Практическое занятие №
Тема: «4 разрядный 7-сегментный индикатор и сдвиговый регистр 74HC595N»

Цель работы: приобрести практические навыки по подключению и программированию 7-сегментного индикатора и сдвигового регистра 74HC595N на платформе Arduino.

Последовательность выполнения работы:

– Собрать схемы на макетной плате, иначе при отсутствии набора Arduino в web-приложениях (https://wokwi.com/projects/new/arduino-uno или https://www.tinkercad.com/) для приведенных примеров.

– Запрограммировать микроконтроллер согласно заданию в примере.

– Выполнить задание для самостоятельной работы.

Содержание отчета:

– Название практического занятия, его цель.

– Фото или скриншоты собранной схемы.

– Написанный программный код вставить текстом, Courier New, 12 кегль, одинарный отступ без абзацев.

– Вывод о проделанной работе.

– Файл Fritzing с принципиальной и монтажной схемой.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

4-разрядный 7-сегментный индикатор

4-разрядный 7-сегментный индикатор — это электронное устройство, представляющее собой дисплей с четырьмя отдельными 7-сегментными индикаторами. Каждый индикатор может отображать цифры от 0 до 9, а также некоторые буквы.

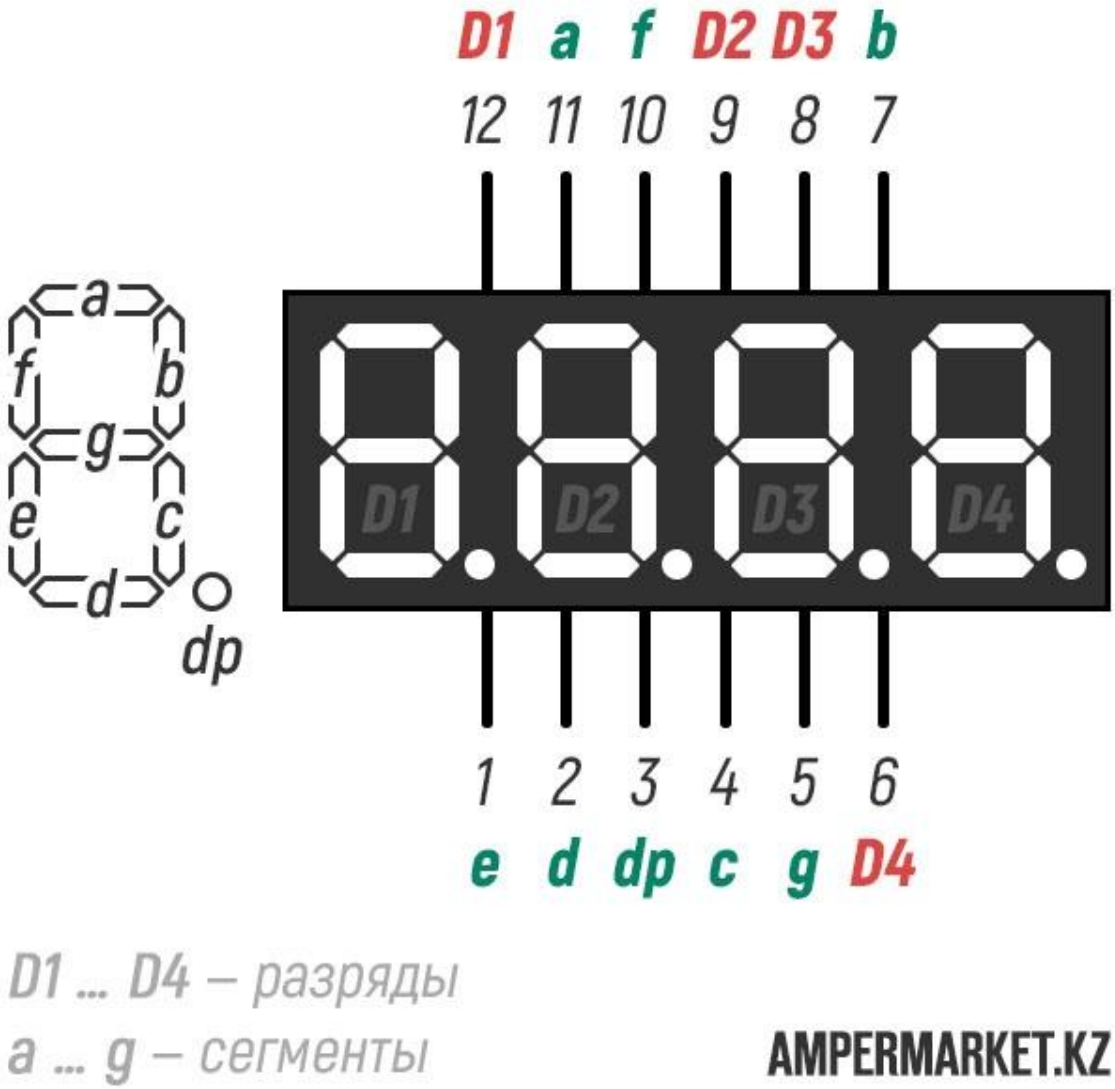

Рисунок 1 – 4 разрядный 7-сегментный цифровой LED индикатор

Восьмиразрядный сдвиговый регистр

74HC595N – восьмиразрядный сдвиговый регистр с последовательным вводом, последовательным или параллельным выводом информации, с триггером-защелкой и тремя состояниями на выходе. Самое распространенное применение данного регистра – экономия выходов микроконтроллера. Данный сдвиговый регистр позволяет управлять напряжением на своих восьми выходах, заняв всего три выхода микроконтроллера. Таким образом количество рабочих выводов увеличивается на пять.

Кроме того, регистры 74HC595 можно подключать каскадом один за другим (через пин **Q7'**), и таким образом из всё тех же 3 входящих линий получать 16, 24, 32 и т.д. цифровых выходов.

74HC595N имеет следующие входы:

– **[10] MR** — сброс регистра, при подаче логического нуля на MR и единицы на STCP переводит все выходы в состояние логического нуля;

– **[11] SH_CP** — вход для тактовых импульсов;

– **[12] ST_CP** — линия прерываний;

– **[13] OE** — вход, переводящий выходы из высокоимпедансного состояния в рабочее;

– **[14] DS** — вход данных;

– **[8] GND** — Ground. Земля

– **[16] VCC** — Питание +5 В.

Рисунок 2 – Восьмиразрядный сдвиговый регистр

ЗАДАНИЯ

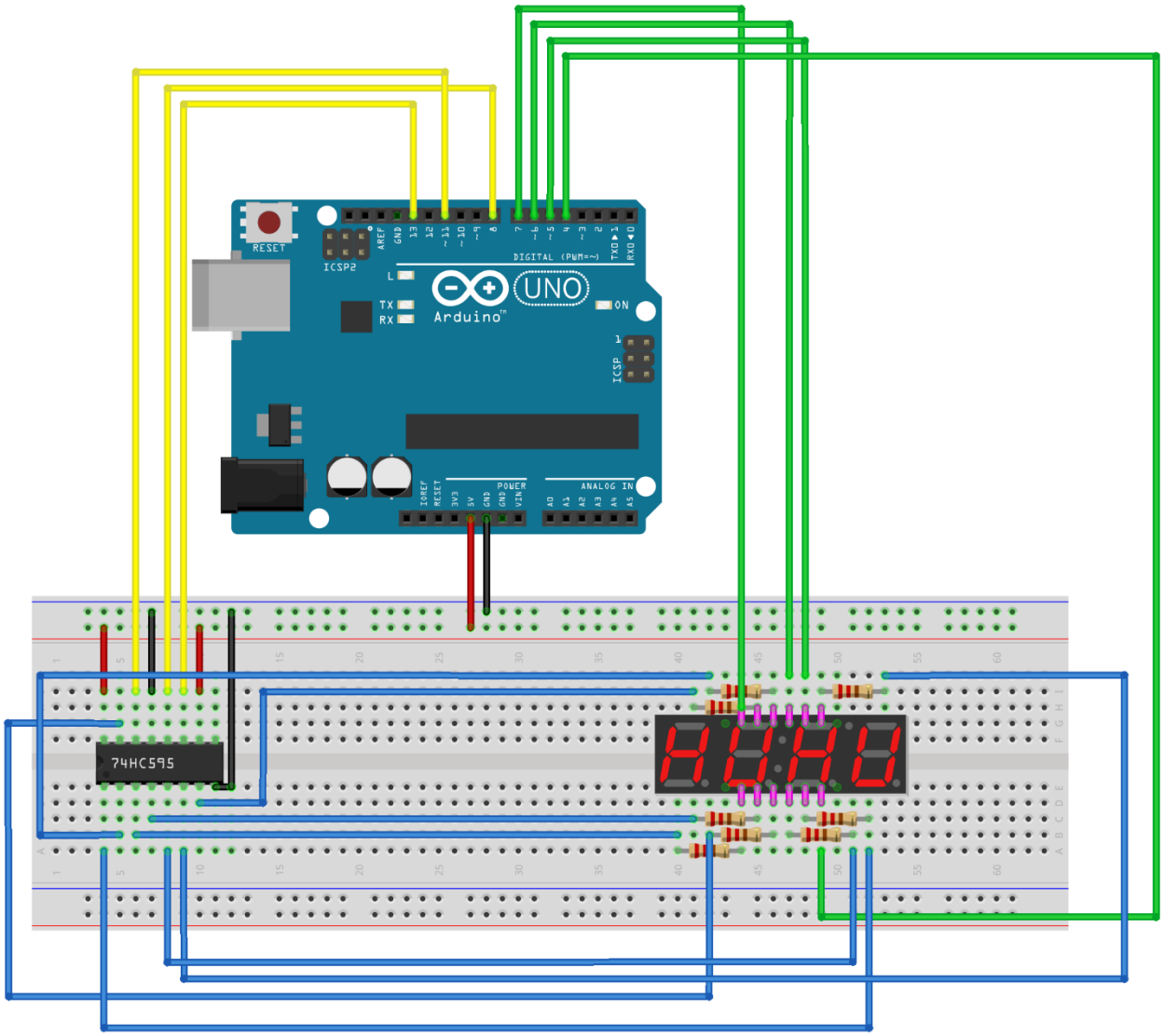

Рисунок 3 – Схема подключения

Программный код:

```
#include <SPI.h>

// Пин для latch (SS)
const int pin_spi_ss = 8;

byte numbers[10] = {
  //ABCDEFGDp
  B00000011, // 0 - A,B,C,D,E,F
  B10011111, // 1 - B,C
  B00100101, // 2 - A,B,D,E,G
  B00001101, // 3 - A,B,C,D,G
  B10011001, // 4 - B,C,F,G
  B01001001, // 5 - A,C,D,F,G
```

```
  B01000001, // 6 - A,C,D,E,F,G
  B00011111, // 7 - A,B,C
  B00000001, // 8 - A,B,C,D,E,F,G
  B00001001   // 9 - A,B,C,D,F,G
};

// Переменные для хранения значений
int number = 0;
int number1 = 0;
int number2 = 0;

// Пины для управления разрядами индикатора (катоды)
int pindigits[4] = {4, 5, 6, 7};

// Буфер для ввода данных
String inputString = "";
bool stringComplete = false;

void setup() {
  Serial.begin(9600);
  SPI.begin();

  // Настройка пина latch как выхода
  pinMode(pin_spi_ss, OUTPUT);
  digitalWrite(pin_spi_ss, HIGH);

  // Настройка пинов разрядов как выходов
  for (int i = 0; i < 4; i++) {
    pinMode(pindigits[i], OUTPUT);
    digitalWrite(pindigits[i], LOW); // Изначально выключаем
все разряды
  }

  Serial.println("Инициализация завершена");
  Serial.println("Введите число от 0 до 9999:");
  Serial.println("(завершите ввод нажатием Enter)");
}

void loop() {
  // Обработка ввода из последовательного порта
  if (stringComplete) {
    // Преобразуем строку в число
    number = inputString.toInt();
```

```
    // Ограничиваем число диапазоном 0-9999
    if (number < 0) number = 0;
    if (number > 9999) number = 9999;

    Serial.print("Отображаю число: ");

    // Выводим число с ведущими нулями
    if (number < 10) {
      Serial.print("000");
    } else if (number < 100) {
      Serial.print("00");
    } else if (number < 1000) {
      Serial.print("0");
    }
    Serial.println(number);

    // Очищаем строку для следующего ввода
    inputString = "";
    stringComplete = false;
  }

  // Разделяем число на цифры и отображаем их
  number1 = number;
  for (int i = 0; i < 4; i++) {
    number2 = number1 % 10;
    number1 = number1 / 10;

    // Выключаем все разряды перед обновлением
    for (int j = 0; j < 4; j++) {
      digitalWrite(pindigits[j], LOW);
    }

    // Включаем текущий разряд
    digitalWrite(pindigits[i], HIGH);

    // Выводим цифру на индикатор
    showNumber(number2);

    // Короткая задержка для динамической индикации
    delay(2);
  }
}

// Функция вывода цифры на семисегментный индикатор
```

```
void showNumber(int num) {
  digitalWrite(pin_spi_ss, LOW);
  SPI.transfer(numbers[num]);
  digitalWrite(pin_spi_ss, HIGH);
}

// Обработка данных из последовательного порта
void serialEvent() {
  while (Serial.available()) {
    char inChar = (char)Serial.read();

    // Если получен символ новой строки, устанавливаем флаг
    if (inChar == '\n') {
      stringComplete = true;
    } else if (inChar >= '0' && inChar <= '9') {
      // Добавляем только цифры в строку
      inputString += inChar;
    }
    // Игнорируем все остальные символы
  }
}
```