Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет энергетический
Кафедра автоматизации производственных процессов и электротехники
Направление подготовки 15.03.04 - Автоматизация технологических процессов и производств
Профиль Автоматизация технологических процессов и производств в энергетике

ДОПУСТИТЬ К ЗАЩИТЕ
И.о.зав. кафедрой
_____ О.В. Скрипко
«07» июля 2020 г.

БАКАЛАВРАСКАЯ РАБОТА

на тему: Система автоматического управления термоэлектрическим модулем Пельтье

Исполнитель
студент группы 641об _____ (подпись, дата) А.А. Ильин

Руководитель
профессор, д-р техн.наук _____ 02.07.2020 (подпись, дата) О.В. Скрипко

Консультант по безопасности
и экологичности
доцент, канд. физ.-мат.наук _____ 02.07.2020 (подпись, дата) В.Н. Аверьянов

Нормоконтроль
профессор, д-р техн.наук _____ 05.07.2020 (подпись, дата) О.В. Скрипко

Благовещенск 2020

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет Энергетический
Кафедра автоматизации производственных процессов и электротехники

УТВЕРЖДАЮ
и.о. Зав. кафедрой
_____О.В. Скрипко
«07» июня 2020 г.

ЗАДАНИЕ

К выпускной квалификационной работе студента *641 группы Ильина Александра Андреевича*

1. Тема выпускной квалификационной работы: Система автоматического управления термоэлектрическим модулем Пельтье
(утверждена приказом от 30.04.2020. № 810-уч)
2. Срок сдачи студентом законченной проекта: *1 июля 2020 года.*
3. Исходные данные к выпускной квалификационной работе: *1) ФГОС направления подготовки бакалавров 15.03.04 Автоматизации технологических процессов и производств; 2) Учебный план направления подготовки бакалавров 15.03.04 Автоматизации технологических процессов и производств.*

4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов):

1) Разработка программного обеспечения микроконтроллера;

2) Разработка печатной платы.

3) Разработка пользовательского интерфейса

5. Перечень материалов приложения (наличие чертежей, таблиц, графиков, схем, программных продуктов, иллюстративного материала и т.п.):

Лист 1: Схема охладителя;

Лист 2: Проектирование печатной плат;

Лист 3: Алгоритм программы;

Лист 5: Комплектующие установки;

Лист 6: Структурная схема

6. Дата выдачи задания 10.03.2020

Руководитель курсовой работы Скрипко Ольга Валерьевна, профессор кафедры АПП и Э, доктор техн. наук.

Задание принял к исполнению (дата): 10.03.2020 Ильин

(подпись студента)

РЕФЕРАТ

Дипломная бакалаврская работа содержит 62с., 54 рисунка, 5 таблиц, 11 источников.

ПЕЛЬТЬЕ, АЛГОРИТМ, ПК, ПЛАТА, ИНТЕРФЕЙС, СИСТЕМА ОХЛАЖДЕНИЯ, РЕГУЛЯТОР

В работе объектом исследования является термоэлектрический модуль Пельтье. Данная работа направлена на разработку автоматической системы управления.

Целью работы является разработка алгоритмов управления температурой микроконтроллером.

В ходе выполнения работы была разработана программа для микроконтроллера по регулированию температуры модуля Пельтье. Создано ПО для отображения данных с микроконтроллера на ПК.

Результатом разработки является автоматическая система управления термоэлектрическим модулем Пельтье на базе микроконтроллерной платформы Arduino uno.

СОДЕРЖАНИЕ

Введение 4
1 Описание объекта управления 5
1.1 История создания (ТЭМ) 5
1.3 Технические характеристики элемента Пельтье 9
1.4 Достоинства и недостатки 11
1.5 Особенности процесса нагрева и охлаждения 13
1.6 Области применения ТЭМ 14
1.7 Стенд для Пельтье предложенный Валуем Р.А. 19
1.8 Описание основных элементов входящих в стенд 21
2 Детальная информация по аппаратной реализации системы 25
2.1 Выбор основных элементов 25
3 Разработка устройства 30
3.1 Изготовление платы для балансировки напряжения 30
3.2 Широтно-импульсная модуляция 36
4. Разработка алгоритма и программы 38
4.1 Алгоритм 38
4.2 Разработка исходного текста программы 41
5 Разработка интерфейса пользователя на пк 50
5.1 Создание стартового окна 51
6 Безопасность жизнедеятельности 54
6.1 Техника безопасности при эксплуатации установки 54
6.1.1 При коротком замыкании электропроводки установки 55
6.2 Экологичность 55
6.3 Правила поведения при чрезвычайной ситуации на установке 56
6.3.1 При пожаре 56
6.3.2 Поломка элемента Пельтье 56
6.3.3 При обнаружении других неисправностей 56
7 Разработка технического задания 57
Заключение 58

Библиографический список 59
Приложение А 61
Приложение Б 64
Приложение В 65
Приложение Г 66
Приложение Д 67
Приложение Е 68
Приложение Ж 69
Приложение З 70

ВВЕДЕНИЕ

В данной работе нужно научиться управлять нагревом с помощью элементов Пельтье. Элемент Пельтье – это термоэлектрический преобразователь, принцип действия которого базируется на эффекте Пельтье - возникновении разности температур при протекании электрического тока.

Для использования термоэлемента Пельтье необходимо подключить его полярность питания в соответствии с поставленной задачей (нагрев или охлаждение). Если поменять полярность питания, поменяется направление тепла (холода), т.е. можно простым переключением превратить его из холодильника в нагреватель. Чем эффективнее будет отвод тепла с горячей стороны, тем эффективнее будет охлаждение.

Главной целью является регулирование температуры холодной стороны по программе, а регулирование температуры горячей стороны стабилизацией температуры.

1 ОПИСАНИЕ ОБЪЕКТА УПРАВЛЕНИЯ

1.1 История создания (ТЭМ)

Эффект Зеебека.Физик Томас Иоганн Зеебек (1770 – 1831), родившийся в семье балтийских немцев, в 1821 году обнаружил, что стрелка компаса отклоняется в замкнутой цепи проводников из двух разнородных металлов, если контакты между ними имеют различную температуру. В то же время датский физик ХансКристиан Эрстед, изучавший явления электромагнетизма, правильно интерпретировал «термомагнитный» эффект Зеебека, введя термин «термоэлектрический эффект», и связал протекание тока с разностью температур, которое в свою очередь и является причиной возникновения замеченного Зеебеком магнитного поля.

Эффект Пельтье Лишь спустя тринадцать лет после открытия термоэлектрического эффекта французский физик Жан Шарль Атаназ Пельтье (1785 – 1845) обнаружил обратный эффект, а именно возникновение разности температур при протекании электрического тока в месте контакта двух разнородных проводников. Пельтье в то время не смог правильно обосновать данное явление, что позже исправил английский физик Уильям Томсон (именуемый также бароном Кельвином, в честь которого и назвали единицу измерения температуры в СИ), сформулировав в 1860 году теорию термодинамики [11].

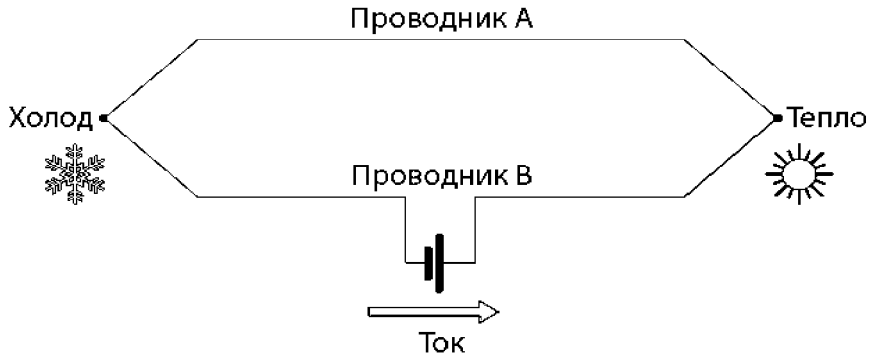


Рисунок 1 – Наглядный пример эффекта

1.2 Принцип действия термоэлектрического элемента

Принцип действия элемента Пельтье заключается в том, что в нем лежит контакт двух полупроводниковых материалов с разными уровнями энергии электронов в зоне проводимости. При протекании тока через контакт электрон должен приобрести энергию, чтобы перейти в более высокоэнергетическую зону проводимости другого полупроводника. При поглощении этой энергии происходит охлаждение места контакта полупроводников. При протекании тока в обратном направлении происходит нагревание места контакта полупроводников.

Если охлаждать нагревающуюся сторону элемента Пельтье, при помощи воздушного охлаждения (радиатора и вентилятора) или жидкостного (водяного радиатора и помпы), то температура холодной стороны становится ниже. В зависимости от типа элемента и величины тока, разность температур может достигать приблизительно 70 °C.

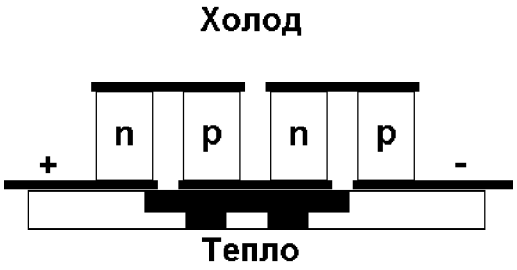


Рисунок 2– Использование полупроводников p- и n-типа в термоэлектрических холодильниках

Элемент Пельтье состоит из нескольких пар небольших полупроводниковых паралелепипедов – одного n-типа и одного p-типа в паре, которые совместно соединены с помощи металлических перемычек. Металические перемычки и изолированы непроводящей пленкой или керамической пластиной и служат термическими контактами. Пары

соединяются таким образом, что образуется последовательное соединение нескольких пар полупроводников с разным типом проводимости, так чтобы сверху были одни последовательности соединений с n-p переходом, а снизу - p-n переходом.

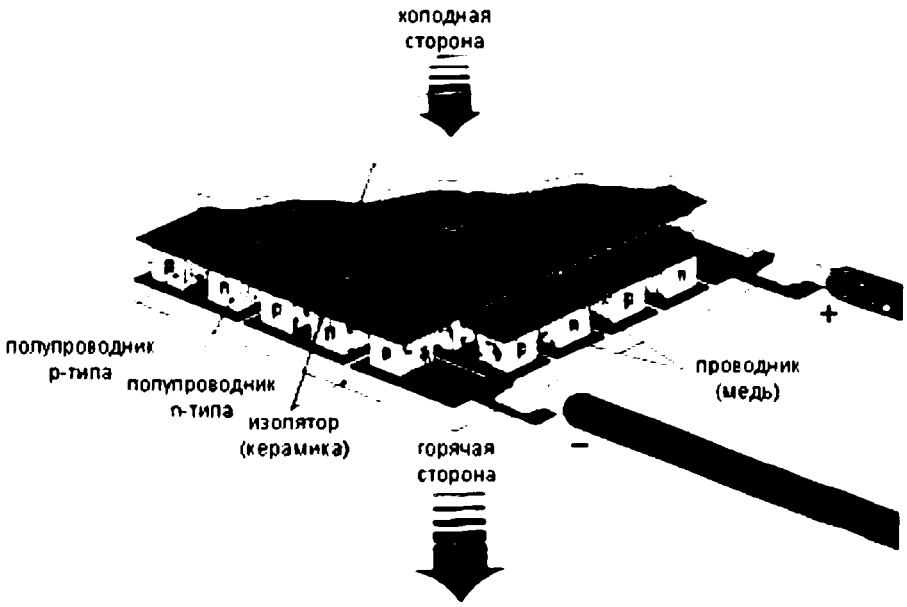


Рисунок 3– Структура модуля Пельтье

В зависимости от направления тока контакты, находящиеся сверху, охлаждаются, а, находящиеся снизу, нагреваются или наоборот. Таким образом, электрический ток переносит тепло с одной стороны ТЭМ на другую и образует разность температур.

В результате прохождения электрического тока определенной полярности создается перепад температур между радиаторами модуля Пельтье: один радиатор используется для отвода тепла, а другой радиатор служит как холодильник. На рисунке 4 представлен внешний вид модуля Пельтье.

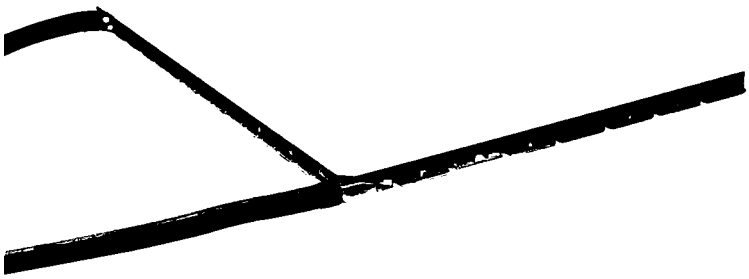


Рисунок 4 – Внешний вид модуля Пельтье

Типичный модуль образует значительный температурный перепад, составляющий несколько десятков градусов. Второй радиатор, который служит как холодильник позволяет достигать отрицательных значений температур, при принудительном охлаждении радиатора отвечающего за отвод тепла. Для увеличения разности температур можно использовать каскадное подключение термоэлектрических элементов Пельтье при обеспечении адекватного для них охлаждения. Это позволяет получить простыми средствами температурный перепад значительнее и образовать эффективное охлаждение защищаемых элементов. На рисунке 5 представлен пример каскадного включения типовых модулей Пельтье.

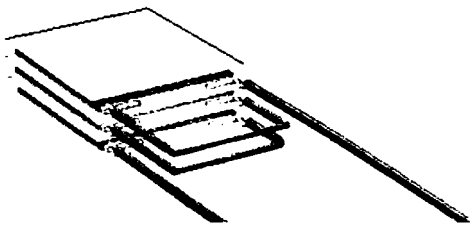


Рисунок 5 – Пример каскадного включения модулей Пельтье

1.3 Технические характеристики элемента Пельтье

Таблица 1 – Технические параметры TEC1-12706 [8].

Обозначение	Параметр	Значение, при температуре	
	Температура горячей стороны	25 °C	50 °C
Qmax	Холодопроизводительность	50 Вт	58 Вт
DeltaTmax	Температурная разность	65 °C	75 °C
Imax	Ток максимальный	6,5 А	6,5 А
Umax	Напряжение максимальное	14,3 В	16,3 В
Resistance	Сопротивление	1,97 Ом	2,28 Ом

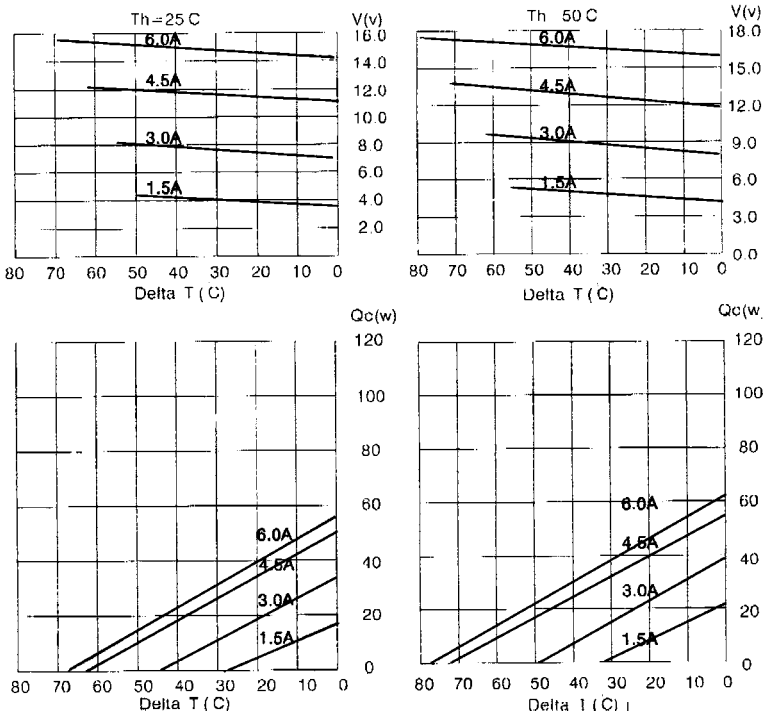


Рисунок 6 – Графические характеристики

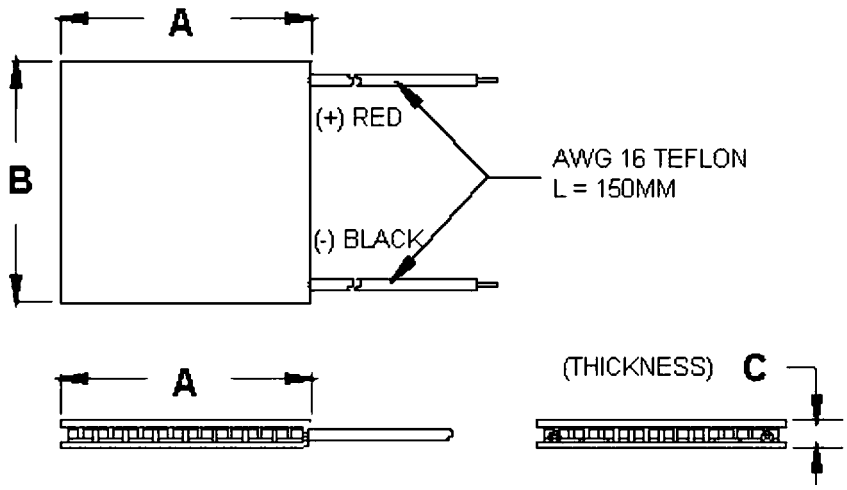


Рисунок 7 – Габаритный чертеж модуля TEC1-12706.

Таблица 2 – Габаритные размеры

Обозначение	Размер
A	40 мм
B	40 мм
C	3,8 мм

Максимально допустимая температура 138 °C. Недопустимо превышение значения параметров Imax и Umax. Срок службы 200 000 часов.

1.4 Достоинства и недостатки

У элемента Пельтье достоинствами являются малогабаритные размеры, нет движущихся частей, также главным достоинством можно отметить отсутствие шума при работе.

Из-за отсутствия движущихся частей и быстроизнашивающихся деталей все элементы Пельтье являются очень надежными, долговечными и не нуждаются в техобслуживании. Еще одним преимуществом Пельтье является недорогое производство. В добавок, они не создают никаких вибраций и шумов. Поэтому элементы компактны и позволяют объединять даже несколько модулей в один блок. В системах с Пельтье нет хладагентов, которые легко воспламеняются, разрушают озоновый слой и вызывают парниковый эффект.

Отпадает надобность в целом контуре охлаждения с компрессором, дросселем и конструктивными элементами испарителя и конденсатора, занимающие много места. Модули не нуждаются в техобслуживании, а в случае выхода из строя легко заменяются.

Современные технологии регулирования могут точнее дозировать эффект охлаждения, чем в стандартном компрессоре. К тому же, при переключении полюсов функция может изменяться на противоположную, то есть охлаждение может стать нагревание или наоборот[11].

Из недостатков элемента Пельтье низкий коэффициент полезного действия, чем у компрессорных холодильных установок на фреоне, что приводит к большому потреблению мощности для достижения заметной разности температур. Тогда как, сейчас ведутся разработки по повышению теплового КПД, а модулям Пельтье нашлось широкое применение в технике, так как без каких-нибудь дополнительных устройств можно реализовать Отрицательные температуры.

Основная проблема в построении элементов Пельтье с высоким КПД заключается в том, что в веществе свободные электроны одновременно переносят электрический ток, и тепло. И из-за этого материл в модуле Пельтье должен и хорошо проводить ток и плохо проводить тепло.

В батареях элементов Пельтье можно достигнуть большой разницы температур, но из-за этого мощность охлаждения будет ниже. Лучше для стабилизации температурыиспользовать импульсный источник питания, потому что это даст повысить эффективность системы. И при том желательно сглаживать пульсации тока – это увеличит эффективность работы Пельтье и, возможно, продлит срок его службы. И все же работа элемента Пельтье будет неэффективной, если использовать широтно-импульсную модуляцию (ШИМ) и с помощью нее стабилизировать температуру.

При создании модулей Пельтье технически неизбежно очень тесное расположение теплой и холодной сторон. На практике толщина современных модулей Пельтье составляет около 3-5 мм, в этом случае требуется очень эффективный подвод и отвод тепла. Эта техническая проблема решается установлением больших радиаторов с вентиляторами. Производительность элемента Пельтье зависит в первую очередь от необходимой разности температур. Чем больше разность температур, тем меньше требуется

мощность, а с разностью около 70К мощность упадет до нуля (на данном этапе развития технологии Пельтье). Более высокие значения разности удается реализовать лишь за счет затратной установки многоступенчатых модулей.

1.5 Особенности процесса нагрева и охлаждения

Для применения модулей Пельтье есть ряд требований, невыполнение которых может привести: к снижению эффективности системы, деградации модуля или выхода его из строя. элементы выделяют значительное количество тепла и для его отвода необходимо поставить радиатор.

Иначе невозможно достичь нужной температуры холодной стороны, потому что модуль Пельтье понижает температуру относительно горячей стороны. С высокими температурами кристалы элементов деградируют, из-за этого снижается эффективность и срок службы элемента. Максимально допустимый нагрев горячей стороны + 80 °С (в высокотемпературных до 150 °С). Иначе модуль может просто выйти из строя.

Важен надежный тепловой контакт модуля с радиатором охлаждения.

Источник питания для модуля должен производить ток с пульсациями не более 5%. Если будет более высокий уровень пульсаций эффективность модуля снизится на 30-40%.

Так же не допустимо, для управления модуелм Пельтье, применять релейные регуляторы. Это может привести к быстрой деградации элементов. Каждое включение и выключение вызывает деградацию а полупроводниковых термопарах. Из-за таких резких изменений температуры между сторонами элемента возникают механические напряжения в местах спайки полупроводников.

Так же модуль Пельтье образует высокую теплопроводность между соторонами. В выключенном состоянии, тепло радиатора горячей стороны, через элемент будет передаваться на холодную сторону.

Из-за этого недопустимо, для регулирования мощности на модуле Пельтье, использовать ШИМ модуляцию.

Для питания модуля Пельтье часто используют источник напряжения. Его проще применить. Но вольт-амперная характеристика элемента Пельтье нелинейная и крутая. Так как при небольшом изменении напряжения ток сильно меняется. И к тому же, характеристика меняется при изменении температуры сторон эелмента Пельтье. Нужно стабилизировать мощность, т.е. произведение тока через элемент на напряжение на нем. Охлаждающая способность элемента Пельтье напрямую зависит от электрической мощности. Конечно, для этого нужен достаточно сложный регулятор.

Напряжение модуля зависит от количества термопар в нем. Обчно это 127 термопар, что соответствует напряжению 16 В. Рекомендуется подавать 12 В на Пельтье, или 75% от максимального напряжения. С таким напряжением обеспечивается оптимальная эффективность элементов.
Из-за герметичного исполнения элементы Пельтье можно использовать даже в воде.

Полярность модуля отмечена цветами проводов – черный и красный. Как правило, черный (отрицательный) провод расположен слева, относительно холодной стороны.

1.6 Области применения ТЭМ

Основные направления практического использования эффекта Пельтье в полупроводниках: получение холода для создания термоэлектрических охлаждающих устройств, подогрев для целей отопления, термостатирование, управление процессом кристаллизации в условиях постоянной температуры. Термоэлектрические модули (ТЭМ) применяются в устройствах охлаждения радиоэлектронных компонентов и различных устройствах термостатирования ввиду легкости прецизионного электронного регулирования температуры, как для нагрева, так и для охлаждения.

Максимальная холодопроизводительность ТЭМ получается при определенном значении тока, который при заданном значении напряжения питания показывается как Imax. Нестационарный режим питания импульсами

тока, в несколько раз превышающими Imax, на некоторое время позволит получить холодопроизводительность, намного превышающую паспортную. Это объясняется тем, что сам эффект Пельтье безынерционен, в отличие от распространения теплоты джоуля и явления теплопроводности, и, в течение нескольких секунд, этим можно воспользоваться. Впрочем, нестационарные режимы широкого применения не получили.

Ввиду обратимости термоэлектрических эффектов, ТЭМ может использоваться и в качестве термоэлектрических генераторов (ТЭГ). Вдали от удобств цивилизации это может быть один из немногих доступных источников электрической энергии, например, для подзарядки аккумуляторов или прямого питания радиоэлектронной аппаратуры или других устройств. Достаточно широко используются устройства, в которых разница температур создается между наружной металлической оболочкой, нагреваемой открытым огнем (костром), и внутренней оболочкой, охлаждаемой водой. «Холодная» сторона будет ограничена температурой кипения воды, поэтому такой ТЭМ должен быть рассчитан на рабочую температуру 500 – 600°С. Следует иметь в виду, что тепловой баланс для ТЭГ качественно отличается от ТЭМ на основе эффекта Пельтье, и этот эффект (вместе с теплотой Джоуля) вносит всего несколько процентов в общий вклад, что требует совершенно других акцентов при конструировании ТЭГ. ТЭГ широко применяются в космической технике, где температура «горячей» стороны поддерживается радиоизотопным источником. Вживляемые в тело человека кардиостимуляторы также снабжены ТЭГ с радиоизотопным источником для создания разности температур.

Также элементы Пельтье часто применяются для охлаждения и термостатирования диодных лазеров, с тем, чтобы стабилизировать длину волны излучения. В приборах, при низкой мощности охлаждения, элементы Пельтье часто используются как вторая или третья ступень охлаждения. Это позволяет достичь температур на 30-40°С ниже, чем с помощью обычных компрессионных охладителей.

Основным отличием ТЭМ для промышленного применения является их высокая надёжность, максимальная эффективность и стабильность параметров. Эксплуатация в условиях высоких температур, необходимости быстрой смены режима охлаждения на термоэлектрический нагрев (в этом случае эффективность нагрева >1), работа в условиях повышенной влажности, всё это определяет требования, предъявляемые к ТЭМ для промышленного применения. Ниже приведены некоторые области применения охлаждения и температурной стабилизации с помощью термоэлектричества.

Основные области применения:

1) В общепромышленных блоках и узлах:

– шкафы и стойки с оборудованием;

– процессоры и быстродействующие устройства;

– сложные оптические приборы;

– лазеры.

2) Технологическое оборудование:

– жидкостное охлаждение и прецизионная термостабилизация рабочих зон технологических линий по производству микросхем, дисплеев и др.

– прецизионное поддержание рабочих температур в технологических ваннах (гальванических, травления, промывки и др.);

– термокамеры для испытания электронных компонентов и узлов на воздействие климатических, в том числе циклических воздействий;

– устройства закрепления деталей для прецизионной механической обработки с помощью локального замораживания.

Холодильные боксы с модулями Пельтье, предназначенные для интенсивного охлаждения продуктов питания и напитков, идеально подходят для автомобилей и дач на колесах, ведь они мобильны в своем использовании и могут питаться от 12-вольтовой бортовой сети.

Главный недостаток элементов Пельтье применяется в осушителе воздуха. Осушаемый воздух пропускается через холодную сторону элемента Пельтье, а образующийся конденсат стекает в специальный поддон[11].

3) Медицинская техника:

– охлаждение и кондиционирование медицинской техники;

– портативные холодильники для перевозки биологических препаратов;

– приборы для проведения ДНК анализов;

– теплообмен для хирургии;

– терапевтические приборы;

– холодные поверхности и изотермальные ванны для биологии и фармакопеи.

Одной из причин широкого применения охлаждения с помощью элементов в медицинских приборах является безопасность напряжения питания термо-элетрическим модулем. Ярким примером термостатирования по технологии Пельтье является амплификатор – лабораторный прибор для амплификации последовательностей ДНК. При так называемой реакции ПЦР (полимеразной цепной реакции) требуется три различных быстро сменяющихся температуры реакции. Ввиду своего устройства компрессорные системы бывают лишь определенных размеров. А модули Пельтье могут быть крошечными, обладая при этом удивительно высокой мощностью охлаждения. Миниатюрные элементы Пельтье используются, например, в сцинтилляторах, где требуется подавление шумов фотодиодов путем охлаждения.

Функционирование большинства компрессорных систем зависит от положения. В наклоненном или перевернутом состоянии они перестают работать, равно как и в невесомости. А модули Пельтье можно встраивать в мобильные устройства, которые питаются от широко распространенного 12-вольтового аккумулятора. Пример использования: денсиметры, вискозиметры, реометры или рефрактометры.

4) Измерительная техника:

- осушение проб газа в газоанализаторах;
- температурная стабилизация эталонов «чёрного тела»;
- датчики точки росы;
- измерители тепловых потоков.

5) Транспортные средства:

- локальное кондиционирование в кабинах транспортных средств;
- нагрев/охлаждение сидений автомобилей и др. видов транспорта;
- холодильники и охладители напитков.

6) Пищевая промышленность:

- устройства охлаждения и поддержания температуры пищи во время её приготовления, упаковки, транспортировки и хранения;
- охладители напитков для баров, ресторанов и кафе.

7) Промышленное применение может быть разделено на следующие конструктивные группы:

- высокоэффективные однокаскадные элементы;
- обычные однокаскадные элементы;
- круглые элементы;
- прямоугольные элементы с отверстием;
- двухсекционные однокаскадные элементы.

Таблица 3– используемые сокращения

Сокращение	Расшифровка
ТЭМ	термоэлектрический модуль (охлаждающий)
ТГМ	термоэлектрический генераторный модуль
DTmax	максимально достижимая разница температур между сторонами термоэлектрического модуля
Imax	максимальный электрический ток проходящий через термоэлектрический элемент, соответствующий режиму максимальной разницы температур

Umax	Максимальное напряжение на контактах элемента
Qmax	максимальная холодопроизводительность (холодильная мощность) элемента. Определяется при максимальном токе через термоэлектрический модуль и нулевой разности температур между его сторонами
Rac	Измеренное сопротивление тока на элементе при протекающем токе с частотой 1 кГц

1.7 Стенд для Пельтье предложенный Валуем Р.А.

На рисунке 21 показана структурная схема устройства. Объектом управления является элемент Пельтье.

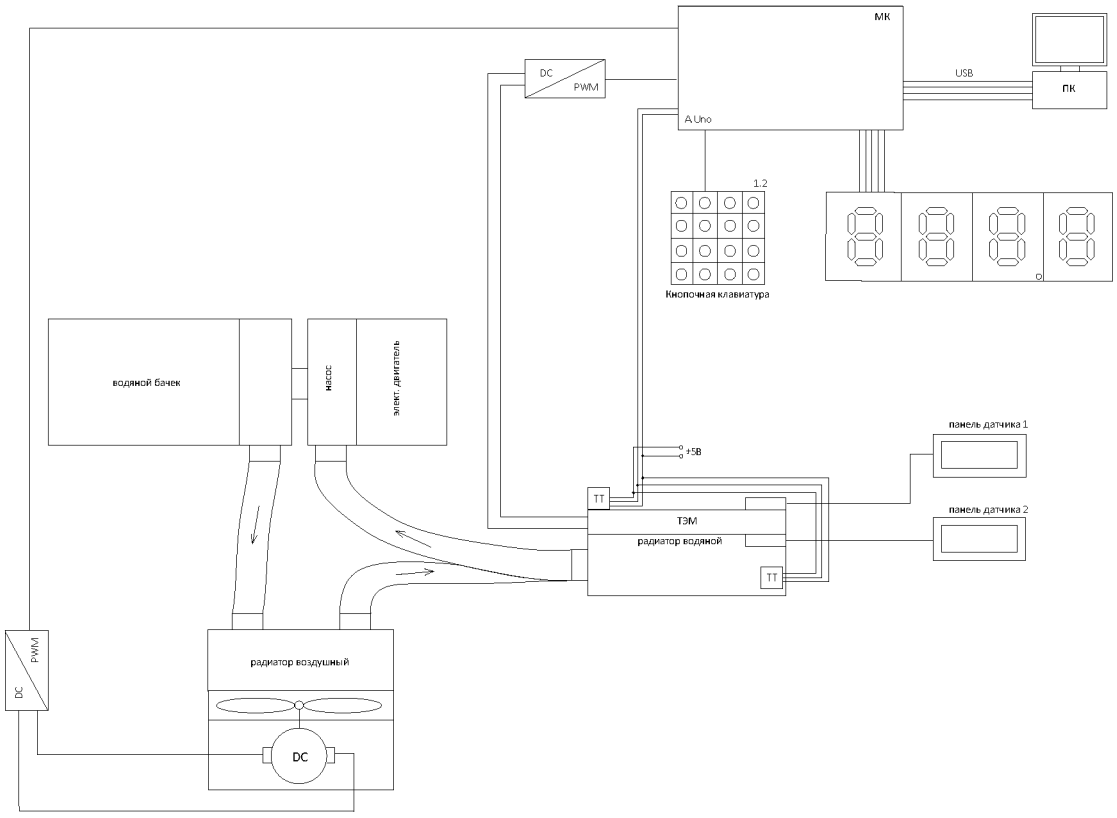


Рисунок 8 - Схема охладителя

К радиатору с одной стороны крепится водяной радиатор для отвода тепла, а с другой ТЭМ. Радиатор поставлен на небольшие проставки, сделанные из орг. стекла. При прохождении тока одна сторона нагревается, а другая охлаждается.

После того как задали температуру с кнопочной клавиатуры, она будет отображаться на индикаторе, запускается насос гоняющий жидкость по всей системе, а также вентилятор, который охлаждает жидкость. Это сделано для избежание перегрева самого элемента. Затем запускается сам ТЭМ.

Температура жидкости регулируется при помощи вентилятора. Если нужно охладить холодную сторону сильнее, то увеличиваем расход подачей напряжения на вентилятор.

Показания температуры сторон элемента выводятся на панели термодатчиков.

ШИМ(PWM) управляет нагрузкой (управляет скоростью вращения двигателя)

Схема охладителя была собрана в виде стенда. Рисунок 9 и 10.

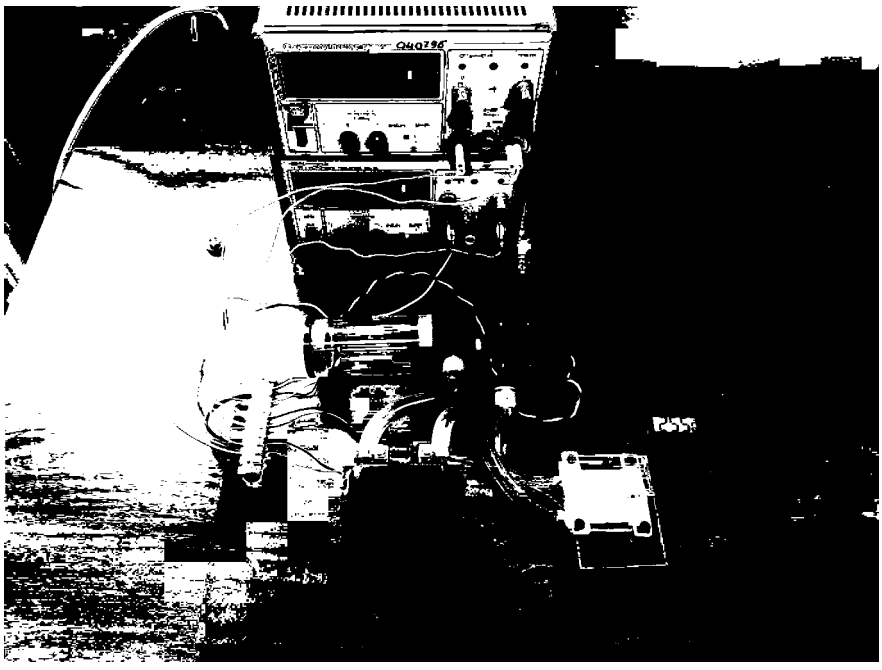


Рисунок 9 – Установка общий вид

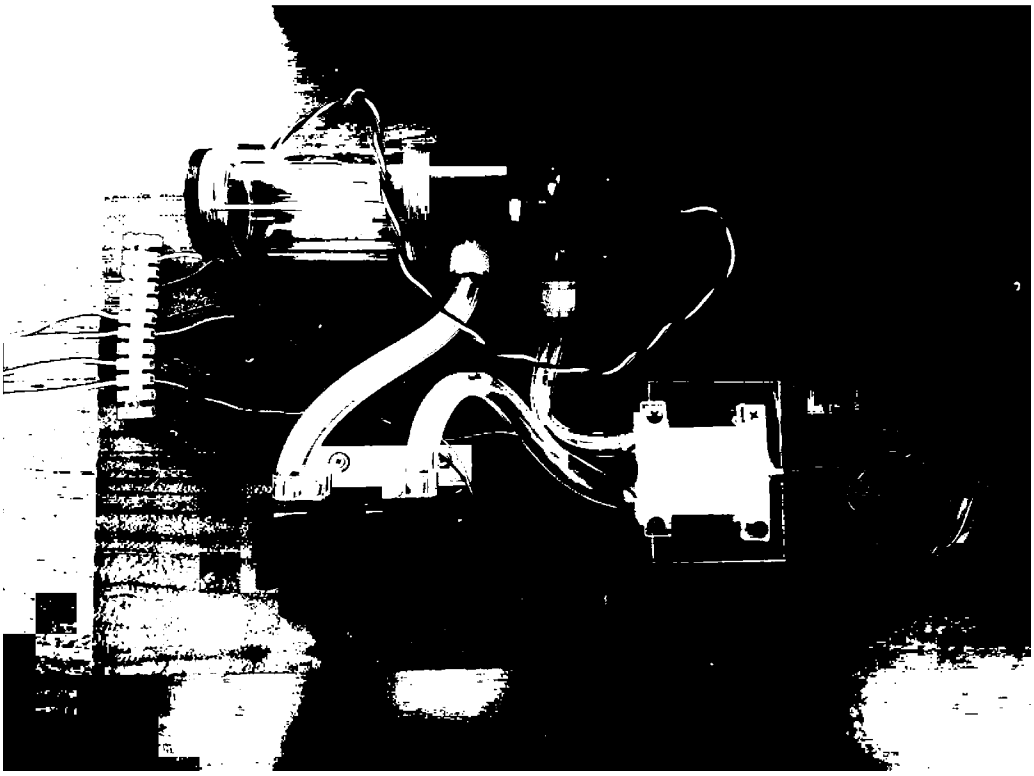


Рисунок 10 – Установка вид сверху

1.8 Описание основных элементов входящих в стенд

Вентилятор используем обычный, размером 40 х 40 мм, толщина 10 мм. Рабочее напряжение 12 В. Четыре крепежных отверстия под болты М3 или саморезы. Без датчика оборотов.

Рисунок 11 – Вентилятор

Приобретен элемент Пельтье. TEC1-12706

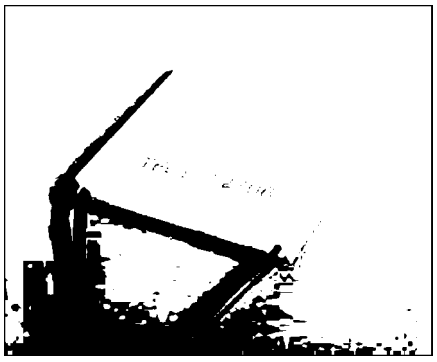


Рисунок 12 – ТЭМ

Термодатчик выносной. Питание от батареек. Вся техническая информация в таблице 4.

Таблица 4–Технические характеристики термодатчика

Наименование	Значение
Область применения	измерение температуры жидкостей и газов
Длина кабеля	1м.
Температура, диапазон измерения	-50 +110 °С
Размер дисплея	48х28мм.
Диаметр нержавеющей гильзы	4мм.
Длина гильзы	20мм.
Тип крепления выводов к индикатору	разъемное
Точность измерения	±0,5°С
Масса	0,025кг.

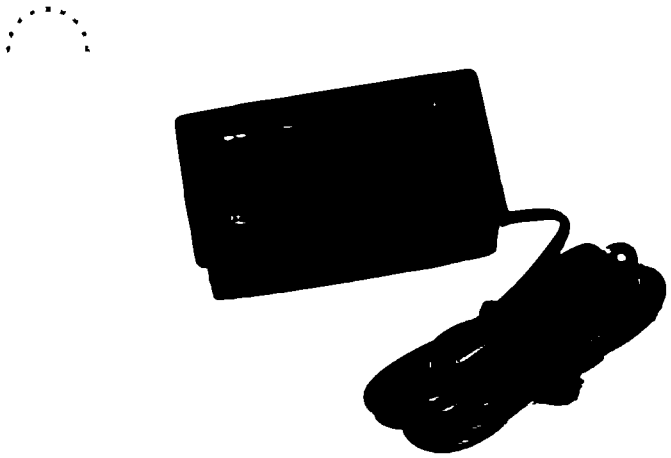


Рисунок 13 – Термодатчик

Для охлаждения ТЭМ используется алюминиевый блок водяного охлаждения. На котором имеется в ход и выход для протока жидкости.

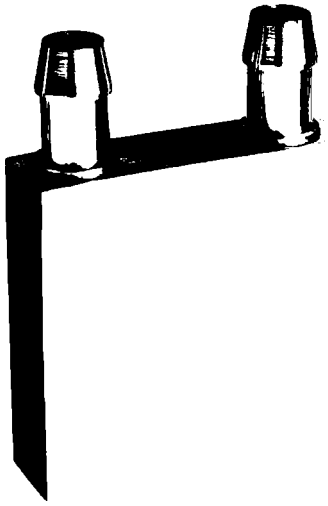


Рисунок 14 – Радиатор водяной

Помпа СВО (водяной насос) SC600 для систем жидкостного охлаждения компьютера. Высокая производительность при минимальном уровне шума.

Характеристики помпы SC600:

- напряжение питания - 12 Вольт,
- ток потребления - 0,8 Ампер,

– производительность - 500 литров в час,
– скорость вращения - 5600 об/мин,
– пусковое напряжение - 6 Вольт,
– габаритные размеры - 71х56х46 мм,
– резьба G1/4,
– керамический вал,
– керамическая втулка.

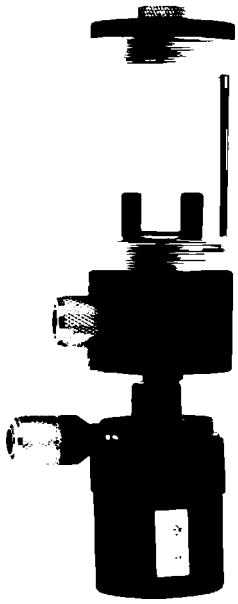


Рисунок 15 – Помпа

2 ДЕТАЛЬНАЯ ИНФОРМАЦИЯ ПО АППАРАТНОЙ РЕАЛИЗАЦИИ СИСТЕМЫ

2.1 Выбор основных элементов

В нашей работе будет необходим микроконтроллер, где будет не менее 24 входов/выходов. Нам подходит микроконтроллер серии ATmega328.

Его плюсы заключаются в том, что он высокопроизводительный, маломощный микроконтроллер на базе AVRRISC, балансирует потребление энергии и скорость обработки. Все необходимое для установки покупалось на сайте AliExpress. Чтобы не заказывать все отдельно был приобретен набohArduino. В комплекте, которого входило все необходимое.

Подобраны кнопки KLS7-TS6601, рабочий ток до 0,05А [4]. В нашей установке кнопки не несут большой важности, поэтому взяли недорогие и более простые по конструкции.

Так же был выбран текстовый дисплей, элемент работает при значении прямого тока равного 1 мА.

Цифровой датчик температуры Dallas 18B20 [7].

Помпа СВО (водяной насос) SC600 для систем жидкостного охлаждения компьютера. Высокая производительность при минимальном уровне шума [9].

Транзисторы будем использовать BC817 и BC807. Напряжение коллектор-эмитер не более 45В, напряжение эммитер-база не более 5V, напряжение коллектор-база не более 50В. Ток коллектора не более 0,5А.

Arduino - это небольшая плата на основе микроконтроллера, которая может быть использована при разработке цифровых устройств

Таблица 5 – Характеристики платы Arduino Uno.

Микроконтроллер	ATmega328
Рабочее напряжение	5В
Напряжение питания	7-12В
Напряжение питания	6-20В
Цифровые входы/выходы	14 (из них 6 могут использоваться в качестве ШИМ-выходов)
Аналоговые входы	6
Максимальный ток одного	40 мА
Максимальный выходной	50 мА
Flash-память	32 КБ (ATmega328) из которых 0.5
SRAM	2 КБ (ATmega328)
EEPROM	1 КБ (ATmega328)
Тактовая частота	16 МГц

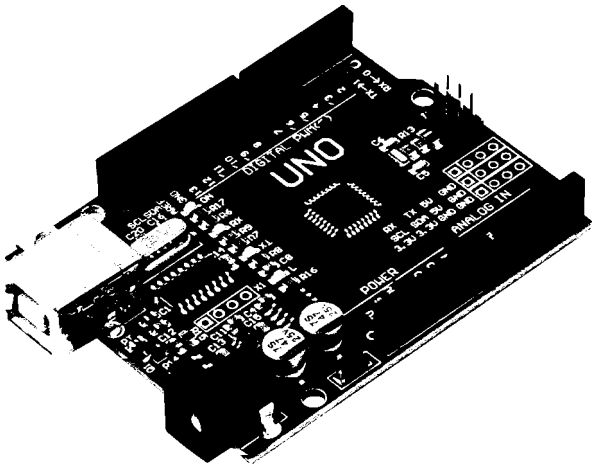


Рисунок 16 - Плата Arduino Uno на микроконтроллере ATmega328

Для опытов с программной частью понадобится:

Тактовая кнопка KLS7-TS6601 имеет прямой тип, четыре ножки для более лучшей фиксации. Рабочий ток до 0,05 А. внешний вид показан на рисунке 20[4].

Рисунок 17 – Тактовая кнопка

Транзисторы будем использовать BC817 и BC807 рисунок 22,. Напряжение коллектор-эмитер не более 45В, напряжение эммитер-база не более 5V, напряжение коллектор-база не более 50В. Ток коллектора не более 0,5А[14].

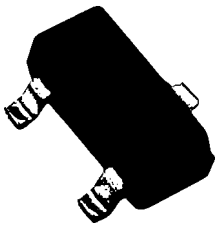


Рисунок 18 – Транзистор BC817 и BC807

IRF3205 (HEXFET) говорится – это современный, высокопроизводительное устройство от компании International Rectifier (IR) с индуцированной конструкцией затвора

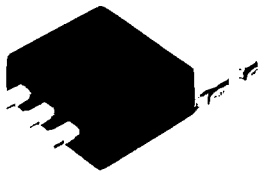


Рисунок 19 - Наименование прибора: IRF3205

- Тип транзистора: MOSFET
- Полярность: N
- Максимальная рассеиваемая мощность (Pd): 36 W
- Предельно допустимое напряжение сток-исток (Uds): 350 V
- Максимально допустимый постоянный ток стока (Id): 1.7 A
- Максимальная температура канала (Tj): 150 °C
- Сопротивление сток-исток открытого транзистора (Rds): 5 Ohm
- Тип корпуса: TO220

Текстовый экран MT-16S2H вывода текста с подсветкой. На экране одновременно могут поместиться 32 символа (по 16 символов в 2х рядах) .

Для подключения экран имеет 16 контактов. Для удобства использования к контактам дисплея уже впаяны штырьковые соеденители и его можно с легкостью вставить в макетную плату, либо подключить при помощи проводов

Таблица 6– Технические характеристики текстового экрана

Напряжение питания	5 В.
Максимальный ток потребления	1 мА.
Потребляемый ток подсветки	100 мА.
Индикация	2 строки по 16 символов
Габариты	88×44×13 мм.

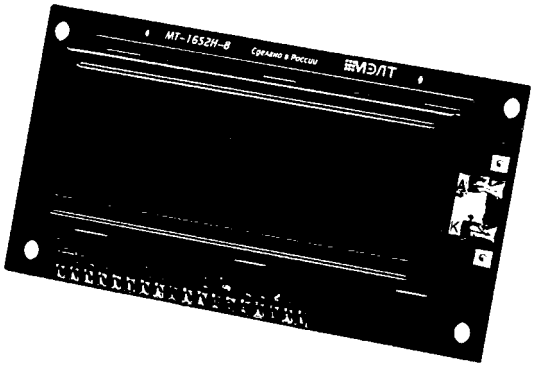


Рисунок 20 – Текстовый экран

Цифровой датчик температуры Dallas18B20 (именуемый также DS18B20) является компактным, точным и недорогим цифровым датчиком температуры. Датчик использует интерфейс 1-Wire, разработанный производителем - компанией Dallas. Данный интерфейс широко распространен и довольно прост в освоении. Дополнительным его плюсом является возможность параллельного подключения нескольких датчиков на одну шину данных.

Микросхема 18B20 просто подключается к микроконтроллерным устройствам на базе Arduino, AVR, PIC, ARM и др. Для работы с Arduino существует готовая библиотека, созданная производителем.

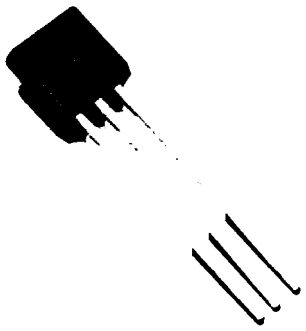


Рисунок 21 – Датчик температуры

3 РАЗРАБОТКА УСТРОЙСТВА

3.1 Изготовление платы для балансировки напряжения

Нам необходимо реализовать узел импульсного регулятора со следующими функциями:

- собственно, ключевой регулятор (ключ, дроссель, рекуперативный диод, сглаживающий конденсатор);
- цепь измерения напряжения на нагрузке;
- цепь измерения тока регулятора;
- аппаратная защита от превышения тока.

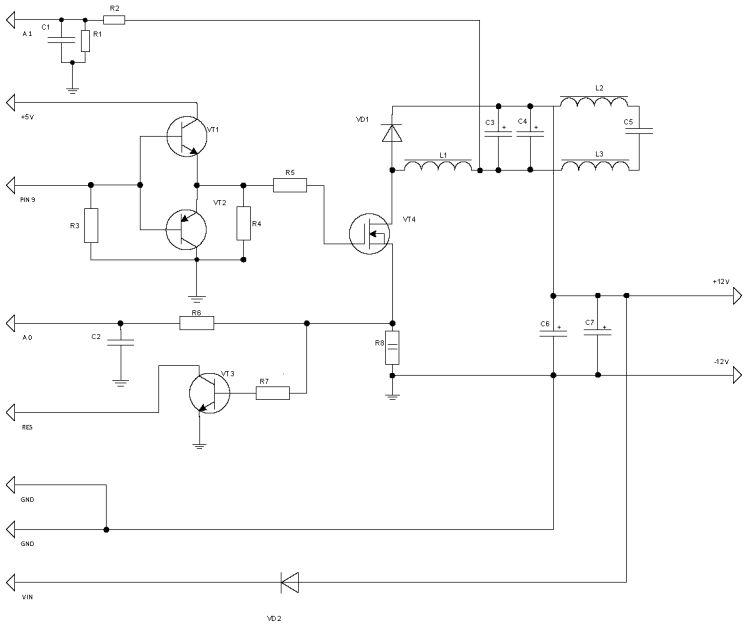


Рисунок 22 - Схема импульсного регулятора

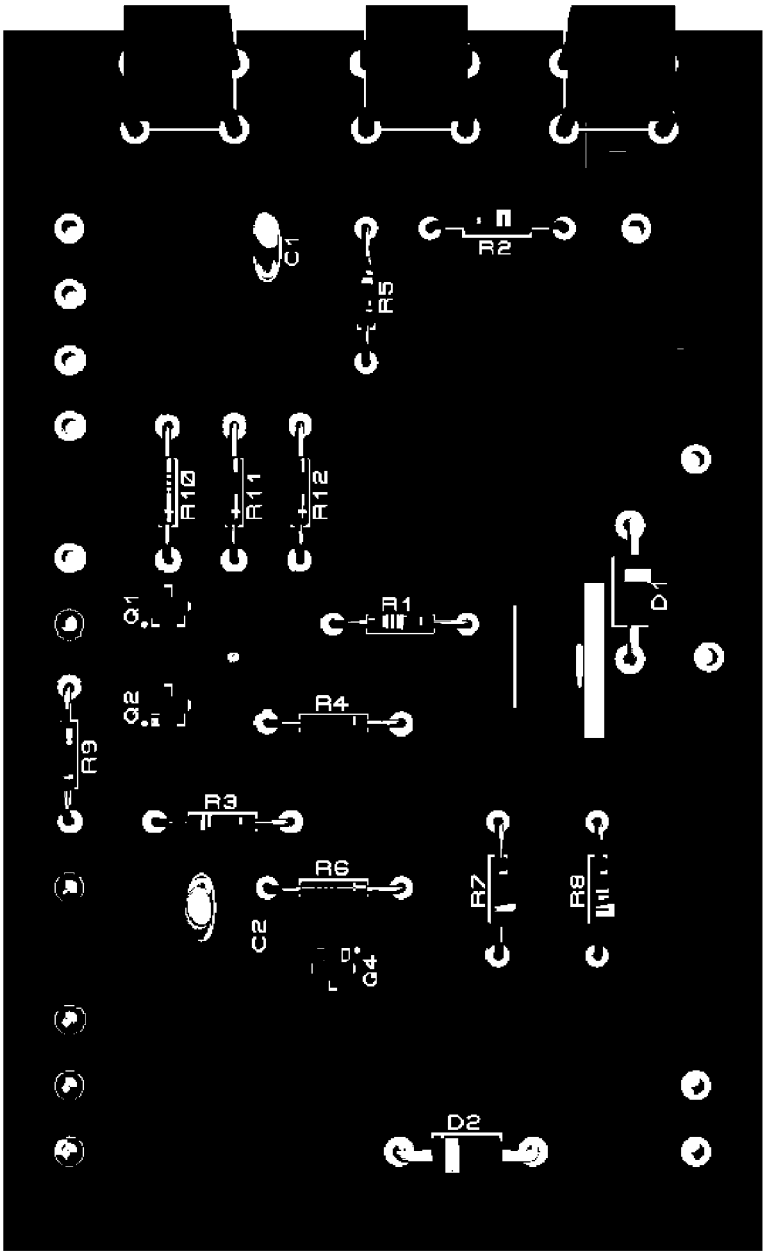


Рисунок 23 - 3Д вид импульсного регулятора

Для начала было решено сделать небольшую часть платы для балансировки напряжения и протестировать её.

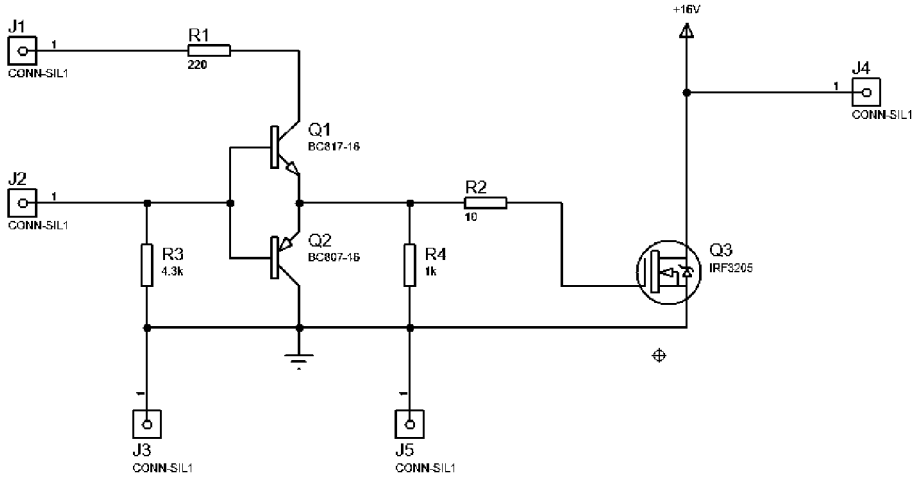


Рисунок 24 - Схема балансировки напряжения в программе «Proteus»

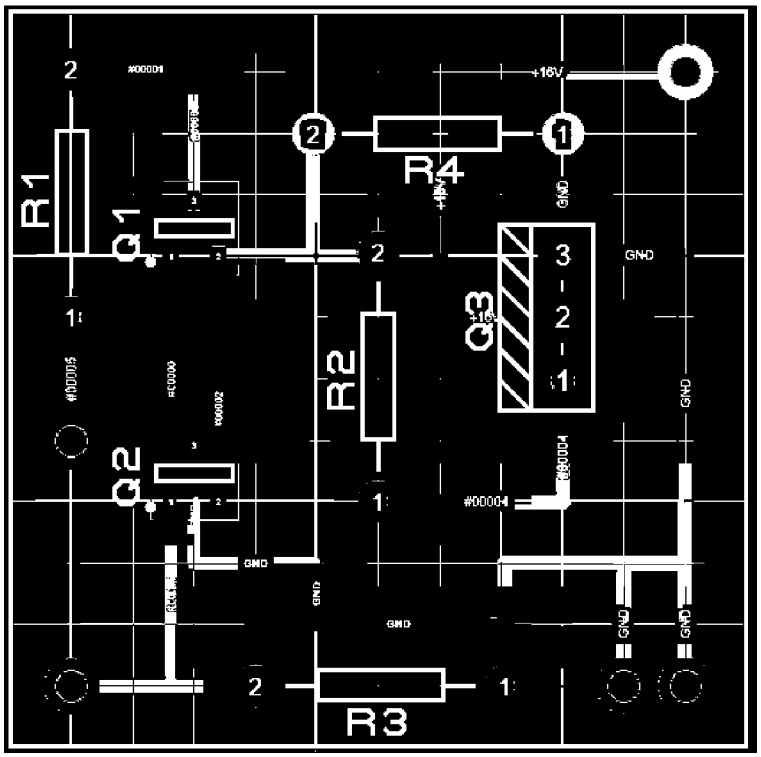


Рисунок 25 - Вид печатной платы в подпрограмме Layout

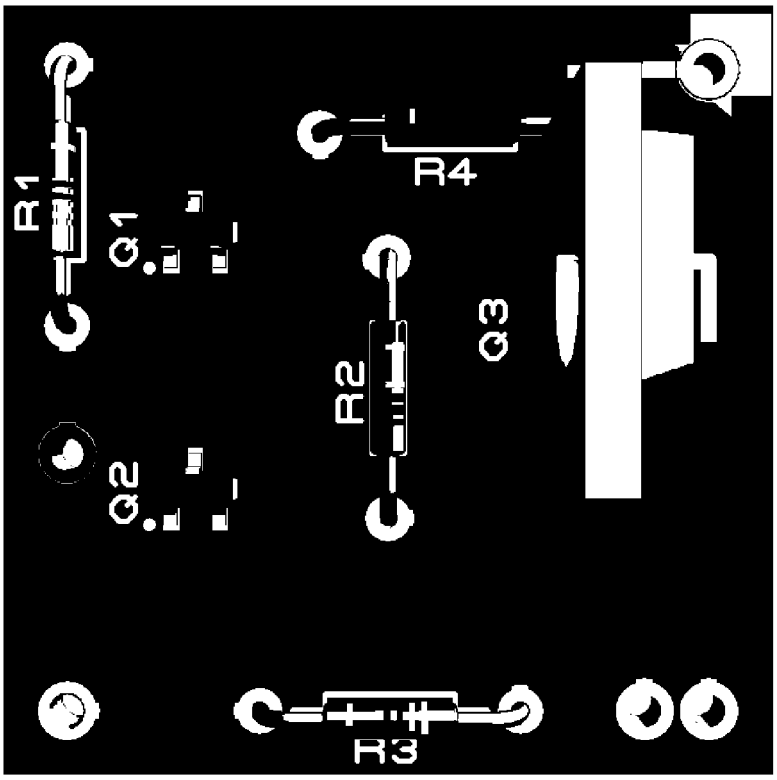


Рисунок 26 - 3Д вид монтажной схемы в Proteus

Разработанная плата будет изготавливаться технологией ЛУТ – лазерно-утюжная технология. ЛУТ или лазерно-утюжная технология применяется для самостоятельного изготовления печатных плат в домашних условиях. Технология ЛУТ зародилась совсем недавно с появлением лазерных принтеров. Лазерные принтеры заправляются порошком – тонером. При большой температуре тонер плавиться и “оседает” на печатной поверхности. Именно этот принцип и заложен в ЛУТ. Поэтому, чтобы использовать эту технологию, нам нужен всего-навсего лазерный принтер и утюг.

Рисунок для платы так же можно сделать в программе Proteus. Но нужно иметь ввиду, что рисунок на текстолите отобразится зеркально, поэтому, в программе нужно поставить галочку «Mirror», чтобы рисунок сразу был отображен зеркально.

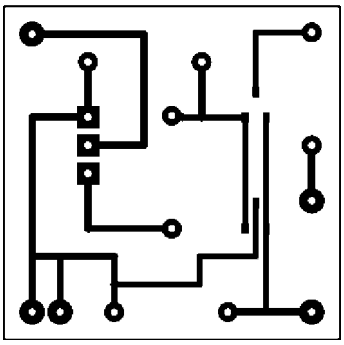


Рисунок 27 - Зеркальный рисунок для нанесения на текстолит

Бумага, на которой будет печататься рисунок, должна быть глянцевой, может подойти даже лист глянцевого журнала.

Для того чтобы подготовить кусок фольгированного текстолит для переноса рисунка на него, нужно самой мелкой шкуркой-микронкой зачистить поверхность до блеска. Как зашкурили нужно почистить и обезжирить спиртом.

После того, как мы подготовили нашу ошкуренную платку, кладем нашу бумажку рисунком вниз на платку. Включаем утюг и ставим его на максимальную температуру. После того, как он нагрелся, начинаем гладить бумажку. Прижимая утюг тщательно проглаживать все края около двух минут. Далее отклеиваем бумажку и рисунок должен быть перенесен на текстолит.

Теперь нужно вытравить плату, для этого нужна лимонная кислота, перекись водорода и соль. Наливаем все в какую-нибудь емкость, кладем туда платку и ждем минут 15. Затем нужно убрать тонер с дорожек. Далее лудим дорожки и плата готова.

Теперь нужно научиться управлять коэффициентом заполнения или скважностью ШИМ с Arduino. Для этого была на писана программа в среде Arduino IDE.

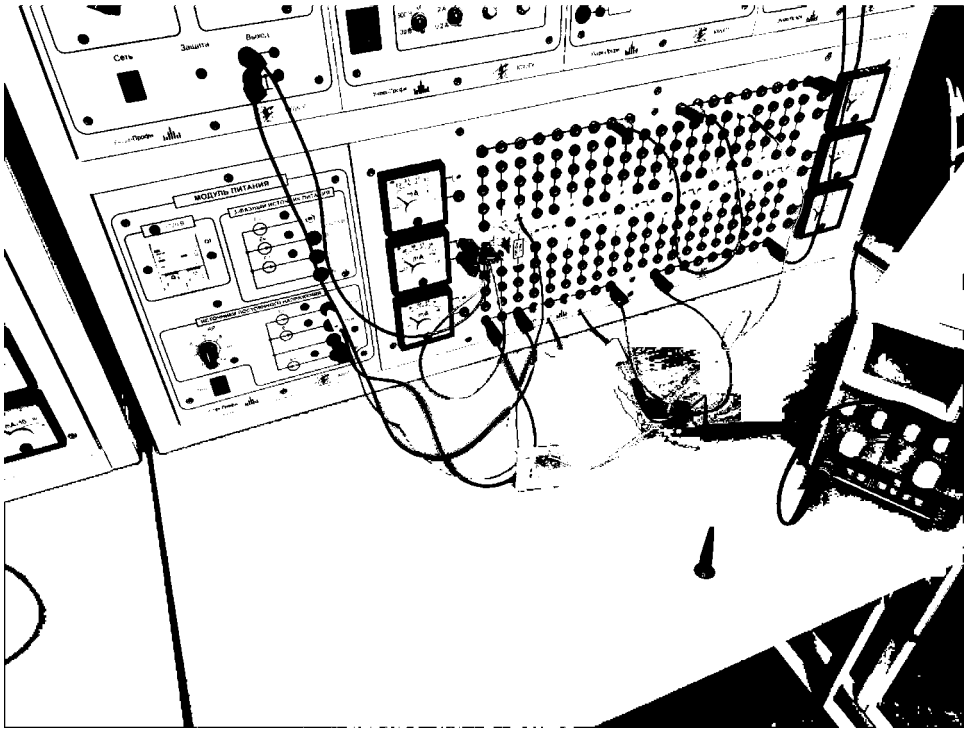


Рисунок 28 – Проверка платы на работоспособность

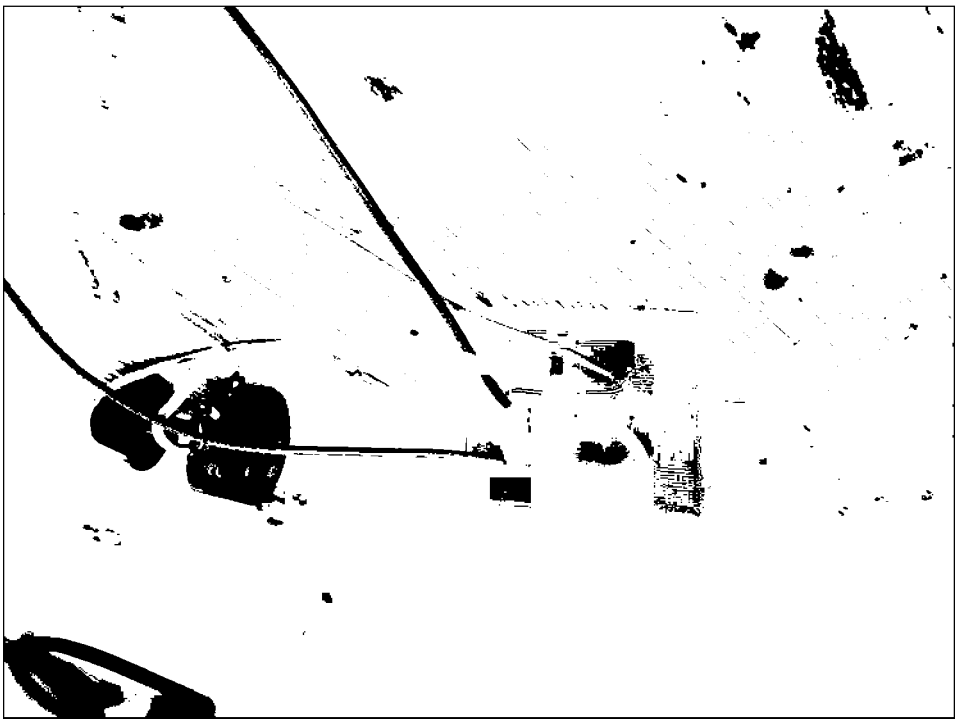


Рисунок 29 – Обратная сторона платы

С помощью платы было проведено несколько экспериментов по стабилизации напряжения на выходе с подключенной нагрузкой и без.

3.2 Широтно-импульсная модуляция

Алгоритм широтно-импульсной модуляции применяется для плавного изменения мощности на нагрузке, поступающей от источника питания. Например, с целью регулирования скорости вращения вала двигателя; плавности изменения яркости освещения или подсветки.

Чтобы понять принцип ШИМ воспользуемся следующим графиком. Разобьем ось времени на равные промежутки, называемые периодом *T*. Теперь, например половину периода мы будем замыкать ключ *K*. Когда ключ замкнут, к нагрузке *H* подается напряжение от источника питания Uип. Вторую часть полупериода ключа находится в закрытом состоянии. А потребитель останется без питания.

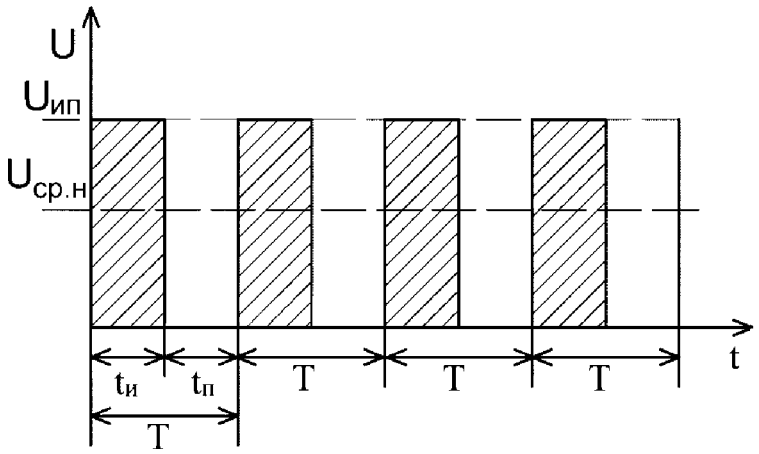


Рисунок 30 – График ШИМ

Время, в течение которого ключ замкнут, называется временем импульса tи. А время длительности разомкнутого ключа называют временем паузы tп. Если измерить напряжение на нагрузке, то оно будет равно половине *Uип*.

Среднее значение напряжения на нагрузке можно выразить следующей зависимостью:

$$U_{ср.н} = U_{ип}\, t_и / T. \qquad (1)$$

Отношение времени импульса *tu* к периоду *T* называют коэффициентом заполнения D. А величина, обратная ему называется скважностью:

$$S = 1/D = T/tи. \quad (2)$$

Удобнее пользоваться коэффициентом заполнения, который зачастую выражают в процентах. Когда транзистор полностью открыт на протяжении всего времени, то коэффициент заполнения D равен единице или 100 %.

Если D = 50 %, то это означает, что половину времени за период транзистор находится в открытом состоянии, а половину в закрытом. В таком случае форма сигнала называется меандр. Следовательно, изменяя коэффициент D от 0 до единицы или до 100 % можно изменять величину Uср.н от 0 до Uип:

$$Uср.н = Uип \cdot D. \quad (3)$$

А соответственно регулировать и величину подводимой мощности:

$$Pср.н = Pип \cdot D. \quad (4)$$

4. РАЗРАБОТКА АЛГОРИТМА И ПРОГРАММЫ

4.1 **Алгоритм**

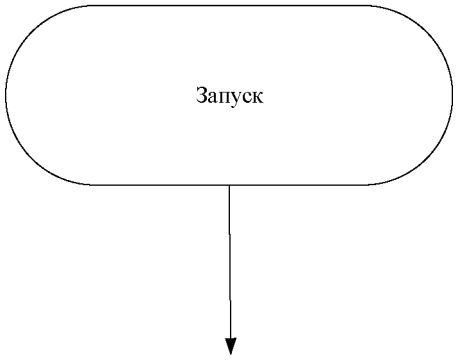


Рисунок 31 – Начало алгоритма

Программа начинается с запуска и переходит к выполнению первых команд.

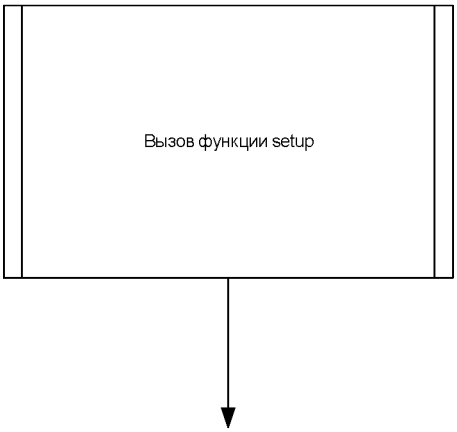


Рисунок 32 – Инициализация программы Программа

Программа начинается с установки частоты микроконтроллера, запуска таймера, инициализация дисплея, по дисплею пробегает звездочка.

Далее следует алгоритм основного цикла программы. В основном цикле происходит чтение данных с датчиков, вычисление величин опрос кнопок, вывод данных на дисплей, передача данных на ПК. Рассмотрим более детально работу основного цикла программы.

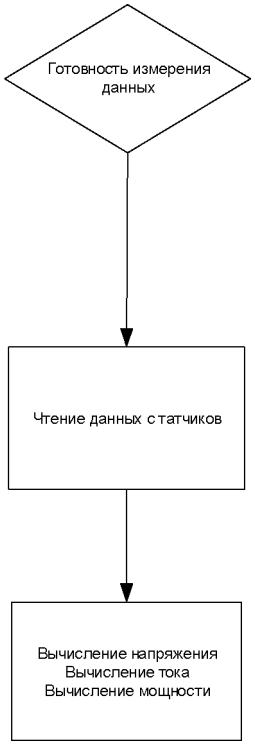


Рисунок 33

Проверка готовности измерения данных, затем чтение данных с датчиков и из этих данных высчитываются напряжение, ток и мощность.

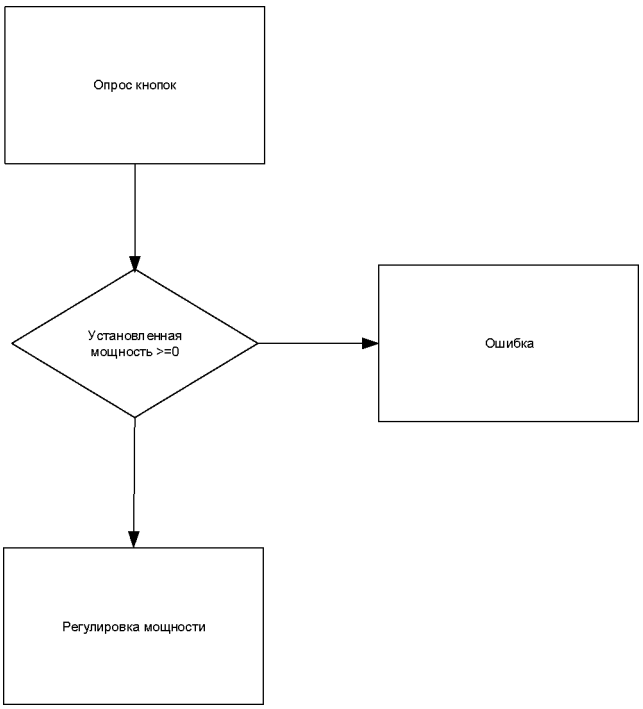


Рисунок 34 – Опрос кнопок

В данном фрагменте происходит опрос кнопок, считывается установленная мощность и сравнивается с нулем, если больше и равно нулю, то регулируется выходная мощность, если нет, то ошибка.

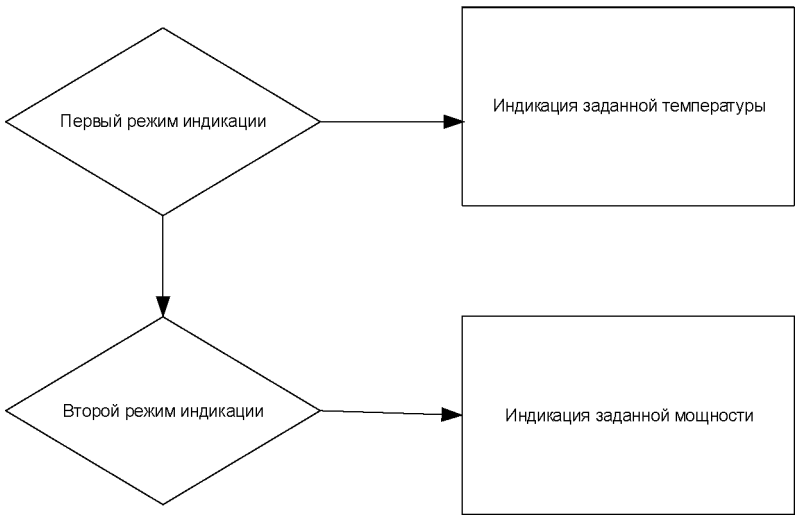


Рисунок 35 – Выбор режима индикации

Кнопками производится выбор режима индикации. Один режим показывает температуру, второй показывает мощность.

Рисунок 36 – Передача данных в порт

В итоге все данные с микроконтроллера отправляется в порт и дальше они отображаются в пользовательском интерфейсе.

4.2 Разработка исходного текста программы

```
#include <MsTimer2.h>
#include <avr/wdt.h>
#include <OneWire.h>
#include <LiquidCrystal.h>
#include <Button.h>
#include <EEPROM.h>

// адреса EEPROM
#define EEPROM_SET_TEMP_REF 5 // заданная температура
#define EEPROM_MAX_SET_POWER 10 // заданная максимальная мощность

// выводы
#define ERROR_LED_PIN 13  // вывод светодиода ОШИБКА _-_
#define VENT_PIN 10  // вывод вентилятор _-_

#define powerU  12.11     // напряжение источника питания
#define koeffU  0.01679 // масштабный коэффициент напряжения,
                        // koeffU = 1.1 / R1 * (R1 + R2) / 1024 = 1.067/820*(820 + 12000)/1024 =0.01629 В/ед. АЦП
#define koeffI  0.02148 // масштабный коэффициент тока
                        // koeffI =  1,067 / R8 / 1024 = 0.02084 А/ед. АЦП

#define koeffRegPwrInt 0.05 // интегральный коэффициент регулятора мощности
#define DEAD_TIME 8     // мертвое время ШИМ (* 62,5 нс)
#define koeffRegTmpInt 0.002 // интегральный коэффициент регулятора температуры
#define koeffRegTmpPr 0.5 // пропорциональный коэффициент регулятора температуры
#define koeffRegTmpDif 50. // дифференцирующий коэффициент регулятора температуры

#define tempOnVent 30.  // температура включения вентилятора
#define tempOffVent 27.  // температура выключения вентилятора

#define MAX_POWER 10.   // максимальная выходная мощность контроллера
#define INC_MAX_POWER 1.   // приращение максимальной выходной мощности при установке
#define MAX_SET_TEMP 30.   // максимальная заданная температура
#define MIN_SET_TEMP -10.   // минимальная заданная температура
#define INC_SET_TEMP 0.1   // приращение заданной температуры при установке

#define TIME_PRESS  40    // время удержания кнопок + и - для ускорения изменения параметра (* 20 мс)
#define INC_BUTT_TIME 10  // время изменения параметров при удержании кнопки (* 20 мс)

#define overHeatTemp 50.  // температура перегрева

#define MEASURE_PERIOD 10  // время периода измерения (* 2 мс)
#define MAX_PWM 255.      // максимальное значение ШИМ

#define MEASURE_MIN_TEMP  -40.  // минимальная температура измерения
#define MEASURE_MAX_TEMP  100.  // максимальная температура измерения
#define NUM_VER_TEMP_SEN  3     // число подтверждений ошибок датчиков температуры

float measureU; // измеренное напряжение на нагрузке, В
float measureI; // измеренный ток потребления регулятора, А
float measureP; // измеренная мощность на нагрузке, Вт
float measureTempRef;  // измеренная температура
float measureTempRad;  // измеренная температура

OneWire sensTempRef (16);  // датчик температуры в камере подключен к выводу 16 (A2)
OneWire sensTempRad (17);  // датчик температуры радиатора подключен к выводу 17 (A3)

LiquidCrystal disp(3, 2, 7, 6, 5, 4); // создаем объект дисплей

Button buttonPlus(8, 15); // кнопка "+"
Button buttonMinus(11, 15); // кнопка "-"
Button buttonChoice(12, 15); // кнопка "выбор"
```

Рисунок 37 – Запуск программы

При запуске программы объявляются библиотеки MsTimer2 позволяющая пользоваться возможностями внутреннего прерывания по таймеру от Timer2, avr/wdt для управления сторожевым таймером, OneWire для работы с интерфейсом 1-Wire, LiquidCrysta выводить информацию удобно

на экран, EEPROM.h для взаимодействия с памятью EEPROM. Далее происходит объявление констант, переменных и присвоение им начальных значений. Назначаются выходы для датчиков температуры, создается объект дисплей и объявляются выходы для кнопок.

Далее переходим к подпрограмме инициализации системы

```
void setup() {
 // установка ШИМ 8 разрядов, 62,5 кГц
  TCCR1A = TCCR1A & 0xe0 | 1;
  TCCR1B = TCCR1B & 0xe0 | 0x09;

  Serial.begin(19200);  // инициализируем последовательный порт, скорость 9600
  analogReference(INTERNAL); // опорное напряжение 1,1 В
  MsTimer2::set(2, timerInterupt); // прерывания по таймеру с периодом 2 мс
  MsTimer2::start();              // разрешение прерывания
  wdt_enable(WDTO_15MS); // разрешение работы сторожевого таймера, тайм-аут 15 мс

  pinMode(ERROR_LED_PIN, OUTPUT); // светодиод ОШИБКА
  pinMode(VENT_PIN, OUTPUT); // вентилятор
  digitalWrite(VENT_PIN, LOW);  // вентилятор

  // по индикатору пробегает звездочка
  disp.begin(8, 1);  // инициализация дисплея 1 строка, 8 символов
  disp.write('*');
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
}
```

Рисунок 38 – Функция void setup()

После объявления библиотек и переменных, назначаем режим работы системы и выполнение настроечных команд. В функции void setup() происходит настройка частоты микроконтроллера, затем запуск работы последовательного порта (UART) и задаем ему скорость обмена данными 9600 бит/сек. Далее устанавливается время периода прерывания в мс. Устанавливаются выходы для светодиода и для вентилятора. И происходит инициализация дисплея, на дисплее отображается пробегающий символ «*».

Далее переходим к выполнению основного цикла программы.

```
if (flagReady == true) {
  flagReady= false;

  //---------------- вычисление измеренных значений
  // вычисление напряжения
  measureU = powerU - (float)(averageU / MEASURE_PERIOD) * koeffU;
  if (measureU < 0) measureU=0;

  // вычисление тока
  measureI = (float)(averageI / MEASURE_PERIOD) * koeffI;

  // вычисление мощности
  measureP = powerU * measureI;
```

Рисунок 39 – Расчет значений

Если есть признак готовности данных, то производятся вычисления напряжения, тока и мощности.

```
if ( setPower >= 0) {
  regPwrInt = regPwrInt + (setPower - measureP) * koeffRegPwrInt;
  if (regPwrInt < 0) regPwrInt=0;     // ограничение снизу
  if (regPwrInt > MAX_PWM) regPwrInt=MAX_PWM; // ограничение сверху

  unsigned int  pwm = (unsigned int)regPwrInt;  // перевод в ШИМ
  if (pwm < DEAD_TIME) pwm=0;
  if (pwm > (MAX_PWM - DEAD_TIME)) pwm=MAX_PWM;
  analogWrite(9, pwm); // ШИМ
}
else {            // выключение
  regPwrInt=0;
  analogWrite(9, 0); // ШИМ
}
```

Рисунок 40 – Регулирование мощности

Затем, если установленная мощность больше нуля, происходит регулировка мощности и значение переводится в ШИМ сигнал. Затем полученный сигнал подается с 9-го пина платы. Иначе все выключается.

```
//--------------------------- управление индикацией ------------------

noDispPar = false; // отображать параметры на дисплее
```

Рисунок 40 – управление индикацией

Данной строчкой происходит управление индикацией, отображаются рассчитанные значения.

```
// отсчет времени удержания кнопок + и -
if (buttonPlus.flagPress == true) {
  butPlusTime++;
  if (butPlusTime == 0) butPlusTime= 255;
}
else  butPlusTime= 0;
if (buttonMinus.flagPress == true) {
  butMinusTime++;
  if (butMinusTime == 0) butMinusTime= 255;
}
else  butMinusTime= 0;
```

Рисунок 41 – Отсчет времени удержания кнопок «+» и «-»

Затем происходит отсчет времени удержания «+» и «-» 800 мс, для ускорения изменения параметров. «+» - повышение температуры, «-» - понижение температуры.

```
if (buttonChoice.flagPress == false) {   // кнопка "выбор" не нажата
  if (wasChoice == false) { // ---------------------------------- кнопка "выбор" не нажата и была не нажата
    wasPress=false; // сброс ризнака нажатия кнопок + или -

    // режим индикации
    if ( buttonPlus.flagClick == true ) {
      buttonPlus.flagClick = false;
      indMode ++;
      if (indMode > 2) indMode=0;
      dispParameters();
      noDispPar = true; // не отображать параметры в этом цикле
    }
    if ( buttonMinus.flagClick == true ) {
      buttonMinus.flagClick = false;
      indMode --;
      if (indMode > 128) indMode=2;
      dispParameters();
      noDispPar = true; // не отображать параметры в этом цикле
    }
  }
  else  { // ------------------------------------------------- кнопка "выбор" не нажата, а была нажата (отпустили)
    wasChoice = false;

    // запись параметра в EEPROM
    if (wasPress == true) {
```

Рисунок 42 - Режим индикации

Эта часть кода отвечает за выбор режима индикации. Сначала проверяется была ли нажата кнопка выбора. Если кнопка нажата, то происходит смена режима индикации значения.

```
cycle20mcCount++;   // счетчик циклов 20 мс
if (cycle20mcCount >= 50) cycle20mcCount= 0;  // время цикла 1 сек
```

Рисунок 43 - Выполнение распределенных операций в цикле 1 с.

Цикл состоит из мини циклов по 20 мс. В общем 50 мини циклов и весь цикл занимает 1с.

```
if (buttonPlus.flagClick == true) {   // нажали кнопку +
  buttonPlus.flagClick= false;
  wasPress= true; // признак кнопка нажималась
  incrParameter();
}

if (buttonMinus.flagClick == true) {   // нажали кнопку -
  buttonMinus.flagClick= false;
  wasPress= true; // признак кнопка нажималась
  decrParameter();
}

// изменение параметра при удержании кнопки +
if (butPlusTime >  TIME_PRESS) {
  incButtTime++;
  if (incButtTime > INC_BUTT_TIME) {
    incButtTime= 0;
    incrParameter();
  }
}
else incButtTime= 0;

// изменение параметра при удержании кнопки -
if (butMinusTime >  TIME_PRESS) {
  decButtTime++;
  if (decButtTime > INC_BUTT_TIME) {
    decButtTime= 0;
    decrParameter();
  }
}
else decButtTime= 0;
```

Рисунок 44 – Опрос кнопок «+» и «-»

Опрос кнопок «+» и «-». Если была нажата кнопка, то заданное значение увеличивается или уменьшается в зависимости от нажатой кнопки.

Если кнопка остается нажатой дольше чем установленное время нажатой кнопки, то есть 800 мс, тогда каждые 200 мс будет увеличиваться или уменьшаться заданное значение в зависимости от кнопки.

```
disp.begin(8, 1);  // инициализация дисплея 1 строка, 8 символов
noDispPar = true; // не отображать параметры

if (indMode == 0) {   // индикация заданной температуры
  disp.print("T= ");
  disp.print(setTempRef,1);
}
if (indMode == 1) {   // индикация заданной мощности
  disp.print("P< ");
  disp.print((int)maxSetPower);
  disp.print(" W");
}
if (indMode == 2) {   // индикация температуры перегрева
  disp.print("< ");
  disp.print(overHeatTemp,1);
  disp.print(" C");
}
```

Рисунок 45 - Режимы индикации

На данном фрагменте показаны режимы индикации на дисплее. Первый отвечает за отображение заданной температуры. Второй отвечает за отображение заданной мощности и третий отвечает за индикацию температуры перегрева.

```
if (cycle20mcCount == 5) {
  //--------------------------- интервал 5, инициализация измерения температуры
  sensTempRef.reset();  // сброс шины 1-Wire
  sensTempRef.write(0xCC, 1); // пропуск ROM
  sensTempRef.write(0x44, 1); // инициализация измерения
  // датчик радиатора
  sensTempRad.reset();  // сброс шины 1-Wire
  sensTempRad.write(0xCC, 1); // пропуск ROM
  sensTempRad.write(0x44, 1); // инициализация измерения
}
```

Рисунок 46 - Инициализация датчика

На данном интервале происходит инициализация датчиков температуры. Сначала сбрасывается шина 1-Wire, затем пропуск ROM и инициализация измерения.

```
if (cycle20mcCount == 10) {
  //--------------------------- интервал 10, передача информации на компьютер
  Serial.print("U="); Serial.print(measureU, 2);  // напряжение
  Serial.print(" I="); Serial.print(measureI, 2);  // ток
  Serial.print(" P="); Serial.print(measureP, 2);  // мощность
  Serial.print(" p="); Serial.print(regPwrInt, 2);  // интегральное звено регулятора мощности
  Serial.print(" t="); Serial.print(measureTempRef, 2);  // температура в камере
  Serial.print(" t="); Serial.print(measureTempRad, 2);  // температура радиатора
}
```

Рисунок 47 - Передача информации на компьютер

В данном фрагменте рассчитанные значения и информация с датчиков температуры отправляется в порт, чтобы потом эти значения можно было вывести в пользовательском интерфейсе на ПК.

```
  if (cycle20mcCount == 45) {
    //--------------------------- интервал 45, чтение датчика температуры
    sensTempRef.reset();  // сброс шины 1-Wire
    sensTempRef.write(0xCC, 1); // пропуск ROM
    sensTempRef.write(0xBE, 1); // команда чтения памяти датчика
    sensTempRef.read_bytes(bufData, 9);  // чтение памяти датчика, 9 байтов
```

Рисунок 48 - Опрос датчика температуры

На данном интервале происходит чтение датчика температуры. Сначала командой сброса сбрасывается шина 1-Wire затем пропускается ROM и следующей командой читается память датчика.

```
if (cycle20mcCount == 46) {
  //--------------------------- интервал 46, чтение датчика температуры радиатора
  sensTempRad.reset();  // сброс шины 1-Wire
  sensTempRad.write(0xCC, 1); // пропуск ROM
  sensTempRad.write(0xBE, 1); // команда чтения памяти датчика
  sensTempRad.read_bytes(bufData, 9);  // чтение памяти датчика, 9 байтов
```

Рисунок49 - Чтение данных с памяти датчика температуры

Здесь происходить опрос датчика температуры аналогично предыдущему.

```
if (cycle20mcCount == 47) {
  //--------------------------- интервал 47, контроль температуры радиатора

  // управление вентилятором
  pinMode(VENT_PIN, OUTPUT);
  if ( measureTempRad > tempOnVent ) digitalWrite(VENT_PIN, HIGH);  // включение вентилятора
  if ( measureTempRad < tempOffVent ) digitalWrite(VENT_PIN, LOW);  // выключение вентилятора

  // состояние вентилятора
  if (digitalRead(VENT_PIN) == HIGH) errors |= 2;
  else errors &= B11111101; // выключен

  // контроль перегрева
  if ( measureTempRad > overHeatTemp ) {
    fatalError= true; // признак фатальной ошибки
    errors |= 4;  // признак ошибки перегрев
  }
  else errors &= B11111011; // ошибки нет

  // признак фатальной ошибки
  if (fatalError == true) errors |= 8;
  else errors &= B11110111; // нет ошибки
}
```

Рисунок 50 - Контроль температуры радиатора Управление вентилятором

Если измеренная температура больше температуры включения вентилятора, то подается питания на пин вентилятора и вентилятор охлаждает радиатор.

```
if (cycle20mcCount == 48) {
  //--------------------------- интервал 48, регулятор температуры в камере

  if ( fatalError == false ) {
    // фатальной ошибки нет
    flashErrorLed= false; // светодиод ОШИБКА не мигает
    regTmpErr= measureTempRef - setTempRef;  // вычисление ошибки рассогласования

    regTmpInt= regTmpInt + regTmpErr * koeffRegTmpInt; // интегральная часть
    if ( regTmpInt > maxSetPower) regTmpInt= maxSetPower; // ограничение сверху
    if ( regTmpInt < 0 ) regTmpInt= 0;                    // ограничение снизу

    regTmpPr= regTmpErr * koeffRegTmpPr; // пропорциональная часть
    regTmpDif= (regTmpErr - regTmpErrPrev) * koeffRegTmpDif;  // дифференцирующая часть
    regTmpErrPrev= regTmpErr; // перегрузка предыдущей ошибки

    setPower= regTmpInt + regTmpPr + regTmpDif; // сумма составляющих
    if ( setPower > maxSetPower) setPower= maxSetPower; // ограничение сверху
    if ( setPower < 0 ) setPower= 0;                    // ограничение снизу
  }
  else {
    // фатальная ошибка, выключение
    setPower= -200.;  // аварийное выключение регулятора мощности
    regTmpInt= 0;     // сброс интегрального звена регулятора температуры
    flashErrorLed= true; // светодиод ОШИБКА мигает
  }
}

if (cycle20mcCount == 49) {
  //--------------------------- интервал 49, отображение данных на дисплее
  if (noDispPar == false) dispParameters();
}
```

Рисунок 51 - Регулятор температуры

Проверка на наличие ошибок затем вычисляется рассогласование ошибки

5 РАЗРАБОТКА ИНТЕРФЕЙСА ПОЛЬЗОВАТЕЛЯ НА ПК

Разработка интерфейса пользователя производится в программной среде Microsoft Visual Studio на языке C# windows forms. Microsoft Visual Studio — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Window Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.[6] Для работы с микроконтроллером через последовательный порт следует использовать ресурсы класса SerialPort, который в свою очередь предоставляет средства для синхронного и управляемого событиями ввода – вывода, для доступа к состоянию подключения – отключения устройства. Для использования объектов класса SerialPort в шапке программы пропишем строку using System.IO.Ports; Пространство имен System.IO.Ports содержит классы для управления последовательными портами.

5.1 **Создание стартового окна**

Визуальный вид стартового окна программы представлен на рисунке 52

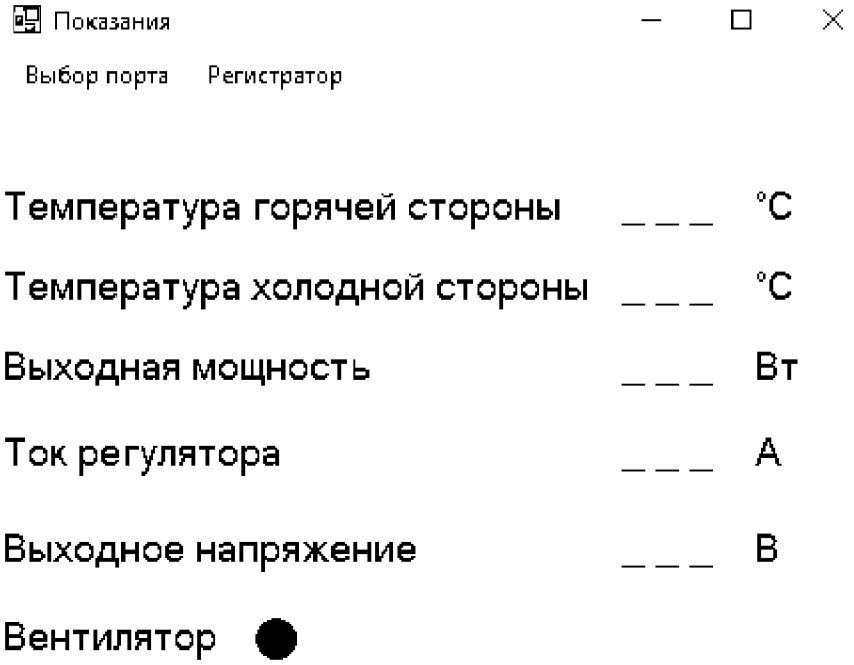


Рисунок 52 – Стартовое окно программы

С использованием компонентов «Label», «MenuStrip» составим визуальное представление стартового окна приложения, в компонентах «Label» прописывается текст «Температура горячей стороны» и остальных параметров так же «Label» используется, чтобы отображать значение переданных параметров с микроконтроллера. В компоненте «MenuStrip» находятся вкладки: «Выбор порта», после нажатие на которую открывается окно «COM порт». Так же там присутствует вкладка «Регистратор», которая открывает окно регистратора.

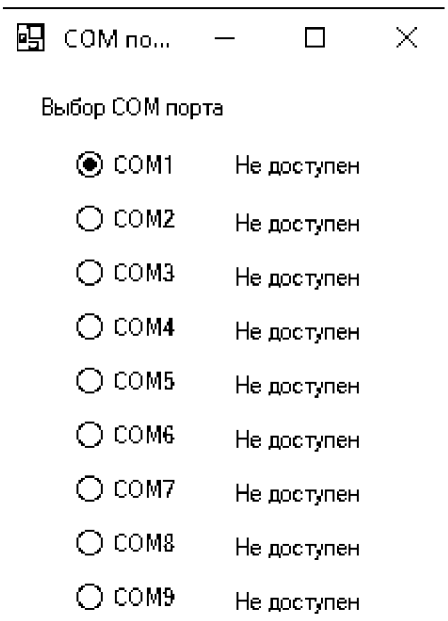


Рисунок 53 – Окно выбора COM порта

Окно «COM порт» состоит и компонентов «Label» и «RadioButton». С помощью компонента «RadioButton» осуществляется выбор COM порта. На против каждого COM порта показано его состояние.

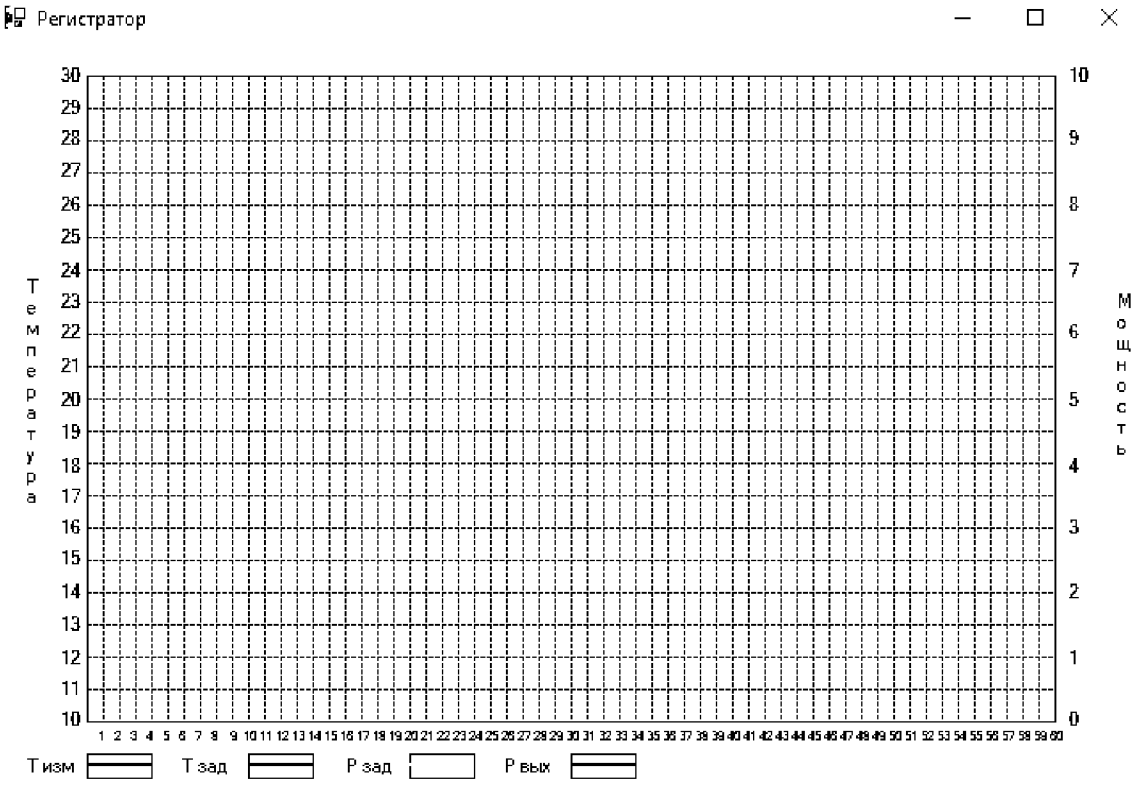


Рисунок 54 – Окно регистратора

Окно регистратора так же состоит из «Label» и сетки которая чертится при помощи библиотеки using System.Drawing и её класса Graphics. Здесь наглядно в графиках отображаются значения температур и мощности.

6 БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

6.1 Техника безопасности при эксплуатации установки

При эксплуатации установки необходимо перед ее включением проверить:

– Фиксацию всех элементов установки, чтобы они жестко прилегали к поверхности;

– Обратить внимание на трубки, нет ли сколов или трещин, как они зафиксированы на местах соединений;

– На колбе проверяются ее целостность бочка, все входные и выходные отверстия, а также место соединения колбы с помпой;

– Проверить вентилятор, нет ли в нем инородных тел

– Прилегание тормодатчиков к поверхности ТЭМ, это нужно для более точного снятия показаний;

– Подключение всех проводов к элементам установки и к клемной коробке;

– Проверить блок питание, переведены ли все переключатели и тумблеры в нужное положение;

– Осмотреть вилку и провод питающий блок питания, на наличее пробоев и других неисправностей.

После предварительной проверки подключается блок питания к сети.

Поскольку на установку требуется два блока питания, подключаем оба, последовательность подключения не важна.

Затем включается тот блок питания, который питает водяной насос. Этим самым проверяем наличие потока жидкости в системе. Это необходимо для того чтобы элемент не перегрелся после его включения.

Подключается второй блок питания который питает сам ТЭМ и вентилятор. Визуально осматриваем установку, если поток воды движется по

трубкам, а вентилятор в рабочем состоянии, то можно подавать напряжение на элемент Пельтье.

После использования установки необходимо:

- Отключить подаваемое напряжение с блока питания на ТЭМ;
- Отключить второй блок питания который питает элемент Пельтье;
- Отключить первый блок питания который питает насос и вентилятор;
- Визуально осмотреть установку на наличие подтеков жидкости.

6.1.1 При коротком замыкании электропроводки установки

При коротком замыкании электропроводки необходимо выключить первый и второй блоки питания с кнопки. После чего вытащить питающие провода блоков питания из розетки. И устранять неисправность в установке.

Если вышел из строя один из блоков питания необходимо отключить питание помещения с тумблера. После отключения в резиновой перчатке отключить провод блока питания из розетки. Далее устраняется неисправность либо блок заменяется на другой.

При неисправности розетки так же выключается питание помещения с тумблера.

6.2 Экологичность

Достоинствами элемента Пельтье являются небольшие размеры, отсутствие шума, каких-либо движущихся частей, а также газов и жидкостей. При смене направления тока возможно, как охлаждение, так и нагревание — это даёт возможность термостатирования при температуре окружающей среды как выше, так и ниже установленного порога.

В системах Пельтье нет хладагентов, которые легко воспламеняются, разрушают озоновый слой и вызывают парниковый эффект. Поэтому ТЭМ полностью экологичен.

6.3 Правила поведения при чрезвычайной ситуации на установке

– При протекании либо срыве одного из шлангов с охлаждающей жидкостью, необходимо отключить питание установки во избежание попадания воды на контакты;

– При обрыве одного из проводов системы отключаем блоки питания, исправляем неисправность;

– При задымлении из любой части установки отключаем питание, ищем неисправность, устраняем ее;

– При показаниях находящихся в зоне критичности с термодатчика, отключаем питание, заменяем элемент Пельтье.

6.3.1 При пожаре

Если на установке произошло возгорание ее элементов или проводки, необходимо обесточить помещение с тумблера. При малом возгорании накрываем место горения одеждой или плотной тканью. При большом возгорании используем огнетушитель.

Если огонь перекинулся на другие объекты, и нет возможности потушить пожар, необходимо покинуть помещение. Вызвать пожарных.

6.3.2 Поломка элемента Пельтье

Обнаружить поломку или причину выхода из строя модуля Пельтье очень просто. Это помогут увидеть с помощью темпродатчиков. Если температура холодной стороны начинает расти в плюсовую сторону, первым делом проверяем систему охлаждения. Если система в порядке и полностью работоспособна (поток воды движется и вентилятор работает), то высока вероятность того, что элемент вышел из строя.

6.3.3 При обнаружении других неисправностей

При обнаружении каких-либо других неисправностей, требуется отключить питание на установке. После этого обнаружить и устранить неисправности.

7 РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ

Техническое задания является одной из самой важной стадии разработки проекта.

Техническое задания представлено в Приложении А.

Этапы, которые должно включать техническое задания:

1. Общие сведения. Должно включать в себя общая информация о разработки и сроки выполнения работы.

2. Назначения и цели развития, включает в себя:

2.1 Назначение системы. Подробную информацию.

2.2 Цели развития системы. Цели, которые должны быть выполнить в ходе выполнения работы.

3. Характеристики объекта. Приведены характеристики объекта или прилагаются документы к ним, которые содержат эти данные.

4. Требования к системе. Который должен включать в себя подразделы:

4.1 Требования к численности и квалификации персонала системы.

4.2 Требования к надежности.

4.3 Требования к безопасности.

4.4 Требования к эксплуатации, техническому обслуживанию, ремонту и хранение компонентов системы.

5. Состав и содержание работ по развитию системы. В соответствии с ГОСТ 2.601 приводится информация о стадиях работы, что было проведено.

6. Требования к документации. Производится перечень чертежей и схем.

7. Источники разработки. Перечисление документов и материалов основание которых должно подходить для разработки ТЗ и которые должны использоваться для системы.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационного проекта была разработана автоматизированная система регулирования температуры модуля Пельтье на базе микроконтроллера.

Также была разработана программа для микроконтроллера, спроектирована и разработана плата для балансировки напряжения, а также пользовательский интерфейс для ПК.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Datasheet. [Электронный ресурс]: © 2016 Atmel ATmega328/P. Режим доступа: https://static.chipdip.ru/lib/549/DOC001549488.pdf (Дата обращения: 20.03.2020);

2. Уроки программирования arduino. [Электронный ресурс]: Оборудование технологии разработки блог технической поддержки моих разработок. Режим доступа: http://mypractic.ru/uroki-programmirovaniya-arduino-navigaciya-po-urokam

3. Техническая документация ATmega328. [Электронный ресурс]: Техническая документация к электронным компонентам на русском языке. Режим доступа: https://rudatasheet.ru/datasheets/atmega328/

4. Техническая документация TES1-12706. [Электронный ресурс]: Hebei I.T. (Shanghai) Co., Ltd. Режим доступа: https://www.peltiermodules.com/peltier.datasheet/TEC1-12706.pdf

5. Белов А.В. Микроконтроллеры AVR. От азов программирования до создания практических устройств (2016).

6. Руководство по эксплуатации. [Электронный ресурс]: © 2020 ООО Научно-производственное предприятие «Учебная техника – Профи». Режим доступа: http://labstand.ru/catalog/tau_ (Дата обращения - 17.03.2020);

7. Техническая документация Dallas 18B20. [Электронный ресурс]: DS18B20 Programmable Resolution 1-Wire Digital Thermometer. Режим доступа: https://static.chipdip.ru/lib/242/DOC004242266.pdf (Дата обращения - 17.03.2020);

8. Осциллографы цифровые. [Электронный ресурс]: © 2015 руководство по эксплуатации. 2015-73с.;

9. IRF3205 MOSFET - описание производителя. [Электронный ресурс]: Справочник MOSFET. IRF3205. Режим доступа: https://alltransistors.com/ru/mosfet/transistor.php?transistor=2363

10. Техническая документация транзистор BC817-40. [Электронный ресурс]: RadioLibrary справочник радиолюбителя. Режим доступа: https://www.radiolibrary.ru/reference/transistor-imp/bc817-40.html

11. Техническая документация KLS7-TS6601. [Электронный ресурс]: Switch KLS7-TS6601 6x6 Tact Switch Series. Режим доступа: https://static.chipdip.ru/lib/551/DOC001551938.pdf

ПРИЛОЖЕНИЕ А

Техническое задание на разработку

Техническое задание разработано согласно с ГОСТ 19.201-78

4.1 **Общие сведения**

Разрабатываемое ТЗ относится к системе автоматического управления термоэлектрическим элементом Пельтье

Плановые сроки начала работы по развитию систем автоматического управления: 10.10.2019

Плановые сроки окончание работы по развитию системы автоматического управления: 30.06.2020

4.2 **Назначение и цели развития системы**

4.2.1 Система автоматического управления термоэлектрическим модулем Пельтье предназначена для:

Регулирование температуры холодной стороны по программе, а регулирование температуры горячей стороны стабилизацией температуры.

4.2.2 Цели развития системы

Разработка программного обеспечения микроконтроллера;

Разработка печатной платы.

Разработка пользовательского интерфейса

4.2.3 Характеристики объекта

Объектом управления в данной работе является термоэлектрический модуль Пельтье.

Холодопроизводительность 50 Вт;

Разность температур 66 °C;

Максимальный ток 14.4 В;

Сопротивление 1.98 Ом ;

Длина – 40 мм;

Ширина – 40 мм;

Высота – 3.8 мм;

4.3 **Требования к системе**

4.3.1 Требования к численности и квалификации персонала системы

Выполнение любых действий должно руководствоваться под действием специального персонала.

4.3.2 Требования к надежности

Стенд должен эксплуатироваться при номинальном давлении эксплуатации $P_{ном}$= 2 МПа или меньших значениях и отсутствии пиковых давлений, превышающих 1,3 $P_{ном}$.

4.3.3 Требования к безопасности

Эксплуатация стенда должна производиться в соответствии с требованиями пожарной безопасности и требованиями ГОСТ Р 52543-2006 «ГИДРОПРИВОДЫ ОБЪЕМНЫЕ. Требования безопасности» и ГОСТ 12.2.003.

4.3.4 Требования к эксплуатации, техническому обслуживанию, ремонту и хранение компонентов системы

К обслуживанию стенда допускается персонал после ознакомления с настоящим руководством по эксплуатации.

Запрещается производить ремонтные работы и монтаж/демонтаж грузов при работающем стенде.

При работе стенда следует находится на удалении (минимум 0,5м) от подвижных частей, запрещается нарушать границы заграждения 34 (рис. 2).

При работе со стендом запрещается прикасаться к подвижным частям, осям, отверстиям, зазорам, гидроцилиндрам.

Запрещается производить монтажные работы при наличии давления в гидросистеме, или присутствии массовой нагрузки на балке 36 (рис. 2).

4.5 **Состав и содержание работ по развитию системы**

В соответствии с ГОСТ 2.601 в таблице приведены стадии и этапы работы проекта.

Продолжение приложение А

Стадии	Этап работы
Разработка системы управления.	Разработка структурной схемы
Проектирование печатной платы.	Разработка принципиальной электрической схемы системы.

4.6 **Требования к документации**

Структурная схема автоматической системы;

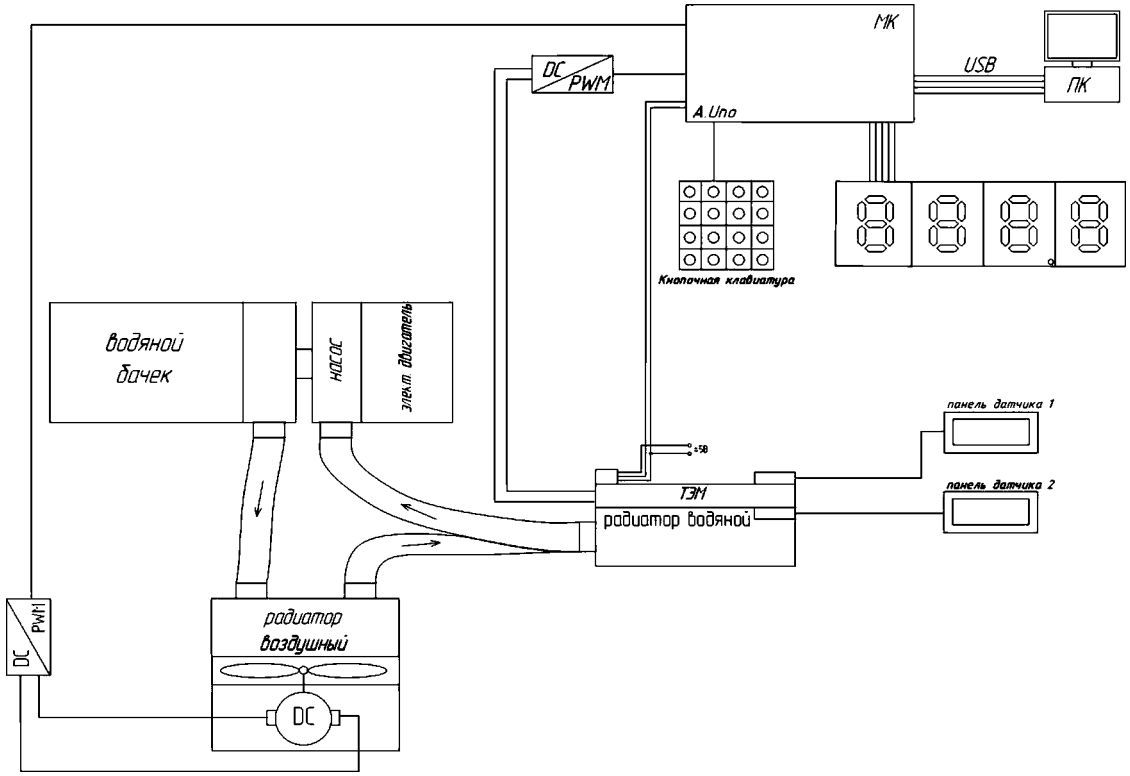
Принципиальная электрическая схема системы;

4.7 **Источники разработки**

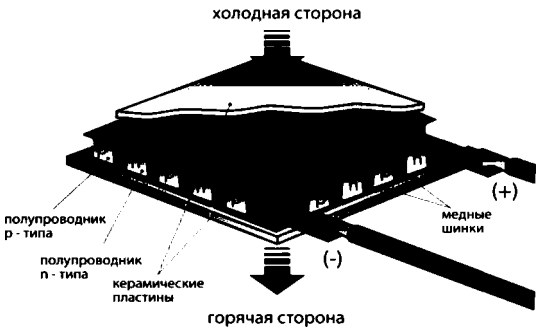
Полный перечень используемой информацией приведено в разделе «БИБЛИОГРАФИЧЕСКИЙ СПИСОК»

КП.164004.15.03.04 СХ

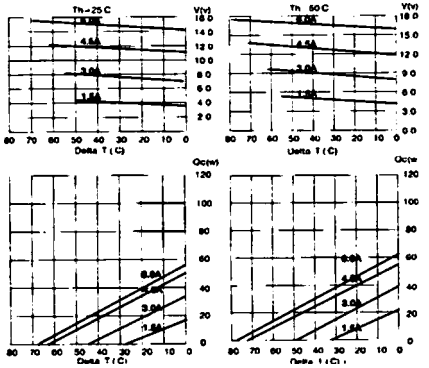
Общая структурная схема АСУ ТП



Структура модуля Пельтье



Технические характеристики TEC1-12706



| | | |

КП.164004.15.03.04 СХ

Схема принципиальная электрическая

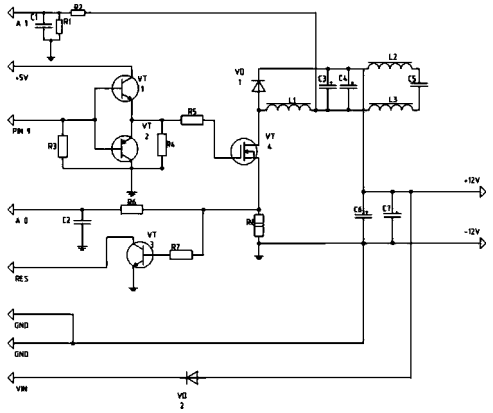
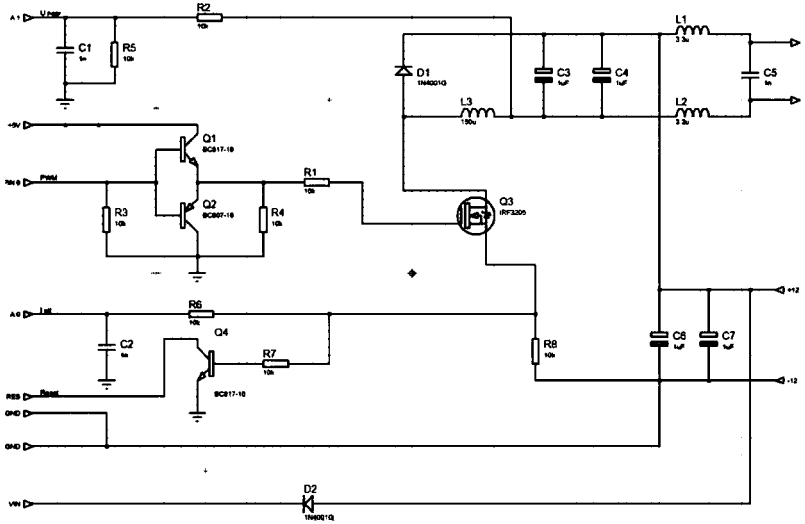
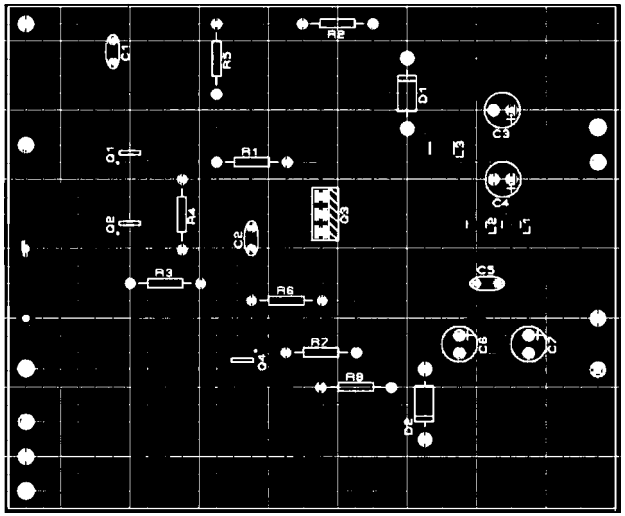


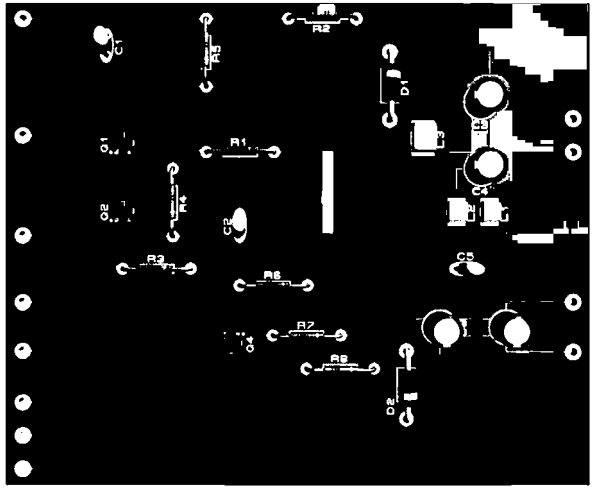
Схема принципиальная электрическая в программе Proteus



Печатная плата



3D Модель платы



Поз. Обозн.	Наименование	Кол.	Примечание
	Резисторы		
R6	С1-4-0.25-0.05 Ом ± 5	1	
R5	С1-4-0.25-10 Ом ± 5	1	
R4, R6, R7	С1-4-0.25-1 кОм ± 5	3	
R3	С1-4-0.25-4.3 кОм ± 5	1	
R2	С1-4-0.25-12 кОм ± 5	1	
R1	С1-4-0.25-820 Ом ± 5	1	
	Дроссели		
L2, L3	3 мкГн 5 А	2	
L1	150 мкГн 8 А	1	
	Конденсаторы		
C4, C7	К25 - 470 мкФ - 25 В ± 10%	2	
C3, C6	К23 - 4.7 мкФ - 25 В ± 10%	2	
C2, C5	К50-9нкФ-100В ± 10%	2	
C1	К41-0 1нкФ-10В ± 10%	1	
	Транзисторы		
VT4	IRF3205 N-канал 55В 110А	1	
VT2	BC807 - PNP 45В 0.5А 0.31Вт	1	
VT1, VT3	BC817 - NPN 45В 0.5А 0.31Вт	2	
	Диоды		
VD2	SB340 - 5А 250А 40В	1	
VD1	S1M - 1А 1000В	1	

КП.164.004.15.03.04 СХ

ПРОЕКТИРОВАНИЕ ПЕЧАТНОЙ ПЛАТЫ

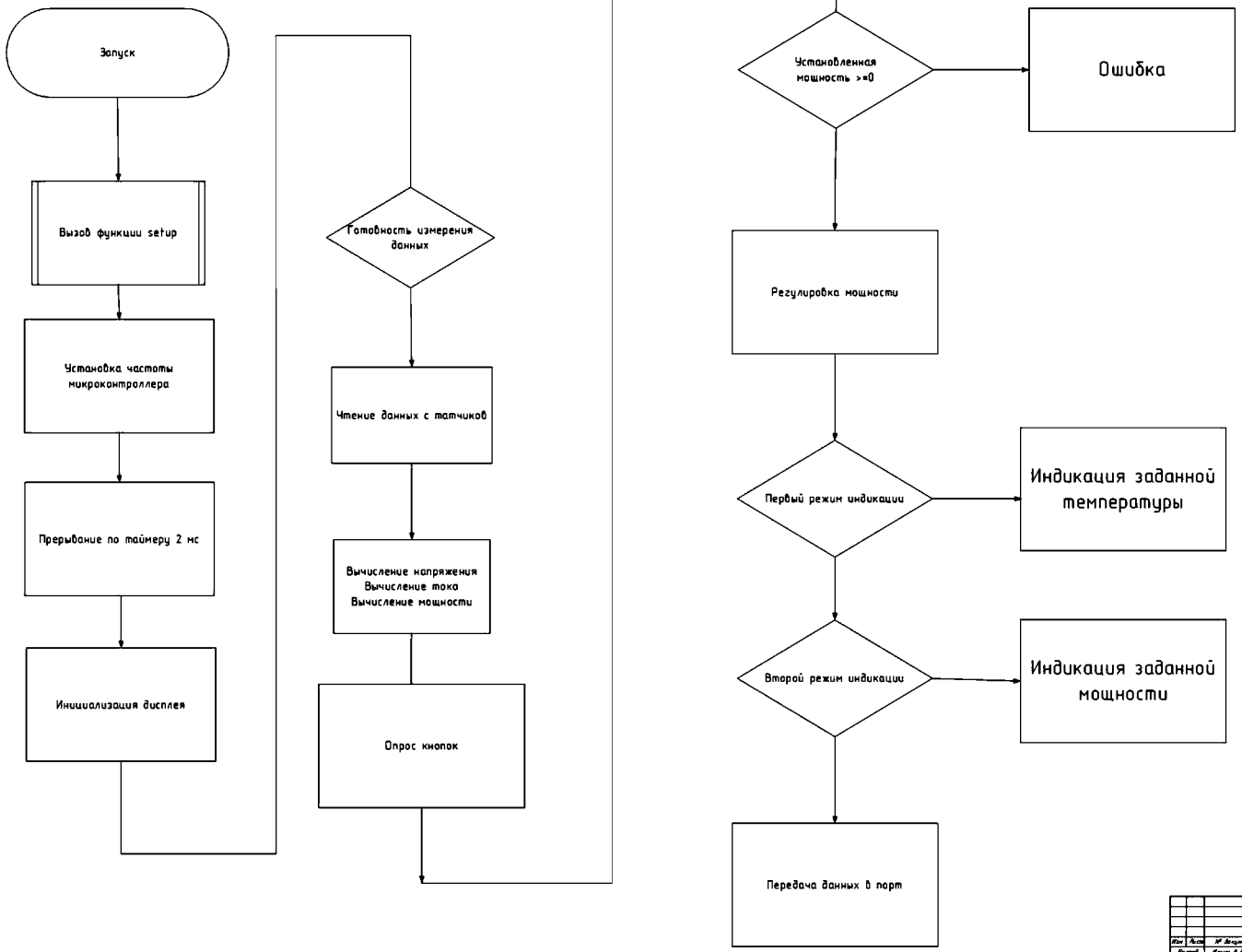
Литера: — Масса: — Масштаб: 1:1

Лист 2 Листов 4

АмГУ Кафедра АПИиЭ

КП.164004.15.03.04 СХ

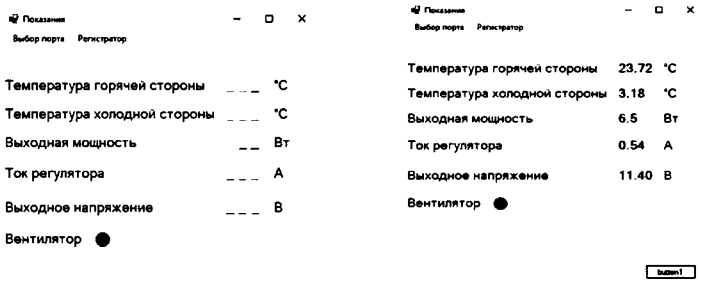
Алгоритм программы



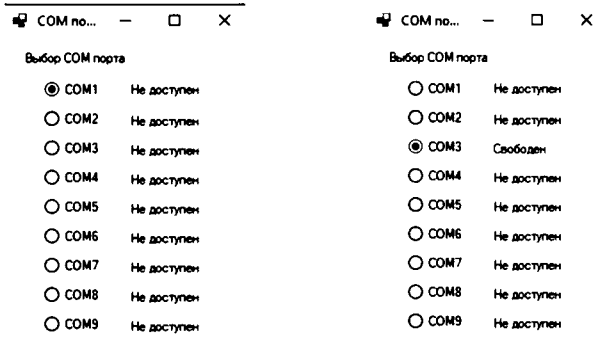
Изм.	Лист	№ докум.	Подп.	Дата	КП.164004.15.03.04 СХ		
Разраб.		Кощин А.А.			Алгоритм программы	Литера: у	Масса:	Масштаб: 1:1
Провер.		Скрыпник О.В.				Лист 1	Листов 6	
Т.контр.		Скрыпник О.В.			Система управления энергосбережением частотно-регулируемого электропривода	АлтГУ Кафедра АПУиЭ		
Н.контр.		Скрыпник О.В.						
Утвердил		Скрыпник О.В.						

КП 164004 15 03 04 СХ

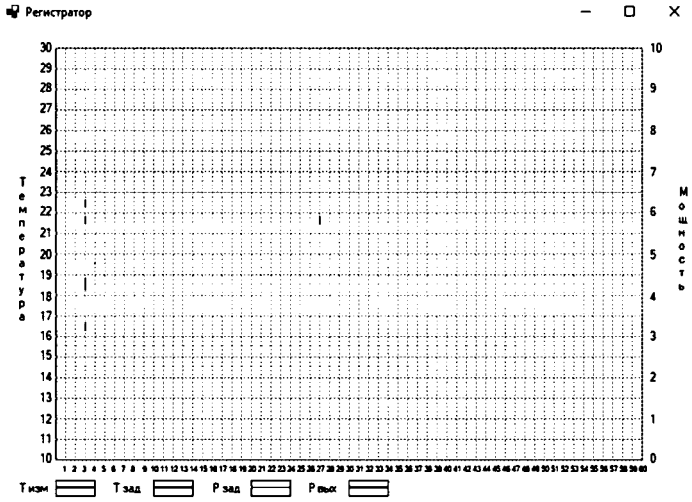
Главное окно



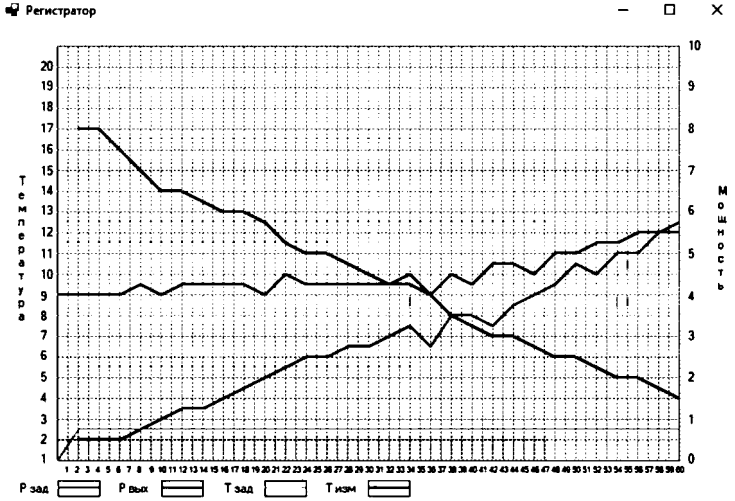
Окно выбора COM порта



Окно регистратора в отключенном состоянии



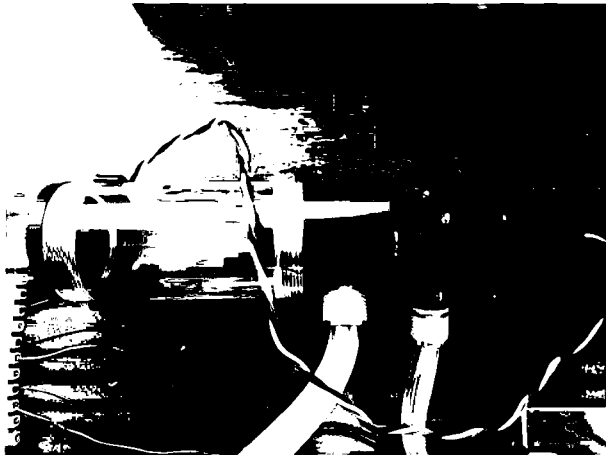
Окно регистратора в подключенном состоянии



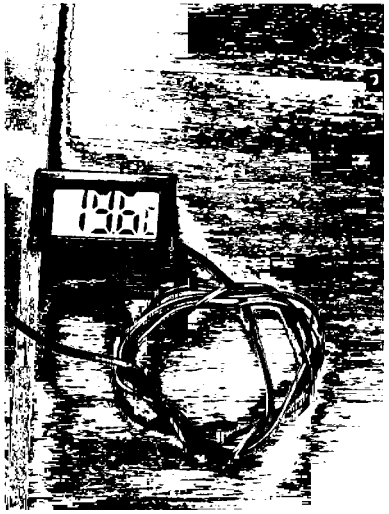
					КП.164004.15.03.04 СХ	
Изм.	Лист	№ докум.	Подп.	Дата		
Разраб.		Ильин В.А.			Интерфейс	Литера: У; Масса: ; Масштаб: 1:1
Провер.		Скрипка О.В.				
Т.контр.		Скрипка О.В.				
Н.контр.		Скрипка О.В.				Лист 4; Листов 6
Утв.		Скрипка О.В.				АмГУ Кафедра АПиЭ

Вентилятор (кулер) И радиатор воздушный

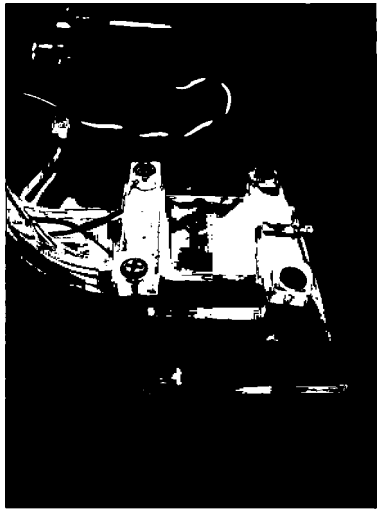
Насос с ёмкостью SC600



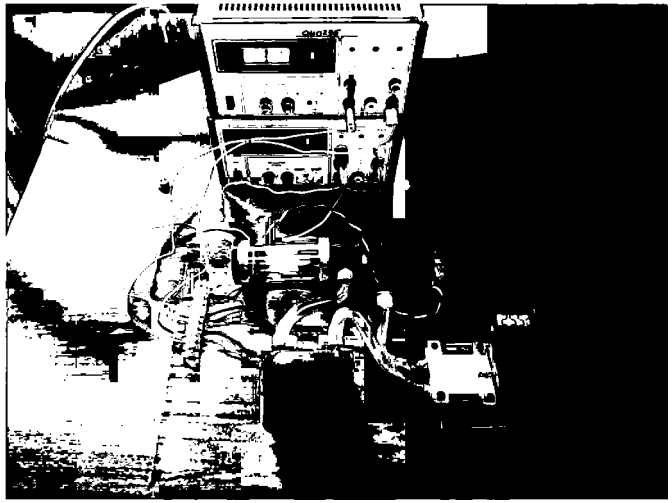
Термодатчик



ТЭМ TEC1_12706 и радиатор водяной



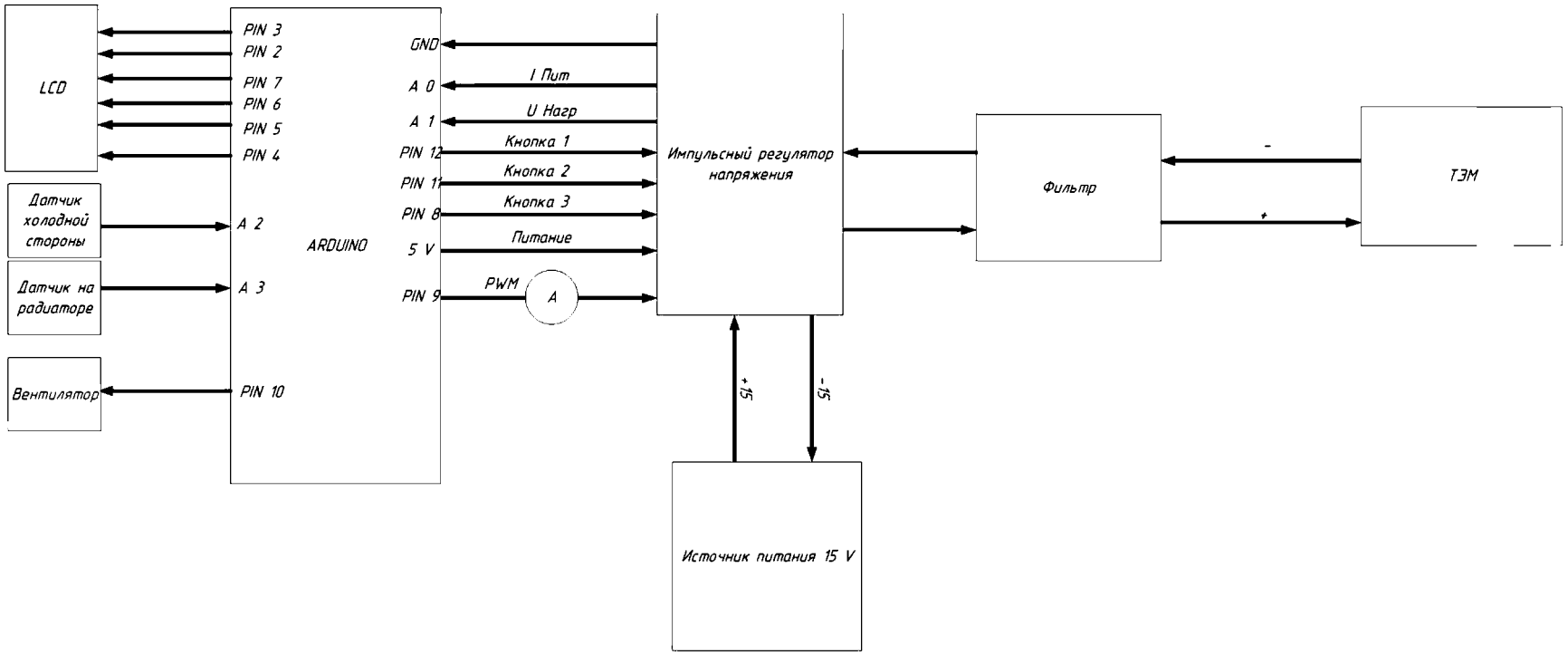
Общий вид установки



								КП.164004.15.03.04. СХ	
Изм.	Лист	№ докум.	Подп.	Дата					
Разраб.		Ильин А.А.			Комплектующие установки		Литера	Масса	Масштаб
Пров.		Скрыпник О.В.							1:1
Т.контр.		Скрыпник О.В.					Лист 5	Листов 6	
Н.контр.		Скрыпник О.В.			Макет интеллектуальной системы управления параметрами научного эксперимента		АмГУ Кафедра АПЦиЭ		
Утвердил		Скрыпник О.В.							

КП 164004 15 03 04 СХ

Структурная схема



						КП 164004 15 03 04 СХ	
Изм.	Лист	№ докум.	Подп.	Дата			
Разраб.		Жиляк А.А.			Структурная схема	Литера	Масса	Масштаб
Провер.		Скрыпник О.В.						1:1
Т.контр.		Скрыпник О.В.				Лист 1	Листов 1
Н.контр.		Скрыпник О.В.			Разработка системы управления параметрами процесса электронагрева Пельтье	АмГУ Кафедра АПиЭ	
Утвердил		Скрыпник О.В.					

ПРИЛОЖЕНИЕ 3

```
// контроллер элемента Пельтье


#include <MsTimer2.h>
#include <avr/wdt.h>
#include <OneWire.h>
#include <LiquidCrystal.h>
#include <Button.h>
#include <EEPROM.h>


#define Ver "1.1"   // версия программы


// адреса EEPROM
#define EEPROM_SET_TEMP_REF 5 // заданная температура в камере
#define EEPROM_MAX_SET_POWER 10 // заданная максимальная мощность


// выводы
#define ERROR_LED_PIN 13  // вывод светодиода ОШИБКА _-_
#define VENT_PIN 10  // вывод вентилятор _-_


#define powerU  12.11     // напряжение источника питания
#define koeffU  0.01679 // массштабный коэффициент напряжения,
                        // koeffU = 1.1 / R1 * (R1 + R2) / 1024 = 1.067/820*(820 + 12000)/1024 =0.01629 В/ед. АЦП
#define koeffI  0.02148 // массштабный коэффициент тока
                        // koeffI =  1,067 / R8 / 1024 = 0.02084 А/ед. АЦП
```

```
#define koeffRegPwrInt 0.05 // интегральный коэффициент регулятора мощности

#define DEAD_TIME 8     // мертвое время ШИМ (* 62,5 нс)

#define koeffRegTmpInt 0.002 // интегральный коэффициент регулятора температуры

#define koeffRegTmpPr 0.5 // пропорциональный коэффициент регулятора температуры

#define koeffRegTmpDif 50. // дифференцирующий коэффициент регулятора температуры


#define tempOnVent 30.  // температура включения вентилятора

#define tempOffVent 27.  // температура выключения вентилятора


#define MAX_POWER 10.   // максимальная выходная мощность контроллера

#define INC_MAX_POWER 1.   // приращение максимальной выходной мощности при установке

#define MAX_SET_TEMP 30.   // максимальная заданная температура

#define MIN_SET_TEMP -10.   // минимальная заданная температура

#define INC_SET_TEMP 0.1   // приращение заданной температуры при установке


#define TIME_PRESS  40    // время удержания кнопок + и - для ускорения изменения параметра (* 20 мс)

#define INC_BUTT_TIME 10  // время изменения параметров при удержании кнопки (* 20 мс)


#define overHeatTemp 50.  // температура перегрева радиатора


#define MEASURE_PERIOD 10  // время периода измерения (* 2 мс)

#define MAX_PWM 255.      // максимальное значение ШИМ
```

```
#define MEASURE_MIN_TEMP  -40.  // минимальная температура измерения

#define MEASURE_MAX_TEMP  100.  // максимальная температура измерения

#define NUM_VER_TEMP_SEN  3     // число подтверждений ошибок датчиков температуры


float measureU; // измеренное напряжение на нагрузке, В

float measureI; // измеренный ток потребления регулятора, А

float measureP; // измеренная мощность на нагрузке, Вт

float measureTempRef;  // измеренная температура в камере (< -200 - ошибка)

float measureTempRad;  // измеренная температура радиатора (< -200 - ошибка)


OneWire sensTempRef (16);  // датчик температуры в камере подключен к выводу 16 (A2)

OneWire sensTempRad (17);  // датчик температуры радиатора подключен к выводу 17 (A3)


LiquidCrystal disp(3, 2, 7, 6, 5, 4); // создаем объект дисплей


Button buttonPlus(8, 15); // кнопка "+"

Button buttonMinus(11, 15); // кнопка "-"

Button buttonChoice(12, 15); // кнопка "выбор"


float regPwrInt=0; // интегральное звено регулятора мощности

float setPower; // заданная мощность

float maxSetPower=MAX_POWER; // максимальная заданная мощность
```

```
float regTmpInt=0; // интегральное звено регулятора температуры
float regTmpPr; // пропорциональное звено регулятора температуры
float regTmpDif; // дифференциальное звено регулятора температуры
float regTmpErr; // ошибка рассогласования температуры
float regTmpErrPrev; // предыдущая ошибка рассогласования температуры
float setTempRef; // заданная температура в камере

byte  interruptCount=0; // счетчик циклов прерываний
unsigned int  sumU, sumI; // переменные для суммирования кодов АЦП
unsigned int  averageU, averageI; // средние значения кодов АЦП
boolean flagReady;  // признак готовности данных измерения
byte  cycle20mcCount; // счетчик циклов 20 мс
byte  indMode=0;    // режим индикации 0 - темп. в камере, 1 - мощность, 2 - темп. радиатора
boolean noDispPar=false;  // признак не индицировать параметры
boolean wasChoice=false;  // признак кнопка выбор была нажата
byte  butPlusTime=0;  // счетчик времени удержания кнопки +
byte  butMinusTime=0;  // счетчик времени удержания кнопки -
byte  incButtTime=0;  // счетчик времени изменения параметра
byte  decButtTime=0;  // счетчик времени изменения параметра
boolean wasPress=false;  // признак было нажатие кнопок + или -
boolean fatalError;   // признак фатальной ошибки
boolean flashErrorLed;  // признак мигания светодиода ОШИБКА
byte  countErrorTempRef=0;  // счетчик ошибок датчика температуры в камере
byte  countErrorTempRad=0;  // счетчик ошибок датчика температуры радиатора
float x;  // промежуточная переменная
```

```
byte  errors=0; // признаки ошибок для передачи на компьютер 0 - ошибка EEPROM, 1 - вентилятор включен, 2 - перегрев, 3 - фатальная ошибка


byte bufData[9];  // буфер данных


void setup() {
 // установка ШИМ 8 разрядов, 62,5 кГц
  TCCR1A = TCCR1A & 0xe0 | 1;
  TCCR1B = TCCR1B & 0xe0 | 0x09;


  Serial.begin(19200);  // инициализируем последовательный порт, скорость 9600
  analogReference(INTERNAL); // опорное напряжение 1,1 В
  MsTimer2::set(2, timerInterupt); // прерывания по таймеру с периодом 2 мс
  MsTimer2::start();              // разрешение прерывания
  wdt_enable(WDTO_15MS); // разрешение работы сторожевого таймера, тайм-аут 15 мс


  pinMode(ERROR_LED_PIN, OUTPUT); // светодиод ОШИБКА
  pinMode(VENT_PIN, OUTPUT); // вентилятор
  digitalWrite(VENT_PIN, LOW);  // вентилятор


  // по индикатору пробегает звездочка
  disp.begin(8, 1);  // инициализация дисплея 1 строка, 8 символов
  disp.write('*');
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
```

```
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
  delay(200);
  disp. scrollDisplayRight();  // сдвиг вправо
}

//------------------------------ основной цикл -----------------------------
void loop() {

//------------------------------ цикл 20 мс ( регулятор мощности )
  if (flagReady == true) {
    flagReady= false;

    //---------------- вычисление измеренных значений
    // вычисление напряжения
    measureU = powerU - (float)(averageU / MEASURE_PERIOD) * koeffU;
    if (measureU < 0) measureU=0;

    // вычисление тока
```

```
        measureI = (float)(averageI / MEASURE_PERIOD) * koeffI;

        // вычисление мощности
        measureP = powerU * measureI;

        //----------------- регулятор мощности
        if ( setPower >= 0) {
          regPwrInt = regPwrInt + (setPower - measureP) * koeffRegPwrInt;
          if (regPwrInt < 0) regPwrInt=0;     // ограничение снизу
          if (regPwrInt > MAX_PWM) regPwrInt=MAX_PWM; // ограничение
сверху
          // мертвое время ШИМ
          unsigned int  pwm = (unsigned int)regPwrInt;  // перевод в ШИМ
          if (pwm < DEAD_TIME) pwm=0;
          if (pwm > (MAX_PWM - DEAD_TIME)) pwm=MAX_PWM;
          analogWrite(9, pwm); // ШИМ
        }
        else {          // выключение
          regPwrInt=0;
          analogWrite(9, 0); // ШИМ
        }


        //------------------------- управление индикацией ----------------------------

        noDispPar = false; // отображать параметры на дисплее

        // отсчет времени удержания кнопок + и -
```

```
if (buttonPlus.flagPress == true) {
  butPlusTime++;
  if (butPlusTime == 0) butPlusTime= 255;
}
else  butPlusTime= 0;
if (buttonMinus.flagPress == true) {
  butMinusTime++;
  if (butMinusTime == 0) butMinusTime= 255;
}
else  butMinusTime= 0;


if (buttonChoice.flagPress == false) {   // кнопка "выбор" не нажата
  if (wasChoice == false) { // ---------------------------------- кнопка "выбор" не нажата и была не нажата
    wasPress=false; // сброс ризнака нажатия кнопок + или -

    // режим индикации
    if ( buttonPlus.flagClick == true ) {
      buttonPlus.flagClick = false;
      indMode ++;
      if (indMode > 2) indMode=0;
      dispParameters();
      noDispPar = true; // не отображать параметры в этом цикле
    }
    if ( buttonMinus.flagClick == true ) {
      buttonMinus.flagClick = false;
      indMode --;
```

```
            if (indMode > 128) indMode=2;
            dispParameters();
            noDispPar = true; // не отображать параметры в этом цикле
          }
        }
        else  { // ---------------------------------------------- кнопка "выбор" не нажата, а была нажата (отпустили)
          wasChoice = false;

          // запись параметра в EEPROM
          if (wasPress == true) {
            // кнопка + или - нажималась, надо записывать EEPROM

            // запись параметров в EEPROM
            if (indMode == 0) writeFloatToEeprom(EEPROM_SET_TEMP_REF, & setTempRef); // заданная температура в камере
            if (indMode == 1) writeFloatToEeprom(EEPROM_MAX_SET_POWER, & maxSetPower); // заданная максимальная мощность
          }
            dispParameters();
            noDispPar = true; // не отображать параметры в этом цикле
          }
        }

        else {   // ---------------------------------------------- кнопка "выбор" нажата
          wasChoice = true;

          if (buttonPlus.flagClick == true) {   // нажали кнопку +
```

```
  buttonPlus.flagClick= false;
  wasPress= true; // признак кнопка нажималась
  incrParameter();
}

if (buttonMinus.flagClick == true) {   // нажали кнопку -
  buttonMinus.flagClick= false;
  wasPress= true; // признак кнопка нажималась
  decrParameter();
}

// изменение параметра при удержании кнопки +
if (butPlusTime >  TIME_PRESS) {
  incButtTime++;
  if (incButtTime > INC_BUTT_TIME) {
    incButtTime= 0;
    incrParameter();
  }
}
else incButtTime= 0;

// изменение параметра при удержании кнопки -
if (butMinusTime >  TIME_PRESS) {
  decButtTime++;
  if (decButtTime > INC_BUTT_TIME) {
    decButtTime= 0;
    decrParameter();
  }
```

```
    }
    else decButtTime= 0;

    // индикация заданных значений
    disp.begin(8, 1);  // инициализация дисплея 1 строка, 8 символов
    noDispPar = true; // не отображать параметры

    if (indMode == 0) {   // индикация заданной температуры в камере
     disp.print("T= ");
     disp.print(setTempRef,1);
    }
    if (indMode == 1) {   // индикация заданной мощности
     disp.print("P< ");
     disp.print((int)maxSetPower);
     disp.print(" W");
    }
    if (indMode == 2) {   // индикация температуры перегрева радиатора
     disp.print("< ");
     disp.print(overHeatTemp,1);
     disp.print(" C");
    }
   }

   // ---------------------------- управление светодиодом ОШИБКА
   if ( flashErrorLed == true ) {
    // мигает
    if ( (cycle20mcCount & 8) == 0 )
digitalWrite(ERROR_LED_PIN,HIGH);
```

```
        else digitalWrite(ERROR_LED_PIN,LOW);
      }
      else digitalWrite(ERROR_LED_PIN,LOW);

      //--------------- выполнение распределенных операций в цикле 1 сек -----
----------
      cycle20mcCount++;   // счетчик циклов 20 мс
      if (cycle20mcCount >= 50) cycle20mcCount= 0;  // время цикла 1 сек



      if (cycle20mcCount == 1) {
       //--------------------------- интервал 1, загрузка параметров из EEPROM

       fatalError= false;  // сброс фатальной ошибки
       errors &= B11111110; // сброс признака ошибки EEPROM

       if (noDispPar == false) {


        // заданная температура в камере
        if ( readFloatFromEeprom(EEPROM_SET_TEMP_REF, &
setTempRef) == false ) {
          if ( readFloatFromEeprom(EEPROM_SET_TEMP_REF, &
setTempRef) == false ) {
            fatalError= true; // признак фатальной ошибки
            errors |= 1;  // признак ошибки EEPROM
            setTempRef= 15.0; // если ошибка, то 15 C
          }
        }
```

```
            // заданная максимальная мощность
            if ( readFloatFromEeprom(EEPROM_MAX_SET_POWER, &
maxSetPower) == false ) {
              if ( readFloatFromEeprom(EEPROM_MAX_SET_POWER, &
maxSetPower) == false ) {
                fatalError= true; // признак фатальной ошибки
                errors |= 1;  // признак ошибки EEPROM
                maxSetPower= 20.0; // если ошибка, то 20 Вт
              }
            }

          }
        }


        if (cycle20mcCount == 5) {
          //-------------------------- интервал 5, инициализация измерения
температуры
          // датчик в камере
          sensTempRef.reset();  // сброс шины 1-Wire
          sensTempRef.write(0xCC, 1); // пропуск ROM
          sensTempRef.write(0x44, 1); // инициализация измерения
          // датчик радиатора
          sensTempRad.reset();  // сброс шины 1-Wire
          sensTempRad.write(0xCC, 1); // пропуск ROM
          sensTempRad.write(0x44, 1); // инициализация измерения
        }

        if (cycle20mcCount == 10) {
```

```
        //-------------------------- интервал 10, передача информации на
компьютер
        Serial.print("U="); Serial.print(measureU, 2);  // напряжение
        Serial.print(" I="); Serial.print(measureI, 2);  // ток
        Serial.print(" P="); Serial.print(measureP, 2);  // мощность
        Serial.print(" p="); Serial.print(regPwrInt, 2);  // интегральное звено
регулятора мощности
        Serial.print(" t="); Serial.print(measureTempRef, 2);  // температура в
камере
        Serial.print(" t="); Serial.print(measureTempRad, 2);  // температура
радиатора
      }

      if (cycle20mcCount == 11) {
        //-------------------------- интервал 11, передача информации на
компьютер
        Serial.print(" E="); Serial.print(regTmpErr, 2);  // ошибка
рассогласования
        Serial.print(" W="); Serial.print(setPower, 2);  // заданная мощность
        Serial.print(" I="); Serial.print(regTmpInt, 2);  // интегральная часть
        Serial.print(" P="); Serial.print(regTmpPr, 2);  // пропорциональная
часть
        Serial.print(" D="); Serial.print(regTmpDif, 2);  // дифференциальная
часть
      }

      if (cycle20mcCount == 12) {
        //-------------------------- интервал 12, передача информации на
компьютер
        Serial.print(" K="); Serial.print(koeffRegPwrInt, 5);  // интегральный
коэффициент регулятора мощности
```

```
        Serial.print(" i="); Serial.print(koeffRegTmpInt, 5);  // интегральный коэффициент регулятора температуры
        Serial.print(" p="); Serial.print(koeffRegTmpPr, 5);  // пропорциональный коэффициент регулятора температуры
        Serial.print(" d="); Serial.print(koeffRegTmpDif, 5);  // дифференцирующий коэффициент регулятора температуры
      }

      if (cycle20mcCount == 13) {
        //-------------------------- интервал 13, передача информации на компьютер
        Serial.print(" V="); Serial.print(tempOnVent, 1);  // температура включения вентилятора
        Serial.print(" v="); Serial.print(tempOffVent, 1);  // температура выключения вентилятора
        Serial.print(" H="); Serial.print(overHeatTemp, 1);  // температура перегрева радиатора
        Serial.print(" P="); Serial.print(maxSetPower, 1);  // максимальная заданная мощность
        Serial.print(" T="); Serial.print(setTempRef, 2);  // заданная температура в камере
        Serial.print(" E="); Serial.print(errors);  // признаки ошибок
        Serial.print(" V="); Serial.print(Ver);  // версия программы
        Serial.println(" ");
      }

      if (cycle20mcCount == 45) {
        //-------------------------- интервал 45, чтение датчика температуры камеры
        sensTempRef.reset();  // сброс шины 1-Wire
```

```
            sensTempRef.write(0xCC, 1); // пропуск ROM
            sensTempRef.write(0xBE, 1); // команда чтения памяти датчика
            sensTempRef.read_bytes(bufData, 9);  // чтение памяти датчика, 9
байтов


            // проверка CRC и диапазона
            x= (float)((int)bufData[0] | (((int)bufData[1]) << 8)) * 0.0625 + 0.03125;
            if ( (OneWire::crc8(bufData, 8) == bufData[8]) && (x >
MEASURE_MIN_TEMP) && (x < MEASURE_MAX_TEMP) ) {
              // правильно
              measureTempRef= x;
              countErrorTempRef=0; // счетчик ошибок
            }
            else {
              // ошибка измерения температуры
              countErrorTempRef++;  // счетчик ошибок
              if (countErrorTempRef > NUM_VER_TEMP_SEN) { // проверка
числа ошибок
                // фатальная ошибка
                countErrorTempRef= 100;
                measureTempRef= -300.;   // ошибка измерения
                fatalError= true; // признак фатальной ошибки
              }
            }
          }


          if (cycle20mcCount == 46) {
            //-------------------------- интервал 46, чтение датчика температуры
радиатора
```

```
            sensTempRad.reset();  // сброс шины 1-Wire
            sensTempRad.write(0xCC, 1); // пропуск ROM
            sensTempRad.write(0xBE, 1); // команда чтения памяти датчика
            sensTempRad.read_bytes(bufData, 9);  // чтение памяти датчика, 9
байтов


            // проверка CRC и диапазона
            x= (float)((int)bufData[0] | (((int)bufData[1]) << 8)) * 0.0625 + 0.03125;
            if ( (OneWire::crc8(bufData, 8) == bufData[8]) && (x >
MEASURE_MIN_TEMP) && (x < MEASURE_MAX_TEMP) ) {
              // правильно
              measureTempRad= x;
              countErrorTempRad=0; // счетчик ошибок
            }
            else {
              // ошибка измерения температуры
              countErrorTempRad++;  // счетчик ошибок
              if (countErrorTempRad > NUM_VER_TEMP_SEN) { // проверка
числа ошибок
                // фатальная ошибка
                countErrorTempRad= 100;
                measureTempRad= -300.;   // ошибка измерения
                fatalError= true; // признак фатальной ошибки
              }
            }
          }

          if (cycle20mcCount == 47) {
            //------------------------- интервал 47, контроль температуры радиатора
```

```
        // управление вентилятором
        pinMode(VENT_PIN, OUTPUT);
        if ( measureTempRad > tempOnVent ) digitalWrite(VENT_PIN, HIGH);
// включение вентилятора
        if ( measureTempRad < tempOffVent ) digitalWrite(VENT_PIN, LOW);
// выключение вентилятора

        // состояние вентилятора
        if (digitalRead(VENT_PIN) == HIGH) errors |= 2;
        else errors &= B11111101; // выключен

        // контроль перегрева
        if ( measureTempRad > overHeatTemp ) {
          fatalError= true; // признак фатальной ошибки
          errors |= 4;  // признак ошибки перегрев
        }
        else errors &= B11111011; // ошибки нет

        // признак фатальной ошибки
        if (fatalError == true) errors |= 8;
        else errors &= B11110111; // нет ошибки
      }

      if (cycle20mcCount == 48) {
        //-------------------------- интервал 48, регулятор температуры в камере

        if ( fatalError == false ) {
          // фатальной ошибки нет
```

```
            flashErrorLed= false; // светодиод ОШИБКА не мигает
            regTmpErr= measureTempRef - setTempRef;  // вычисление ошибки
рассогласования


            regTmpInt= regTmpInt + regTmpErr * koeffRegTmpInt; //
интегральная часть
            if ( regTmpInt > maxSetPower) regTmpInt= maxSetPower; //
ограничение сверху
            if ( regTmpInt < 0 ) regTmpInt= 0;                  // ограничение снизу


            regTmpPr= regTmpErr * koeffRegTmpPr; // пропорциональная часть
            regTmpDif= (regTmpErr - regTmpErrPrev) * koeffRegTmpDif;  //
дифференцирующая часть
            regTmpErrPrev= regTmpErr; // перегрузка предыдущей ошибки


            setPower= regTmpInt + regTmpPr + regTmpDif; // сумма
составляющих
            if ( setPower > maxSetPower) setPower= maxSetPower; // ограничение
сверху
            if ( setPower < 0 ) setPower= 0;                  // ограничение снизу
          }
          else {
            // фатальная ошибка, выключение
            setPower= -200.;  // аварийное выключение регулятора мощности
            regTmpInt= 0;     // сброс интегрального звена регулятора
температуры
            flashErrorLed= true; // светодиод ОШИБКА мигает
          }
        }
```

```
    if (cycle20mcCount == 49) {
      //-------------------------- интервал 49, отображение данных на дисплее
      if (noDispPar == false) dispParameters();
    }
  }
}


//----------------------------- индикация параметров
void dispParameters() {

  disp.begin(8, 1);  // инициализация дисплея 1 строка, 8 символов

  if (indMode == 0) {   // индикация температуры в камере
    disp.print(" ");
    if (measureTempRef < -200.)  disp.print("----");  // ошибка
    else  disp.print(measureTempRef,1);
    disp.print(" C");
  }

  if (indMode == 1) {   // индикация мощности
    disp.print(" ");
    disp.print(measureP,1);
    disp.print(" W");
  }

  if (indMode == 2) {   // индикация температуры радиатора
    disp.print("R ");
```

```
    if (measureTempRad < -200.)  disp.print("----");  // ошибка
    else  disp.print(measureTempRad,1);
    disp.print(" C");
   }

  }

  //------------------------------- увеличение параметра
  void incrParameter() {
   // + к заданному значению параметра
   if (indMode == 0) {   // заданная температура в камере
    setTempRef +=  INC_SET_TEMP;
    if (setTempRef > MAX_SET_TEMP) setTempRef = MAX_SET_TEMP; // ограничение
   }
   if (indMode == 1) {   // максимальная мощность
    maxSetPower +=   INC_MAX_POWER;
    if (maxSetPower >  MAX_POWER) maxSetPower =  MAX_POWER; // ограничение
   }
  }

  //------------------------------- уменьшение параметра
  void decrParameter() {
   // - от заданного значения параметра
   if (indMode == 0) {   // заданная температура в камере
    setTempRef -=  INC_SET_TEMP;
    if (setTempRef < MIN_SET_TEMP) setTempRef = MIN_SET_TEMP; // ограничение
```

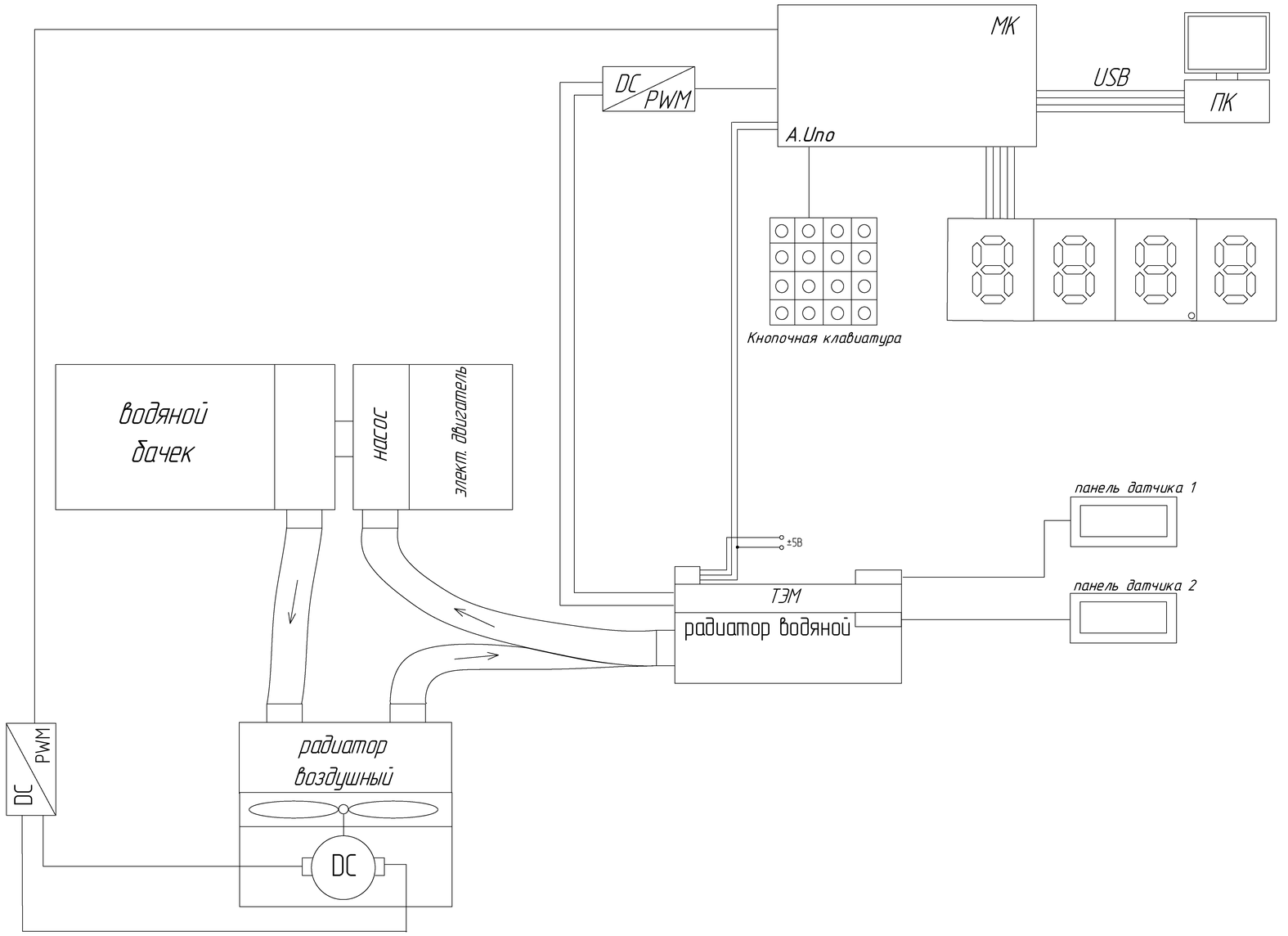
```
  }
  if (indMode == 1) {   // максимальная мощность
    maxSetPower -=   INC_MAX_POWER;
    if (maxSetPower < 0) maxSetPower =  0; // ограничение
  }
}


//------------------------------ обработка прерывания по таймеру 2 мс
void  timerInterupt() {
  wdt_reset();  // сброс сторожевого таймера
  interruptCount++; // счетчик циклов прерываний

  // измерение напряжения и тока
  sumU += analogRead(A1);  // суммирование выборок АЦП
  sumI += analogRead(A0);  // суммирование выборок АЦП
  // проверка числа выборок усреднения
  if ( interruptCount >= MEASURE_PERIOD ) {
    interruptCount= 0;
    averageU = sumU; // перегрузка среднего значения
    averageI = sumI; // перегрузка среднего значения
    sumU= 0;
    sumI= 0;
    flagReady= true;  // признак результат измерений готов
  }

    // кнопки
    buttonPlus.scanState();
```

```
    buttonMinus.scanState();
    buttonChoice.scanState();

  }

  // ------------------------------ запись данного типа Float в EEPROM с контрольной суммой
  void writeFloatToEeprom(int adress, float *value) {

    EEPROM.write(adress, *((byte*)value));
    EEPROM.write(adress+1, *(((byte*)value)+1));
    EEPROM.write(adress+2, *(((byte*)value)+2));
    EEPROM.write(adress+3, *(((byte*)value)+3));
    EEPROM.write(adress+4, (*((byte*)value) + *(((byte*)value)+1) + *(((byte*)value)+2) + *(((byte*)value)+3)) ^ 0xe5);
  }

  //------------------------------ чтение данного типа Float из EEPROM с проверкой контрольной суммы
  // при успешной операции возвращает true
  boolean readFloatFromEeprom(int adress, float *value) {

    *((byte*)value) = EEPROM.read(adress);
    *(((byte*)value)+1) = EEPROM.read(adress+1);
    *(((byte*)value)+2) = EEPROM.read(adress+2);
    *(((byte*)value)+3) = EEPROM.read(adress+3);

    if ( (byte)((*((byte*)value) + *(((byte*)value)+1) + *(((byte*)value)+2) + *(((byte*)value)+3)) ^ 0xe5) == EEPROM.read(adress+4) ) return true;
```

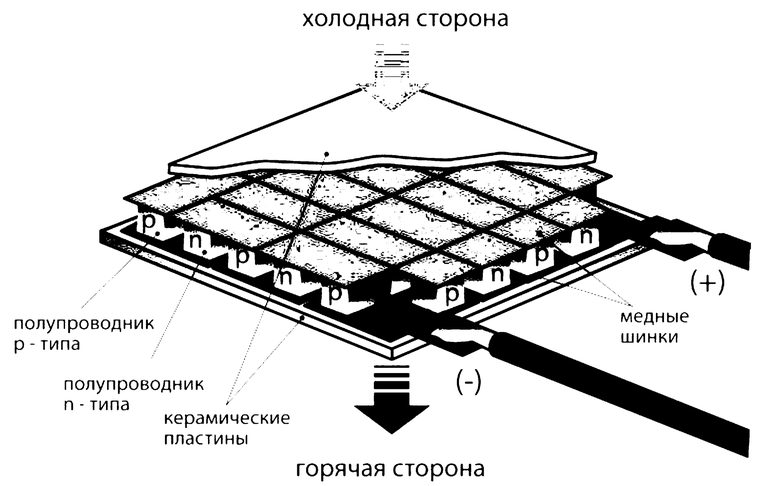
```
  return false;
}
```

КП.164004.15.03.04 СХ

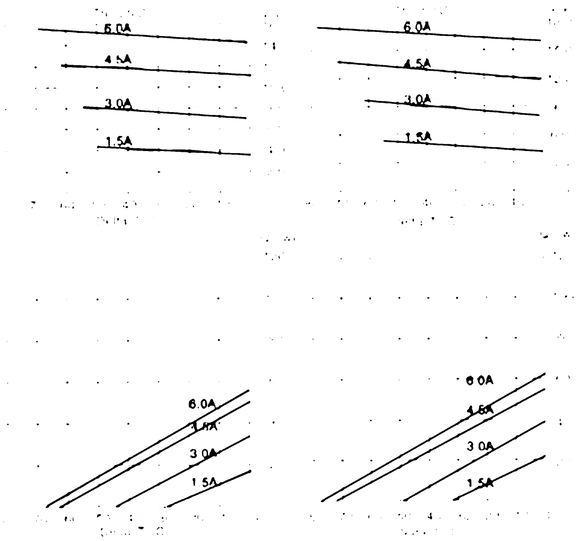
Общая структурная схема АСУ ТП



Структура модуля Пельтье



Технические характеристики TEC1-12706



					КП.164004.15.03.04 СХ		
Изм.	Лист.	№ докум.	Подп.	Дата	СХЕМА ОХЛАДИТЕЛЯ	Литера: У	Масса: — / Масштаб: 1:1
Разраб.		Ильин А.А.				Лист 1	Листов 6
Провер.		Скрипко О.В			Микроконтроллерная система управления термоэлектрическим элементом Пельтье	АмГУ Кафедра АППиЭ	
Т.Контр.		Скрипко О.В					
Н.Контр.		Скрипко О.В					
Утвержд.		Скрипко О.В					

КП.164004.15.03.04 СХ

Схема принципиальная электрическая

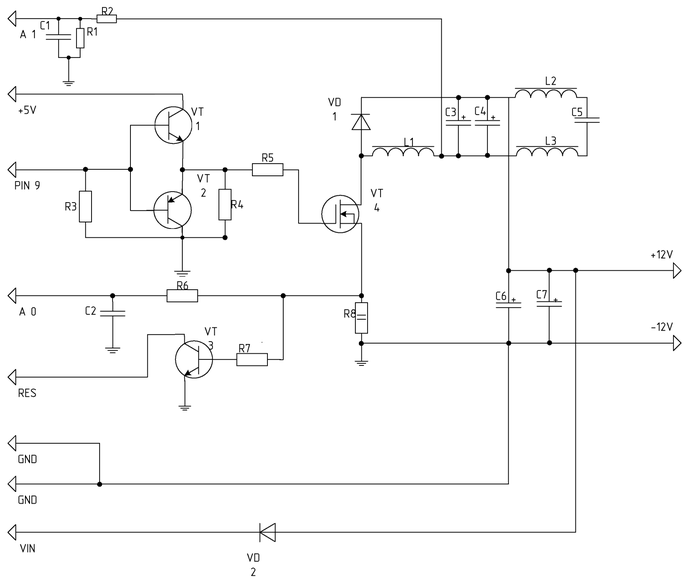
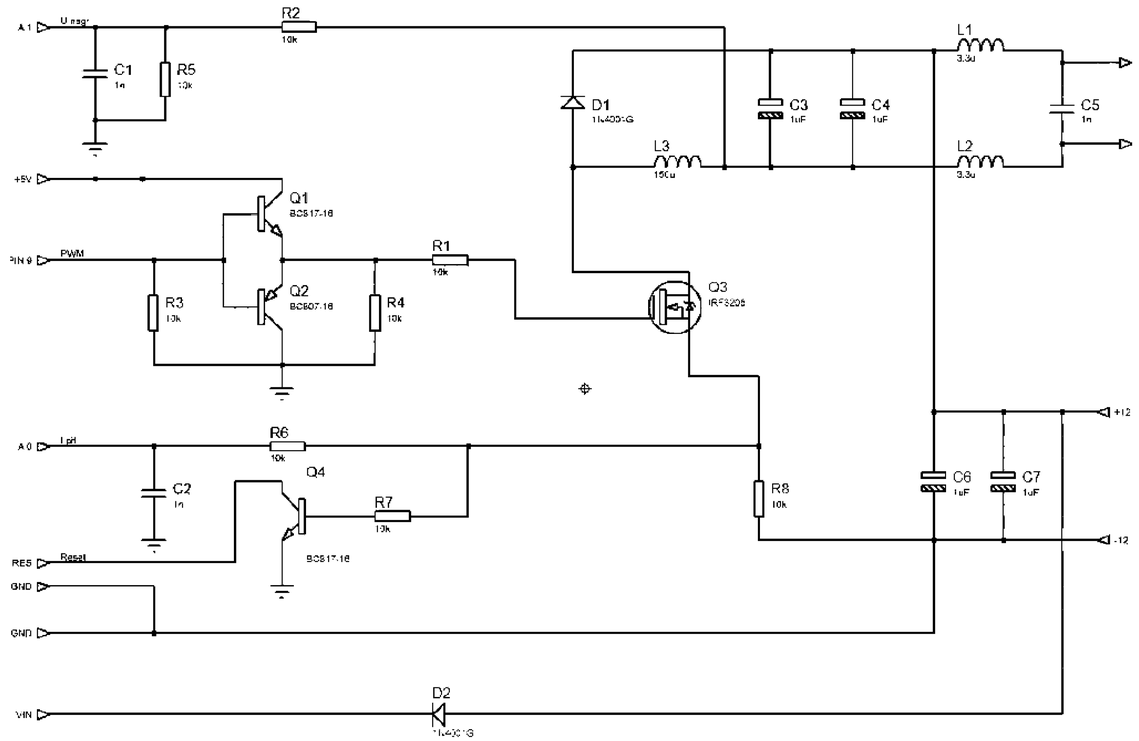
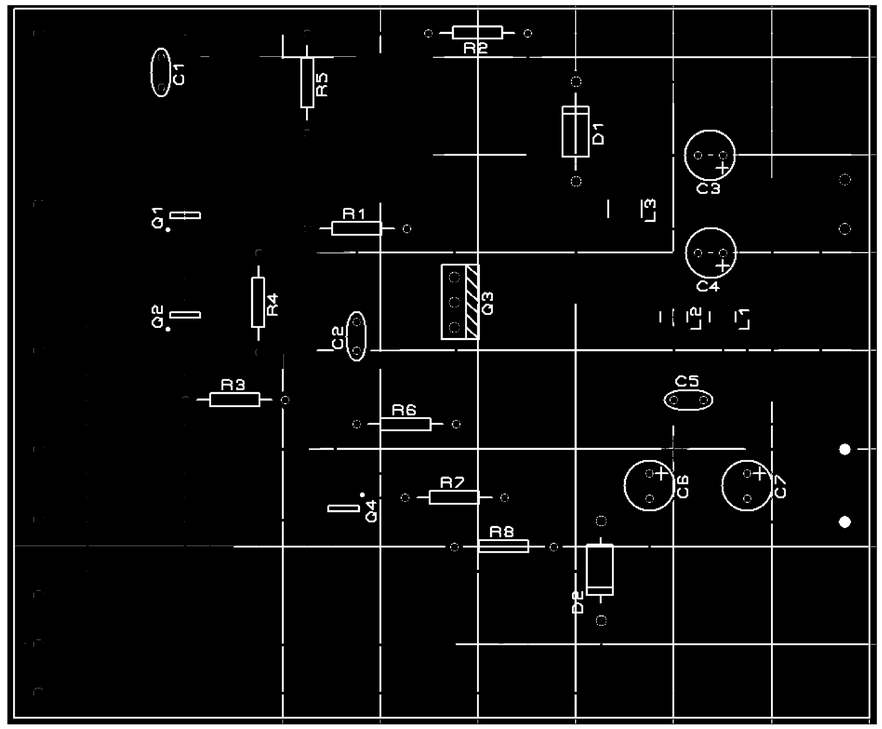


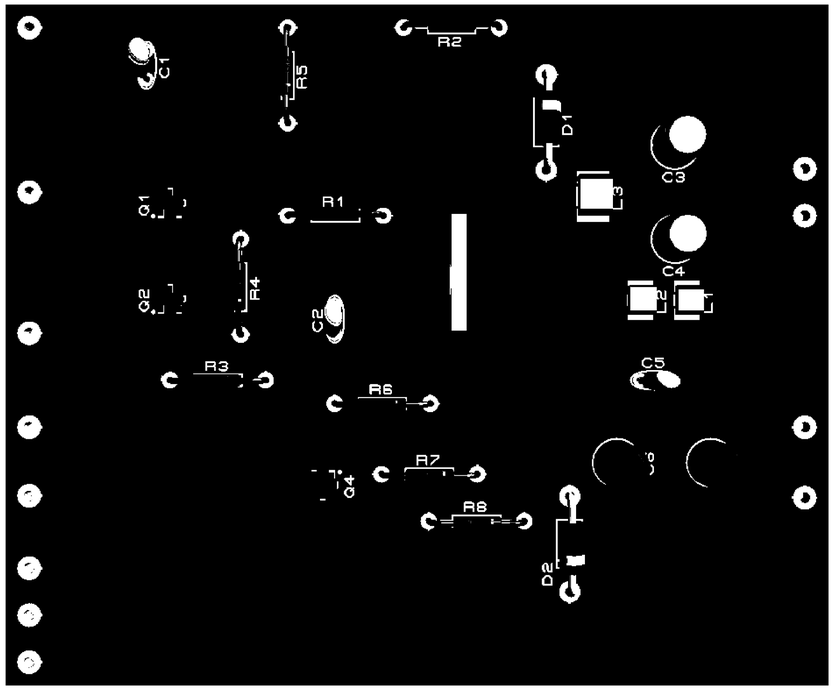
Схема принципиальная электрическая в программе Proteus



Печатная плата



3D Модель платы

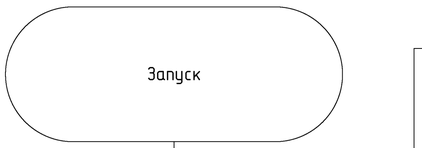


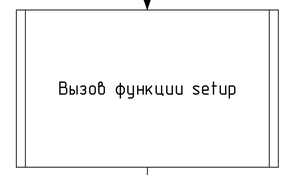
Поз. Обозн.	Наименование	Кол.	Примечание
	Резисторы		
R8	С1-4-0.25-0.05 Ом ± 5	1	
R5	С1-4-0.25-10 Ом ± 5	1	
R4, R6, R7	С1-4-0.25-1 кОм ± 5	3	
R3	С1-4-0.25-4.3 кОм ± 5	1	
R2	С1-4-0.25-12 кОм ± 5	1	
R1	С1-4-0.25-820 Ом ± 5	1	
	Дроссели		
L2, L3	3 мкГн 5 А	2	
L1	150 мкГн 8 А	1	
	Конденсаторы		
C4, C7	К25 – 470 мкФ - 25 В ± 10%	2	
C3, C6	К23 – 4.7 мкФ - 25 В ± 10%	2	
C2, C5	К50-1мкФ-100В ± 10%	2	
C1	К41-0.1мкФ-10В ± 10%	1	
	Транзисторы		
VT4	IRF3205-N-канал 55В 110А	1	
VT2	BC807 - PNP 45В 0.5А 0.31Вт	1	
VT1, VT3	BC817 - NPN 45В 0.5А 0.31Вт	2	
	Диоды		
VD2	SR540 - 5А/250А 40В	1	
VD1	S1M - 1А 1000В	1	

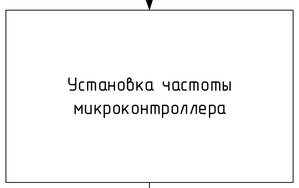
КП.164004.15.03.04 СХ

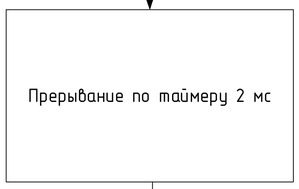
Изм	Лист	№ докум.	Подп.	Дата	Проектирование печатной платы	Литера	Масса	Масштаб
Разраб.		Ильин А.А.				У		1:1
Провер.		Скрипко О.В.				Лист 2		Листов 6
Т.Контр.		Скрипко О.В.			Микроконтроллерная система управления термоэлектрическим модулем Пельтье	АмГУ Кафедра АППиЭ		
Н.Контр.		Скрипко О.В.						
Утвержд.		Скрипко О.В.						

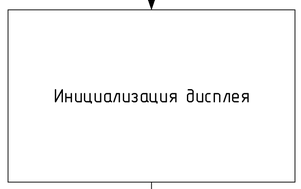
Алгоритм программы

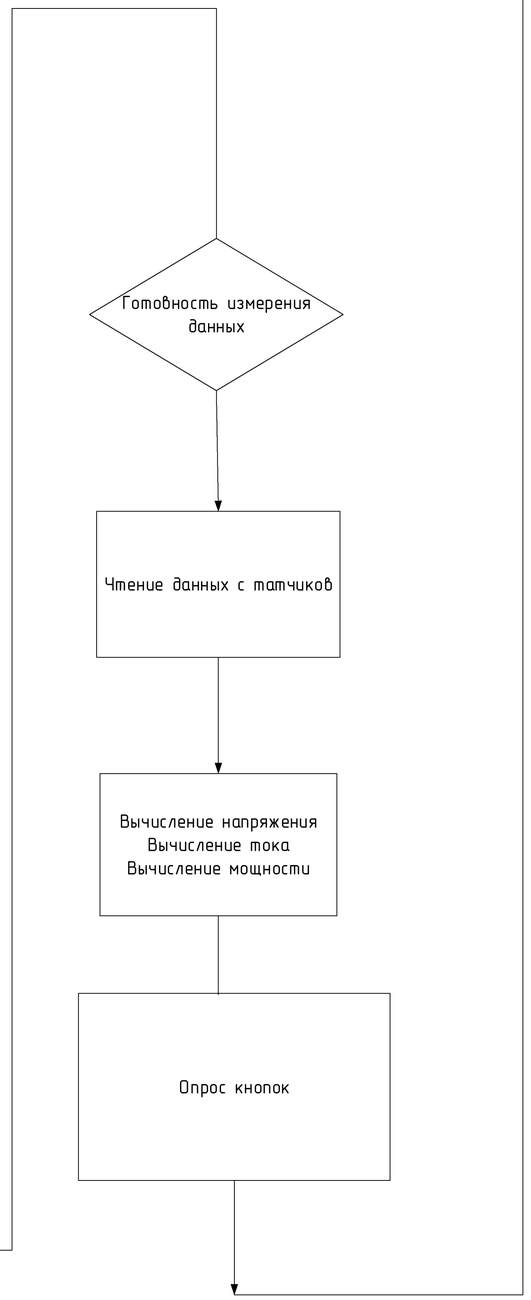


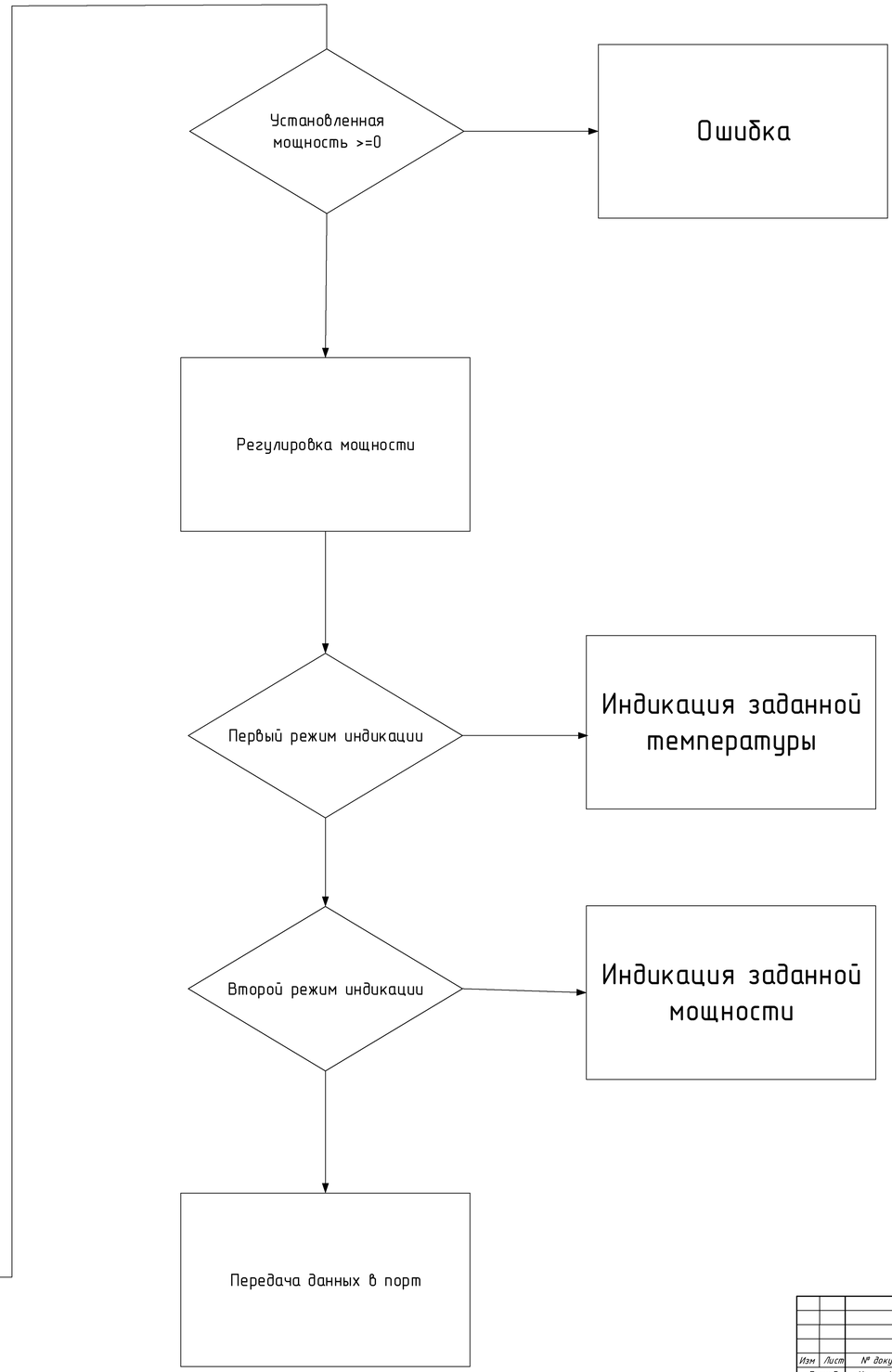






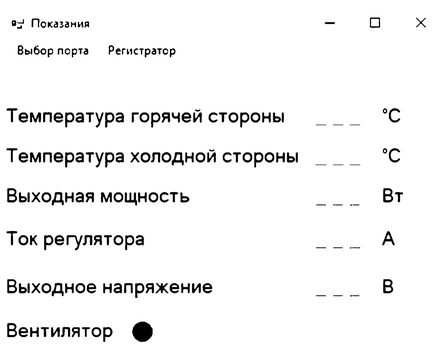


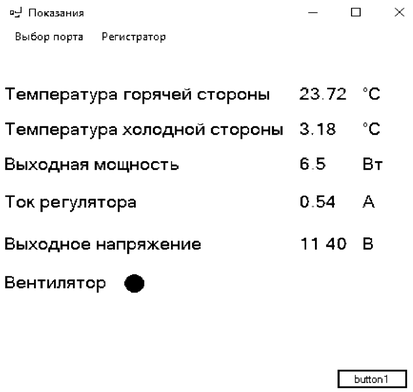




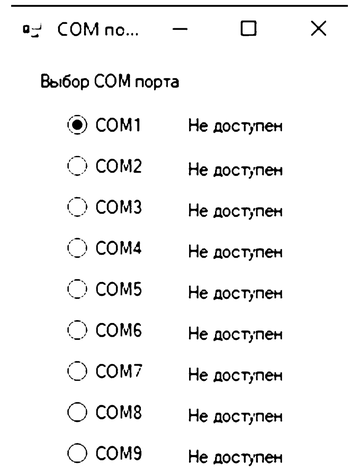
						КП.164004.15.03.04 СХ	
Изм.	Лист	№ докум.	Подп.	Дата			
Разраб.		Ильин А.А.			Алгоритм программы	Литера: У	Масса: 1:1
Провер.		Скрипко О.В				Лист 3	Листов 6
Т.Контр.		Скрипко О.В			Микроконтроллерная система управления термоэлектрическим модулем Пельтье	АмГУ Кафедра АППиЭ	
Н.Контр.		Скрипко О.В					
Утвержд.		Скрипко О.В					

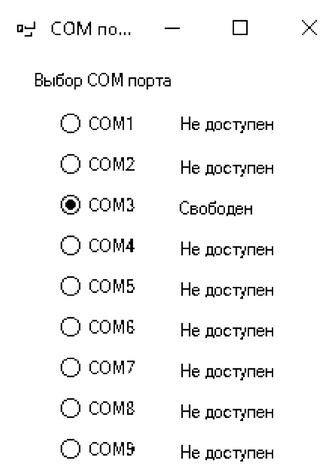
Главное окно



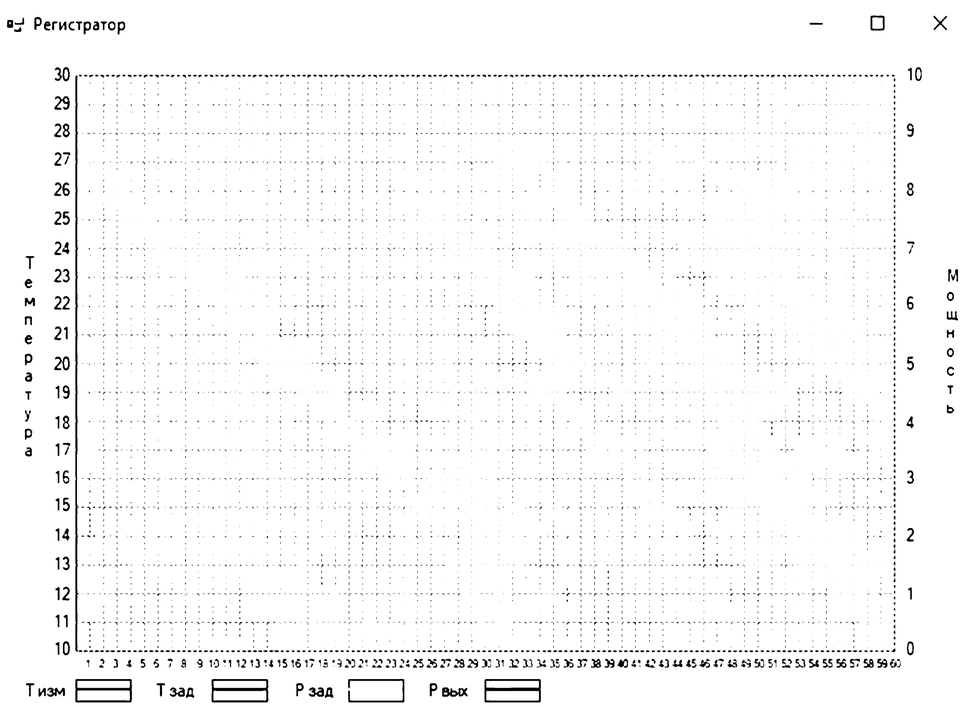


Окно выбора COM порта





Окно регистратора в отключенном состоянии

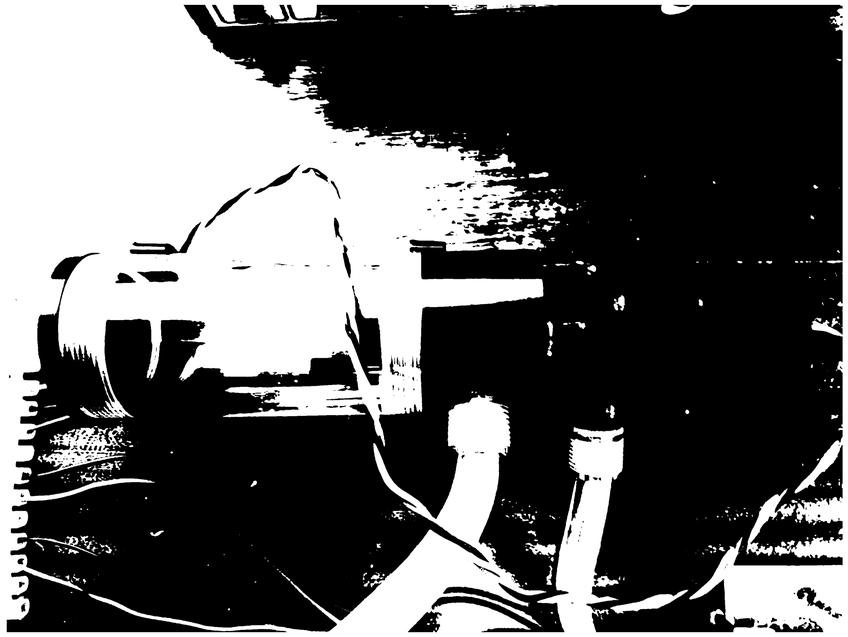


Окно регистратора в подключенном состоянии

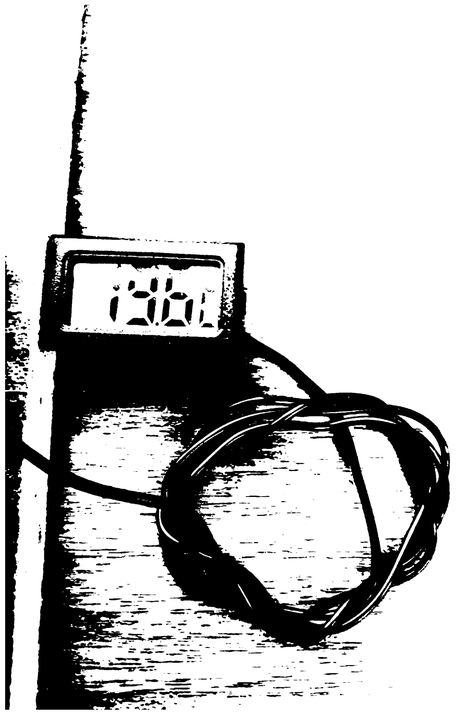
						КП.164004.15.03.04 СХ	
Изм	Лист	№ докум.	Подп.	Дата		Интерфейс	Литера: У; Масса: ; Масштаб: 1:1
Разраб.		Ильин А.А.					Лист 4 Листов 6
Провер.		Скрипко О.В					
Т.Контр.		Скрипко О.В					
Н.Контр.		Скрипко О.В				Микроконтроллерная система управления термоэлектрическим элементом Пельтье	АмГУ Кафедра АППиЭ
Утвержд.		Скрипко О.В					

Вентилятор (кулер) И радиатор воздушный

Насос с ёмкостью SC600

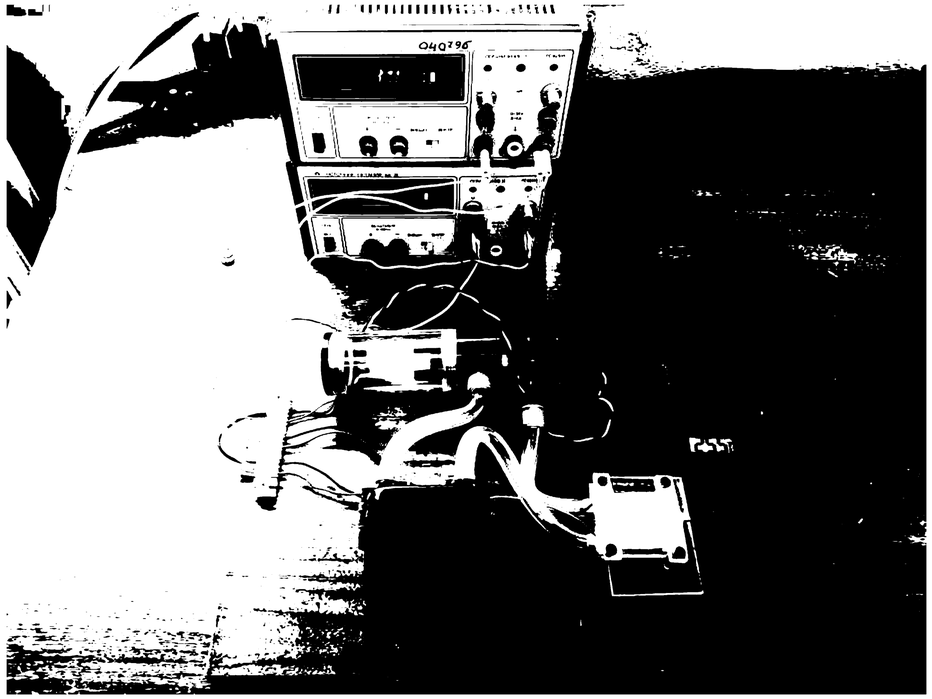


Термодатчик



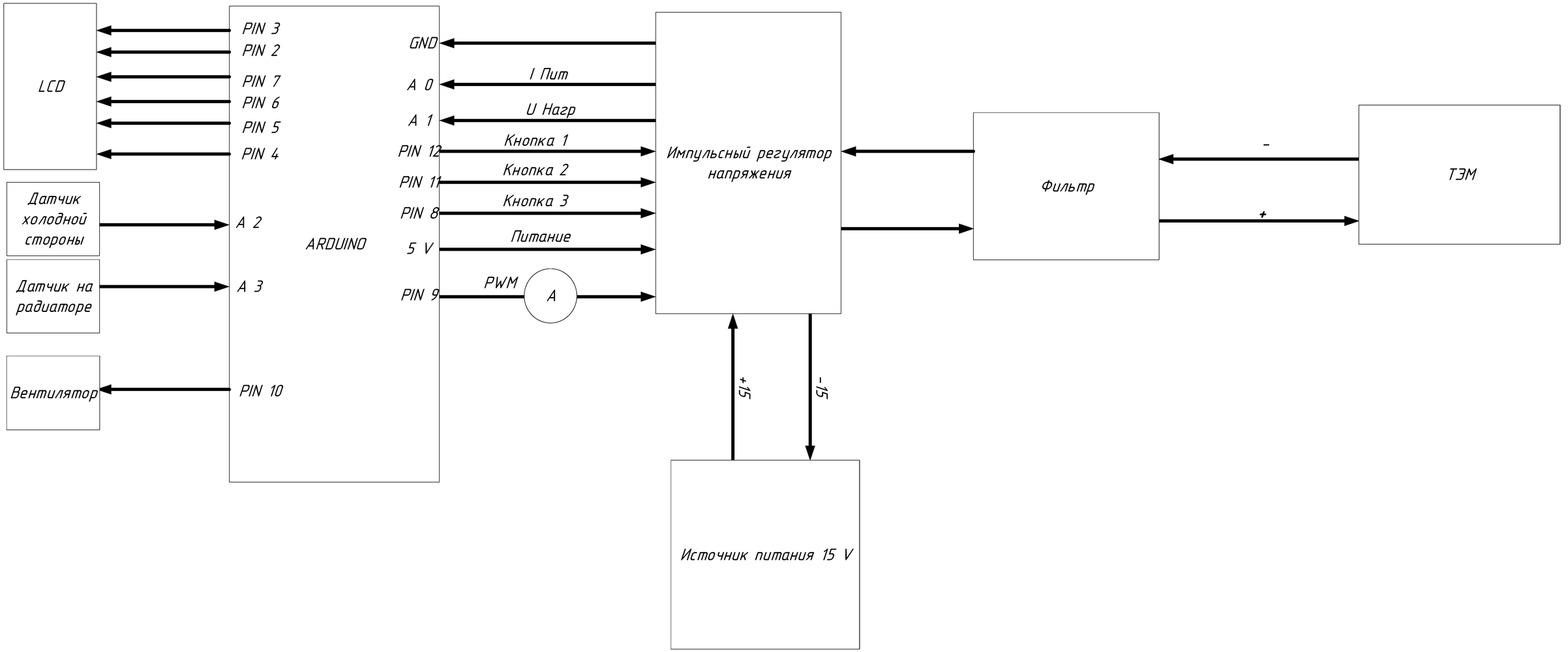
ТЭМ TEC1_12706 и радиатор водяной

Общий вид установки



						КП.164004.15.03.04 СХ	
Изм.	Лист	№ докум.	Подп.	Дата			
Разраб.		Ильин А.А.			Комплектующие установки	Литера: У	Масса:	Масштаб: 1:1
Провер.		Скрипко О.В				Лист 5	Листов 6	
Т.Контр.		Скрипко О.В						
Н.Контр.		Скрипко О.В			Микроконтроллерная система управления термоэлектрическим элементом Пельтье	АмГУ Кафедра АППиЭ	
Утвердил.		Скрипко О.В					

Структурная схема



						КП.164004.15.03.04 СХ		
Изм	Лист	№ докум.	Подп.	Дата				
Разраб.		Ильин А.А.			Структурная схема	Литера: У	Масса:	Масштаб: 1:1
Провер.		Скрипко О.В.						
Т.Контр.		Скрипко О.В.				Лист 6	Листов 6	
Н.Контр.		Скрипко О.В.			Микроконтроллерная система управления термоэлектрическим элементом Пельтье	АмГУ Кафедра АППиЭ		
Утвержд.		Скрипко О.В.						