# Отчёт по теме “Система автоматизированного управления и регулирования системы освещения”

# Подбор элементной базы

Для создания системы автоматизированного управления и регулирования системы освещения, были подобраны следующие компоненты: Одноплатный компьютер Raspberry PI, силовой блок управления noolite SU-1-500, пульт радиопередатчик noolite МТ1132.

Лампы в системе, используемые в проекте, имеют максимальную мощность 220Вт. Исходя из того, что диммеры необходимо подбирать с запасом мощности +30%, то нам необходимы диммеры с регулируемой мощностью от 0 до 286Вт (верхний предел может быть и больше).



Рисунок 1. Блок питания лампы.

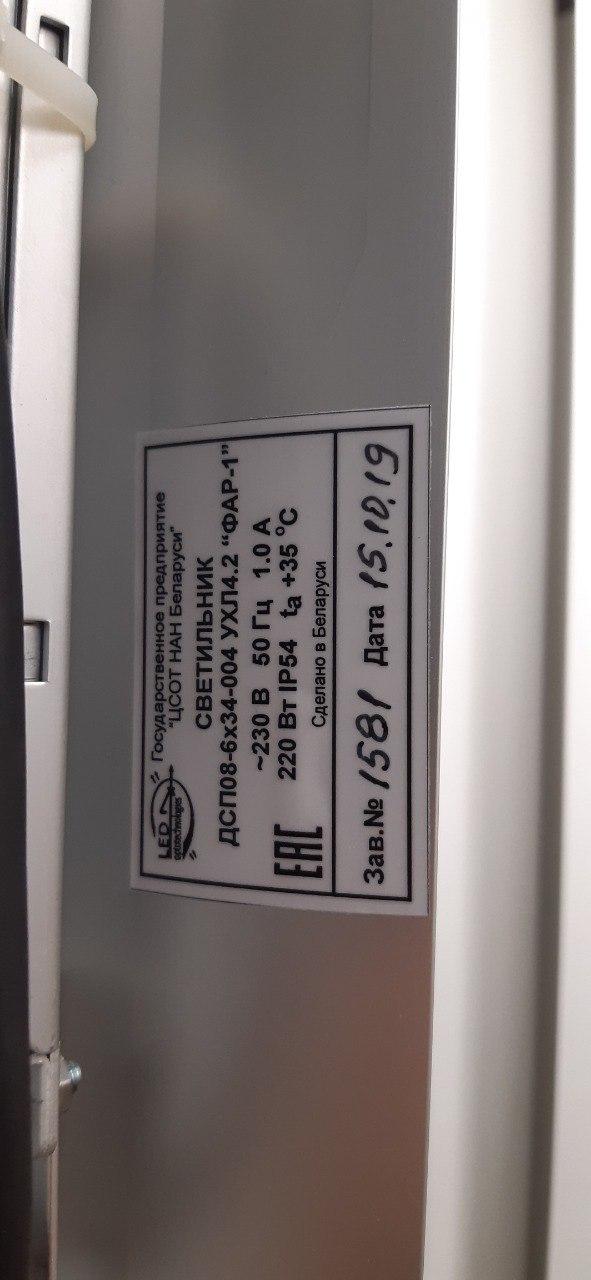
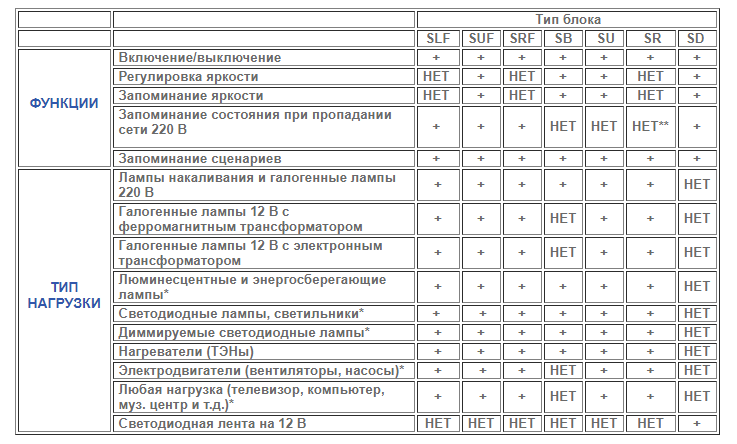


Рисунок 2. Входные параметры для питания светильника.

 Рисунок 3. Типы силовых блоков noolite.

Из рисунка 3 следует что нам необходимы силовые блоки типа SU и SUF. Силовой блок серии SUF представлен только в одной модели это SUF-1-300 датчики этой серии умеют запоминать своё последнее положение при отключении от электричества. Блоки серии SU представлены в нескольких моделях, однако нас интересуют только модели от 300Вт ближайшая модель блока типа SU это SU-1-500 который выдерживает нагрузки до 500Вт.

Система управления делится на две части: пульт радиопередатчик и выключатель радиоуправляемый (силовой блок), он же диммер.

# Схемы подключения элементов системы

На рисунке 4 представлена схема подключения пульта к одноплатному компьютеру Raspberry PI, который используется для управления пультом передатчиком.

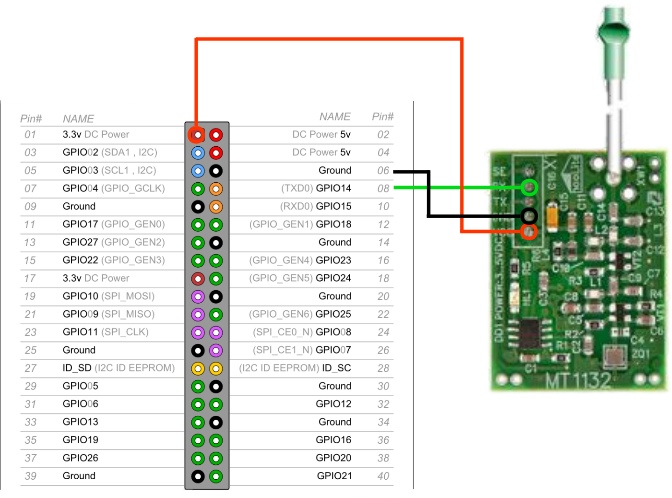


Рисунок 4. Схема подключения пульта noolite к RPI.

На рисунке 5 представлена схема подключения силового блока SU-1-200/300.

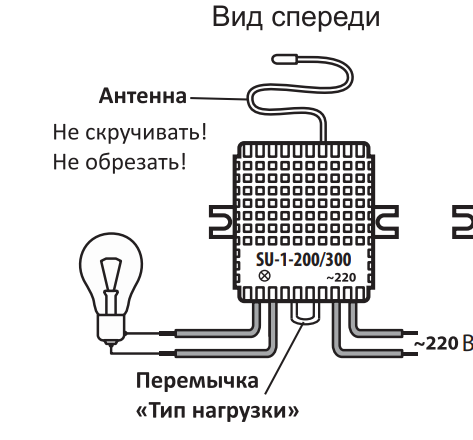


Рисунок 5. Силовой блок noolite.

# Программная часть

Подача сигналов для пульта управления noolite осуществляется при помощи скрипта на языке программирования Python, который представлен в приложении 1. Скрипт принимает 2 входных параметра: номер канала связи пульта с силовым блоком и команду. Поскольку модуль может отправлять команды на 32 силовых блока, он обязательно должен получить номер канала (0-31). Во втором параметре указываются числа от 0 до 15 и 35-155. Команды от 0 до 10 и 15 выполняют следующие функции:

0 - выключить нагрузку

1 - запустить плавное понижение яркости

2 - включить нагрузку

3 - запустить плавное повышение яркости

4 - включить или выключить нагрузку

5 - запустить плавное изменение яркости в обратном направлении

6 - установить заданную в «Байт данных 0» яркость

7 - вызвать записанный сценарий

8 - записать сценарий

9 - отвязка

10 - остановить регулировку

15 - привязка к силовому блоку

Команды от 35 до 155 являются значением яркости, передаваемым на силовой блок.

На рисунке 6 представлена структура команды для передачи сигнала силовому блоку. Данная команда несёт в себе 12 байт информации.

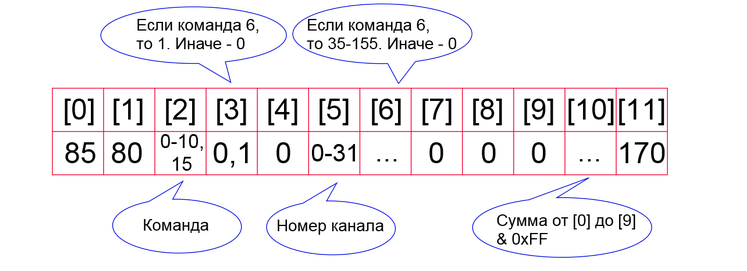


Рисунок 6. Структура команды для передачи сигнала силовому блоку.

Поскольку модуль может отправлять команды на 32 силовых блока, он обязательно должен получить номер канала (0-31). Это элемент [5].

На силовой блок в элементе [2] будет отправлена одна из возможных команд.

Если это команда 6 (установить яркость), то в элемент [3] нужно записать единицу, а в элемент [6] - значение яркости. Значение это ни в процентах, ни в привычных байтах, а просто, значение из ряда 35 - 155.

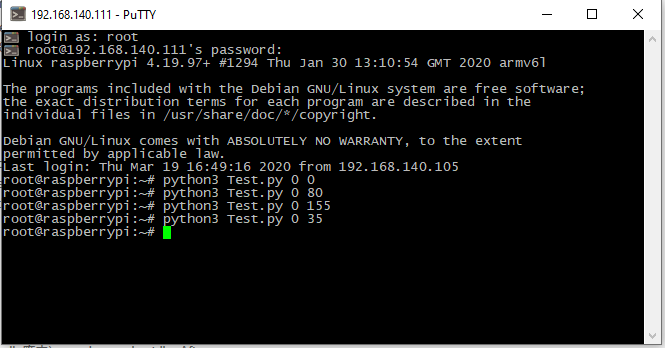


Рисунок 7. Пример выполнения скрипта управления пультом.

# Приложение 1.

import serial

import sys

ser = serial.Serial("/dev/ttyAMA0", baudrate=9600)

#print(sys.argv[1])

command = [85,80,0,0,0,0,0,0,0,0,0,170]

command[5] = int(sys.argv[1])

tempcommand = int(sys.argv[2])

#print(command)

if (tempcommand < 16):

command[2] = tempcommand;

command[3] = 0;

command[6] = 0;

else :

command[2] = 6;

command[3] = 1;

command[6] = tempcommand;

check = 0

for i in range(0, 10):

check += command[i]

check = check & 0xFF

command[10] = check

#print(command)

tmpBuffer = bytearray(command)

ser.write(tmpBuffer)

ser.close()