МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«**КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Институт среднего профессионального образования**

**(ИНСПО)**

|  |  |
| --- | --- |
|  | ДОПУСТИТЬ К ЗАЩИТЕ В ГЭК  Директор ИНСПО  канд. пед. наук, доцент  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.П. Хлопова  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019г. |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**РАЗРАБОТКА АППАРАТНОЙ ЧАСТИ СИСТЕМЫ МОНИТОРИНГА МЕДИЦИНСКИХ ПОКАЗАТЕЛЕЙ**

Работу выполнил\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н. А. Сиркин

(подпись, дата)

Специальность 09.02.03 Программирование в компьютерных системах

Руководитель

Преподаватель ИНСПО \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.Г. Титов

(подпись, дата)

Нормоконтролер

Преподаватель ИНСПО \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Э. С. Егозаров

(подпись, дата)

Краснодар

2019

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc11088418)

[1 Микроконтроллеры семейства Advanced Virtual RISC. 6](#_Toc11088419)

[1.1 Существующие отладочные платы 6](#_Toc11088420)

[1.1 Архитектура микроконтроллеров AVR 8](#_Toc11088421)

[1.2.1 Память. Типы используемой памяти 9](#_Toc11088422)

[1.2.2 Периферийные модули 11](#_Toc11088423)

[1.2.3 Стандарты питания 15](#_Toc11088424)

[1.2.4 Схема подключения 16](#_Toc11088425)

[1.2.5 Система команд 17](#_Toc11088426)

[1.2.6 Классификация по напряжению и току потребления 18](#_Toc11088427)

[1.2.7 Классификация по типу корпуса 18](#_Toc11088428)

[1.2.8 Cовременные микроконтроллеры 20](#_Toc11088429)

[1.3 Программаторы и их типы 21](#_Toc11088430)

[1.3.1 AVR STK500 V2.0 22](#_Toc11088431)

[1.3.2 USB ISP и USB ASP 22](#_Toc11088432)

[1.3.3 COM программатор 23](#_Toc11088433)

[1.3.4 Параллельный программатор 24](#_Toc11088434)

[1.3.5 Arduino ISP 24](#_Toc11088435)

[1.3.6 USB – UART 25](#_Toc11088436)

[1.4 Интегрированные среды разработки программ для AVR 26](#_Toc11088437)

[1.4.1 Arduino IDE 26](#_Toc11088438)

[1.4.2 ATMEL Studio 28](#_Toc11088439)

[1.4.3 WinAVR 30](#_Toc11088440)

[1.5 Беспроводной интерфейс Arduino. 31](#_Toc11088441)

[1.5.1 Микроконтроллер. 31](#_Toc11088442)

[1.5.2 Распиновка ESP8266 32](#_Toc11088443)

[1.5.3 Программный комплект разработчика 32](#_Toc11088444)

[1.5.4 Функциональность ESP8266 33](#_Toc11088445)

[1.6. NodeMсu ESP9266 33](#_Toc11088446)

[1.6.1 Распиновка NodeMcu v3 35](#_Toc11088447)

[2 Разработка аппаратной части системы мониторинга медицинских показателей 36](#_Toc11088448)

[2.1 Основной модуль 36](#_Toc11088449)

[2.1.1 Функциональные возможности 37](#_Toc11088450)

[2.1.2 Сборка устройства и индикация 37](#_Toc11088451)

[2.1.3 Сервисное обслуживание устройства 37](#_Toc11088452)

[2.1.4 Строение устройства 38](#_Toc11088453)

[2.1.5 Принципиальная схема 39](#_Toc11088454)

[2.1.6 Алгоритм работы устройства 40](#_Toc11088455)

[2.1.7 Тестирование устройства 40](#_Toc11088456)

[ЗАКЛЮЧЕНИЕ 42](#_Toc11088457)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 44](#_Toc11088458)

[ПРИЛОЖЕНИЕ А – Техическое задание 46](#_Toc11088459)

[ПРИЛОЖЕНИЕ Б – Листинг 48](#_Toc11088460)

[ПРИЛОЖЕНИЕ В – Акт использования программного продукта 50](#_Toc11088461)

**ВВЕДЕНИЕ**

В последнее время, стараясь улучшить качество и результативность лечения пациентов, требующих постоянного наблюдения, медицинские учреждения нуждаются в более технологичных решениях, например, удаленный мониторинг медицинских показателей всех пациентов непосредственно с компьютера или смартфона, имеющий доступ в сеть.

Интегрированные решения такого типа с системами сигнализации и мгновенным обменом данными позволяют медработникам вовремя узнавать, когда пациент находится в стабильном или критическом состоянии. Удаленный мониторинг упрощает лечение хронических заболеваний путем контроля таких критических риск–факторов, как частота пульса, температура тела и т.п.

Объектом исследования является дистанционный мониторинг медицинских показателей.

Предметом исследования является устройство, позволяющее осуществлять дистанционный мониторинг медицинских показателей, с помощью интегрированных датчиков.

Целью дипломной работы является разработка аппаратной и программных частей автоматизированной системы мониторинга медицинских показателей с выводом данных в глобальную сеть, посредством web – интерфейса.

Для достижения цели выпускной квалификационной работы поставлены следующие задачи:

* + Рассмотреть основные принципы работы микроконтроллеров архитектуры Advanced Virtual RISC.
  + Описать используемые технологии для достижения заданной цели.
  + Разработать платформу для мониторинга медицинских показателей.

Структура работы обусловлена предметом, целью и задачами исследования. Работа состоит из введения, двух глав, заключения, списка использованных источников и 3 приложений.

Введение раскрывает актуальность, определяет степень научной разработки темы, объект, предмет, цель, задачи.

Первая глава содержит в себе подробную информацию о микроконтроллерах архитектуры AVR и серии отладочных плат Arduino, программаторах, а также описывает популярные среды разработки микропрограмм.

Вторая глава представляет собой 2 раздела и описывает возможности разработанного устройства «Biomark», используемые в проекте компоненты, сборку и отладку.

Заключение включает итоги работы, а также сложности, возникшие в процессе разработки.

В приложениях находятся техническое задание и листинг разработанной микропрограммы.

# 1 Микроконтроллеры семейства Advanced Virtual RISC.

Идея разработки нового RISC–ядра принадлежит двум студентам Norwegian University of Science and Technology из норвежского города Тронхейма – Альфу Богену и Вегарду Воллену. В 1995 году Боген и Воллен решили предложить американской корпорации Atmel, которая была известна своими чипами с flash – памятью, выпускать новый 8–битный RISC–микроконтроллер и снабдить его Flash–памятью для программ на одном кристалле с вычислительным ядром.

Идея была одобрена Atmel Corp., и было принято решение незамедлительно инвестировать в данную разработку. Во второй половине 1997 г. корпорация Atmel приступила к серийному производству нового семейства микроконтроллеров, а также их рекламной и технической поддержкой. Новое ядро было запатентовано и получило название AVR.

На сегодняшний день данная архитектура является самой ходовой. На ней базируется широкий диапазон устройств, начиная от любительских проектов, заканчивая объектами автоматизации на крупных заводах.

## **1.1 Существующие отладочные платы**

Для удобства разработки и отладки прототипов устройств используются отладочные платы, позволяющие быстро собрать ту или иную схему без пайки за короткий промежуток времени. Некоторые оснащаются программным обеспечением.

Arduino – торговая марка аппаратно–программных средств для построения простых систем автоматики и робототехники. Программная часть состоит из бесплатной программной оболочки (IDE) для написания микропрограмм, их компиляции и прошивки отладочной платы. Аппаратная часть представляет собой набор собранных печатных плат, продающихся как официальным производителем, так и сторонними разработчиками с небольшими модификациями. Полностью открытая архитектура системы позволяет свободно копировать или дополнять линейку продукции. Самая популярная плата – Arduino UNO, она представлена на рисунке 1.

Arduino – представляет собой плату с микроконтроллером и процессором ATmega328p. Данный контроллер обладает архитектурой AVR. Данная архитектура обладает рядом особенностей, из которых стоит отметить относительно малую производительность и высокую вибро – температурную устойчивость. 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.

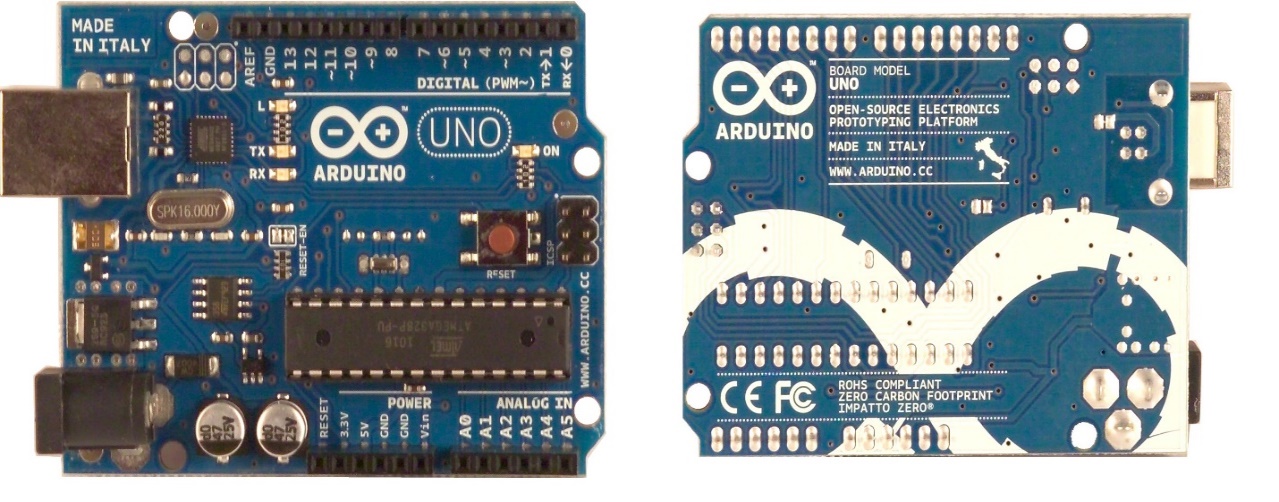


Рисунок 1 – Плата Arduino UNO

Питание. Arduino UNO может получать питание через подключение USB или от внешнего источника питания. Источник питания выбирается автоматически. 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. 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. Основная необходимая часть контроллера для работы с гироскопами необходимы контакты с возможностью работы с шиной I2C. На плате отдельно выведены контакты SDA и SCL как раз для шины I2C. На данной плате это выводы A4 и A5. В других версиях платы данные выводы могут быть выведены на другие пины.

## **Архитектура микроконтроллеров AVR**

 Основой микроконтроллеров AVR является 8–битное микропроцессорное ядро или центральное процессорное устройство (ЦПУ), построенное на принципах RISC–архитектуры. Основой этого блока служит арифметико–логическое устройство (АЛУ). По системному тактовому сигналу из памяти программ в соответствии с содержимым счетчика команд (Program Counter – PC) выбирается очередная команда и выполняется АЛУ. Во время выбора команды из памяти программ происходит выполнение предыдущей выбранной команды, что и позволяет достичь быстродействия 1 MIPS на 1 МГц. Архитектура микроконтроллеров AVR представлена на рисунке 2.

АЛУ подключено к регистрам общего назначения РОН (General Purpose Registers – GPR). Регистров общего назначения всего 32, они имеют байтовый формат, то есть каждый из них состоит из восьми бит. РОН находятся в начале адресного пространства оперативной памяти, но физически не являются ее частью. Поэтому к ним можно обращаться двумя способами (как к регистрам и как к памяти). Такое решение является особенностью AVR и повышает эффективность работы и производительность микроконтроллера.

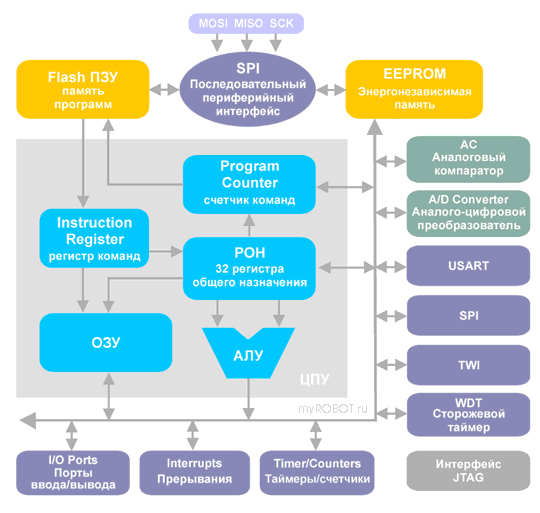


Рисунок 2 – Архитектура микроконтроллера AVR

Отличие между регистрами и оперативной памятью состоит в том, что с регистрами можно производить любые операции (арифметические, логические, битовые), а в оперативную память можно лишь записывать данные из регистров.

**1.2.1 Память. Типы используемой памяти**

 В микроконтроллерах AVR реализована Гарвардская архитектура, в соответствии с которой разделены не только адресные пространства памяти программ и памяти данных, но и шины доступа к ним. Каждая из областей памяти данных также расположена в своем адресном пространстве.

Память программ (FLASH ROM или FLASH ПЗУ) предназначена для хранения последовательности команд, управляющих функционированием микроконтроллера, и имеет 16–ти битную организацию. Все AVR имеют Flash–память программ, которая может быть различного размера – от 1 до 256 КБайт. Ее главное достоинство в том, что она построена на принципе электрической перепрограммируемости, т. е. допускает многократное стирание и запись информации. Программа заносится во Flash–память AVR как с помощью обычного программатора, так и с помощью SPI–интерфейса, в том числе непосредственно на собранной плате. Возможностью внутрисхемного программирования (функция ISP) через коммуникационный интерфейс SPI обладают все микроконтроллеры AVR, кроме Tiny11 и Tiny28.   
 Память данных разделена на три части: регистровая память, оперативная память (ОЗУ – оперативное запоминающее устройство или RAM) и энергонезависимая память (ЭСППЗУ или EEPROM).

Регистровая память включает 32 регистра общего назначения (РОН или GPR), объединенных в файл, и служебные регистры ввода/вывода (РВВ). И те, и другие расположены в адресном пространстве ОЗУ, но не являются его частью. В области регистров ввода/вывода расположены различные служебные регистры (регистры управления микроконтроллером, регистры состояния и т. п.), а также регистры управления периферийными устройствами, входящими в состав микроконтроллера. По сути, управление микроконтроллером заключается в управлении этими регистрами.

Энергонезависимая память данных. Для долговременного хранения различной информации, которая может изменяться в процессе функционирования микроконтроллерной системы, используется EEPROM–память. Все AVR имеют блок энергонезависимой электрически перезаписываемой памяти данных EEPROM от 64 Байт до 4 КБайт. Этот тип памяти, доступный программе микроконтроллера непосредственно в ходе ее выполнения, удобен для хранения промежуточных данных, различных констант, коэффициентов, серийных номеров, ключей и т.п. EEPROM может быть загружена извне как через SPI интерфейс, так и с помощью обычного программатора. Число циклов стирание/запись – не менее 100 тыс.

Внутренняя оперативная статическая память Static RAM (SRAM) имеет байтовый формат и используется для оперативного хранения данных. Размер оперативной памяти может варьироваться у различных чипов от 64 Байт до 4 КБайт. Число циклов чтения и записи в RAM не ограничено, но при отключении питающего напряжения вся информация теряется.

Для некоторых микроконтроллеров возможна организация подключения внешнего ОЗУ объемом до 64 килобайт.

**1.2.2 Периферийные модули**

Периферия микроконтроллеров AVR включает: порты, поддержку внешних прерываний, таймеры–счетчики, сторожевой таймер, аналоговые компараторы, 10–разрядный 8–канальный АЦП, интерфейсы UART, JTAG и SPI, устройство сброса по понижению питания, широтно–импульсные модуляторы.

Порты ввода/вывода AVR имеют число независимых линий "вход/выход" от 3 до 53. Каждая линия порта может быть запрограммирована на вход или на выход. Мощные выходные драйверы обеспечивают токовую нагрузочную способность 20 мА на линию порта (втекающий ток) при максимальном значении 40 мА, что позволяет, например, непосредственно подключать к микроконтроллеру светодиоды и биполярные транзисторы. Общая токовая нагрузка на все линии одного порта не должна превышать 80 мА (все значения приведены для напряжения питания 5 В).

Архитектурная особенность построения портов ввода/вывода у AVR заключается в том, что для каждого физического вывода (пина) существует 3 бита контроля/управления, а не 2, как у распространенных 8–разрядных микроконтроллеров (Intel, Microchip, Motorola и т.д.). Это позволяет избежать необходимости иметь копию содержимого порта в памяти для безопасности и повышает скорость работы микроконтроллера при работе с внешними устройствами, особенно в условиях внешних электрических помех.

Система прерываний – одна из важнейших частей микроконтроллера. Все микроконтроллеры AVR имеют многоуровневую систему прерываний. Прерывание прекращает нормальный ход программы для выполнения приоритетной задачи, определяемой внутренним или внешним событием.

Для каждого такого события разрабатывается отдельная программа, которую называют подпрограммой обработки запроса на прерывание (для краткости – подпрограммой прерывания), и размещается в памяти программ.

При возникновении события, вызывающего прерывание, микроконтроллер сохраняет содержимое счетчика команд, прерывает выполнение центральным процессором текущей программы и переходит к выполнению подпрограммы обработки прерывания.

После выполнения подпрограммы прерывания осуществляется восстановление предварительно сохраненного счетчика команд и процессор возвращается к выполнению прерванной программы. Для каждого события может быть установлен приоритет. Понятие приоритет означает, что выполняемая подпрограмма прерывания может быть прервана другим событием только при условии, что оно имеет более высокий приоритет, чем текущее. В противном случае центральный процессор перейдет к обработке нового события только после окончания обработки предыдущего.

Таймеры/счетчики имеются в AVR с разрядностью 8 или 16 бит, могут работать и как таймеры от внутреннего источника тактовой частоты, и как счетчики внешних событий. Их можно использовать для точного формирования временных интервалов, подсчета импульсов на выводах микроконтроллера, формирования последовательности импульсов, тактирования приемопередатчика последовательного канала связи. В режиме ШИМ (PWM) таймер/счетчик может представлять собой широтно–импульсный модулятор и используется для генерирования сигнала с программируемыми частотой и скважностью. Таймеры/счетчики способны вырабатывать запросы прерываний, переключая процессор на их обслуживание по событиям и освобождая его от необходимости периодического опроса состояния таймеров. Поскольку основное применение микроконтроллеры находят в системах реального времени, таймеры/счетчики являются одним из наиболее важных элементов.

Сторожевой таймер (WatchDog Timer) предназначен для предотвращения катастрофических последствий от случайных сбоев программы. Он имеет свой собственный RC–генератор, работающий на частоте 1 МГц. Как и для основного внутреннего RC–генератора, значение 1 МГц является приближенным и зависит прежде всего от величины напряжения питания микроконтроллера и от температуры.

Идея использования сторожевого таймера предельно проста и состоит в регулярном его сбрасывании под управлением программы или внешнего воздействия до того, как закончится его выдеpжка вpемени и не пpоизойдет сбpос пpоцессоpа. Если пpогpамма pаботает ноpмально, то команда сбpоса стоpожевого таймеpа должна pегуляpно выполняться, пpедохpаняя поцессоp от сбpоса. Если же микpоконтроллер случайно вышел за пpеделы пpогpаммы (напpимеp, от сильной помехи по цепи питания) либо зациклился на каком–либо участке пpогpаммы, команда сбpоса стоpожевого таймеpа скоpее всего не будет выполнена в течение достаточного вpемени и пpоизойдет полный сбpос пpоцессоpа, инициализиpующий все pегистpы и пpиводящий систему в pабочее состояние.

Аналоговый компаратор (Analog Comparator) сравнивает напряжения на двух выводах (пинах) микроконтроллера. Результатом сравнения будет логическое значение, которое может быть прочитано из программы.

Выход аналогового компаратора можно включить на прерывание от аналогового компаратора. Можно установить срабатывание прерывания по нарастающему или спадающему фронту или по переключению.

Аналого–цифровой преобразователь (АЦП) служит для получения числового значения напряжения, поданного на его вход. Этот результат сохраняется в регистре данных АЦП. Какой из выводов (пинов) микроконтроллера будет являться входом АЦП, определяется числом, занесенным в соответствующий регистр.

Универсальный синхронный/асинхронный приемопередачик (Universal Synchronous/Asynchronous Receiver and Transmitter – UART или USART) – удобный и простой последовательный интерфейс для организации информационного канала обмена микроконтроллера с внешним миром. Способен работать в дуплексном режиме (одновременная передача и прием данных). Он поддерживает протокол стандарта RS–232, что обеспечивает возможность организации связи с персональным компьютером.

Последовательный периферийный трехпроводный интерфейс SPI (Serial Peripheral Interface) предназначен для организации обмена данными между двумя устройствами. С его помощью может осуществляться обмен данными между микроконтроллером и различными устройствами, такими, как цифровые потенциометры, ЦАП/АЦП, FLASH–ПЗУ и др. С помощью этого интерфейса удобно производить обмен данными между несколькими микроконтроллерами AVR. Кроме того, через интерфейс SPI может осуществляться программирование микроконтроллера.

Интерфейс JTAG – был разработан группой ведущих специалистов по проблемам тестирования электронных компонентов (Joint Test Action Group) Четырехпроводной интерфейс JTAG используется для тестирования печатных плат, внутрисхемной отладки, программирования микроконтроллеров.

Многие микроконтроллеры семейства Mega имеют совместимый интерфейс JTAG для встроенной отладки. Кроме того, все микроконтроллеры Mega с флэш–памятью емкостью 16 кбайт и более могут программироваться через интерфейс JTAG.

Тактовый генератор вырабатывает импульсы для синхронизации работы всех узлов микроконтроллера. Внутренний тактовый генератор AVR может запускаться от нескольких источников опорной частоты (внешний генератор, внешний кварцевый резонатор, внутренняя или внешняя RC–цепочка). Минимальная допустимая частота ничем не ограничена (вплоть до пошагового режима). Максимальная рабочая частота определяется конкретным типом микроконтроллера и указывается Atmel в его характеристиках, хотя практически любой AVR–микроконтроллер с заявленной рабочей частотой, например, в 10 МГц при комнатной температуре легко может быть ускорен до 12 МГц и выше.

Система реального времени (RTC) реализована во всех микроконтроллерах серии MEGA. Таймер/счетчик RTC имеет отдельный определитель, который может быть программным способом подключен или к источнику основной тактовой частоты, или к дополнительному асинхронному источнику опорной частоты (кварцевый резонатор или внешний синхросигнал). Для этой цели зарезервированы два вывода микросхемы. Внутренний осциллятор оптимизирован для работы с внешним "часовым" кварцевым резонатором 32,768 кГц.

**1.2.3 Стандарты питания**

AVR функционируют при напряжениях питания от 1,8 до 6,0 Вольт. Ток потребления в активном режиме зависит от величины напряжения питания и частоты, на которой работает микроконтроллер, и составляет менее 1 мА для 500 кГц, 5...6 мА для 5 МГц и 8...9 мА для частоты 12 МГц.  
AVR могут быть переведены программным путем в один из трех режимов пониженного энергопотребления.

Режим холостого хода (IDLE).Прекращает работу только процессор и фиксируется содержимое памяти данных, а внутренний генератор синхросигналов, таймеры, система прерываний и сторожевой таймер продолжают функционировать. Ток потребления не превышает 2,5 мА на частоте 12 МГц.

Стоповый режим (POWER DOWN). Сохраняется содержимое регистрового файла, но останавливается внутренний генератор синхросигналов, и, следовательно, останавливаются все функции, пока не поступит сигнал внешнего прерывания или аппаратного сброса. При включенном сторожевом таймере ток потребления в этом режиме составляет около 80 мкА, а при выключенном – менее 1 мкА. (Все приведенные значения справедливы для напряжения питания 5 В).

Экономичный режим (POWER SAVE). Продолжает работать только генератор таймера, что обеспечивает сохранность временной базы. Все остальные функции отключены.

Сброс при снижении напряжения питания (BOD). Схема BOD (Brown–Out Detection) отслеживает напряжение источника питания. Если схема включена, то при снижении питания ниже некоторого значения она переводит микроконтроллер в состояние сброса. Когда напряжение питания вновь увеличится до порогового значения, запускается таймер задержки сброса. После формирования задержки внутренний сигнал сброса снимается и происходит запуск микроконтроллера.

**1.2.4 Схема подключения**

Для обеспечения оптимальной работы микроконтроллера необходимо подать напряжение на ядро и узел аналогово–цифрового преобразователя, а также защитить его от помех.

Для питания ядра на Vcc подается плюс питания, GND подключается к минусу. Для питания узла АЦП на AVcc подается плюс питания, причем оптимально, если напряжение стабилизировано и поступает с другого источника, GND подключается к минусу обоих источников.

Вход AREF это вход опорного напряжения АЦП. Туда подается напряжение относительно которого будет считать АЦП. Обычно используется либо внутренний источник опорного напряжения на 2.56 вольта, либо напряжение на AVCC, поэтому на AREF рекомендуется подключать конденсатор, что улучшит качество опорного напряжения АЦП. Индуктивность между Vcc и AVCC фильтрует высокочастотные помехи. Перед контроллером устанавливают конденсаторы во избежание падения напряжения питания в коротком промежутке времени. Вывод RST является входом для кнопки сброса контроллера. Он подтягивается к 5 вольт с помощью резистора, чтобы избежать помех и наводок, а вследствие перезагрузки всего микроконтроллера. В случае необходимости обеспечить большую стабильность частоты к контроллеру подключается внешний кварцевый резонатор. Для прошивки контроллера необходим программатор (см. Рисунок 3).

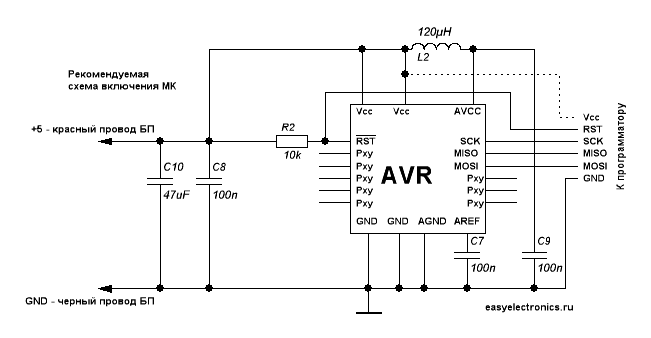


Рисунок 3 – Схема подключения микроконтроллера

**1.2.5 Система команд**

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 и команды сдвига;
* команды операции с битами;
* команды пересылки данных;
* команды передачи управления;
* команды управления системой.

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

**1.2.6 Классификация по напряжению и току потребления**

AT (mega/tiny)xxx – базовая версия.

ATxxxL – версии контроллеров, работающих на пониженном (Low) напряжении питания (2,7В).

ATxxxV – версии контроллеров, работающих на низком напряжении пи–тания (1,8 В).

ATxxxP – малопотребляющие версии (до 100 нА в режиме Power–down)

ATxxxA – уменьшен ток потребления, перекрывается весь диапазон так–товых частот и напряжений питания двух предыдущих версий (также, в некоторых моделях, добавлены новые возможности и новые регистры, но сохранена полная совместимость с предыдущими версиями). Микро–контроллеры «А» и «не–А» обычно имеют одинаковую сигнатуру, что вы–зывает некоторые трудности, так как Fuse–bit отличаются. FUSE bit уста–навливают частоту микроконтроллера, необходимость отключения при понижении напряжения питания и другие свойства.

**1.2.7 Классификация по типу корпуса**

Самым популярным у разработчиков начального уровня является корпус DIP, так как он легко поддается пайке. Например, ATmega328 (см. Рисунок 4) имеет 28 контактный корпус. Именно микроконтроллер в DIP корпусе используется в ARDUINO UNO. Выводы ATmega328 в корпусе DIP и разводку Arduino UNO можно увидеть на рисунке 5.

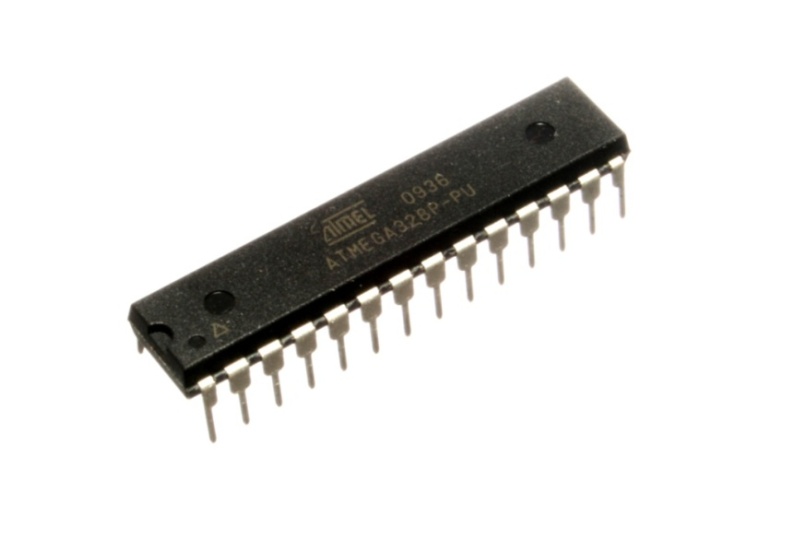


Рисунок 4 – ATmega328 в корпусе DIP

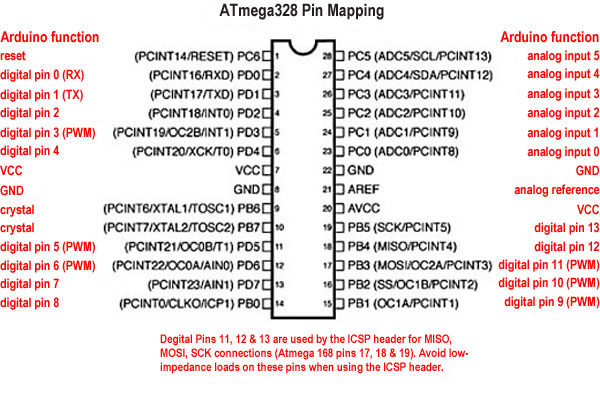


Рисунок 5 – Выводы ATmega328 в корпусе DIP и разводка Arduino UNO

Также распространенными являются корпуса TQFP (см Рисунок 6). Данный корпус является компактным относительно DIP и используется в готовых устройствах, выпущенных на заводах с относительно точным оборудованием. Используются медные выводы с шагом 0.4, 0.5, 0.65, 0.8 и 1 миллиметр. TQFP позволяет решить такие задачи, как увеличение плотности размещения компонентов на печатных платах, уменьшение размеров подложки, уменьшение толщины корпусов устройств.

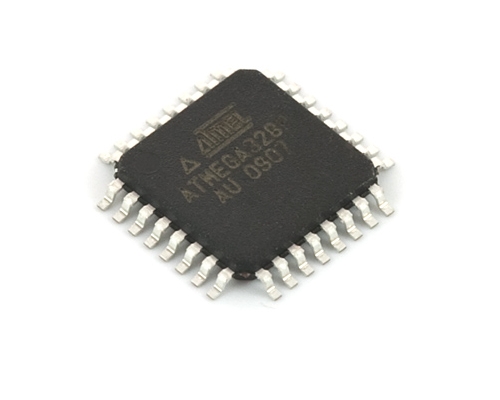


Рисунок 6 – ATmega328 в корпусе TQFP

При изготовлении микроконтроллеров AVR также используются также другие корпуса, но особого распространения они не получили.

**1.2.8 Cовременные микроконтроллеры**

В данный момент актуальны микроконтроллеры AVR серий ATtiny, ATmega, ATxmega.

ATtiny являются урезанными и характеризуются меньшими размером корпуса, малым объемом памяти, количеством портов, низкой частотой и зачастую отсутствием шины I2C. Считаются малопроизводительными.

ATXmega – топовая серия. Используется в высокопроизводительных устройствах. Встречается редко, так как по цене проигрывают более производительным микроконтроллерам других архитектур.

ATMega – актуальная серия, которая включает в себя множество моделей, используемых в проектах, где необходима вычислительная мощность.

Самые ходовые из AVR на сегодняшний день ATmega8, ATmega16, Atmega328.

ATmega8 – старый, но дешевый микроконтроллер, обладающий хорошим соотношением цена/производительность.

ATmega16 – обладает 40 выводами. Более ничем не примечателен.

ATmega328 – используется как микроконтроллер в популярной плате для прототипирования Arduino UNO.

## **1.3 Программаторы и их типы**

Программатор – устройство, служащее посредником между компьютером и микроконтроллером. В задачи, выполняемые программатором, входят перенос микропрограммы в память контроллера и двусторонний обмен данными с ним. Коннекторы для внутрисхемного программирования AVR представлены на рисунке 7.

В случае использования микроконтроллера без загрузчика (микропрограммы, работающей поверх прошивки), программирование осуществляется через последовательный интерфейс SPI. Для подключения программатора к устройству используются три линии интерфейса: SCK (тактовый сигнал), MOSI (вход данных) и MISO (выход данных), а также линия сброса – RESET, плюс и минус питания (VCC и GND). Программирование осуществляется путем посылки 4–байтных команд на вывод MOSI микроконтроллера. Результат выполнения команд чтения снимается с вывода MISO. Стандартно программаторы используют 6 и 10 контактные коннекторы.

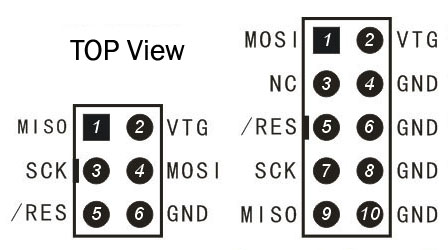


Рисунок 7 – Коннекторы для внутрисхемного программирования AVR

В случае наличия загрузчика связь с программатором идет посредством выводов RX и TX, они же передающий и принимающий выводы последовательной шины SERIAL. Также используется вывод RESET. Загрузчик является программой, работающей поверх прошивки.

**1.3.1 AVR STK500 V2.0**

Поддерживается средой разработки Atmel Studio. Производительный программатор. Поддерживает множество популярных микроконтроллеров AVR. Надежен. К недостаткам относится стоимость (см. Рисунок 8).

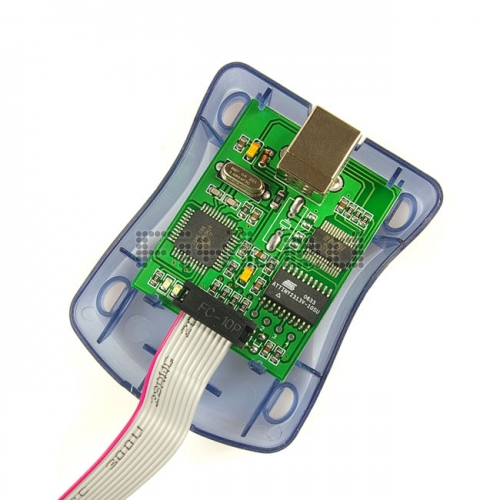


Рисунок 8 – AVR STK500 V2.0

**1.3.2 USB ISP и USB ASP**

Представленные AVR программаторы подключается к ПК с помощью USB интерфейса. USB ISP является версией USB ASP с улучшенной прошивкой. В роли управляющего устройства в них установлен микроконтроллер архитектуры AVR. Являются самыми популярными и дешевыми программаторами (см Рисунок 9).

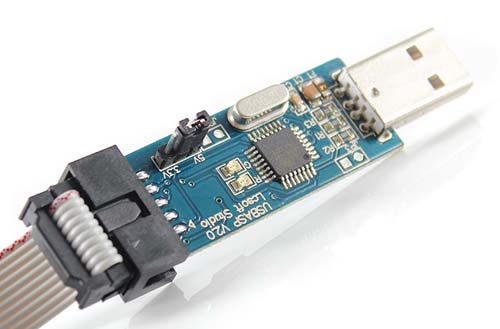


Рисунок 9 – USB ASP

**1.3.3 COM программатор**

Надежный, проверенный временем программатор. Подключается к COM порту компьютера. Устарел. Является сборкой диодов, резисторов и стабилитронов (см. Рисунок 10).

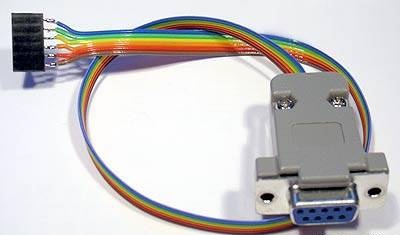


Рисунок 10 – COM программатор

**1.3.4 Параллельный программатор**

Один из первых программаторов.

Состоит только из резисторов. Крайне ненадежен, так как велик риск вывести parallel port из строя (см. рисунок 11).

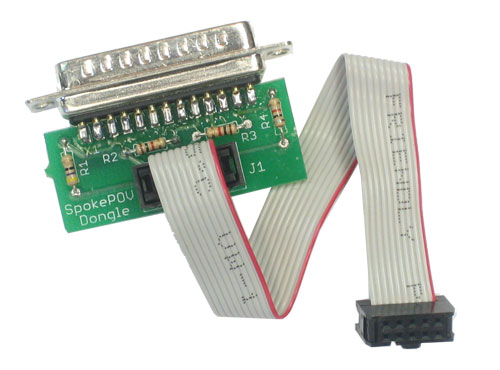


Рисунок 11 – Параллельный программатор

**1.3.5 Arduino ISP**

В данном случае в роли программатора выступает плата Arduino любой версии. В микроконтроллер на плате прошивается скетч из примеров Arduino IDE, преобразуя его в ISP программатор. Далее процесс прошивки отдельностоящего микроконтроллера ничем не отличается от прошивки с помощью обычного ISP программатора. К недостаткам относится неудобство постоянной прошивки Arduino и построения цепи проводов (см. Рисунок 12).

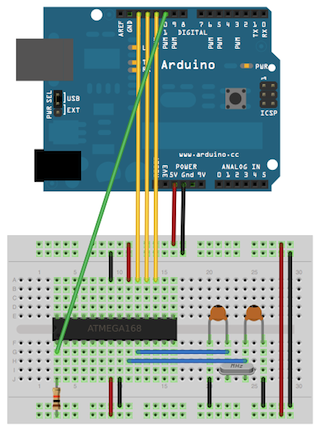


Рисунок 12 – Arduino ISP

**1.3.6 USB – UART**

Не является программатором в классическом понимании.  Протокол UART (**Universal asynchronous receiver/transmitter**) — старейший и самый распространенный на сегодняшний день физический протокол передачи данных. Хотя существует большое количество разных протоколов семейства UART, наиболее известен протокол **RS–232** (в народе – **COM**–порт) (см. Рисунок 13).



Рисунок 13 – USB – UART

## **1.4 Интегрированные среды разработки программ для AVR**

AVR Studio 4 – новая профессиональная интегрированная среда разработки (Integrated Development Environment – IDE), предназначенная для написания и отладки прикладных программ для AVR микропроцессоров в среде Windows 9x/NT/2000. AVR Studio 4 содержит ассемблер и симулятор. Также IDE поддерживает такие средства разработки для AVR как: ICE50, ICE40, JTAGICE, ICE200, STK500/501/502 и AVRISP. В ближайшие месяцы будет расширен список поддерживаемых AVR Studio 4 микроконтроллеров и средств разработки. Обо всех обновлениях можно будет узнать на интернет сайте.

Среды разработки программ для микроконтроллеров AVR позволяют разрабатывать микропрограммы на языках Assembler, C++, Processing/Wiring.

**1.4.1 Arduino IDE**

Среда разработки Arduino состоит из встроенного текстового редактора программного кода, области сообщений, окна вывода текста(консоли), панели инструментов с кнопками часто используемых команд и нескольких меню. Для загрузки программ и связи среда разработки подключается к аппаратной части Arduino, либо совместимому программатору. Программирование ведется на языке Processing/Wiring, представляющем собой набор библиотек и функций.

Программа, написанная в среде Arduino, называется скетч (см Рисунок 14). Ниже представлен листинг кода моргания светодиодом. Скетч пишется в текстовом редакторе, имеющем инструменты вырезки/вставки, поиска/замены текста. Во время сохранения и экспорта проекта в области сообщений появляются пояснения, также могут отображаться возникшие ошибки. Окно вывода текста (консоль) показывает сообщения Arduino, включающие полные отчеты об ошибках и другую информацию. Кнопки панели инструментов позволяют проверить и записать программу, создать, открыть и сохранить скетч, открыть мониторинг последовательной шины. Ниже представлен листинг кода моргания светодиодом.

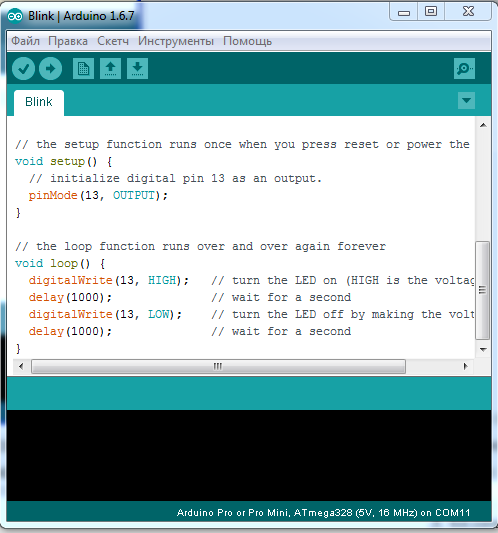


Рисунок 14 – Arduino IDE blink LED

Стандартный пример моргания светодиодом на Processing/Wiring:

void setup() {

// initialize digital pin 13 as an output.

pinMode(13, OUTPUT);

}

void loop() {

digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)

delay(1000); // wait for a second

digitalWrite(13, LOW); // turn the LED off by making the voltage LOW

delay(1000); // wait for a second

}

**1.4.2 ATMEL Studio**

ATMEL Studio – интегрированная среда разработки (IDE) для разработки 8 – и 32–битных AVR–приложений от компании Atmel, работающая в операционных системах Windows. AVR Studio содержит компилятор C/C++ и симулятор, позволяющий отследить выполнение программы. 7 версия поддерживает все выпускаемые на сегодняшний день контроллеры AVR. AVR Studio содержит в себе менеджер проектов, редактор исходного кода, инструменты виртуальной симуляции и внутрисхемной отладки, позволяет писать программы на ассемблере или на C/C++.

Код моргания светодиодом на с++ в Atmel Studio:

#define F\_CPU 16000000UL // 16 MHz clock speed

#endif

#include <avr/io.h>

#include <util/delay.h>

int main(void)

{

DDRC = 0xFF; //Nakes PORTC as Output

while(1) //infinite loop

{

PORTC = 0xFF; //Turns ON All LEDs

\_delay\_ms(1000); //1 second delay

PORTC= 0x00; //Turns OFF All LEDs

\_delay\_ms(1000); //1 second delay

}

}

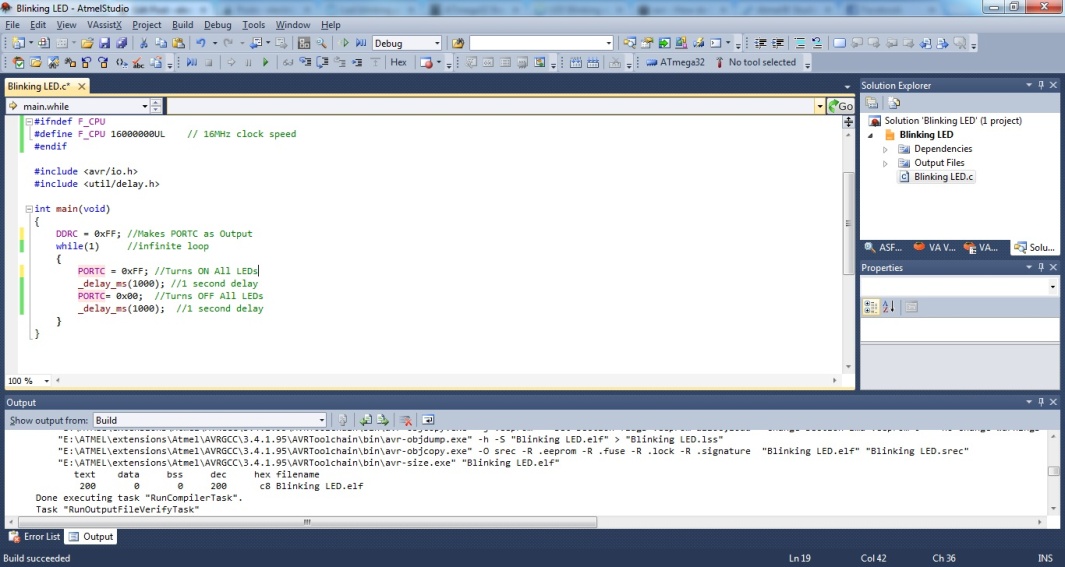


Рисунок 15 – Код моргания светодиодом в Atmel Studio 7

К Atmel Studio подключается модуль разработки микропрограмм для Arduino.

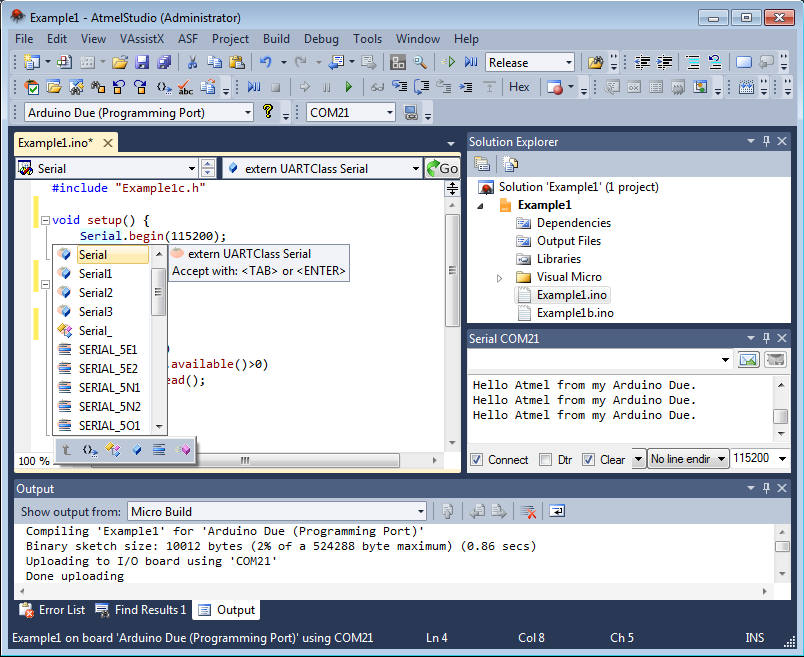


Рисунок 16 – Окно системы разработки Atmel Studio 7

Atmel Studio обладает огромным функционалом и производительностью на фоне Arduino IDE. К минусам относится малое количество поддерживаемых программаторов, а также их дороговизна.

**1.4.3 WinAVR**

WinAVR — программный пакет для операционных систем семейства Windows, включающий в себя кросс–компилятор и инструменты разработки для микроконтроллеров серий AVR и AVR32 фирмы [Atmel](https://ru.wikipedia.org/wiki/Atmel" \o "Atmel).

WinAVR и все входящие в него программы являются открытым программным обеспечением, выпущенным под лицензией GNU, но распространяются в скомпилированном виде. Версия WinAVR обозначается датой выхода (например, WinAVR–20060125 вышла 25 января 2006 года).

Последний релиз был выпущен 20 января 2010 года, после чего разработка была заморожена. Прекращение разработки было объяснено разработчиком тем, что он в настоящее время занят в разработке Atmel Studio, практически идентичный по набору входящих в него программ и включивший в себя большинство наработок WinAVR.

WinAVR включает мощные компиляторы Си и Assembler, программатор AVRDUDE, отладчик, симулятор и множество других вспомогательных программ и утилит.

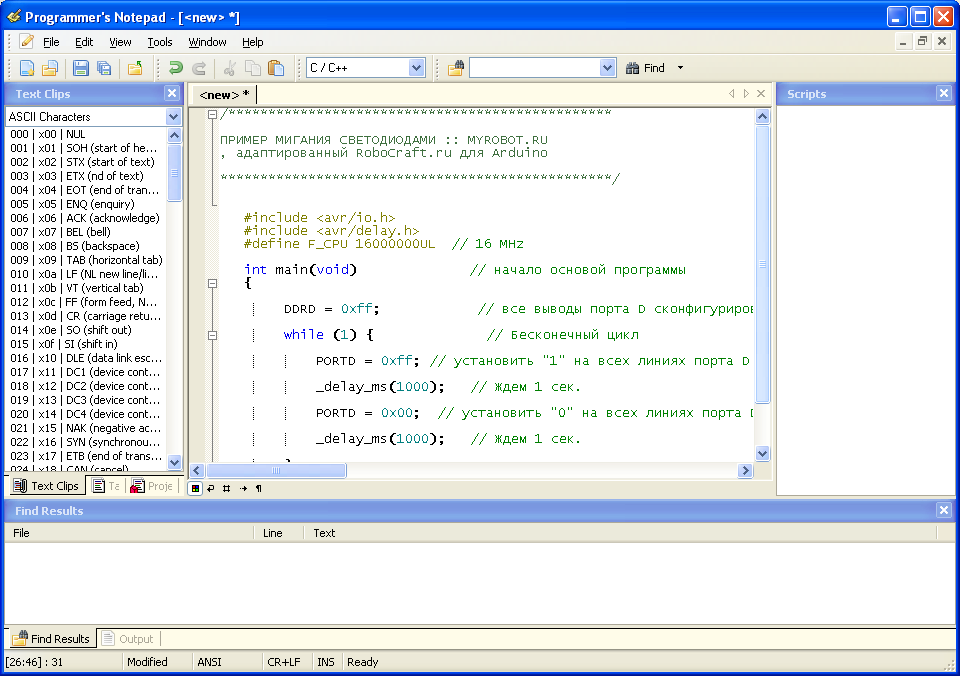


Рисунок 17– WinAVR

**1.5 Беспроводной интерфейс Arduino.**

Беспроводная связь Arduino – способ передачи информации без проводов. Можно управлять на дистанционном расстоянии самодельными роботами, осуществлять передачу звука на расстоянии, и другие виды информации.

ESP8266 – микроконтроллер китайской компании Espressif с интерфейсом Wi–Fi. Кроме Wi–Fi, микроконтроллер отличается отсутствием флеш–памяти в SoC, программы пользователя исполняются из внешней флеш–памяти с интерфейсом SPI. В связи с низкой ценой, микроконтроллер привлек большое внимание в 2014 году. Весной 2016 года началось производство ESP8285, совмещающей ESP8266 и флеш–память на 1 МБайт. Осенью 2015 года Espressif представила развитие линейки — микросхему ESP32 и модули на её основе.

**1.5.1 Микроконтроллер.**

Микроконтроллер не имеет на кристалле пользовательской энергонезависимой памяти. Исполнение программы ведется из внешней SPI ПЗУ путём динамической подгрузки требуемых участков программы в кэш инструкций. Подгрузка идет аппаратно, прозрачно для программиста. Поддерживается до 16 МБ внешней памяти программ. Возможен Standard, Dual или Quad SPI интерфейс.

Производитель не предоставляет документации на внутреннюю периферию микроконтроллера. Вместо этого он дает набор библиотек, через API которых программист получает доступ к периферии. Поскольку эти библиотеки интенсивно используют ОЗУ контроллера, то производитель в документах не указывает точное количество ОЗУ на кристалле, а только приблизительную оценку того количества ОЗУ, что останется пользователю после линковки библиотек – порядка 50 кБ. Энтузиасты, исследовавшие библиотеки ESP8266, предполагают, что он содержит 32 кБ кэша инструкций и 80 кБ ОЗУ данных.

**1.5.2 Распиновка ESP8266**

Источник исполняемой программы ESP8266 задается состоянием портов GPIO0, GPIO2 и GPIO15 в момент окончания сигнала Reset (то есть подачи питания). Наиболее интересны два режима: исполнение кода из UART (GPIO0 = 0, GPIO2 = 1 и GPIO15 = 0) и из внешней ПЗУ (GPIO0 = 1, GPIO2 = 1 и GPIO15 = 0). Режим исполнения кода из UART используется для перепрошивки подключенной флеш–памяти, а второй режим – штатный рабочий.

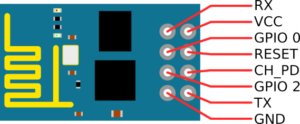


Рисунок 18– Распиновка ESP8266

**1.5.3 Программный комплект разработчика**

Программные средства разработки (SDK) состоят из:

* Компилятора. Компилятор для Xtensa LX106 входит в пакет компиляторов GNU Compiler Collection. Компилятор имеет открытые исходные тексты. В разных SDK могут содержаться разные сборки этого компилятора, немного отличающиеся поддерживаемыми опциями.
* Библиотек для работы с периферией контроллера, стеков протоколов WiFi, TCP/IP.
* Средств загрузки исполняемого файла в память программ микроконтроллера.
* Опциональной IDE.

**1.5.4 Функциональность ESP8266**

ESP8266 может работать как в роли точки доступа, так и оконечной станции. При нормальной работе в локальной сети ESP8266 конфигурируется в режим оконечной станции. Для этого устройству необходимо задать SSID Wi–Fi сети и в закрытых сетях, пароль доступа. Для первоначального конфигурирования этих параметров удобен режим точки доступа. В режиме точки доступа устройство видно при стандартном поиске сетей в планшетах и компьютерах. Остается подключиться к устройству, открыть HTML страничку конфигурирования и задать сетевые параметры. После чего устройство штатно подключится к локальной сети в режиме оконечной станции. В случае исключительно местного использования возможно всегда оставлять устройство в режиме точки доступа, что снижает необходимые усилия пользователя по его настройке.

**1.6. NodeMсu ESP9266**

NodeMCU – это платформа IoT (интернет вещей) с открытым исходным кодом. Он включает в себя встроенное программное обеспечение, которое работает на ESP8266 Wi–Fi SoC (однокристальная система) от Espressif Systems, и аппаратное обеспечение, основанное на модуле ESP–12. Термин «NodeMCU» относится не к комплектам для разработки, а к прошивке. В прошивке используется язык сценариев Lua. Он был основан на проекте eLua и основан на Espressif Non–OS SDK для ESP8266.

Существует несколько поколений плат NodeMcu – V1(версия 0.9), V2(версия 1.0) и V3 (версия 1.0). Обозначения V1, V2, V3 используются при продаже в интернет-магазинах. Нередко происходит путаница в платах – например, V3 внешне идентична V2. Также все платы работают по принципу open-source, поэтому их могут производить любые фирмы. Но в настоящее время производством плат NodeMcu занимаются Amica, DOIT и LoLin/Wemos. Внешний вид ESP8266 NodeMcu представлен на рисунке 19.



Рисунок 19 – ESP8266 NodeMcu

NodeMCU был создан вскоре после выхода ESP8266. 30 декабря 2013 года Espressif Systems начала производство ESP8266. ESP8266 – это Wi–Fi SoC, интегрированный с ядром Tensilica Xtensa LX106, широко используемое в приложениях IoT. NodeMCU был запущен 13 октября 2014 года, когда Хонг передал первый файл nodemcu – firmware на GitHub. Два месяца спустя, проект расширился и включил открытую аппаратную платформу, когда разработчик Хуанг Р. передал файл gerber платы ESP8266 с именем devkit v0.9. Позже в том же месяце Tuan PM перенес клиентскую библиотеку MQTT из Contiki на платформу SoC ESP8266 и посвятил себя проекту NodeMCU, затем NodeMCU смог поддержать протокол IQT MQTT, используя Lua для доступа к брокеру MQTT. Еще одно важное обновление было сделано 30 января 2015 года, когда Devsaurus портировал u8glib в проект NodeMCU, позволяя NodeMCU легко управлять ЖК–дисплеями, экранами, OLED и даже VGA – дисплеями. Летом 2015 года создатели отказались от проекта прошивки, и группа независимых разработчиков вступила во владение. К лету 2016 года в NodeMCU вошло более 40 различных модулей. Из–за нехватки ресурсов пользователям необходимо выбрать модули, соответствующие их проекту, и создать прошивку с учетом их потребностей.

**1.6.1 Распиновка NodeMcu v3**

Модуль V3 имеет 11 контактов ввода-вывода общего назначения. Помимо этого, некоторые из выводов обладают дополнительными функциями:

* D1-D10 – выводы с широтно-импульсной модуляцией;
* D1, D2– выводы для интерфейса I²C/TWI;
* D5–D8 – выводы для интерфейса SPI;
* D9, D10 – UART;
* A0 – вход с АЦП.



Рисунок 20 – Распиновка NodeMcu v3

**2 Разработка аппаратной части системы мониторинга медицинских показателей**

В наше время медицинские учреждения всё чаще госпитализируют людей на длительный срок. Для качественного лечения пациента требуется постоянный мониторинг жизненно важных медицинских показателей, а именно: температуры тела и сердечного ритма. Для более удобного мониторинга все показатели будут отображаться на экране у врача, что увеличит оперативность реагирования во время экстренных ситуаций. Именно для выполнения этой цели разработано устройство мониторинга медицинских показателей «Biomark».

**2.1 Основной модуль**



Рисунок 19– Внешний вид прототипа устройства

Данное устройство служит для обработки данных и обеспечивает взаимодействие с web – интерфейсом.

**2.1.1 Функциональные возможности**

Устройство выполняет беспрерывный мониторинг медицинских показателей пользователя с помощью датчика температуры и датчика сердечного ритма, связывается с сетью для просмотра данных на web – сайте.

Так же это устройство не привязано к розетке и работает автономно в среднем 10 часов.

Датчики подключаются с внешней стороны устройства с помощью штекеров Jack3,5, что существенно облегчает транспортировку и эксплуатацию устройства.

**2.1.2 Сборка устройства и индикация**

Для обеспечения работоспособности устройства, необходимо:

* подключить аккумуляторные батарейки;
* аккумуляторных батарейки AA. Рекомендуемое напряжение одной аккумуляторной батарейки – 3,7 вольт;
* включится желтый светодиод – он обозначает, что устройство включено;
* включится синий светодиод – он обозначает, что устройство подключено к точке Wi–Fi и то, что данные передаются на сервер;
* закрепить датчик пульса на палец с помощью специального зажима, а датчик температуры разместить в подмышечной впадине;

**2.1.3 Сервисное обслуживание устройства**

Для замены датчиков пульса и сердечного ритма считывающие медицинские показатели, необходимо при выключенном устройстве извлечь старый датчик и заменить его на рабочий. Датчики должны соответствовать прежним конфигурациям.

Для работы сети требуется подключиться к точке Wi–Fi. При настройке точки Wi–Fi, требуется извлечь из корпуса модуль Wi–Fi NodeMCU, подсоединить к компьютеру с помощью кабеля microUSB. Требуется установить программу Arduino IDE и запустить скетч «Wi–Fi\_to\_serv.ino». Изменяем параметры под конфигурацию точки Wi–Fi:

* ssid – полное название точки доступа;
* password – пароль точки доступа;

В случае обламывания штекера датчика или гнезда, требуется заменить на новые.

Обслуживание состоит в замене аккумуляторных батареек питания устройства.

* + 1. **Строение устройства**

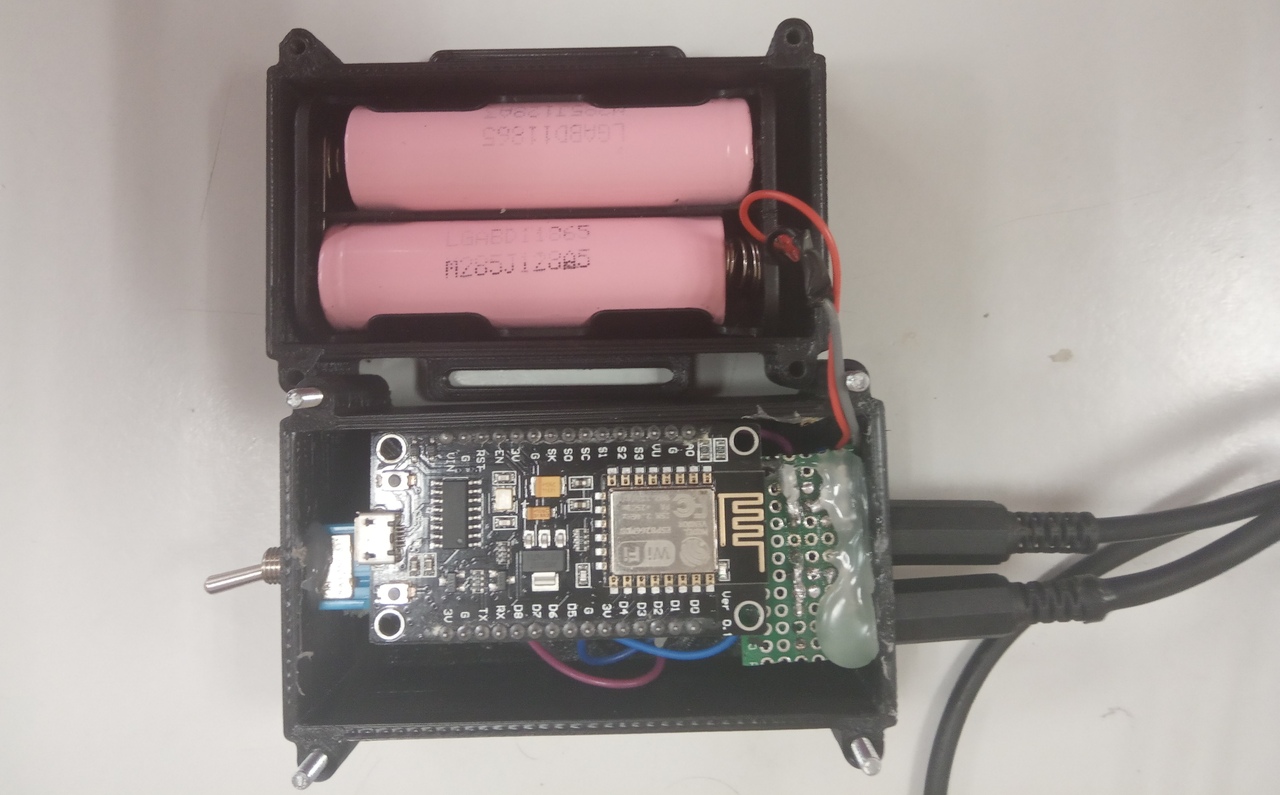


Рисунок 20– Внутренняя часть прототипа устройства

Состав электронного блока (см. Рисунок 20):

* Arduino Nano;
* NodeMCU esp8266 12–E;
* Датчик температуры;
* Датчик сердечного ритма;
* Стягивающий резистор на 4,7кОм;
* Жёлтый светодиод;
* Синий светодиод;
* Держатель для аккумуляторных батареек;
* Модельная плата.
  + 1. **Принципиальная схема**

Для общего понятия как работает устройство, была разработанна схема устройства (см. Рисунок 21). Это облегчит понимание работы устройства.

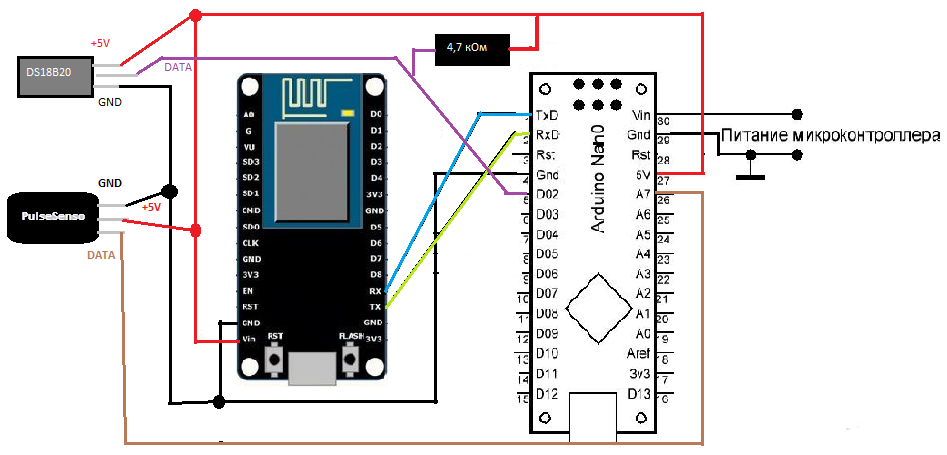


Рисунок 21– Схема устройства.

* + 1. **Алгоритм работы устройства**

В устройстве 2 микроконтроллера: Arduino Nano и NodeMCU esp8266 12–E. Arduino Nano служит главным микроконтроллером, NodeMCU esp8266 12–E служит для передачи показателей с датчиков на сервер.

Алгоритм работы Arduino Nano: После подключения питания запускаются все датчики, устанавливается соединение с Wi–Fi. В процессе работы оба датчика выводят показатели в NodeMCU esp8266 12–E для отправки на сервер. При наличии интернет соединения, NodeMCU esp8266 12–E подключается к серверу. Инициализируется связь между Arduino Nano и NodeMCU esp8266 посредством программного Serial. Во время загрузки устройства, подключение к интернету и последующей работы идет индикация о текущем состоянии посредством светодиодов.

Алгоритм работы NodeMCU*:* После соединения с интернетом устанавливается связь с сервером. С помощью бесконечного цикла данные передаются беспрерывно и отображаются на сайте.

* + 1. **Тестирование устройства**

Таблица 1 – Тестирование устройства

|  |  |
| --- | --- |
| Тест | Результат |
| Включение питания с подключенным модулем Wi–Fi. | Устройство сразу подключается к сети интернет и начинает передачу показателей с датчиков. |
| Включение питания без доступа в интернет. | Данные считываются, но не передаются в сеть. |
| Измерение температуры в подмышечной впадине. | Измерение происходит в штатном режиме, погрешность не обнаружена. |
| Тест | Результат |
| Измерение пульса. | Пульс измеряется в штатном режиме, погрешность н обнаружена. |
| Отключение питания во время работы устройства. | При следующем соединении с сервером поток данных возобновляется. |

**ЗАКЛЮЧЕНИЕ**

За время выполнения выпускной квалификационной работы было разработано и собрано устройство мониторинга медицинских показателей «Biomark».

В процессе разработки ставилась задача спроектировать устройство, которое отличалось простотой и удобством. Цели и задачи выполнены в полном объёме. Были использованы различные электронные компоненты, микроконтроллеры AVR, датчики, а также модули для них. Также было проведено тестирование устройства.

Были выполнены разработка, сборка, программирование и отладка устройства. В процессе разработки, также были решены следующие проблемы:

* В связи с не большим объемом FLASH и оперативной памяти, на микроконтроллерах AVR, для адекватной работы потребовалось провести оптимизацию кода посредством переосмысления используемых типов данных, а также использования функций.
* Измерение температуры потребовало много разных вариантов датчиков температуры. В связи с погрешностями в показателях измерения температуры, потребовалось выполнить перебор многих датчиков и выбрать с минимальной погрешностью.
* Возникла проблема в поиске антиаллергенного материала для крепления датчиков.
* Для удобства была создана система подключения датчиков к устройству снаружи, с помощью штекеров. Проблематичность была вызвана в подборке подходящих штекеров, учитывая форм–фактор устройства.

Разработанное устройство обеспечивает оперативный доступ к медицинским показателям человека.

Достоинства устройства:

* практичность и удобство в эксплуатации устройства;
* дешевизна устройства;
* мониторинг данных ведется через любое устройство с доступом в интернет;
* используется алгоритм, экономящий интернет–трафик;
* устройство полностью портативное;

**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Белов, А.В. Создаем устройства на микроконтроллерах [Электронный ресурс] : учебное пособие / А.В. Белов. — Электрон. дан. — Санкт–Петербург: Наука и Техника, 2007.—304 с. — Режим доступа: https://e.lanbook.com/book/35947.
2. Белов, А.В. ARDUINO: от азов программирования до создания практических устройств [Электронный ресурс] / А.В. Белов. — Электрон. дан. — Санкт–Петербург: Наука и Техника, 2018.—480 с. — Режим доступа: https://e.lanbook.com/book/109415.
3. Бейктал, Д. Конструируем роботов на Arduino. Первые шаги [Электронный ресурс] : руководство / Д. Бейктал ; пер. с англ. О.А. Трефиловой. — Электрон. дан. — Москва: Лаборатория знаний, 2016. — 323 с. — Режим доступа: https://e.lanbook.com/book/94205.
4. Баранов, В.Н. Применение микроконтроллеров AVR: схемы, алгоритмы, программы [Электронный ресурс] : учебное пособие / В.Н. Баранов. — Электрон. дан. — Москва : ДМК Пресс, 2010. — 288 с. — Режим доступа: <https://e.lanbook.com/book/60980>
5. Белов, А.В. Самоучитель разработчика устройств на микроконтроллерах AVR [Электронный ресурс]: самоучитель / А.В. Белов. — Электрон. дан. — Санкт–Петербург: Наука и Техника, 2008. — 544 с. — Режим доступа: https://e.lanbook.com/book/35894
6. Гололобов, В.Н. ARDUINO для любознательных или паровозик из Ромашково [Электронный ресурс] / В.Н. Гололобов. — Электрон. дан. — Санкт–Петербург : Наука и Техника, 2018. — 368 с. — Режим доступа: https://e.lanbook.com/book/109412
7. Боровский, А.С. Программирование микроконтроллера Arduino в информационно–упрвляющих системах [Электронный ресурс] : учебное пособие / А.С. Боровский, М.Ю. Шрейдер. — Электрон. дан. — Оренбург : ОГУ, 2017. — 113 с. — Режим доступа: https://e.lanbook.com/book/110615
8. Белов, А.В. Программирование ARDUINO. Создаем практические устройства [Электронный ресурс] / А.В. Белов. — Электрон. дан. — Санкт–Петербург : Наука и Техника, 2018. — 272 с. — Режим доступа: https://e.lanbook.com/book/109413
9. Белов, А.В. Микропроцессоры AVR В радиолюбительской практике [Электронный ресурс] : справочник / А.В. Белов. — Электрон. дан. — Санкт–Петербург : Наука и Техника, 2007. — 352 с. — Режим доступа: https://e.lanbook.com/book/35895
10. Алиев, М.Т. Микропроцессоры и микропроцессорные системы управления. 8–разрядные процессоры семейства AVR: лабораторный практикум [Электронный ресурс] : учебное пособие / М.Т. Алиев, Т.С. Буканова. — Электрон. дан. — Йошкар–Ола : ПГТУ, 2016. — 64 с. — Режим доступа: https://e.lanbook.com/book/92576
11. Гуров, В.В. Архитектура микропроцессоров [Электронный ресурс] : учебное пособие / В.В. Гуров. — Электрон. дан. — Москва : , 2016. — 327 с. — Режим доступа: https://e.lanbook.com/book/100570

# ПРИЛОЖЕНИЕ А

Техническое задание

**Наименование программы**

Системы мониторинга медицинских показателей.

**Краткая характеристика области применения**

Данное устройство применяется в медицинских учреждениях.

**Основание для разработки**

Разработка проводится на основании задания на дипломный проект, для студентов Института среднего профессионального образования (ИНСПО), обучающихся по специальности 09.03.02 «Программирование в компьютерных системах».

**Наименование темы разработки**

«Разработка аппаратной части системы мониторинга медицинских показателей».

**Условное обозначение темы разработки** – «Biomark».

**Руководителем разработки является** – Титов Н.Г.

**Назначение разработки**

Устройство «Biomark» предназначено для мониторинга медицинских показателей, просмотр данных осуществляется с помощью web – интерфейса.

**Требования к устройству:**

* Устройство беспрерывно считывает данные и отправляет на сервер;
* Защищена от механического повреждения;
* Длительная автономная работа устройства;
* Бесперебойная работа устройства;

**Требования к функциональным характеристикам:**

* Выдавать запрашиваемую информацию;
* Вывод информации;
* Корпус защищает устройства от повреждений электронных компонентов;
* Заменять батарейки после истечения заряда;
* Устройство автономно, может проработать до 10 часов;

**Требования к надёжности**

* Выполнять свою работу бесперебойно;
* Не допускать сбоев во время работы;

**Условия эксплуатации**

Эксплуатация состоит в замене батареек;

В случае неисправности, необходимо обратиться к программисту организации.

**Этапы разработки:**

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

* разработка изделия;
* разработка программной документации;
* испытания изделия.

На стадии внедрения должен быть выполнен этап разработки – подготовка и передача программы.

# ПРИЛОЖЕНИЕ Б

Листинг программы

#include <iarduino\_SensorPulse.h>

#include <OneWire.h>

#include <DallasTemperature.h>

#define ONE\_WIRE\_BUS 2

#define esp Serial

iarduino\_SensorPulse Pulse(A7);

OneWire oneWire(ONE\_WIRE\_BUS);

DallasTemperature sensors(&oneWire);

void setup() {

esp.begin(9600);

Serial.begin(9600);

pinMode(9, OUTPUT);

pinMode(7, OUTPUT);

digitalWrite(7, HIGH);

sensors.begin();

Pulse.begin();

}

void loop() {

sensors.requestTemperatures();

double kol;

kol = sensors.getTempCByIndex(0);

int pulse;

//pulse = Pulse.check(ISP\_PULSE);

pulse = random(60,80);

Serial.println(kol);

int id = 1;

String url = "/temp?temp1="+(String)kol+"&pulse="+(String)pulse+"&ids="+(String)id+'\n';

esp.print(url);

esp.flush();

delay(1000);

digitalWrite(9, HIGH);

}

**ПРИЛОЖЕНИЕ В**

Акт использования программного продукта

Настоящий Акт свидетельствует, что система мониторинга медицинских показателей, разработанный Сиркиным Никитой Алексеевичем, внедрен в лабораторию робототехники и мехатроники КубГУ.

Система мониторинга медицинских показателей предполагала наличие следующих основных функций:

* мониторинг медицинских показателей;
* отображение данных на Web – сайте;
* обратная связь;
* информация о зарегистрированных пользователях;

В ходе эксплуатации устройства подтверждено, что он обладает всеми заявленными функциями.

Сотрудник НИЧ

Лаборатория робототехники и мехатроники КубГУ Прутский А.С.

Сиркин Никита Алексеевич «Разработка аппаратной части системы мониторинга медицинских показателей» – Краснодар, 2019. – \_\_\_с.

Научный руководитель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Рецензент: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дипломный проект выполнен мною совершенно самостоятельно. Все использованные в работе материалы из опубликованной литературы и других источников имеют ссылки на них.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО выпускника, (подпись)

Выпускная квалификационная работа представлена к защите

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Подпись)

Защищена «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_ 2019 г.

Протокол № \_\_\_\_ от «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г.

Оценка: «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»