МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт машиностроения
(наименование института полностью)

Кафедра «Промышленная электроника»
(наименование)

11.03.04 Электроника и наноэлектроника
(код и наименование направления подготовки, специальности)

Электроника и робототехника
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Терморегулируемая холодильная камера на элементе Пельтье»

Студент В.В. Райков
(И.О. Фамилия) (личная подпись)

Руководитель к.т.н., доцент М.В. Позднов
(ученая степень, звание, И.О. Фамилия)

Тольятти 2022

Аннотация

Объем 98 с., 37 рисунок, 6 табл., 21 источников

Терморегулируемая холодильная камера на элементе Пельтье

Объектом исследования являются термоэлектрические холодильные установки.

Цель работы: создание условий для хранения продуктов и медикаментов.

Задачи работы:

1. Выбрать элементы для установки
2. Разработать электричскую схему питания
3. Разработать программу для контроллера, обеспечивающую режим термостатирования в камере.
4. Провести экспериментальные измерения работы камеры.

Работа состоит из трех разделов, в которых решены упомянутые задачи.

Вработе использовались: программная среда Arduino IDE для разработки управляющей программы для контроллера осуществляющего термостатирование камеры.

В процессе работы был создана камера с функцией термостатирования на элементе Пельтье, проведены экспериментальные иследования работы камеры.

Степень внедрения – экспериментальный образец рабочей установки с заданными по техническому заданию требованиями.

Областью применения данной разработки является использование для охлаждения продуктов и хранения медикаментов в условиях мобильного использования.

Содержание

Введение

Использование холодильных установок для различных целей, например сохранения продуктов, сохранения медицинсктих препаратов весьма актуален. Особенно важен режим сохранения заданной температуры на протяжении длительного времени – термостатирования. Данные цели достигаются применением компрессорных холодильных установок показавших свою эффективность. Их номенклатура весьма широка в современной технике. Однако они обладают рядом недостатков являющихся их специфическими чертами. Например, они достаточно громоздки, требуют, как правило, промышленной сети переменного тока. Использование альтернативных холодильных элементов на эффекте Пельтье, может решить выше указанную проблему, особенно если требуется компактная малогаборитная и переносная установка. Из особенностей так же следует отметить, что если требуется маломощная установка, то холодильник на Пельте элементах обходится дешевле, чем на компрессорной основе.

Объектом исследования в данной работе являются холодильные камеры, а предметом исследования - термоэлектрические холодильные камеры.

Цель работы: создание условий для хранения продуктов и медикаментов.

Для достижения цели необходимо решить ряд задач:

- выбрать элементы для установки;
- разработать электричскую схему питания;
- разработать программу для контроллера, обеспечивающую режим термостатирования в камере;
- провести экспериментальные измерения работы камеры.

1 Состояние вопроса

Элемент Пельтье довольно капризный полупроводниковый прибор. По этой причине приходиться предъявлять жесткие требования к управляющему контроллеру. Так как если этим пренебречь в процессе разработки холодильника, работающем в круглосуточном режиме, это может привести к снижению ресурса работы холодильника. Пример элемента приведен на рисунке 1. Технические характеристики показаны в таблице 1.

Рисунок 1- Двухслойный элемент TEC2-19006

Таблица 1-Технические характеристики элемента TEC2-19006

Модель No.	Зарядное устройство Imax A	Tmax ℃	Vmax V	Qmax W	Размеры мм
TEC2-19006	6	≥ 80 ℃	16	36	40*40*6,3

При проектировании питания элемента Пельтье необходимо, чтобы пульсации тока были небольшие и ток при этом был близок к постоянному. Поскольку от среднего тока зависит холодопроизводительность элементом, а тепловые потери зависят от действующего значения напряжения приложенного к элементу или току, то при питании элемента пельтье

импульсным напряжением или током эти потери резко возрастают и уменьшают холодопроизводительность элемента.

Использование традиционных выпрямителей на основе трансформатора - неуправляемого выпрямителя и стабилизирующей ёмкости неудобно так как такая система не позволяет регулировать напряжение. Возможно, конечно, использование для этих целей импульсно-фазового управления выпрямителем, которое дают регулировку выходного напряжения в небольшом диапазоне напряжений. Однако она требует использования нескольких управляемых ключей тиристоров, формирователей импульсов для них, схемы следящей за фазой сетевого напряжения. Самый простой метод питания заключается в подаче напрямую на элемент пельтье ШИМ-сигнала напряжения, который может быть получен использованием источника постоянного напряжения - силового ключа, например полевого транзистора. Для получения сглаженого тока необходимо последовательно с элементами Пельтье необходимо установить катушку индуктивности и зашунтировать их диодом. Т.е. схемотехнически это будет понижающий преобразователь постоянного напряжения.

Регулирование температуры в холодильных установках, например компрессорного типа, чаще всего выполняется с помощьюрелейного алгоритма. То есть имеется два заданных порога температуры - верхний и нижний. Если температура в камере выше верхнего порога, то, включается компрессор, температура начинает снижаться и при достижении нижнего порога компрессор выключается, и температура постепенно повышается самостоятельно за счет потерь холодильника. Такой алгоритм управления является простым и надёжным, период коммутаций при этом составляет десятки - сотни секунд. Однако, в применении с элементом Пельтье он имеет недостатки. При отключении элемента тепло быстро натекает с горячей поверхности элемента на холодную и приводит к его резкому нагреву.

На горячей на холодной стороне при включениях и отключениях вы возникают большие колебания температуры порядка десятка градусов. При таком режиме внутри пластин полупроводников соединяемых с поверхностями элементы пельтье возникают большие механические напряжения, что приводит к уменьшению ресурса работы элемента пельтье. Есть информация, которая говорит о том, что количество таких циклов за время жизни элемента пельтье не превышает нескольких тысяч. При постоянной работе в аналогичном режиме время работы такого холодильника будет составлять несколько месяцев, что конечно неприемлемо для рабочего стола новки.

Эффект перетекания тепла с горячей поверхности элемента Пельтье на холодную также является недопустимым и является вредным, поскольку приводит к бесполезным нагреву камеры и уменьшает холодоэффективность работы холодильника.

Таким образом, необходимо использовать непрерывное регулирование тока и напряжения на элементе пельтье без образования режимов частого включения и выключения тока напряжения на элементе Пельтье

Рассмотрим некоторые характеристики элемента Пельтье.

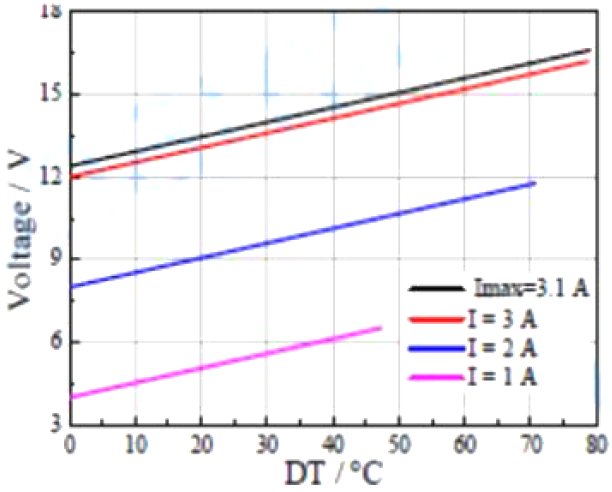

Рисунок 2 - Семейство зависимостей напряжения на элементе TEC 12702 от перепада температур при разных токах.

Зависимость напряжения от тока является температурной (рисунок 2). Как видно из графика с увеличением перегрева между горячей и холодной сторонами элемента пельтье для сохранения заданного тока напряжение необходимо повышать.

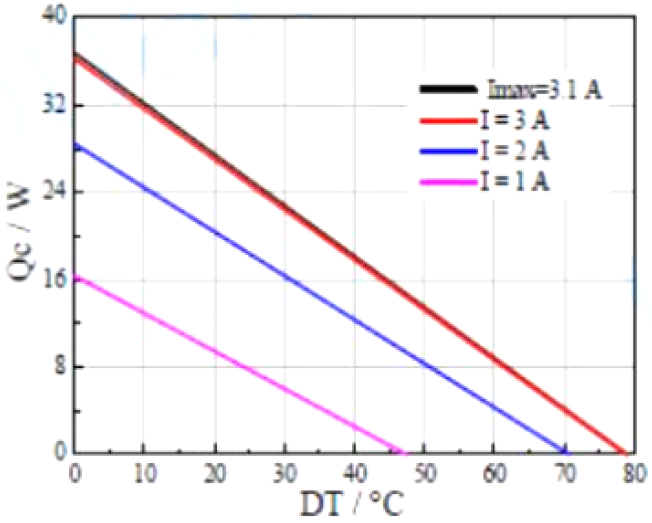

Рисунок 3 - Семейство зависимостей потока мощности на элементе TEC 12702 от перепада температур при разных токах.

От уровня тока, протекающего через элемент зависит поток перекачиваемой мощности (рисунок 3). Как видно из графика, при минимальной разнице температур между холодной и горячей сторонами холодопроизводительность максимальна. Такой режим наблюдается при пуске холодильника, когда температура в камере и окружающего воздуха одинаковы. По мере охлаждения камеры и нагрева внешнего радиатора нарастает разница температур и холодопроизводительность уменьшается для получения. Для получения как можно меньших температур в холодильнике необходимо, чтобы сопротивление тепловое стенок холодильной камеры было как можно больше. Тогда в пределе разница температур, получаемая в камере определяется точкой пересечения графиков при заданном токе с осью температур, например при токе 1 ампер как это следует из графика (рисунок3) эта температура равна 47 градусов. Тогда если считать, что

тепловое сопротиаление радиатора очень мало и падение температуры на нём близка к нулю, то при внешней температуре 27 град, можно создать минимальную температуру в холодильной камере не ниже 27-47=-20 град.

1.1 Требования к контроллеру элемента Пельтье, связанные со спецификой эксплуатации холодильника

Особые условия работы холодильника предъявляют к его разработке ряд требований. Температура внутри камеры должна поддерживаться с заданной точностью и не должно происходить резких колебаний температуры на элементе, таким образом для осуществления термостабилизации необходимо использовать алгоритм по типу ПИД-регулирования, ане релейного. При этом управление температуры необходимо осуществлять через управлениетока или напряжения или мощности в элементе.

Ещё одним требованием является поддержание равномерной температуры внутри холодильника. Это можно обеспечить установкой на холодную сторону элемента пельтье радиатора с вентилятором. Он будет поддерживать постоянную конвекцию внутри холодильника тем самым позволит улучшать эффективность отвода тепла и улучшать динамику охлаждения в камере. Если внутри камеры установить только теплоотводящую пластину, то динамика камеры может резко ухудшиться, то есть установление температуры займёт десятки минут.

Устройство предполагает режим непрерывной работы, что также следует учесть при проектировании силового модуля управляющего питания установки, поэтому для регулирования напряжения или тока необходимо использовать импульсные регуляторы.

ПИД-регулятор должен при запуске установки осуществлять режим включения установки на максимальную мощность, по мере установления

температуры регулятор должен уменьшать потребляемую мощность до уровня который необходим для поддержания заданной температуры в камере.

Система должна осуществлять возможность аварийного отключения питания холодильника при выходе из строя вентилятора на горячей стороне поскольку при этом температура горячей стороны резко повысится и может привести к выходу из строя элемента пельтье вследствие перегрева.

Контроллер должен иметь простой интуитивно понятный интерфейс в нём должна быть предусмотрена возможность задание температуры необходимой в холодильнике также текущей температуры в холодильнике. Контроллер должен иметь экран для вывода информации, а также органы управления для изменения температуры и режимов работы холодильника. Вследствие реверсивного эффекта работы элемента пельтье холодильник может работать также и на нагрев.

Предполагаемыми параметрами разрабатываемой установки являются параметрами сведеные в таблицу (таблица 2).

Таблица 2 - Параметры холодильника

Точность поддержания температуры в камере	0.5 С
Диапазон перепада температуры внешняя-внутренняя	0 – 25 С
Максимальная выходная мощность охлаждения (в начале цикла)	30 Вт
Максимальный выходной ток	4 А
Максимальное выходное напряжение	12 В
Напряжение питания, нестабилизированное	10 - 12 В
Габариты	500 х500 х 500 мм

1.2 Общие сведения о регуляторах

В общем случае регулятор это устройство которое следит за за тем чтобы выходной параметр температура ток Напряжение мощность на объекте Tele2 в соответствии с заданным значением таким образом регулятор работает в случае наличия обратной связи. Снимая с объекта или связи которая вычисляется косвенным образом через известные параметры в самом простом случае требуется поддержания выходного параметра на заданном уровне, в нашем случае каким параметрам является температура внутри холодильной камеры, которая должна поддерживаться в вне зависимости от изменения внешней температуры

В регуляторах требуется выделить элементы ты которые позволяют осуществить регулирование а именно на элемент, снимающий или определяющий выходной параметр в текущем состоянии в нашей установке таким элементам будет являться датчик температуры установленный внутри холодильной камеры далее требуется определить механизм и устройство регулирования этой температуры в нашем случае удобно для регулирования использовать изменения напряжения на элементе пельтье а именно увеличение напряжения на элементе пельтье увеличивает мощность и холодопроизводительность и способствует уменьшению температуры внутри камеры предполагается такое управление сделать с помощью широтно импульсного регулирования напряжения на элементе Пельтье.

На рисунке 4 показан регулятора мощности, регулируемый параметр - электрическая мощность на нагрузке, а регулирующий элемент - широтно-импульсный модулятор.

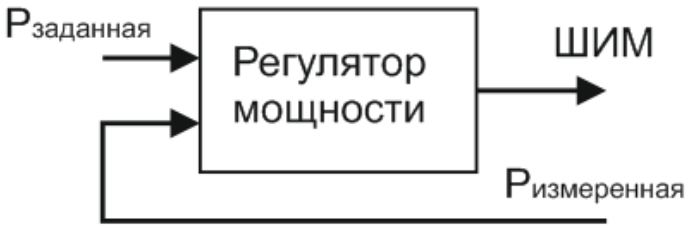

Рисунок 4- Регулятор мощности.

Любой регулятор может по-разному регулировать температуру, для оценки его работы используют критерии оценки качества его работы:

- скорость регулирования (быстродействие) - время уменьшения ошибки регулирования до заданной величины;
- точность регулирования - ошибка параметра регулирования в установившемся состоянии;
- устойчивость регулятора - отсутствие колебаний параметра регулирования.

Выполненеие этих критериев на заданном уровне состоит в настройке ПИД-регулятора.

В силу инерционности работы системы, ее тау превышает тысячу секунд, опрос датчика температуры (дискретность системы) в 1 с вполне достаточен для расчета воздействия на управление ШИМ.

1.3 Интегральный регулятор

«Самый простой для поддерживания значения мощности на выходе является интегоальный способ управления.

Во - первых, сравним заданную мощность с измеренной, во - вторых, если заданное значение больше реального, то ШИМ увеличить на 1 ;в - третьих,.если заданное значение меньше реального, то ШИМ уменьшить на 1.»[1]

«Регулятор с таким алгоритмом управления будет работать, только критерии качества регулирования у него не на высоте. Причем абсолютно все. »[1]

«Для более качественного регулирования необходимо прибавлять к текущему значению ШИМ величину, зависящую от ошибки параметра регулирования, т.е. ввести пропорциональную составляющую.»[1]

Математически закон управления интегрального регулятора выглядит (формула 1),

$$Kw = Ki \times \int e(t)dt \quad (1),$$

где: Kw – коэффициент заполнения ШИМ;

Ki – интегральный коэффициент;

e(t) – ошибка рассогласования, т.е. разница между заданным и реальным значениями регулируемого параметра.

«Выходная функция интегрального регулятора пропорциональна интегралу по времени от отклонения регулируемого параметра. »[1]

«Интегральный регулятор это регулятор последовательного приближения. Большая ошибка - он изменяет ШИМ большими шагами. Маленькая ошибка он медленно ее компенсирует. Ошибка накапливается в интеграторе и сколь малой она не была бы, все равно со временем она окажет воздействие на регулирующий элемент. » [1]

«В более понятном виде, близком к дискретной реализации схема интегрального регулятора выглядит так (рисунок 5). »[1]

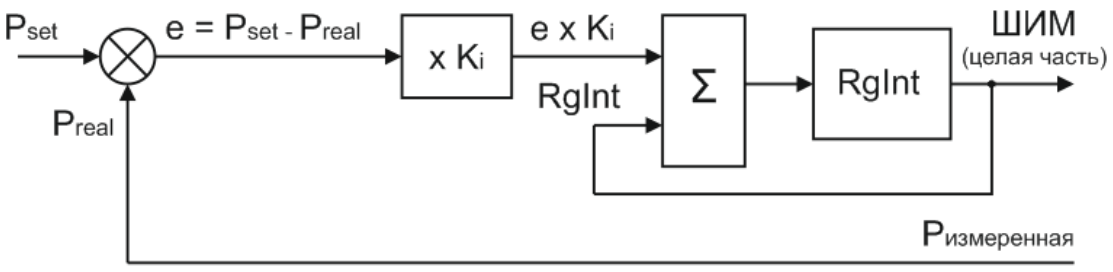

Рисунок 5- Схема интегрального регулятора

Вычисляется ошибка рассогласования e, как разность между заданной мощностью Pset и измеренной на выходе Preal :e = Pset – Preal. Ошибка рассогласования e умножается на интегральный коэффициент Ki и накапливается в регистре-интеграторе RgI. Целые разряды регистра поступают на широтно-импульсный модулятор.

«Как правило, интегратор имеет достаточно большую разрядность с дробной частью. А ШИМ может воспринимать только целые значения. Значения меньше еденицы постепенно накапливаются в интеграторе и переходят в целую часть, а значит и в ШИМ. Это позволяет при малых ошибках рассогласования или малом значении Ki уменьшать быстродействие регулятора. Малые значения долго переходят в целую часть.» [1]

Интегральные регуляторы обладают:

- высокой точностью;
- низкой скоростью регулирования;
- посредственной устойчивостью, зависящей от скорости регулирования.

«Выбор для регулятора мощности на элементе Пельтье интегральный закон управления. Связано, во- первых, элемент Пельтье меняет свои параметры в зависимости от температуры. Но происходит это крайне медленно. Быстрый регулятор нам просто не нужен. Во- вторых, более того резкое изменение мощности на элементе Пельтье ведет к деградации полупроводниковых кристаллов модуля. Из-за резких изменений температуры в местах спайки полупроводников возникают механические напряжения, что ведет к снижению эффективности элемента и даже выходу его из строя. Поэтому как бы быстро не менял регулятор температуры заданное значение для регулятора мощности необходимо, чтобы изменение мощности на элементе Пельтье происходило плавно. В - третьих, конденсатор большой емкости на выходе регулятора, который также лучше заряжать медленно.»[1]

«Например, при включении питания при не охлажденной камере холодильника регулятор температуры должен включить элемент на полную мощность. Необходимо, чтобы это произошло не мгновенно, а в течение нескольких секунд.»[1]

«Ко всем этим требованиям идеально подходит именно интегральный регулятор. Более того коэффициент Ki мы специально снизим, чтобы обеспечить медленное изменение мощности на нагрузке. » [1]

1.4 Программа регулятора мощности

Нужны следующие переменные и константы:

float measureP; //измеренная мощность на нагрузке, Вт– эта переменная в программе уже есть.

Добавим:

float setPower; //заданная мощность float regPwrInt=0; //интегральное звено регулятора мощности #define koeffRegPwrInt 0.05 //интегральный коэффициент регулятора мощности

Сам регулятор уместился в одну строку:

regPwrInt = regPwrInt + (setPower - measureP) * koeffRegPwrInt;

И еще надо перегрузить целую часть из интегратора в ШИМ:

analogWrite(9, (unsigned int) regPwrInt); //ШИМ

В принципе эта программа уже работает. Можно временно задать мощность равной, например, 5 Вт:

setPower = 5; //временно заданная мощность 5 Вт

вставить регулятор в цикл 20 мс и проверить. Но не хватает еще кое-каких операций.

1.5 Ограничение интегратора

«Задана мощность, которую регулятор не способен обеспечить, например, 50 Вт. Регулятор должен сформировать максимальный ШИМ. Но интегральное звено нашего регулятора будет продолжать увеличиваться. Когда оно превысит максимально-допустимое значение ШИМ (у нас это 255), ШИМ перестанет правильно работать. Скорее сбросится в 0 и опять начнет увеличиваться. Т.е. необходимо ввести ограничения интегрального звена. Оно не должно быть больше максимального значения ШИМ и не допустимо, чтобы оно стало отрицательным. »[2]

if (regPwrInt < 0) regPwrInt=0;//ограничение снизу if (regPwrInt > MAX_PWM) regPwrInt=MAX_PWM; //ограничение сверху

«Мертвое время» ШИМ**.**

«Есть еще одна тонкость работы с ШИМ. Импульсы на выходе ШИМ переключают реальный ключ. При уменьшении коэффициента заполнения импульсы включения ключа могут стать очень короткими. Для ШИМ значение 1 соответствует импульсу длительностью 62,5 нс. За такое короткое время ключ не успеват открываться полностью и нормально работать не будет. Но в высоковольтных цепях источников питания (300 В и более) такая коммутация приводит к катастрофическим последствиям. Поэтому хороший стиль управления ШИМ - это запрет слишком коротких импульсов управления. »[2]

«Вводят два временных отрезка, на котором запрет работы ШИМ. Один отрезок вблизи нуля, второй около максимального значения. Общепринято длительность этих отрезков называть «мертвым временем» ШИМ (dead time). »[2]

Алгоритм простой:

- если значение ШИМ меньше «мертвого времени», то ШИМ равен 0.

- если значение ШИМ больше разницы максимального ШИМ и «мертвого времени», то ШИМ равен максимальному значению.

Реализация этого алгоритма в программе выглядит так:

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

Задано «мертвого времени» равным 500 нс:

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

1.6 Выключение регулятора

«Регулятор должен медленно изменять мощность на элементе Пельтье. Но это не касается аварийного выключения. При setPower=0 будем выключать регулятор мгновенно. »[2]

«Полностью программный блок интегрального регулятора мощности выглядит так: »[2]

«//---------- 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 }»[2]

«Его целая часть соответствует коэффициенту заполнения ШИМ. Поэтому значение коэффициента я на компьютер не передает. »[2]

«Serial.print(" p="); Serial.print(regPwrInt, 2); //интегральное звено регулятора мощности»[2]

1.7 Проверка и настройка регулятора

На рисунке 6 показана схема работы регулятора на реальной нагрузке и определим интегральный коэффициент, нагрузка 20 Ом.

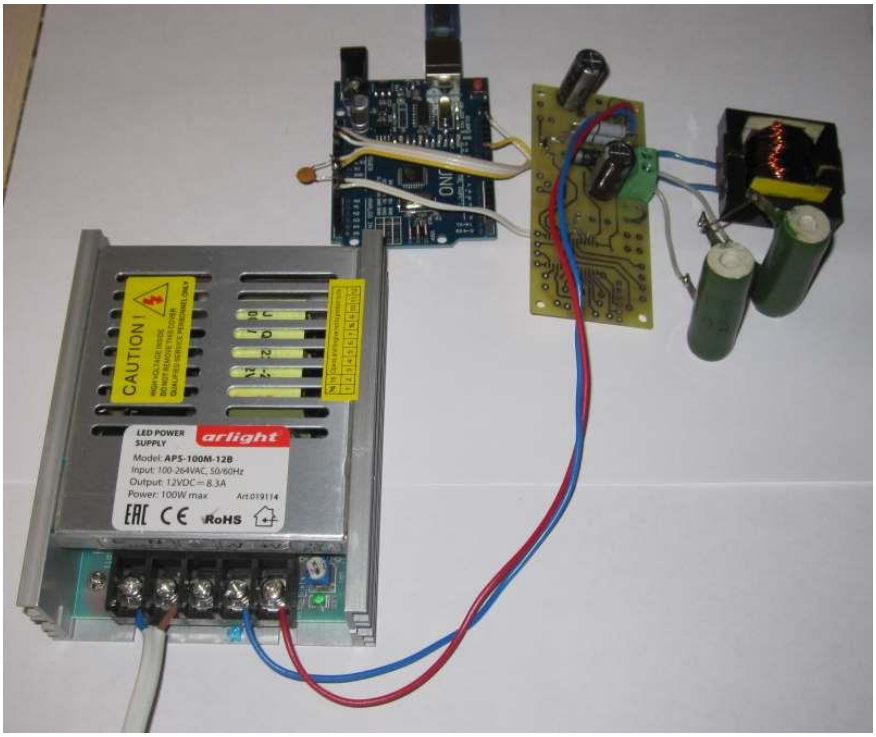

Рисунок 6- Схема работы регулятора на нагрузке

«В качестве средства контроля работы регулятора будем использовать монитор последовательного порта. Установливаем скорость 19200 бод. Сначала задаем интегральный коэффициент равным 0,1. »[2]

#define koeffRegPwrInt 0.1 //интегральный коэффициент регулятора мощности

Запустили монитор последовательного порта и увидели такую картину (рисунок 7).

«Работает устойчиво. Об устойчивости надо судить по изменению целой части интегрального звена, т.е. ШИМ. В идеальном регуляторе ШИМ должен меняться на 1. Ток, а значит, и мощность «скачут» из-за не очень

точной дискретизации АЦП при малых значениях. Установили интегральный коэффициент равным 1 и увидели следующее (рисунок 9). »[2]

Мощность устанавливается на заданном уровне примерно за 1 сек. и регулятор продолжает работать устойчиво.

Теперь коэффициент 10 (рисунок 10).

«Работает еще быстрее, но ШИМ начал «скакать» на 2-5 единицы. Регулятор работает неустойчиво. Такой коэффициент использовать нельзя. »[2]

Чисто в демонстративных целях задаем коэффициент равным 20 (рисунок 10).

Колебания достигли 30 единиц ШИМ. Вот осциллограмма напряжения на выходе. Все пошло в разнос (рисунок 11).

«Регулятор устанавливает заданную мощность 5 Вт за 30 сек. (рисунок 13). При большей мощности будет работать еще медленнее. Но тем лучше для элемента Пельтье, да и коэффициент всегда можно изменить. »[2]

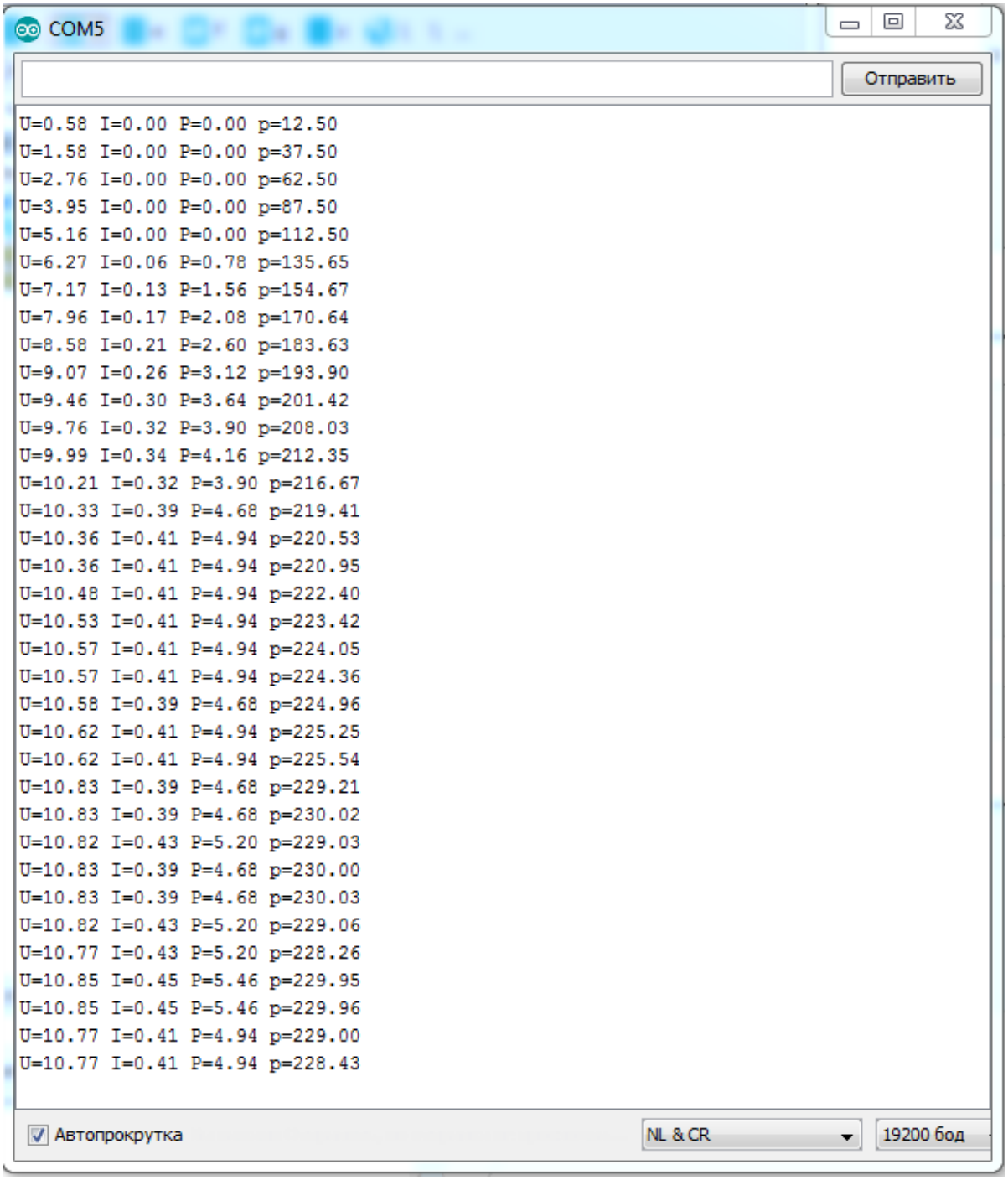

Рисунок 7- Работа регулятора интегральный коэффициент равным 0,1

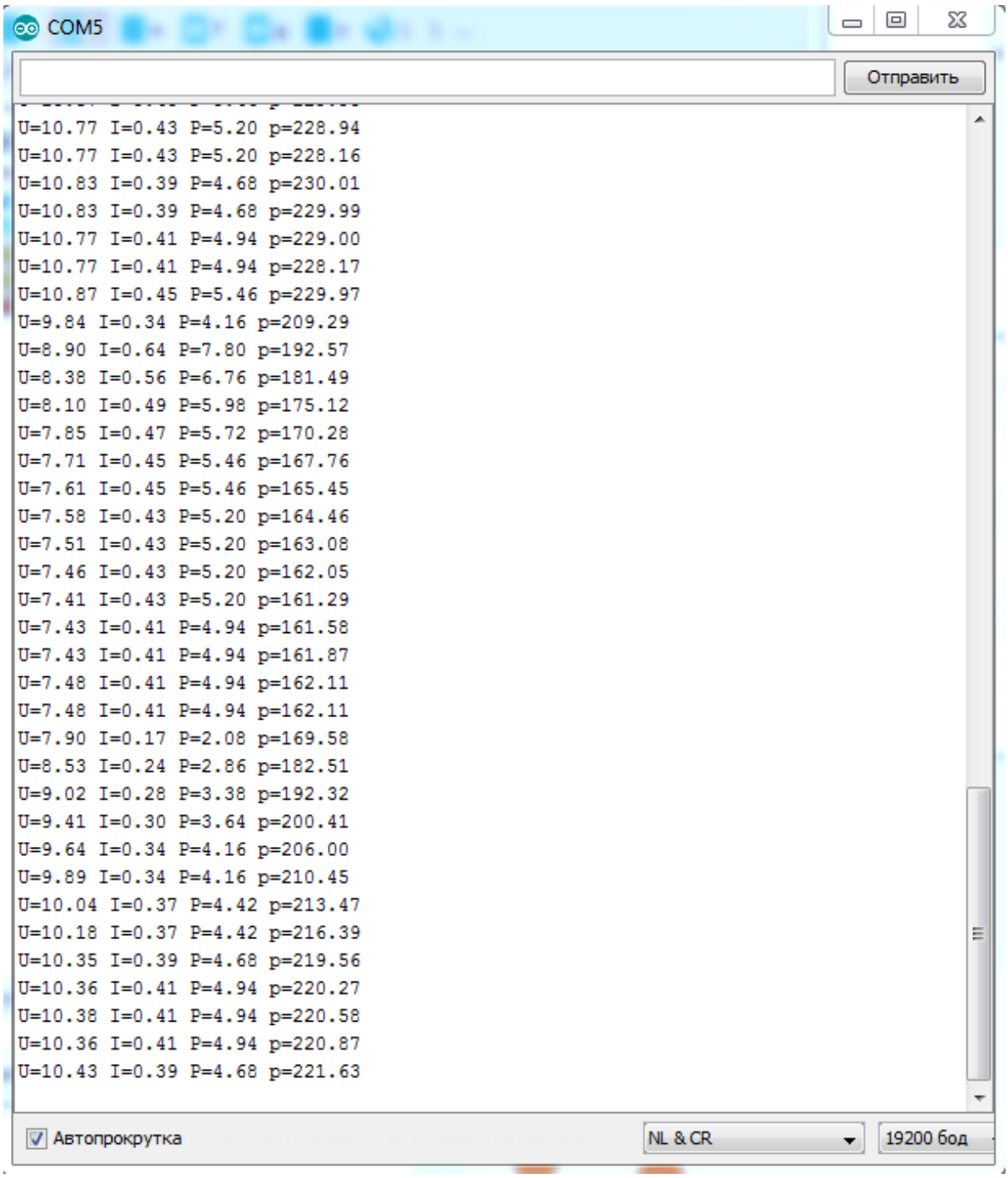

Рисунок 8 – Работа регулятора

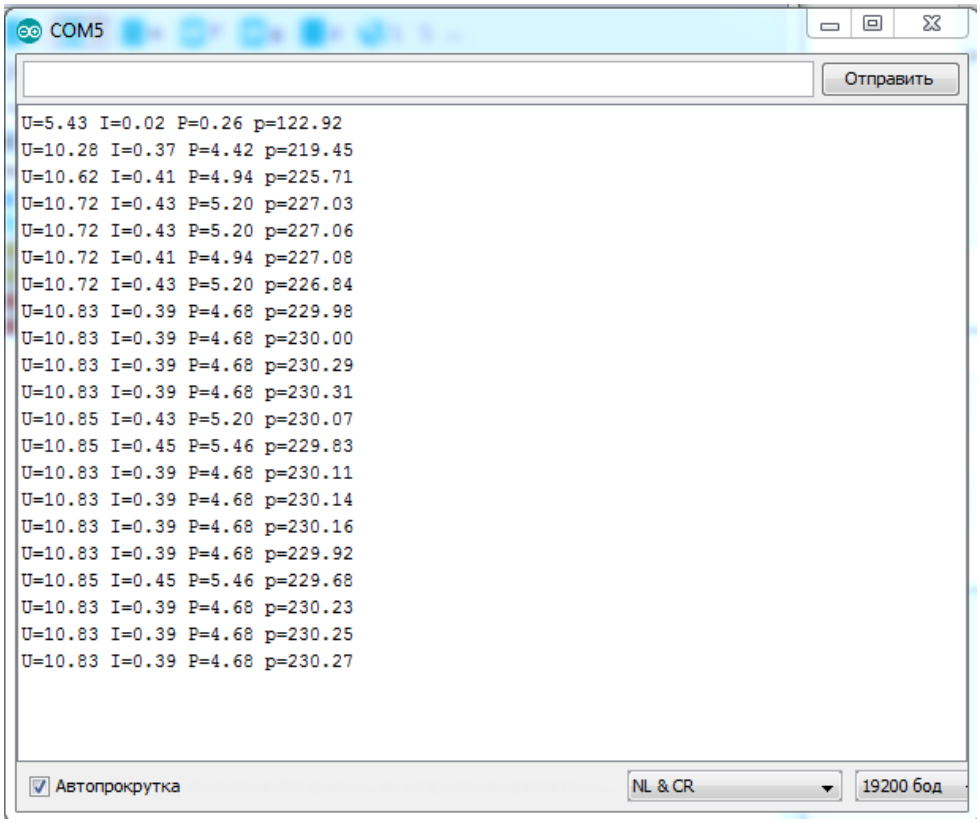

Рисунок 9 - Интегральный коэффициент равным 1

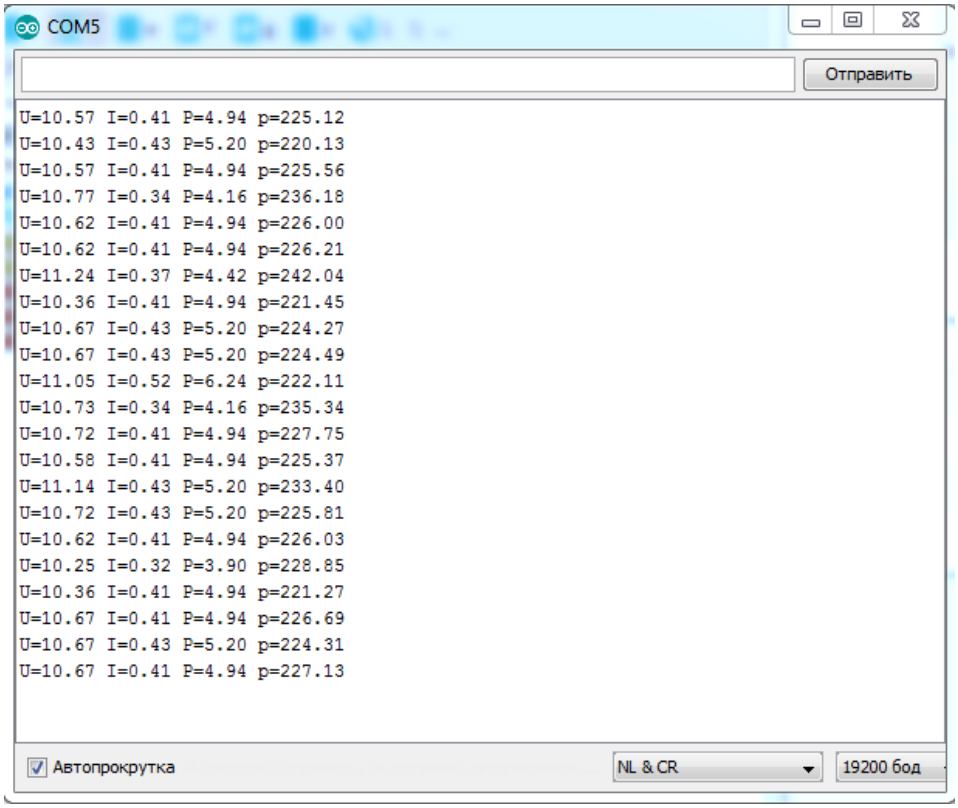

Рисунок 10 - Интегральный коэффициент равным 10

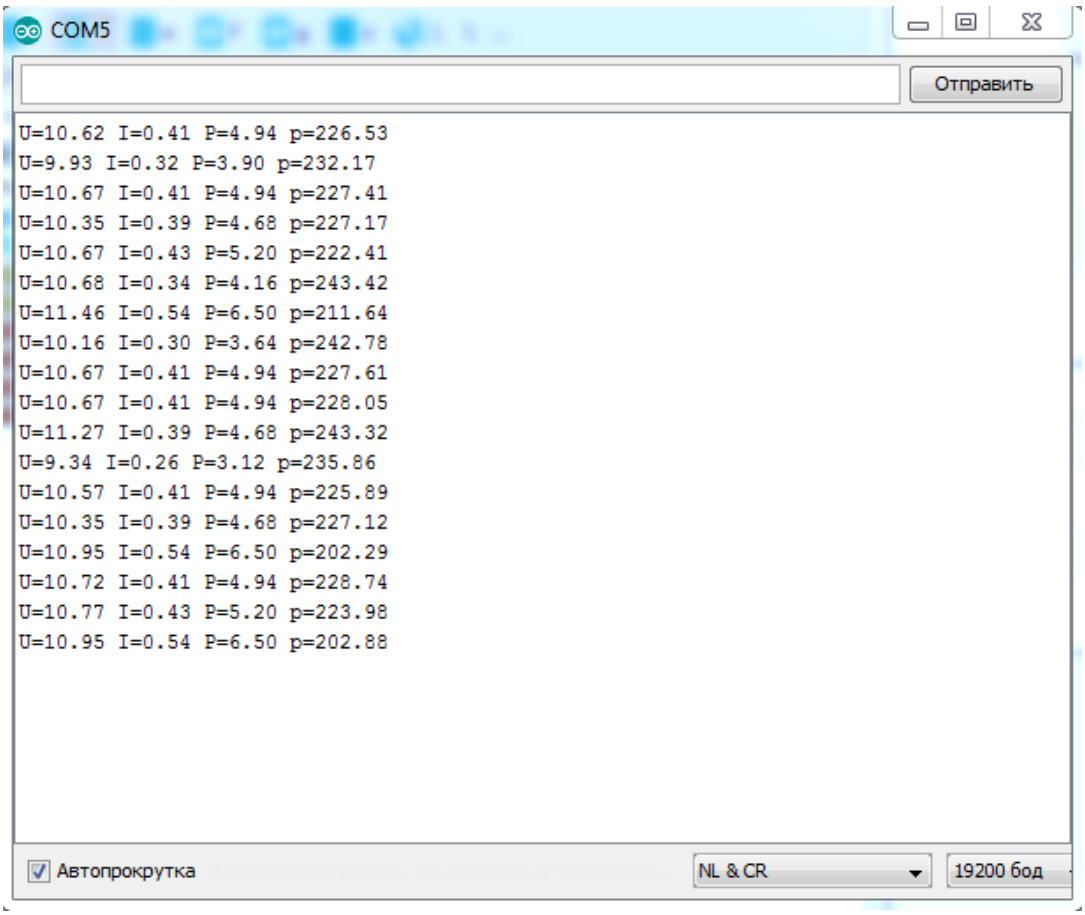

Рисунок 11- Интегральный коэффициент равным 20

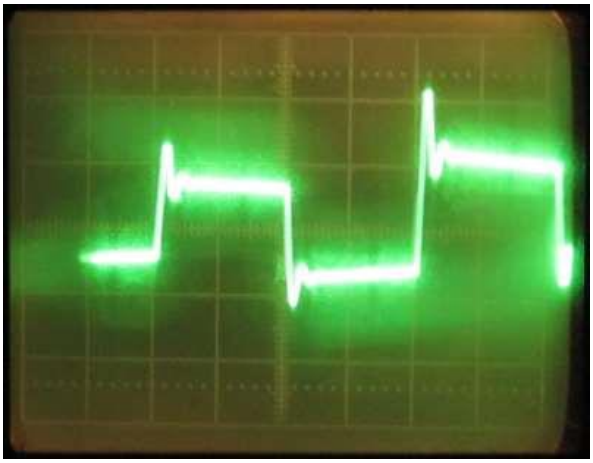

Рисунок 12- Осциллограмма напряжения на выходе

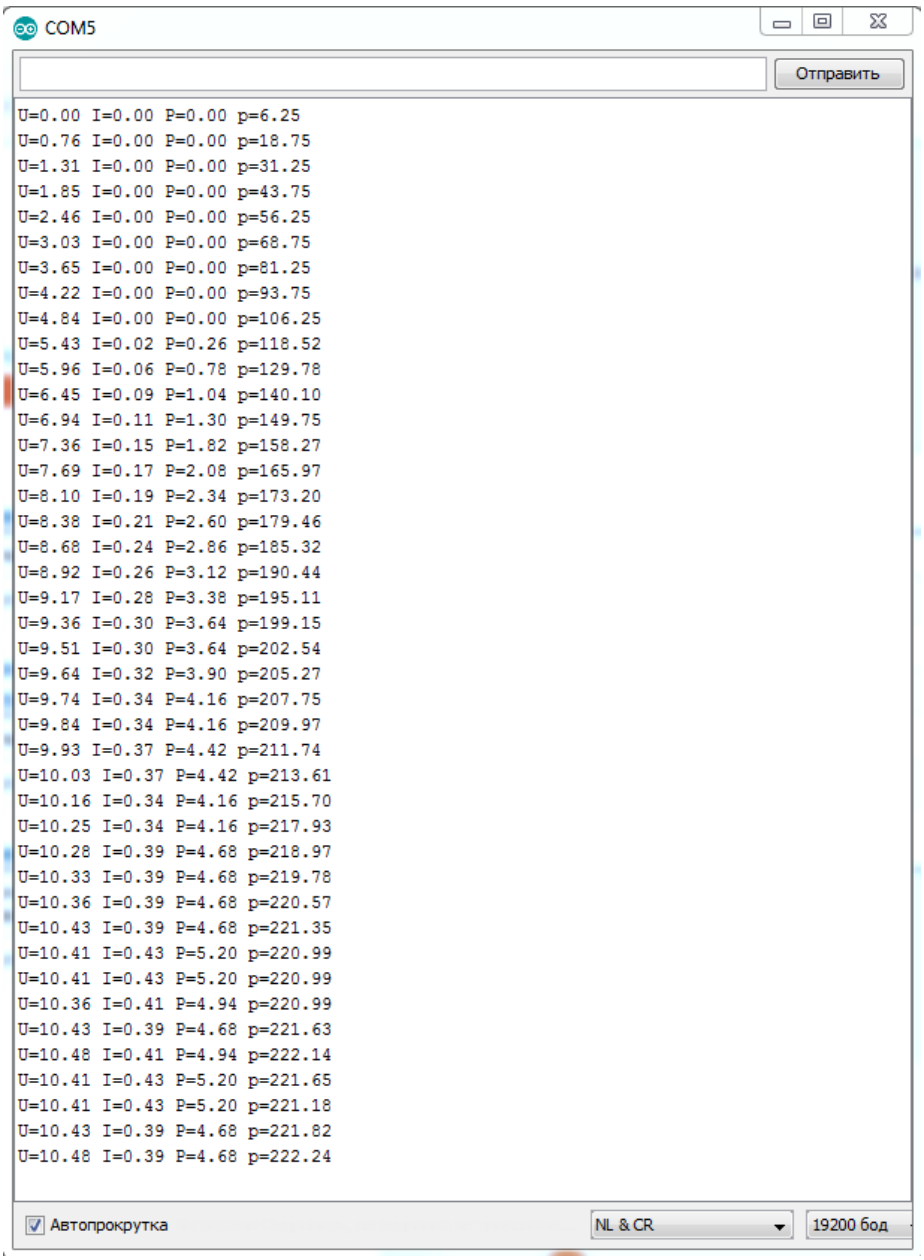

Рисунок 13- Регулятор мощностью 5 Вт за 30 сек

Вывод: В данном разделе была обоснована необходимость разработки холодильника на основе элементах Пельтье. Были сформулированы необходимые условия для работы элементов в составе холодильника – точность регулирования, необходимость построения системы слежения за температурой. Было выбрано решение в виде ПИД-регулятора для поддержания температуры на заданном уровне для чего проанализированы принципы построения аналогичных систем.

2 Основная часть

«Основная задача контроллера холодильника – поддержание в камере заданной температуры. Делать это будет регулятор температуры за счет изменения электрической мощности на модуле Пельтье (рисунок 14). »[2]

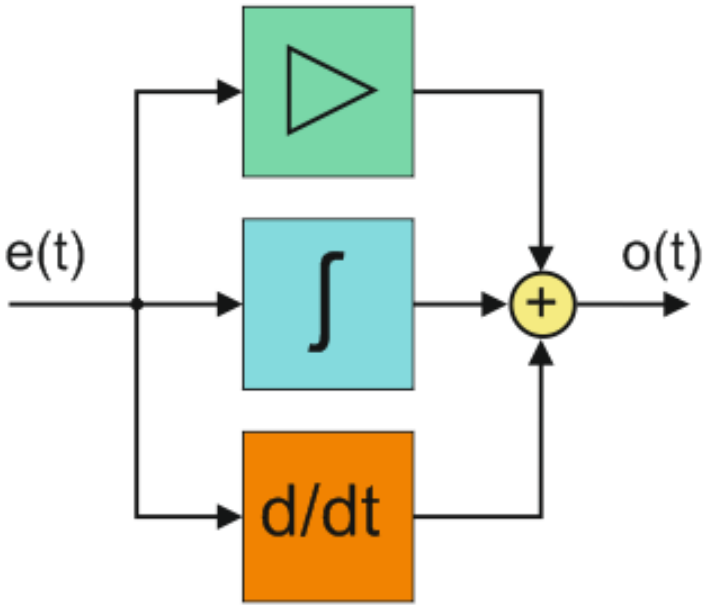

Рисунок 14- Разработка контроллера модуля Пельтье.

Связь регуляторов мощности и температуры выглядит на рисунке 15.

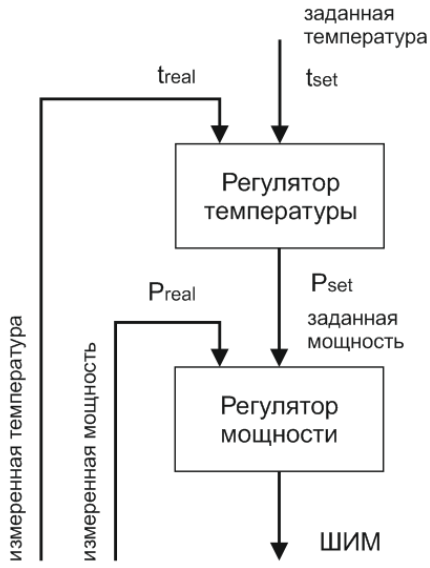

Рисунок 15 - Связь регуляторов мощности и температуры

«Регулятор температуры получает измеренную температуру, сравнивает ее с заданной температурой и вычисляет значение заданной мощности для регулятора мощности. Регулятор мощности формирует ШИМ, соответствующий заданной мощности. Регулятор мощности строится по интегральному закону регулирования. Для стабилизации температуры применяют более сложный алгоритм управления – пропорционально-интегрально-дифференцирующий (ПИД) регулятор. »[2]

2.1 ПИД регулятор

«Регулятор, работающий по такому принципу, обладает высокой точностью. Остальные критерии качества регулирования - быстродействие и устойчивость - у него не на высоте. »[2]

«Для того чтобы добиться высоких показателей для всех критериев необходимо использовать регулятор, объединяющий в себе разные законы регулирования. »[2]

«Именно таким устройством является пропорционально-интегрально-дифференцирующий (ПИД) регулятор. Он формирует выходной сигнал, являющийся суммой трех составляющих с разными передаточными характеристиками. Благодаря этому ПИД регулятор обеспечивает высокое качество регулирования и позволяет оптимизировать управление по отдельным критериям. »[2]

«В формировании выходного сигнала ПИД регулятора участвуют: »[2]

- пропорциональная составляющая - значение пропорционально ошибке рассогласования (разности заданного и реального значений регулируемого параметра).
- интегрирующая составляющая - интеграл ошибки рассогласования.
- дифференцирующая составляющая - производная ошибки рассогласования.

Математическая форма записи закона ПИД регулятора имеет вид (формула 2):

$$o(t) = P + I + D = K_p e(t) + K_i \int e(t)dt + K_d de(t)/dt \quad (2),$$

где o(t) – выходной сигнал;

P – пропорциональная составляющая;

I – интегрирующая составляющая;

D – дифференцирующая составляющая;

Kp, Ki, Kd – коэффициенты пропорционального, интегрирующего, дифференцирующего звеньев;

e(t) – ошибка рассогласования.

«В схематичном виде ПИД регулятор можно представлен на рисунке 16. »[2]

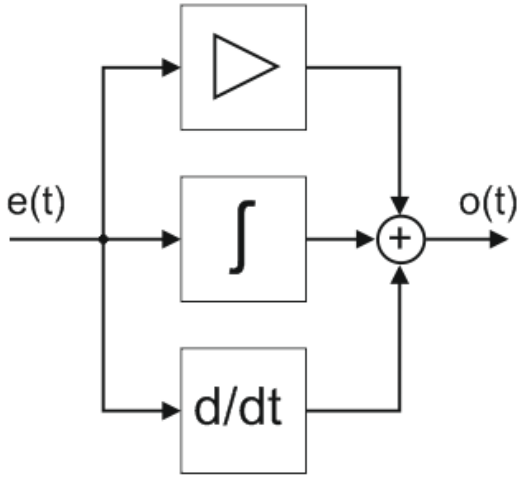

Рисунок 16- Схема ПИД регулятора

Структурная схема ПИД регулятора напряжения U показан на рисунке 17.

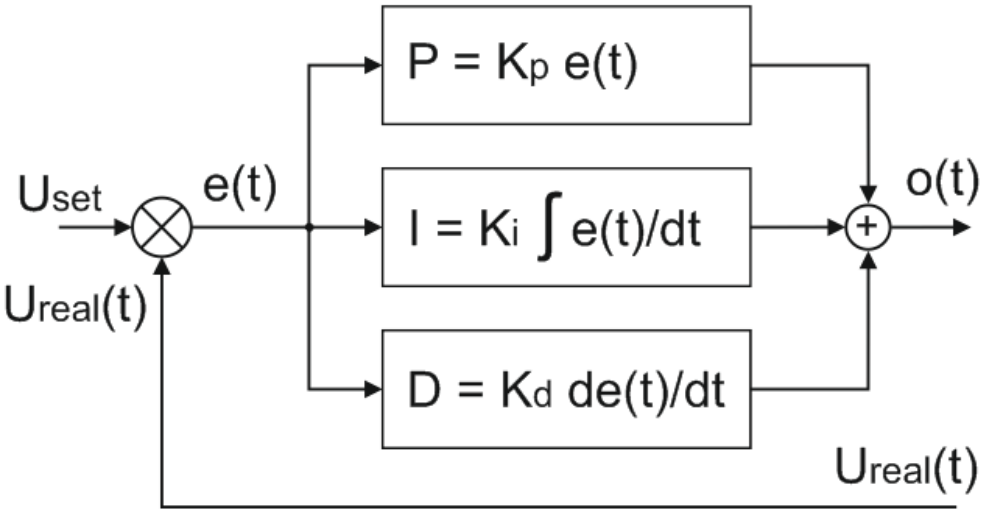

Рисунок 17- Структурная схема ПИД регулятора напряжения U

«Измеренное напряжение Ureal(t) вычитается из заданного Uset. Полученная ошибка рассогласования e(t) поступает на пропорциональное, интегрирующее и дифференцирующее звенья. В результате суммы составляющих получается управляющее воздействие o(t), которое подается на регулирующий элемент. »[2]

«При программной реализации ПИД регулятора вычисления выходного сигнала происходят через равные промежутки времени. Т.е. регулятор является дискретным по времени. »[2]

2.2 Составляющие ПИД регулятора

«Выходной сигнал ПИД регулятора это сумма трех составляющих: »[3]

- пропорциональной;
- интегрирующей;
- дифференцирующей.

Пропорциональная составляющая (формула 3).

$$P(t) = K_p \times e(t) \quad (3)$$

«Не имеет памяти, т.е. значение выходного сигнала не зависит от предыдущего состояния системы. Просто ошибка рассогласования, умноженная на коэффициент, передается на выход. Выходной сигнал компенсирует отклонение регулируемого параметра. Сигнал тем больше, чем больше ошибка рассогласования. При ошибке равной 0, сигнал на выходе тоже равен 0. «[3]

«Пропорциональная составляющая не способна компенсировать ошибку полностью. Это видно из формулы. Выходной сигнал в Kp раз больше ошибки. Если ошибка рассогласования равна 0, то и выходной сигнал регулятора равен 0. А тогда и компенсировать нечем. »[3]

«Поэтому в пропорциональных регуляторах всегда существует так называемая статическая ошибка. Уменьшить ее можно за счет увеличения коэффициента Kp, но это может привести к снижению устойчивости системы и даже к автоколебаниям. »[3]

К недостаткам пропорциональных регуляторов следует отнести:

- наличие статической ошибки регулирования;
- невысокая устойчивость при увеличении коэффициента.

Есть весомое преимущество:

- высокая скорость регулирования;
- реакция пропорционального регулятора на ошибку рассогласования ограничена только временем дискретизации системы.

«Регуляторы, работающие только по пропорциональному закону, применяют редко. Главная задача пропорциональной составляющей в ПИД регуляторе – повысить быстродействие. »[3]

2.3 Интегрирующая составляющая

$$I(t) = K_i \int e(t)dt \quad (4)$$

«Пропорционально интегралу ошибки рассогласования. С учетом временной дискретности регулятора можно написать так (формула 4 и 5): »[3]

$$I(t) = I(t_{-1}) + K_i * e(t) \quad (5)$$

I(t-1) – значение I в предыдущей точке временной дискретизации.

«Ошибка рассогласования умножается на коэффициент и прибавляется к предыдущему значению интегрирующего звена. Т.е. выходной сигнал все время накапливается и со временем увеличивает свое воздействие на объект. Таким образом, ошибка рассогласования полностью компенсируется даже при малых значениях ошибки и коэффициента Ki. В установившемся состоянии выходной сигнал регулятора полностью обеспечивается интегрирующей составляющей. »[3]

«К недостаткам интегрального регулятора следует отнести: »[3]

- низкое быстродействие;
- посредственная устойчивость.

Достоинство:

- способность полностью компенсировать ошибку рассогласования при любом коэффициенте усиления.

«На практике часто используют интегрирующие регуляторы (только интегрирующая составляющая) и пропорционально-интегрирующие (интегрирующая и пропорциональная составляющие). »[3]

«Главная задача интегрирующего звена в ПИД регуляторе – компенсация статической ошибки, обеспечение высокой точности регулирования. »[3]

2.4 Настройка ПИД регулятора

«Качество регулирования ПИД регуляторов в значительной мере зависит от того, насколько оптимально выбраны коэффициенты. Коэффициенты ПИД регулятора определяются на практике в системе с реальным объектом путем подбора. Существуют разные методики настройки. О качестве регулирования судят по переходной характеристике регулятора, т.е. по графику изменения регулируемого параметра во времени. »[3]

«Критерии оптимизации могут быть разными. В общем случае ПИД регуляторы настраивают для обеспечения всех критериев качества регулирования на высоком уровне. Составляющие ПИД регулятора настраиваются отдельно. Отключается интегрирующее и дифференцирующее звенья и выбирается коэффициент пропорционального звена. Если регулятор пропорционально-интегрирующий (отсутствует дифференцирующее звено), то добиваются полного отсутствия колебаний на переходной характеристике. При настройке регулятора на высокое быстродействие колебания могут остаться. Их попытается скомпенсировать дифференцирующее звено. Подключается дифференцирующее звено. Его коэффициентом стремятся убрать колебания параметра регулирования. Если не удается, то уменьшают пропорциональный коэффициент. За счет интегрирующего звена убирают остаточную ошибку рассогласования. »[3]

«Настройка ПИД регулятора носит итерационный характер, т.е. пункты подбора коэффициентов могут многократно повторяться до тех пор, пока не будет достигнут приемлемый результат. »[3]

«Благодаря высоким характеристикам и универсальности ПИД регуляторы широко применяются в системах автоматизации производства. »[3]

2.5 Разработка контроллера элемента Пельтье. ПИД регулятор температуры

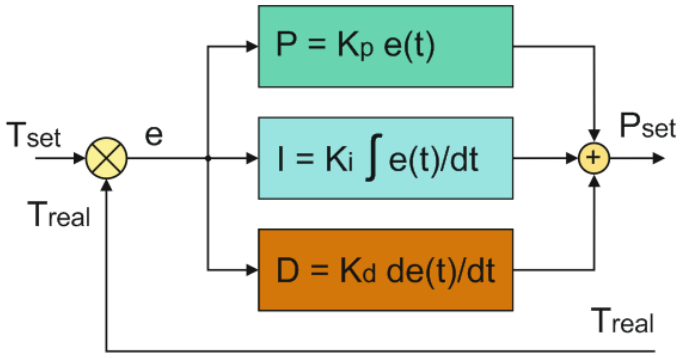

Рисунок 18 -ПИД регулятор температуры

«Регулятор температуры выполняет главную функцию холодильника - стабилизацию температуры в камере. Именно он во многом определяет основные параметры устройства: точность поддержания температуры и скорость реакции на возмущающие воздействия. Поэтому будем реализовывать регулятор температуры по закону ПИД регулирования. Регулятор температуры получает измеренное значение температуры в камере, сравнивает его с заданным и формирует значение заданной мощности для регулятора мощности. Итак, в регуляторе температуры (рисунок 18): »[3]

- регулируемый параметр - температура в камере;
- регулирующий элемент - заданная мощность для регулятора мощности.

«Для того, чтобы стабилизировать температуру надо ее измерять. Поэтому первое, что добавим в программу - это измерение температуры в камере. Заодно реализуем измерение температуры радиатора горячей стороны элемента Пельтье. »[3]

2.6 Проектирование схемы

Анализируя принципы регулирования, а также техническое задание и функции, которые необходимо получить от данного устройства было проведено синтезирование принципиальной схемы устройства (рисунок2.6). Для удобства вывода текущей информации о работе устройство удобно использовать LCD индикатор A2. В наборах Arduino часто для уменьшения количества выводов идущих от Arduino в направлении индикатора используют специальные микросхемы, осуществляющие преобразование интерфейса последовательный / параллельный. Для этой цели в работе предлагается использовать специальную микросхему DD4 преобразующую интерфейс i2c в параллельный необходимый для управления LCD индикатором.

Для измерения температуры горячей холодной стороны удобно использовать датчики температуры DD2,3 с цифровым интерфейсом, подключаемым также по последовательной шине. Такое включение позволяет экономить на количестве подключаемых проводов. Ввод информации в Arduino удобно проводить с помощью энкодера DD1, который позволяет увеличивать или уменьшать значение параметров методом вращения ручки влево или вправо, а также выбирать пункты меню нажатием этой ручки, которая осуществляет также функцию кнопки.

Регулирование мощности или напряжения на элементе пельтье можно осуществить с помощью полевого транзистора VT1, который может управляться непосредственно с ШИМ вывода Arduino.

2.6.1 Описание микросхемы интерфейса I2C (PCF8574)

Микросхема PCF8574 представляет собой расширитель портов на сдвиговых регистрах, особенностью этой микросхемы является наличие шины i2c (рисунок 19). Таким образом, для работы этой микросхемы необходимо всего две линии сигнала, которые подключаются к встроенным

в Arduino блоку работы с шиной i2c, 19 и 18 пины , что соответствует линии SCL и SDA.

Адрес устройства формируется подачи на на входы адреса 3 битов A0A1A2, для упрощения можно задать адрес устройства равный 000, тогда все эти входы должны быть подключены к общему выводу. Питание микросхемы составляет 5 вольт, а ток потреблени около 100 мкА, поэтому его можно взять с питания Arduino. Определимся, что выходные биты P0-P3 биты информации вывода на индикатор, биты P4-P6 биты управления состоянием индикатора.

Это микросхема также позволяет также формировать сигнал прерывания с вывода INT в сторону контроллера Arduino при изменении состояния на её пинах. Однако мы не будем использовать эту функцию, поскольку от этой микросхемы требуется только лишь передача сигнала от Arduino в сторону индикатора.

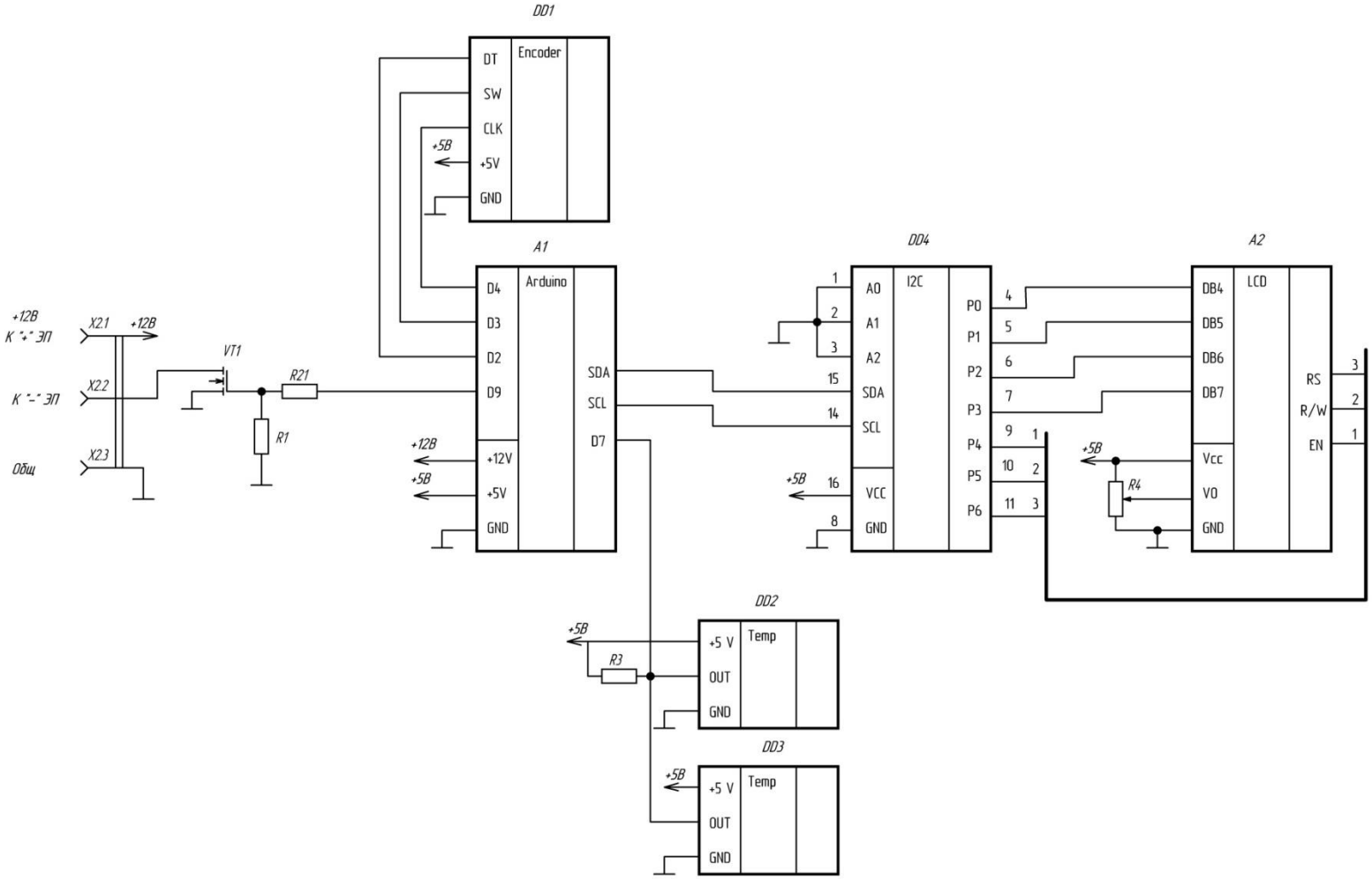

Рисунок 19 - Принципиальная схема холодильника.

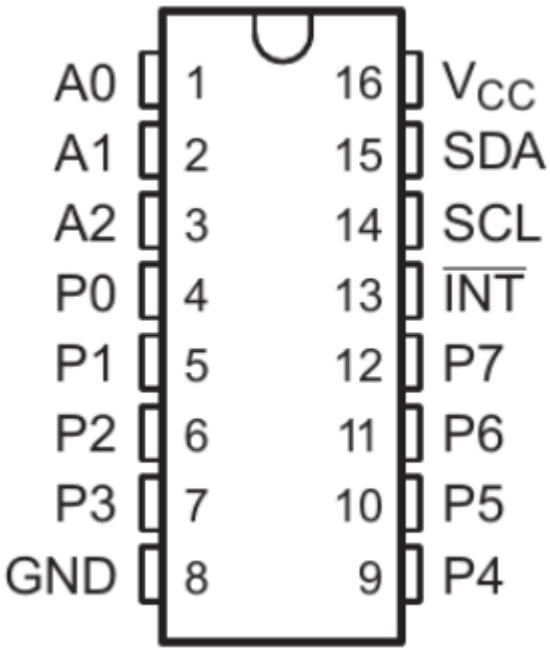

Рисунок 20 – Распиновка микросхемы PCF8574

2.6.2 Описание дисплея

Для вывода информации о работе системы удобно использовать жидкокристаллические индикаторы (дисплеи) удобство заключается в том, что дисплей содержит достаточно большое количество символов в строке 8-16 в зависимости от его типа. Из доступных LCD индикаторов чаще всего в проектах используют LCD1602. Он позволяет ввиду наличия готовых библиотек текст, а также другую символическую информацию в две строчки. Каждый его элемент представляет собой матрицу из пикселей размерами 5 на 7. На борту индикатора содержится специальный контроллер, который позволяет дешифрировать сигналы, приходящие по параллельной шине от контроллера. Шина содержит биты интерфейса DB0-DB7, пины питания VDD, VSS, пин управления контрастом дисплея V0. А также ряд пинов для

управления работой индикатора: выбор регистра RS, выбор режима записи или чтения RW, строб E.

а)

б)

Рисунок 21 – Внешний вид индикатора LCD1602: а) переди, б) сзади.

Ток потребления контроллера составляет 1,1мА , а подсветки до 100мА в зависимости от настроенной яркости. При питании дисплея надо учитывать, что питание Ардуино от USB позволет выдать ток до 500мА (минимум). При внешнем питании (максимальном) 12В, выделяемая в стабилизаторе напряжения +5В мощность от тока потребления 100мА равна 0,1*(12-5)=0,7 Вт, может привести к перегреву. Поэтому рекомендуется не подавать такое напряжение, а питать от меньшего напряжения.

2.6.3 Описание энкодера (KY-40)

Для управления работой системы используют энкодер KY-40 (рисунок 22). Это популярный анкодер, который используется в проектах Arduino.

«Подключается модуль энкодера следующим образом: питание на питание (GND и VCC), логические пины CLK, DT (тактовые выводы

энкодера) и SW (вывод кнопки) на любые пины Arduino (D или A). У круглых модулей выводы энкодера подписаны как S1 и S2, а вывод кнопки как Key, подключаются точно так же. От порядка подключения тактовых выводов энкодера зависит «направление» его работы, но это можно поправить в программе. »[3]

«У модулей энкодера тактовые выводы подтянуты к питанию и дают низкий сигнал при срабатывании, на них нестоят RC цепи для гашения дребезга. Промышленный энкодер подключается точно так же, чёрный и красный провода у него питание, остальные – тактовые выходы DT, CLK. Однако в целом можно подключить энкодер напрямую без обвязки с RC-цепями, как на схеме ниже, есть библиотеки например GyverEncoder которые отработывают и подтяжку средствами микроконтроллера (INPUT_PULLUP), и программный антидребезг. »[3]

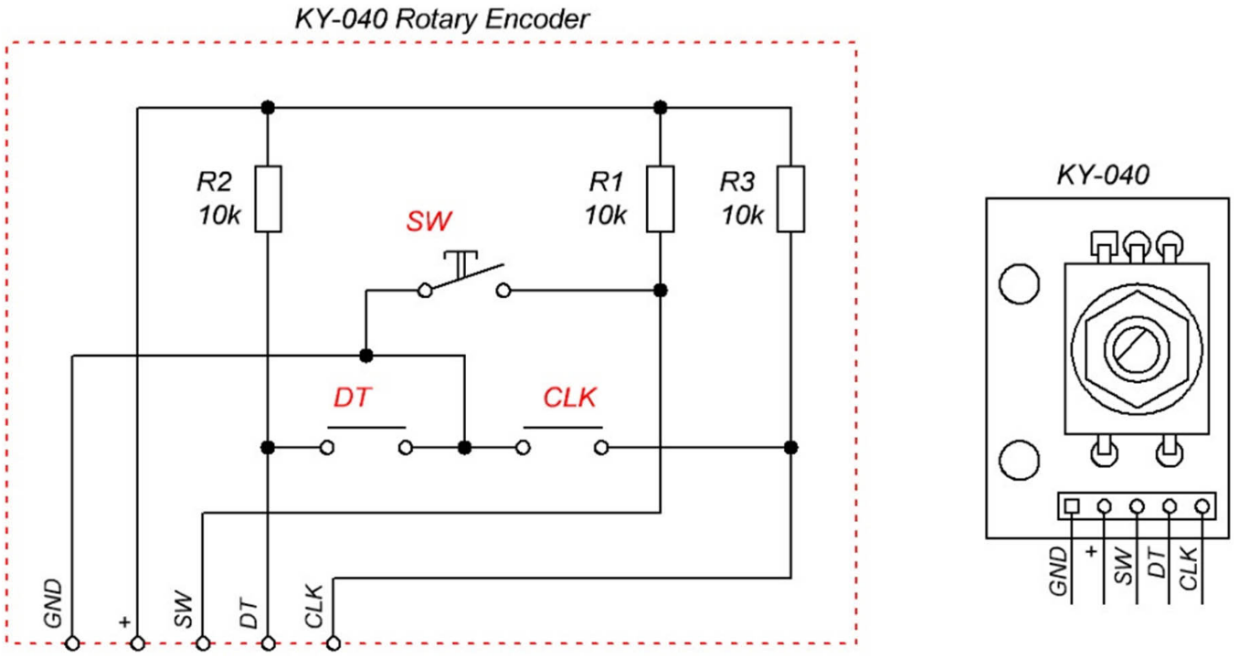

Рисунок 22 – Внутренняя схема и внешний вид энкодера KY-40.

2.6.4 Описание датчика температруры

Для измерения температуры горячей и холодной части элемента Пельтье используем цифровой датчик температуры DS18B20 (рисунок2.10). Этот датчик имеет приемлемую точность (+-0,5 град), а также имеет достаточный диапазон температур (-55...+125С) наблюдаемый при работе холодильника. Предполагается, что минимальная температура будет не ниже 0 градусов, а максимальная на горячей стороне не выше 60 градусов. Основные параметры датчика температуры приведены ниже.

Диапазон: -55.. 125 °C

Точность: 0.5 °C

Разрешение: 9.. 12 бит (0.48.. 0.06 °C)

Питание: 3-5.5V

Период выдачи результата:

750 мс при точности 12 бит

94 мс при точности 9 бит

Интерфейс связи: 1-Wire (OneWire)

Корпус: TO-92, SOIC-8 или герметичное исполнение

Рисунок 23 – Варианты исполнения датчика температуры.

Для работы с датчиком Также имеется множество библиотек которое позволяет подключать его к системе Arduino, например, библиотека

DallasTemperature.h, для работы которой также понадобится библиотека OneWire.h.

2.6.5 Описание коммутатора мощности нагрузки

Для управления мощностью напряжение на нагрузке в элементе пельтье можно использовать полевой транзистор. Необходимо, чтобы для его управления было достаточно на напряжение с выхода Arduino, то есть плюс 5 Вольт. Такие ключи называются полевыми транзисторами с логическим управлением. Учитывая, что максимальный ток, который может протекать через элемент пельтье в соответствии с его документацией не превышает 6 ампер, можно с некоторым запасом выбрать полевой транзистор типа IRFZ44N (рисунок24). Максимальный ток этого транзистора не превышает 35 ампер при температуре корпуса порядка 100С. Частоту ШИМ выберем исходя из двух условий. С одной стороны, частота должна быть достаточной, чтобы считать период ШИМ много меньше постоянной времени тепловых процессов в холодильнике с другой стороны частота ШИМ должна быть невысокой для того, чтобы динамические потери в ключе были несущественными. Из этого условия можно выбрать стандартную частоту ШИМ, которую формирует Arduino без дополнительной настройки таймеров, то есть частоту равную 390 Герц.

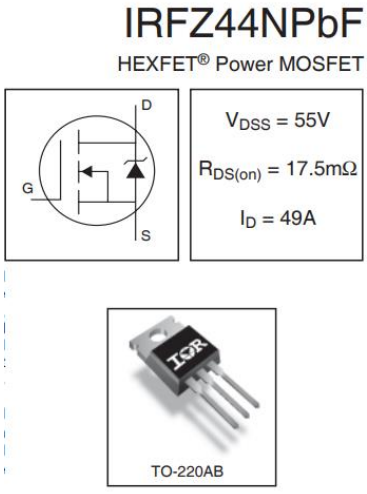

Рисунок 24 – Основные параметры и внешний вид транзистора IRFZ44N.

Для предотвращения ситуаций с обрывом управления полевым транзистором на затворе транзистора параллельно затвору и истоку необходимо установить резистор 51кОм, в затворную цепь для ограничения тока зряда затворной цепи и установления скорости открытия и закрытия транзистора в его цепь необходимо установить резистор 5кОм. При этом резисторе ток выхода ШИМ будет ограничен i=1мА, что допустимо для выходного тока Ардуино. Время включения и выключения транзистора при заряде Q=10нКл, взятой из параметров транзистора будет равн $t = \frac{Q}{i} = \frac{10^{-9}}{0{,}001}$ $=10^{-6}$с. Таким образом , время сучщественно меньше периода и коммутация близка к идеальной.

2.6.6 Измерения температуры горячей и холодной стороны

«Температуру измеряем интегральными датчиками DS18B20. Подключим датчики DS18B20 к плате Ардуино по стандартной схеме. Собранная схема выглядит на рисунке (рисунок 19). »[3]

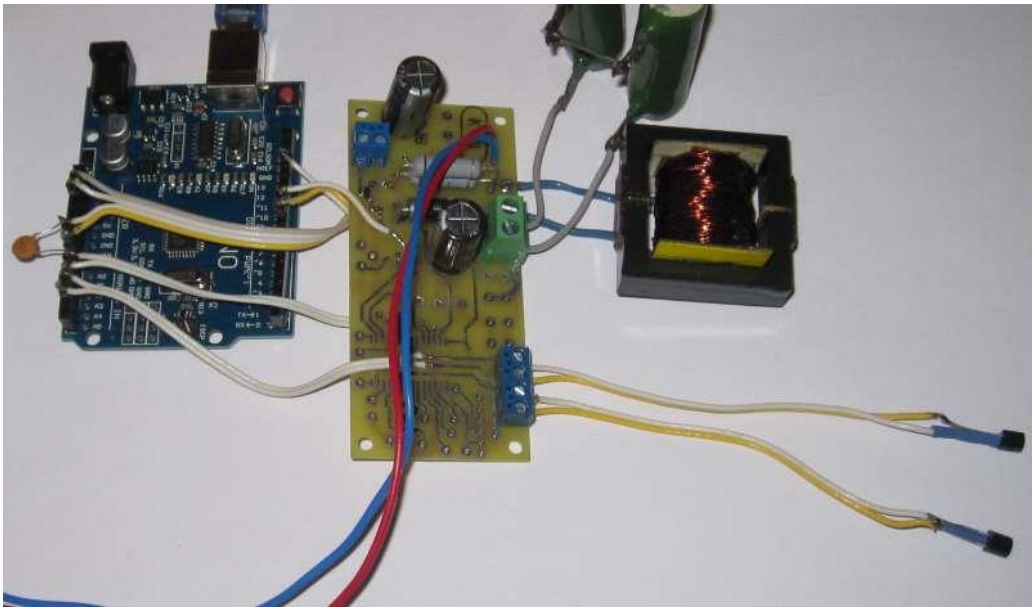

Рисунок 25 – Собранная схема

Подключим библиотеку OneWire:

#include <OneWire.h>

Добавим переменные для измеренных температур:

«float measureTempRef; //измеренная температура в камере (< -200 - ошибка) float measureTempRad; //измеренная температура радиатора (< -200 - ошибка) »[3]

Создадим объекты OneWire (датчики температуры).

«OneWire sensTempRef (16); //датчик температуры в камере подключен к выводу 16 (A2) OneWire sensTempRad (17); //датчик температуры радиатора подключен к выводу 17 (A3) »[3]

«При измерении температуры могут возникнуть аппаратные ошибки датчиков DS18B20. Для индикации этих ошибок не будем заводить отдельные признаки. Значение температуры менее - 200 C° говорит об аппаратной ошибки измерения. »[3]

«Последовательность действий для измерения температуры с помощью датчиков DS18B20 состоит из трех операций: »[3]

- инициализация измерения температуры;
- пауза не менее 750 мс;
- чтение результата измерения из датчика и проверка контрольной суммы.

«Инициализация измерения температуры реализуется 3 функциями библиотеки OneWire: »[3]

«sensTempRef.reset(); //сброс шины 1-Wire sensTempRef.write(0xCC, 1); //пропуск ROM sensTempRef.write(0x44, 1); //инициализация измерения» [3]

«Чтение результата измерения из датчика и проверки контрольной суммы требует следующих функций: »[3]

«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 if (OneWire::crc8(bufData, 8) == bufData[8]) { //проверка CRC »[3]

«Понятно, что блок инициализации надо выполнить в начале длинного цикла (1 сек) программы, а блок чтения результата ближе к концу длинного цикла. Время между выполнением этих блоков должно быть не менее 750 мс. »[3]

«Но вопрос заключается в том, сколько времени требуется на выполнение функций класса OneWire. Можно ли для выполнения операций инициализации измерения и чтения результата использовать по одному циклу 20 мс или необходимо «размазать» их на несколько циклов. Не забудем, что мы измеряем температуру с помощью двух датчиков DS18B20. »[3]

Методика измерения и результаты:

- операция инициализации измерения температуры требует 2063 мкс;

- чтение результата с проверкой контрольной суммы занимает 6,8 мс.

В результате инициализацию измерения температуры для обоих датчиков выполнить в одном цикле 20 мс на интервале 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 }»[3]

«А чтение результата с проверкой контрольной суммы для датчиков температур в камере и радиатора сделал в отдельных интервалах 45 и 46. »[3]

«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 »[3]

« if (OneWire::crc8(bufData, 8) == bufData[8]) { //проверка CRC //правильно measureTempRef= (float)((int)bufData[0] | (((int)bufData[1]) << 8)) * 0.0625 + 0.03125; if ((measureTempRef < MEASURE_MIN_TEMP) || (measureTempRef > MEASURE_MAX_TEMP)) measureTempRef= -300.; //ошибка измерения } else measureTempRef= -300.; //ошибка измерения } »[3]

« 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»[3]

« if (OneWire::crc8(bufData, 8) == bufData[8]) { //проверка CRC // правильно measureTempRad= (float)((int)bufData[0] | (((int)bufData[1]) << 8)) * 0.0625 + 0.03125; if ((measureTempRad < MEASURE_MIN_TEMP) || (measureTempRad > MEASURE_MAX_TEMP)) measureTempRad= -300.; //ошибка измерения } else measureTempRad= -300.; //ошибка измерения }»[3]

«Полученные результаты проверяются еще на верхнее и нижнее максимальное значение. Если температура в нашей системе больше 100 или ниже 40 C° это явная ошибка. »[3]

«Осталось вывести измеренные температуры на компьютер для проверки. »[3]

«Serial.print(" t="); Serial.print(measureTempRef, 2); //температура в камере Serial.print(" t="); Serial.print(measureTempRad, 2); //температура радиатора»[3]

Загружаем скетч в плату, открываем монитор последовательного порта. Все работает.

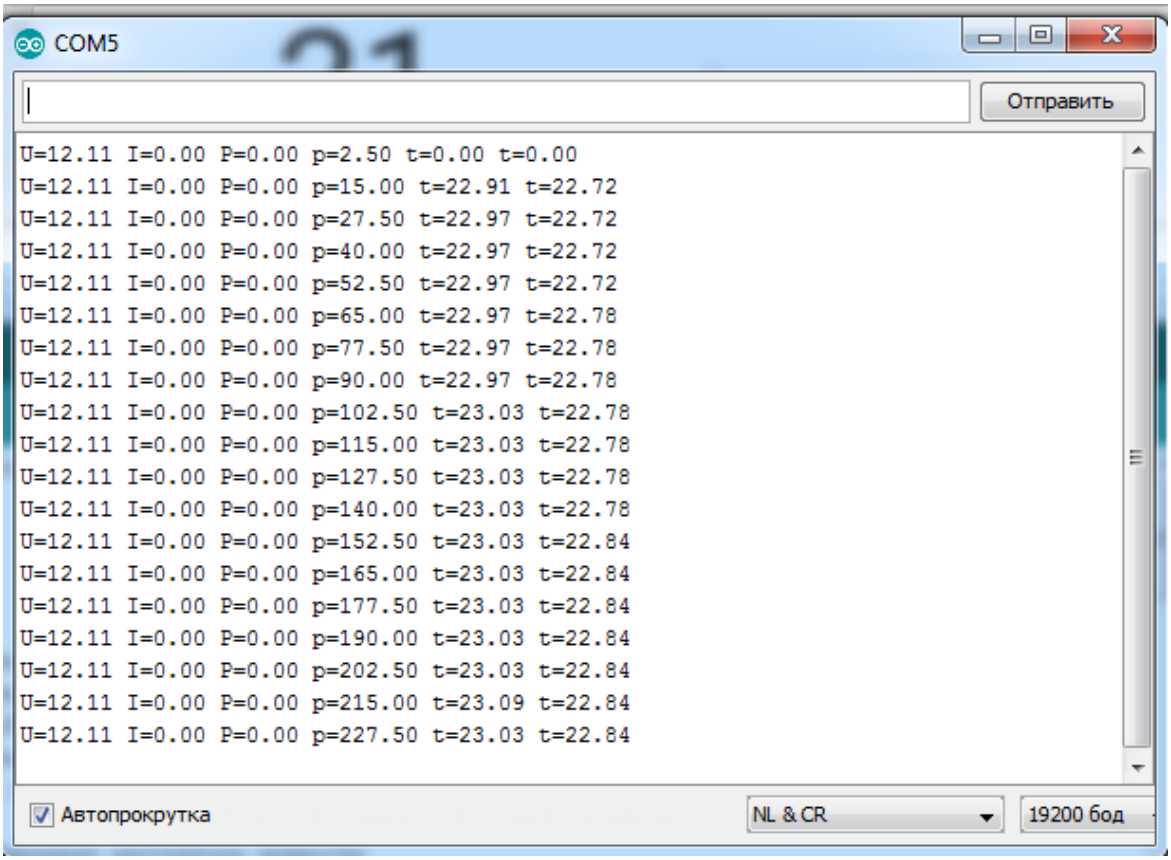

Рисунок 26– Показатели температуры

Температура измеряется правильно (рисунок 26).

2.6.7 Реализация ПИД регулятора температуры

«При разработке регулятора мощности совершили небольшую ошибку. Для быстрого выключения регулятора использовали условие setPower == 0, т.е. при заданной мощности равной 0 регулятор мгновенно выключался, интегральное звено сбрасывалось. Быстрое выключение требуется для реализации в будущем аварийных режимов. »[3]

«Но заданная мощность может на короткое время стать равной 0 под действием дифференцирующей составляющей. Я изменил условие аварийного выключения регулятора мощности: »[3]

if (setPower >= 0) {…}

«Теперь выключение регулятора происходит при любом отрицательном значении заданной мощности. Создаем все переменные и константы, необходимые для регулятора температуры. »[3]

« #define koeffRegTmpInt 0.002 //интегральный коэффициент регулятора температуры#define koeffRegTmpPr 0.5 //пропорциональный коэффициент регулятора температуры#define koeffRegTmpDif 50 //дифференцирующий коэффициент регулятора температуры »[3]

« #define MAX_POWER 5. //максимальная выходная мощность контроллера »[3]

« float regPwrInt=0; //интегральное звено регулятора мощности 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 »[3]

«Константа MAX_POWER определяет максимальную мощность, которую способен создать контроллер. Это аппаратные ограничения на блок питания, ключевой стабилизатор, модуль Пельтье и т.п. »[3]

«Переменная maxSetPower задается пользователем и в любой момент может быть изменена кнопками контроллера. Она вводит дополнительное ограничение на выходную мощность. Например, кто-то хочет ограничить потребляемую мощность для экономии электроэнергии. »[3]

«Сам регулятор выполним в цикле 20 мс на интервале 47, сразу после получения результатов измерения температуры. »[3]

« if (cycle20mcCount == 47) { //---------- интервал 47, регулятор температуры в камере »[3]

setTempRef= 20.; //временно заданная температура 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 }

«В регуляторе мощности при реальной мощности меньше заданной надо было увеличивать ШИМ. Больше ШИМ, больше мощность. В регуляторе температуры мощность надо увеличивать при реальной температуре больше заданной. Охлаждаем, а не нагреваем. И чем больше мощность, тем меньше температура. Остается вывести промежуточные и основные результаты работы регулятора на компьютер. В результате увидели следующее: »[3]

- ошибку рассогласования;
- заданную мощность;
- интегральную составляющую;
- пропорциональную составляющую;
- дифференциальную составляющую.

«Заданная мощность это результат работы регулятора. Все остальное - результаты промежуточных вычислений. Но они показывают работу регулятора и облегчают настройку коэффициентов. »[3]

«С выводом данных на компьютер через последовательный порт все не так просто. Функция Serial.print() помещает данные в передающий буфер

последовательного порта. А встроенный класс Serial передает эти данные через аппаратный интерфейс UART. Но размер буфера Serial не безграничен. »[3]

«Для Arduino UNO R3 размеры буферов приема и передачи составляют 64 байт. Поэтому при большем размере передаваемых данных должны делать паузы на передачу данных через аппаратный интерфейс UAR,. т.е. для того, чтобы данные «ушл» из буфера в UART, скорость передачи данных 19200 бод. Один байт передается за 0,5 мс. За цикл 20 мс будет передано 40 байтов. »[3]

«За один цикл 20 мс все данные не передать. Надо передачу данных на компьютер разбивать на интервалы по 20 мс. ВыбралИ 10 и 11 интервалы. »[3]

« 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 Serial.println(" "); } »[3]

« Регулятор температуры готов. Можем загрузить итоговый скетч программы: »[3]

2.6.8 Выбор коэффициентов регулятора

«Выбор коэффициентов сильно зависит от конструкции системы охлаждения и прежде всего от инерционности системы. Самый инерционный вариант это радиатор холодной стороны Пельтье в камере, который охлаждает воздух. Самый быстрый - датчик температуры, расположенный непосредственно на холодной поверхности элемента Пельтье. В первом случае время реакции измеряется десятками минут, во втором - секундами. Поэтому регулятор настраивается на реальный объект. Возьмем коэффициенты эмпирически для инерционного варианта холодильника. »[3]

Пропорциональный коэффициент.

«Выбрали значение koeffRegTmpPr = 0,5. Это значит, что при ошибке рассогласования 1 С° пропорциональное звено даст 0,5 Вт на элементе Пельтье. Допустим, включили нагретый холодильник. Разница между заданной и реальной температурами 10 С°. В этом случае пропорциональная составляющая мгновенно задаст регулятору мощности 5 Вт. »[3]

Интегральный коэффициент.

«Выбрали koeffRegTmpInt = 0,002. Это значит, что при ошибке рассогласования 1 С° каждые 1 сек выходная мощность будет увеличиваться на 0,002 Вт. За 8 минут она увеличится на 1 Вт. При ошибке рассогласования 10 С° мощность будет увеличиваться в 10 раз быстрее, т.е. на 1 Вт за 50 сек. »[3]

Дифференцирующий коэффициент.

Задали koeffRegTmpDif = 50. Это значит, что при изменении ошибки рассогласования на 0,1 С° мощность изменится на 5 Вт.

« С этим коэффициентом не все так просто. Включили дифференцирующую составляющую только в демонстрационных целях. Она имеет смысл только в системах с малой инерционность. Температура увеличилась на 0,1 С°. Дифференцирующая составляющая выдала уменьшение заданной мощности на 5 вт. Но уменьшение мощности будет

сформировано только в течение одной единицы временной дискретности регулятора, 1 сек.. В принципе его быстродействие позволяет отработать за такое время, но значительно снизили скорость его работы из соображений щадящего режима для модуля Пельтье. »[3]

«Значительно увеличили дифференцирующий коэффициент. Тогда при медленном регуляторе мощности произойдет следующее. Допустим, при изменении ошибки рассогласования дифференцирующее звено даст такую составляющую, которая уменьшит заданную мощность до 0. Это значение будет держаться в течение 1 сек. Регулятор мощности у нас работает в цикле 20 мс, т.е. 50 проходов за 1 сек. В этом случае выброс дифференцирующей составляющей в какой-то мере запомнится в интегральном звене. Но как это будет работать, зависит от нескольких коэффициентов. »[3]

«Есть еще проблема с дифференцирующим звеном. Датчик температуры дискретный, с разрешением 0,0625 С°. Как и любой узел преобразования аналоговой величины в цифровую он имеет особенность, связанную со значением выходного кода на границе ступеней преобразования. Код на границах «скачет» на плюс минус единицу преобразования. В нашем случае значение температуры будет «дергаться» на 0,0625 С°. Это будет воспринято дифференцирующим звеном и не лучшим образом повлияет на работу регулятора в целом. »[3]

«Если кому-нибудь действительно необходим полный ПИД регулятор температуры, то он должен изменить временную дискретность дифференцирующего регулятора и обработать код измеренной температуры, чтобы исключить «дергание» на единицу дискретности. Сделать цифровую фильтрацию и ввести гистерезис. »[3]

Вывод

В данной главе проведен анализ электрической системы для управления работой холодильника. Проведено обоснование и выбор элементов, необходимых для построения замкнутой системы автоматического регулирования температуры. Получена экспериментальная установка и проведена предварительная настройка системы авторегулирования. Разработана программа с пользовательским интерфейсом и функциями обеспечивающими тепловую защиту элемента Пельтье.

3 Экспериментальная часть

3.1 Принципиальная схема контроллера

Проверяемая схема содержит блок питания, плата Ардуино, импульсный регулятор, датчики температуры и нагрузку 20 Ом.

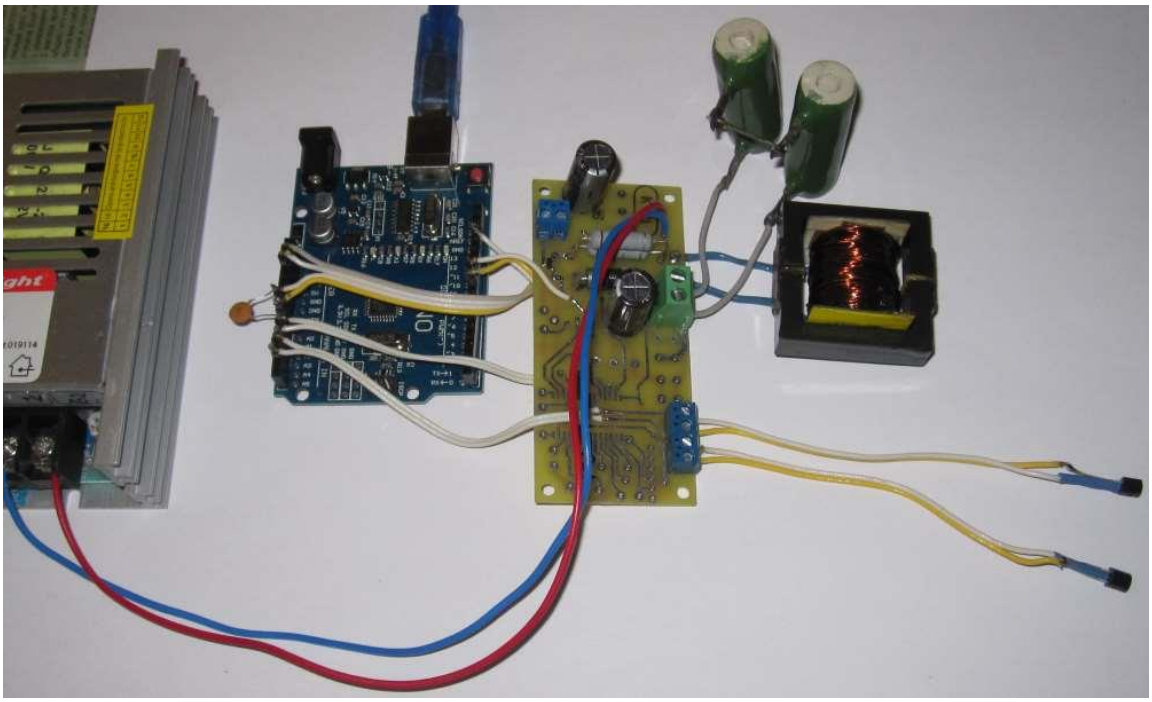

Рисунок 27 – Схема с датчиками температуры

Схема с датчиками температуры изображена на рисунке 27.

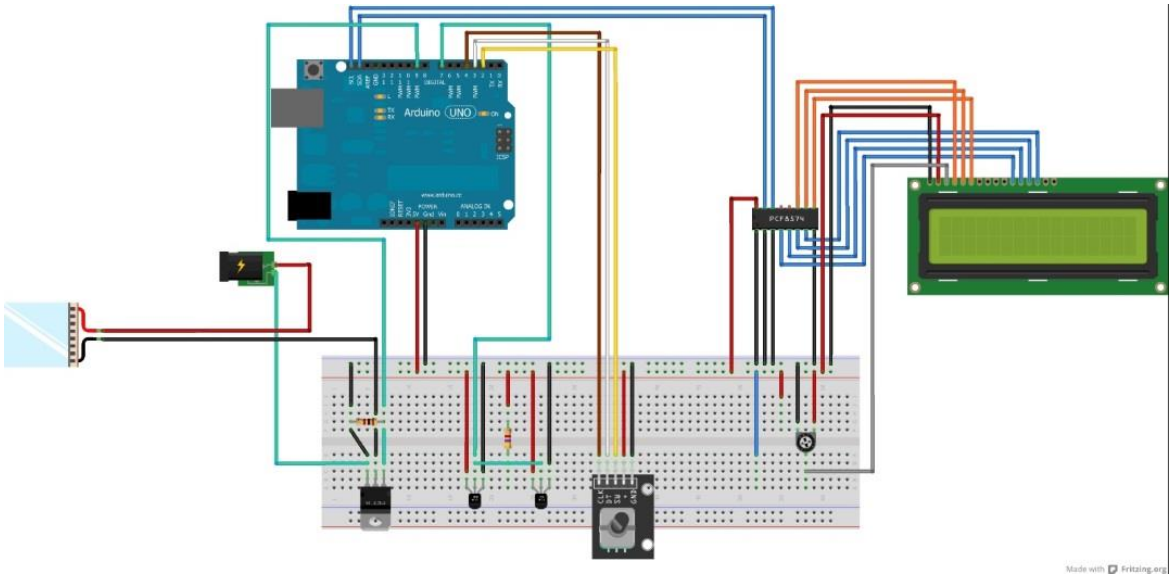

Рисунок 28 - Схема электрическая общая

С помощью системы эмуляции (рисунок28) Tinkercad Arduino собрана и запрограммирована схема системы.

Проработано решение радиаторов камеры (рисунок29,30). Внутренний радиатор через элемент Пельтье и с термоизолирующей прокладкой (вспененный полипропилен) устанавливается в окно коробки камеры. С другой стороны элемент Пельтье крепится на внешний радиатор с вентилятором. Внешний вид элементов камеры и сама камера показаны на рисунках 29-31.

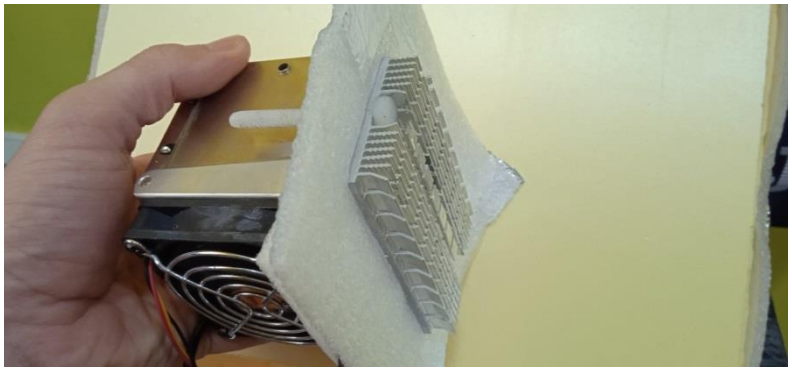

Рисунок 29- Внешний вид

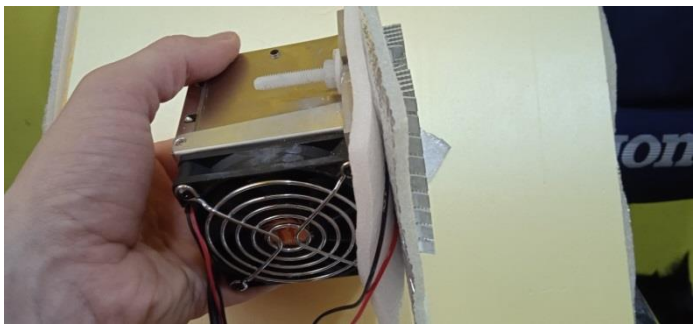

Рисунок 30- Внешний вид с боку

Рисунок 31- Экспериментальная модель камеры

Приведем сравнительные графики, полученные в эксперименте при разных тестированиях камеры (рисунок 32-36).

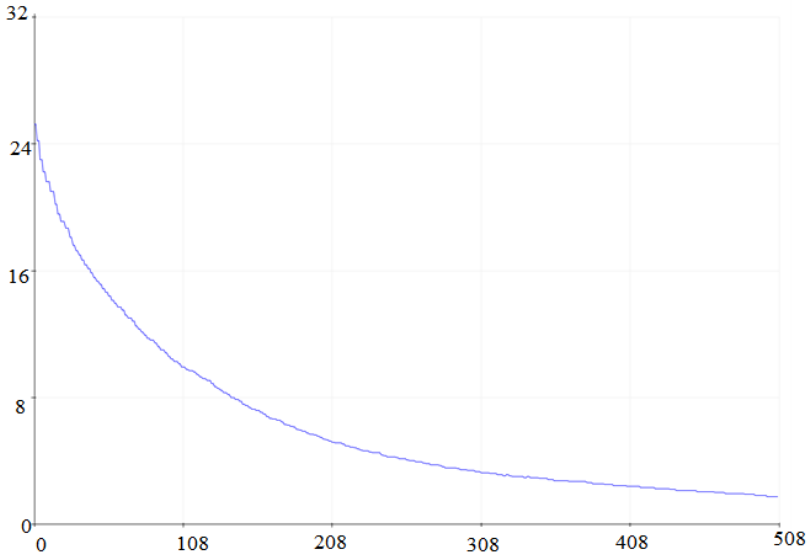

Рисунок 32 - Минимально достигнутая температура в камере t= +2^{0}C

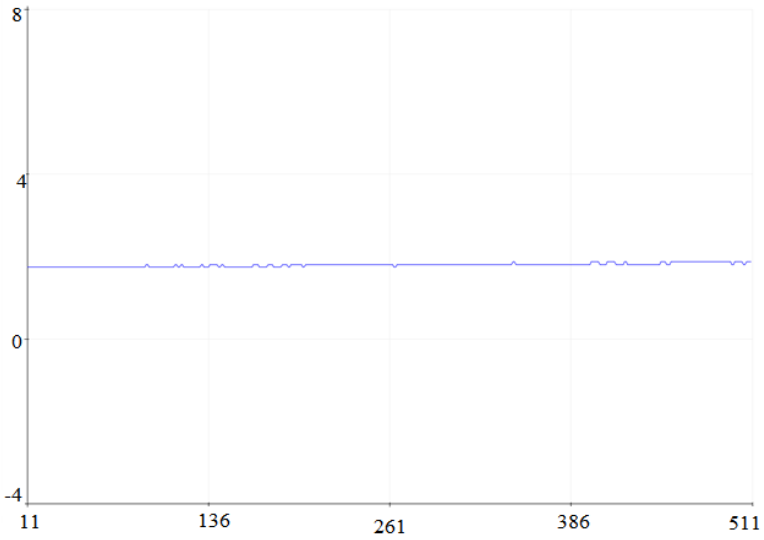

Рисунок 33 - Поддержания заданной температуры. t=5^{0}C

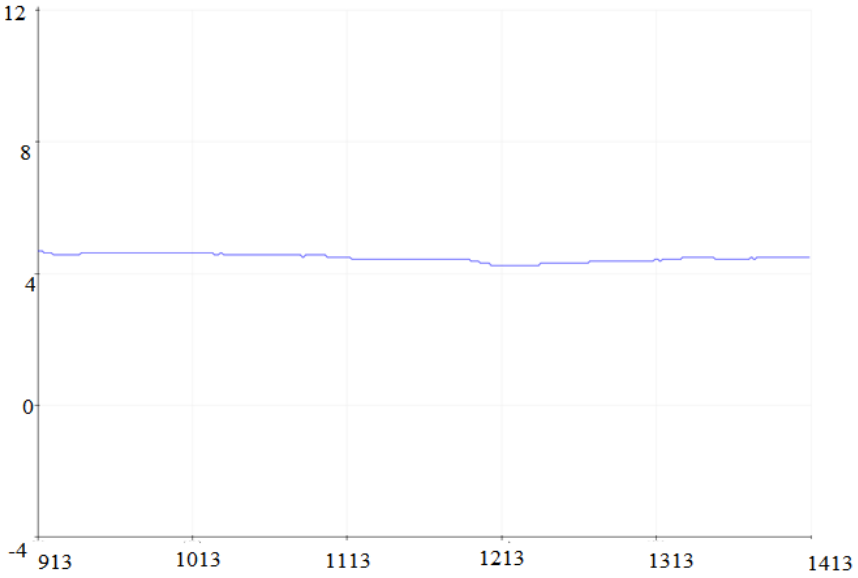

Рисунок 34 - Изменения температуры в камере за 2 часа.

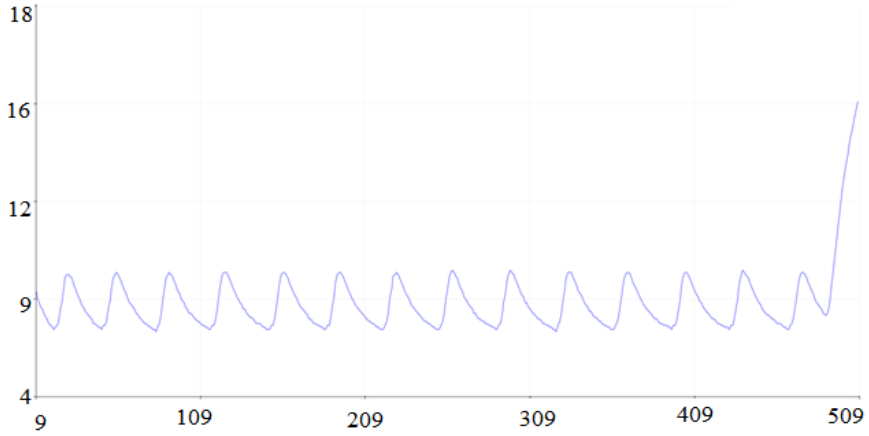

Рисунок 35- Изменения температуры в камере при установленной температуре t=9^{0}C.

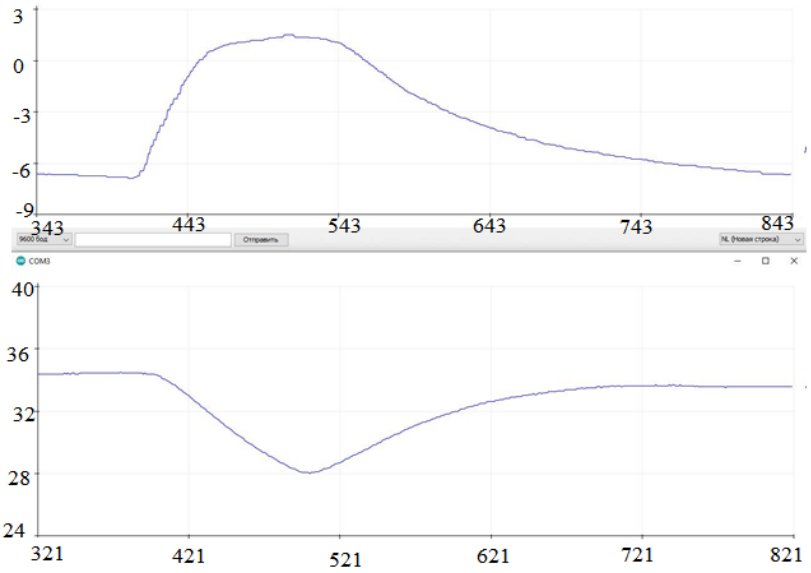

Рисунок 36- Изменения температуры в камере при установленной температуре t=9^{0}C.

Установили в программе заданную температуру 20 C°.

setTempRef= 20.; //временно заданная температура

Загрузили скетч с выбранными коэффициентами в плату. Запустили монитор последовательного порта, блок питания 12 В пока не подключали (рисунок 37).

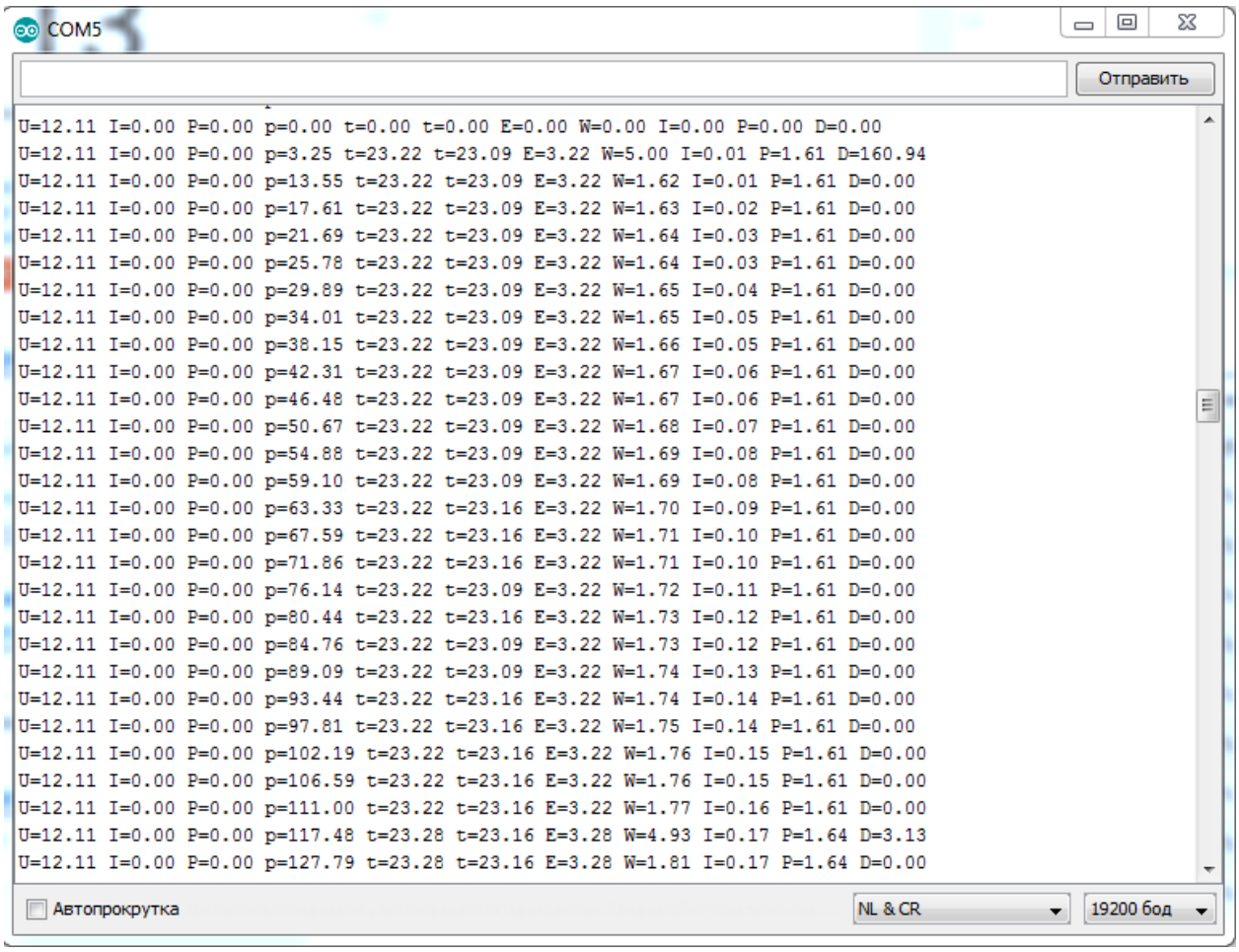

Рисунок 37 – Результаты на заданную температуру 20 С°.

« Ошибка рассогласования E = 3.22 С°. Пропорциональная составляющая должна быть 3,22 * koeffRegTmpPr = 3,22 * 0,5 = 1,61 Вт. Так и есть P=1.61. »[3]

« Интегральная составляющая должна увеличиваться каждую секунду на 3,22 * koeffRegTmpInt = 3,22 *0,002 = 0,0064 Вт. То же все правильно. За 20 секунд (20 строчек) I увеличилось с 0 до 0,13. »[3]

« В последних строчках видно, что ошибка рассогласования увеличилась на 0,08 С°. Дифференциальная составляющая должна быть 0,08 * koeffRegTmpDif = 0,06 * 50 = 3 Вт. Регулятор отработал правильно D=3,13. Небольшая неточность, объясняется тем, что мы выводим только 2 знака после запятой. Установили заданную температуру 25 С° и увидели, что

ошибка рассогласования стала отрицательной E=-1.53 С°, регулятор отключился, пропорциональное звено дало правильное значение -1,53 * 0,5 = -0,765 Вт (рисунок 38). »[3]

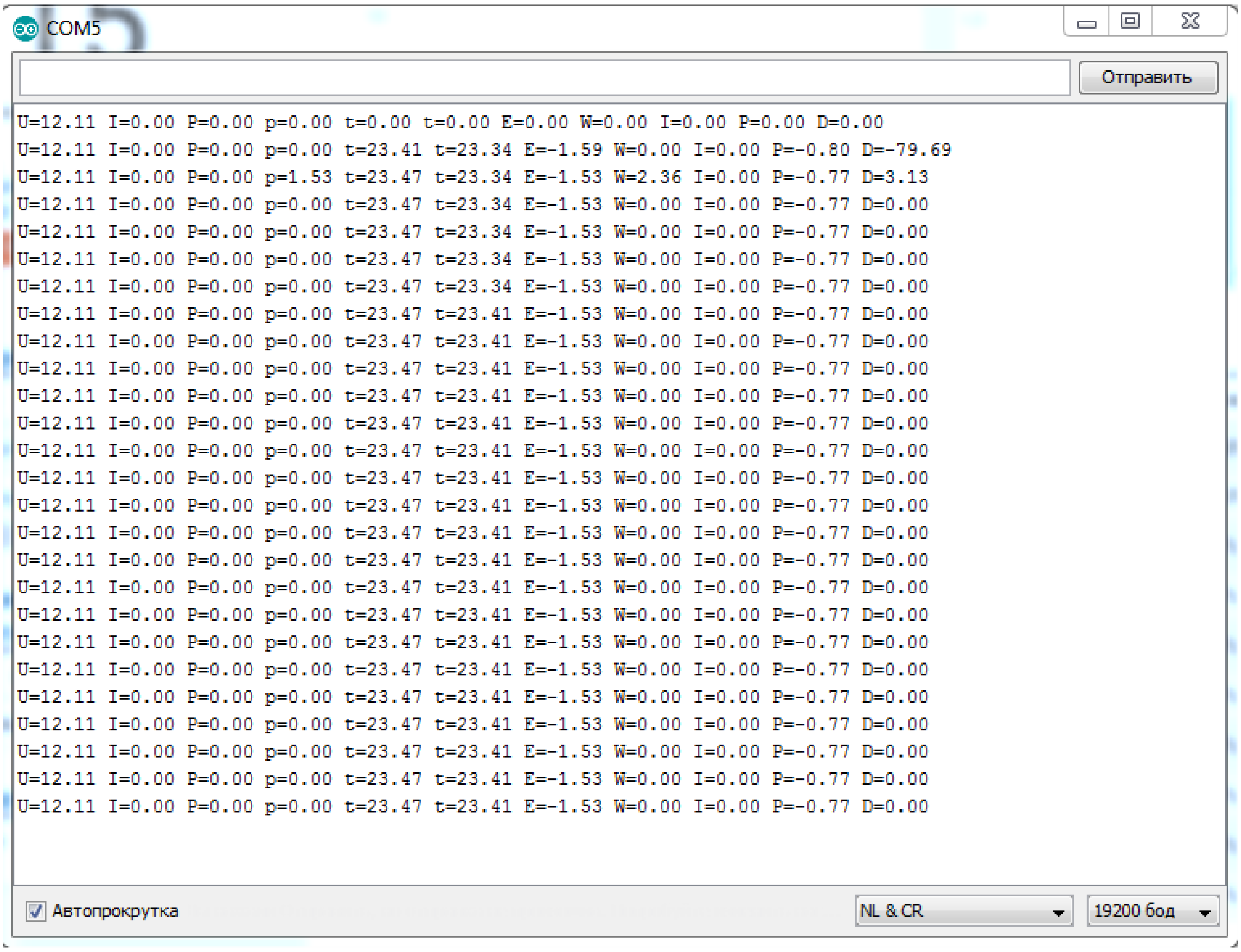

Рисунок 38 - Результаты на заданную температуру 25 С°.

«Вернул заданную температуру 20 С°. Подключил блок питания 12 В. и увидели, как меняется реальная мощность на нагрузке (рисунок 39). »[3]

```
U=0.59 I=0.00 P=0.00 p=13.76 t=23.84 t=23.66 E=3.84 W=1.94 I=0.02 P=1.92 D=0.00
U=0.78 I=0.00 P=0.00 p=18.61 t=23.84 t=23.66 E=3.84 W=1.94 I=0.02 P=1.92 D=0.00
U=0.98 I=0.00 P=0.00 p=23.47 t=23.84 t=23.66 E=3.84 W=1.95 I=0.03 P=1.92 D=0.00
U=1.18 I=0.00 P=0.00 p=28.36 t=23.84 t=23.66 E=3.84 W=1.96 I=0.04 P=1.92 D=0.00
U=1.41 I=0.00 P=0.00 p=33.27 t=23.84 t=23.72 E=3.84 W=1.97 I=0.05 P=1.92 D=0.00

U=6.89 I=0.11 P=1.30 p=148.29 t=23.84 t=23.72 E=3.84 W=2.18 I=0.25 P=1.92 D=0.00
U=6.97 I=0.13 P=1.56 p=150.13 t=23.84 t=23.66 E=3.84 W=2.18 I=0.26 P=1.92 D=0.00
U=7.02 I=0.13 P=1.56 p=151.74 t=23.84 t=23.72 E=3.84 W=2.19 I=0.27 P=1.92 D=0.00
U=7.12 I=0.13 P=1.56 p=153.31 t=23.84 t=23.72 E=3.84 W=2.20 I=0.28 P=1.92 D=0.00
U=7.17 I=0.13 P=1.56 p=154.74 t=23.84 t=23.72 E=3.84 W=2.21 I=0.28 P=1.92 D=0.00

U=9.76 I=0.32 P=3.90 p=207.80 t=23.72 t=23.59 E=3.72 W=4.33 I=2.47 P=1.86 D=0.00
U=9.79 I=0.34 P=4.16 p=208.32 t=23.72 t=23.53 E=3.72 W=4.33 I=2.47 P=1.86 D=0.00
U=9.79 I=0.32 P=3.90 p=208.82 t=23.72 t=23.53 E=3.72 W=4.34 I=2.48 P=1.86 D=0.00
U=9.84 I=0.34 P=4.16 p=209.26 t=23.72 t=23.59 E=3.72 W=4.35 I=2.49 P=1.86 D=0.00
U=9.84 I=0.34 P=4.16 p=209.69 t=23.72 t=23.53 E=3.72 W=4.36 I=2.50 P=1.86 D=0.00

U=10.38 I=0.41 P=4.94 p=220.83 t=23.59 t=23.41 E=3.59 W=5.00 I=5.00 P=1.80 D=0.00
U=10.41 I=0.41 P=4.94 p=221.03 t=23.59 t=23.41 E=3.59 W=5.00 I=5.00 P=1.80 D=0.00
U=10.43 I=0.41 P=4.94 p=221.19 t=23.59 t=23.41 E=3.59 W=5.00 I=5.00 P=1.80 D=0.00
U=10.43 I=0.41 P=4.94 p=221.33 t=23.59 t=23.41 E=3.59 W=5.00 I=5.00 P=1.80 D=0.00
U=10.41 I=0.41 P=4.94 p=221.48 t=23.59 t=23.47 E=3.59 W=5.00 I=5.00 P=1.80 D=0.00
```

Рисунок 39 – Результат мощность на нагрузке

«Мощность растет и останавливается на заданном в программе пределе 5 Вт. Все правильно. Скачек температуры на 0,07 С° дифференциальное звено отработало правильно, но на выходной мощности это не отразилось (рисунок 40). »[3]

```
U=10.46 I=0.41 P=4.94 p=223.00 t=23.66 t=23.47 E=3.66 W=5.00 I=4.90 P=1.83 D=0.00
U=10.46 I=0.41 P=4.94 p=223.00 t=23.66 t=23.47 E=3.66 W=5.00 I=4.91 P=1.83 D=0.00
U=10.46 I=0.41 P=4.94 p=222.09 t=23.59 t=23.41 E=3.59 W=3.59 I=4.91 P=1.80 D=-3.13
U=10.33 I=0.41 P=4.94 p=219.76 t=23.59 t=23.47 E=3.59 W=5.00 I=4.92 P=1.80 D=0.00
U=10.38 I=0.41 P=4.94 p=220.09 t=23.59 t=23.41 E=3.59 W=5.00 I=4.93 P=1.80 D=0.00
```

Рисунок 40- Результат скачка температуры на 0,07 С°

«Те самые «скачки» температуры на границе ступеней преобразования датчика (рисунок 41). »[3]

```
U=8.28 I=0.21 P=2.60 p=177.55 t=23.78 t=23.66 E=3.78 W=2.91 I=1.02 P=1.89 D=0.00
U=8.38 I=0.21 P=2.60 p=179.69 t=23.84 t=23.59 E=3.84 W=5.00 I=1.03 P=1.92 D=3.13
U=8.58 I=0.24 P=2.86 p=183.74 t=23.84 t=23.66 E=3.84 W=2.96 I=1.04 P=1.92 D=0.00
U=8.53 I=0.24 P=2.86 p=182.05 t=23.78 t=23.66 E=3.78 W=0.00 I=1.04 P=1.89 D=-3.13
U=8.28 I=0.21 P=2.60 p=177.30 t=23.78 t=23.66 E=3.78 W=2.94 I=1.05 P=1.89 D=0.00
U=8.33 I=0.21 P=2.60 p=178.17 t=23.78 t=23.66 E=3.78 W=2.95 I=1.06 P=1.89 D=0.00
U=8.43 I=0.21 P=2.60 p=180.35 t=23.84 t=23.66 E=3.84 W=5.00 I=1.07 P=1.92 D=3.13
U=8.53 I=0.24 P=2.86 p=182.46 t=23.78 t=23.66 E=3.78 W=0.00 I=1.07 P=1.89 D=-3.13
U=8.30 I=0.21 P=2.60 p=177.63 t=23.78 t=23.66 E=3.78 W=2.97 I=1.08 P=1.89 D=0.00
```

Рисунок 41 -«Скачки» температуры на границе ступеней преобразования датчика

«Видно, как они отражаются на интеграторе регулятора мощности p, т.е. на ШИМ. Полную проверку и настройку регулятора температуры будем производить на реальном объекте с программой верхнего уровня, регистрирующей состояние системы. »[3]

«Далее продолжим эксперимен выводить температуру на горячий радиатор (рисунок 42-44) »[3]

Рисунок 42 - Шаг установки 0,5 гр.

«В верхней строке, с лева выводится реальная температура горячего радиатора. Справа в верхней строке реальная температура в камере. На нижней строке выводится заданная температура (шаг установки 0,5 гр). »[3]

Рисунок 43 - Температура радиатора 30,1 C^0

Рисунок 44 - Температура радиатора 50 C^0

«Если в процессе эксплуатации выйдет из строя элетродвигатель вентилятора горячего радиатора, то при превышении температуры горячего радиатора 50 гр. на нижней строке выводится «Error» и питание элемента Пельтье прекращается. Для повторного запуска необходимо выкл/вкл питание от сети. »[3]

3.2 Интерфейс с пользователем

Теперь необходимо определить пользовательский интерфейс, т.е.:

- что и в каком виде отображать на дисплее;
- какие параметры задавать;
- назначение кнопок, как ими устанавливать параметры;
- как отображать аварийные ситуации.

На индикаторах отображаются следующие параметры:

- измеренная температура в камере;
- заданная температура в камере;

- электрическая мощность на модуле Пельтье;
- заданная максимальная мощность;
- измеренная температура радиатора горячей стороны модуля;
- предельная температура радиатора.

Пользователь может задавать:

- температуру в камере;
- максимальную мощность на охлаждающем элементе.

Управление происходит с помощью трех кнопок, которые называются: «+», «-» и «ВЫБОР».

3.3 Управление контроллером

«После включения устройства по экрану дисплея пробегает звездочка (*) (рисунок 45). Это примитивная проверка индикатора. К тому же неплохо при проверке устройства видеть, когда происходит сброс контроллера. »[3]

Рисунок 45 – Дисплей со (*)

«После этого на дисплее отображается текущее значение температуры в камере (рисунок 46). »[3]

Рисунок 46- Дисплей со значением температуры

«При нажатой кнопке «ВЫБОР» дисплей отображает значение заданной температуры в камере (рисунок 47). »[3]

Рисунок 47 - Значение заданной температуры в камере

«Установить заданную температуру можно с помощью кнопок «+» и «-» при нажатой кнопке «ВЫБОР». Каждое нажатие кнопки «+» или «-» изменяет параметр на 0,1 °С. Если удерживать одну из кнопок «+» или «-» нажатой, то через 0,8 сек параметр будет автоматически изменяться на 0,1 °С со скоростью 5 раз в секунду. »[3]

«Запись параметра в энергонезависимую память (EEPROM) происходит при отпускании кнопки «ВЫБОР». Запись происходит только в случае, если параметр был изменен. »[3]

«Если нажать кнопку «+» при отжатой кнопке «ВЫБОР», то устройство перейдет в режим отображения электрической мощности на модуле Пельтье (рисунок 48). »[3]

Рисунок 48 - Электрическая мощность на модуле Пельтье

«Нажав кнопку «ВЫБОР» можно посмотреть заданную максимальную мощность на охлаждающем модуле (рисунок 3.8). »[3]

Рисунок 49 - Максимальная мощность на охлаждающем модуле

«Кнопками «+» и «-» (при нажатой кнопке «ВЫБОР») можно установить максимальную мощность устройства. Каждое нажатие на кнопку «+» или «-» изменяет значение мощности на 1 Вт. »[3]

«Следующее нажатие кнопки «+» (с отжатой кнопкой «ВЫБОР») переведет устройство в режим индикации температуры радиатора (рисунок 50). »[3]

Рисунок 50 - Температура радиатора

«Если в этом режиме нажать кнопку «ВЫБОР», то на дисплее отобразится максимально допустимое значение температуры радиатора (рисунок 51). »[3]

Рисунок 51 - Максимально допустимое значение температуры радиатора

«При нагреве радиатора свыше заданной температуры устройство отключится, чтобы защитить элемент Пельтье от перегрева. Изменить максимально допустимое значение температуры радиатора кнопками нельзя. »[3]

«Резюмируя можно сказать, что при отжатой кнопке «ВЫБОР» кнопки «+» и «-» меняют режим отображения информации в последовательности: »[3]

- > Температура в камере - > выходная мощность - > температура радиатора ->

< - Температура в камере < - выходная мощность < - температура радиатора < -

«А нажатая кнопка «ВЫБОР» переводит устройство в режим установки параметра. Контроллер обрабатывает следующие аварийные ситуации: »[3]

«1) Ошибки чтения обоих термодатчиков. При неисправности датчика вместо значения температуры на дисплее отображается «---». Датчик температуры считается неисправным, если недостоверные данные приходят 3 раза в подряд. Т.е. если в течение 3 секунд считывается хотя бы одно правильное значение температуры, то система работает в обычном режиме. »[3]

«2) Ошибка заданных параметров в EEPROM. Параметры сохраняются в энергонезависимой памяти Ардуино и сопровождаются контрольным кодом. При нарушении целостности данных возникает ошибка. Реально такая ошибка может появиться при первом включении контроллера после загрузки программы в плату Ардуино. Исправляется ошибка повторным заданием параметров. »[3]

« 3) Ошибка перегрева радиатора. Появляется в случае, если радиатор горячей стороны модуля Пельтье нагрелся выше заданной температуры. Такая ошибка может появиться при неисправном вентиляторе. »[3]

«4) При появлении любой ошибки контроллер отключает охлаждающий модуль и индицирует об ошибке миганием светодиода «ОШИБКА». »[3]

3.4 Резидентное программное обеспечение контроллера

Для программного обеспечения контроллера используем:

-блок индикации и установки параметров (цикл 20 мс);

-блок загрузки параметров из EEPROM (интервал 1);

-блок контроля температуры радиатора (интервал 47);

-функции записи и чтения данных из EEPROM с контролем целостности данных;

-добавили нужные параметры в блок передачи данных на компьютер (интервалы 11-13).

Для управления LCD индикатором использовали библиотеку LiquidCrystal.h. Перед применением библиотеки измерили время выполнения функций.

Получил следующие результаты (таблица 3).

Таблица 3- Результаты управления LCD индикатором

Функция	Время выполнения	Назначение
disp.begin(8, 1);	1406 мкс	Инициализация дисплея
disp.print("1");	287 мкс	Вывод текстовой строки (1 символ)
disp.print("12345678");	2264 мкс	Вывод текстовой строки (8 символов)
disp.clear();	2280 мкс	Очистка экрана
disp.home();	2280 мкс	Курсор в начало экрана
disp.write(ccc);	281 мкс	Вывод байта (char) на экран
disp.print(ddd);	1610 мкс	Вывод числа int на экран

Таблица 4- Параметры в EEPROM

Адрес	Содержимое	Параметр
5…8	число float	Заданная температура в камере
9	сумма байтов 5…8 ^ 0xe5	
10…13	число float	Максимальная выходная мощность
14	сумма байтов 10…13 ^ 0xe5	

Числа с форматом типа float обязательно указываются с децимальной точкой. Например, 10.5, 1.23, 1., 1.0.

На компьютер контроллер передает данные в текстовом виде, со скоростью 19200 бод, в следующей последовательности (таблица 5).

Таблица 5-Данные контроллера

Обозначение	Формат	Ед. измерения	Параметр
U=xx.xx	float	В	Напряжение на выходе
I=xx.xx	float	А	Ток потребления регулятора
P=xx.xx	float	Вт	Выходная мощность
p=xx.xx	float	ед. ШИМ	Интегральное звено регулятора мощности
t=xx.xx	float	°C	температура в камере
t=xx.xx	float	°C	Температура радиатора
E=xx.xx	float	°C	Ошибка рассогласования
W=xx.xx	float	Вт	Заданная мощность для регулятора мощности
I=xx.xx	float	Вт	Интегральная часть
P=xx.xx	float	Вт	Пропорциональная часть
D=xx.xx	float	Вт	Дифференциальная часть
K=xx.xx	float	-	интегральный коэффициент регулятора мощности
i=xx.xx	float	-	Интегральный коэффициент регулятора температуры
p=xx.xx	float	-	Пропорциональный коэффициент регулятора температуры
d=xx.xx	float	-	Дифференцирующий коэффициент регулятора температуры
V=xx.x	float	°C	Температура включения вентилятора
v=xx.x	float	°C	Температура выключения вентилятора
H=xx.x	float	°C	Температура перегрева радиатора
P=xx.x	float	Вт	Максимальная заданная мощность
T=xx.xx	float	°C	Заданная температура в камере
E=xx	DEC	-	Признаки ошибок: 0 - ошибка EEPROM; 1 - вентилятор включен; 2 - перегрев; 3 -фатальная ошибка
V=x.x	string	-	Версия программы

Включили, проверили. Контролировали работу устройства по данным монитора последовательного порта (рисунок 52).

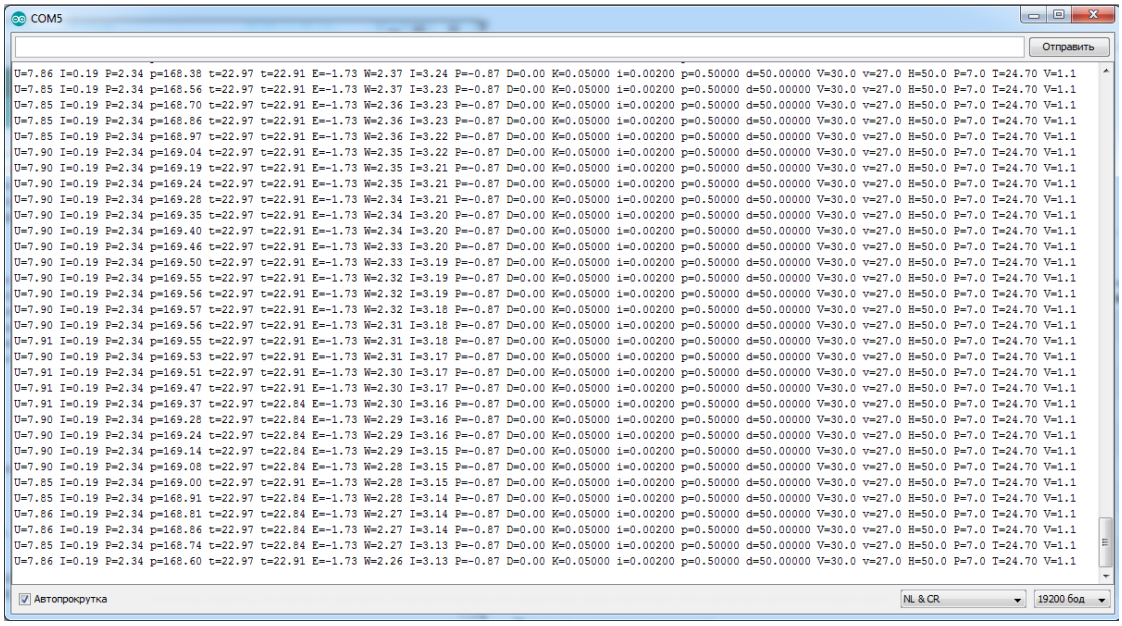

Рисунок 52- Даные работы устройства

Программа для Ардуино к данной работе в Приложение А.

Вывод

Получены экспериментальные характерситики. Проверены на реальном объекте релейный метод регулирования и метод ПИД-регулирования, установлено что методом релейного регулирования снижает характеристики холодильника из-за термоциклирования холодной и горячей стороны элемента Пельтье, а так же дает большую пульсацию температуры. Поэтому рабочим был выбран релейный алгоритм. Измерения проводились с помощью датчиков температуры внутри камеры.

Заключение

В ходе выполнения дипломной выпускной квалификационной работы была разработана система холодильника на элементе пельтье TEC2-19006 поскольку элементы пельтье обладают совокупностью положительных свойств заключается низкой стоимости простоте управления и компактности. Для управления температурой была выбрана система с использованием ПИД-регулятора, который спроектирован на базе Arduino-подобной системы, в которой Arduino выполняет как функции самого регулятора, который можно настраивать так и устройства обеспечивающего интерфейс с LCD экраном.

В процессе изготовления и экспериментальной апробации холодильника был проверен релейный- и ПИД-регуляторы поддерживающие температуру внутри холодильника. Экспериментально было доказано, что релейный регулятор управления температурой приводит к большим колебаниям температуры на горячей и холодной стороне элемента Пельтье, что может влиять на его ресурс работы.

В системе предусмотрена регулировка ПИД-регулятор: имеется возможность задавать параметры работы холодильника, температуру внутри холодильника с точностью $0{,}5^0$C, а также можно производить динамическую настройку задания температуры с помощью параметров ПИД-регулятора.

Для управления мощностью в системе предусмотрен силовой ключ, а температура отслеживается с помощью цифровых датчиков температуры.

В работе проведены экспериментальные исследования, показавшие возможность работы системы с максимальной разницей температур между окружающей средой и камерой холодильника порядка 20 ^{0}C.

Улучшение характеристик холодильника выполнено с помощью установки внутри холодильника дополнительного вентилятора.

Список используемой литературы

1. Анатычук Л.И., Семенюк В.А. Оптимальное управление свойствами термоэлектрических материалов и приборов. Черновцы, Прут, 1992. 135 с.
2. Анатычук Л.И. Термоэлементы и термоэлектрические устройства. Справочник. Киев: Наукова Думка, 1979. 385 с.
3. Бернштейн А.С. Термоэлектрические генераторы. Л.: Гос.энергиздат, 1970. 48с.
4. Булат Л.П. Термоэлектрическое охлаждение: Текст лекций СПб.: СПбГУНиПТ, 2002. 147 с.
5. ГОСТ 7.1-2003 Библиографическая запись. Библиографическое описание. Общие требования и правила составления. Введ. 2004-07-01. М.: Изд-во стандартов, 2004. 170 с.
6. Забродин Ю.С. Промышленная электроника : Учебник для вузов М.: Высш. школа, 1982. 496 с.
7. Ильярский О. И., Удалов Н. П. Термоэлектрические элементы М. : Энергия, 1970. 72 с
8. Коленко Е.А. Термоэлектрические охлаждающие приборы : 2-е изд., перераб. и доп. Л.: Наука (Ленинградское отделение), 1967. 282 с.
9. Косенко Е. А. Термоэлектрические охлаждающие приборы: 2-е изд., перераб. И доп. Л. : Наука (Ленинградское отделение), 1967. 282 с.
10. Методические указания по оформлению выпускных квалификационных работ по программам бакалавриата, программам специалитета, программам магистратуры: [Электронный ресурс]. URL: https://yadi.sk/d/Fs-9ts_VInrE3Q/BKP_(Diplom) / Оформление ВКР (Дата обращения: 18.02.2021)

11. Охотин А.С., Ефремов А.А., Охотин В.С., Пушкарский А.С. Термоэлектрические генераторы М., Атомиздат, 1971. 292 с. Под ред.доктора физико-математических наук А. Р. Регеля.

12. Поздняков Б.С., Коптелов Е.А. Термоэлектрическая энергетика М.: Атомиздат, 1974. 264 с.

13. Порядок обеспечения самостоятельности выполнения письменных работ в ТГУ: [Электронный ресурс]. URL: https://yadi.sk/d/Fs-9ts_VInrE3Q/BKP_(Diplom) / Положение о Антиплагиате (Дата обращения: 18.02.2021)

14. Положение о выпускной квалификационной работе: утв. решен.учен. совет. от 21.11.2019 решение №254 : [Электронный ресурс]. URL: https://yadi.sk/d/Fs-9ts_VInrE3Q/BKP_(Diplom) / Положение о ВКР (Дата обращения: 18.02.2021)

15. Позднов М.В., Прядилов А.В. Электроника и наноэлектроника, управление в технических системах, электроэнергетика и электротехника. Выполнение бакалаврской работы . Тольятти: ТГУ, 2019. 41 с.

16. AT89S8252 Datasheet (PDF) - ATMEL Corporation [Электронныйресурс]. URL: http://www.alldatasheet.com/datasheet-pdf/pdf/175000/ATMEL/AT89S8252.html (дата обращения: 27.05.2021)

17. AT89S8252 Primer [Электронный ресурс]. URL: http://www.shrubbery.net/~heas/willem/PDF/ATMEL%20Flash%20Microcontroller/8051-Architecture/Application%20Notes/AT89S8252%20Primer.pdf (дата обращения: 27.05.2021)

18. ElectronicComponentsDatasheets [Электронный ресурс] URL: http://www.datasheets.ru (дата обращения: 17.02.2021)

19. Electrical resistance and conductance [Электронныйресурс]. URL: https://en.wikipedia.org/wiki/Electrical_resistance_and_conductance (дата обращения: 17.05.2021)

20. ExtremeTech [электронный ресурс]. -, США. ExtremeTech, 1996 - 2016. — Режим доступа: http://www.extremetech.com/extreme/169951

21. Goupil C. (Ed.) Continuum Theory of Thermoelectric Elements — Wiley-VCH Verlag GmbH & Co., Weinheim, Germany, 2016, — 363 p. — ISBN: 3527413375

Приложение А

Программный код

Программа содержит ряд библиотек, необходимых для работы отдельных частей системы для:

- работы с датчиками

#include <Wire.h> //подключаем библиотеку I2C

#include <OneWire.h> // подключаем библиотеку для создания шины OneWire и приема данных с даттчиков температуры

-работы с ПИД-регулятором

#include <PID_v1.h> //подключаем библиотеку ПИД регулятора

-работы с LCD-индикатором

#include <LiquidCrystal_I2C.h> // подключаем библиотеку для вывода информации на LCD дисплей

-работы с флэш памятью для хранения начальных установок

#include <EEPROM.h> // подключаем библиотеку EEPROM для сохранения параметров

В начале программы описываются переменны необходимы для работы, настройки ПИД регулятора

PID _PID_1(&_PID_1_In, &_PID_1_Out, &_PID_1_SP, 400, 100, 100, REVERSE); //указываем параметры для ПИД регулятора

Номер вывода подключения датчиков

OneWire _ow7(7); //указываем пин 7 для подключения датчиков температуры

Назначаются выводы для управления подключения энеодера

```
pinMode(2, INPUT_PULLUP);    //назначаем пины 2,3,4 как вход
pinMode(3, INPUT);
pinMode(4, INPUT);
```

```
void loop()
{
   if (_isNeedClearDisp1)
   {
      _lcd1.clear();
      _isNeedClearDisp1= 0;
   }
   bool  _bounceInputTmpD2 =  (digitalRead (2));
   if (_bounseInputD2S)
   {
      if (millis() >= (_bounseInputD2P + 40))
      {
         _bounseInputD2O= _bounceInputTmpD2;
         _bounseInputD2S=0;
      }
   }
   else
   {
      if (_bounceInputTmpD2 != _bounseInputD2O)
      {
         _bounseInputD2S=1;
         _bounseInputD2P = millis();
      }
   }

   if(_isTimer(_d18x2x2Tti, 1000)) //устанавливаем временной интервал опроса датчика температуры
```

```
{
    _d18x2x2Tti = millis();
    if (! _ow7W)
    {
       _tempVariable_float =  _readDS18_ow7(_d18x2x2Addr, 0);
       if (_tempVariable_float < 500)
       {
          _d18x2x2O = _tempVariable_float;
       }
    }
}
bool  _tmp1 = ((int((_d18x2x2O)))) >= (50); //сравниваем температуру горячего радиатора с устанавленной температурой аварийного отключения
if (_tmp1)
{
    if (! _trgt1I) _trgt1 = ! _trgt1;
}
_trgt1I = _tmp1;
if(_isTimer(_d18x2x1Tti, 1000))
{
   _d18x2x1Tti = millis();
   if (!_ow7W)
   {
      _ow7W=1;
      _d18x2x1W= 1;
      _ow7P= millis();
   }
    else
```

```
        {
            if(_d18x2x1W)
            {
               if(abs(millis()-_ow7P)>1000)
               {
                  _tempVariable_float =  _readDS18_ow7(_d18x2x1Addr, 0);
                  if (_tempVariable_float < 500)
                  {
                     _d18x2x1O = _tempVariable_float;
                  }
                  _d18x2x1W=0;
                  _ow7W=0;
               }
            }
         }
         if (1)
         {
            _dispTempLength1                                                    =
(((_floatToStringWitRaz((_d18x2x1O),1)))).length();
            if (_disp4oldLength > _dispTempLength1)
            {
               _isNeedClearDisp1 = 1;
            }
            _disp4oldLength = _dispTempLength1;
            _lcd1.setCursor(12, 0);                              //указываем место
отображения на экране
```

```
        _lcd1.print(((_floatToStringWitRaz((_d18x2x1O),1))));        //выводим
показания датчика температуры установленного в холодильной камере
      }
       else
      {
        if (_disp4oldLength > 0)
        {
          _isNeedClearDisp1 = 1;
          _disp4oldLength = 0;
        }
      }
      _tempVariable_bool = !(_trgt1);
      if(!((_PID_1.GetMode()) == _tempVariable_bool))          //работа ПИД
регулятора
      {
        _PID_1.SetMode(_tempVariable_bool);
      }
      if(! _tempVariable_bool)
      {
        _PID_1_Out = 0;
      }
      _PID_1_In = (_d18x2x1O);
      _PID_1_SP = (_menuValueArray_float[0]);
      _PID_1.Compute();
      analogWrite(9, _PID_1_Out);
      if (_trgt1)
      {
```

```
  _dispTempLength1 = ((_gtv2)).length();
    if (_disp3oldLength > _dispTempLength1)
    {
       _isNeedClearDisp1 = 1;
    }
    _disp3oldLength = _dispTempLength1;
    _lcd1.setCursor(0, 1);
    _lcd1.print((_gtv2));
  }
  else
  {
    if (_disp3oldLength > 0)
    {
       _isNeedClearDisp1 = 1;
       _disp3oldLength = 0;
    }
  }
  if (1)
  {
    _dispTempLength1                                                    =
(((_floatToStringWitRaz((_d18x2x2O),1)))).length();
    if (_disp5oldLength > _dispTempLength1)
    {
       _isNeedClearDisp1 = 1;
    }
    _disp5oldLength = _dispTempLength1;
    _lcd1.setCursor(0, 0);
```

```
_lcd1.print(((_floatToStringWitRaz((_d18x2x2O),1))));          //выводим реальную температуру на дисплей
}
 else
{
   if (_disp5oldLength > 0)
   {
      _isNeedClearDisp1 = 1;
      _disp5oldLength = 0;
   }
}
bool  _tmp2 = !(_bounseInputD2O);
if (_tmp2)
{
    if (! _trgt2I) _trgt2 = ! _trgt2;
}
_trgt2I = _tmp2;
DT_65759081_1 =  (digitalRead (3));  //указываем данные для работы с энкодером
CLK_65759081_1 =  (digitalRead (4));
En_65759081_1 = _trgt2;
Up_65759081_1  =0;
Down_65759081_1  =0;
if(En_65759081_1)
{
   bitWrite(New_65759081_1, 0,  DT_65759081_1);
   bitWrite(New_65759081_1, 1,  CLK_65759081_1);
```

```
switch(EncState_65759081_1)

 {
     case 2:

   {
         if(New_65759081_1 == 3) plus_65759081_1();
         if(New_65759081_1 == 0) minus_65759081_1();
         break;

   }
    case 0:

   {
         if(New_65759081_1 == 2) plus_65759081_1();
         if(New_65759081_1 == 1) minus_65759081_1();
         break;

   }
    case 1:

   {
         if(New_65759081_1 == 0) plus_65759081_1();
         if(New_65759081_1 == 3) minus_65759081_1();
         break;

   }
```

```
case 3:

    {
        if(New_65759081_1 == 1) plus_65759081_1();
        if(New_65759081_1 == 2) minus_65759081_1();
        break;

    }

  }
  EncState_65759081_1 = New_65759081_1;

 }
 if (1)
 {
  _tempVariable_bool  =  1;
  if (! _MenuBlock_216651450_OEIS)      //работа блока меню
  {
    _MenuBlock_216651450_OEIS = 1;
  }
  _tempVariable_int                                                    =
pgm_read_byte(&_menuParametrsArray[((_MainMenus[0].currentItem).startInArr
ayIndex)+10]);
  _MenuBlock_216651450_MNO        =        _readStringFromProgmem
((char*)pgm_read_word(&(_flprogMenuStringsArray[_tempVariable_int - 1])));
  _MenuBlock_216651450_VNO = _menuOutputValueString (0);
 }
```

```
else
   {
      _tempVariable_bool  =  0;
      if (_MenuBlock_216651450_OEIS)
      {
         _MenuBlock_216651450_OEIS = 0;
         _menuUpdateToEEpromItems();
      }
      _MenuBlock_216651450_MNO = "";
      _MenuBlock_216651450_VNO = "";
   }
   if(Up_65759081_1)
   {
       if (! _MenuBlock_216651450_OVUIS)
      {
         _MenuBlock_216651450_OVUIS = 1;
         if(_tempVariable_bool)
         {
              _valueUpEvents(0);
         }
      }
   }
    else
   {
      _MenuBlock_216651450_OVUIS = 0;
   }
   if(Down_65759081_1)
```

```
{
    if (! _MenuBlock_216651450_OVDIS)
    {
       _MenuBlock_216651450_OVDIS = 1;
       if(_tempVariable_bool)
       {
            _valueDownEvents(0);
       }
    }
}
 else
{
    _MenuBlock_216651450_OVDIS = 0;
}
if (1)
{
    _dispTempLength1 = ((_MenuBlock_216651450_VNO)).length();
    if (_disp2oldLength > _dispTempLength1)
    {
       _isNeedClearDisp1 = 1;
    }
    _disp2oldLength = _dispTempLength1;
    _lcd1.setCursor(12, 1);
    _lcd1.print((_MenuBlock_216651450_VNO));
}
 else
{
```

```
        if (_disp2oldLength > 0)
            {
               _isNeedClearDisp1 = 1;
               _disp2oldLength = 0;
            }
         }
         if (1)
         {
            _dispTempLength1 = ((_MenuBlock_216651450_MNO)).length();
            if (_disp1oldLength > _dispTempLength1)
            {
               _isNeedClearDisp1 = 1;
            }
            _disp1oldLength = _dispTempLength1;
            _lcd1.setCursor(6, 1);                    //указываем место вывода
параметра меню на LCD экран
            _lcd1.print((_MenuBlock_216651450_MNO));        // выводим меню
на экран
         }
          else
         {
            if (_disp1oldLength > 0)
            {
               _isNeedClearDisp1 = 1;
               _disp1oldLength = 0;
            }
         }
```

```
        if (1)
        {
          _dispTempLength1 = (String("Real t")).length();
          if (_disp6oldLength > _dispTempLength1)
          {
            _isNeedClearDisp1 = 1;
          }
          _disp6oldLength = _dispTempLength1;
          _lcd1.setCursor(5, 0);              //указываем место вывода текста на
экран
          _lcd1.print(String("Real t"));      //указываем текст выводимый на
экран для реальной температуры
        }
        else
        {
          if (_disp6oldLength > 0)
          {
            _isNeedClearDisp1 = 1;
            _disp6oldLength = 0;
          }
        }
      }
      String  _floatToStringWitRaz(float value, int raz)
      {
        return String(value,raz);
      }
      bool _isTimer(unsigned long startTime, unsigned long period)
```

```
{
   unsigned long currentTime;
   currentTime = millis();
   if (currentTime>= startTime)
   {
      return (currentTime>=(startTime + period));
   }
    else
   {
      return (currentTime >=(4294967295-startTime+period));
   }
}
String _readStringFromProgmem (char *string)
{
   String result = String("");
   while (pgm_read_byte(string)!='\0')
   {
      result=result+ char(pgm_read_byte(string));
        string++;

   }
   return result;
}
void _valueUpEvents (int menuIndex)
{
```

```
        int valIndex = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+2]);
        int itemType = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+1]);
        int indexMax = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+6]);
        int indexStep = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+8]);
        if (itemType == 8)
        {
          if (! indexMax == 0)
          {
            if (! (float(pgm_read_float(&_menuConstantValuesArray_float[indexMax -1])) > float(_menuValueArray_float[valIndex -1])))
            {
              return;
            }
          }
          _menuValueArray_float[valIndex -1] = _menuValueArray_float[valIndex -1] + (pgm_read_float(&_menuConstantValuesArray_float[indexStep -1]));
        }
```

```
}
void _valueDownEvents (int menuIndex)
{
   int valIndex = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+2]);
   int itemType = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+1]);
   int indexMin = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+7]);
   int indexStep = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+8]);
   if (itemType == 8)
   {
      if (! indexMin == 0)
      {
         if (! ((float(pgm_read_float(&_menuConstantValuesArray_float[indexMin - 1]))) < float(_menuValueArray_float[valIndex - 1])))
         {
            return;
         }
      }
```

```
_menuValueArray_float[valIndex - 1] = _menuValueArray_float[valIndex - 1] - (pgm_read_float(&_menuConstantValuesArray_float[indexStep - 1]));
        }
    }
    String _menuOutputValueString (int menuIndex)
    {
        int itemType = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+1]);
        int valIndex = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+2]);
        int indexMin = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+7]);
        int indexMax = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+6]);
        if(valIndex == 0)
        {
            return "";
        }
        int convFormat = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+9]);
        if(itemType == 8)
        {
```

```
            return _convertFloat(itemType, convFormat, valIndex, indexMax,
indexMin);
          }
          return "";
        }
        void _menuUpdateToEEpromItems()
        {
          (updateFloatToEEPROM(1, 0, 0x0, ((_menuValueArray_float[0]))));
        }
        String _convertNamber(int itemType, int convFormat, int valIndex, int
indexMax, int indexMin)
        {
          if (itemType== 8)
          {
            if (convFormat == 4)
            {
                return String((_menuValueArray_float[valIndex - 1 ]),DEC);
            }
            if (convFormat == 5)
            {
                return String((_menuValueArray_float[valIndex - 1]),HEX);
            }
            if (convFormat == 6)
            {
                return String((_menuValueArray_float[valIndex -1]),BIN);
            }
          }
```

```
}
String _convertFloat(int itemType, int convFormat, int valIndex , int indexMax, int indexMin)
{
   return String((_menuValueArray_float[valIndex -1]),convFormat);
}
float _convertDS18x2xData(byte type_s, byte data[12])
{
   int16_t raw = (data[1] << 8) | data[0];
   if (type_s)
   {
      raw = raw << 3;
      if (data[7] == 0x10)
      {
          raw = (raw & 0xFFF0) + 12 - data[6];
      }
   }
   else
   {
      byte cfg = (data[4] & 0x60);
      if (cfg == 0x00)raw = raw & ~7;
      else if(cfg == 0x20)raw = raw & ~3;
      else if(cfg == 0x40) raw = raw & ~1;
   }
   return  (float)raw / 16.0;
}
float _readDS18_ow7(byte addr[8], byte type_s)
```

```
{
   byte data[12];
   byte i;
   _ow7.reset();          //очищаем шину
   _ow7.select(addr);
   _ow7.write(0xBE);      //отправляем код на датчик температуры
   for (i = 0; i < 9; i++)
   {
       data[i] = _ow7.read();
   }
   _ow7.reset();                  //очищаем шину
   _ow7.select(addr);
   _ow7.write(0x44, 1);       //отправляем код на датчик температуры
   if (_ow7.crc8(data, 8) != data[8])
   {
      return 501;
   }
   return _convertDS18x2xData(type_s, data);
}
void plus_65759081_1()               //действие + для энкодера
{
   ps_65759081_1++;
   if(ps_65759081_1>= 2)
   {
      Up_65759081_1  =1;
      ps_65759081_1=0;
   }
```

```
}
void minus_65759081_1()              //действие - для энкодера
{
  ms_65759081_1++;
  if(ms_65759081_1>= 2)
  {
    Down_65759081_1  =1;
    ms_65759081_1=0;
  }
}
byte readByteFromEEPROM(int addres, byte bitAddres, byte chipAddres)
//считываем данные переменных из eeprom
{
  return EEPROM.read(addres);
}
void updateByteToEEPROM(int addres, byte bitAddres, byte chipAddres,
byte value)
{
  return EEPROM.update(addres, value);
}
float readFloatFromEEPROM(int addres, byte bitAddres, byte chipAddres)
{
  byte x[4];
  for(byte i = 0; i < 4; i++)
  {
    x[i] = readByteFromEEPROM((addres+i),  bitAddres,  chipAddres);
  }
```

```
    float *y = (float *)&x;
    return y[0];
}
void updateFloatToEEPROM(int addres, byte bitAddres, byte chipAddres, float value)  //записываем данные переменных в eeprom
{
    byte *x = (byte *)&value;
    for(byte i = 0; i < 4; i++)
    {
      updateByteToEEPROM((addres+i),  bitAddres,  chipAddres, x[i]);
    }
}
```