|  |  |
| --- | --- |
|  | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ   Радиоэлектроника и лазерная техника (РЛ)

КАФЕДРА   Радиоэлектронный системы и устройства (РЛ1)

**ОТЧЕТ ПО технологической ПРАКТИКЕ**

Студент   Чекановский Сергей Александрович

(Фамилия, имя, отчество)

Группа   РЛ1-61

Тема практики  Передача отсчетов цифрового радиосигнала с ПЛИС на ПК и их графическое отображение. Управление формирователем радиосигналов на ПЛИС с помощью микроконтроллера

Название предприятия  АО «Российские космические системы»

Руководитель практики

от МГТУ им. Н.Э. Баумана \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

Руководитель практики

от  **АО «Российские**

**космические системы»**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**      Чекановский С.А.

(Подпись, дата) (И.О.Фамилия)

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2021г.*

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

Отчет

В рамках практики была разработана программа для микроконтроллеров Arduino, а так же программа на языке Python для коммуникации с микроконтроллером через COM порт и графического отображения данных, полученных с контроллера.

Ниже представлены листинги программ на Python и C++, написанные для работы с контроллером Arduino.

1. Коммуникация с контроллером.

import serial #requires pip install pyserial

import matplotlib.pyplot as plt

import numpy  as np

serial\_port = 'COM19' #set your COM port value

baud\_rate = 9600; #set same as Arduino has

write\_to\_file\_path = "output.txt"

with serial.Serial(serial\_port, baud\_rate, timeout = 0) as ardu:

    with open(write\_to\_file\_path, "w") as output\_file:

        num = input("Enter a number: ") # Taking input from user

        ardu.write(bytes(num, 'utf-8')) #Input -> COM port

        while True:

            # time.sleep(0.01)

            value = ''

            while (value.find('\n') == -1):

                value += ardu.readline().decode('utf-8')

            value = value.strip()

            if (value == 'END'):

                break

            print(value) # printing the value

            output\_file.write(value + '\n')

data = np.loadtxt('output.txt')

y = data[:]

plt.plot(y)

plt.ylabel('y')

plt.show()

Программа приглашает пользователя ввести числовую команду, затем отправляет команду в выбранный COM порт и ждет поступления данных в порт от контроллера. Затем записывает данные в файл и на основе данных из файла строит график.

2. Алгоритм работы микроконтроллера.

Работа платы полностью описывается одним классом.

Класс — в объектно-ориентированном программировании, представляет собой шаблон для создания объектов, обеспечивающий начальные значения состояний: инициализация полей-переменных и реализация поведения функций или методов.

Диаграмма UML представлена на рис. 1.

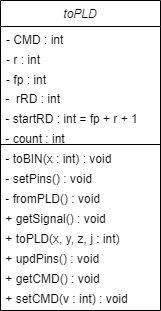


Рис. 1.

Листинг программы:

#include <Arduino.h>

int s;

double pi = 3.1415;

class toPLD

{

private:

  int CMD; //Содержит введенную команду

  int r;   //Разрядность шины

  int fp;

  int rRD; //Hазрядность принимаемых данных

  int startRD = fp + (r + 1);

  int count;        //Количество отсчетов сигнала

  void toBIN(int x)

  {

    for (int i = r - 1; i >= 0; --i)

    {

      int f = ((int)((x >> i) & 1));

      switch (f)

      {

      case 1:

        digitalWrite(i + fp, HIGH);

        break;

      case 0:

        digitalWrite(i + fp, LOW);

        break;

      }

    }

    digitalWrite(r + fp, HIGH);

    delay(1000);

    digitalWrite(r + fp, LOW);

  }

  void setPins()

  {

    for (int i = fp; i <= r + fp; i++)

    {

      pinMode(i, OUTPUT);

    }

    for (int i = r + fp + 1; i <= r + fp + rRD; i++)

    {

      pinMode(i, INPUT);

    }

  }

  void fromPLD()

  {

    String s = "";

    for (int i = fp + (r + 1); i <= fp + r + rRD; i++)

    {

      int c = digitalRead(i);

      s += (char)('0' + c);

    }

    int result = 0;

    for (int i = s.length() - 1, pos = 0; i >= 0; i--, ++pos)

      result += s[i] == '0' ? 0 : 1 << pos;

    Serial.println(result);

  }

public:

  void getSignal() //Получаем числа с шины данных и выводим в порт

  {

    for (int i = 0; i < count; i++)

    {

      digitalWrite(fp + r + rRD + 1, HIGH);

      fromPLD();

      digitalWrite(fp + r + rRD + 1, LOW);

    }

    Serial.println("END");

  }

  toPLD(int x, int y, int z, int j) //Конструктор класса

  {

    r = x;     //Разрядность шины управления

    fp = y;

    rRD = z;   //Разрядность шины данных

    count = j; //Количество отсчетов сигнала с плис

  }

  void updPins()

  {

    setPins();

    if (CMD != -1)

      toBIN(CMD);

  }

  int getCMD() //Просто так

  {

    return CMD;

  }

  void setCMD(int v)

  {

    if ((v >= 0) && (v <= pow(2, r) - 1))

      CMD = v;

    else

    {

      Serial.println("Ардуинка не понимает такого обращения");

      CMD = -1;

    }

  }

};

void setup()

{

  Serial.begin(9600);

}

void loop()

{

  toPLD arduino(2, 2, 6, 16384);

  while (true)

  {

    if (Serial.available())

    {

      String ss = Serial.readStringUntil('\n');

      if (ss != "")

      {

        int s = atoi(ss.c\_str());

        arduino.setCMD(s);

        arduino.updPins(); //Обновляем управляющий сигнал

        delay(500);

        pinMode(4, INPUT);

        pinMode(13, INPUT);

        while (digitalRead(4) != 1)

        {

        }

        arduino.getSignal(); //Получаем отсчеты сигнала с ПЛИС

      }

    }

  }

}

Конструктор класса принимает значения: разрядности управляющей шины, номер первого пина, разрядность шины данных, подключенной к выходу ПЛИС, а так же количество остчетов сигнала, приходящих с ПЛИС.

 toPLD(int x, int y, int z, int j) //Конструктор класса

  {

    r = x;     //Разрядность шины управления

    fp = y;    //Первый пин. Порядок: пины управления -> пин RDY -> шина приема данных -> пин RDY для шины данных

    rRD = z;   //Разрядность шины данных

    count = j; //Количество отсчетов сигнала с плис

  }

Это позволяет быстро перенастроить программу для работы с другими микроконтролерами или же в других условиях, например при изменении разрядности шин.

Метод безопасного приема числового значения команды внутрь класса не позволяет ввести команду, значение которой превысило бы разрядность управляющей шины.

  void setCMD(int v)

  {

    if ((v >= 0) && (v <= pow(2, r) - 1))

      CMD = v;

    else

    {

      Serial.println("Ардуинка не понимает такого обращения"); //Исключение, ничего не выполняется

      CMD = -1;

    }

  }

};

Публичный метод updPins() вызывает приватные методы, которые перепределяют входные и выходные пины платы, а затем передают команду на управляющую шину.

  void updPins()

  {

    setPins();

    if (CMD != -1)

      toBIN(CMD);

  }

  void toBIN(int x)

  {

    for (int i = r - 1; i >= 0; --i)

    {

      int f = ((int)((x >> i) & 1)); //преобразование куска числа в двоичный код, начиная со старшего разряда

      switch (f)

      {

      case 1:

        digitalWrite(i + fp, HIGH);

        break;

      case 0:

        digitalWrite(i + fp, LOW);

        break;

      }

    }

    digitalWrite(r + fp, HIGH); //Импульс READY, означает окончание формирования сигнала на шине

    delay(1000);

    digitalWrite(r + fp, LOW);

  }

  void setPins() //Инициализирует выходные пины в зависимости от разрядности шины

  {

    for (int i = fp; i <= r + fp; i++)

    {

      pinMode(i, OUTPUT);

    }

    for (int i = r + fp + 1; i <= r + fp + rRD; i++)

    {

      pinMode(i, INPUT);

    }

  }

Далее вызывается метод getSignal(). Он последовательно считывает 8-разрядные числа с ПЛИС и выводит их в COM порт. Программа на Python записывает эти значения в файл и строит из них график.

  void getSignal() //Получаем числа с шины данных и выводим в порт

  {

    for (int i = 0; i < count; i++)

    {

      digitalWrite(fp + r + rRD + 1, HIGH);

      fromPLD();

      digitalWrite(fp + r + rRD + 1, LOW);

    }

    Serial.println("END"); //Для завершения цикла записи в файл в Python

  }

  void fromPLD() //Принимает код с шины ПЛИС, преобразует в десятичное число, выводит в порт

  {

    String s = "";

    for (int i = fp + (r + 1); i <= fp + r + rRD; i++)

    {

      int c = digitalRead(i);

      s += (char)('0' + c);

    }

    int result = 0;

    for (int i = s.length() - 1, pos = 0; i >= 0; i--, ++pos)

      result += s[i] == '0' ? 0 : 1 << pos;

    Serial.println(result);

  }