МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет» Факультет автоматики и вычислительной техники Кафедра электронных вычислительных машин

# Дневник

по производственной практике на ПАО «Кировский завод «Маяк», г. Киров

Выполнил студент группы ИВТ-31 /Птахова А.М. /

Руководитель практики от ВятГУ /Караваева О.В./ Руководитель практики от предприятия / Ермолаев А. О./

Киров 2022

|  |  |
| --- | --- |
| Дата | Вид деятельности |
| 06.09.2022 | Получение задания на практику |
| 07.09.2022 | Выполнение задания |
| 20.11.2022 | Подготовить и оформить отчет |

Введение

Производственная практика является неотъемлемой и важной частью учебного процесса. В ходе неё не только закрепляются теоретические знания, полученные во время обучения, но и отрабатываются необходимые практические умения для эффективной работы в профессиональной сфере, осваивается инструментарий для работы и взаимодействия в команде, оттачиваются коммуникативные навыки для общения с коллегами.

Данный документ представляет собой отчет по производственной практике, проходившей в период с 6 сентября по 27 ноября ПАО «Кировский завод «Маяк», г. Киров.

1. Общие сведения о предприятии

ПАО «Кировский завод «Маяк» был создан 30 июля 1941 года.

Физический адрес: Российская Федерация, Кировская область, город Киров, улица Молодой Гвардии, 67

Публичное акционерное общество (ПАО) — это общество, чьи акции и ценные бумаги, которые конвертируются в акции, публично размещаются или публично обращаются. То есть их можно свободно приобрести на рынке ценных бумаг.

Сегодня это крупное машиностроительное предприятие, специализирующееся на выпуске товаров народного потребления.

1. Описание выполненной работы

2.1. Подключение элементов

Для того, чтобы подключить какой-нибудь элемент к плате нужно

1. установить элемент на макетной плате
2. добавить резистор, сопротивление которого вычислить по формуле

**R = (Vps — Vdf) / If**

Где Vps — напряжение источника питания (5В)

Vdf — падение напряжения на элементе

If — номинальный ток элемента

1. подключить к питанию
2. написать скетч для управления

2.1.1. Подключение светодиода

Vdf = 2 Вольта, If = 20 мА = 0.02 А=> R=150 Ом

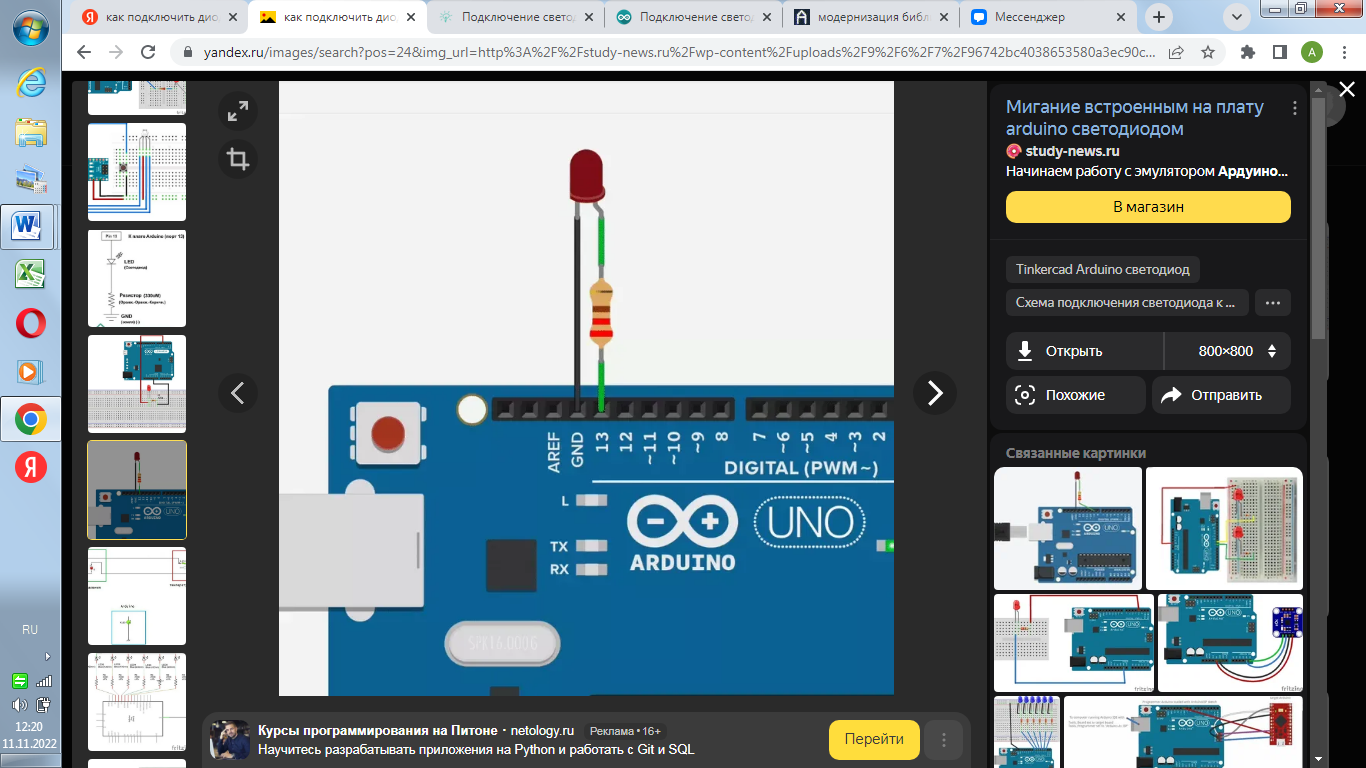


Рисунок 1 – подключение светодиода к плате

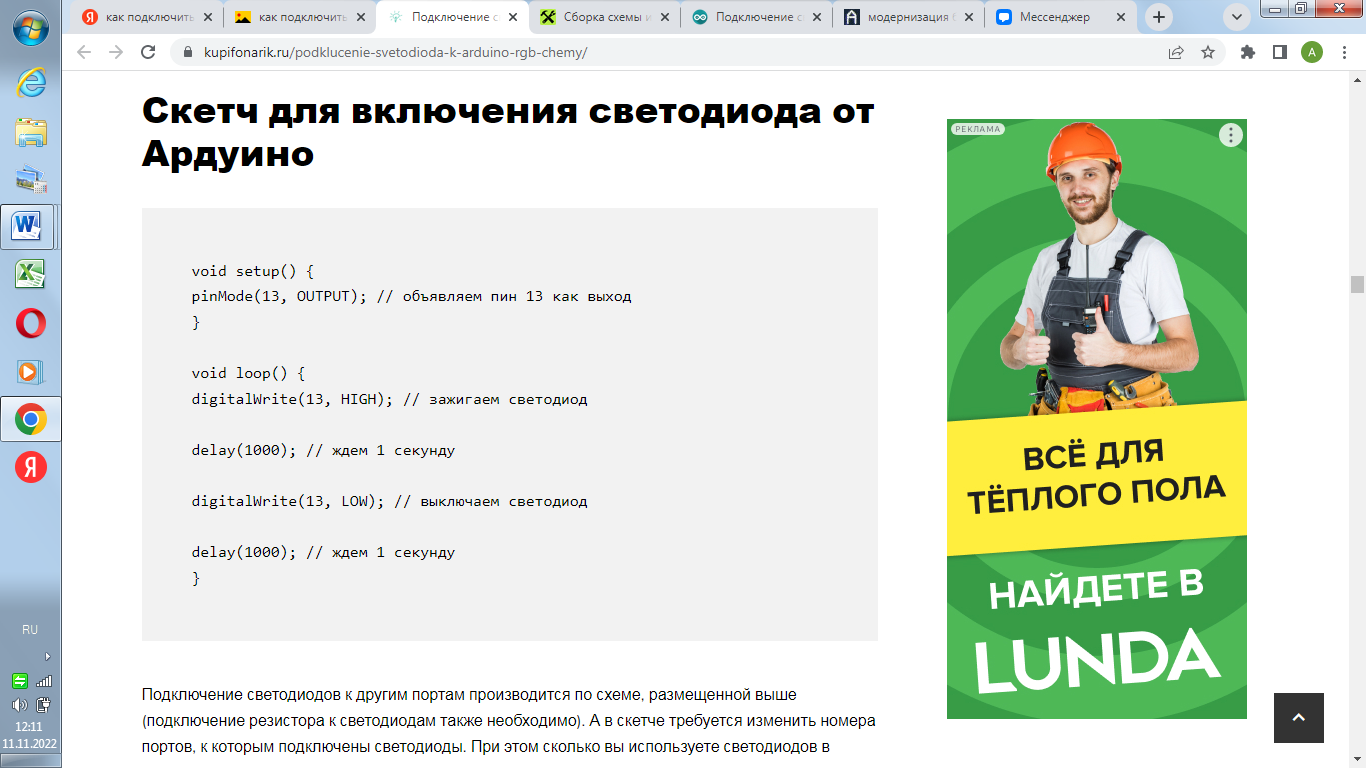


Рисунок 2 – скетч для управления светодиодом

2.1.2. Подключение dip переключателя к плате

Для более наглядной демонстрации работы дип переключателя к выходам были подключены светодиоды. Их подключение было рассмотрено в предыдущем пункте.

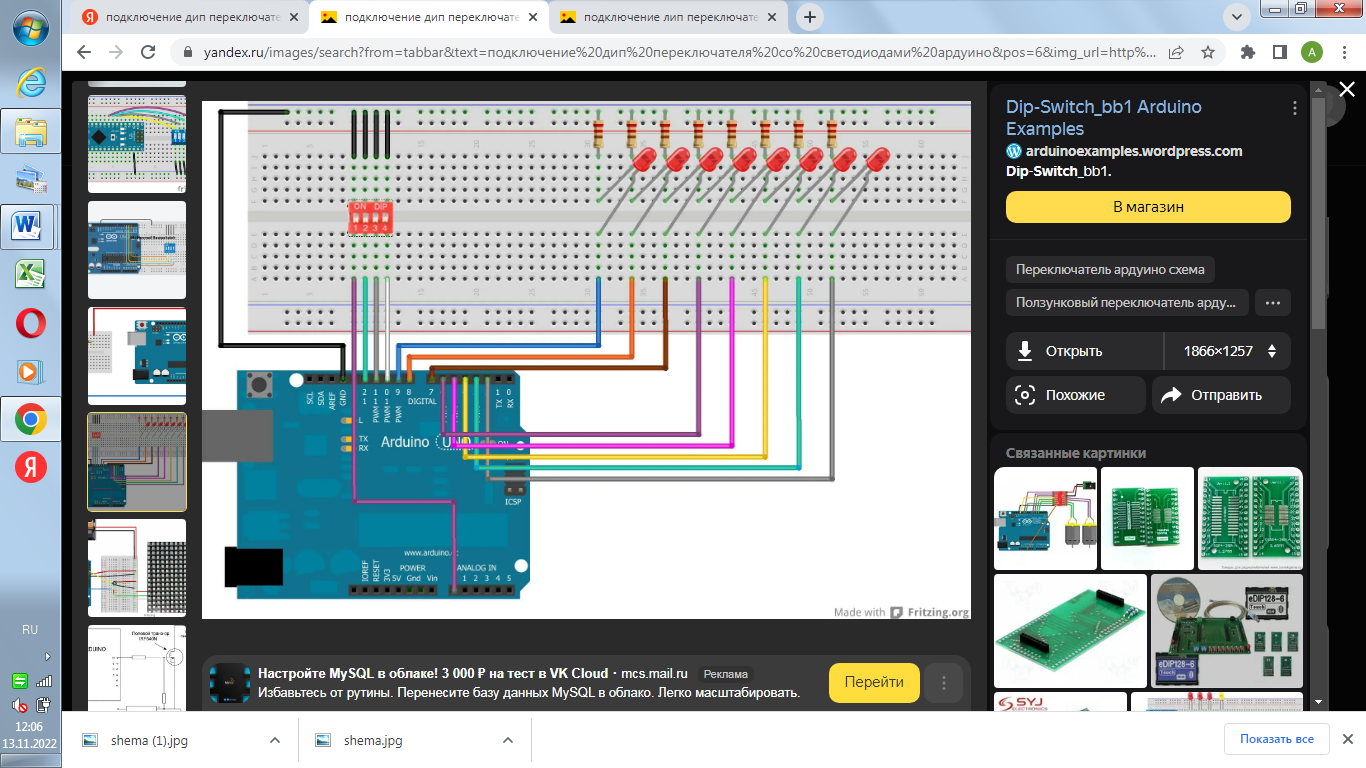


Рисунок 3 – подключение dip переключателя

#define LED1 2

#define LED2 3

#define LED3 4

#define LED4 5

#define S1 10

#define S2 11

#define S3 12

#define S4 А0

int s1state;

int s2state;

int s3state;

int s4state;

void setup() {

pinMode(LED1, OUTPUT);

pinMode(LED2, OUTPUT);

pinMode(LED3, OUTPUT);

pinMode(LED4, OUTPUT);

pinMode(S1, INPUT);

pinMode(S2, INPUT);

pinMode(S3, INPUT);

pinMode(S4, INPUT);

// setup serial port

Serial.begin(9600);

Serial.println("Serial port open");

}

Рисунок 4 – скетч для управления dip переключателем

void loop() {

s1state = digitalRead(S1);

digitalWrite(LED1, s1state);

s2state = digitalRead(S2);

digitalWrite(LED2, s2state);

s3state = digitalRead(S3);

digitalWrite(LED3, s3state);

s4state = digitalRead(S4);

digitalWrite(LED4, s4state);

Serial.print(s1state);

Serial.print(s2state);

Serial.print(s3state);

Serial.print(s4state);

Serial.println();

}

Рисунок 5 – скетч для работы с dip переключателем продолжение

2.1.3. Подключение регистров

Подключать несколько одинаковых элементов непосредственно к плате может быть не очень рациональным решением, так как количество пинов на плате ограниченно. Для решения этой проблемы можно использовать регистры, ко входам которых можно подсоединить элементы.

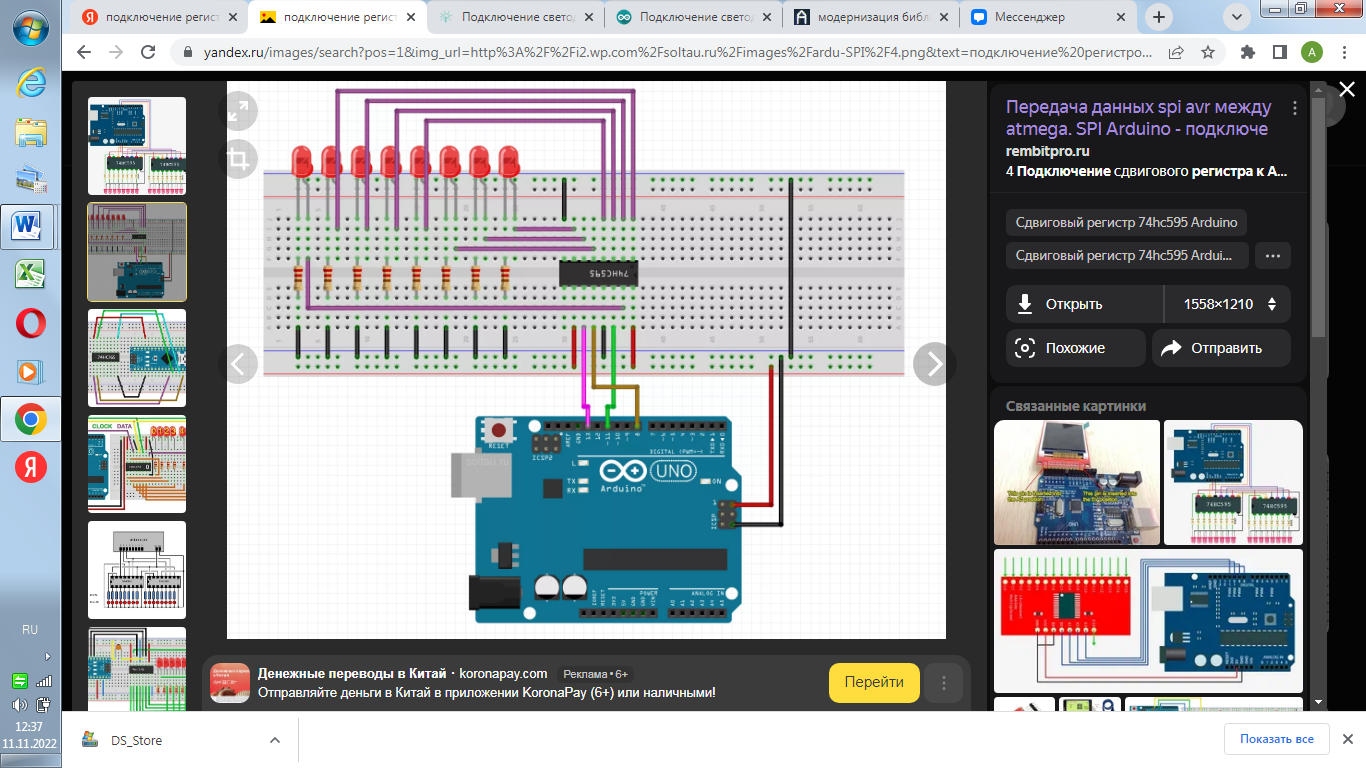


Рисунок 6 – подключение регистра к плате

const int latchPin=11;

const int dataPin=8;

const int clockPin= 13;

void setup()

{

pinMode(latchPin, OUTPUT);

pinMode(clockPin, OUTPUT);

pinMode(dataPin, OUTPUT);

}

void loop()

{

digitalWrite(latchPin, LOW);

shiftOut(dataPin, clockPin, LSBFIRST, 255);

digitalWrite(latchPin, HIGH);

delay(200);

}

Рисунок 7 – скетч для управления регистром

2.2. Создание каскада из регистров

Во время разработки устройства может возникнуть проблема, когда количество элементов, которые мы хотим подключить к регистру больше, чем количество входов этого регистра. Для решения этой проблемы можно создать каскад регистров.

2.3. Разработка устройства для автоматизированной проверки блоков релейной логики

2.3.1. Формирование требований к функционалу

Устройство автоматизированной проверки должно обладать следующим функционалом:

1. выбор режима работы – проверка или диагностирование
2. В режиме проверки 2 возможности – проверка и сброс
3. установка состояний входов
4. Чтение состояний выходов
5. проверка состояний входов и выходов в соответствии заданному алгоритму
6. выдача результатов о состоянии проверки

2.3.2. Выбор инструментов

Для реализации выбора режима диагностирования и возможностей проверки использовался 2 разрядный dip переключатель. Первый разряд – выбор режима. Второй разряд – проверка или сброс

Для установки состояний выходов в режиме диагностирования использовалось 2 8 разрядных dip переключателей. Их подключение к плате осуществлялось при помощи 2 сдвиговых регистров 165 серии

Для отображения состояний входов использовались 16 светодиодов, которые подключались к плате при помощи 2 сдвиговых регистров 595 серии.

Для отображения состояний проверки использовались 3 светодиода: зеленый, желтый и красный. Желтый загорался в тот момент, когда проверка или диагностирование запрещено. Зеленый – проверка прошла успешно. Красный – проверка выявила неисправности, устройство не исправно.

Для осуществления проверки были разработано устройство с 16 коммутируемыми цепями. Так как основной задачей было разработка устройства проверки, то разработка устройства для проверки не приводится.

2.3.3. Разработка устройства

В результате было разработано устройство автоматизированной проверки блоков релейной логики, электрическая схема которого представлена в приложении А.

Программа для управления схемой представлена в приложении Б.

3. Вывод

В ходе выполнения производственной практики были получены навыки работы с конструктором Arduino для изготовления макетов устройств. Кроме того, были получены знания о применении регистров для увеличения числа входов. Все полученные навыки и знания были применены для разработки устройства автоматизированной проверки.

Приложение А

Схема разработанного устройства

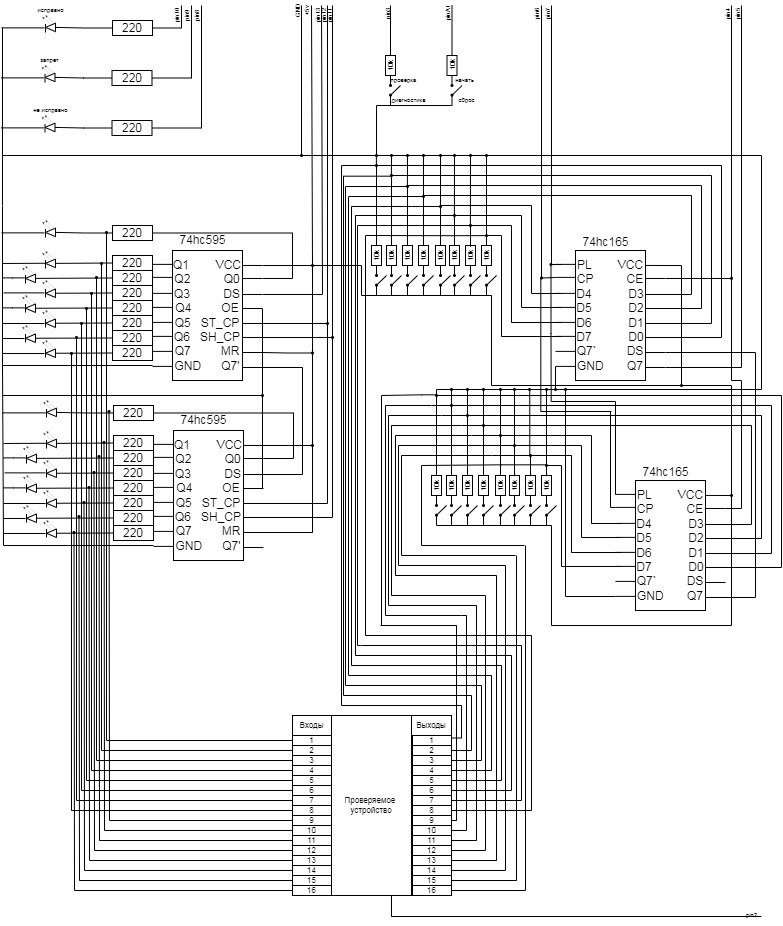


Рисунок 8 –схема разработанного устройства

Приложение Б

Листинг кода

//установка соответствия между названием и номером пина

//595 регистр

int latchPin=12;

int clockPin=11;

int dataPin=13;

//165 регистр

uint8\_t latchIn=7;

uint8\_t clockIn=6;

uint8\_t dataIn=5;

uint8\_t clockEnableIn=4;

//настройки

int regim =3;

int number=1;

int ledRed=8;

int ledYellow=9;

int ledGreen=10;

int vcc=2; //проверка питания

int for\_check=A1;

//инициализация

void setup() {

Serial.begin(9600); //подключение монитора порта

//установка пинов на вход или на выход

//led

pinMode(ledGreen, OUTPUT);

pinMode(ledRed, OUTPUT);

pinMode(ledYellow, OUTPUT);

//165

pinMode(latchIn, OUTPUT);

pinMode(clockIn, OUTPUT);

pinMode(dataIn, INPUT);

pinMode(clockEnableIn, OUTPUT);

//595

pinMode(latchPin, OUTPUT);

pinMode(dataPin, OUTPUT);

pinMode(clockPin, OUTPUT);

//настройки

pinMode(regim, INPUT);

pinMode(vcc, INPUT);

pinMode(for\_check, INPUT);

//начальная установка

digitalWrite(latchPin, LOW);//установить защелку, начать передавать бит

shiftOut(dataPin, clockPin, LSBFIRST, 0);

shiftOut(dataPin, clockPin, LSBFIRST, 0);

digitalWrite(latchPin, HIGH); // сигнализировать, что больше не надо считывать

}

//функция диагностики

void diagnostic()

{

//записывание

digitalWrite(latchPin, LOW);//установить защелку, начать передавать бит

//сдвинуть байт данных по одному биту за раз, записывая на вывод данных

shiftOut(dataPin, clockPin, LSBFIRST, 0);

//так как 16 разрядов - 2 байта, поэтому 2 функции

shiftOut(dataPin, clockPin, LSBFIRST, 0);

digitalWrite(latchPin, HIGH); // сигнализировать, что больше не надо считывать

//устанавить защелку, чтобы захватить состояния выходов

digitalWrite(latchIn, LOW);

delayMicroseconds(5);

digitalWrite(latchIn, HIGH);

delayMicroseconds(5);

//считывание

digitalWrite(clockIn, HIGH); //без этого не считывает последний бит

digitalWrite(clockEnableIn, LOW); //установить запись, начать считывать бит

//сдвигаем байт данных по одному за раз, записывая на ввод данных

uint8\_t in1=shiftIn(dataIn, clockIn, LSBFIRST);

uint8\_t in2=shiftIn(dataIn, clockIn, LSBFIRST);

digitalWrite(clockEnableIn, HIGH); //сигнализировать, что больше не надо считывать

//вывод в монитор порта

Serial.println(in1, DEC);

Serial.println(in2,DEC);

Serial.print('\n');

//записывание

digitalWrite(latchPin, LOW);//установить защелку, начать передавать бит

shiftOut(dataPin, clockPin, LSBFIRST, in2);

shiftOut(dataPin, clockPin, LSBFIRST, in1);

digitalWrite(latchPin, HIGH); // сигнализировать, что больше не надо считывать

delay(10);

}

//функция проверки устройства

void check(byte data, uint8\_t \*in1, uint8\_t \*in2)

{

//записывание

digitalWrite(latchPin, LOW);//установить защелку, начать передавать бит

shiftOut(dataPin, clockPin, LSBFIRST, data);

shiftOut(dataPin, clockPin, LSBFIRST, data);

digitalWrite(latchPin, HIGH); // сигнализировать, что больше не надо считывать

//устанавить защелку, чтобы захватить состояния выходов

digitalWrite(latchIn, LOW);

delayMicroseconds(5);

digitalWrite(latchIn, HIGH);

delayMicroseconds(5);

//считывание

digitalWrite(clockIn, HIGH); //без этого не считывает последний бит

digitalWrite(clockEnableIn, LOW); //устанавить запись, начать считывать бит

\*in1=shiftIn(dataIn, clockIn, LSBFIRST);

\*in2=shiftIn(dataIn, clockIn, LSBFIRST);

digitalWrite(clockEnableIn, HIGH); //сигнализировать, что больше не надо считывать

}

//функция проверки состояний переключателя в режиме проверки

void check\_begining(uint8\_t \*in1, uint8\_t \*in2)

{

//устанавливаем защелку, чтобы захватить состояния выходов

digitalWrite(latchIn, LOW);

delayMicroseconds(5);

digitalWrite(latchIn, HIGH);

delayMicroseconds(5);

//считывание

digitalWrite(clockIn, HIGH); //без этого не считывает последний бит

digitalWrite(clockEnableIn, LOW); //установить запись, начать считывать бит

\*in1=shiftIn(dataIn, clockIn, LSBFIRST);

\*in2=shiftIn(dataIn, clockIn, LSBFIRST);

digitalWrite(clockEnableIn, HIGH); //сигнализировать, что больше не надо считывать

}

//основная программа

void loop() {

uint8\_t in1;

uint8\_t in2;

int s1=digitalRead(regim); //считывание состояние переключателя режима

int s2=digitalRead(vcc); // считывание состояние питания

int s3=digitalRead(for\_check); //считываем состояние переключателя проверки

if (s1==0) //режим диагностики

{

if (s2==0) //не подключено проверяемое устройство

{

digitalWrite(ledRed, LOW);// все диоды потушить

digitalWrite(ledYellow, LOW);

digitalWrite(ledGreen, LOW);

diagnostic(); //диагностика

}

else //проверяемое устройство подключено

{

digitalWrite(ledRed, LOW);

digitalWrite(ledYellow, HIGH); //зажечь желтый диод - диагностика не возможна

digitalWrite(ledGreen, LOW);

}

number=1; //номер проверки =1

}

else //режим проверки

{

//подать нули на входы

digitalWrite(latchPin, LOW);//установить защелку, начать передавать бит

shiftOut(dataPin, clockPin, LSBFIRST, 0);

shiftOut(dataPin, clockPin, LSBFIRST, 0);

digitalWrite(latchPin, HIGH); // сигнализировать, что больше не надо считывать

check\_begining(&in1, &in2); //проверить все ли переключатели, задающие выходы в режиме диагностики, выключены

Serial.println(in1, DEC);

Serial.println(in2,DEC);

Serial.print('\n');

delay(1000);

if ((in1!=0 ||in2!=0) && number==1 || s2==0 ) //переключатели не выключены в момент первой проверки или проверяемое устройство не подключено

{

digitalWrite(ledYellow, HIGH); //зажечь желтый диод - проверка не возможна

}

else //всё норм

{

if (s3==1)//начать проверку

{

switch(number) //номер проверки

{

case 1:

{

digitalWrite(ledGreen, LOW);

digitalWrite(ledYellow, LOW);

digitalWrite(ledGreen, LOW);

check(0, &in1, &in2);//проверка

//вывод в монитор порта

Serial.println(in1, DEC);

Serial.println(in2,DEC);

Serial.print('\n');

if (in1==0 && in2==0)//сравнить полученные значения с эталонными

{

digitalWrite(ledGreen, HIGH); //проверка прошла

number++; //следующая проверка

} //if

else

{

digitalWrite(ledRed, HIGH);//не исправно

number=5;//заканчиваем проверку

}//else

delay(1000);

break;

}//1

case 2:

{

digitalWrite(ledGreen, LOW);

digitalWrite(ledYellow, LOW);

digitalWrite(ledGreen, LOW);

delay(1000);

check(51, &in1, &in2);//проверка

//вывод в монитор порта

Serial.println(in1, DEC);

Serial.println(in2,DEC);

Serial.print('\n');

if (in1==51 && in2==51 )

{

digitalWrite(ledGreen, HIGH);

number++;

} //if

else

{

digitalWrite(ledRed, HIGH);

number=5;

}//else

delay(1000);

break;

}//2

case 3:

{

digitalWrite(ledGreen, LOW);

digitalWrite(ledYellow, LOW);

digitalWrite(ledGreen, LOW);

delay(1000);

check(204, &in1, &in2);//проверка

//вывод в монитор порта

Serial.println(in1, DEC);

Serial.println(in2,DEC);

Serial.print('\n');

if (in1==136 && in2==136 )

{

digitalWrite(ledGreen, HIGH);

number++;

} //if

else

{

digitalWrite(ledRed, HIGH);

number=5;

}//else

delay(1000);

break;

}//3

case 4:

{

digitalWrite(ledGreen, LOW);

digitalWrite(ledYellow, LOW);

digitalWrite(ledGreen, LOW);

delay(1000);

check(255, &in1, &in2);//проверка

//вывод в монитор порта

Serial.println(in1, DEC);

Serial.println(in2,DEC);

Serial.print('\n');

if (in1==221 && in2==221 )

{

digitalWrite(ledGreen, HIGH);

number++;

} //if

else

{

digitalWrite(ledRed, HIGH);

number=5;

}//else

delay(1000);

break;

}

default:

{

//записывание

digitalWrite(latchPin, LOW);//установить защелку, начинать передавать бит

shiftOut(dataPin, clockPin, LSBFIRST, 0); //сброс входов

shiftOut(dataPin, clockPin, LSBFIRST, 0);

digitalWrite(latchPin, HIGH); // сигнализировать, что больше не надо считывать

break;

}

}//switch

} // готовы к проверке

else //сброс

{

number=1;

digitalWrite(ledGreen, LOW);

digitalWrite(ledYellow, LOW);

digitalWrite(ledRed, LOW);

}

}//питание

} // regim = check

}