КОНКУРСНОЕ ЗАДАНИЕ

III региональный чемпионат
ЮниорПрофи 2019

компетенция

Электроника 14+

Задание: Изготовление прибора «Автономный контроллер дистанционного мониторинга и управления».

Назначение прибора.

Сбор, обработка данных с внешних датчиков.

Время на выполнение задания: 3 дня соревнований.

Задание.

Модули задания:

1. Монтаж электронного модуля прибора.
2. Сборка прибора.
3. Наладка, поиск неисправностей.
4. Выявление и устранение механических неполадок.
5. Настройка модулей Bluetooth на уникальное автоматическое соединение.
6. Программирование модуля датчиков на сбор данных и подготовку их к передаче.
7. Программирование «Автономного контроллера дистанционного мониторинга и управления» на выполнение основных функций прибора.
8. Проверка работоспособности прибора на стенде.

Модуль 1. Монтаж печатной платы прибора.

Установка компонентов осуществляется на основании спецификации и чертежа монтажной платы.

Порядок, особенности установки компонентов указываются в спецификации.

Электронная часть прибора состоит из трех блоков.

1. ATMEGA328DUAL
2. Keyboard
3. MAINUNIT

Блок **ATMEGA328DUAL** представляет собой печатную плату для установки микроконтроллера ATMEGA328 в DIP корпусе.

Блок **Keyboard** – блок клавиатуры. На плате размещаются тактовые кнопки и резисторы.

Блок **MAINUNIT** – «материнская плата». Блок **ATMEGA328DUAL** устанавливается на MAINUNIT с помощью штыревых разъемов.

Модуль RTC «часы реального времени» устанавливается в раз]ем на плате

MAINUNIT

Блок **Keyboard** подключается монтажным проводом.

Так же монтажным проводом к плате **MAINUNIT** подключается дисплей, джойстик.

Питание устройства осуществляется от батареи из 6 элементов питания по 1, 5В

Батарейный отсек подключается к клемме питания блока **MAINUNIT**.

Клеммы для подключения внешних устройств устанавливаются входными отверстиями в сторону края печатной платы.

Внешние модули крепятся на переднюю панель корпуса винтами и соединяются с соответствующими разъемами на плате монтажным проводом.

Модуль 2. Сборка прибора.

Плата прибора устанавливается на пластиковое шасси на металлические шестигранные стойки высотой 10 мм и закрепляется винтами М3 х 6 мм (стандарты ГОСТ 17473-80, DIN 7985 и ISO 7045).

Батарейный отсек закрепляется на пластиковом шасси винтами М3 х 6 мм и гайками М3.

Собранное шасси устанавливается на основание корпуса винтами М3 х 6 мм через шестигранные стройки.

Дисплей, блок клавиатуры и джойстик крепятся к передней панели согласно сборочному чертежу.

Дисплей крепится винтами М2 х 8 мм (8 шт) с полукруглой головкой (типа DIN 7985, ISO 7045) через стойки высотой 5 мм (типа PCHSS2-05).

Джойстик крепится винтами М2 х 8 мм (8 шт) с полукруглой головкой через стойки высотой 17-18 мм (типа PCHSS2-17).

На задней стенке корпуса устанавливается включатель и, при необходимости, интерфейсный разъем.

Основание прибора соединяется с боковыми стенками винтами М3 х 10 мм и гайками М3.

К боковым стенкам крепится лицевая панель винтами М3 х 10 мм и гайками М3.

На нижнюю стенку на клеящий слой по периметру устанавливаются 4 резиновые ножки.

После проверки и наладки прибора к боковым стенкам крепится крышка винтами М3 х 10 мм и гайками М3.

Модули 3- 4. Наладка, поиск неисправностей. Выявление и устранение механических неполадок

Для проверки работоспособности прибора в контроллер прибора загружается тестовая программа SelfTest. Файл программы, необходимые библиотеки и описания размещены в папке JS_2018 на рабочем столе компьютера.

После загрузки кода необходимо провести проверку работоспособности всех узлов

автомата:

- наличие и уровень питающего напряжения
- работа кнопок
- работа джойстика
- работа дисплея
- работа элементов индикации
- работа порта UART
-
- работа других элементов устройства.

В случае неисправности какого-либо узла, необходимо определить неисправный электронный компонент и произвести замену.

Модуль 5. Программирование модуля датчиков на сбор данных и подготовку к передаче

Участники выполняют набор из 7 заданий по программированию микроконтроллера на плате Arduino Uno.

В качестве датчиков и исполнительных элементов используются светодиоды, датчики температуры, влажности, движения, освещенности.

Установка датчиков, исполнительных элементов осуществляется на беспаечной макетной плате. Соединение компонентов производится проволочными

перемычками.

Программирование контроллера осуществляется в среде ARDUINO IDE.

Модуль 6. Программирование контроллера на выполнение основных функций прибора

Участники выполняют настройку имеющегося оборудования, осуществляют дистанционное соединение модулей. Подготовка прибора к проверке работоспособности на стенде с помощью прилагаемой программы mainTest.

Модуль 7. Проверка работоспособности прибора на стенде

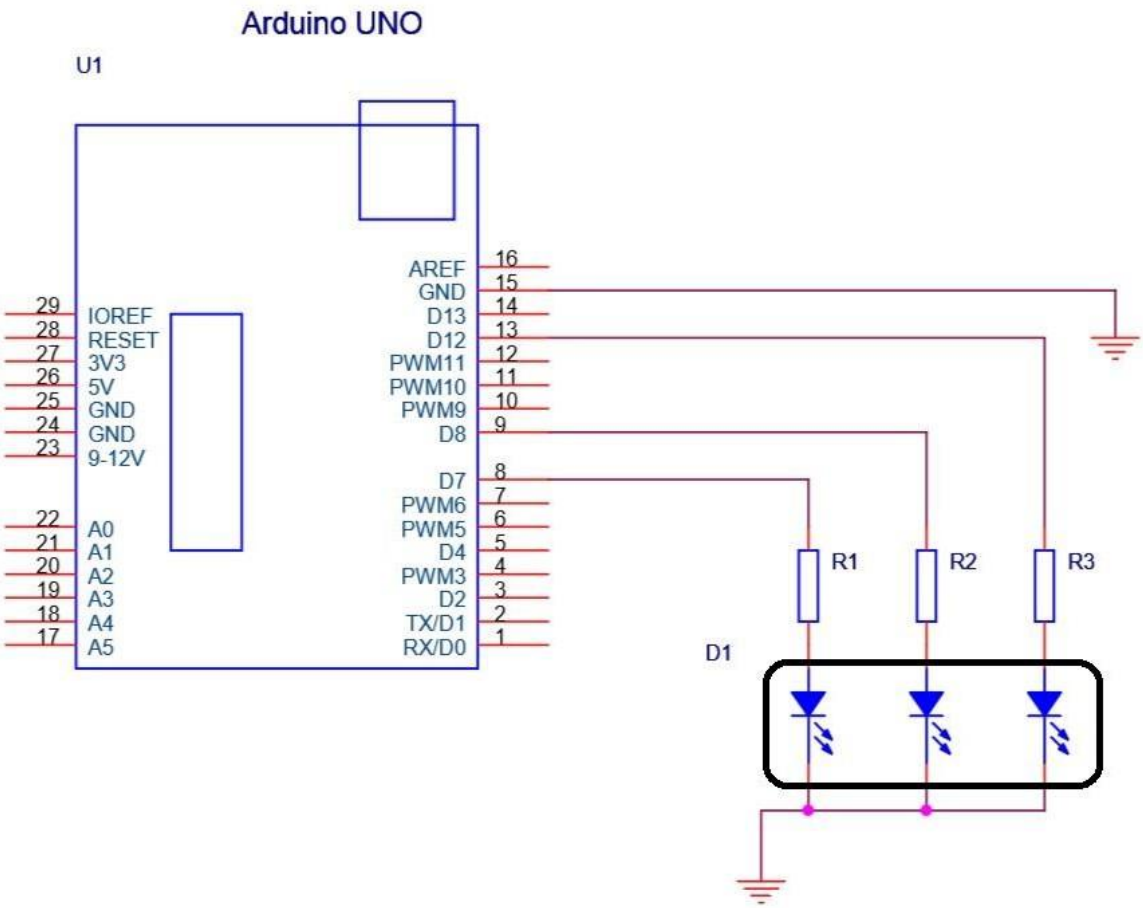
Участники проводят выходной контроль прибора на испытательном стенде. На стенде осуществляется контроль указанных в техническом задании параметров. Соответствие алгоритма работы автомата заданным требованиям.

Участники выполняют набор из 3 заданий по программированию микроконтроллера прибора.

Примеры заданий:

I. RGB светодиод

Выполните сборку схемы, представленную на рисунке ниже.



R1-R3 – 360 Ом

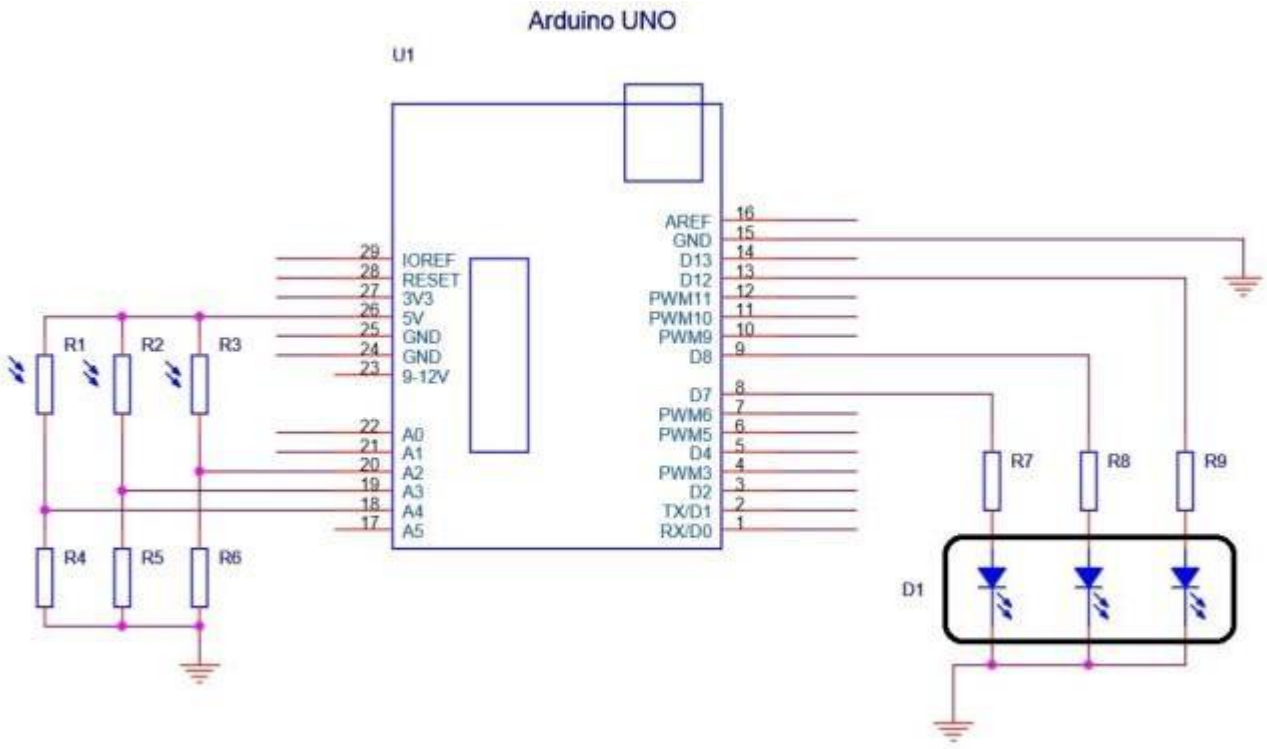
D1- RGB светодиод

Напишите программу для контроллера Arduino.

Светодиод может светить только 3 цветами – красный, зеленый, синий. Цвет светодиода выбирается случайным образом. Выбранный светодиод должен плавно загореться и затем плавно потухнуть. После этого производится случайный выбор другого светодиода. Повторное зажигание одного и того же светодиода не допускается.

II.Фоторезисторы управляют светодиодами

Выполните сборку схемы, представленную на рисунке ниже.



R1-R3 – фоторезисторы R4-R6 - 10к

R7-R9 – 360 Ом

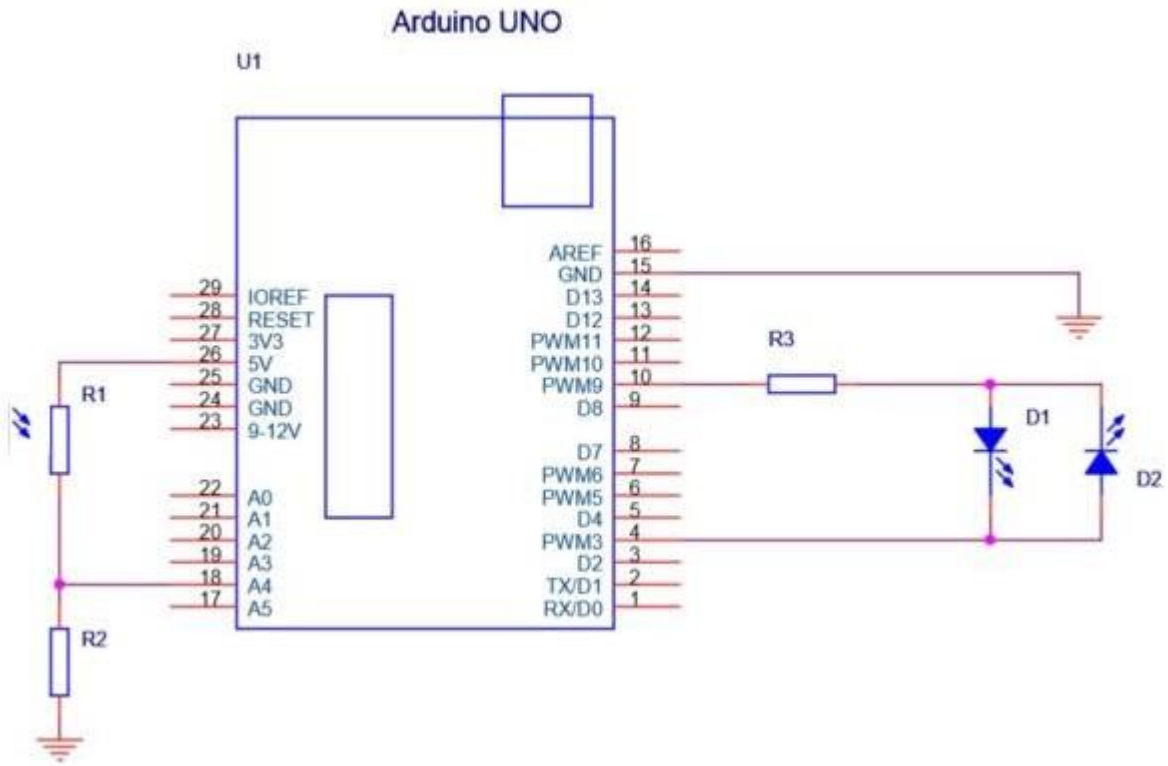
D1 – RGB светодиод с общим катодом

Напишите программу для контроллера Arduino. После запуска программы яркость свечения каждого светодиода должна регулироваться отдельным датчиком освещенности. При регулировке минимальная яркость свечения соответствующего цвета светодиода должна достигаться при полной

темноте, максимальная яркость свечения – при яркой освещенности датчика.

III.Светодиоды поочередно и освещенность

Установите элементы на макетную плату по схеме, представленной на рисунке ниже.

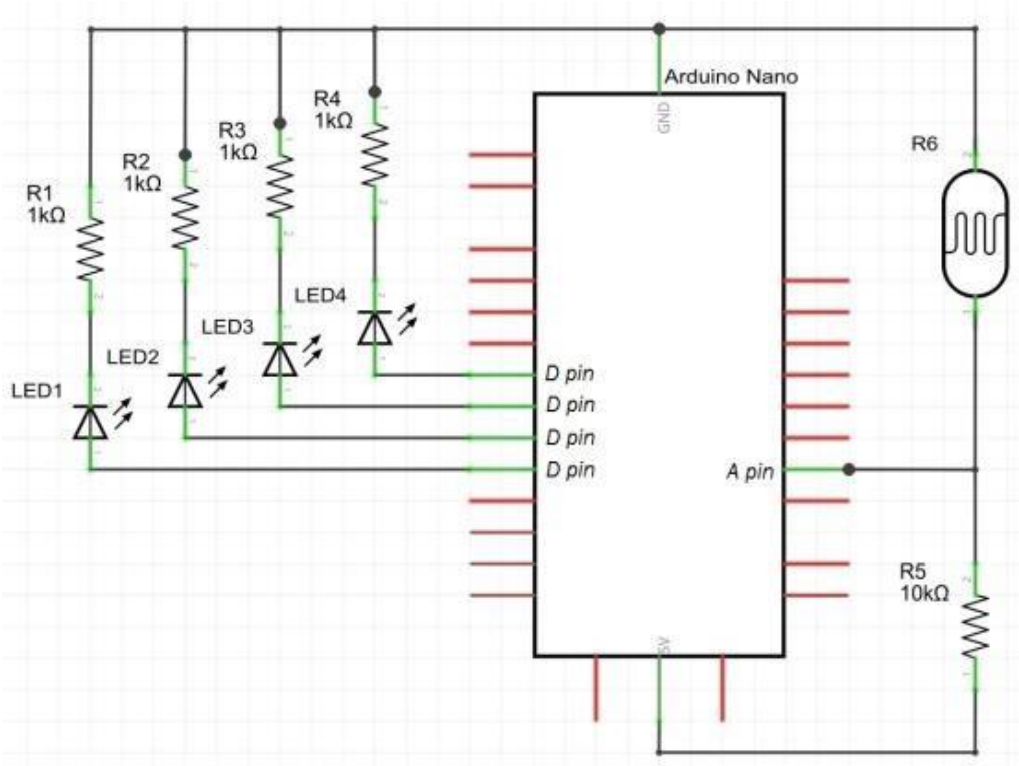


Напишите программу для контроллера Arduino. После запуска программы при максимальной освещенности светодиод LED1 должен светиться с максимально возможной яркостью, а светодиод LED2 не должен светиться. При уменьшении уровня освещенности яркость светодиода LED1 должна уменьшаться, светодиод LED2, по-прежнему, не должен светиться. При некотором среднем уровне освещенности оба светодиода не должны светиться. При дальнейшем уменьшении уровня освещенности светодиод LED1 не должен светиться, а яркость светодиода LED2 должна увеличиваться.

IV. Светобегущие огни

Выполните сборку схемы, представленную на рисунке ниже.

При подключении элементов LED1, LED2, LED3, LED4 необходимо самостоятельно выбрать порты Arduino согласно их назначению.

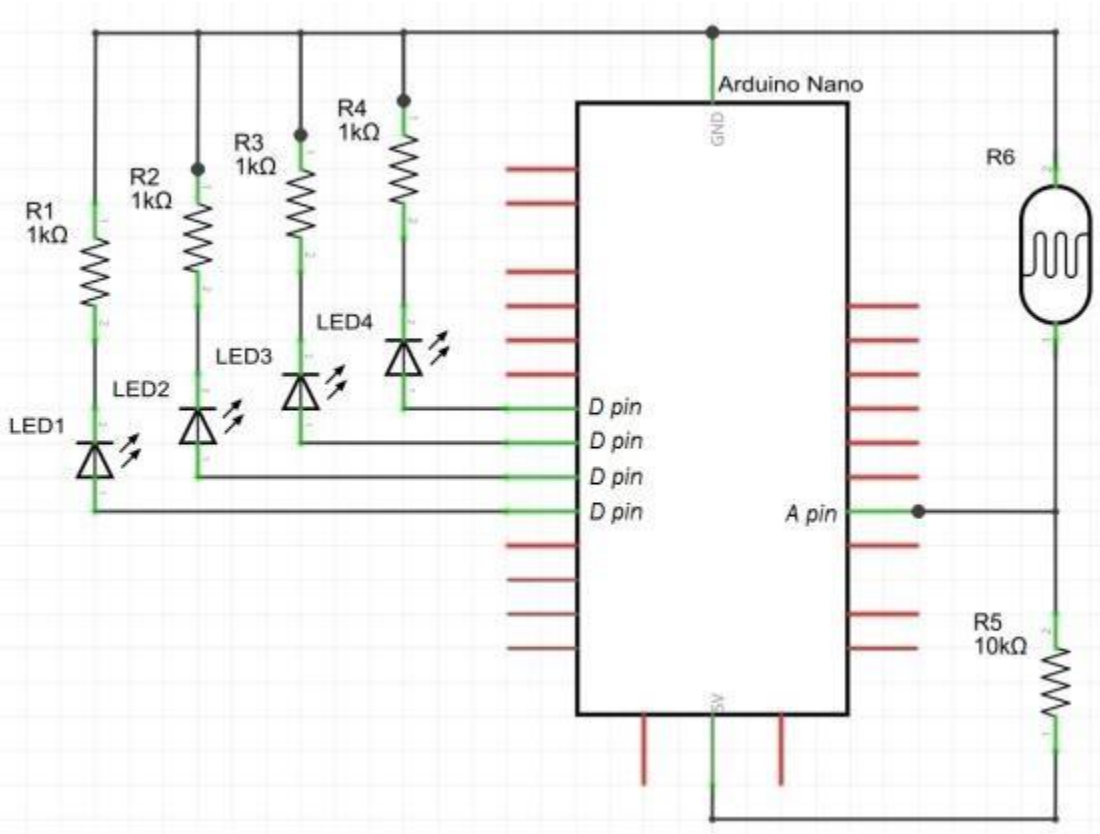


Напишите программу для контроллера Arduino. После запуска программы элементы LED1 – LED4 должны поочередно включаться. Интервал между включениями должен зависеть от значения, передаваемого от фоторезистора на *A pin* , после включения элемента LED4 элементы LED1 – LED4 должны выключаться в таком же порядке. Интервал между выключениями элементов должен зависеть от значения, передаваемого от фоторезистора на *A pin*.

V. Индикатор освещенности

Выполните сборку схемы, представленную на рисунке ниже.

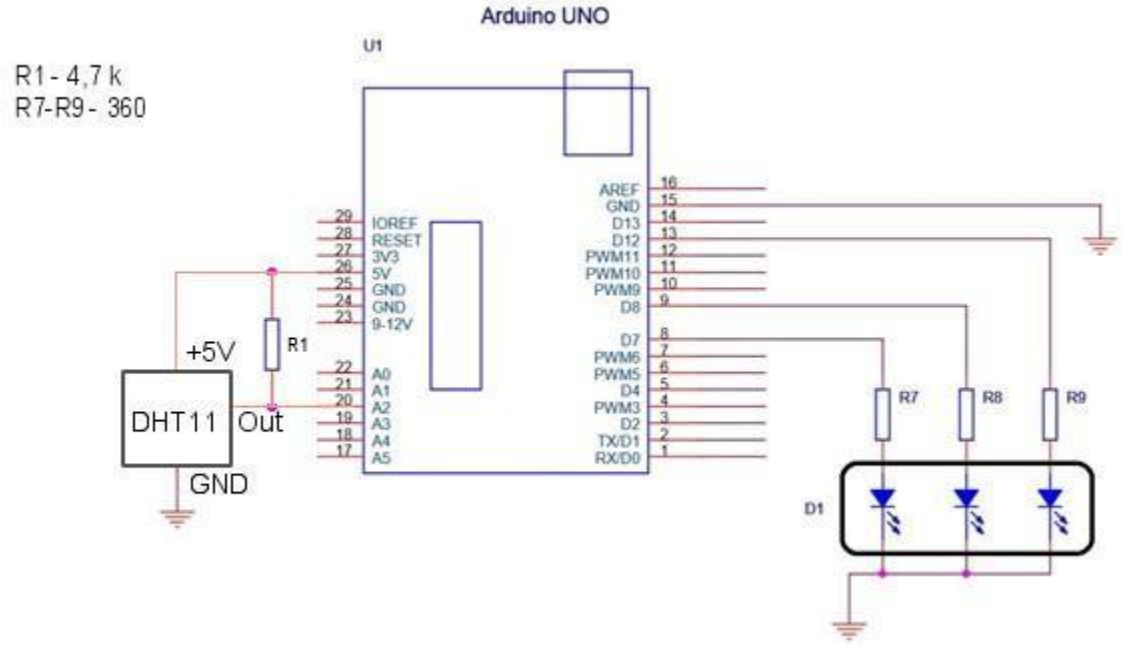
При подключении элементов LED1, LED2, LED3, LED4 необходимо самостоятельно выбрать порты Arduino согласно их назначению.



Напишите программу для контроллера Arduino. После запуска программы включение элементов LED1 – LED4 должно зависеть от уровня освещенности. При полной темноте должен включиться элемент LED1. При постепенном увеличении освещенности поочередно должны включаться соответственно элементы LED2 – LED4 при этом уже включенные элементы не должны выключаться. При наиболее высоком освещении должны быть включены все элементы LED1-LED4. При снижении освещенности элементы LED1 – LED4 должны выключаться в обратном порядке.

VI.Индикатор температуры

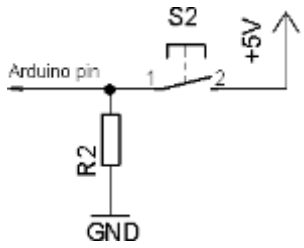
Выполните сборку схемы, представленную на рисунке ниже.



Напишите программу для контроллера Arduino. После запуска программы текущее показание датчика в помещении воспринимается как температура ниже комфортной (горит синий светодиод, «холодно»). При повышении температуры вначале зажигается зеленый светодиод (зона комфортной температуры «шириной» в 2 градуса), а затем красный (температура выше комфортной, «жарко»). Необходимо постараться поддерживать комфортную температуру.

VII.Удаленный светодиод. Задание с инструкцией

1. Подключите к первой макетной плате с Arduino тактовую кнопку по следующей схеме:



R2 – 10 кОм. S2 – тактовая кнопка. pin – любой, кроме 0, 1 и 13 Напишите программу для контроллера Arduino, которая анализирует

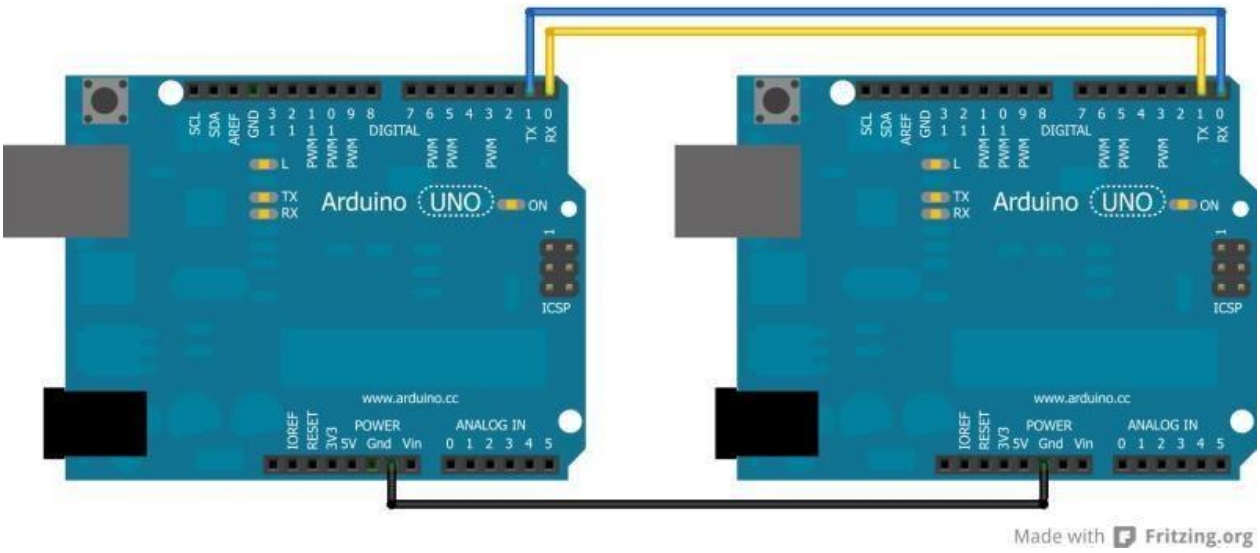
состояние тактовой кнопки и выводит в монитор порта значения «0», если кнопка не нажата, или «1», если кнопка нажата (новое сообщение с новой строки).

2. Подключите вторую плату Arduino.

Напишите программу, которая принимает значения «0» и «1», введенные в мониторе порта, и управляет встроенным светодиодом (пин 13):

получено «0» - светодиод выключить получено «1» - светодиод включить

3. Отсоедините USB-провод и соедините обе платы следующим образом:



Подключите батарейные блоки. Если правильно подключены платы и написаны программы, то светодиод на второй Arduino должен управляться кнопкой, подключенной к первой. Допускается использование одного батарейного блока, с соединением пинов +5V обеих плат.

Что нужно знать:

- Скорость обмена информацией установите для обеих плат одинаковую, например, 9600 бод.

- Цифровые пины 0 и 1 используются для организации обмена данными через последовательный порт. TX – трансмиттер (передатчик), RX – ресивер (приемник). Если с одной платы данные ВЫВОДЯТСЯ (т.е. из пина TX), то в другую плату они будут ВВОДИТЬСЯ (т.е. в пин RX). В результате соединительные провода должны быть соединены «крест-накрест», от пина 0 первой платы к пину 1 второй платы, и наоборот.

- В связи с тем, что последовательный порт, организованный на пинах 0 и 1, используется также и встроенным USB-интерфейсом, USB-кабель необходимо отключать для проверки разрабатываемых программ при соединении двух плат Arduino.

- Соединение контактов GND двух плат – обязательно (вы же знаете, что для того, чтобы электрический ток начал свой путь, цепь должна быть замкнута?).

- Обратите внимание, что передаваемая информация («0» и «1») – это символы (char), а не цифры.

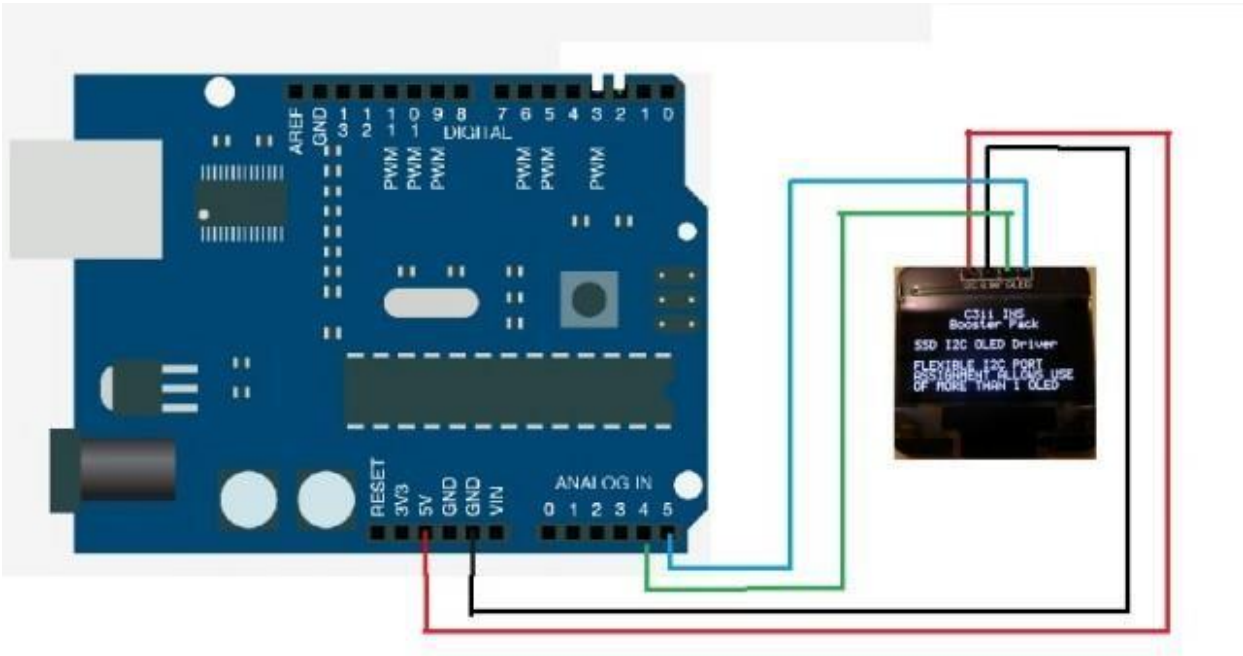
- В библиотеке Serial есть две похожие функции – print() и println(), различие между которыми заключается в том, что при выводе информации функцией println() будут добавляться в конце еще два байта – CR («возврат каретки») и LF («перевод строки»), которые позволяют выводить следующую информацию с новой строки. Т.е. эти байты (а точнее, символы с кодами 13 и 10 соответственно) аналогичны нажатию клавиши Enter. В мониторе порта есть соответствующие настройки. Возможно, что при первом соединении двух плат Arduino, программа не будет работать и светодиод не будет управляться. В этом случае проверьте, как вы выводите информацию и как её получаете, зная теперь про символы перехода на начало новой строки.

- Протокол, использующийся при таком соединении двух устройств, называется **UART** (универсальный асинхронный приёмопередатчик)

VIII.Графический экран I2C

Особенности:

- графический экран разрешением 128 x 64 пиксела
- технология OLED, большой угол обзора, не требуется по
- рабочее напряжение 3-5 вольт
- требуется всего два пина для подключения



Подключение: VCC □□ 5V GND □□ GND SDA □□ A4 SCL □□ A5

После подключения любых i2c-устройств необходимо определить их адрес. Это можно сделать с помощью программы i2cScanner (доступна по адресу http://playground.arduino.cc/Main/I2cScanner)

Для работы с экраном используем библиотеку OLED_I2C (автор - Henning Karlsen), с помощью которой можно выводить различными шрифтами текст, целые и дробные числа, а также картинки, линии, прямоугольники, окружности и отдельные пикселы.

Алгоритм вывода на дисплей:

1. Сформировали изображение в буфере дисплея

2. Отобразили сформированное изображение на дисплее Особенность работы контроллера дисплея:

При отображении функцией update() сформированного изображения на

дисплее, содержимое буфера не очищается. Что это означает?

Пример:

myOLED.print(“text 1”, 0, 0); myOLED.update(); delay(2000); myOLED.print(“text 2”, 0, 2); myOLED.update();

Здесь через 2 секунды после вывода надписи “text 1” выведется надпись “text 2”, которая перекроет первую. Т.е. вторая выводимая информация

«суммируется» с изображением, что хранится в буфере.

Для начала вывода информации с «чистого листа» необходимо предварительно использовать функцию clrScr();

Необходимые функции:

1) объявление класса:

OLED myOLED(SDA, SCL);

2) инициализация:

myOLED.begin();

3) отображение сформированного изображения на дисплее: myOLED.update();

4) очистка изображения: myOLED.clrScr();

5) вывод строки:

myOLED.print(строка, x, y);

6) вывод целого числа: myOLED.printNumI(число, x, y);

7) вывод дробного числа: myOLED.printNumF(число, x, y);

8) установка шрифта перед выводом: myOLED.setFont(имя шрифта);

Дополнительные функции:

1) установка пиксела на изображении: myOLED.setPixel(x, y);

2) очистка пиксела на изображении: myOLED.clrPixel(x, y);

3) начертить линию между двумя точками: myOLED.drawLine(x1, y1, x2, y2);

Остальные функции, а также примеры шрифтов, можно посмотреть в файле OLED_I2C.pdf.

4) инверсный (true) или обычный (false) режим цвета текста: myOLED.invertText(режим);

Что нужно знать:

- Микроконтроллеры серии Atmega, используемые в платах типа Arduino UNO, имеют встроенную шину i2c, которая реализована на выводах, соответствующих пинам A4 (SDA – Serial DAta – последовательные данные) и A5 (SCL – Serial CLock – тактирование для синхронизации)

- Интерфейс i2c позволяет одновременно подключать до 127 устройств, для чего каждое устройство должно иметь свой собственный адрес (номер)

- На самом деле устройств будет скорей всего гораздо меньше, т.к. существует ограничение на суммарную ёмкость подключенных входов – 400 пФ. При превышении этой величины будут сильно искажаться сигналы, и обмен данными будет либо осложнен, либо невозможен.

- Устройство, которое инициирует подключение и вырабатывает импульсы SCL, называется ведущим (master)

- Устройство, которое отозвалось ведущему на какой-то адрес и установило с ним связь, называется ведомым (slave)

- Стандартная скорость обмена по шине i2c – 100 кбит/с

- Несмотря на то, что i2c считается шиной для соединения разных устройств в пределах одного корпуса (т.н. внутрисистемная шина), можно обеспечить связь от нескольких метров до нескольких километров (с небольшими вспомогательными дополнениями)

- Существуют универсальные библиотеки для работы с шиной i2c, прежде всего штатная для Arduino IDE – <Wire.h>

- Есть ещё популярные библиотеки для работы с графическими дисплеями. Наиболее популярные: Adafruit SSD1306 + Adafruit GFX, OzOLED и U8glib

IX.Номер кнопки на удалённом дисплее

1. Подключите к первой плате Arduino i2c-дисплей и тактовую кнопку, как это показано в задании VII.

Напишите программу, выводящую в центре дисплея, по нажатию тактовой кнопки, либо номер нажатой кнопки на второй плате (”1”, ”2” или ”3”), либо строковую информацию о ней (”button A”, “button B” или “button C”).

2. Установите на макетную плату со второй платой Arduino три тактовых кнопки и подключите их к цифровым пинам 2, 3 и 4.

Напишите программу, отправляющую через UART-соединение номер нажатой кнопки.

При соединении двух плат Arduino на дисплее должна отображаться информация о нажатой кнопке. При запуске программы отображается номер кнопки.

X.Состояние удалённых кнопок на дисплее

Используйте собранные в задании IX схемы.

Напишите программы для двух соединяемых плат Arduino:

1 плата: на дисплее должна отображаться информация о состоянии всех трёх кнопок на макетной плате в формате

button A: ON button B: OFF button C: OFF

где ON соответствует нажатому состоянию, а OFF – не нажатому

2 плата: по UART должна последовательно передаваться информация о состоянии всех трёх кнопок на макетной плате (0 – не нажата, 1 – нажата).

XI. RTC - Часы реального времени

Модуль DS3231 (RTC, ZS-042) — представляет собой недорогую плату с чрезвычайно точными часами реального времени (RTC), с температурной компенсацией кварцевого генератора и кристалла. Модуль включает в себя литиевую батарею, которая поддерживает бесперебойную работу, даже при отключении источник питания.

Технические параметры:

Напряжение питания: 3.3В и 5В Точность: ± 0.432 сек в день Поддерживаемый протокол: I2C

Контроллер DS3231 поддерживает секунды, минуты, часы, день недели, число, месяц и год, а так же следит за количеством дней в месяце и делает поправку на високосный год. Поддерживает работу часов в форматах 24 и 12 часов, а также даёт возможность запрограммировать два будильника.

Модуль работает по стандартной шине I2C. Подключение совершенно аналогично рассмотренному выше OLED-дисплею: VCC 5V

GND □□ GND SDA □□ A4 SCL □□ A5

Таким образом, и OLED-дисплей, и RTC-модуль подключаются параллельно, не требуя выделения дополнительных пинов – это одно из преимуществ I2C шины.

Библиотека для работы с модулем - iarduino_RTC, отличающаяся от других прежде всего универсальностью (позволяет подключать модули на различных чипах).

Функции для считывания параметров из RTC:

1. Функция gettime("строка с параметрами") - функция получает и выводит

строку, заменяя описанные ниже символы на текущее время.

пример: time.gettime("d-m-Y, H:i:s, D"); вернёт строку "01-10-2015, 14:00:05, Thu"
пример: time.gettime("s"); вернёт строку "05"

s - секунды от 00 до 59 (два знака) i - минуты от 00 до 59 (два знака)

h - часы в 12-часовом формате от 01 до 12 (два знака) H - часы в 24-часовом формате от 00 до 23 (два знака) d - день месяца от 01 до 31 (два знака)

w - день недели от 0 до 6 (один знак: 0-воскресенье, 6-суббота)

D - день недели наименование от Mon до Sun (три знака: Mon Tue Wed Thu Fri Sat Sun)

m - месяц от 01 до 12 (два знака)

M - месяц наименование от Jan до Dec (три знака: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec)

Y - год от 2000 до 2099 (четыре знака) y - год от 00 до 99 (два знака)

a - время суток am или pm (два знака, в нижнем регистре) A - время суток AM или PM (два знака, в верхнем регистре)

2. Если требуется получить время в виде цифр, то можно вызвать функцию gettime() без параметра, после чего получить время из переменных

пример: time.gettime();

Serial.print(time.Hours); Serial.print(":"); // Вывод часов Serial.print(time.minutes); Serial.print(":"); // Вывод минут Serial.print(time.seconds); Serial.println(""); // Вывод секунд

seconds	- секунды	0-59
minutes	- минуты	0-59
hours	- часы	1-12
Hours	- часы	0-23
midday	- полдень	0-1 (0-am, 1-pm)

day - день месяца 1-31

weekday - день недели 0-6 (0-воскресенье, 6-суббота) month - месяц 1-12

year - год 0-99

Функция установки времени:

settime(секунды [, минуты [, часы [, день [, месяц [, год [, день недели]]]]]]): записывает время в модуль

год указывается без учёта века, в формате 0-99

часы указываются в 24-часовом формате, от 0 до 23

день недели указывается в виде числа: 0-воскресенье, 1-понедельник, 2- вторник ..., 6-суббота

если предыдущий параметр надо оставить без изменений, то можно указать отрицательное или заведомо большее значение

пример: time.settime(-1, 10);

установит 10 минут, а секунды, часы и дату, оставит без изменений

пример: time.settime(0, 5, 13);

установит 13 часов, 5 минут, 0 секунд, а дату оставит без изменений

пример: time.settime(-1, -1, -1, 9, 2, 17);

установит дату 09.02.2017 , а время и день недели оставит без изменений

пример: time.settime(0,51,21, 27, 10, 15, 2);

установит 0 сек, 51 мин, 21 час, 27, октября, 2015 года, вторник

XII. Часы

Подключите к макетной плате OLED-дисплей как показано в задании VIII, и модуль часов реального времени. Составьте программу реализующие часы в соответствии с изображением:

Полезная информация: для выравнивания текста по левому краю, или по правому краю, или по центру вместо координаты x в функциях библиотеки OLED_I2C можно использовать служебные слова LEFT, RIGHT и CENTER.

Например:

myOLED.print(“Hello!”, CENTER, 20);