БАЛТИЙСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ им. ИММАНУИЛА КАНТА

С. Н. Чижма, С. В. Молчанов

МОДЕЛИРОВАНИЕ МИКРОКОНТРОЛЛЕРОВ В СРЕДЕ PROTEUS

Учебно-методическое пособие

Учебное электронное издание

Калининград
Издательство Балтийского федерального университета им. И. Канта
2026

ISBN 978-5-9971-1045-1

Рецензент

С. М. Русаков, канд. техн. наук, доцент,
Балтийская государственная академия рыбопромыслового флота

Чижма, С. Н.

Моделирование микроконтроллеров в среде Proteus : учебно-методическое пособие / С. Н. Чижма, С. В. Молчанов [Электронный ресурс] : учебное электронное издание. — Калининград : Издательство БФУ им. И. Канта, 2026. — https://publish.kantiana.ru/catalog/non-periodical/uchebnye-posobiya/modelirovanie-mikrokontrollerov-v-srede-proteus/

Посвящено изучению процесса программирования микроконтроллеров с помощью программной среды моделирования электронных схем Proteus. Приведено описание программного пакета, рассмотрены основные приемы работы с измерительными приборами. Особое внимание уделено процессу программирования микроконтроллеров, агрегации программы Proteus с программной средой Arduino IDE. Приведены методики моделирования нескольких электронных схем, включающих микроконтроллеры.

Предназначено для студентов высших учебных заведений по специальности 09.03.02 «Информационные системы и технологии» и смежных специальностей.

ISBN 978-5-9971-1045-1

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

При разработке микропроцессорных систем неизбежно появляются ошибки, которые можно обнаружить, изучая работу опытного образца. Однако создание прототипа устройства, включающего микросхемы с большим количеством выводов, сопряжено со значительными трудностями. Эта проблема решается с помощью программного обеспечения для схемотехнического моделирования. В этом случае объектом анализа выступает принципиальная схема устройства с обозначением связей между ее элементами. С учетом моделей всех компонентов и знания их соединений появляется возможность моделировать функционирование всего устройства. Она особенно ценна при изучении микроконтроллеров и нюансов проектирования микропроцессорных систем. Отпадает необходимость в изготовлении физической платы или ее прототипа, что исключает риск повреждения компонентов из-за неправильного монтажа.

Среди программных инструментов, таких как PSpice, MicroCap, Multisim, DesignLab и других, выделяется система Proteus от Labcenter Electronics. Она дает возможность моделировать принципиальные схемы, используя обширную библиотеку готовых моделей электронных компонентов, в том числе различных микроконтроллеров (AVR, MCS51, PIC, ARM и др.). Proteus оснащена комплектом виртуальных измерительных приборов, таких как осциллограф, логический анализатор, вольтметр, спектроанализатор и др. Они позволяют определять и наглядно представлять электронное состояние в любой точке моделируемой схемы, а также отслеживать происходящие в ней процессы.

Программный пакет Proteus состоит из двух основных частей: ISIS, предназначенной для создания и моделирования

электронных схем, и ARES, используемой для разработки печатных плат. Вместе с пакетом Proteus поставляется набор демонстрационных проектов, облегчающих освоение функционала системы. Ознакомление с ними перед детальным изучением Proteus будет весьма полезным. Данный контекст сконцентрирован только на модуле ISIS, рассматриваются лишь его базовые возможности, достаточные для эффективной работы с системой. Более глубокое понимание можно достичь путем практического исследования или изучения материалов, доступных в меню «Справка» .

1. СИСТЕМА СХЕМОТЕХНИЧЕСКОГО МОДЕЛИРОВАНИЯ PROTEUS

При запуске Proteus появляется главное окно программы (рис. 1), имеющее несколько функциональных зон.

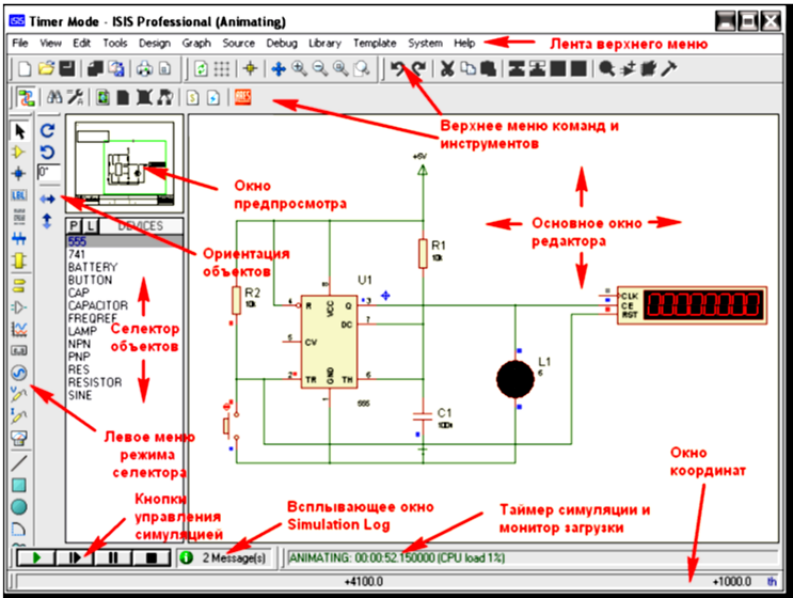

Рис. 1. Главное окно системы схемотехнического моделирования Proteus

Центральное место занимает редактор схем, служащий для создания электрических схем проектируемых устройств. В редакторе пользователь располагает электронные компоненты, а также измерительные и отображающие приборы, которые импортируются из имеющихся библиотек. Затем элемен-

ты соединяются проводами согласно проекту, исправляются обнаруженные ошибки, после чего начинается этап моделирования функционирования схемы.

В верхней части окна находятся пункты меню, предоставляющие доступ ко всем возможностям Proteus. Под ними расположены кнопки верхней панели инструментов, предназначенные для часто используемых действий, таких как открытие и сохранение проектов, изменение размера и расположения схемы, а также копирование и перемещение выбранных фрагментов. Слева располагается панель инструментов, в основном используемая для построения принципиальных схем. Доступ к функциям Proteus возможен через меню, иконки на панелях или контекстные меню, вызываемые правым кликом мыши.

В левом верхнем углу главного окна находится окно предпросмотра, облегчающее навигацию по схеме проекта. Под ним расположено информационное окно, содержание которого меняется в зависимости от выбранной кнопки на левой панели инструментов. Здесь может отображаться перечень компонентов, меток, виртуальных приборов, пробников и прочего. В нижней части окна находятся элементы управления моделированием: «Воспроизвести» для начала симуляции, «Шаг» для пошагового выполнения кода микроконтроллера, «Пауза» для приостановки процесса и «Стоп» для его завершения.

Навигация по инструментам Proteus.

Ключевым элементом для построения схем в Proteus является вертикальная панель слева от рабочего поля. Рассмотрим ее основные функции.

 *Режим выбора:* данный инструмент позволяет редактировать схему, включая выделение компонентов, их копирование, удаление, масштабирование и прочее. Позиционирование схемы осуществляется перемещением курсора в окне предварительного просмотра. Фиксация положения происходит по двойному клику мыши. Кнопки масштабирования в верхней панели дают возможность настроить любой участок схемы.

 *Компоненты:* при активации этой кнопки открывается список компонентов, применяемых в проекте. Выбрав компонент и кликнув в окне редактирования, можно разместить его на схеме. Двойной правый клик удаляет компонент. Кнопка «P» под окном предварительного просмотра открывает диалоговое окно Pick Device (рис. 1) для доступа к библиотеке компонентов Proteus. Компоненты сгруппированы по категориям. Библиотека содержит большое количество элементов, например, категория Microprocessors включает 16 семейств микроконтроллеров. Каждый компонент имеет описание и графическое представление. Кнопка «Ok» добавляет выбранный компонент в список проекта.

 *Точка соединения:* этот инструмент связывает точки пересечения проводников.

 *Метка соединения:* позволяет присваивать имена проводникам, устанавливая условные связи между выводами компонентов и цепями без прокладывания проводников.

 *Текстовый скрипт:* добавляет текстовые комментарии к схеме, облегчая документирование.

 *Шина:* создает на схеме шины, объединяющие несколько проводников.

 *Субсхема:* позволяет формировать именованные функциональные блоки с входными и выходными разъемами.

 *Терминал:* размещает на схеме различные концевые элементы, например, источники питания.

 *Выводы компонента:* добавляет выводы к создаваемым компонентам.

 *Диаграмма:* открывает доступ к инструментам для визуализации, анализа и математической обработки сигналов.

 *Генератор:* предоставляет список генераторов сигналов различных форм для формирования тестовых сигналов.

 *Датчик напряжения/тока:* используется для измерения напряжения или тока в определенной точке проводника.

 *Виртуальные инструменты:* отображает перечень доступных виртуальных приборов, таких как осциллограф, измерители напряжения и тока, виртуальный терминал.

Ключевое отличие виртуального генератора от обычного заключается в возможности настройки параметров тестового сигнала в реальном времени, не останавливая симуляцию.

Осциллограф позволяет наблюдать сигналы в режиме реального времени с регулировками скорости развертки и коэффициента отклонения. Для контроля напряжения или тока разработчик подключает вольтметр или амперметр к соответствующим цепям. Виртуальный терминал имитирует обмен данными между микроконтроллером и компьютером через последовательный RS-интерфейс.

2. ОСНОВНЫЕ ПРИЕМЫ РАБОТЫ С СИСТЕМОЙ PROTEUS

Начальный этап моделирования.

Моделирование работы электронного устройства начинается с создания его принципиальной схемы. Для этого необходимо подобрать в библиотеке Proteus нужные компоненты, перенести их в рабочую область редактора и соединить проводниками. По мере необходимости к схеме добавляются цепи питания, общая шина и виртуальные измерительные приборы.

Формирование перечня компонентов.

Первый шаг — формирование перечня всех необходимых элементов для создаваемого устройства. Для этого на левой инструментальной панели активируйте опцию «Компоненты», а затем нажмите клавишу «P», расположенную под областью предварительного просмотра. Перед вами появится окно выбора элементов из библиотеки, известное как Pick Devices (см. рис. 1, с. 6). Здесь все компоненты организованы в соответствии с их категориями, подкатегориями и производителями.

Библиотека содержит не только базовые составляющие, такие как резисторы, конденсаторы, транзисторы и микросхемы, но и широкий ассортимент дополнительных модулей и устройств. Это включает в себя оптоэлектронные компоненты (буквенно-цифровые и графические индикаторы, дисплеи, оптопары), электромеханические устройства (двигатели, реле), разнообразные разъемы и переключатели, датчики, а также функциональные блоки с предзаданными характеристиками и многое другое. Каждая позиция обладает своим уникальным графическим представлением на схеме, что облегчает идентификацию ее функции и особенностей.

Предположим, в вашем проекте необходим микроконтроллер ATmega128. В окне Pick Devices выберите раздел «Microprocessor ICs», затем подраздел «AVR Family». В появившемся перечне найдите ATmega128 (рис. 2). В верхнем правом углу окна отобразится его схематический образ. Нажатие кнопки

«OK» добавит ATmega128 в общий список компонентов проекта, который находится справа от основной панели инструментов. Аналогичный процесс применяется для выбора всех остальных элементов. Созданный список допускает последующие изменения: можно добавлять новые элементы или удалять те, что больше не требуются. Для ускорения также предусмотрена функция поиска по маске — достаточно ввести полное название компонента или его часть.

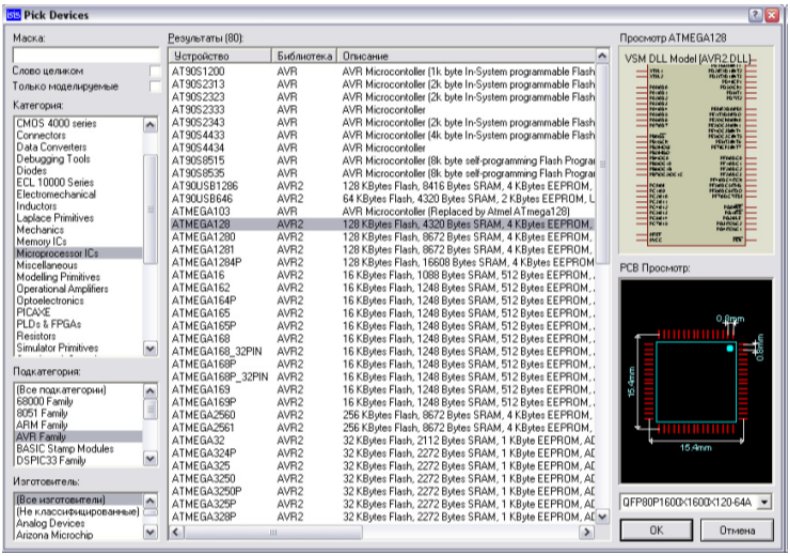

Рис. 2. Окно выбора библиотечных компонентов Pick Devices

Размещение элементов в рабочей области.

Для помещения элемента со списка на монтажную схему его необходимо выделить (используя мышь или кнопки клавиатуры). Затем, переместив курсор в свободное пространство рабочего поля, нажать левую кнопку мыши. Это действие приведет к появлению элемента в выбранной точке. Повторение этой процедуры позволяет разместить все необходимые компоненты. Позже их расположение можно корректировать: пе-

редвигать, вращать под нужным углом, дублировать через буфер обмена или удалять. Эти операции применимы как к отдельным элементам, так и к их группам. Для работы с группами предварительно выделяется нужный блок удерживанием левой кнопки мыши.

Двойной клик левой кнопкой мыши по любому компоненту открывает контекстное меню. Оно предоставляет доступ к редактированию свойств элемента, например, номиналов резисторов или емкостей конденсаторов. Для повышения наглядности схемы на нее можно добавлять текстовые пояснения, например, уточняющие функциональное назначение отдельных групп элементов. Для этого в свободном месте рабочего поля необходимо вызвать контекстное меню правой кнопкой мыши, выбрать опцию «Разместить > Текстовый скрипт», а затем кликом левой кнопки указать место для размещения надписи.

Проводное соединение компонентов.

После расстановки элементов их требуется соединить в соответствии с принципиальной схемой. Для этого курсор подводят к выводу компонента (на экране отображается контактная площадка), нажимают левую кнопку мыши и протягивают проводник к другому выводу или существующему проводнику. Фиксация соединения выполняется повторным нажатием левой кнопки. Если проводник прокладывается не так, как было задумано, его можно выстраивать по частям, фиксируя точки изгиба промежуточными кликами мыши.

Если после прокладки проводников потребуется соединить их в местах пересечения, следует воспользоваться кнопкой «Точка соединения» на боковой панели. Естественно, такие соединения можно формировать и в процессе прокладки проводников.

Для упрощения работы со сложными схемами, вместо физического соединения зачастую используют именование проводников. Присвоение одинакового имени двум или более линиям делает их электрически связанными. Для этого на боковой панели выбирают кнопку «Метка соединения», подводят

курсор к нужному проводнику, кликают левой кнопкой мыши и в появившемся окне вводят метку. Проводникам, помеченным таким образом, можно задавать оборванный конец (обрыв формируется двойным кликом левой кнопки в точке разрыва).

Еще один метод улучшения читаемости схемы при большом числе связей — использование шин, объединяющих несколько проводников. На боковой панели нажимается кнопка «Шина», после чего шина прокладывается в заданном месте рабочего поля, и к ней подключаются отдельные проводники. Все проводники, входящие в состав шины, должны иметь собственные названия (метки).

Навигация и масштабирование.

В процессе создания схемы часто возникает потребность изменить масштаб изображения. Для этого предназначены соответствующие кнопки на верхней панели инструментов. Для перемещения всей схемы используется окно предварительного просмотра, расположенное в левом верхнем углу основного окна. Курсор помещается в это окно, и при удерживаемой левой кнопке мыши перекрестие «прицела» перемещается, вызывая смещение схемы в рабочем поле. Повторный клик левой кнопкой мыши фиксирует новое положение. В случае нежелательных результатов всегда можно отменить последние действия или вернуть исходное отображение, нажав на верхней панели кнопку «Вписать во весь лист».

Формирование цепей питания и общей шины.

Любое устройство включает общую шину и одно или несколько питающих напряжений. Для их размещения на боковой панели инструментов нажимается кнопка «Терминал», выбирается пункт «GROUND», и затем двойным кликом левой кнопки мыши в нужном месте рабочего поля устанавливается «земля». Источник питания добавляется аналогичным образом через пункт «POWER», но для него дополнительно требуется указать величину напряжения. Для этого двойным кликом по графическому изображению источника открывается окно настройки, куда вводится значение, например +5.2V.

Подключение тестовых сигналов.

Для отладки работы устройства часто требуется подавать на вход или в другие точки схемы тестовые сигналы. На боковой панели инструментов нажимается кнопка «Генератор» и выбирается нужный тип источника (синусоидальный, импульсный и т. д.). Затем двойным кликом левой кнопки в рабочем поле устанавливается графический образ генератора, после чего повторным двойным кликом по нему открывается окно настройки параметров сигнала.

Для синусоидального сигнала (рис. 3) задаются амплитуда, пиковое или среднеквадратичное значение, частота (или период), фаза, коэффициент затухания, а также величина постоянной составляющей (смещение).

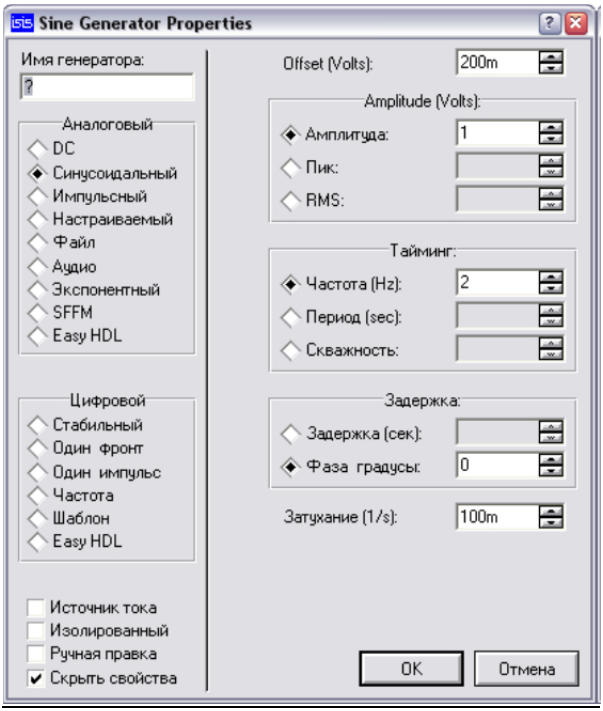

Рис. 3. Окно задания параметров синусоидального тестового сигнала

Proteus предлагает обширный выбор тестовых сигналов. Можно, например, использовать сигнал, заданный в цифровой форме и сохраненный в файле, или нарисовать нужную форму сигнала мышью на графике в координатах «напряжение — время» (для этого выбирается настраиваемый тип аналогового сигнала).

Использование виртуальных измерительных приборов.

В процессе симуляции часто возникает необходимость измерять параметры сигналов в отдельных точках схемы, проверять работу отдельных узлов или отлаживать интерфейсы микроконтроллера. Для этих целей предназначены виртуальные инструменты, вызываемые кнопкой «Виртуальные инструменты» на боковой панели. Наиболее востребованные из них: осциллограф, вольтметры и амперметры, генератор сигналов специальной формы, а также виртуальный терминал для передачи данных от микроконтроллера в компьютер.

Чтобы установить виртуальный прибор, его выбирают в соответствующем списке (см. рис. 1, с. 6), после чего кликают левой кнопкой в нужном месте рабочего поля. Появляется графическое изображение прибора, которое затем подключается к контролируемым точкам схемы или выводам компонентов. После запуска симуляции некоторые виртуальные приборы (осциллограф, генератор сигналов, логический анализатор и др.) преобразуются в лицевые панели с органами управления, позволяющими настраивать их под конкретные задачи.

Пример настройки 4-канального осциллографа.

В нем можно изменять развертку по времени и чувствительность каждого канала, инвертировать сигналы или складывать их. Для каждого канала доступен выбор режима входа — открытый или закрытый, что позволяет измерять полный сигнал или только его переменную составляющую. Если нажать правую кнопку мыши на лицевой панели осциллографа и выбрать пункт «Setup…», откроется диалог настройки цветовой гаммы дисплея (цвет луча, фона, сетки и курсоров).

3. ИЗМЕРИТЕЛИ: ВОЛЬТМЕТРЫ И АМПЕРМЕТРЫ

В составе библиотеки компонентов Proteus представлены интерактивные модели для измерения напряжения и тока. Эти виртуальные приборы функционируют в реальном времени и подключаются к схеме так же, как и физические компоненты. После запуска симуляции они мгновенно отображают измеренные значения — напряжение на выводах или проходящий через них ток — в легком для восприятия формате.

Доступные модели охватывают полный диапазон отклонения шкалы (FSD) для значений 10, 100 мА и 100 мкА. Разрешение измерений составляет три значащие цифры, а максимальное отображаемое число имеет два десятичных знака. К примеру, модель VOLTMETER может показывать напряжение от 0.01 до 99.9 В, а AMMETER-MILLI — ток от 0.01 до 100 мА. По умолчанию вольтметры обладают очень высоким входным сопротивлением, достигающим 100 МОм, которое можно скорректировать в настройках компонента. Если поле сопротивления оставить пустым, прибор будет работать без учета этого параметра.

Приборы, предназначенные для работы с переменным током, отображают среднеквадратические (RMS) значения, усредненные с заданной пользователем постоянной времени.

Для каждого типа измерителя предусмотрена возможность переключения диапазонов. Так, обычный вольтметр легко трансформируется в милливольтметр или даже микровольтметр.

4. ГЕНЕРАТОР СИГНАЛОВ

Генератор сигналов специальной формы (рис. 4) позволяет настраивать форму сигналов, их амплитуду и частоту. Важно отметить, что все настройки производятся непосредственно в процессе симуляции, что обеспечивает полное соответствие поведению реального лабораторного оборудования.

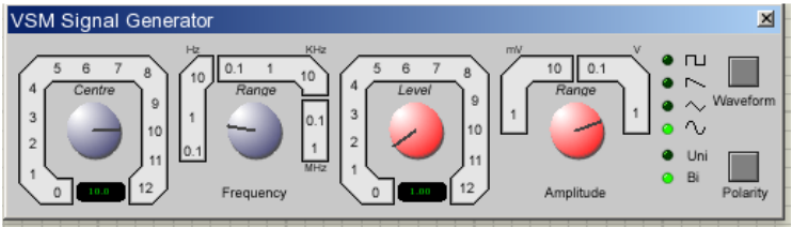

Рис. 4. Окно панели управления генератором

Общие сведения.

VSM Signal Generator — это простой генератор, способный формировать сигналы звукового диапазона, обладающий следующими характеристиками:

- генерирует сигналы различных форм: прямоугольные, пилообразные, треугольные и синусоидальные;
- рабочая частота находится в пределах от 0 до 12 МГц с возможностью выбора из 8 диапазонов;
- диапазон выходной амплитуды составляет от 0 до 12 В с возможностью выбора из 4 диапазонов;
- имеет входы для амплитудной (AM) и частотной (FM) модуляции.

Подключение и настройка генератора.

Для генерации простого звукового сигнала выполните следующие шаги.

1. Включите режим виртуальных инструментов (Virtual Instruments Mode). В появившемся окне выберите SIGNAL GENERATOR, поместите его на схему и подключите выход к требуемой точке схемы. Для схем, требующих симметричного сигнала, обычно необходимо подключить отрицательный вывод генератора к земле. Это можно сделать, добавив терминал GND через контекстное меню. Входы AM и FM на начальном этапе можно оставить отключенными.

2. Запустите симуляцию, нажав кнопку Play. Откроется окно настроек генератора.

3. Выберите необходимый частотный диапазон. Он определяет базовую частоту, соответствующую положению «1» на ручке-верньере.

4. Установите желаемую амплитуду. Диапазон амплитуды также учитывает положение «1» на верньере. Отображаемое значение — это пиковое значение выходного сигнала.

5. Нажимайте кнопку Waveform, пока не загорится индикатор нужной формы сигнала.

Использование входов AM и FM-модуляции.

Генератор поддерживает амплитудную и частотную модуляцию. Оба входа модуляции работают следующим образом:

- коэффициенты передачи модуляции (Гц/В для частоты, В/В для амплитуды) задаются параметрами Frequency Range и Amplitude Range;
- входное напряжение модуляции ограничено диапазоном ± 12 В;
- входы модуляции имеют бесконечно высокое входное сопротивление;
- напряжение на входе модуляции суммируется с текущим положением верньера, а затем результат умножается на установки диапазона для определения мгновенных значений частоты или амплитуды.

Например, если выбран частотный диапазон 1 кГц, верньер установлен на 2.0, а на вход FM подано 2 В, то выходная частота составит 4 кГц.

5. ЦИФРОВОЙ ОСЦИЛЛОГРАФ

Технические характеристики.

Цифровой осциллограф VSM представляет собой стандартное приложение для всех модификаций ProSPICE, включая как полнофункциональные, так и базовые варианты моделирования аналоговых схем (рис. 5).

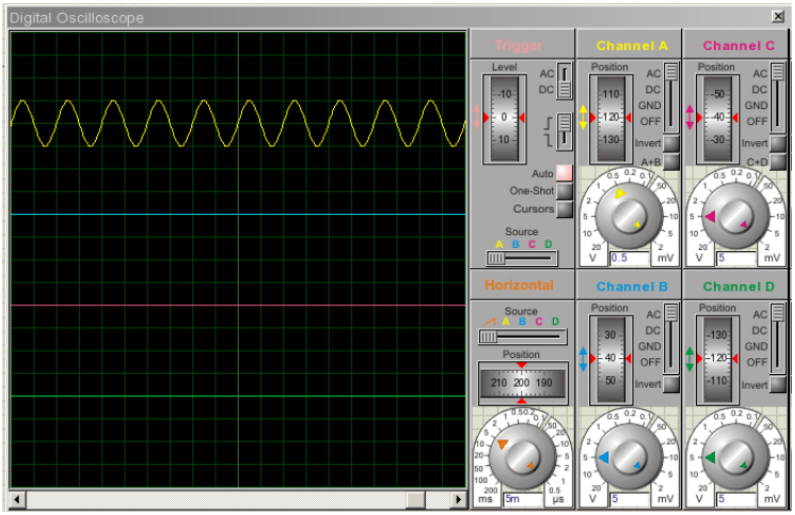

Рис. 5. Окно панели управления осциллографом

Устройство обладает следующими параметрами:

- четыре входных тракта, поддерживающих режим отображения X-Y;
- диапазон вертикального усиления: от 20 В/деление до 0,002 В/деление;
- диапазон горизонтальной развертки: от 0,2 с/деление до 500 нс/деление;

• автоматическая настройка уровня запуска для любого из каналов;

• возможность выбора входного режима: AC (переменный ток) или DC (постоянный ток).

Инструкция по эксплуатации.

Для визуализации аналогового сигнала выполните следующие шаги.

1. Активируйте режим виртуальных приборов, выбрав «OSCILLOSCOPE» из списка. Разместите символ осциллографа на рабочем пространстве и соедините его с интересующим узлом схемы.

2. Запустите симуляцию, нажав кнопку «Play». Появится окно осциллографа.

3. Подберите оптимальные настройки временной развертки и чувствительности по напряжению, учитывая частоту сигнала (развертка должна быть обратно пропорциональна частоте).

4. При наличии постоянной составляющей в сигнале, переключите соответствующие каналы в режим AC.

5. Настройте вертикальное усиление (Y) и смещение для наглядного отображения сигнала. Если переменная составляющая мала на фоне значительного постоянного напряжения, может потребоваться внешний разделительный конденсатор.

6. Выберите канал для запуска развертки.

7. Регулируйте уровень запуска, чтобы получить стабильное изображение. Триггер срабатывает по переднему или заднему фронту в зависимости от положения соответствующего регулятора.

Режимы работы.

Осциллограф VSM поддерживает несколько режимов:

• режим отдельных каналов (рис. 6): автоматический запуск развертки по выбранному каналу. Уровень запуска и временные параметры настраиваются соответствующими регуляторами. Этот режим удобен для анализа временных и фазовых сдвигов;

• режим одиночного запуска (One-Shot) (рис. 7): позволяет получить «снимок» сигнала в сложных случаях, когда автоматическая синхронизация нестабильна;

• режим курсорных измерений (рис. 8): используется для точного определения амплитудных и временных характеристик сигнала путем перемещения маркеров;

• режим построения фигур Лиссажу (рис. 9): позволяет визуализировать зависимость между двумя сигналами.

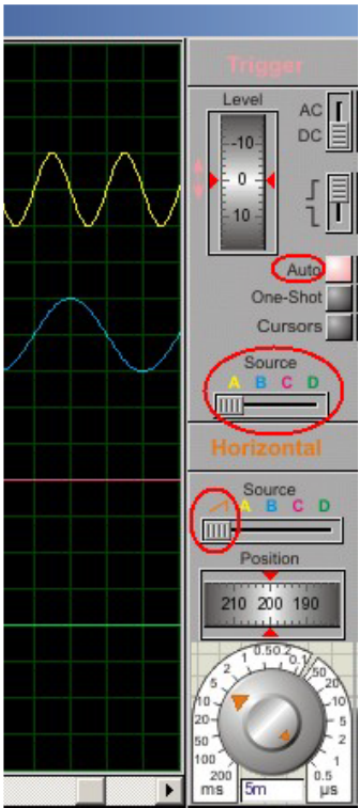

Рис. 6. Окно панели управления осциллографом в режиме отдельных каналов

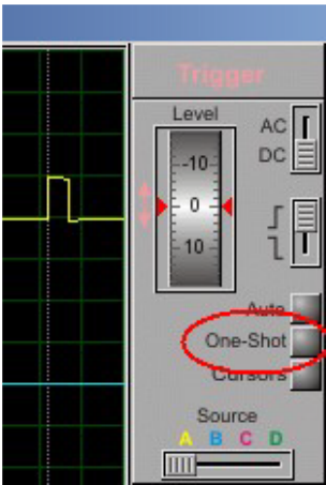

Рис. 7. Окно панели управления осциллографом
в режиме одиночного запуска

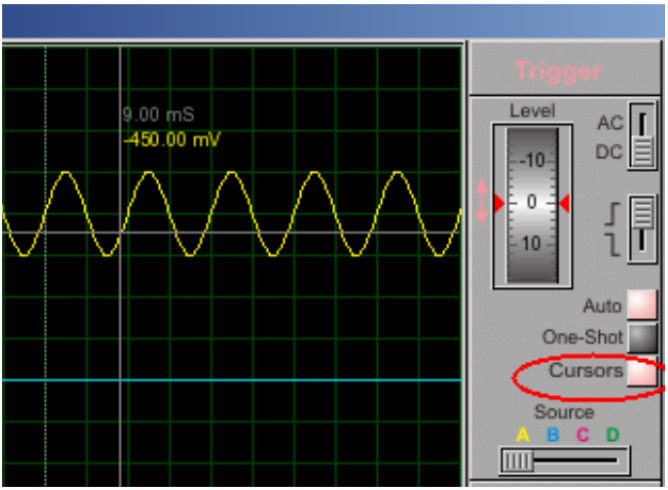

Рис. 8. Окно панели управления осциллографом
в режиме курсорных измерений

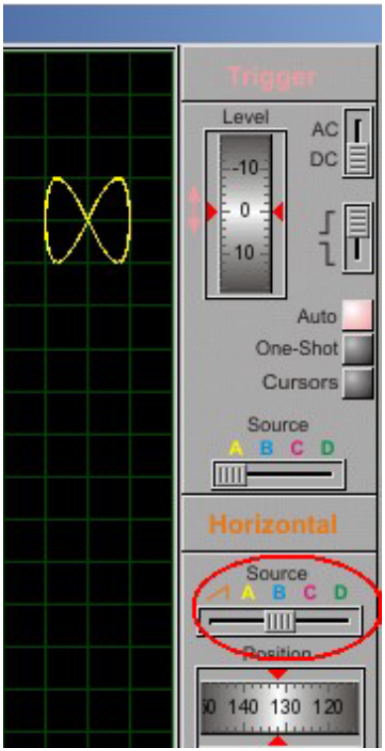

Рис. 9. Окно панели управления осциллографом
в режиме построения фигур Лиссажу

Для удобства пользователей, каждый из четырех каналов имеет свой цвет, соответствующий маркировке органов управления. Каналы можно отключать индивидуально для более детального анализа отдельных сигналов, а также использовать смещение луча для лучшей наглядности.

Синхронизация (Triggering).

Режим автоматической синхронизации позволяет привязать временную развертку ко входному сигналу. Выбор канала для синхронизации, установка уровня напряжения срабатыва-

ния и выбор фронта (передний / задний) осуществляются с помощью настроек на панелях Horizontal и Trigger. При отсутствии синхронизации в пределах одного интервала развертки она запускается автоматически.

Организация входных цепей.

Входные каналы могут работать в режимах DC (прямое подключение) или AC (через разделительный конденсатор). Режим AC рекомендуется для сигналов с малой переменной составляющей на фоне значительного постоянного напряжения. Положение GND (заземление) удобно для настройки координатной сетки.

6. АНАЛИЗАТОР ЛОГИЧЕСКИХ СИГНАЛОВ

Логический анализатор (рис. 10) является составной частью пакета PROTEUS VSM Professional; в версии Lite он предоставляется как опция. Его функционирование основано на методичном сохранении поступающих цифровых сигналов в обширном приемном буфере. Имеется возможность регулировки разрешения, определяющего минимальный регистрируемый временной интервал импульса.

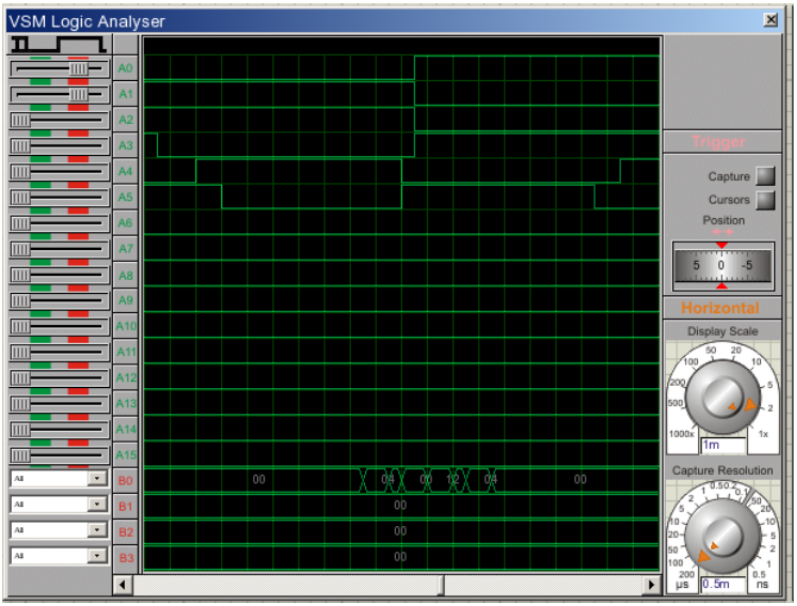

Рис. 10. Окно панели управления логическим анализатором

Кнопка «Capture» на панели запуска (Trigger) инициирует сбор данных, прекращающийся через определенное время после выполнения заданных условий срабатывания. Во время

записи кнопка меняет цвет, а после окончания процесса гаснет. Отображаемый диапазон данных включает фрагмент как до триггера, так и после него благодаря большому размеру буфера (10 000 точек). Для удобства работы с таким объемом информации предусмотрены инструменты масштабирования и панорамирования. Кроме того, измерительные маркеры «Cursors» позволяют с высокой точностью определять характеристики импульсов.

Анализатор логических сигналов имеет следующие параметры:

• 16 однобитных линий и 4 восьмибитных шины для мониторинга;

• буфер хранения данных: 10 000 интервалов по 24 бита;

• разрешение при захвате: от 200 мкс/отсчет до 0,5 нс/отсчет с соответствующим временем захвата от 2 с до 5 мс;

• масштаб отображения: от 1000 отсчет/деление до 1 отсчет/деление;

• условия срабатывания (триггер): комбинирование состояний (AND), переход фронтов с учетом данных на шинах;

• размещение точки срабатывания: 0, 25, 50, 75 и 100 % заполнения буфера;

• наличие курсоров для точных временных измерений;

• возможность настройки цветовой схемы (кривые, маркеры, текст и т. п.) через контекстное меню дисплея.

Анализ цифровых сигналов производится по следующему алгоритму.

1. Активируйте режим виртуальных инструментов, выберите «LOGIC ANALYSER», разместите его на рабочем пространстве и подключите входные выводы к исследуемым сигналам.

2. Запустите симуляцию (кнопка «Play») и откройте окно анализатора.

3. Настройте разрешение, учитывая, что более высокое значение позволит регистрировать более короткие импульсы, но уменьшит общее время сбора данных.

4. Сконфигурируйте условия запуска на левой панели. Например, для срабатывания при высоком уровне на канале 1 и при переходе сигнала на канале 3 из низкого в высокое состояние установите соответствующие параметры.

5. Определите предпочтительное соотношение отображаемых данных до и после триггера, отрегулировав положение ползунка «Position».

6. Нажмите «Capture». Индикатор загорится, и начнется сбор данных до выполнения триггера. После этого индикатор станет зеленым, и сбор продолжится до полного заполнения буфера. Затем индикатор погаснет, и результаты будут представлены на дисплее (рис. 11).

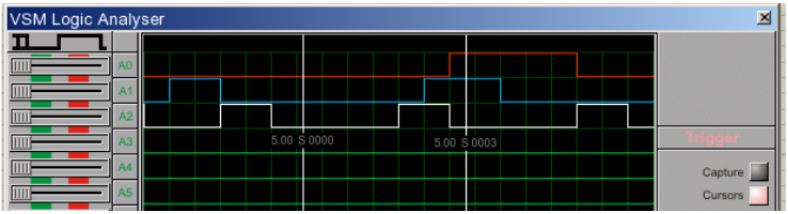

Рис. 11. Результат анализа логических сигналов

Для детального анализа данных в большом буфере (10 000 отсчетов) на ограниченном экране (250 пикселей) предусмотрены инструменты масштабирования и панорамирования. Параметр «Display Scale» регулирует количество отсчетов на одно деление шкалы, а полоса прокрутки позволяет перемещать изображение.

Для точных временных замеров используются курсоры. Перемещение каждого курсора осуществляется с помощью соответствующего элемента управления. Отображается временная позиция каждого маркера относительно точки срабатывания, а также разница между ними (Delta A-B). Детальное описание работы с курсорами представлено в разделе «Дисковые шкалы» применительно к логическому анализатору.

Практическая работа № 1

ИЗУЧЕНИЕ РАБОТЫ ВИРТУАЛЬНЫХ ИНСТРУМЕНТОВ В СИСТЕМЕ СХЕМОТЕХНИЧЕСКОГО МОДЕЛИРОВАНИЯ PROTEUS

Цели работы:

— изучение системы схемотехнического моделирования Proteus;

— изучение виртуальных приборов;

— изучение принципов моделирования электронных схем.

1.1. Краткие сведения

Proteus представляет собой программный комплекс класса САПР (система автоматизированного проектирования), предназначенный для разработки электронных схем. В основе работы этого пакета лежит моделирование на базе компонентных моделей, принятых в системе PSpice.

Ключевая особенность Proteus — возможность симуляции работы программируемых устройств: микроконтроллеров, микропроцессоров, цифровых сигнальных процессоров (DSP) и аналогичных компонентов. При этом в Proteus реализован принцип сквозного проектирования: если разработчик вносит изменения в логику работы схемотехники, система трассировки практически мгновенно учитывает эти правки.

Библиотека компонентов содержит не только графические модели, но и справочные данные. Дополнительно в состав Proteus включен инструмент для проектирования печатных плат.

Структурно пакет Proteus делится на две основные подпрограммы (модуля):

- ISIS — отвечает за синтез и моделирование непосредственно электронных схем;
- ARES — предназначен для разработки топологии печатных плат.

Для начального знакомства с возможностями системы вместе с программой поставляется набор демонстрационных проектов.

Кроме того, начиная с восьмой версии, в состав пакета входит среда разработки VSM Studio. Она позволяет оперативно писать и компилировать код для микроконтроллера, который используется в текущем проекте.

1.2. Порядок выполнения работ

1. В среде Proteus создайте схему, включающую виртуальный генератор сигналов произвольной формы и виртуальный осциллограф (рис. 12).

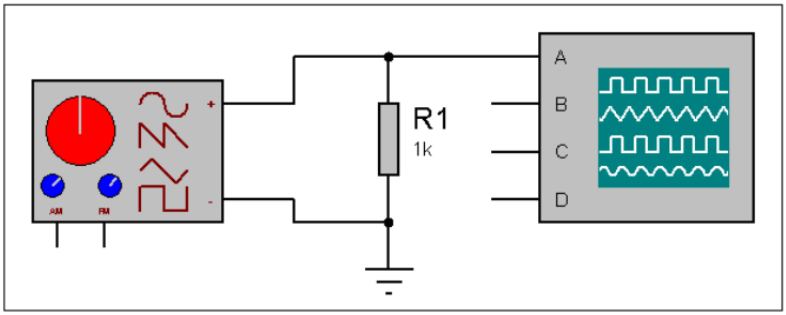

Рис. 12. Схема для изучения возможностей генератора и осциллографа

Сигналы от генератора подаются на резистивную нагрузку. Используя осциллограф, изучите форму сигналов и их ключевые характеристики, в том числе амплитуду, частоту и коэффициент заполнения.

2. Замените в схеме виртуальный генератор сигналов на физический, активируемый кнопкой «Генератор» из левой панели инструментов, и выполните аналогичные измерения, изменяя форму выходных сигналов и их основные параметры.

3. Подключите к нагрузке вольтметр переменного тока и определите действующее значение напряжения, которое изменяется по синусоидальному закону.

4. Подключите к нагрузке пробник для измерения постоянного напряжения, установите период синусоидального сигнала примерно 100 с и проведите измерения напряжения на нагрузке с помощью этого пробника.

5. Расширьте схему, добавив аналоговый графопостроитель «ANALOGUE» (кнопка «Диаграмма» на левой панели инструментов), и проведите измерение и анализ синусоидального сигнала с частотой приблизительно 20 Гц. Установите конечное время моделирования равным 5 с.

6. Дополните схему графическим спектроанализатором FOURIER и выполните измерение спектра синусоидального и импульсного сигналов с частотой около 10 Гц. Установите конечное время моделирования равным 10 с, максимальную частоту — 100 Гц, разрешение — 1 Гц. Перемещая курсор по спектру, измерьте амплитуду основной гармоники и сопоставьте ее со значением, заданным в настройках генератора синусоидального сигнала.

1.3. Содержание отчета

1. Цель работы.
2. Скриншоты схем по каждому выполненному пункту.
3. Скриншоты экранов генераторов и измерительных приборов.
4. Выводы.

1.4. Контрольные вопросы

1. Какие микроконтроллеры поддерживает Proteus? Как задать требуемые свойства микроконтроллеру?
2. Как выбрать элементы электрической схемы?
3. Как организовать связь между элементами электрической схемы, используя линии, шину, терминалы?

4. Как выбрать и подключить требуемый генератор?

5. Почему не рекомендуется использовать аналоговые генераторы без необходимости?

6. В каких случаях более рационально использовать Signal Generator?

7. Какие средства предусмотрены для измерения напряжения, тока? Как их подключить к схеме?

8. Как измерить частоту, период, временной интервал?

9. Какие задачи можно решить с помощью осциллографа? В каких случаях его следует использовать?

10. Для каких целей предназначен логический анализатор? Его функциональные особенности.

Практическая работа № 2

ПОДКЛЮЧЕНИЕ БИБЛИОТЕК ARDUINO К СИСТЕМЕ PROTEUS

Цели работы:

— изучение методов совместной работы Arduino IDE и Proteus;

— подключение библиотек Arduino к системе Proteus;

— эмуляция работы Arduino IDE в системе Proteus.

2.1. Краткие сведения

Для разработки под Arduino потребуется среда программирования Arduino IDE. Загрузить ее можно с официального сайта разработчика: https://www.arduino.cc/en/software.

Библиотека компонентов Proteus насчитывает огромное количество элементов, однако на данный момент в ней отсутствуют платформы семейства Arduino. Чтобы добавить поддержку Arduino в Proteus, необходимо установить соответствующую стороннюю библиотеку. Один из вариантов доступен для скачивания по адресу: https://www.theengineeringprojects.com/2021/03/download-proteus-library-of-arduino-modules.html.

Поскольку проекты, связанные с интеграцией Arduino в Proteus, постоянно развиваются, найти подходящую библиотеку можно и на других сайтах — любую поисковую систему легко сориентировать на соответствующий запрос.

2.2. Порядок выполнения работы

1. После скачивания распаковать полученные файлы баблиотек и переместить в корневую папку Proteus: ... \Labcenter

Electronics\Proteus 8 Professional\LIBRARY. Теперь в библиотеке компонентов можно найти основные варианты платформы Arduino (рис. 13).

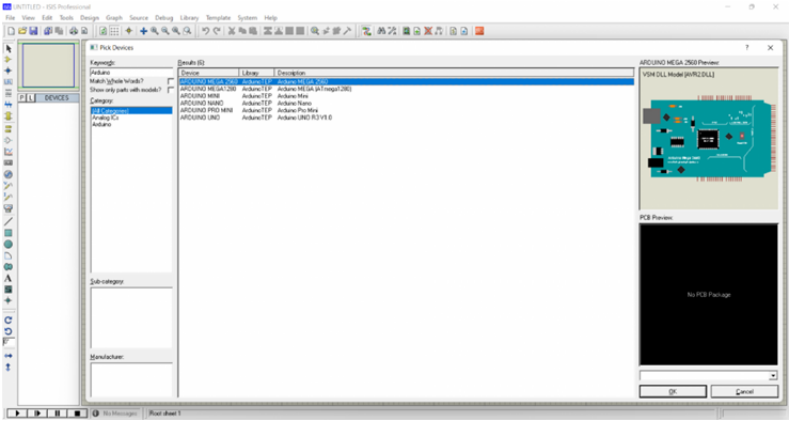

Рис. 13. Режим выбора платформы Arduino

2. Для моделирования схемы с Arduino нужно произвести компиляцию программы, написанной на языке высокого уровня, в так называемый HEX-файл (исполняемый, загружаемый) и загрузить его в микроконтроллер. Для системы моделирования Proteus процедура аналогичная — необходимо получить HEX-файл и загрузить его в модель Arduino.

Для того, чтобы получить HEX-файл для прошивки виртуальной платы Arduino надо скопировать программу Arduino IDE последней версии с любого сайта, дающего к ней доступ.

3. Для освоения технологии загрузки программы в модель Arduino воспользуемся готовым кодом. Для этого используется готовая программа Blink (рис. 14). Ее следует открыть в новом окне (рис. 15).

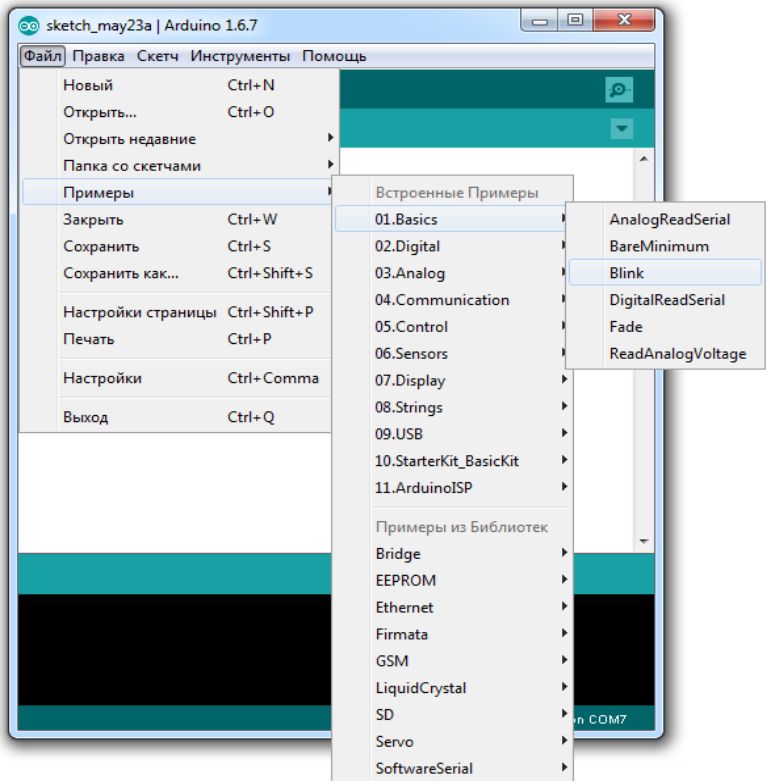

Рис. 14. Arduino IDE, выбор программы «Blink»

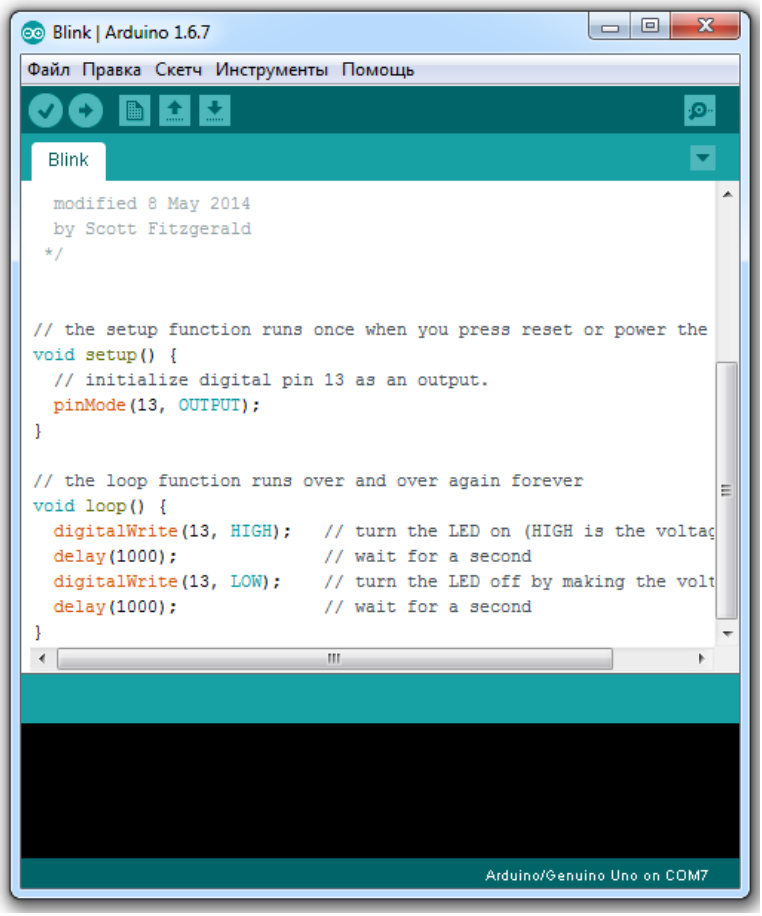

Рис. 15. Скриншот программы «Blink»

4. Затем необходимо скомпилировать код. Для этого в пункте меню «Скетч» нужно выбрать пункт «Проверить/компилировать» (рис. 16).

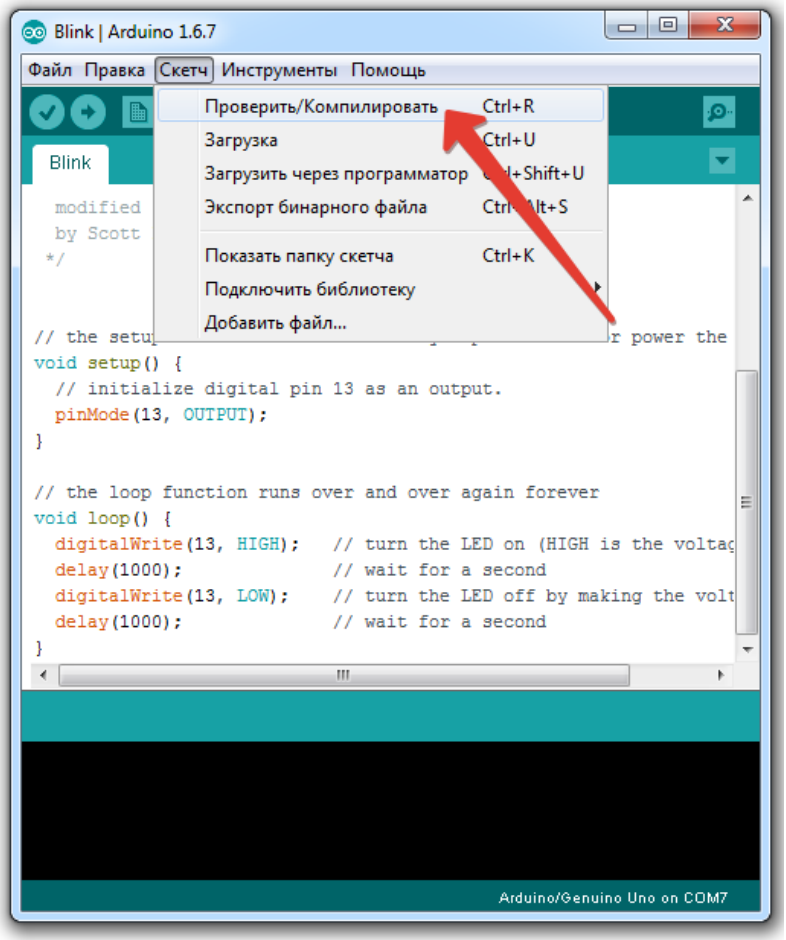

Рис. 16. Arduino IDE, компиляция программы

После удачной компиляции выйдет сообщение, показанное на рисунке 17.

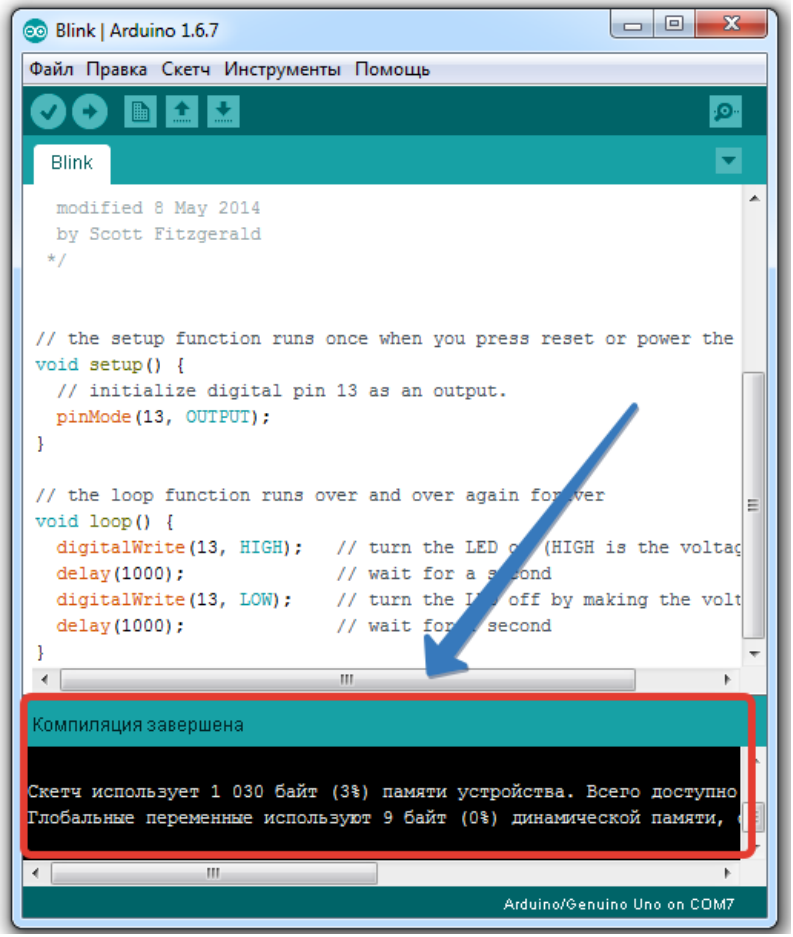

Рис. 17. Сообщение об удачной компиляции

5. После компиляции в пункте меню «Файл» нужно выбрать «Сохранить» (рис. 18). Однако лучше использовать пункт «Сохранить как» и сохранить его в нужную папку, например, в ту, где находятся все примеры.

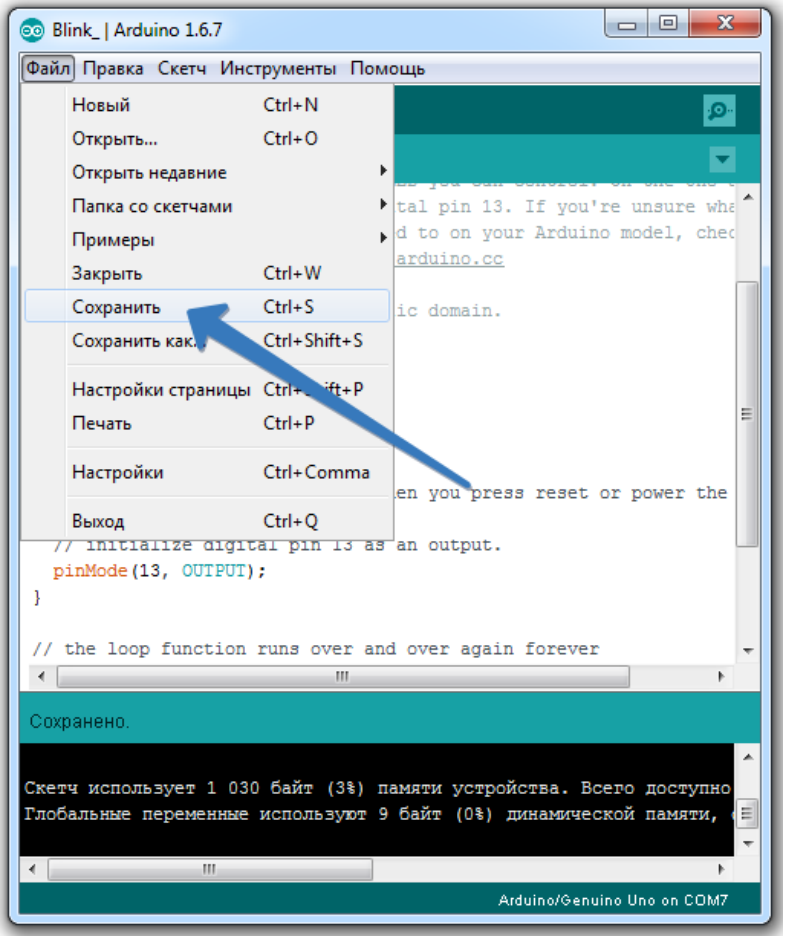

Рис. 18. Пример сохранения полученного файла

6. Следующим шагом в пункте меню «Скетч» нужно выбрать «Экспорт бинарного файла» (рис. 19).

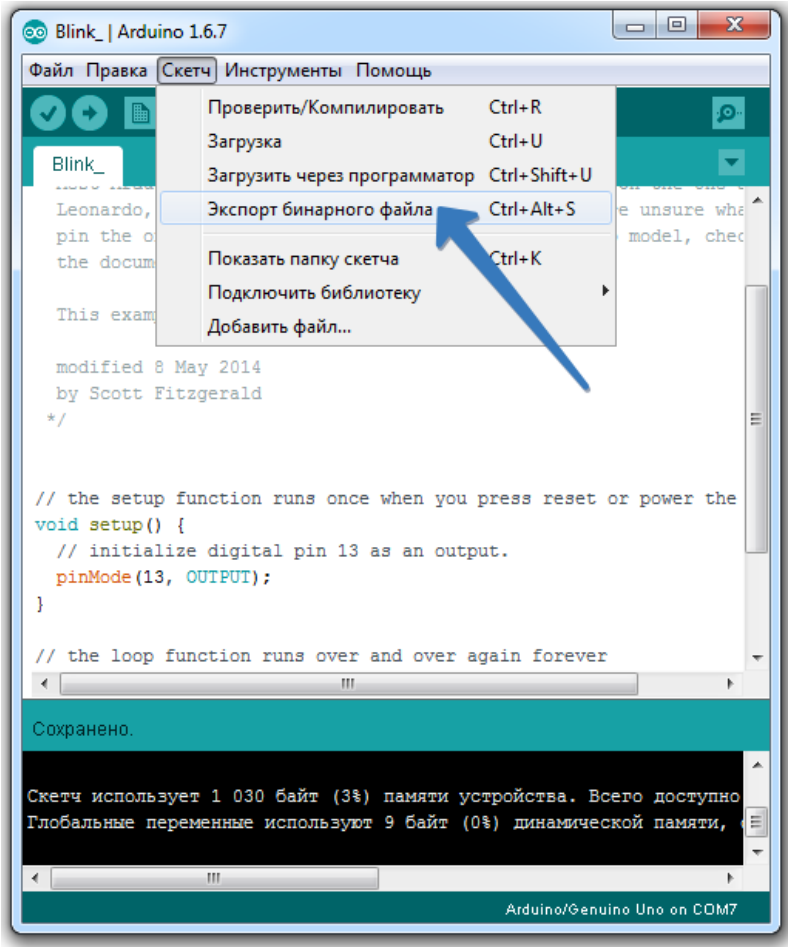

Рис. 19. Пример экспорта бинарного файла

В результате вместе с сохраненным проектом появятся еще два файла. Выбрать для загрузки в Proteus тот, который имеет самое длинное название (рис. 20).

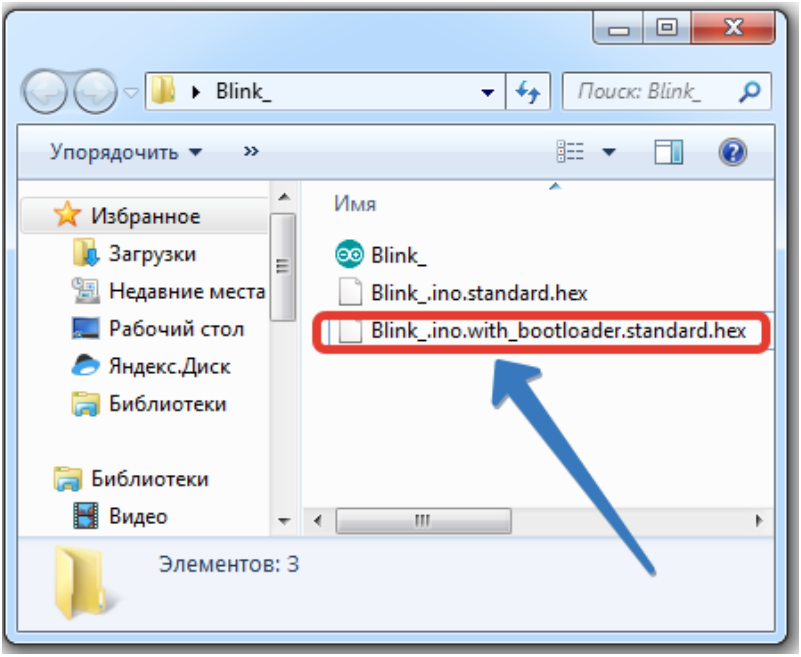

Рис. 20. Выбор файла для Proteus

7. Необходимо открыть Proteus и найти 4 элемента для построения схемы: резистор на 200 Ом, желтый светодиод, землю и саму плату Arduino (рис. 21).

8. Два раза щелкнуть на изображение Arduino на схеме. Должно появиться окно, показанное на рисунке 22. Нажать на значок «папка» и выбрать требуемый HEX-файл.

9. Запустить моделирование схемы (кнопка «Play» внизу слева рабочего окна Proteus) и наблюдать моргание светодиода.

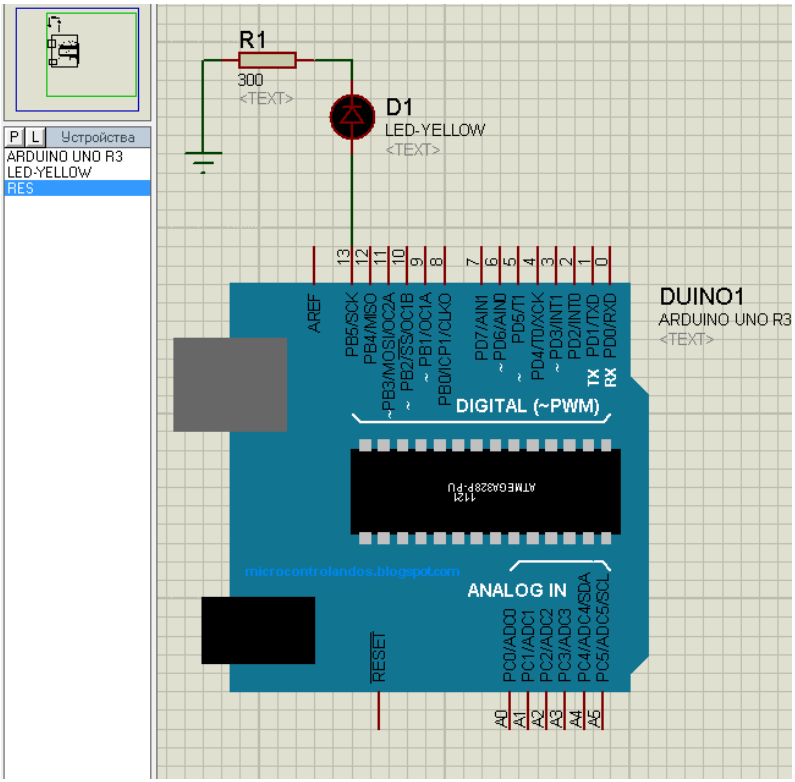

Рис. 21. Схема для проверки разработанной программы
в среде Proteus

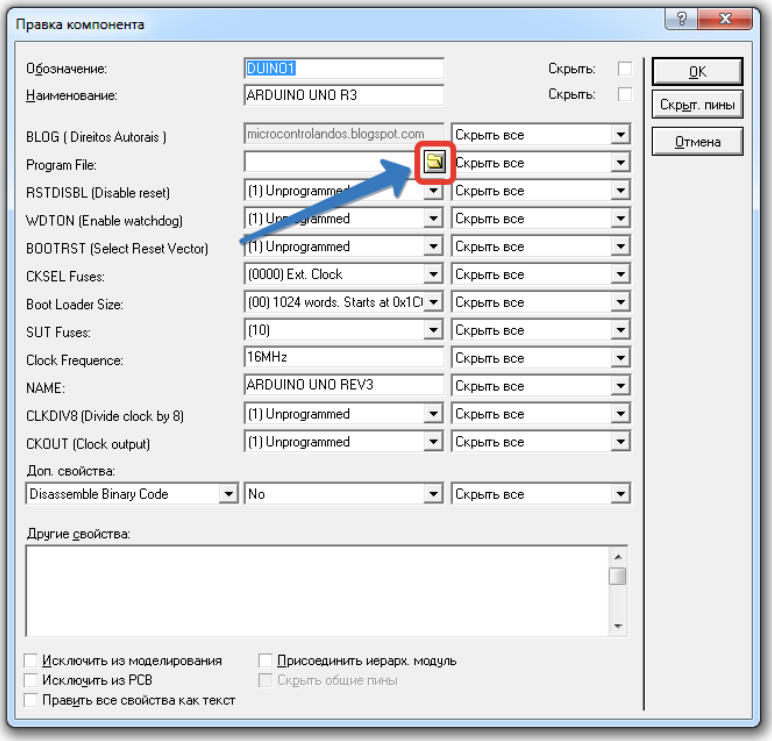

Рис. 22. Выбор HEX-файла

1.3. Содержание отчета

1. Цель работы.
2. Скриншоты экранов приборов.
3. Скриншот схемы.
4. Листинг кода.
5. Выводы.

1.4. Контрольные вопросы

1. Назначение системы Proteus.
2. Назначение управляющих элементов Proteus.
3. Принципы моделирования схем в Proteus.
4. Технология подключения библиотек Arduino.
5. Принципы программирования Arduino в Proteus.

Практическая работа № 3

СХЕМА УПРАВЛЕНИЯ СВЕТОФОРОМ В СРЕДЕ PROTEUS

Цели работы:

— приобрести навыки сборки схем в программе Proteus;

— научиться разрабатывать для микроконтроллера Arduino коды, управляющие огнями светофора и производящие опрос состояния кнопки.

3.1. Краткие сведения

Arduino — это открытая аппаратная платформа, основанная на микроконтроллерах семейства Atmega. Устройство состоит из микропроцессора с памятью, периферийных модулей и собрано на двух интегральных микросхемах.

Плата Arduino оснащена цифровыми и аналоговыми выводами для ввода-вывода данных. Цифровые выводы могут находиться в двух состояниях: логическая единица (высокий уровень) соответствует напряжению в пределах 3—5 В, а логический ноль (низкий уровень) — напряжению от 0 до 1,5 В. Аналоговые выходы, в свою очередь, обеспечивают непрерывное напряжение в диапазоне от 0 до 5 В.

Светодиод (LED) представляет собой полупроводниковый компонент, излучающий свет под действием прямого тока. В данной конфигурации каждый светодиод подсоединяется к отдельному порту Arduino через резистор, ограничивающий ток. Выводы имеют следующее назначение:

— красный светодиод подключается к порту 2;

— желтый светодиод подключается к порту 3;

— зеленый светодиод подключается к порту 4.

Когда на порт 2 подается высокий логический уровень, активируется красный светодиод. Аналогично при подаче сигнала на порт 3 загорается желтый светодиод, а при подаче на порт 4 — зеленый.

Особенности питания. При моделировании в среде Proteus подключение внешнего источника питания к Arduino не является обязательным. Однако при сборке реальной (физической) схемы питание необходимо подавать в обязательном порядке.

Оформление отчетной документации. При подготовке отчета по данной и последующим работам графические схемы алгоритмов должны оформляться в соответствии с ГОСТ 19.505-79 «Единая система программной документации». Пример оформления представлен на рисунке 23.

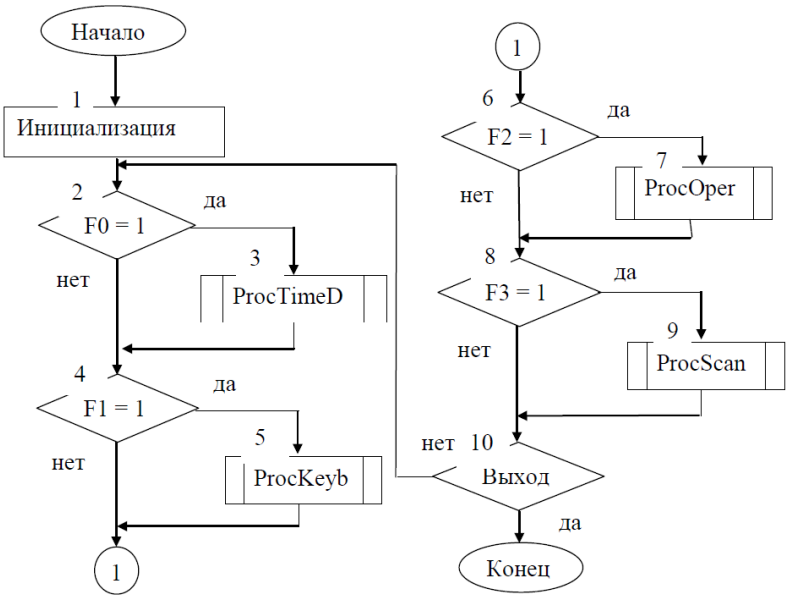

Рис. 23. Пример графической схемы алгоритма

3.2. Порядок выполнения работы

1. Собрать схему управления огнями светофора в Proteus (рис. 24).

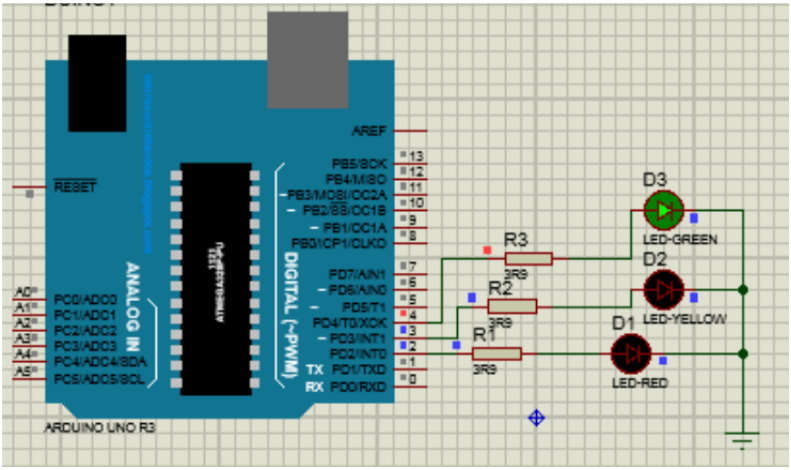

Рис. 24. Схема управления огнями светофора в Proteus

2. В программе Arduino IDE составить листинг кода программы (рис. 25).

```
void setup() {
  // put your setup code here, to run once:
pinMode(2, OUTPUT);
pinMode(3, OUTPUT);
pinMode(4, OUTPUT);
}
void loop() {
digitalWrite (2, HIGH); // красный светодиод загорается
delay(5000); // задержка 5 секунд
digitalWrite (2, LOW); // красный светодиод гаснет
digitalWrite (3, HIGH); // желтый светодиод загорается
delay(5000); // задержка 5 секунд
digitalWrite (3, LOW); // желтый светодиод гаснет
digitalWrite (4, HIGH); // зеленый светодиод загорается
delay(5000); // задержка 5 секунд
digitalWrite (4, LOW); // зеленый светодиод гаснет
}

```

Рис. 25. Скриншот программы управления огнями светофора

3. Произвести компиляцию кода программы, получить HEX-файл, загрузить его в виртуальную модель Arduino в соответствии с методикой, изложенной в предыдущей лабораторной работе.

4. Запустить эмуляцию программы кнопкой «Play».

5. Собрать схему подключения кнопки в Proteus (рис. 26): один свободный проводник кнопки соединить с питанием или землей, другой — с цифровым выводом Arduino.

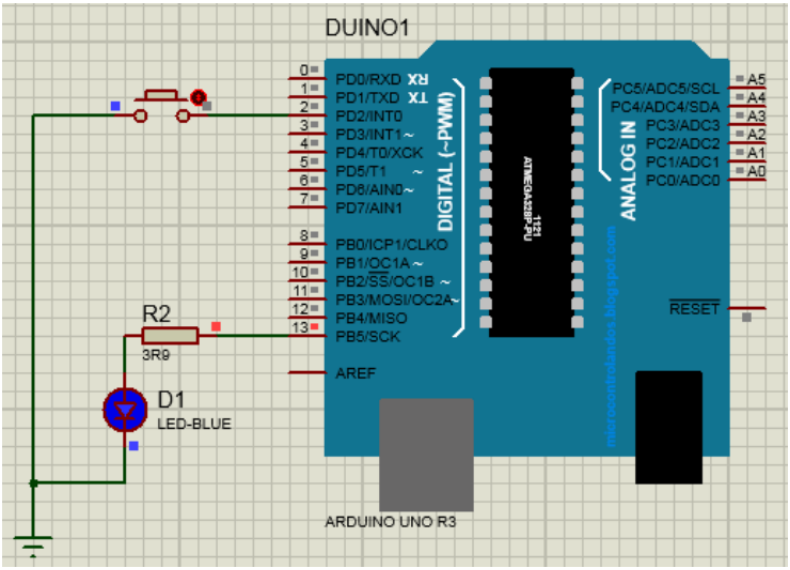

Рис. 26. Схема подключения кнопки в Proteus

В реальной схеме для избежания наводок цифровой вывод обычно подключают через резистор 10 кОм к земле или к питанию. В среде Proteus не требуется подключения подтягивающего или стягивающего резистора. Принцип работы схемы: при нажатии на кнопку светодиод гаснет, при отпускании кнопки — горит.

6. В программе Arduino IDE составить листинг кода программы (рис. 27).

```
const int buttonPin = 2; // пин подключения кнопки 2
const int ledPin = 13; //пин подключения светодиода 13
// переменные будут меняться
int buttonState = 0; // переменная для чтения статуса кнопки
void setup() {
// конфигурация вывода 13 на выход
pinMode(ledPin, OUTPUT);
// конфигурация вывода 2 на вход
pinMode(buttonPin, INPUT);
}
void loop() {
// чтение состояния значения кнопки:
buttonState = digitalRead(buttonPin);
// кнопка нажата - buttonState is HIGH:
if (buttonState == HIGH) {
// включить светодиод:
digitalWrite(ledPin, HIGH);
// нет:
} else {
// выключить светодиод:
digitalWrite(ledPin, LOW);
}
}

```

Рис. 27. Листинг кода опроса состояния кнопки

7. Произвести компиляцию кода программы, получить HEX-файл, загрузить его в виртуальную модель.

8. Запустить эмуляцию программы.

3.3. Содержание отчета

1. Цель работы.
2. Скриншоты схем светофора и подключения кнопки.
3. Графическая схема алгоритма программы.
4. Листинги кодов. Комментарии к каждой команде.
5. Выводы.

3.4. Контрольные вопросы

1. Как выглядит структура скетча Arduino?
2. Зачем объявляются типы переменных и данных?
3. Как осуществляется конфигурация портов ввода-вывода?
4. Как изменить длительность горения ламп светофора?
5. Как изменить схему и программу, чтобы предусмотреть кнопку включения?

Практическая работа № 4

СХЕМА УПРАВЛЕНИЯ ШАГОВЫМ ДВИГАТЕЛЕМ В СРЕДЕ PROTEUS

Цель работы:

— приобретение навыков работы в программе Proteus;

— создание схемы управления шаговым двигателем;

— разработка соответствующего кода для микроконтроллера платы Arduino Uno.

4.1. Краткие сведения

Шаговый привод представляет собой электромеханический аппарат, предназначенный для перемещения своего выходного вала. Его движение осуществляется путем дискретных вращательных перемещений, каждое из которых соответствует определенному числу шагов, а направление вращения контролируется программным обеспечением микроконтроллера. Указанные двигатели нашли широкое применение в различных областях, включая конструирование роботов, работу периферийных печатающих устройств, создание управляемых манипуляторов, функционирование станков с ЧПУ, а также в составе множества других электронных устройств и систем.

Главное преимущество шаговых двигателей по сравнению с моторами постоянного вращения заключается в способности обеспечивать высокоточное угловое позиционирование ротора. Кроме того, шаговые моторы позволяют быстро осуществлять пуск, мгновенную остановку и реверс (изменение направления вращения).

Драйвер — это промежуточное устройство, которое осуществляет сопряжение между управляющим контроллером и шаговым двигателем, преобразуя управляющие сигналы в токи, необходимые для работы обмоток.

Принцип работы. В конструкции шагового двигателя ротор содержит постоянные магниты, а на статоре размещены несколько обмоток (катушек). Когда микроконтроллер через драйвер подает электрические импульсы, внутри двигателя формируется вращающееся магнитное поле. Оно воздействует на магниты ротора, заставляя вал поворачиваться таким образом, что каждый импульс вызывает смещение вала ровно на один дискретный шаг.

4.2. Порядок выполнения работы

1. Собрать схему управления шаговым двигателем в Proteus (рис. 28).

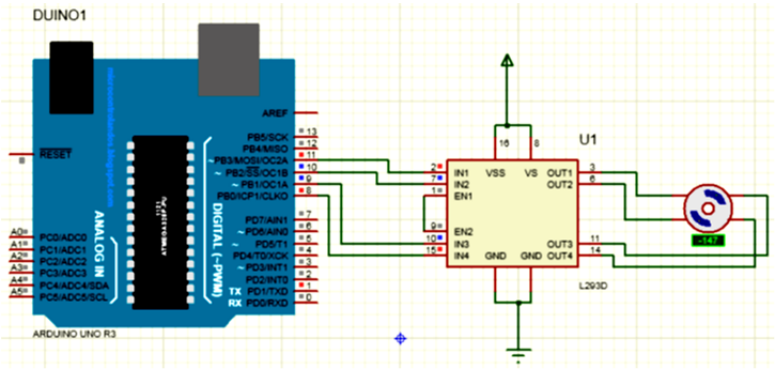

Рис. 28. Схема управления шаговым двигателем в Proteus

2. В программе Arduino IDE составить листинг кода программы (рис. 29). Для управления шаговыми двигателями в Arduino IDE есть стандартная библиотека (Stepper.h), которая осуществляет полношаговый режим коммутации.

3. Произвести компиляцию кода программы, получить HEX-файл, загрузить его в виртуальную модель.

4. Запустить эмуляцию программы.

```
#include <Stepper.h>
const int stepsPerRevolution = 200; // количество шагов за оборот
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11); // подключение к пинам 8-11
void setup() {
myStepper.setSpeed(60); // установка скорости вращения ротора
Serial.begin(9600); // инициализация последовательного порта
}
void loop() { // Функция ожидает, пока поступит команда, преобразовывает текст
  // и подает сигнал на двигатель для его вращения на указанное число шагов
Serial.println("clockwise"); // по часовой стрелке
myStepper.step(stepsPerRevolution);
delay(500);
Serial.println("counterclockwise"); // против часовой стрелки
myStepper.step(-stepsPerRevolution);
delay(500);
}
```

Рис. 29. Листинг кода программы

4.3. Содержание отчета

1. Цель работы.
2. Скриншот схемы управления в Proteus.
3. Графическая схема алгоритма программы.
4. Листинг кода. Комментарии к каждой команде.
5. Выводы.

4.4. Контрольные вопросы

1. Какую функцию выполняет драйвер шагового двигателя L293D?
2. Какая последовательность управления шаговым двигателем называется полушаговой?
3. Почему один из режимов работы шагового двигателя называется полношаговым?
4. Как осуществить реверс шагового двигателя?
5. Какие преимущества есть у шагового двигателя?

Практическая работа № 5

СХЕМА УПРАВЛЕНИЯ LCD В СРЕДЕ PROTEUS

Цель работы:

— моделирование LCD в программе Proteus;

— создание кода управления дисплеем для микроконтроллера платы Arduino Uno.

5.1. Краткие сведения

LCD, или жидкокристаллический дисплей, представляет собой прибор, задача которого — графически демонстрировать актуальные данные. Эти устройства бывают двух основных типов: способные отображать символы и работающие с графикой.

Различают следующие разновидности символьных жидкокристаллических дисплеев, исходя из габаритов рабочей поверхности:

- 16 × 2—16 колонок и 2 строки символов;
- 20 × 4—20 колонок и 4 строки символов.

Назначение выводов LCD. Индикатор оснащается 16 выводами. Вывод 1 (VSS) служит для подключения минусового питания контроллера (общий провод). Вывод 2 (VDD) предназначен для подачи плюсового питания на контроллер. Вывод 3 (VO) отвечает за регулировку контрастности изображения.

Вывод 4 (RS) используется для выбора регистра — команды или данных. Вывод 5 (R/W) управляет режимами чтения и записи; если соединить его с «землей», устройство переходит в режим записи. Вывод 6 (E) — это строб активации (Enable).

Выводы с 7 по 10 (DB0—DB3) представляют собой младшие разряды 8-битного интерфейса данных. Выводы с 11 по 14 (DB4—DB7) — старшие разряды того же интерфейса.

Вывод 15 (A) — это анод (положительный контакт) для питания подсветки дисплея. Вывод 16 (K) — катод (отрицательный контакт) для питания подсветки.

5.2. Порядок выполнения работы

1. Собрать схему в среде Proteus (рис. 30).

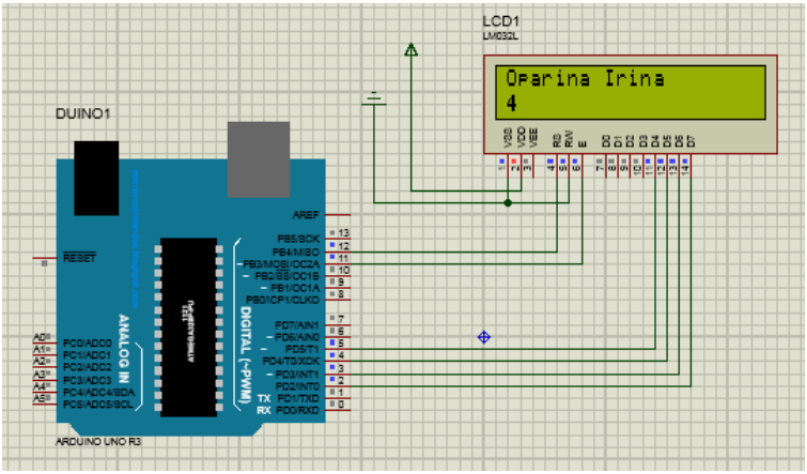

Рис. 30. Собранная схема в Proteus

2. В программе Arduino IDE составить листинг кода программы (рис. 31).

3. Проверить наличие библиотеки LiquidCrystal.h в папке Arduino/libraries. В случае отсутствия названной библиотеки скачать ее с сайта https://github.com и установить через пункт меню Arduino IDE «Скетч/Подключить библиотеку».

3. Произвести компиляцию кода программы, получить HEX-файл, загрузить его в виртуальную модель.

4. Запустить эмуляцию программы.

```
#include <LiquidCrystal.h>
// инициализация библиотеки номерами пинов интерфейса
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup() {
// настройка размерности LCD:
lcd.begin(16, 2);
}
void loop() {
// Включение дисплея:
lcd.setCursor(0,0); //1 столбец, 1 строка
lcd.print("Oparina Irina"); //вывод текста
lcd.setCursor(0,1); //1 столбец, 2 строка
lcd.print(millis ()/1000); //вывод скорости
}
```

Рис. 31. Листинг кода программы

5.3. Содержание отчета

1. Цель работы.
2. Скриншот схемы в среде Proteus.
3. Графическая схема алгоритма программы.
4. Листинг кода. Комментарии к каждой команде.
5. Выводы.

5.4. Контрольные вопросы

1. Как выделяются комментарии в коде Arduino?
2. Почему вывод R/W LCD заземляется?
3. Почему вывод VSS LCD заземляется?
4. Каким числом заканчивается счет?
5. Какой тип LCD применяется в данной программе?

Практическая работа № 6

МОДЕЛИРОВАНИЕ ЦИФРОВОГО ВОЛЬТМЕТРА В СРЕДЕ PROTEUS

Цель работы:

— освоить принцип работы и нюансы разработки цифрового вольтметра в симуляторе Proteus.

6.1. Краткая информация

Вольтметр является измерительным инструментом, который определяет величину напряжения или электродвижущей силы в электрических цепях. Для работы он подключается параллельно элементу, потребляющему энергию, или источнику этой энергии.

Теоретически для идеального вольтметра требуется бесконечно высокое внутреннее сопротивление. На практике это означает, что чем больше внутреннее сопротивление реального прибора, тем меньше оно влияет на работу исследуемой цепи, что, в свою очередь, увеличивает точность получаемых измерений.

6.2. Порядок выполнения задания

1. Воспроизвести в Proteus схему цифрового вольтметра, представленную на рисунке 32. Установить для синусоидального генератора амплитуду 50 В, а частоту — 0,01 Гц. При изменении напряжения, выдаваемого источником V1 (VSINE), на дисплее LCD будет отображаться измеренное значение напряжения.

2. В программе Arduino IDE составить листинг кода программы (рис. 33).

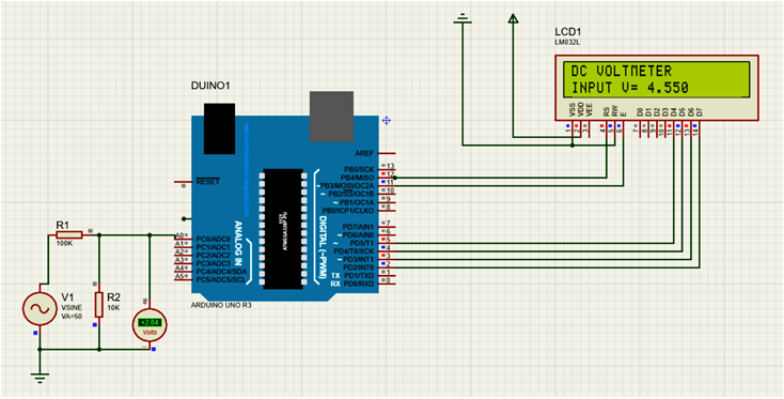

Рис. 32. Схема модели вольтметра в Proteus

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
int analogInput = 0;
float vout = 0.0;
float vin = 0.0;
float R1 = 100000.0; // сопротивление R1 (100K)
float R2 = 10000.0; // сопротивление R2 (10K)
int value = 0;
void setup(){
pinMode(analogInput, INPUT);
lcd.begin(16, 2);
lcd.print("DC VOLTMETER");
}
void loop(){
// считывание аналогового значения
value = analogRead(analogInput);
vout = (value * 5.0) / 1024.0;
vin = vout / (R2/(R1+R2));
if (vin<0.09) {
vin=0.0;// обнуляем нежелательное значение
}
lcd.setCursor(0, 1);
lcd.print("INPUT V= ");
lcd.print(vin);
delay(500);
}
```

Рис. 33. Листинг кода программы

3. Произвести компиляцию кода программы, получить HEX-файл, загрузить его в виртуальную модель.

4. Запустить эмуляцию программы.

5. Настроить коэффициенты в программе таким образом, чтобы показания вольтметра и LCD-индикатора совпадали.

6.3. Содержание отчета

1. Цель работы.
2. Скриншот схемы в среде Proteus.
3. Графическая схема алгоритма программы.
4. Листинг кода. Комментарии к каждой команде.
5. Выводы.

6.4. Контрольные вопросы

1. Как работает АЦП?
2. От чего зависит точность измерения цифрового вольтметра?
3. Какую функцию в схеме выполняют резисторы R1 и R2?
4. С какого значения напряжения начинается измерение?
5. Какой тип LCD применяется в данной программе?

Практическая работа № 7

МОДЕЛИРОВАНИЕ ЧАСОВ И СЕКУНДОМЕРА В СРЕДЕ PROTEUS

Цель работы:

— изучить принцип действия и особенности проектирования устройств, работающих в режиме реального времени в среде Proteus.

7.1. Краткие сведения

В работе моделируются простые часы и секундомер на базе одноплатного компьютера Arduino Uno. Секундомер можно также реализовать в качестве таймера и использовать для измерения промежутков времени при разработке таких устройств, как реле времени, автоматические выключатели и т. д. Секундомер измеряет промежуток времени между нажатиями кнопок START и STOP. Замыкать контакты может и какое-либо внешнее устройство, если использовать контакты реле или транзисторные ключи, подключенные к соответствующим к выводам компьютера.

Для реализации модели часов необходима библиотека «Time». Если она еще не установлена, можно скачать архив по ссылке https://github.com. Подключение происходит следующим образом:

```
#include <Time.h>
```

Затем устанавливаются текущие дата и время с помощью функции «setTime»:

```
setTime(23, 59, 59, 12, 04, 2026).
```

Цифры показывают часы, минуты, секунды, месяц, число, год соответственно.

Для вывода даты используются функции:

— year() — позволяют отредактировать год;

— month() — месяц;

— day() — день;

— hour() — часы;

— minute() — минуты;

— second() — секунды.

Обратите внимание! Если посчитать количество символов в типовой записи даты: «12.04.2026 23:59:59», получим 19, в то время как в одну строчку вмещается всего 16. Решить проблему можно еще одной полезной функцией — «setCursor». Она устанавливает курсор в нужную позицию. Например, lcd.setCursor(0,1); — установит его в начало второй строчки.

Курсор — это место символа, с которого начнется вывод текста командой «print». Воспользуемся этой функцией для вывода даты в первой строчке, а времени — во второй.

После каждого заполнения дисплея, необходимо его чистить функцией «clear()»:

```
lcd.clear();
```

Нет смысла выводить данные на дисплей чаще чем раз в секунду, поэтому между двумя итерациями устанавливается пауза в 1000 миллисекунд.

Таким образом, получается требуемая программа.

7.2. Порядок выполнения работы

1. Собрать схему модели часов в программе Proteus (рис. 34).

2. В программе Arduino IDE составить листинг кода программы (рис. 35).

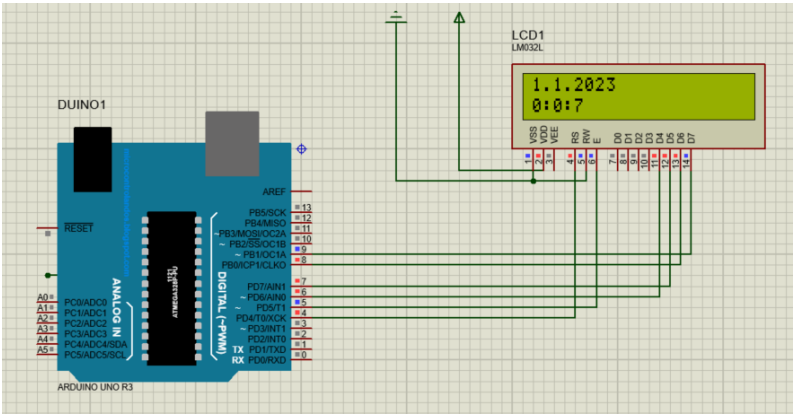

Рис. 34. Схема «Часы» в Proteus

```
#include <TimeLib.h>

#include <Time.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(4, 5, 6, 7, 8, 9);

void setup(){
    lcd.begin(16, 2);
    setTime(0,0,0,1,01,2023); // 0 утра, первого января 2023 года
}

void loop(){
    lcd.clear();
    lcd.print( day() );
    lcd.print( "." );
    lcd.print( month() );
    lcd.print( "." );
    lcd.print( year() );

    lcd.setCursor(0, 1);
    lcd.print( hour() );
    lcd.print( ":" );
    lcd.print( minute() );
    lcd.print( ":" );
    lcd.print( second() );

    delay(1000);
}
```

Рис. 35. Листинг программы «Часы»

3. Произвести компиляцию кода программы, получить HEX-файл, загрузить его в виртуальную модель.

4. Запустить эмуляцию программы.

5. Сравнить показания модели часов в Proteus и реальных часов. В случае ошибочной работы модели определить величину расхождения результатов и произвести коррекцию программы.

6. Собрать схему модели секундомера в Proteus, приведенную на рисунке 36. Помимо платы Arduino Uno, потребуется несколько дополнительных компонентов: двухстрочный шестнадцатибуквенный LCD-дисплей LM032L, два резистора по 10 кОм, три нормально-разомкнутые кнопки. Запуск и остановка таймера осуществляется двумя кнопками.

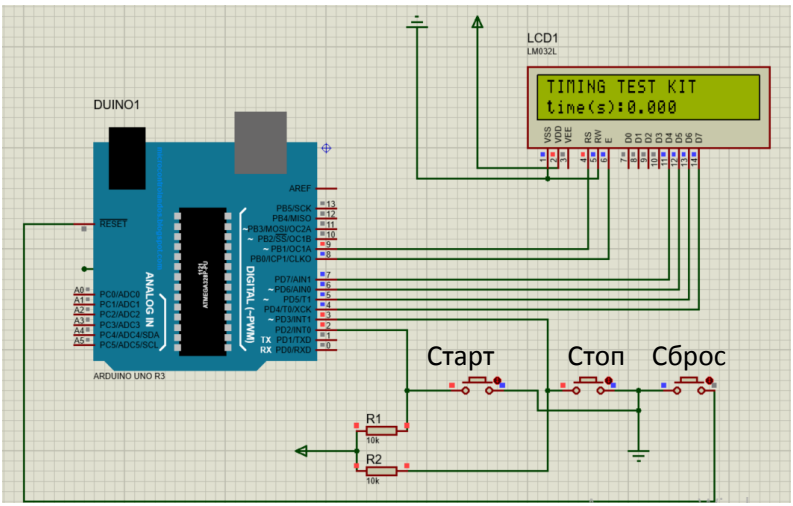

Рис. 36. Принципиальная схема модели секундомера в среде Proteus

7. В программе Arduino IDE составить листинг кода программы (рис. 37).

```
#include<LiquidCrystal.h>

// digital pins 9 - RS, 8 - Enable, 7,6,5,4 - D4,D5,D

LiquidCrystal lcd(9, 8, 7, 6, 5, 4 );

unsigned long previousmillis = 0;
float duration = 0;

void setup() {

lcd.begin(16, 2);
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("TIMING TEST KIT");
pinMode(2, INPUT);
pinMode(3, INPUT);
attachInterrupt(0, count, CHANGE);
attachInterrupt(1, count1, CHANGE );
}

void loop() {
lcd.setCursor(0, 1);
lcd.print("time(s):");
lcd.print(duration/1000,3);
Serial.print("time(msec)=");
Serial.println(duration);
delay(1000);
}

void count() {
previousmillis = millis();
}

void count1() {
duration = (millis() - previousmillis);
}
```

Рис. 37. Листинг программы «Секундомер»

8. Произвести компиляцию кода программы, получить HEX-файл, загрузить его в виртуальную модель.

9. Для начала эксплуатации программного обеспечения следует запустить его эмуляцию. Непосредственно перед началом замеров времени нажимается клавиша «Сброс». На экране возникнет первая строка с текстом «Timing test kit» и вторая строка, отображающая «Time(s): 0.000». Затем необходимо инициировать отсчет, нажав кнопку «Старт». Следует учесть, что сам отсчет в этот момент не будет виден на экране. Для прерывания процесса подсчета времени в любой момент можно нажать кнопку «Стоп». Результат, полученный в ходе подсчета, будет продемонстрирован на дисплее.

10. Требуется доработать программное обеспечение следующим образом. В текущей реализации время демонстрируется лишь единожды, после нажатия кнопки «Стоп». Необходимо добиться того, чтобы при нажатии кнопки «Пуск» происходило непрерывное отображение текущего временного показателя, а при нажатии кнопки «Стоп» процесс фиксации времени прекращался.

7.3. Содержание отчета

1. Цель работы.
2. Скриншот схемы в среде Proteus.
3. Графическая схема алгоритма программы.
4. Листинг кода. Комментарии к каждой команде.
5. Выводы.

7.4. Контрольные вопросы

1. Каково назначение библиотек?
2. Чем отличаются библиотеки от драйверов?
3. Какие библиотеки используются в программах настоящей работы?
4. Перечислите функции команды «print».
5. Перечислите выводы дисплея и их назначение.

Практическая работа № 8

МОДЕЛИРОВАНИЕ ДАТЧИКА ТЕМПЕРАТУРЫ И ВЛАЖНОСТИ DHT11 В СРЕДЕ PROTEUS

Цель работы:

— создание модели схемы цифрового термометра и измерителя влажности в среде Proteus;

— создание и отладка программы в среде Arduino IDE.

8.1. Краткие сведения

DHT11 представляет собой современное цифровое устройство, интегрирующее в себе термистор и сенсор влажности емкостного типа. Он функционирует при напряжении от 3,5 до 5 В. Диапазон измерения температуры составляет от 0 до +50 °С с погрешностью в 2 °С, а определение уровня влажности осуществляется в пределах от 20 до 95 % с отклонением в 5 %.

Термистор по своей сути является резистором, чувствительным к температурным колебаниям. Его электрическое сопротивление напрямую зависит от температуры окружающей среды: при ее повышении сопротивление термистора закономерно снижается.

Емкостной датчик влажности функционирует благодаря изменению параметров конденсатора. Изменения влажности воздуха воздействуют на диэлектрические свойства слоя, покрывающего герметичный корпус конденсатора. Влага, абсорбируемая этим слоем, изменяет его диэлектрическую проницаемость, что, в свою очередь, вызывает колебания в емкости устройства.

8.2. Порядок выполнения работы

1. Собрать схему, приведенную на рисунке 38.

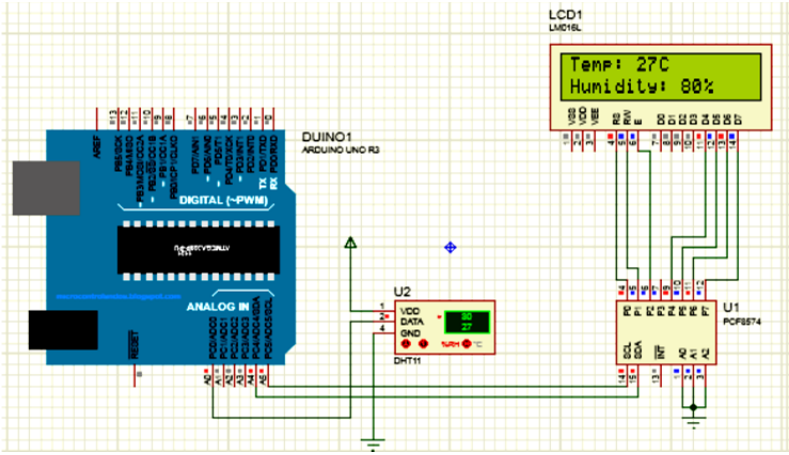

Рис. 38. Схема цифрового термометра в Proteus

2. В программе Arduino IDE составить листинг кода программы (рис. 39).

```
// подключение библиотек
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "DHT.h"
#define DHTPIN A0 // директива, которая позволяет дать имя константе
//перед тем как программа будет скомпилирована
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE); // создание и обозначение объекта "dht"
LiquidCrystal_I2C lcd(0x20,16,2); // адрес и размерность lcd
void setup()
{
lcd.begin(16, 2); // на дисплее будет 16 столбцов и 2 строчки
dht.begin(); // подготовка датчика DHT11 к работе
}
void loop() {
int h = dht.readHumidity();
int t = dht.readTemperature(); // ввод переменных "h" и "t" ,
//в которые будет считываться значение с датчика DHT11
```

Рис. 39. Листинг кода программы
(начало, окончание на с. 67)

```
19 lcd.setCursor(0,0); // 1 столбец, 1 строка
20 lcd.print("Temp: "); // вывод текста
21 lcd.print(t); //
22 lcd.print("C");
23 lcd.setCursor(0,1); // 1 столбец, 2 строка
24 lcd.print("Humidity: "); // вывод текста
25 lcd.print(h);
26 lcd.print("%");
27 delay (200); // ожидание
28 }
```

Рис. 39. Листинг кода программы
(окончание, начало на с. 66)

3. Произвести компиляцию кода программы, получить HEX-файл, загрузить его в виртуальную модель.
4. Запустить эмуляцию программы.

8.3. Содержание отчета

1. Цель работы.
2. Скриншот схемы в среде Proteus.
3. Графическая схема алгоритма программы.
4. Листинг кода. Комментарии к каждой команде.
5. Выводы.

8.4. Контрольные вопросы

1. Как работает термистор?
2. Как измеряется влажность среды?
3. Каким числом заканчивается измерение влажности?
4. Каким числом заканчивается измерение температуры?
5. Какой тип LCD применяется в данной программе?

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. *PROTEUS* по-русски // Радиоежегодник. Вып. 24. 2013. URL: https://www.rlocman.ru/book/book.html?di=148418&ysclid=mo14fys5f2890257356 (дата обращения: 16.04.2026).

2. *Байкенов Б. С., Аязбай А. Е.* Микропроцессорные системы управления и контроля : метод. указания по выполнению лабораторных работ для студентов специальности 5В071600 «Приборостроение». Алматы, 2019.

3. *Петин В. А.* Проекты с использованием контроллера Arduino. СПб., 2016.

4. *Филатов М.* Проектирование схем электрических принципиальных с использованием микроконтроллеров в программной среде Proteus 8.1 // Компоненты и технологии. 2015. № 7. С. 101—110.

5. *Proteus* design suite. Getting Started Guide. 2014. URL: https://labcenter.s3.amazonaws.com/downloads/Tutorials.pdf (дата обращения: 16.04.2026).

6. *Болдырев А. В.* Моделирование устройств интерфейсов в среде Proteus : практикум. Ростов н/Д, 2019.

7. *Смирнов В. И.* Проектирование и схемотехническое моделирование микропроцессорных устройств : учеб. пособие для студентов, обучающихся по направлению 211000 «Конструирование и технология электронных средств». Ульяновск, 2013.

8. *Иоффе В. Г.* Разработка и отладка микропроцессорных устройств в виртуальной среде моделирования : метод. указания. Самара, 2017.

Чижма Сергей Николаевич
Молчанов Сергей Васильевич

МОДЕЛИРОВАНИЕ МИКРОКОНТРОЛЛЕРОВ В СРЕДЕ PROTEUS

Учебно-методическое пособие

Учебное электронное издание

Редактор *О. И. Бессчастнова*
Компьютерная верстка *Г. И. Винокуровой*

Дата выхода в свет 04.06.2026 г.
Формат 60×90 $^{1}/_{16}$. Усл. печ. л. 4,3