электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

 # 04, апрель 2016

УДК 004.67, 53.087

Мониторинг и обработка значений температуры и влажности воздуха с использованием малобюджетного оборудования

Желудков А. В., *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Программное обеспечение ЭВМ и информационные технологии»

Макаров Д. В., *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Программное обеспечение ЭВМ и информационные технологии»

Научный руководитель: Ковтушенко А.П., к.ф.-м.н., доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Программное обеспечение ЭВМ и информационные технологии»
bauman@bmstu.ru

Описание проблемы

Температура (в дальнейшем под этим термином будет подразумеваться температура воздуха, как окружающей среды) и влажность (в дальнейшем под этим термином будет подразумеваться относительная влажность воздуха, как окружающей среды) являются важными характеристиками для многих устройств, так как при достижении определённых пределов изменяются условия протекания различных процессов, и, как следствие, оборудование может начать давать сбои, выводить некорректные данные или перестать работать. Именно поэтому важно постоянно наблюдать даже за незначительными температурными изменениями и изменениями влажности воздуха. Для мониторинга значений этих физических величин используются приборы, называемые датчиками температуры и влажности (предполагается обобщённое название измерительных приборов).

Основная проблема состоит в том, что профессиональные системы реального времени, решающие задачи подобного рода имеют высокую стоимость, но во многих ситуациях не требуется предоставляемая ими точность полученных данных и скорость реакции на какие-либо изменения. Далее приведён перечень подобных областей:

- проведение дополнительного мониторинга в серверных комнатах;
- выполнение различных действий в системах типа «умный дом»;

- экологические и климатические исследования в школьных и студенческих проектах;

и т.д.

Абсолютная погрешность значений температуры не превышающая 3 °C и значений относительной влажности воздуха не более 7 % в данных ситуациях можно считать приемлемыми. Здесь и возникает задача мониторинга значений температуры и влажности воздуха с использованием малобюджетного оборудования. Для обработки поступающий данных в подобных ситуациях актуально задействовать персональный компьютер, а в качестве считывающего устройства – микроконтроллер с подключённым к нему датчиком. Для взаимодействия с микроконтроллером можно использовать стандартный драйвер, поставляемый с оборудованием, но в таком случае происходит жёсткая привязка к определённому приложению, проводящему всю предварительную настройку соответствующих портов и знающему протоколы «общения» с внешним устройством. Ни о каком расширении системы или её модернизации без углубления в описанные вопросы в таком случае не может идти и речи. Также при любом изменении протокола взаимодействия появляется необходимость сильного изменения программного кода. Ещё одним недостатком является то, что обработчик получаемых данных оказывается жёстко привязан к определённой операционной системе, так как функции и реализуемые ими подходы для выполнения подготовительной работы предоставляются ОС. Таким образом, получается «монолитная система» и любое, даже незначительное изменение в любой её части приводит к существенным изменениям во всех остальных модулях.

В качестве варианта решения описанной проблемы выступает возможность переноса всех подготовительных операций в драйвер операционной системы, который будет предоставлять приложению для обработки данных интерфейс для взаимодействия с устройством через подключаемую библиотеку. Обойтись только библиотекой в данной ситуации нельзя, так как в подобном случае не будет поддерживаться технология Plug-and-Play (в Windows), т.е. появится необходимость ручного выбора порта и образуется зависимость от способа взаимодействия с внешним устройством. При использовании драйвера одна библиотека может обслуживать несколько устройств с совершенно разными протоколами передачи данных.

Ввод подобных дополнительных прослоек позволит абстрагировать вычисления, проводимые на уровне пользователя, а единый кроссплатформенный интерфейс, предоставляемый соответствующими библиотеками, на различных ОС обеспечит необходимую гибкость предлагаемого комплекса.

Целью данной работы является разработка загружаемого модуля ядра Windows (везде в дальнейшем под данной фразой будет подразумеваться операционная система Windows 8.1 x86) и Linux (везде в дальнейшем под данной фразой будет подразумеваться операционная система Linux Mint 17), который будет запрашивать значения температуры и влажности, измеряемые датчиком, расположенным на плате микроконтроллера, и передавать эти данные пользовательскому приложению для их обработки.

Для достижения поставленной цели необходимо решить следующие задачи:

- выбрать подходящий микроконтроллер, имеющий датчик температуры и влажности;
- определить тип порта, через который будут передаваться данные;
- выбрать наиболее подходящий вид драйвера уровня ядра Windows;
- спроектировать и реализовать модуль ядра;
- спроектировать и реализовать библиотеку для обращения к драйверу из пространства пользователя;
- спроектировать и реализовать пользовательское приложение для обработки полученных данных.

Выбор микроконтроллера для решения сформулированной задачи

Рассмотрим две популярные торговые марки одноплатных компьютеров и определим наиболее подходящую для получения данных с датчика температуры и влажности.

1. Arduino — торговая марка аппаратно-программных средств для построения простых систем автоматики и робототехники. Программная часть состоит из бесплатной программной оболочки (IDE) для написания программ, их компиляции и программирования аппаратуры. Аппаратная часть представляет собой набор смонтированных печатных плат, продающихся как официальным производителем, так и сторонними производителями. Полностью открытая архитектура системы позволяет свободно копировать или дополнять линейку продукции Arduino. Arduino может использоваться как для создания автономных объектов автоматики, так и подключаться к программному обеспечению на компьютере через стандартные проводные и беспроводные интерфейсы.

Под торговой маркой Arduino выпускается несколько плат с микроконтроллером (англ. boards) и платы расширения. Большинство плат с микроконтроллером снабжены минимально необходимым набором обвязки для нормальной работы микроконтроллера (стабилизатор питания, кварцевый резонатор, цепочки сброса и т. п.).

Arduino и Arduino-совместимые платы спроектированы таким образом, чтобы их можно было при необходимости расширять, добавляя в устройство новые компоненты. Эти платы расширений подключаются к Arduino посредством установленных на них штыревых разъёмов. Существует ряд плат с унифицированным конструктивом, допускающим конструктивно жесткое соединение процессорной платы и плат расширения в стопку через штыревые линейки. Кроме того, выпускаются платы с уменьшенными габаритами (например, Nano, Lilypad) и уменьшенным числом специальных конструктивов для задач робототехники. Независимыми производителями также выпускается большая гамма всевозможных датчиков и исполнительных устройств, в той или иной степени совместимых с базовым конструктивом Arduino.

В концепцию Arduino не входит корпусный или монтажный конструктив. Разработчик выбирает метод установки и механической защиты плат самостоятельно. Сторонними производителями выпускаются наборы робототехнической электромеханики, ориентированной на работу совместно с платами Arduino.

Микроконтроллер данной торговой марки имеет существенное преимущество, а именно возможность использования датчика температуры и влажности в виде платы-расширения, которую при необходимости или в случае выхода из строя можно будет заменить. Также полностью открытая архитектура системы позволит в процессе написания драйвера обращаться к необходимым данным. К минусам можно отнести отсутствие у платы корпуса и необходимость отдельной покупки датчика.

2. Raspberry Pi — одноплатный компьютер размером с банковскую карту, изначально разработанный как бюджетная система для обучения информатике, впоследствии получивший намного более широкое применение и популярность, чем ожидали его авторы. Выпускается в следующих версиях:

- «A» (Процессор ARM1176JZ-F, 256 Мб ОЗУ, 26 пинов GPIO, 1 USB порт);
- «A+» (Процессор ARM1176JZ-F, 512 Мб ОЗУ, 40 пинов GPIO, 1 USB порт);
- «B» (Процессор ARM1176JZ-F, 512 Мб ОЗУ, 26 пинов GPIO, 2 USB порта, с ethernet);
- «B+» (Процессор ARM1176JZ-F, 512 Мб ОЗУ, 40 пинов GPIO, 4 USB порта, с ethernet);
- «2B» (Процессор Broadcom BCM2836 - 4 ядра ARM Cortex-A7, 1 Гб ОЗУ, 40 пинов GPIO, 4 USB порта, с ethernet).

Компьютер распространяется полностью собранным на четырёхслойной печатной плате, размером с банковскую карту. В стандартный комплект поставки входит только

сама плата. Корпус, блок питания, флеш карту и различные датчики необходимо заказывать отдельно.

Raspberry Pi работает в основном на операционных системах, основанных на Linux ядре. Запуск Windows возможен благодаря средствам виртуализации таким, как XenDesktop. ARM11 основан на 6 версии ARM, на котором несколько популярных версий Linux больше не запускаются. Для установки операционных систем существует инструмент NOOBS.

К плюсам микроконтроллера Raspberry Pi относится возможность покупки корпуса для защиты платы от внешних воздействий и возможность использования внешнего датчика температуры и влажности. Основными минусами являются закрытая архитектура системы, избыточность для поставленной задачи в виде наличия операционных систем и, как следствие, ресурсов для их запуска, что существенно увеличивает стоимость самого устройства. Также имеется необходимость отдельной покупки требуемого датчика.

В итоге был выбран микроконтроллер Arduino Mega 2560 с подключённым к нему датчиком DHT11, так как он обладает меньшей стоимостью (в соответствии с данными системы «яндекс маркет» на 05.11.2015), отсутствием избыточных возможностей в виде наличия полноценного компьютера с операционной системой на борту и открытой архитектурой, что позволит упростить процесс создания драйвера. Выбранный датчик имеет абсолютную погрешность измеренных значений температуры 2 °C и значений относительной влажности воздуха 5 %, что удовлетворяет заявленным ранее приемлемым значениям не более 3 °C и 7 %.

Определение типа порта, через который будут передаваться данные о температуре

В соответствии с выбранным микроконтроллером были проанализированы два порта передачи данных с платы на персональный компьютер под управлением операционной системы Windows 8.1: usb порт и порт rs-232, эмулируемый через usb соединение.

1. USB (англ. Universal Serial Bus — «универсальная последовательная шина») — последовательный интерфейс передачи данных для среднескоростных и низкоскоростных периферийных устройств в вычислительной технике. Символом USB являются четыре геометрические фигуры: большой круг, малый круг, треугольник и квадрат, расположенные на концах древовидной блок-схемы.

Разработка спецификаций на шину USB производится в рамках международной некоммерческой организации USB Implementers Forum (USB-IF), объединяющей разработчиков и производителей оборудования с шиной USB.

Для подключения периферийных устройств к шине USB используется четырёхпроводный кабель, при этом два провода (витая пара) в дифференциальном включении используются для приёма и передачи данных, а два провода — для питания периферийного устройства. Благодаря встроенным линиям питания USB позволяет подключать периферийные устройства без собственного источника питания (максимальная сила тока, потребляемого устройством по линиям питания шины USB, не должна превышать 500 мА, у USB 3.0 — 900 мА, у USB 3.1 до 5А).

Несмотря на все вышеперечисленные плюсы, в рамках поставленной задачи данный вид порта обладает сильной избыточностью, что серьёзно усложняет работу по написанию драйвера уровня ядра.

2. RS-232 (англ. Recommended Standard 232) — физический уровень асинхронного (UART) интерфейса. Исторически имел широкое распространение в телекоммуникационном оборудовании для персональных компьютеров. В настоящее время всё ещё широко используется для подключения всевозможного специального оборудования к компьютерам, однако в основном он уже вытеснен интерфейсом USB.

Данный вид порта хотя и является устаревшим, но обладает значительным преимуществом, а именно более простой спецификацией для его использования на уровне драйвера.

В итоге был выбран метод передачи данных через порт RS-232, так как он обладает меньшей избыточностью для поставленной задачи и не накладывает дополнительных технических ограничений, по причине отсутствия необходимости в его физическом присутствии на персональном компьютере, поскольку связь осуществляется путём эмулирования соединения через USB шину.

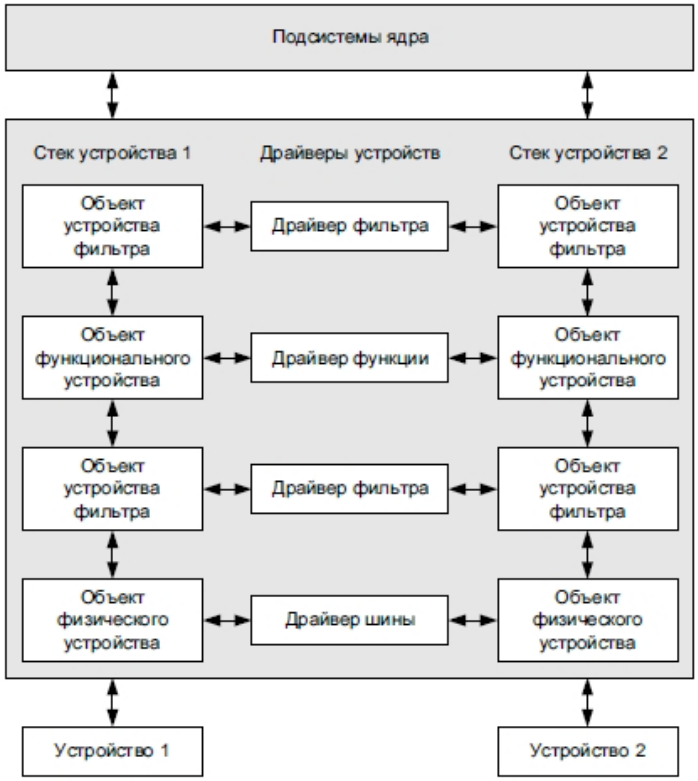
Анализ вида драйверов уровня ядра ОС Windows

Запрос ввода/вывода, направленный подсистемой ядра устройству, обрабатывается одним или несколькими драйверами. Каждый драйвер имеет ассоциированный с ним объект устройства, который представляет участие драйвера в обработке запросов ввода/вывода данного устройства. Объект устройства — это структура данных, содержащая указатели на рабочие процедуры, которые позволяют менеджеру ввода/вывода взаимодействовать с драйвером.

Объекты устройств организованы в стек устройства, при этом для каждого устройства создается отдельный стек. Обычно под термином «стек устройства» подразумевается стек объектов устройства с их связанными драйверами. Но стек устройства ассоциируется только с одним устройством, в то время как набор драйверов

может обслуживать несколько стеков устройств. Набор драйверов иногда называется стеком драйверов.

На рисунке показаны два устройства (каждый со своим стеком устройства), которые обслуживаются одним набором драйверов.



Стек устройства состоит из следующих компонентов:

- Драйвер шины и объект физического устройства. Внизу стека находится объект физического устройства (в дальнейшем PDO), который ассоциирован с драйвером шины. Устройства обычно подключаются к стандартным аппаратным шинам, таким как, например, PCI или USB. Драйвер шины обычно управляет несколькими аппаратными устройствами, подключенными к физической шине.

Например, после установки драйвер шины перечисляет устройства, подключенные к шине, и запрашивает ресурсы для этих устройств. PnP-менеджер использует эту информацию для выделения ресурсов каждому устройству. Каждое устройство имеет собственный объект PDO. PnP-менеджер идентифицирует драйверы для каждого устройства и создает соответствующий стек устройств вверху каждого объекта PDO.

- Драйвер функции и объект функционального устройства. Основным компонентом стека устройств является объект функционального устройства (в дальнейшем объект FDO (Functional Device Object)), который ассоциируется с драйвером функции. Драйвер функции преобразует абстракцию устройства, создаваемую Windows, в

настоящие команды, необходимые для обмена данными с физическим устройством. Он предоставляет так называемую «верхнюю кромку» (иными словами, интерфейс устройства) для взаимодействия с приложениями и устройствами, и обычно определяет, каким образом драйвер реагирует на изменения в состоянии Plug and Play или энергопотребления. Так называемая «нижняя кромка» драйвера устройства обрабатывает взаимодействие с устройством или другими драйверами, такими как, например, расположенный ниже драйвер фильтра или драйвер шины.

- Драйверы фильтров и объекты устройства фильтра. Стек устройства может иметь несколько объектов устройства фильтра (в дальнейшем объект FiDO (Filter Device object)), которые могут располагаться вокруг объекта FDO. С каждым объектом FiDO ассоциируется драйвер фильтра. Драйверы фильтров не являются обязательными, но часто присутствуют. Сторонние поставщики могут снабжать стек устройства драйверами фильтров с целью придания ему дополнительных возможностей, таким образом, повышая его стоимость. Обычно драйвер фильтра служит для модифицирования некоторых запросов ввода/вывода, когда они проходят через стек устройств, в значительной степени подобно тому, как аудиофильтр модифицирует аудиопоток.

Например, с помощью драйверов фильтра можно зашифровывать или расшифровывать запросы на чтение и запись. Драйверы фильтра можно также применять для выполнения задач, не требующих модифицирования запросов ввода/вывода, например, таких как отслеживание запросов и предоставление информации обратно отслеживающему приложению.

Для считывания данных о температуре и влажности с датчика микроконтроллера в ОС Windows было принято решение о написании Plug and Play драйвера фильтра верхнего уровня (расположенного над драйвером функции в стеке устройств), который будет выполнять следующие задачи:

- подготавливать порт для работы с платой, на которой расположен датчик;
- конфигурировать и отправлять запрос о чтении данных по обращению пользовательского приложения;
- принимать ответ от платы, вычленять оттуда требуемое значение и отправлять его обратно в пользовательское приложение.

Был выбран драйвер фильтр, как вид драйвера уровня ядра ОС Windows, так как его функциональности достаточно для выполнения поставленной задачи и отсутствует необходимость в реализации сложных операций по управлению питанием и прямым

взаимодействием с платой, поскольку они выполняются посредством стандартных инструментов, предоставляемых разработчиками операционной системы.

Анализ вида драйверов ОС Linux

Традиционно Unix разделяет все устройства на три основных типа. Каждый модуль, как правило, реализует функциональность одного из этих типов и отсюда может классифицироваться как символьный модуль, блочный модуль или сетевой модуль. Это деление не жесткое, программист может создать один большой модуль, в котором будет реализовано несколько различных драйверов. Ниже приводятся краткие описания этих трех классов:

Символьные устройства

Символьное устройство - это устройство, которое поставляет данные как поток байт. Драйвер символьного устройства отвечает за реализацию поддержки подобного поведения. Такие драйверы реализуют по меньшей мере четыре системных вызова: open, close, write и read. Типичными примерами устройств могут служить текстовая консоль (/dev/console) и последовательный порт (/dev/ttyS*). К символьным устройствам можно обращаться посредством элементов файловой системы, таких как /dev/tty1 или /dev/lp0. Единственное важное отличие обычных файлов от символьных устройств заключается в том, что имеется возможность перемещаться по файлам назад и вперёд, в то время как символьные устройства - это лишь каналы передачи данных, доступ к которым осуществляется в заданном порядке. Тем не менее, существуют символьные устройства, допускающие произвольный доступ к данным в потоке. Типичным примером служат устройства захвата изображения, где приложения могут обращаться к файлу целиком, используя mmap или lseek.

Блочные устройства

Подобно символьным устройствам, к блочным устройствам также можно обращаться посредством элементов файловой системы в каталоге /dev. Самым известным примером блочного устройства может служить жёсткий диск. Обмен данными с блочным устройством производится порциями байт - блоками. В большинстве Unix-систем размер одного блока равен 1 килобайту или другому числу, являющемуся степенью числа 2. Linux позволяет приложениям обращаться к блочному устройству так же, как к символьному - он разрешает передачу любого числа байт в блоке. В результате все различие между блочными и символьными устройствами сводится к внутреннему представлению данных в ядре. Драйвер блочного устройства реализует точно такой же интерфейс с ядром, что и драйвер символьного устройства, но дополнительно реализуется еще и блочно-ориентированный интерфейс, который «невидим» для пользователя или

приложения, открывающего доступ к блочному устройству посредством псевдофайловой системы /dev. Тем не менее, блочный интерфейс необходим, для выполнения монтирования (mount) файловой системы.

Сетевые интерфейсы

Любой сетевой обмен выполняется через сетевой интерфейс, т.е. устройство, которое способно обмениваться данными с другими узлами сети. Как правило, это некая аппаратная составляющая, но возможна и исключительно программная реализация сетевого девайса, например петлевое устройство loopback (локальный сетевой интерфейс). Сетевой интерфейс отвечает за передачу и получение пакетов данных, которыми управляет сетевая подсистема ядра, ничего не зная о том, к каким соединениям эти пакеты принадлежат. Несмотря на то, что соединения по протоколам Telnet и FTP используют один и тот же сетевой интерфейс, само устройство не различает эти соединения, оно «видит» только пакеты данных.

Не являясь потоковым устройством, сетевой интерфейс не может быть отображен на файловую систему, как это делается в случае с /dev/tty1. Традиционно Unix предоставляет доступ к интерфейсу, назначая ему уникальное имя (например, eth0), но файла с этим именем в каталоге /dev нет. Принципы взаимодействия ядра с драйвером сетевого устройства в корне отличаются от принципов, применимых к символьным или блочным устройствам. Вместо функций write и read, ядро вызывает соответствующие функции, управляющие передачей пакетов.

В Linux существуют другие классы модулей драйверов, которые добавляют в ядро возможность взаимодействия с определенными типами устройств, но их рассмотрение выходит за рамки данной работы.

Для реализации драйвера, отвечающего за взаимодействие с платой, разработан модуль символьного устройства, так как он полностью соответствует типу выбранного микроконтроллера.

Выбор языка программирования для реализации пользовательского приложения и библиотеки

Пользовательское приложение и библиотека для обращения к драйверу написаны на языке высокого уровня C++ в составе кроссплатформенной среды визуального программирования Qt.

Данный язык был выбран, потому что он поддерживает парадигму объектно-ориентированного программирования, обеспечивает модульность, раздельную

компиляцию, обработку исключений, абстракцию данных, объявление типов (классов) объектов, виртуальные функции. Стандартная библиотека включает, в том числе, общеупотребительные контейнеры и алгоритмы. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков.

Кроме того, среда визуального программирования Qt предоставляет большое количество стандартных визуальных компонентов для создания интерфейса и ряд библиотек с различными часто используемыми полезными функциями. Поскольку программа написана с использованием библиотек Qt, то для её запуска они должны быть установлены на компьютере.

Выводы о возможностях реализованного программного комплекса и рекомендации к его использованию

На основе вышеприведённого анализа были разработаны программа, библиотека и драйвер уровня ядра ОС Windows и Linux, которые решают поставленные задачи, а именно:

- производится конфигурация порта, к которому подключено оборудование для взаимодействия с операционной системой;
- считывается информация о температуре и влажности с датчика, расположенного на плате микроконтроллера, и эти данные передаются пользовательскому приложению для их обработки;
- полученные данные анализируются и представляются в наглядном графическом виде.

Разработанный драйвер и библиотека предоставляют разработчикам такие возможности как:

- отсутствие необходимости в дополнительной работе по конфигурированию устройства независимо от номера порта, к которому оно подключено, благодаря поддержке технологии Plug and Play в ОС Windows;
- использование объектно-ориентированного подхода при взаимодействии с устройством;
- использование встроенной системы исключений, готовой к конечному взаимодействию с пользователем благодаря возможности получения текстового user-friendly сообщения об ошибках выполнения;
- возможность компилирования исходного кода драйвера для успешной работы в операционных системах Windows 7 x86, Windows 7 x64, Windows 8 x86,

Windows 8 x64, Windows 8.1 x64, Windows 8.1 x86, Windows 10 x64, Windows 10 x86 из-за использования современного подхода к написанию драйверов на базе пакета Windows Development Kit (для работы на -x64 операционных системах, возможно, потребуется отдельное приобретение цифровой подписи компании Microsoft);

- лёгкость установки драйвера благодаря использованию –inf файлов в ОС Windows.

Разработанное программное обеспечение, использующее библиотеку и драйвер, обладает такими преимуществами как:

- мультиплатформенность, достигнутая за счёт использования средств разработки Qt, т.е. возможность использования соответствующего интерфейса на других платформах с другой библиотекой и драйвером;
- независимость от использования конкретного драйвера благодаря абстрагированию от реализации с помощью библиотеки и через паттерн проектирования wrapper;
- возможность использования получившейся системы на любом персональном компьютере, имеющем порт USB и работающем под управлением операционной системы Windows 8.1 x86 или Linux Mint 17.

Реализованный программный комплекс рекомендуется использовать в областях, не требующих высокой точности получаемых данных и реакции на их изменения в режиме реального времени, т.е. где абсолютную погрешность значений температуры не превышающую 3 °C и значений относительной влажности воздуха не более 7 % можно считать приемлемыми, например, в студенческих проектах.

Список литературы

[1]. Arduino. Available at: https://ru.wikipedia.org/wiki/Arduino (accessed 06.11.2015).

[2]. Penny O., Guy S. Developing Drivers with the Windows Driver Foundation. Washington: Microsoft Press, 2007. 928 p.

[3]. Jonathan C., Alessandro R., Greg Kroah-Hartman. Linux Device Drivers. 3th ed. Sebastopol: O'Reilly Media, 2005. 636 p.

[4]. Raspberry Pi. Available at: https://ru.wikipedia.org/wiki/Raspberry_Pi (accessed 06.11.2015).

[5]. USB. Available at: https://ru.wikipedia.org/wiki/USB (accessed 06.11.2015).

[6]. RS-232. Available at: https://ru.wikipedia.org/wiki/RS-232 (accessed 06.11.2015).

[7]. Характеристики датчика температуры и влажности DHT11. Режим доступа: http://www.avislab.com/blog/wp-content/uploads/2014/02/DHT11.pdf (дата обращения 06.11.2015).