

Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных средств

Дисциплина: Микропроцессорные средства и системы

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту
на тему

УСТРОЙСТВО УПРАВЛЕНИЯ ПОЛИВОМ

БГУИР КП 1-40 02 02 023 ПЗ

Студент: гр. 850701 Филиппов Д. А.

Руководитель: Порхун М.И.

Минск 2021

СОДЕРЖАНИЕ

Введение. постановка задачи	3
1 Анализ задачи. Функциональная спецификация системы.....	4
2 Предварительное проектирование системы	6
2.1 Разбиение системы на модули	6
2.2 Построение структурной схемы аппаратной части системы	7
2.3 Описание структурной схемы	7
3 Проектирование аппаратных средств системы.....	8
3.1 Выбор типа микроконтроллера	8
3.2 Разработка принципиальной схемы системы	10
3.2.1 Управление светодиодами с помощью ATmega328p.....	10
3.2.2 Управление кнопками с помощью ATmega328p.....	11
3.2.3 Управление пьезодинамиком с помощью ATmega328p.....	12
3.2.4 Подключение насоса к ATmega328p	13
3.2.5 Подключение модуля реле к ATmega328p.....	14
3.2.6 Подключение дисплея к ATmega328p через драйвер TM1637	15
3.3 Описание работы системы по принципиальной схеме	16
4 Проектирование программного обеспечения.....	17
4.1 Разработка схемы алгоритма работы системы и программы	17
4.2 Описание алгоритма работы системы и программы	17
5 Моделирование работы системы.....	20
5.1 Выбор системы моделирования	20
5.2 Описание процесса моделирования	20
Заключение	23
Список использованных источников	24
ПРИЛОЖЕНИЕ А (Обязательное) Схема электрическая структурная	25
ПРИЛОЖЕНИЕ Б (Обязательное) Схема электрическая принципиальная....	27
ПРИЛОЖЕНИЕ В (Обязательное) Блок-схема алгоритма	31
ПРИЛОЖЕНИЕ Г (Обязательное) Код программы	33

ВВЕДЕНИЕ. ПОСТАНОВКА ЗАДАЧИ

При разведении растений в домашних либо тепличных условиях требуется осуществлять регулярный полив для более равномерного их развития и защиты от пересыхания. Вручную заниматься этим довольно трудно, особенно при достаточно большом количестве рассады. Для этого зачастую приходится тратить силы и время ежедневно. Так же такая обязанность может лишить человека возможности отлучиться на несколько дней. У некоторых растений фазы полива могут приходиться и на ночное время суток, когда люди обычно спят, либо на полуденное, когда большинство находится на рабочем месте, вне дома. Определение подходящего объёма жидкости и её равномерное распределение между растениями на глаз – также довольно непростая задача.

Устройство управления поливом может автоматизировать все эти задачи и избавить человека от необходимости уделять им столько внимания и усилий.

Целью данного курсового проекта является разработка устройства управления поливом на базе микроконтроллера.

Устройство должно выполнять следующие задачи:

- обеспечивать автоматическое включение и выключение полива в заданное время суток;
- обеспечивать установку времени включения и выключения полива путем выбора из фиксированного (8 значений) набора значений;
- обеспечивать задание времени включения и выключения полива с помощью устройства с кнопками (переключателем) и светодиодными индикаторами;
- обеспечивать подачу звуковой и световой сигнализации при включении полива;
- быть несложным в управлении и максимально удобным в использовании.

Для решения данной задачи использовались следующие программные среды разработки и проектирования: Proteus и Arduino IDE. В качестве микроконтроллера был выбран ATmega328p.

1 АНАЛИЗ ЗАДАЧИ. ФУНКЦИОНАЛЬНАЯ СПЕЦИФИКАЦИЯ СИСТЕМЫ

В данном курсовом проекте необходимо разработать устройство управления поливом. В данной реализации это устройство содержит мотор постоянного тока насоса, подающего воду по системе трубок к растениям из общего резервуара.

Данное устройство устанавливает 8 режимов работы, а точнее разное время суток включения полива:

- 1:00;
- 4:00;
- 7:00;
- 10:00;
- 13:00;
- 16:00;
- 19:00;
- 22:00.

При нажатии кнопки переключения режима должен загораться соответствующий светодиод и устанавливаться определённый режим полива.

В месте, в котором необходимо производить автополив (место расположения комнатных растений в помещении или теплица) устанавливается система трубок, подключаемая к насосу.

Сам насос подключается к микроконтроллеру и питанию по определённой схеме, а также к ёмкости с водой, достаточной для забора воды на более-менее длительный промежуток времени (по меньшей мере, на несколько дней).

При подключении устройства к питанию пользователю необходимо произвести первичную настройку. Путём нажатия на кнопки выставить реальное на данный момент время в часах и минутах и выбора режима полива. Режим полива можно менять и впоследствии, уже во время эксплуатации.

Для отображения того, что устройство подключено и настроено, используется светодиодная индикация зелёного цвета свечения, обозначающая так же и выбранный режим. Также устройство выводит на 7-сегментный дисплей реальное время, выставленное на нём при первичной настройке, что позволяет контролировать верность работы.

Непосредственно во время полива для удобства осуществляется световая (красный светодиод) и звуковая (пьезодинамик) индикации.

Далее представлена функциональная спецификация устройства.

Таблица 1.1– Функциональная спецификация устройства

Входы	Выходы	Функция
Кнопка	Зажигание соответствующего светодиода	Включение заданного режима, в зависимости от выбора времени полива
Кнопки	Переключение значений на дисплее	Установка реального времени на устройстве
	Пьезодинамик и светодиод	Включает звуковую и световую индикацию полива

В общем случае могут существовать различные способы взаимодействия между пользователем и системой

Тактильные (контактные) входы. Пользователь может нажимать на кнопки.

Визуальные выходы. Светодиоды и дисплей являются примером визуального выхода.

Звуковые выходы. Звуковая сигнализация является примером звукового выхода.

2 ПРЕДВАРИТЕЛЬНОЕ ПРОЕКТИРОВАНИЕ СИСТЕМЫ

2.1 Разбиение системы на модули

Основой аппаратных средств является микро-ЭВМ. Список модулей включает:

- процессорный модуль, который используется для обработки информации;
- модуль ГТИ (генератор тактовых импульсов), используемый для синхронизации системы;
- модуль памяти, используемый для хранения программы и данных;
- модуль интерфейса ввода и интерфейса вывода, которые необходимы для связи процессорного модуля с другими модулями системы;
- модуль преобразования входного сигнала и модуль преобразования выходного сигнала, которые содержат компоненты, необходимые для обмена входными и выходными сигналами с внешним окружением.

Общая модульная структура аппаратных средств системы представлена на рисунке 2.1.

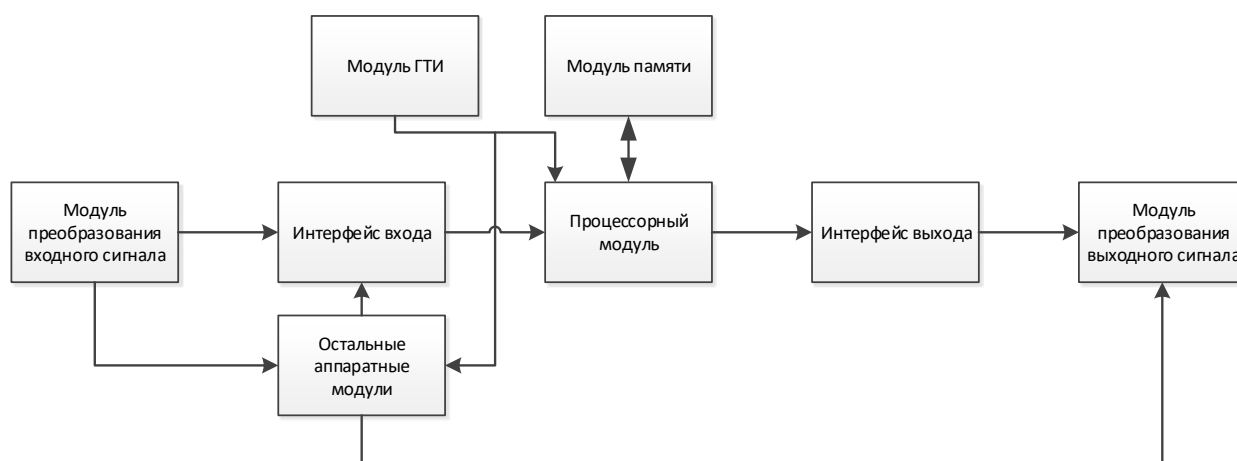


Рисунок 2.1 – Общая модульная структура устройства

Далее разобьём систему устройства управления поливом на функциональные модули. Исходя из функциональной спецификации, делаем вывод, что система состоит из трех частей: вход, выход, функции.

Входной модуль выполняет считывание нажатий с кнопок.

Выходной модуль включает светодиоды, соответствующие выбранному режиму, отображает время на дисплее, запускает полив и индикацию.

Функциональный модуль включает в себя следующие модули:

- модуль управления текущим режимом работы;
- модуль установки реального времени;

- модуль управления дисплеем;
- модуль управления мотором насоса и индикацией полива.

Распределение функций по модулям устройства управления поливом выглядит следующим образом:

- 1) входной модуль:
 - приём сигналов с кнопок;
- 2) выходной модуль:
 - включение звуковой и световой сигнализации полива;
 - зажигание светодиода, соответствующего режиму работы;
 - включение насоса;
- 3) модуль управления текущим режимом работы:
 - изменяет режим полива;
- 4) модуль установки реального времени:
 - настройка времени путём нажатия кнопок.

2.2 Построение структурной схемы аппаратной части системы

На основе выполняемых системой функций, была построена структурная схема аппаратной части.

Электрическая структурная схема проектируемой системы представлена на чертеже ГУИР.271632.001 Э1 в приложении А.

2.3 Описание структурной схемы

Центральным модулем структурной схемы является процессорный модуль (микроконтроллер). Он выполняет функции управления процессом обмена данными с периферийными устройствами и обработку поступающей в него информации. В модуле памяти хранятся коды, константы и переменные программ процессорного модуля. В отдельный блок можно выделить модуль генератора тактовых импульсов (ГТИ). Входной и выходной модули необходимы для координирования операций ввода-вывода информации. Они будут представлены портами микропроцессорного устройства. В нашем случае периферийными устройствами будут служить: кнопки, светодиодные индикаторы, 7-ми сегментный 4-ёх разрядный дисплей, мотор и средство звуковой сигнализации (пьезодинамик). Они и будут подключены к входному и выходному модулям.

3 ПРОЕКТИРОВАНИЕ АППАРАТНЫХ СРЕДСТВ СИСТЕМЫ

3.1 Выбор типа микроконтроллера

Управление и контроль климатом будет производиться микроконтроллером ATmega328P, входящим в состав платформы Arduino Uno рисунок 3.1 [3]. Микроконтроллер ATmega328P является восьми разрядным CMOS микроконтроллером с низким энергопотреблением, основанным на усовершенствованной AVR RISC архитектуре. Он предоставляет 32 КБ флеш-памяти для хранения прошивки, 2 КБ оперативной памяти SRAM и 1 КБ энергонезависимой памяти EEPROM для хранения данных.

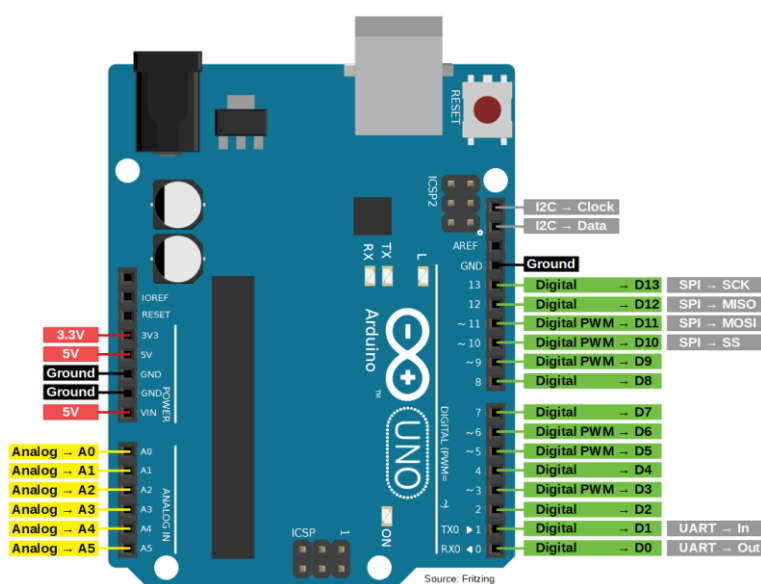


Рисунок 3.1 – Arduino Uno

ATmega328P применяется для создания электронных устройств с возможностью приема сигналов от различных цифровых и аналоговых датчиков, которые могут быть подключены к нему, и управления различными исполнительными устройствами. Проекты устройств, основанные на ATmega, могут работать самостоятельно или взаимодействовать с программным обеспечением на компьютере.

Преимущества ATmega:

- 1 Микроконтроллеры ATmega относительно дешевы по сравнению с другими платформами. Самая недорогая версия модуля Arduino может быть собрана вручную, а некоторые даже готовые модули стоят меньше 50 долларов.

- 2 Кроссплатформенность – программное обеспечение Arduino работает под ОС Windows, Macintosh OSX и Linux. Большинство микроконтроллеров ограничивается ОС Windows.
- 3 Простая и понятная среда программирования – среда Arduino подходит как для начинающих пользователей, так и для опытных.
- 4 Примеры исходного кода. Еще одним большим преимуществом Arduino является библиотека примеров, идущая в комплекте «из коробки». То, чего нет в поставке, легко ищется в интернете, все библиотеки общедоступны, вам не потребуется много кодировать.
- 5 Аппаратные средства с возможностью расширения и открытыми принципиальными схемами – микроконтроллеры ATmega328P и Arduino. Схемы модулей выпускаются с лицензией Creative Commons, а значит, опытные инженеры имеют возможность создания собственных версий модулей, расширяя и дополняя их. Даже обычные пользователи могут разработать опытные образцы с целью экономии средств и понимания работы.

Arduino Uno - это микрокомпьютер на основе контроллера ATmega328P. В его состав входит все необходимое для работы с микроконтроллером: 14 цифровых портов входа-выхода (шесть из них поддерживают режим ШИМ модуляции), шесть аналоговых входов, кварцевый резонатор на 16 МГц, разъем USB, разъем питания, разъем внутрисхемного программирования и кнопка сброса. Для начала работы с устройством достаточно просто подать питание от блока питания или батарейки, либо подключить его к компьютеру с помощью USB-кабеля. Arduino Uno совместим с большинством плат расширения, разработанных для Arduino Duemilanove.

Характеристики:

- Микроконтроллер: ATmega328;
- Тактовая частота: 16 МГц;
- Напряжение логических уровней: 5 В;
- Входное напряжение питания: 5–12 В;
- Портов ввода-вывода общего назначения: 20;
- Максимальный ток с порта ввода-вывода: 40 мА;
- Максимальный выходной ток порта 3.3 В: 50 мА;
- Максимальный выходной ток порта 5 В: 800 мА;
- Портов с поддержкой ШИМ: 6;
- Портов, подключённых к АЦП: 6;
- Разрядность АЦП: 10 бит;
- Flash-память: 32 КБ;

- EEPROM-память: 1 КБ;
- Оперативная память: 2 КБ;
- Габариты (размер Arduino Uno): 69×53 мм.

3.2 Разработка принципиальной схемы системы

Для проектируемой системы понадобится: дисплей с драйвером, светодиоды, пьезодинамик, кнопки, реле, транзистор.

Для иллюстрации работы устройства будем использовать Proteus. Прошивку для схемы будем создавать в программе Arduino IDE.

3.2.1 Управление светодиодами с помощью ATmega328p

Светодиоды применяются для световой сигнализации, а также для указания включённого режима. Для данного устройства подойдут стандартные светодиоды. В схеме последовательно со светодиодом используется подтягивающий резистор. Резисторы для светодиодов нужно рассчитывать по формуле $R = (U_{\text{пит}} - U_{\text{пад}}) / I_{\text{ном}}$, где $U_{\text{пит}}$ — напряжение питания, $U_{\text{пад}}$ — падение напряжения на светодиоде (у каждого цвета разное). Схема управления светодиодом изображена на рисунке 3.3.



Рисунок 3.2 – Светодиод

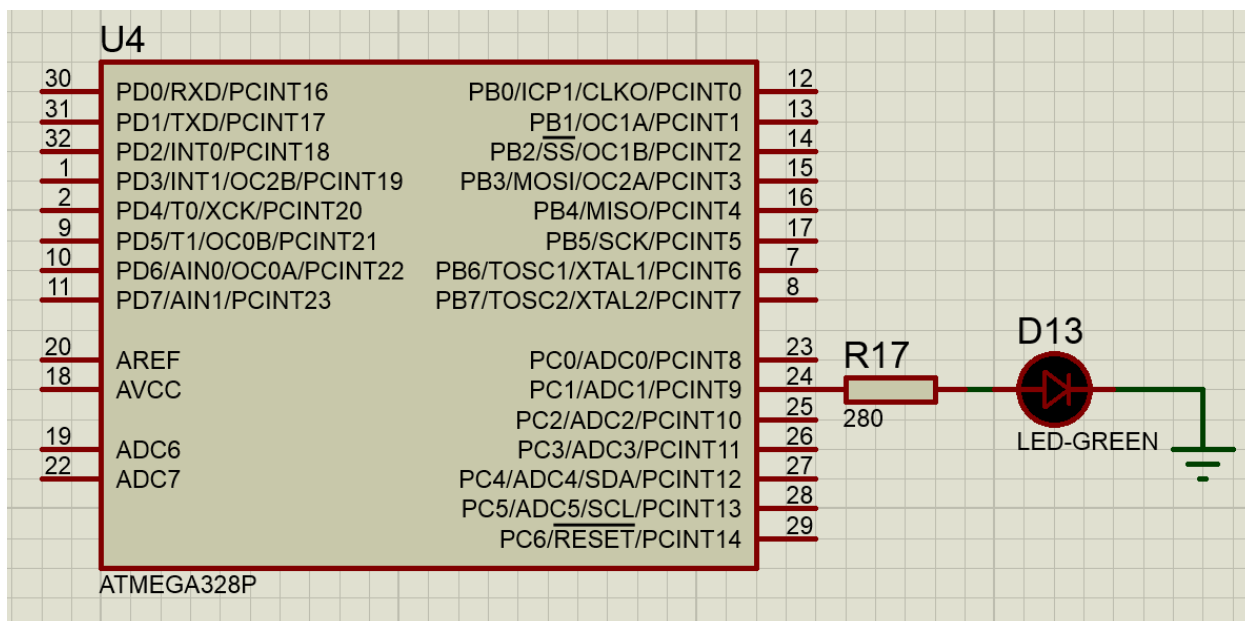


Рисунок 3.3 – Подключение светодиода к АТmega328р

3.2.2 Управление кнопками с помощью АТmega328р

Переключатели применяются для переключения режимов. Для данного устройства подойдут стандартные переключатели. Последовательно подключаем резистор сопротивлением 10 кОм. Схеме подключения, которая изображена на рисунке 3.5.

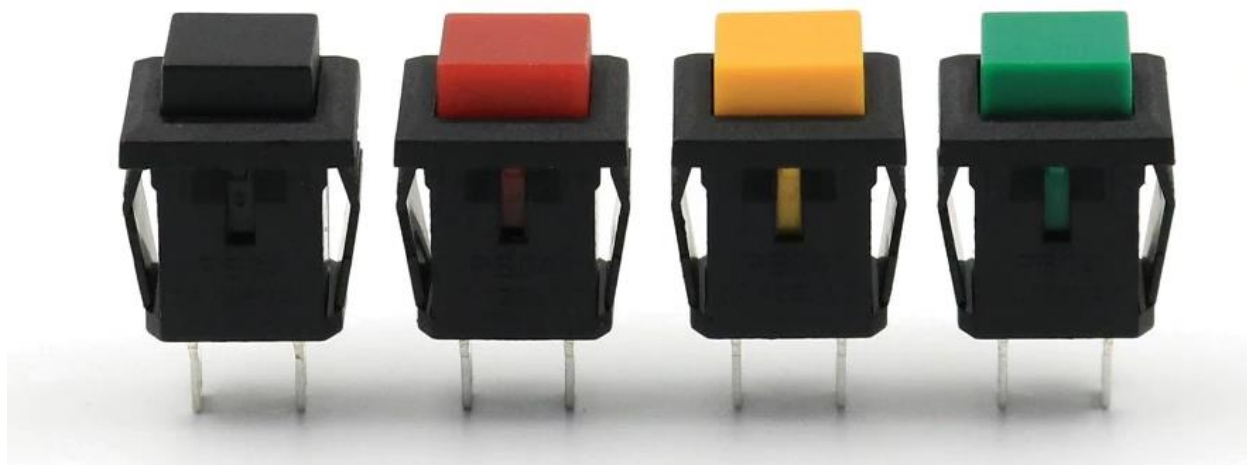


Рисунок 3.4 – Кнопки двухконтактные

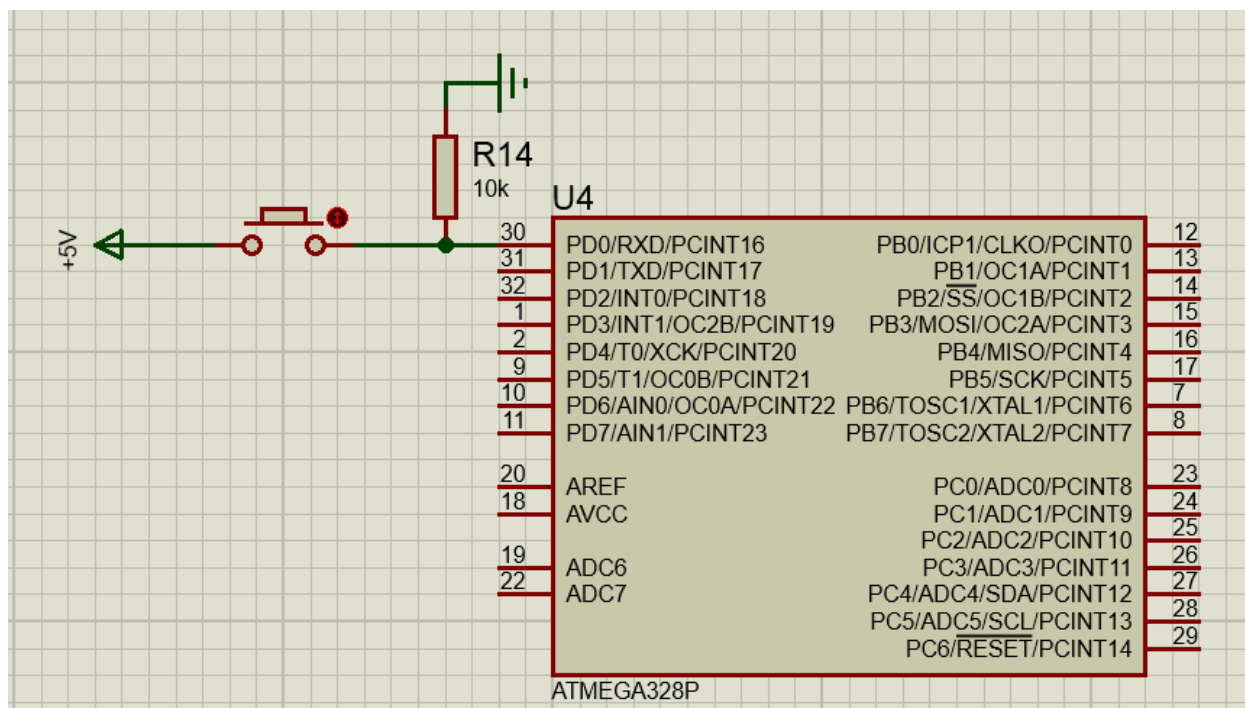


Рисунок 3.5 – Подключение переключателя к АТмега328р

3.2.3 Управление пьезодинамиком с помощью АТмега328р

Пьезоэлемент — электромеханический преобразователь, одной из разновидностей которого является пьезодинамик. Пьезодинамик переводит электрическое напряжение в колебание мембраны. Эти колебания и создают звук (звуковую волну). Пьезодинамик может подключаться к микроконтроллеру напрямую. Схеме подключения, которая изображена на рисунке 3.8.



Рисунок 3.7 – Пьезодинамик

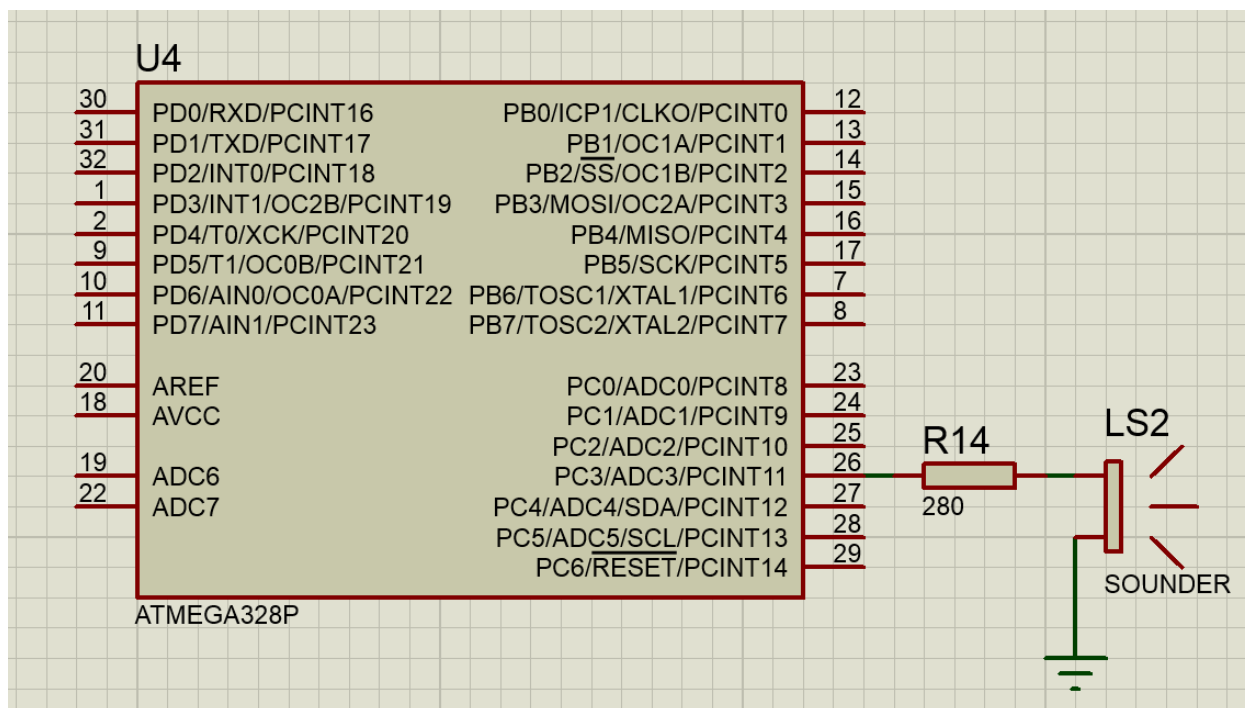


Рисунок 3.8 – Подключение пьезодинамика к АТmega328р

3.2.4 Подключение насоса к АТmega328р

Последовательно к моторчику подключаем резистор сопротивлением 10 кОм. Схема подключения геркона изображена на рисунке 3.10.



Рисунок 3.9 – Насос

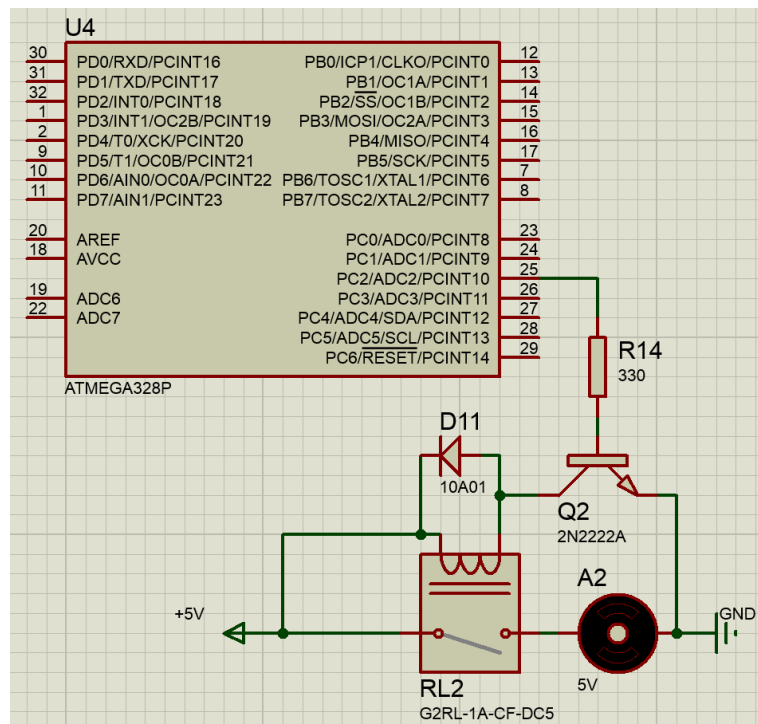


Рисунок 3.10 – Подключение насоса к АТmega328р

3.2.5 Подключение модуля реле к АТmega328р

В качестве нагревательного элемента используется водонагреватель. Его подключение к микроконтроллеру осуществляется через модуль реле.

Электромагнитное реле – это электрическое устройство, которое механическим путем замыкает или размыкает цепь нагрузки при помощи магнита. состоит из электромагнита, подвижного якоря и переключателя.

Модуль реле имеет всего три контакта, подключаются они к Ардуино Uno следующим образом: GND – GND, VCC – +5V, In – 3. Схема подключения модуль реле представлена на рисунке 3.12 [1].



Рисунок 3.11 – Модуль реле

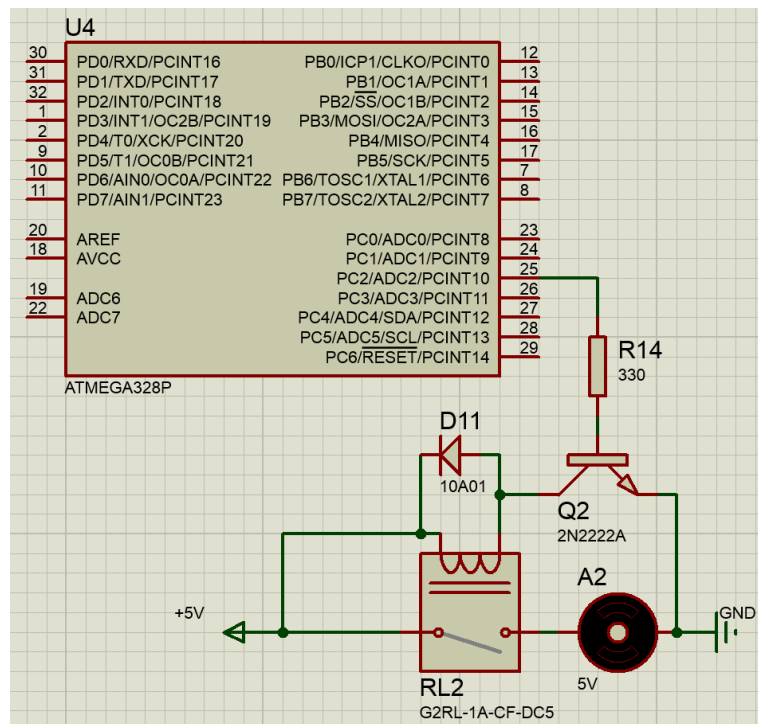


Рисунок 3.12 – Подключение модуля реле к АТmega328р

3.2.6 Подключение дисплея к АТmega328р через драйвер ТМ1637

ТМ1637 – это драйвер 7-ми сегментного 4-ёх разрядного дисплея с разделительным двоеточием. Он помогает подключить данный дисплей при помощи всего двух цифровых пинов.

Схема подключения модуль датчика температуры представлена на рисунке 3.14 [2].



Рисунок 3.13 – Модуль дисплея ТМ1637

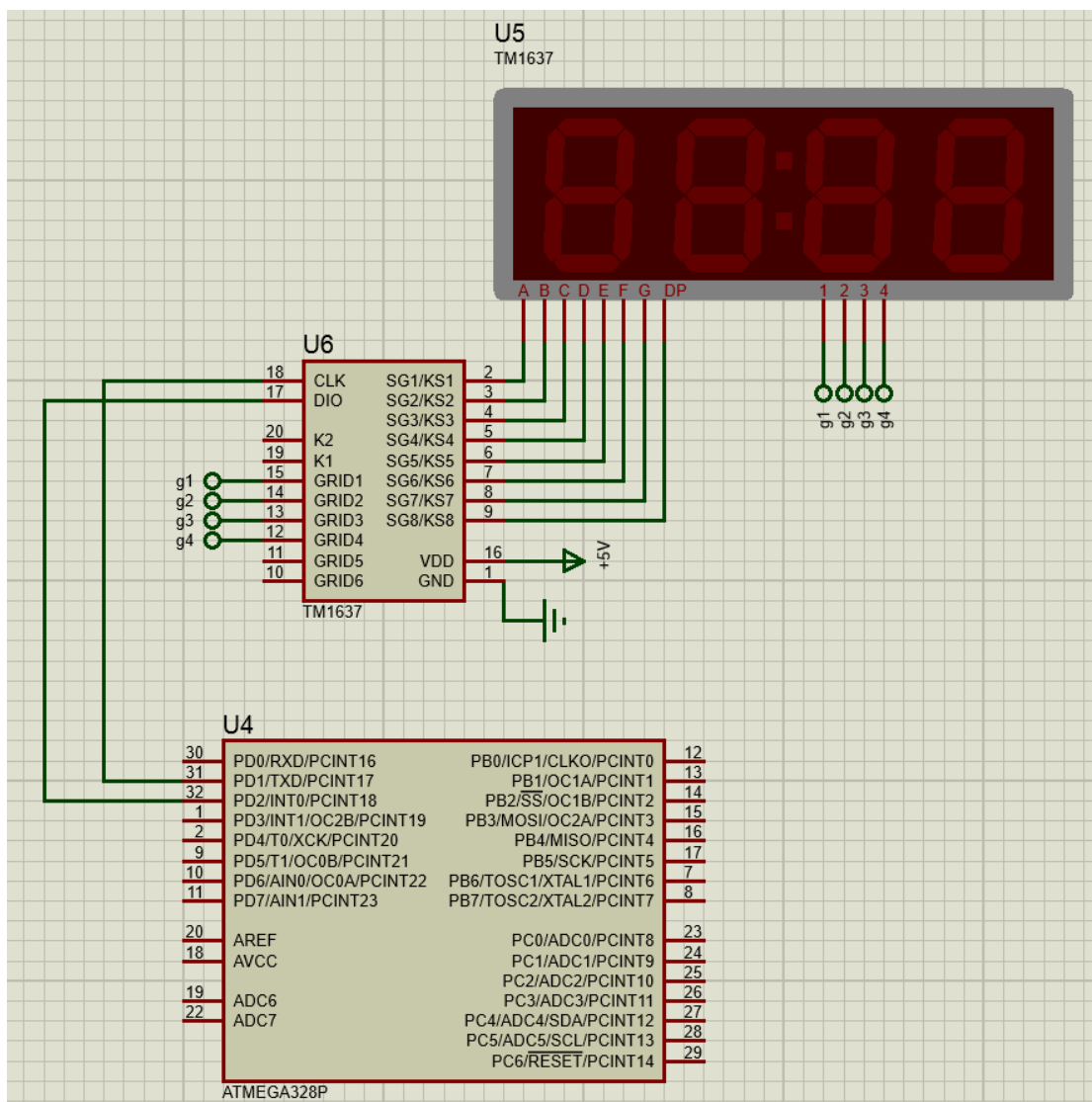


Рисунок 3.13 – Подключение DS18B20 к Arduino Uno

Для работы с данным датчиком используется библиотека TM1637.

3.3 Описание работы системы по принципиальной схеме

Схема электрическая принципиальная устройства управления поливом представлена на чертеже ГУИР.271632.003 ЭЗ. Перечень элементов схемы электрической принципиальной приведён на чертеже ГУИР.271632.003 ПЭЗ.

4 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

4.1 Разработка схемы алгоритма работы системы и программы

Подробная блок-схема алгоритма программного обеспечения устройства, включающая в себя все аспекты периферийных устройств и микроконтроллера приведена на чертеже ГУИР.271632.004 ПД приложения В. Код программы приведен в приложении Г.

4.2 Описание алгоритма работы системы и программы

Программа была написана при помощи среды Arduino IDE. Окно интерфейса используемого программного обеспечения представлено на рисунке 4.1.

Arduino IDE использует синтаксис языка «C++», а поддерживает все его синтаксические конструкции [4]. Объяснение некоторых функций:

- `void setup () {...}` – функция, которая выполняется 1 раз сразу после включения Arduino в работу;
- `void loop () {...}` – функция, которая выполняется бесконечно, в течении всей работы Arduino;
- `void light (int number){...}` – функция, зажигающая светодиод определённого номера;
- `void display_time (time_t t) {...}` – функция, отображающая время на дисплее;
- `void init_time (void) {...}` – функция инициализации реального времени при запуске микроконтроллера;
- `pinMode (pin, mode)` – функция настройки вывода Arduino на вход (INPUT) или выход (OUTPUT);
- `tone (pin, frequency)` – создание частоты на выводе Arduino;
- `delay (ms)` – Останавливает выполнение программы на заданное в параметре количество миллисекунд. ms: количество миллисекунд, на которое приостанавливается выполнение программы.
- `millis ()` – Возвращает количество миллисекунд с момента начала выполнения текущей программы на плате Arduino. Это количество сбрасывается на ноль, в следствие переполнения значения, приблизительно через 50 дней.
- `digitalWrite (pin, value)` – Подает HIGH или LOW значение на цифровой вход/выход (pin);

– digitalRead (pin)- Функция считывает значение с заданного входа - HIGH или LOW.

Также использовались дополнительные библиотеки TimeLib.h (для подключения встроенного счётчика реального времени [6]) и TM1637.h (для работы с одноимённым дисплеем).



```
File Edit Sketch Tools Help
curs_project

#include <TimeLib.h>
#include <TM1637.h>

int CLK = 1;
int DIO = 2;
TM1637 tm(CLK, DIO);

int work_hours [8] = {1, 4, 7, 10, 13, 16, 19, 22}; // рабочие часы полива в зависимости от выбранного режима

int modenumber = 0;
int ledpins [8] = {8, 9, 10, 11, 12, 13, 14, 15}; // номера пинов для зелёных светодиодов

#define MODE_BUTTON 0
#define TIME_BUTTON 1
#define SETUP_BUTTON 2
int buttonpins[3] = {0, 3, 4}; // номера пинов для кнопок MODE, TIME, SETUP
// предыдущие состояния для кнопок
int buttonstates[3] = {LOW, LOW, LOW};

int pumppin = 16; // номер пина для водяного насоса
int soundpin = 17; // номер пина для звукового сигнала
int redpin = 18; // номер пина для световой индикации полива

void setup() {
  // код программы, выполняющийся один раз при запуске
  int i;

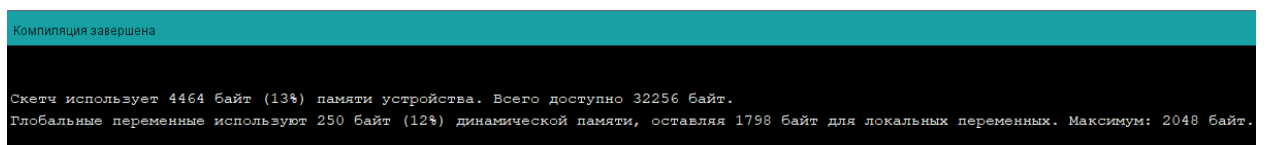
  // инициализация пинов
  pinMode(pumppin, OUTPUT);
  pinMode(soundpin, OUTPUT);
  pinMode(redpin, OUTPUT);

  tm.init();
  // set brightness 0-7
  tm.set(7);

  for (i = 0; i < 8; i++) {
    pinMode(ledpins[i], OUTPUT);
  }
  for (i = 0; i < 3; i++)
```

Рисунок 4.1 – Окно интерфейса среды разработки Arduino IDE

Аппаратные затраты микроконтроллера представлены на рисунке 4.2.



```
Компиляция завершена

Скетч использует 4464 байт (13%) памяти устройства. Всего доступно 32256 байт.
Глобальные переменные используют 250 байт (12%) динамической памяти, оставляя 1798 байт для локальных переменных. Максимум: 2048 байт.
```

Рисунок 4.2 – Использованные ресурсы Arduino Uno

5 МОДЕЛИРОВАНИЕ РАБОТЫ СИСТЕМЫ

5.1 Выбор системы моделирования

Моделирование системы устройства управления климатом в помещении осуществляется в системе автоматизированного проектирования Proteus.

Proteus – это универсальная программа, с помощью которой можно создавать различные виртуальные электронные устройства и выполнять их симуляцию [9]. Она содержит огромную библиотеку аналоговых и цифровых микросхем, датчиков, дискретных элементов: резисторов, конденсаторов, диодов, транзисторов и т.п. Также имеется широкий набор компонентов оптоэлектроники: дисплеи, светодиоды, оптопары и др.

Главным преимуществом и отличием Протеус от других подобных программ для симуляции работы электрических цепей, — это возможность выполнять симуляцию работы микропроцессоров и микроконтроллеров (МК). Библиотека Proteus содержит такие основные типы МК: AVR, ARM, PIC, Cortex.

Как и в любом другом аналогичном софте, предназначенном для симуляции работы электрических цепей, данный софт имеет ряд виртуальных измерительных приборов: амперметры, вольтметры, ваттметр, осциллограф, логический анализатор, счетчик и т.п. Также в Протеусе встроены инструменты для автоматизированной разработки печатных плат и для создания их 3D моделей.

5.2 Описание процесса моделирования

В процессе моделирования были использованы следующие библиотечные элементы:

- а) BUTTON (рисунок 5.1)– переключатель. Используется для переключения режимов;
- б) LED-RED, LED-GREEN (рисунок 5.2) - светодиоды. В данном случае для информирования что включён режим;
- в) SPEAKER (рисунок 5.4) – пьезодинамик. Используется в качестве сигнальной сигнализации;
- г) RES (рисунок 5.5) – резистор. Резисторы используются ограничения тока, когда кнопку, геркон нажмут.
- д) G2RL-1AB-DC5 (рисунок 5.6) – модуль реле. Используется для подключения водонагревателя.

Реализация схемы управления климатом в САПР Proteus представлена на рисунке 5.8.

Чтобы добавить описание устройства в Proteus был скомпилирован код (в среде Arduino IDE) и получен hex-файл, который был помещен в качестве прошивки в микроконтроллер для эмуляции работы системы.

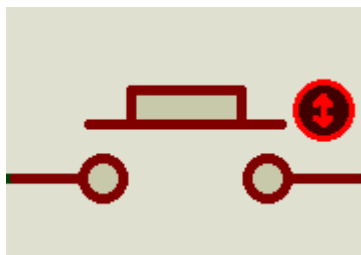


Рисунок 5.1 – Библиотечный элемент BUTTON



Рисунок 5.2– Библиотечный элемент LED-RED, LED-GREEN

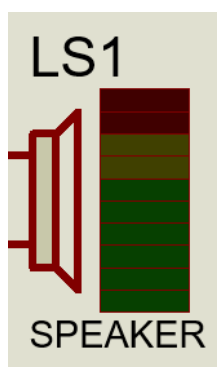


Рисунок 5.3 – Библиотечный элемент SPEAKER



Рисунок 5.4 – Библиотечный элемент RES

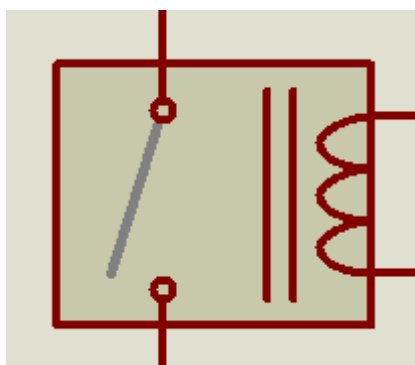


Рисунок 5.5 – Библиотечный элемент G2RL-1AB-DC5

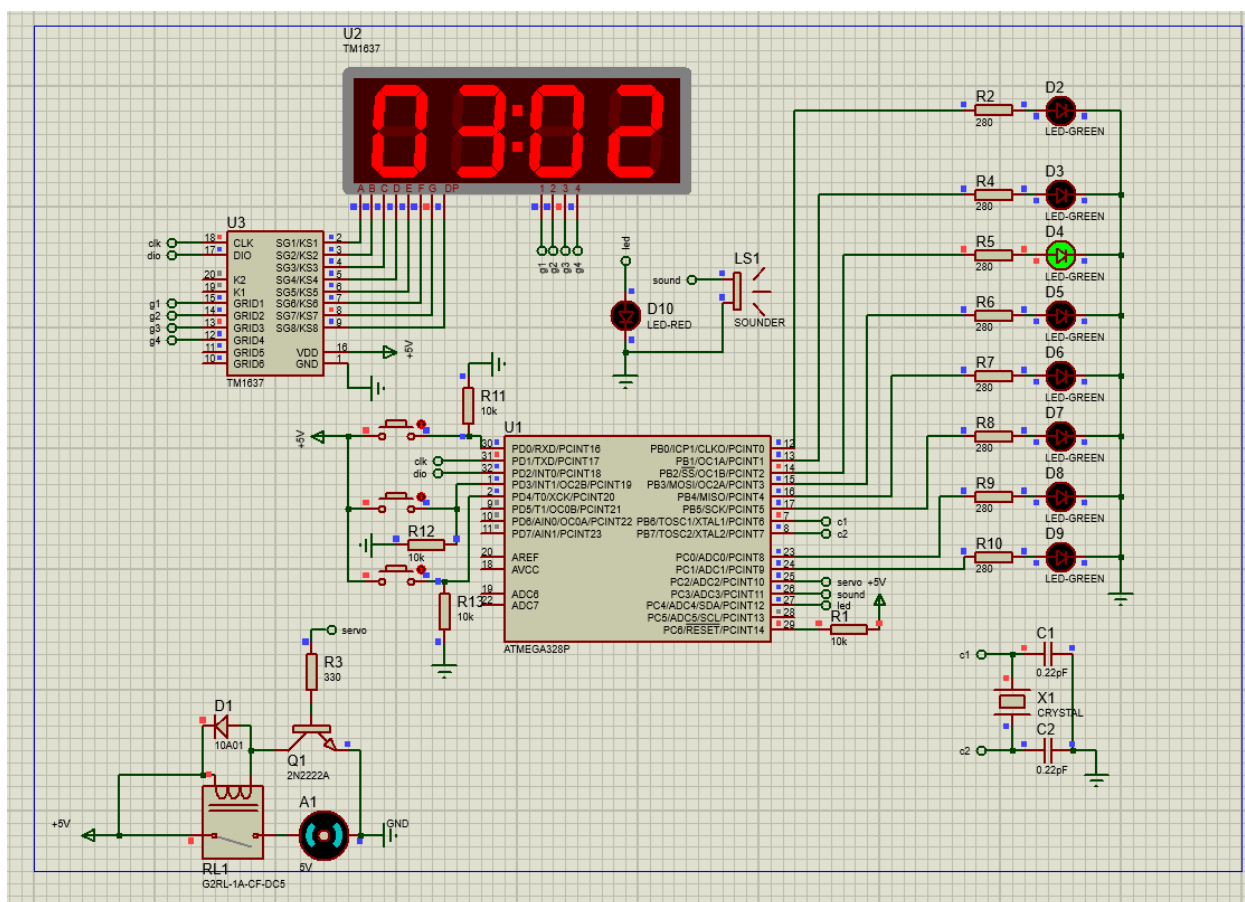


Рисунок 5.8 – Реализация устройства в САПР Proteus

ЗАКЛЮЧЕНИЕ

Микроконтроллеры (ещё одно название — однокристальные микро-ЭВМ) в настоящее время имеют невероятно много областей применения. От промышленной автоматики до бытовых приборов, от управления ядерными станциями до детских игрушек, от секретных военных систем до переключения каналов в вашем радиоприемнике. Одним словом, проще перечислить, где они не применяются. Все это стало возможным в значительной степени благодаря изобретению микропроцессора и созданию микропроцессорных систем. Все эти цифровые устройства сегодня использует большое количество людей. А уже завтра мы получим новые, более «умные» и удобные для нас устройства, которые появятся благодаря грамотному и эффективному использованию всех возможностей микроконтроллеров.

В данном курсовом проекте осуществлялась разработка микропроцессорной устройства управления поливом.

В ходе выполнения данного проекта были рассмотрены реальные задачи, которые решаются проектировщиками, были освещены ключевые моменты, которые требуется знать при проектировании систем управления на базе микроконтроллера. В результате выполнения курсового проекта были получены умения в проектировании реально используемого как на производстве, так и в быту, устройства, были получены навыки в программировании микроконтроллера ATmega328p, а также навыки моделирования его работы в САПР Proteus.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] Описание модуля реле [Электронный ресурс]: 2020 г. URL: <https://arduinomaster.ru/datchiki-arduino/podklyuchenie-rele-k-arduino/>
- [2] Описание датчик температуры [Электронный ресурс]: 2018 г. URL: <https://arduinomaster.ru/datchiki-arduino/arduino-ds18b20/>
- [3] Описание Arduino Uno [Электронный ресурс]: 2020 г. URL: <https://arduinomaster.ru/platy-arduino/plata-arduino-uno/>
- [4] Сайт, посвященный Arduino [Электронный ресурс]: 2021 г. URL: <https://alexgyver.ru/>
- [5] Сайт, посвященный Arduino [Электронный ресурс]: 2021 г. URL: <http://arduino.ru/>
- [6] Сайт, описывающий протокол 1Wire [Электронный ресурс]: 2021 г. URL: <http://wikihandbk.com/wiki/Arduino:Библиотеки/OneWire>
- [7] Статья про микроконтроллер [Электронный ресурс]: 2018 г. URL: <http://elektrik.info/main/automation/549-chto-takoe-mikrokontrollery-naznachenie-ustroystvo-princip-raboty-soft.html>
- [8] Статья про микроконтроллер [Электронный ресурс]: 2018 г. URL: https://myrobot.ru/stepbystep/mc_meet.php
- [9] Статья про proteus 8.12 [Электронный ресурс]: 2020 г. URL: <https://diakov.net/13096-proteus-professional-812-sp0-build-30713-rus.html>
- [10] Статья Г.Горюнов «Почему одни микроконтроллеры надежнее других» [Электронный ресурс]: 2020 г. URL: <https://docplayer.ru/47032710-Pochemu-odni-mikrokontrollery-nadezhnee-drugih.html>
- [11] Сайт, описывающий протокол 1Wire [Электронный ресурс]: 2021 г. URL: <https://avr.ru/beginner/understand/1wire>
- [12] Микропроцессорная техника: учебное пособие / А.А. Петровский [и др.]. – Мн.: БГУИР, 2005. – 51 с.
- [13] Белов, А. Создаем устройства на микроконтроллерах / А. Белов. – СПб. : Наука и Техника, 2007. – 304с.
- [14] Петин, В.А. Проекты с использованием контроллера Arduino / В.Петин. – СПб.: БХВ-Петербург, 2019. – 496с.

ПРИЛОЖЕНИЕ А
(Обязательное)
Схема электрическая структурная

ПРИЛОЖЕНИЕ Б
(Обязательное)
Схема электрическая принципиальная

ПРИЛОЖЕНИЕ В
(Обязательное)
Блок-схема алгоритма

ПРИЛОЖЕНИЕ Г
(Обязательное)
Код программы

```

#include <TimeLib.h>
#include <TM1637.h>

int CLK = 1;
int DIO = 2;
TM1637 tm(CLK,DIO);

int work_hours [8] = {1, 4, 7, 10, 13, 16, 19, 22}; // рабочие
часы полива в зависимости от выбранного режима

int modenumber = 0;
int ledpins [8] = {8, 9, 10, 11, 12, 13, 14, 15}; // номера
пинов для зелёных светодиодов

#define MODE_BUTTON 0
#define TIME_BUTTON 1
#define SETUP_BUTTON 2
int buttonpins[3] = {0, 3, 4}; // номера пинов для кнопок MODE,
TIME, SETUP
// предыдущие состояния для кнопок
int buttonstates[3] = {LOW, LOW, LOW};

int pumppin = 16; // номер пина для водяного насоса
int soundpin = 17; // номер пина для звукового сигнала
int redpin = 18; // номер пина для световой индикации полива

void setup() {
    // код программы, выполняющийся один раз при запуске
    int i;

    // инициализация пинов
    pinMode(pumppin, OUTPUT);
    pinMode(soundpin, OUTPUT);
    pinMode(redpin, OUTPUT);

    tm.init();
    // set brightness 0-7
    tm.set(7);

    for (i = 0; i < 8; i++) {
        pinMode(ledpins[i], OUTPUT);
    }
    for (i=0; i<3; i++)
        pinMode(buttonpins[i], INPUT);
}

void light(int number) { // зажигает светодиод определённого
номера
    int i;
    for (i = 0; i < 8; i++) {
        if (i != number) {

```

```

        digitalWrite(ledpins[i], LOW);
    }
}
digitalWrite(ledpins[number], HIGH);
}

void display_time(time_t t)
{
    int8_t disp[4];
    int h = hour(t);
    int m = minute(t);
    //int s = second(t);

    disp[0] = h/10;
    disp[1] = h%10;
    disp[2] = m/10;
    disp[3] = m%10;

    tm.display(disp);
}

bool isTimeSet = false;
int minutes = 0;
int hours = 0;
int setting_hour = 0; // 0 - меняем минуты, 1 - меняем часы
void init_time(void)
{
    int8_t disp[4];
    // считываем нажатие кнопки изменения времени
    int state = digitalRead(buttonpins[TIME_BUTTON]);
    if (buttonstates[TIME_BUTTON] == LOW && state == HIGH) {
        if(setting_hour) // сейчас меняем час
        {
            hours ++;
            if(hours >= 24)
                hours = 0;
        }
        else // сейчас меняем минуты
        {
            minutes ++;
            if(minutes >= 60)
                minutes = 0;
        }
        //delay(250);
    }
    buttonstates[TIME_BUTTON] = state;

    // считываем кнопку установки
    state = digitalRead(buttonpins[SETUP_BUTTON]);
    if (buttonstates[SETUP_BUTTON] == LOW && state == HIGH) {
        setting_hour ++;
        if(setting_hour == 2)

```

```

        {
            setTime(hours, minutes, 0, 23, 4, 2001);
            isTimeSet = true;
            tm.point(1);
        }
    }
    buttonstates[SETUP_BUTTON] = state;

    disp[0] = hours/10;
    disp[1] = hours%10;
    disp[2] = minutes/10;
    disp[3] = minutes%10;
    tm.display(disp);
}

void change_mode(void) {
    int state = digitalRead(buttonpins[MODE_BUTTON]);
    if (buttonstates[MODE_BUTTON] == LOW && state == HIGH) {
        modenumber++;
    }
    buttonstates[MODE_BUTTON] = state;

    if (modenumber >= 8) {
        modenumber = 0;
    }

    light(modenumber);
}

bool isPumpRunning = false;
void run_pump(void) {
    if (isPumpRunning)
        return;
    digitalWrite(pumppin, HIGH);
    digitalWrite(redpin, HIGH);
    tone(soundpin, 50);
    isPumpRunning = true;
}

void stop_pump(void) {
    if (!isPumpRunning)
        return;
    digitalWrite(pumppin, LOW);
    digitalWrite(redpin, LOW);
    noTone(soundpin);
    isPumpRunning = false;
}

void loop() {
    // код программы, выполняющийся в цикле до выключения
    устройства
    int i;

```

```

if(!isTimeSet)
{
    init_time();
    return;
}

time_t t = now();
display_time(t);

change_mode();

int h = hour(t);
int m = minute(t);
int s = second(t);
if (h == work_hours[modenumber] && m == 0 && s < 30)
    run_pump();
else
    stop_pump();
}

```

Обозначение					Наименование		Дополнительные сведения		
					Текстовые документы				
БГУИР КР 1-40 02 02 023 ПЗ					Пояснительная записка		38 с.		
					Графические документы				
ГУИР 271632.001 Э1					Схема электрическая структурная		Формат А4		
ГУИР 271632.003 Э3					Схема электрическая принципиальная		Формат А3		
ГУИР 271632.004 ПД					Схема алгоритма работы		Формат А3		