МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ГРОДНЕНСКИЙ ГОСУДАРСТВЕНЫЙ УНИВЕРСИТЕТ ИМЕНИ ЯНКИ КУПАЛЫ»

УДК 681.518.3

Физико-технический факультет

Кафедра общей физики

**Калинов Александр Алексеевич**

**АВТОМАТИЗИРОВАННАЯ УСТАНОВКА ДЛЯ ОПРЕДЕЛЕНИЯ КОРОЗИЙНОЙ СТОЙКОСТИ МАТЕРИАЛОВ**

Дипломная работа

студента 4-го курса 1-ой группы дневного отделения

Научный руководитель:   
доцент кафедры общей физики,  
канд. физ.-мат. наук, Герман А.Е.

ГРОДНО 2017

**РЕФЕРАТ**

**Дипломная работа** ? стр., ? рис., ?табл., 11 ист.

**МИКРОКОНТРОЛЛЕР, ШАГОВЫЙ МОТОР, ДАТЧИКИ, АРДУИНО, КОРОЗИЙНАЯ ПРОЧНОСТЬ**

Целью данной работы является разработка и изготовление недорогого устройства, с помощью которого можно исследовать коррозийную стойкость материалов. Устройство погружает заготовку в агрессивную среду на определенное количество времени, а затем поднимает его, данный процесс повторяется необходимое количество раз. Так же установка включает в себя термостат для поддержания необходимой температуры процесса.

В первой главе рассмотрено понятие коррозийной стойкости. Во второй главе рассмотрены используемые датчики и их краткая характеристика, сделан обзор устройства, принципа действия и управления шагового двигателя, рассмотрена аппаратная платформа Arduino, среда разработки программного обеспечения под платформу Arduino и среда разработки Visual Studio, рассмотрен контроллер Atmega328 фирмы Atmel. В третей главе описано изготовление печатных плат, устройство и принцип работы системы мониторинга.

**Содержание**

|  |  |
| --- | --- |
| **Перечень условных обозначений** | 4 |
| **ВВедение** | 5 |
| **1.Корозийная стойкость** | 6 |
| **2.Элементная база** | 6 |
| 2.1 Шаговый двигатель | 8 |
| 2.2 Датчик температуры DS18B20 | 11 |
| 2.3 Нагревательный элемент | 14 |
| 2.4 Платформа Arduino | 15 |
| 2.5Микроконтроллер ATMEGA328 | 18 |
| 2.6 Среда разработки Arduino | 20 |
| **3.Конструктивное исполнение** | 22 |
| 3.1 Изготовление печатных плат | 22 |
| 3.2 Устройство и принцип работы | 22 |
| **4.Энергосбережение** | 25 |
| **Заключение** | 26 |
| **Перечень использованных источников** | 27 |
| **Приложение а. Схема электическая Arduino uno** |  |
| **Приложение Б. Листинг кода контроллера** |  |
| **Приложение В. листинг кода приложения** |  |
| **Приложение Г.** |  |

**Перечень условных обозначений**

|  |  |
| --- | --- |
| **АЦП** | Аналого-цифровой преобразователь |
| **ЛУТ** | Лазерно-утюжная технология |
| **ПК** | Персональный компьютер |
| **ШИМ** | Широтно-импульсная модуляция |
| **AVR** | Семейство восьмибитных микроконтроллером фирмы Atmel |
| **COM** | Последовательный порт компьютера |
| **IDE** | Интегрированная среда разработки |
| **I2C** | Последовательная шина данных |
| **UART** | Универсальный асинхронный приемопередатчик |
| **USB** | Универсальная последовательная шина данных |
| **SPI** | Последовательный синхронный интерфейс |

**МК** Микроконтроллер

**ВВЕДЕНИЕ**

Целью данной работы является разработка и изготовление недорогого устройства, с помощью которого можно исследовать коррозийную стойкость материалов. Для исследования коррозийной стойкости материалы нужно подвергать продолжительному, циклическому влиянию агрессивных сред. Данное устройство позволяет автоматизировать данную задачу. Устройство погружает заготовку в агрессивную среду на определенное количество времени, а затем поднимает его, данный процесс повторяется необходимое количество раз. Так же установка обладает термостатом, что позволяет поддерживать оптимальную температуру на протяжении всего процесса. Так как устройство базируется на аппаратной платформе Arduino, то персональный компьютер используется только для начальной установки параметров, а затем устройство может быть отключено от компьютера и работать автономно.

Данное устройство будет использоваться в научной лаборатории и позволит упростить и автоматизировать процесс проверки коррозионной стойкости материалов. Данное устройство имеет высокую надежность, хорошую ремонтопригодность, малое потребление электроэнергии и низкую стоимость.

**1. Корозийная стойкость**

Коррозией называется разрушение материалов вследствие химического или электрохимического взаимодействия их с коррозионной средой. Коррозия материалов приводит к их безвозвратным потерям, выводу из строя дорогостоящих машин, аппаратов, оборудования и сооружений. Потери от коррозии можно условно разделить на две группы: прямые и косвенные.

К прямым потерям можно отнести преждевременный выход из строя конструкций, стоимость которых значительно превышает стоимость материалов, которые использовались на их изготовление, расходы на ремонт и замену прокоррозировавших частей машин и оборудования. Прямые потери от коррозии составляют до 10% общего объема выплавляемого металла в стране. К косвенным потерям от коррозии следует отнести простой оборудования в результате аварий, порчу пищевых и химических веществ вследствие загрязнения их продуктами коррозии, снижение мощности машин и оборудования, потери вследствие нарушения герметичности резервуаров, трубопроводов и др. В ряде случаев косвенные потери в 100—1000 раз превышают прямые потери. Поэтому очень важно определить механизм коррозионного разрушения и исходя из этого правильно выбрать метод защиты с целью увеличения срока службы конструкций.

Под коррозионной стойкостью понимается способность материала сопротивляться коррозионному воздействию среды. Ввиду большого разнообразия видов коррозионных разрушений невозможно установить единую меру оценки коррозионной стойкости материалов

Существуют качественные и количественные методы оценки коррозионного процесса. Методы качественной оценки играют вспомогательную, хотя и весьма существенную роль, позволяя получить представление о характере и интенсивности процесса. Наиболее распространенными среди методов количественной оценки коррозии являются весовой и объемный, а также метод, учитывающий изменение механических или физических свойств корродирующего материала.

**Качественная оценка**. Сущность этих методов сводится к визуальному, макроскопическому (с помощью лупы) или микроструктурному контролю:

* Состояния поверхности детали (определение степени равномерности коррозии, характера продуктов коррозии и прочности сцепления их с материалом и др.)
* Состояния раствора, в котором находится испытуемый материал (появление продуктов коррозии в виде осадка, мути и др.)
* Изменения цвета индикаторных растворов.

Индикаторные растворы применяются для определения анодного и катодного участков на корродирующем материале. Так, если поверхность стальной детали покрыть раствором, содержащим красную кровяную соль и фенолфталеин, то на анодном участке появляется синее окрашивание с образованием турнбуллевой сини; катодный участок розовеет, так как при коррозии с кислородной деполяризацией происходит подщелачивание раствора.

Коррозионные испытания проводят с целью выяснения механизма коррозии, определения скорости протекания коррозионных разрушений, нахождения факторов, влияющих на коррозионное поведение материала. Первым этапом коррозионных испытаний могут быть испытания лабораторных образцов или реальных деталей с целью определить изменение скорости коррозии во времени. При этом применяются все те количественные оценки, которые приведены выше. Проводимые коррозионные испытания можно подразделить на три группы — лабораторные, полевые и эксплуатационные.

**2.Элементная база**

**2.1 Шаговый двигатель**

Шаговый электродвигатель - это синхронный бесщёточный электродвигатель с несколькими обмотками, в котором ток, подаваемый в одну из обмоток статора, вызывает фиксацию ротора. Последовательная активация обмоток двигателя вызывает дискретные угловые перемещения (шаги) ротора.

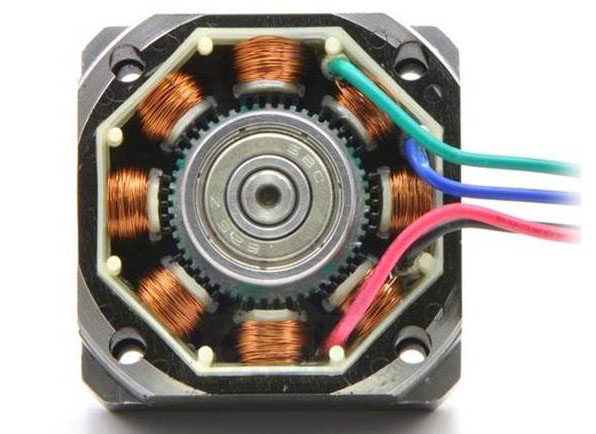


Рисунок 1. Шаговый двигатель

**Достоинства шаговых двигателей:**

* Угол поворота ротора определяется числом импульсов, которые поданы на двигатель.
* Двигатель обеспечивает полный момент в режиме остановки (если обмотки запитаны).
* Прецизионное позиционирование и повторяемость. Хорошие шаговые двигатели имеют точность 3-5% от величины шага. Эта ошибка не накапливается от шага к шагу.
* Возможность быстрого старта/остановки/реверсирования.
* Высокая надежность, связанная с отсутствием щеток, срок службы шагового двигателя фактически определяется сроком службы подшипников.
* Однозначная зависимость положения от входных импульсов обеспечивает позиционирование без обратной связи
* Возможность получения очень низких скоростей вращения для нагрузки, присоединенной непосредственно к валу двигателя без промежуточного редуктора.
* Может быть перекрыт довольно большой диапазон скоростей, скорость пропорциональна частоте входных импульсов.

**Недостатки шаговых двигателей:**

* Шаговым двигателем присуще явление резонанса.
* Возможна потеря контроля положения ввиду работы без обратной связи.
* Потребление энергии не уменьшается даже без нагрузки.
* Затруднена работа на высоких скоростях.
* Невысокая удельная мощность.
* Относительно сложная схема управления.

**Существуют три основных типа шаговых двигателей**:

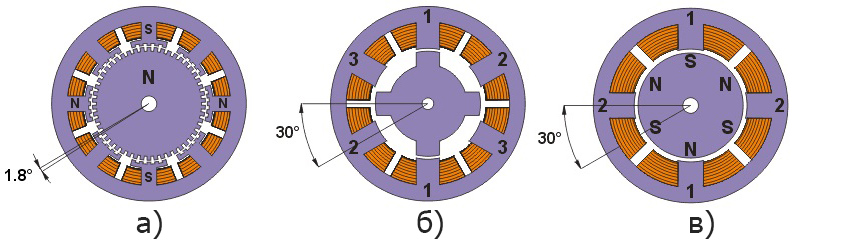
* Двигатели с переменным магнитным сопротивлением.
* Двигатели с постоянными магнитами.
* Гибридные двигатели.

Рисунок 2. Виды двигателей. а) гибридный б) с переменным магнитным сопротивлением в) с постоянными магнитами

**Подключение обмоток.**

В зависимости от конфигурации обмоток двигатели делятся на биполярные и униполярные. Биполярный двигатель имеет одну обмотку в каждой фазе, которая для изменения направления магнитного поля должна переполюсовывается драйвером. Для такого типа двигателя требуется мостовой драйвер, или полумостовой с двухполярным питанием. Всего биполярный двигатель имеет две обмотки и, соответственно, четыре вывода.

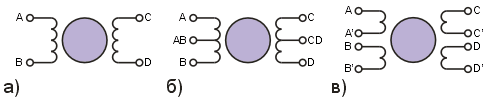


Рисунок 3. Биполярный двигатель (а), униполярный (б) и четырехобмоточный (в)

Если сравнивать между собой биполярный и униполярный двигатели, то биполярный имеет более высокую удельную мощность. При одних и тех же размерах биполярные двигатели обеспечивают больший момент. Момент, создаваемый шаговым двигателем, пропорционален величине магнитного поля, создаваемого обмотками статора. Путь для повышения магнитного поля – это увеличение тока или числа витков обмоток. Естественным ограничением при повышении тока обмоток является опасность насыщения железного сердечника. Однако на практике это ограничение действует редко. Гораздо более существенным является ограничение по нагреву двигателя вследствие омических потерь в обмотках. Как раз этот факт и демонстрирует одно из преимуществ биполярных двигателей. В униполярном двигателе в каждый момент времени используется лишь половина обмоток. Другая половина просто занимает место в окне сердечника, что вынуждает делать обмотки проводом меньшего диаметра. В то же время в биполярном двигателе всегда работают все обмотки, т.е. их использование оптимально. В таком двигателе сечение отдельных обмоток вдвое больше, а омическое сопротивление – соответственно вдвое меньше. Это позволяет увеличить ток в корень из двух раз при тех же потерях, что дает выигрыш в моменте примерно 40%. Если же повышенного момента не требуется, униполярный двигатель позволяет уменьшить габариты или просто работать с меньшими потерями. На практике все же часто применяют униполярные двигатели, так как они требуют значительно более простых схем управления обмотками. Это важно, если драйверы выполнены на дискретных компонентах. В настоящее время существуют специализированные микросхемы драйверов для биполярных двигателей, с использованием которых драйвер получается не сложнее, чем для униполярного двигателя.

**Режимы управления шаговым двигателем.**

Полношаговое управление одной обмоткой. Этот способ описан выше и называется волновым управлением одной обмоткой. Это означает, что только через одну обмотку протекает электрический ток. Этот способ используется редко. В основном, к нему прибегают в целях снижения энергопотребления. Такой метод позволяет получить менее половины вращающего момента мотора, следовательно, нагрузка мотора не может быть значительной.

Вторым, и наиболее часто используемым методом, является полношаговый метод. Для реализации этого способа, напряжение на обмотки подается попарно. В зависимости от способа подключения обмоток (последовательно или параллельно), мотору потребуется двойное напряжение или двойной ток для работы по отношению к необходимым при возбуждении одной обмотки. В этом случае мотор будет выдавать 100% номинального вращающего момента

Полушаговый режим. Это очень интересный способ получить удвоенную точность системы позиционирования.Для реализации этого метода, все пары обмоток могут запитываться одновременно, в результате чего, ротор повернется на половину своего нормального шага. Этот метод может быть также реализован с использованием одной или двух обмоток.

Микрошаговый режим. Идея микрошага состоит в подаче на обмотки мотора питания не импульсами, а сигнала, по своей [форме](http://robotosha.ru/electronics/electrical-waveforms.html), напоминающего синусоиду. Такой способ изменения положения при переходе от одного шага к другому позволяет получить более гладкое перемещение, делая шаговые моторы широко используемыми в таких приложениях как системы позиционирования в станках с ЧПУ. Кроме этого, рывки различных деталей, подключенных к мотору, также, как и толчки самого мотора значительно снижаются. В режиме микрошага, шаговый мотор может вращаться также плавно, как и обычные тока. Форма тока, протекающего через обмотку похожа на синусоиду. Также могут использоваться формы цифровых сигналов.

В нашем устройстве будет использоваться униполярный шаговый двигатель с постоянными магнитами. Управление будет осуществляться в полушаговом режиме с помощью аппаратной платформы Arduino. Драйвер шагового двигателя будет выполнен на 4 полевых транзисторах IRF3205.

**2.2 Датчик температуры DS18B20**

DS18B20 цифровой термометр с программируемым разрешением, от 9 до 12–bit, которое может сохраняться в EEPROM памяти прибора. DS18B20 обменивается данными по 1-Wire шине и при этом может быть как единственным устройством на линии так и работать в группе. Все процессы на шине управляются центральным микропроцессором. Диапазон измерений от –55°C до +125°C и точностью 0.5°C в диапазоне от –10°C до +85°C. В дополнение, DS18B20 может питаться напряжением линии данных (“parasite power”), при отсутствии внешнего источника напряжения. Каждый DS18B20 имеет уникальный 64-битный последовательный код, который позволяет, общаться с множеством датчиков DS18B20 установленных на одной шине. Такой принцип позволяет использовать один микропроцессор, чтобы контролировать множество датчиков DS18B20, распределенных по большому участку. Приложения, которые могут извлечь выгоду из этой особенности, включают системы контроля температуры в зданиях, и оборудовании или машинах, а также контроль и управление температурными процессами.

Таблица 1. Назначение выводов датчика DS18B20

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SO\* | SOP\* | TO-92 | СИМВОЛ | ОПИСАНИЕ |
| 5 | 4 | 1 | GND | Общий. |
| 4 | 1 | 2 | DQ | Вывод данных ввода/вывода (Input/Output pin). Open-drain 1- Wire interface pin. По этой линии подается питание в режиме работы с паразитным питанием. |
| 3 | 8 | 3 | VDD | VDD Вывод питания. Для режима работы с паразитным питание VDD необходимо соединить с общим проводом. |

\*Все остальные выводы должны оставаться не подключенными.

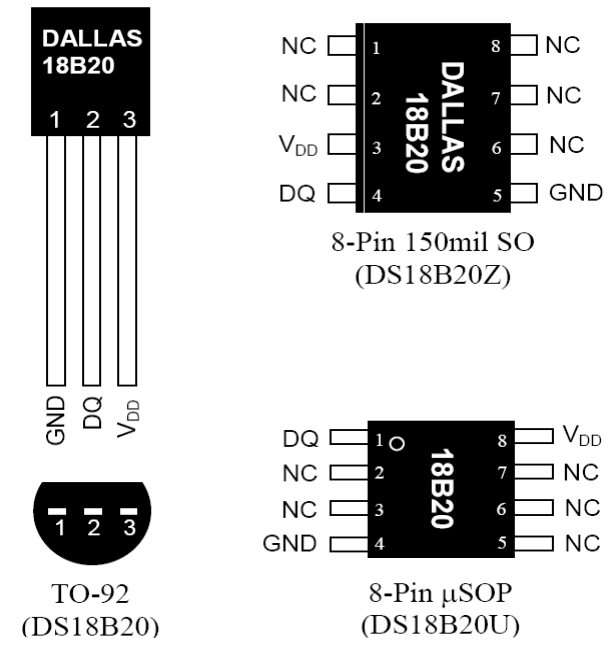


Рисунок 4. Датчик DS18B20

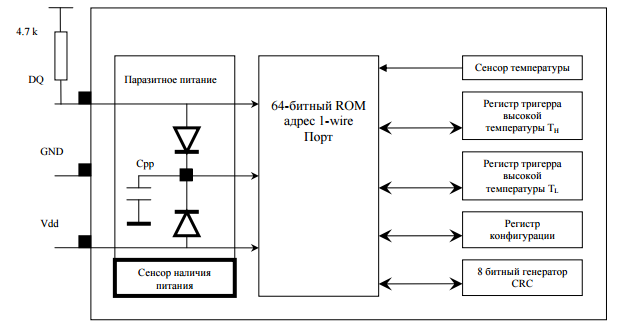
Рисунок 5 показывает блок-схему DS18B20, и описания выводов даются в Таблице 1. 64-битовый ROM запоминает уникальный последовательный код прибора. Оперативная память содержит 2-байтовый температурный регистр, который запоминает цифровой выход от температурного преобразователя. Кроме того, электронный блокнот обеспечивает доступ к 1-байтовым сигнальным регистрам триггерной схемы (TH и TL), и к регистру конфигурации. Регистр конфигурации позволяет пользователю устанавливать разрешающую способность цифрового преобразователя температуры к 9, 10, 11, или 12битам. TH, TL и регистры конфигурации энергонезависимы (EEPROM), таким образом они сохранят данные, когда прибор - выключен. DS18B20 использует исключительно 1-Wire протокол – при этом формируется соединение, которое осуществляет коммуникацию на шине, используя всего один управляющий сигнал. Шина должна быть подключена к источнику питания через подтягивающий резистор, так как все приборы связаны с шиной, используют соединение через Z-состояния или вход открытого стока. Используя эту шину микропроцессор (ведущий) идентифицирует и обращается к датчикам температуры, используя 64-битовыйкод прибора. Поскольку каждый прибор имеет уникальный код, число приборов, к которым можно обратиться на одной шине, фактически неограниченно. Другая особенность DS18B20 - способность работать без внешнего питания. Эта возможность предоставляется через подтягивающий резистор. Высокий сигнал шины заряжает внутренний конденсатор (CPP), который питает прибор, когда на шине низкий уровень. Этот метод носит название «Паразитное питание». При этом максимальная измеряемая температура при этом + 100 °C. Для расширения диапазона температур до +125 °C необходимо использовать внешнее питание.

Рисунок 5. Блок-схема DS18B20

Основные функциональные возможности DS18B20 - его температурный преобразователь. Разрешающая способность температурного преобразователя может быть изменена пользователем и составляет 9, 10, 11, или 12 битов, соответствуя приращениям (дискретности измерения температуры) 0.5 °C, 0.25°C, 0.125°C, и 0.0625°C, соответственно. Разрешающая способность по умолчанию установлена 12-бит. В исходном состоянии DS18B20 находится в состоянии покоя (в неактивном состоянии). Чтобы начать температурное измерение и преобразование, ведущий должен подать команду начала конвертирования температуры [0х44]. После конвертирования, полученные данные запоминаются в 2-байтовом регистре температуры в оперативной памяти, и DS18B20 возвращается к его неактивному состоянию. Если DS18B20 включен с внешним питанием, ведущий может контролировать конвертирование температуры (после команды [0х44]) по состоянию шины. На шине будет присутствовать логический «0» когда происходит температурное преобразование. И логическая «1», когда конвертирование выполнено. Если DS18B20 включен с паразитным питанием, эта технология уведомления не может быть использована, так как шину нужно подать высокий уровень (напряжение питания) в течение всего времени температурного преобразования. Выходные температурные данные DS18B20 калиброваны в градусах Цельсия. Температурные данные запоминаются как 16-битовое число со знаком (см. иллюстрацию2). Биты признака (S) указывают, является ли температура положительная или отрицательная: для положительных S = 0, а для отрицательных чисел S = 1. ЕслиDS18B20 будет настроен для конвертирования 12-битной разрешения, то все биты в температурном регистре будут содержать действительные данные. Для 11-битнойразрешающей способности, бит 0 не определен. Для 10-битной разрешающей способности, биты 1 и 0 неопределенны, и для 9 битной разрешающей способности 2, 1 и 0 неопределенны.

**2.2 Нагревательный элемент**

Нагревательный элемент изготовлен из равномерно намотанной на стеклянный стакан нихромовой проволоки.

Нихром — общее название группы сплавов, состоящих, в зависимости от марки сплава, из 55—78 % никеля, 15—23 % хрома, с добавками марганца, кремния, железа, алюминия.

**Физические свойства нихрома:**

* Удельное электрическое сопротивление — 1,05—1,4 Ом (в зависимости от марки сплава)·длина в метрах/сечение в мм².
* Температурный коэффициент сопротивления — 0,1 — 0,25 \* 1/К.
* Плотность — 8200—8500 кг/м³.
* Температура плавления — 1100—1400 °C.
* Рабочая температура — 800—1100 °C.
* Удельная теплоемкость — 450 Дж/(кг·К) при 25 °C.
* Предел прочности при растяжении — 0,65-0,70 ГПа.

Используемая проволока обладает сопротивлением около 2 Ом/м. Исходя из того, что мы имеем источник питания 24 В и нам необходим нагреватель с мощностью 30-50 Вт, было намотано около 8 метров проволоки.

**2.3 Используемая платформа Arduino**

Arduino — торговая марка аппаратно-программных средств для построения простых систем автоматики и [робототехники](https://ru.wikipedia.org/wiki/%D0%A0%D0%BE%D0%B1%D0%BE%D1%82%D0%BE%D1%82%D0%B5%D1%85%D0%BD%D0%B8%D0%BA%D0%B0). Программная часть состоит из бесплатной программной оболочки ([IDE](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8)) для написания программ, их компиляции и программирования аппаратуры. Аппаратная часть представляет собой набор смонтированных [печатных плат](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%87%D0%B0%D1%82%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D0%B0), продающихся как официальным производителем, так и сторонними производителями. Полностью [открытая архитектура](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D0%B0%D1%8F_%D0%B0%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0) системы позволяет свободно копировать или дополнять линейку продукции Arduino.

****

Рисунок 6. Arduino Uno

Arduino Uno контроллер построен на ATmega328. Платформа имеет 14 цифровых вход/выходов (6 из которых могут использоваться как выходы ШИМ), 6 аналоговых входов, кварцевый генератор 16 МГц, разъем USB, силовой разъем, разъем ICSP и кнопку перезагрузки. Для работы необходимо подключить платформу к компьютеру посредством кабеля USB, либо подать питание при помощи адаптера AC/DC или батареи.

**Питание**

Arduino Uno может получать питание через подключение USB или от внешнего источника питания. Источник питания выбирается автоматически.

Внешнее питание (не USB) может подаваться через преобразователь напряжения AC/DC (блок питания) или аккумуляторной батареей. Преобразователь напряжения подключается посредством разъема 2.1 мм с центральным положительным полюсом. Провода от батареи подключаются к выводам Gnd и Vin разъема питания.

Платформа может работать при внешнем питании от 6 В до 20 В. При напряжении питания ниже 7 В, вывод 5V может выдавать менее 5 В, при этом платформа может работать нестабильно. При использовании напряжения выше 12 В регулятор напряжения может перегреться и повредить плату. Рекомендуемый диапазон от 7 В до 12 В.

Выводы питания:

**VIN**. Вход используется для подачи питания от внешнего источника (в отсутствие 5 В от разъема USB или другого регулируемого источника питания). Подача напряжения питания происходит через данный вывод.

**5V**. Регулируемый источник напряжения, используемый для питания микроконтроллера и компонентов на плате. Питание может подаваться от вывода VIN через регулятор напряжения, или от разъема USB, или другого регулируемого источника напряжения 5 В.

**3V3**. Напряжение на выводе 3.3 В генерируемое встроенным регулятором на плате. Максимальное потребление тока 50 мА.

**GND**. Выводы заземления.

Память

Микроконтроллер ATmega328 располагает 32 кБ флэш памяти, из которых 0.5 кБ используется для хранения загрузчика, а также 2 кБ ОЗУ (SRAM) и 1 Кб EEPROM.(которая читается и записывается с помощью [библиотеки EEPROM](http://arduino.ru/Reference/Library/EERPOM)).

**Входы и Выходы**

Каждый из 14 цифровых выводов Uno может настроен как вход или выход, используя функции[pinMode()](http://arduino.ru/Reference/PinMode), [digitalWrite()](http://arduino.ru/Reference/DigitalWrite), и [digitalRead()](http://arduino.ru/Reference/DigitalRead), . Выводы работают при напряжении 5 В. Каждый вывод имеет нагрузочный резистор (по умолчанию отключен) 20-50 кОм и может пропускать до 40 мА. Некоторые выводы имеют особые функции:

Последовательная шина: 0 (RX) и 1 (TX). Выводы используются для получения (RX) и передачи (TX) данных TTL. Данные выводы подключены к соответствующим выводам микросхемы последовательной шины ATmega8U2 USB-to-TTL.

Внешнее прерывание: 2 и 3. Данные выводы могут быть сконфигурированы на вызов прерывания либо на младшем значении, либо на переднем или заднем фронте, или при изменении значения.

ШИМ: 3, 5, 6, 9, 10, и 11. Любой из выводов обеспечивает [ШИМ](http://arduino.ru/Tutorial/PWM) с разрешением 8 бит при помощи функции [analogWrite()](http://arduino.ru/Reference/AnalogWrite).

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Посредством данных выводов осуществляется связь SPI, для чего используется [библиотека SPI](http://arduino.ru/Reference/Library/SPI).

LED: 13. Встроенный светодиод, подключенный к цифровому выводу 13. Если значение на выводе имеет высокий потенциал, то светодиод горит.

На платформе Uno установлены 6 аналоговых входов (обозначенных как A0 .. A5), каждый разрешением 10 бит (т.е. может принимать 1024 различных значения). Стандартно выводы имеют диапазон измерения до 5 В относительно земли, тем не менее имеется возможность изменить верхний предел посредством вывода AREF и функции [analogReference()](http://arduino.ru/Reference/AnalogReference). Некоторые выводы имеют дополнительные функции:

I2C: 4 (SDA) и 5 (SCL). Посредством выводов осуществляется связь I2C (TWI), для создания которой используется [библиотека Wire.](http://arduino.cc/en/Reference/Wire)

Дополнительная пара выводов платформы:

AREF. Опорное напряжение для аналоговых входов. Используется с функцией [analogReference()](http://arduino.ru/Reference/AnalogReference).

Reset. Низкий уровень сигнала на выводе перезагружает микроконтроллер. Обычно применяется для подключения кнопки перезагрузки на плате расширения, закрывающей доступ к кнопке на самой плате Arduino.

**Связь**

На платформе Arduino Uno установлено несколько устройств для осуществления связи с компьютером, другими устройствами Arduino или микроконтроллерами. ATmega328 поддерживают последовательный интерфейс UART TTL (5 В), осуществляемый выводами 0 (RX) и 1 (TX). Установленная на плате микросхема ATmega8U2 направляет данный интерфейс через USB, программы на стороне компьютера "общаются" с платой через виртуальный COM порт. Прошивка ATmega8U2 использует стандартные драйвера USB COM, никаких стороних драйверов не требуется, но на Windows для подключения потребуется файл ArduinoUNO.inf.  Мониторинг последовательной шины (Serial Monitor) программы Arduino позволяет посылать и получать текстовые данные при подключении к платформе. Светодиоды RX и TX на платформе будут мигать при передаче данных через микросхему FTDI или USB подключение. [1]

Таблица 2. Характеристики Arduino Uno

|  |  |
| --- | --- |
| Микроконтроллер | Atmega328 |
| Рабочее напряжение | 5 В |
| Входное напряжение (рекомендуемое) | 7-12 В |
| Входное напряжение (предельное) | 6-20 В |
| Цифровые Входы/Выходы | 14 (6 из которых могут использоваться как выходы [ШИМ](http://arduino.ru/Tutorial/PWM)) |
| Аналоговые входы | 6 |
| Постоянный ток через вход/выход | 40 мА |
| Постоянный ток для вывода 3.3 В | 50 мА |
| флеш-память | 32 Кб (ATmega328) из которых 0.5 Кб используются для загрузчика |
| ОЗУ | 2 Кб (ATmega328) |
| EPROM | 1 Кб (ATmega328) |
| Тактовая частота | 16 МГц |

**2.2 Микроконтроллер ATMEGA328**

ATmega328P – микроконтроллер семейства AVR 8 битный процессор, позволяет выполнять большинство команд за 1 такт

Память:

* 32 kB Flash (память программ, имеющая возможность самопрограммирования)
* 2 kB ОЗУ
* 1 kB EEPROM (постоянная память данных)
* Периферийные устройства:
* Два 8-битных таймера/счетчика с модулями сравнения и делителями частоты
* 16-битный таймер/счетчик с модулями сравнения и делителями частоты, а так же с режимом записи
* Счетчик реального времени с отдельным генератором
* Шесть каналов PWM (аналог ЦАП)
* 6-канальный ЦАП со встроенным датчиком температуры
* Программируемый последовательный порт UART
* Последовательный интерфейс SPI
* Интерфейс I2C
* Программируемый сторожевой таймер с отдельным внутренним генератором
* Внутренняя схема сравнения напряжений
* Блок обработки прерывания и пробуждения при изменении напряжений на выводах микроконтроллера
* Специальные функции микроконтроллера:
* Сброс при включении питания и программное распознание снижение напряжения питания
* Внутренний калибруемый генератор тактовых импульсов
* Обработка внутренних и внешних прерываний
* 6 режимов сна (пониженное энергопотребление и снижение шумов для более точного преобразования АЦП)

Напряжения питания и скорость процессора:

* 1.8 – 5.5 В при частоте до 4 МГц
* 2.7 – 5.5 В при частоте до 10 МГц
* 4.5 – 5.5 В при частоте до 20 МГц

Система команд микроконтроллеров AVR весьма развита и насчитывает в различных моделях от 90 до 133 различных инструкций.

* Большинство команд занимает только 1 ячейку памяти (16 бит).
* Большинство команд выполняется за 1 [такт](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%BA%D1%82_(%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80)).

Всё множество команд микроконтроллеров AVR можно разбить на несколько групп:

* команды логических операций;
* команды арифметических операций;
* команды пересылки данных;
* команды передачи управления;
* команды управления системой.

Управление периферийными устройствами осуществляется через адресное пространство данных. Для удобства существуют «сокращённые команды» IN/OUT

Программные средства разработки

* [WinAVR](https://ru.wikipedia.org/wiki/WinAVR)— программный пакет под [Windows](https://ru.wikipedia.org/wiki/Windows), включающий в себя компилятор, ассемблер, компоновщик и другие инструменты.
* [Algorithm Builder](https://ru.wikipedia.org/w/index.php?title=Algorithm_Builder&action=edit&redlink=1)— среда разработки программного обеспечения для микроконтроллеров с архитектурой AVR.
* [Code::Blocks](https://ru.wikipedia.org/wiki/Code::Blocks) — кроссплатформенная среда разработки.
* [DDD](https://ru.wikipedia.org/wiki/DDD)— графический интерфейс к avr-gdb.
* [V-USB](https://ru.wikipedia.org/wiki/V-USB)— программная реализация протокола USB для микроконтроллеров AVR.
* [Avrdude](https://ru.wikipedia.org/wiki/Avrdude)— средство для прошивки микроконтроллеров.
* [PonyProg](https://ru.wikipedia.org/wiki/PonyProg)— универсальный программатор через LPT-порт, COM-порт (подерживается и USB-эмулятор COM-порта).[6]

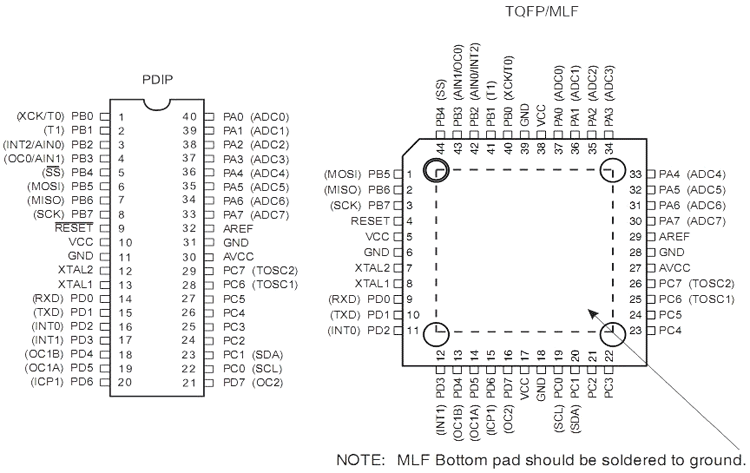


Рисунок 7. Расположение выводов микроконтроллера ATMEGA328[6]

**2.4 Среда разработки Arduino**

Программирование ведется целиком через собственную программную оболочку (IDE), бесплатно доступную на сайте Arduino. В этой оболочке имеется [текстовый редактор](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B9_%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80), менеджер проектов, [препроцессор](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80), [компилятор](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) инструменты для загрузки программы в микроконтроллер. Оболочка написана на [Java](https://ru.wikipedia.org/wiki/Java) на основе проекта [Processing](https://ru.wikipedia.org/wiki/Processing), работает под [Windows](https://ru.wikipedia.org/wiki/Windows), [Mac OS X](https://ru.wikipedia.org/wiki/Mac_OS_X) и [Linux](https://ru.wikipedia.org/wiki/Linux).

Язык программирования Arduino является стандартным [C++](https://ru.wikipedia.org/wiki/C%2B%2B) (используется компилятор [AVR-GCC](https://ru.wikipedia.org/wiki/GNU_AVR_toolchain#avr-gcc)) с некоторыми особенностями, облегчающими новичкам написание первой работающей программы.

Программы, написанные программистом Arduino называются наброски (или иногда скетчи) и сохраняются в файлах с расширением ino. Эти файлы перед компиляцией обрабатываются препроцессором Arduino. Также существует возможность создавать и подключать к проекту стандартные файлы C++.

Обязательную в C++ функцию main () препроцессор Arduino создает сам, вставляя туда необходимые «черновые» действия.

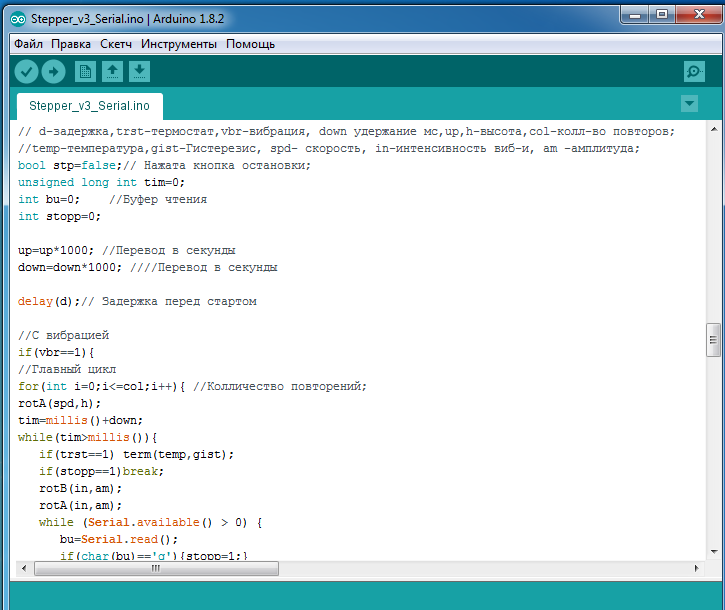


Рисунок 8. Среда разработки Arduino

Программист должен написать две обязательные для Arduino функции setup() и loop(). Первая вызывается однократно при старте, вторая выполняется в бесконечном цикле.

В текст своей программы (скетча) программист не обязан вставлять [заголовочные файлы](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B3%D0%BE%D0%BB%D0%BE%D0%B2%D0%BE%D1%87%D0%BD%D1%8B%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB) используемых стандартных библиотек. Эти заголовочные файлы добавит препроцессор Arduino в соответствии с конфигурацией проекта. Однако пользовательские библиотеки нужно указывать.

Менеджер проекта Arduino IDE имеет нестандартный механизм добавления библиотек. Библиотеки в виде исходных текстов на стандартном C++ добавляются в специальную папку в рабочем каталоге IDE. При этом название библиотеки добавляется в список библиотек в меню IDE. Программист отмечает нужные библиотеки и они вносятся в список компиляции.

Arduino IDE не предлагает никаких настроек компилятора и минимизирует другие настройки, что упрощает начало работы для новичков и уменьшает риск возникновения проблем.

Загрузка программы в микроконтроллер Arduino происходит через предварительно запрограммированный специальный загрузчик (все микроконтроллеры от Arduino продаются с этим загрузчиком). Загрузчик создан на основе Atmel AVR Application Note AN109. Загрузчик может работать через интерфейсы [RS-232](https://ru.wikipedia.org/wiki/RS-232), [USB](https://ru.wikipedia.org/wiki/USB) или [Ethernet](https://ru.wikipedia.org/wiki/Ethernet) в зависимости от состава периферии конкретной процессорной платы. В некоторых вариантах, таких как Arduino Mini или неофициальной Boarduino, для программирования требуется отдельный переходник.

Пользователь может самостоятельно запрограммировать загрузчик в чистый микроконтроллер. Для этого в IDE интегрирована поддержка программатора на основе проекта [AVRDude](https://ru.wikipedia.org/wiki/AVRDude). Поддерживается несколько типов популярных дешёвых программаторов.

Processing — открытый [язык программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), основанный на [Java](https://ru.wikipedia.org/wiki/Java). Представляет собой лёгкий и быстрый инструментарий для людей, которые хотят программировать изображения, анимацию и интерфейсы.

Используется студентами, художниками, дизайнерами, исследователями и любителями, для изучения, прототипирования и производства. Он создан для изучения основ компьютерного программирования в визуальном контексте и служит альбомным программным обеспечением (имеется в виду то, что каждый \*.pde файл визуальной оболочки Processing’а представляет собой отдельное изображение или анимацию, и т. д.) и профессиональным производственным инструментом.

Processing — это открытый проект инициированный [Бенжамином Фраем](https://ru.wikipedia.org/w/index.php?title=%D0%91%D0%B5%D0%BD%D0%B6%D0%B0%D0%BC%D0%B8%D0%BD_%D0%A4%D1%80%D0%B0%D0%B9&action=edit&redlink=1) и [Кэйси Ризом](https://ru.wikipedia.org/w/index.php?title=%D0%9A%D1%8D%D0%B9%D1%81%D0%B8_%D0%A0%D0%B8%D0%B7&action=edit&redlink=1). Он родился из идей, изученных в The Aesthetics and Computation Group в MIT Media Lab.

**3.КОНСТРУКТИВНОЕ ИСПОЛНЕНИЕ**

**3.1 Изготовление печатной платы**

Печатная плата выполнена на одностороннем фольгированном стеклотекстолите при помощи ЛУТ. [?]

Печатные платы спроектированы в программе Sprintloyout. Чертежи печатных плат печатались лазерным принтером на глянцевой бумаге. На заготовку из стеклотекстолита накладывался чертеж печатной платы, после чего производился нагрев с помощью утюга. После нагрева заготовка помещалась под проточную воду и с нее удалялись остатки бумаги. Далее заготовка просушивалась, дефекты, возникшие на предыдущих этапах, убирались с помощью перманентного маркера. Далее заготовка помещалась в резервуар с раствором хлорного железа для травления дорожек. Для ускорения процесса резервуар раствор был нагрет до температуры 45 градусов. После растворения не покрытых участков меди, плату извлекли из раствора и промыли проточной водой.

После удаления тонера с дорожек, плата была обработана наждачной бумагой и обезжирена спиртом. Дорожки покрывались паяльным флюсом, после чего были залужены оловянным припоем. Затем были просверлены отверстия и запаяны радиоэлементы.

Рисунок 9. Печатная плата

**3.2 Устройство и принцип работы**

Устройство представляет из себя четыре блока. Блок управления включает в себя аппаратную платформу Arduino, печатную плату с драйвером шагового двигателя на полевых транзисторах, электромагнитного реле, управляющего нагревательным элементом. Так же к устройству подключен температурный датчик DS18B20. Все компоненты расположены в металлическом корпусе. Второй блок включает в себя униполярный шаговый двигатель с постоянными магнитами, расположенный на металлической площадке, редуктор, маховик с прикреплённой к нему веревкой для крепления заготовок. Третий блок – это стеклянный, термоизолированный стакан с намотанной на него нихромовой проволокой, он служит для поддержания необходимой температуры. Четвертый блок представляет из себя импульсный источник питания с напряжением 12 вольт и максимальным током 10 ампер, а также повышающий преобразователь напряжения с 12 до 24 вольт и максимальной мощностью 130 ватт. Так же устройство имеет USB интерфейс для подключения к компьютеру.

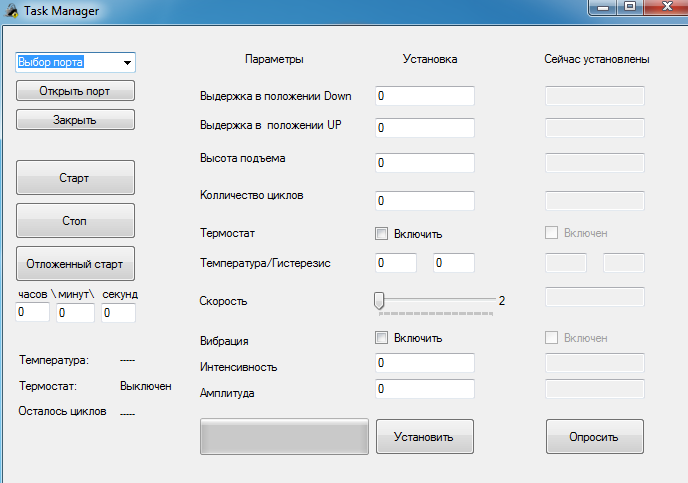
Работой устройства управляет микроконтроллер Atmega328p, листинг программного кода представлен в приложении Б. Для настройки устройства используется компьютер, на котором должна быть установлена программа TaskManager и драйверы для работы с аппаратной платформой Arduino. Программа разработана на языке программирования С# в среде Visual Studio, листинг программного кода представлен в приложении В.

Данное устройство помогает провести испытания образцов на коррозийную стойкость путем многократного погружения их в агрессивную среду. Устройство настраивается с помощью специальной программы, которая может быть установлена на персональный компьютер или ноутбук. Преимуществом данного устройство является то, что компьютер используется только для настройки и после старта установки может быть отключен. Устройство может работать длительное время автономно и после завершения циклов перейдет в режим ожидания.

Порядок работы с устройством следующий:

* Располагаем устройство на рабочем месте так, чтобы металлическая площадка с шаговым двигателем находилась над стеклянным стаканом.
* Подключаем устройство к персональному компьютеру или ноутбуку.
* Выбираем порт, к которому подключено устройство и нажимаем кнопку «открыть порт»
* Вводим необходимые настройки и нажимаем кнопку «Установить» и ожидаем пока пройдет полоса загрузки.
* Нажимаем кнопку «опросить» и проверяем нет ли ошибок, в случае ошибки, перезагружаем контроллер и устанавливаем настройки заново.
* Закрепляем исследуемый материал на конце веревки.
* Наполняем стакан необходимым для эксперимента раствором.
* Нажимаем кнопку «Старт» или «Отложенный старт».
* Ожидаем завершения процесса испытания образца, в случае необходимости завершить процесс досрочно – нажимаем кнопку «Стоп».
* Отключаем установку, отсоединяем образец, моем стакан.

Рисунок 10. Устройство в сборе

Рисунок 11. Интерфейс программы

**4. ЭНЕРГОСБЕРЕЖЕНИЕ**

Электроэнергетика — это стратегическая отрасль, состояние которой отражается на уровне развития государства в целом. В настоящее время электроэнергетика является наиболее стабильно работающим комплексом белорусской экономики. Предприятиями отрасли обеспечено эффективное, надежное и устойчивое энергоснабжение потребителей республики без аварий и значительного экологического ущерба. Главным приоритетом энергетической политики нашего государства является повышение эффективности использования энергии как средства для снижения затрат общества на энергоснабжение, обеспечения устойчивого развития страны, повышения конкурентоспособности производительных сил и охраны окружающей среды.

Одним из важнейших направлений энергосбережения сегодня является замещение дорогостоящих импортируемых энергоресурсов, получение топливной энергии с использованием местных видов топлива, в том числе отходов от производства. С целью достижение существенной экономии энергоресурсов Комитетом по энергоэффективности при совете министров РБ ежегодно определяет приоритетные направления энергосбережения и осуществляется их преимущественная реализация.

В ходе построения установки были использованы различные приемы, позволяющие существенно снизить потребление электроэнергии:

* Использован микроконтроллер с минимальным энергопотреблением.
* Ротор шагового двигателя удерживается редуктором, а с обмоток снимается напряжение, что сокращает электропотребление в несколько раз.
* Изготовлен нагревательный элемент, теплоизолированный от окружающей среды, что повышает его эффективность и снижает энергопотребление.
* После завершения работы установка переходит в режим ожидания, в котором она потребляет минимальное количество электроэнергии.

**Заключение**

В дипломной работе была разработана и изготовлена автоматизированная установка для испытания материалов на коррозийную стойкость. Представлен краткий обзор элементной базы, используемой для создания установки. Была изготовлена схема управления шаговым двигателем и нагревательным элементом. Произведено программирование и отладка микроконтроллера, разработано программное обеспечение для компьютера на базе операционных систем Windows.

Данное устройство было выполнено по заказу лаборатории физико-технического факультета ГрГу им. Янки Купалы и будет использоваться научными сотрудниками для проведения различных экспериментов и испытаний.

В связи с тем, что контроллер изготовлен на универсальной платформе Arduino, то в дальнейшем его функционал может быть расширен, а программное обеспечение доработано.

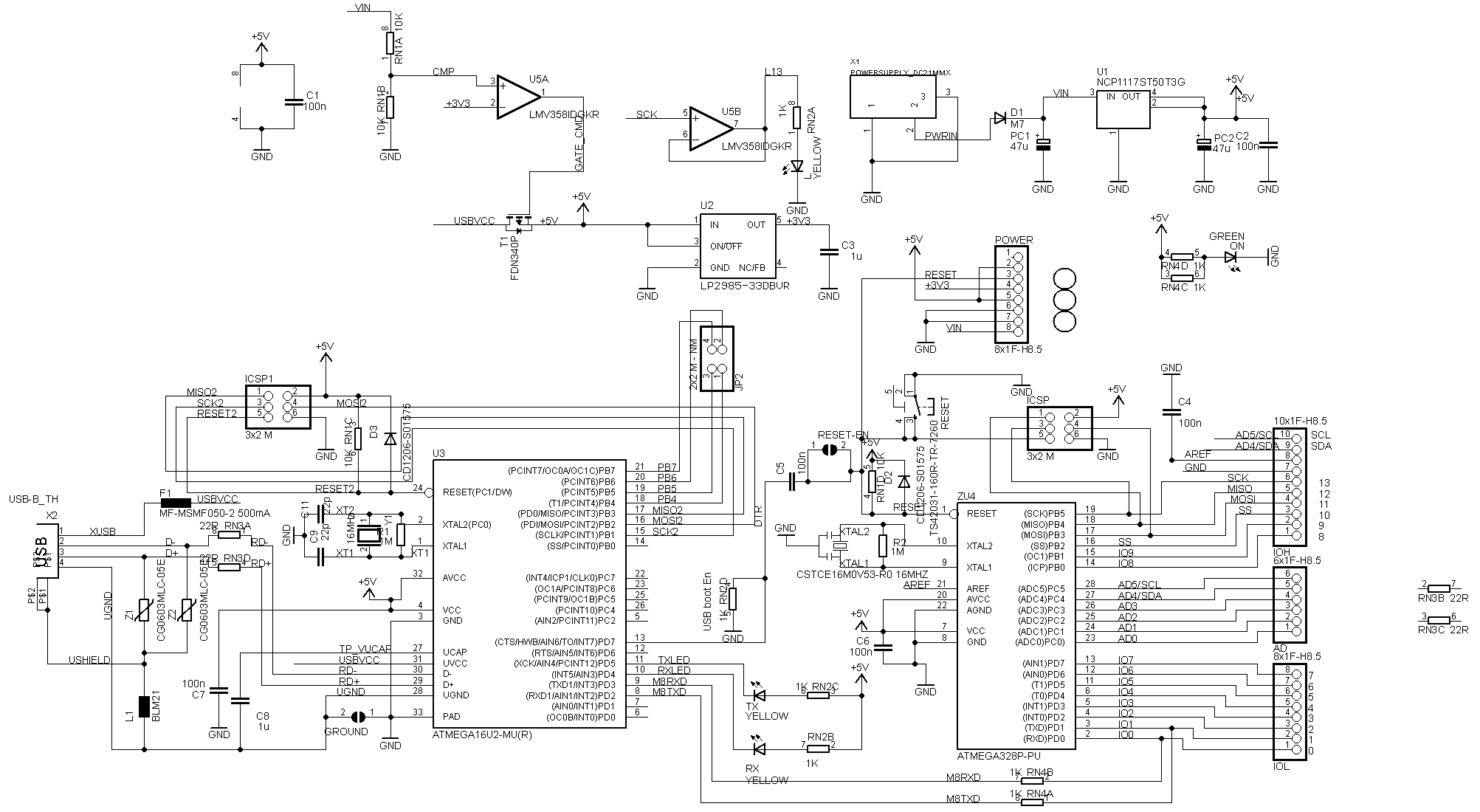
**БИБЛИОГРАФИЧЕСКИЙ СПИСОК**

**Список использованных источников:**

1. Arduino.org [электронный ресурс].-2009. Режим доступа: http://www.arduino.org/ - Дата доступа: 01.04.2016
2. Сайт паяльник [электронный ресурс].-1999. Режим доступа: http://cxem.net/master/45.php- Дата доступа: 02.04.2016
3. Проекты ардуино [электронный ресурс].-2016. Режим доступа: http://arduino-project.net Дата доступа: 05.04.2016
4. Амперка [электронный ресурс].-2010. Режим доступа: http://amperka.ru Дата доступа: 08.04.2016
5. Atmel [электронный ресурс].-2016. Режим доступа: http://www.atmel.com/ru/ru/Дата доступа: 10.04.2016
6. Microsoft [Developer Network](https://msdn.microsoft.com/ru-ru) [электронный ресурс].-2016. Режим доступа: https://msdn.microsoft.com Дата доступа: 12.04.2016
7. Studfiles [электронный ресурс]. -2017. Режим доступа: http://www.studfiles.ru/preview/6144952/page:4/ Дата доступа: 12.04.2016
8. Studfiles [электронный ресурс]. -2017. Режим доступа: <http://www.studfiles.ru/preview/6232218/page:6/> Дата доступа: 12.04.2016
9. Vt-tech [электронный ресурс]. -2017. Режим доступа: https://vt-tech.eu/articles/cnc/50-stepper-motors.html Дата доступа: 12.04.2016
10. Robotosha [электронный ресурс]. -2017. Режим доступа: http://robotosha.ru/electronics/how-stepper-motors-work.htmlДата доступа: 12.04.2016
11. Департамент по энергоэффективности [электронный ресурс].-2007. Режим доступа: http://energoeffekt.gov.by/ - Дата доступа: 21.12.2015

**Приложение А**

**Схема электрическая Arduino uno**

****

**Приложение Б**

**Листинг кода контроллера**

//Для термомтата и датчика Dallas

#include <OneWire.h> // Подключаем библиотеку для взаимодействия с устройствами, работающими на шине и по протоколу 1-Wire

#include <DallasTemperature.h> // Подключаем библиотеку с функциями для работы с DS18B20 (запросы, считывание и преобразование возвращаемых данных)

#define ONE\_WIRE\_BUS 2 // Указываем пин подключения data-вывода датчика температуры

#define term\_power 4 // Указываем пин подключения питания датчика температуры

OneWire oneWire(ONE\_WIRE\_BUS); // Сообщаем библиотеке об устройстве, работающем по протоколу 1-Wire

DallasTemperature sensors(&oneWire); // Связываем функции библиотеки DallasTemperature с нашим 1-Wire устройством (DS18B20)

//Процедуры;

unsigned long int otvet(String str){ //Процедура считывающая данные с порта из string в long int начиная с 3 символа;

String str1;

unsigned long int const1[10];

unsigned long int pes;

str1=str.substring(2);

for (int len=0;len<=str1.length()-1;len++){ //Повторять столько раз, сколько символов в строке;

const1[len+1]=(int(str1[len])-48);

unsigned long int k=1;

pes=0;

for(int l=1;l<=str1.length();l++){

k=k\*10;

}

for(int l=1;l<=str1.length();l++){ //Преобразование

k=k/10;

if(l!=str1.length()) pes=pes+const1[l]\*k;

if(l==str1.length()) pes=pes+const1[l];

}

}

return pes;

}

//Переменные

int a = 9; //Номер пина для обмотки двигателя

int b= 10; //Номер пина для обмотки двигателя

int c= 11; //ДНомер пина для обмотки двигателя

int d= 12; //Номер пина для обмотки двигателя

int ten=8; //Номер выхода управления ключом нагревателя

int buf; // buffer

String str; //buffer stroka

unsigned long int ot[11]; /\*otvet

0- Термостат 1-вкл 0-выкл

1- Вибрация 1-вкл 0-выкл

2- Выдержка в опущенном

3- Выдержка в поднятом

4- Высота подъема

5- Колличество циклов

6- Температура

7- Гистерезис

8- Скорость работы шагового двигателя

9- Интенсивность вибрации

10-Амплитуда вибрации

\*/

int delstart=0; // - задержка перед стартом;

String nomera="0123456789a";//Строка счетчик, для вывода данных

bool opros;

void setup() {

Serial.begin(9600);

sensors.begin(); // Запускаем библиотеку измерения температуры

//Входы обмоток

pinMode(a, OUTPUT);

pinMode(b, OUTPUT);

pinMode(c, OUTPUT);

pinMode(d, OUTPUT);

pinMode(ten,OUTPUT); //Нагреватель

}

void loop() {

str="";

delstart=0;

digitalWrite(ten,LOW);//Отключаем термостат;

Serial.print("t0");

Serial.write("\n");

while (Serial.available() > 0) {

buf=Serial.read();

str=str+char(buf); // Добавление элемента с преобразованием типов данных

}

//Идентифицкация данных;

if(str[0]=='?' and str[1]=='?') opros=true;

if(str[0]=='0' and str[1]=='$') ot[0]=otvet(str);

if(str[0]=='1' and str[1]=='$') ot[1]=otvet(str);

if(str[0]=='2' and str[1]=='$') ot[2]=otvet(str);

if(str[0]=='3' and str[1]=='$') ot[3]=otvet(str);

if(str[0]=='4' and str[1]=='$') ot[4]=otvet(str);

if(str[0]=='5' and str[1]=='$') ot[5]=otvet(str);

if(str[0]=='6' and str[1]=='$') ot[6]=otvet(str);

if(str[0]=='7' and str[1]=='$') ot[7]=otvet(str);

if(str[0]=='8' and str[1]=='$') ot[8]=otvet(str);

if(str[0]=='9' and str[1]=='$') ot[9]=otvet(str);

if(str[0]=='a' and str[1]=='$') ot[10]=otvet(str);

//Вывод данных

if(opros==true){

opros=false;

for(int i=0;i<=10;i++){

Serial.print(nomera[i]);

Serial.print("$");

Serial.print(ot[i]); // вывод символа

Serial.write("\n");//

delay(100);

}}//if for

//Запуск без отложенного старта;

if(str[0]=='!' and str[1]=='$'){

start(0,ot[0],ot[1],ot[2],ot[3],ot[4],ot[5],ot[6],ot[7],ot[8],ot[9],ot[10]); //Включение установки

}

//Запуск с отложенным стартом;

if(str[0]=='!' and str[1]=='d'){

delstart=otvet(str);

start(delstart,ot[0],ot[1],ot[2],ot[3],ot[4],ot[5],ot[6],ot[7],ot[8],ot[9],ot[10]); //Включение установки

}

temp(); // Вывод температуры

off(); //Отключаем обмотки;

delay(100);

}

///Основная Функция, управляет устройством

void start (unsigned long int d,unsigned long int trst,unsigned long int vbr,unsigned long int down,unsigned long int up,int h,unsigned long int col,unsigned long int temp,unsigned long int gist,unsigned long int spd,unsigned long int in,unsigned long int am){

// d-задержка,trst-термостат,vbr-вибрация, down удержание мс,up,h-высота,col-колл-во повторов;

//temp-температура,gist-Гистерезис, spd- скорость, in-интенсивность виб-и, am -амплитуда;

bool stp=false;// Нажата кнопка остановки;

unsigned long int tim=0;

int bu=0; //Буфер чтения

int stopp=0;

up=up\*1000; //Перевод в секунды

down=down\*1000; ////Перевод в секунды

delay(d);// Задержка перед стартом

//С вибрацией

if(vbr==1){

//Главный цикл

for(int i=0;i<=col;i++){ //Колличество повторений;

rotA(spd,h);

tim=millis()+down;

while(tim>millis()){

if(trst==1) term(temp,gist);

if(stopp==1)break;

rotB(in,am);

rotA(in,am);

while (Serial.available() > 0) {

bu=Serial.read();

if(char(bu)=='q'){stopp=1;}

}

}//while

rotB(spd,h);

tim=millis()+up;

while(tim>millis()){

if(stopp==1)break;

if(trst==1) term(temp,gist);

// rotA(in,am); -вибрация отключена в поднятом состоянии.

// rotB(in,am);

while (Serial.available() > 0) {

bu=Serial.read();

if(char(bu)=='q'){stopp=1;}

}

}//while

if(stopp==1)break;

}}//for, if

//Без вибрации

if(vbr==0){

//Главный цикл

for(int j=0;j<=col;j++){ //Колличество повторений;

rotA(spd,h);

off();

tim=millis()+down;

while(tim>millis()){

if(trst>=1)term(temp,gist);

while (Serial.available() > 0) {

bu=Serial.read();

if(char(bu)=='q'){stopp=1;}

}

if(stopp==1)break;

delay(1);

}//while

rotB(spd,h);

off ();

tim=millis()+up;

while(tim>millis()){

if(trst>=1)term(temp,gist);

while (Serial.available() > 0) {

bu=Serial.read();

if(char(bu)=='q'){stopp=1;}

}

if(stopp==1)break;

delay(1);

}//while

if(stopp==1)break;

}}//for, if

}//void

//////////////Отключает обмотки;

void off(){

digitalWrite(a, LOW);

digitalWrite(b, LOW);

digitalWrite(c, LOW);

digitalWrite(d, LOW);

}

//Функция температура

float temp() {

sensors.requestTemperatures(); // Запрос на измерение температуры (повторный)

float t = float(sensors.getTempCByIndex(0)); // Получаем значение температуры

delay(10); // Задержка, чтобы датчик не нагревался от частых измерений

Serial.print("tt");//

Serial.print(t,1);//вывод температуры;

Serial.write("\n");//

return(t);// Возвращаем значение температуры

}

//Функция термостат

void term(int t,int g) {

sensors.requestTemperatures(); // Запрос на измерение температуры (повторный)

int tmp = int(sensors.getTempCByIndex(0)); // Получаем значение температуры

delay(10); // Задержка, чтобы датчик не нагревался от частых измерений

if(tmp<=(t-g)) {digitalWrite(ten, HIGH);Serial.print("t1");Serial.write("\n");}

if(tmp>=(t+g)) {digitalWrite(ten,LOW);Serial.print("t0");Serial.write("\n");}

delay(50);

Serial.print("tt");//

Serial.print(tmp);//вывод температуры;

Serial.write("\n");//

}

//Вращение в сторону А

int rotA (int sp,int n) //Sp-скорость n-колличество повторений

{

for(int i=0;i<=n;i++){

digitalWrite(a, HIGH);

delay(sp);

digitalWrite(d, LOW);

delay(sp);

digitalWrite(b, HIGH);

delay(sp);

digitalWrite(a, LOW);

delay(sp);

digitalWrite(c, HIGH);

delay(sp);

digitalWrite(b, LOW);

delay(sp);

digitalWrite(d, HIGH);

delay(sp);

digitalWrite(c, LOW);

delay(sp);

}

}

//Вращение в сторону Б

int rotB (int sp,int n)

{

for(int i=0;i<=n;i++){

digitalWrite(d, HIGH);

delay(sp);

digitalWrite(a, LOW);

delay(sp);

digitalWrite(c, HIGH);

delay(sp);

digitalWrite(d, LOW);

delay(sp);

digitalWrite(b, HIGH);

delay(sp);

digitalWrite(c, LOW);

delay(sp);

digitalWrite(a, HIGH);

delay(sp);

digitalWrite(b, LOW);

delay(sp);

}

}

**Приложение Б**

**Листинг кода программы**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

using System.IO; // Библиотека для сериал порта;

namespace WindowsFormsApplication2

{

public partial class Form1 : Form

{

public int ts = 0; //Термостат;

public int vb = 0; // Вибрация;

public int delay=0; // Счетчик в таймере1 для задержки передачи данных;

public String str=" "; //Буфер принятых данных;

public int delstart = 0; //Таймер задержки перед стартом.

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

trackBar1.Minimum = 2;

trackBar1.Maximum = 100;

}

private void button1\_Click(object sender, EventArgs e)

{

button1.Enabled = false;

// Выбор порта;

if (comboBox1.Text == "COM 1") { SerialPort1.PortName = "COM1"; }

if (comboBox1.Text == "COM 2") { SerialPort1.PortName = "COM2"; }

if (comboBox1.Text == "COM 3") { SerialPort1.PortName = "COM3"; }

if (comboBox1.Text == "COM 4") { SerialPort1.PortName = "COM4"; }

if (comboBox1.Text == "COM 5") { SerialPort1.PortName = "COM5"; }

if (comboBox1.Text == "COM 6") { SerialPort1.PortName = "COM6"; }

if (comboBox1.Text == "COM 7") { SerialPort1.PortName = "COM7"; }

if (comboBox1.Text == "COM 8") { SerialPort1.PortName = "COM8"; }

if (comboBox1.Text == "COM 9") { SerialPort1.PortName = "COM9"; }

SerialPort1.Open();

}

private void trackBar1\_Scroll(object sender, EventArgs e)

{

label2.Text = trackBar1.Value.ToString();

}

private void button2\_Click(object sender, EventArgs e)

{

if (checkBox3.Checked==true) ts = 1;

if (checkBox3.Checked == false) ts = 0;

if (checkBox1.Checked==true) vb = 1;

if (checkBox1.Checked == false) vb = 0;

timer1.Enabled = true;

MessageBox.Show("Ожидайте 15 секунд");

}

private void SerialPort1\_DataReceived(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)

{

str=(SerialPort1.ReadLine()+" ");

}

private void button6\_Click(object sender, EventArgs e)

{

button1.Enabled = true;

SerialPort1.Close();

}

private void timer1\_Tick(object sender, EventArgs e)

{

progressBar1.Value++;

//Исключение случайного нажатия кнопок

button3.Enabled = false;

button2.Enabled = false;

if (delay == 12) { delay = 0; timer1.Enabled = false; button3.Enabled = true; button2.Enabled = true; progressBar1.Value = 0; }

//Запсиь данных в порт

if (delay == 0) SerialPort1.Write("0$" + ts.ToString());

if (delay == 1) SerialPort1.Write("1$" + vb.ToString());

if (delay == 2) SerialPort1.Write("2$" + textBox1.Text); //down

if (delay == 3) SerialPort1.Write("3$" + textBox2.Text); //UP

if (delay == 4) SerialPort1.Write("4$" + textBox19.Text); //Height

if (delay == 5) SerialPort1.Write("5$" + textBox3.Text); //Koll-vo ciklov

if (delay == 6) SerialPort1.Write("6$" + textBox4.Text); //Temperature

if (delay == 7) SerialPort1.Write("7$" + textBox5.Text); //Gisteresis

if (delay == 8) SerialPort1.Write("8$" + trackBar1.Value.ToString()); // Speed

if (delay == 9) SerialPort1.Write("9$" + textBox6.Text); //Intensivnost

if (delay == 10) SerialPort1.Write("a$" + textBox7.Text);//Ampituda

delay++;

}

private void timer2\_Tick(object sender, EventArgs e)

{

//Получение данных с порта и вывод на экран;

if (str[0] == 't' & str[1] == 't') label15.Text = (str.Substring(2) + " C"); //Вывод температуры

if (str[0] == 't' & str[1] == '0') label18.Text = "Выключен"; //термостат выкл

if (str[0] == 't' & str[1] == '1') label18.Text = "Включен"; //термостат вкл

if (str[0] == 'n' & str[1] == '$') label17.Text = str.Substring(2); //Вывод колличества циклов

if (str[0] == '0' & str[1] == '$' & str[2] == '1') checkBox4.Checked = true;

if (str[0] == '0' & str[1] == '$' & str[2] == '0') checkBox4.Checked = false;

if (str[0] == '1' & str[1] == '$' & str[2] == '1') checkBox2.Checked = true;

if (str[0] == '1' & str[1] == '$' & str[2] == '0') checkBox2.Checked = false;

if (str[0] == '2' & str[1] == '$') textBox14.Text = str.Substring(2);

if (str[0] == '3' & str[1] == '$') textBox13.Text = str.Substring(2);

if (str[0] == '4' & str[1] == '$') textBox20.Text = str.Substring(2);

if (str[0] == '5' & str[1] == '$') textBox12.Text = str.Substring(2);

if (str[0] == '6' & str[1] == '$') textBox11.Text = str.Substring(2);

if (str[0] == '7' & str[1] == '$') textBox10.Text = str.Substring(2);

if (str[0] == '8' & str[1] == '$') textBox15.Text = str.Substring(2);

if (str[0] == '9' & str[1] == '$') textBox9.Text = str.Substring(2);

if (str[0] == 'a' & str[1] == '$') textBox8.Text = str.Substring(2);

}

private void button3\_Click(object sender, EventArgs e)

{

SerialPort1.Write("??");

}

private void button4\_Click(object sender, EventArgs e)

{

SerialPort1.Write("!$");

}

private void button7\_Click(object sender, EventArgs e)

{

SerialPort1.Write("q");

}

private void button5\_Click(object sender, EventArgs e)

{

delstart = Int32.Parse(textBox16.Text) + Int32.Parse(textBox18.Text) \* 60 + Int32.Parse(textBox17.Text) \* 60 \* 60;

delstart = delstart \* 1000;

SerialPort1.Write("!d" + delstart.ToString());

}}}