

Направление подготовки: 140400 Электроэнергетика и электротехника

Профиль подготовки: Электрический транспорт

Квалификация (степень) выпускника: магистр

Форма обучения: очная

Дисциплина:

"Информационные и компьютерные технологии в электротехнике"

Комаров В.Г. Лабораторная работа 6 13.10.2026 г.

Методические указания

Тема: Широтно-импульсная модуляция с помощью таймеров микроконтроллера

Цель: Изучить работу таймеров-счётчиков общего назначения в режиме широтно-импульсной модуляции и освоить моделирование ШИМ-преобразователя постоянного тока в ИСР Proteus для полного цикла разработки микроконтроллерных устройств и систем.

Задание: разработать принципиальную электрическую схему и программу микроконтроллерного устройства ШИМ-преобразователя постоянного тока и промоделировать его работу с помощью симуляции в ИСР Proteus.

Общие сведения

Proteus (Протеус) - это универсальная интегральная среда разработки (ИСР) , с помощью которой можно создавать различные виртуальные электронные устройства, выполнять их симуляцию и проводить полный цикл разработки всей конструкторско-технологической документации на эти устройства. Она содержит огромную библиотеку аналоговых и цифровых микросхем, датчиков, дискретных элементов: резисторов, конденсаторов, диодов, транзисторов и т.п. Также имеется широкий набор компонентов оптоэлектроники: дисплеи, светодиоды, оптопары и др.

Главным преимуществом и отличием Proteus от других подобных программ для симуляции работы электрических цепей, — это возможность выполнять симуляцию работы микропроцессоров и микроконтроллеров (МК) совместно с внешними электрическими цепями. Библиотека Proteus содержит такие основные типы МК, как AVR, ARM, PIC, Cortex.

Как и в любом другом аналогичном ПО, предназначенном для симуляции работы электрических цепей, данный софт имеет ряд виртуальных измерительных приборов: амперметры, вольтметры, ваттметр, осциллограф, логический анализатор, счетчик и т.п.

Также в Протеусе встроены инструменты для автоматизированной разработки печатных плат и для создания их 3D моделей.

Начало работы с Proteus

Любая работа начинается с запуска программы либо из меню установленных на компьютере программ, либо с помощью значка программы на рабочем столе. После запуска программы открывается стартовое окно (рис. 1), в котором можно создать новый проект или открыть уже существующий.

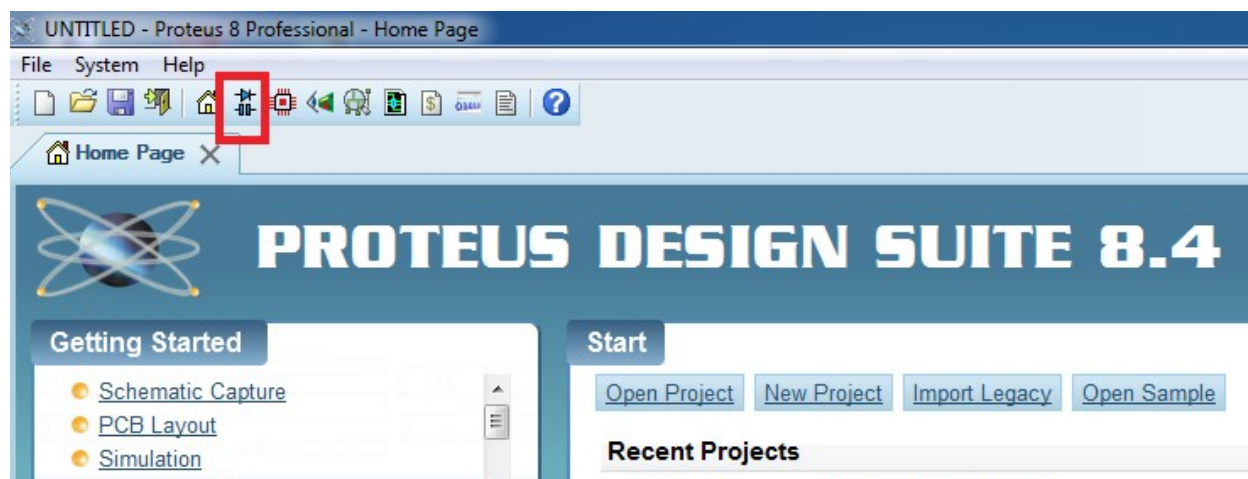


Рис. 1.

Проектирование электрической схемы

Проект электрической принципиальной схемы можно создать при помощи редактора ISIS кнопкой **Schematic Capture** верхней панели инструментов **Proteus** (см. рис. 1). В результате будет открыта новая вкладка **Schematic Capture**, в рабочем поле которой и выполняется разработка схемы.

Выбор компонентов из базы данных для последующего их размещения в рабочей области программы производится в окне **Pick Devices** (рис. 2). Данное окно можно открыть командой контекстного меню с помощью правой кнопки мыши **Place/Component/From Libraries** или нажатием на кнопку **P** на панели **DEVICES** (по умолчанию данная панель расположена в левой части экрана и содержит список имеющихся в проекте компонентов). Открыть эту панель можно кнопкой **Component Mode** на левой панели инструментов редактора ISIS.

Для того чтобы добавить микросхему микроконтроллера в рабочее поле проекта, необходимо в левой верхней части окна **Pick Devices** в поле **Category** выбрать из списка библиотеку **Microprocessor ICs**. Пакет **Microprocessor ICs** позволяет включать в эмуляцию смешанной схемы определенные микроконтроллеры с возможностью написания и отладки программного кода. Выбор библиотеки из списка выполняется щелчком левой кнопки мыши по строке с ее названием. Ниже поля **Category** находится поле **Sub-category**, в котором таким же способом задается семейство микроконтроллеров выбранной библиотеки. В поле **Results** отображаются все компоненты выбранного семейства. Выбор компонента производится посредством выделения при помощи левой кнопки мыши строки с его названием в поле **Results**. В поле **Manufacturer** можно выбрать производителя микроконтроллера. Если производитель не имеет значения — выберите значение **All Manufacturers** в этом поле. Для ускорения поиска компонентов можно воспользоваться строкой фильтра **Keywords**, расположенной в верхнем левом углу окна **Pick Devices**. После того как выбор микроконтроллера произведен, его условное графическое обозначение отобразится в поле предварительного просмотра **Preview**. Посадочное место компонента будет показано в поле **PCB Preview**. Если для выбранного микроконтроллера доступно

несколько посадочных мест, то все возможные варианты будут доступны для выбора из выпадающего меню, которое находится под полем PCB Preview.

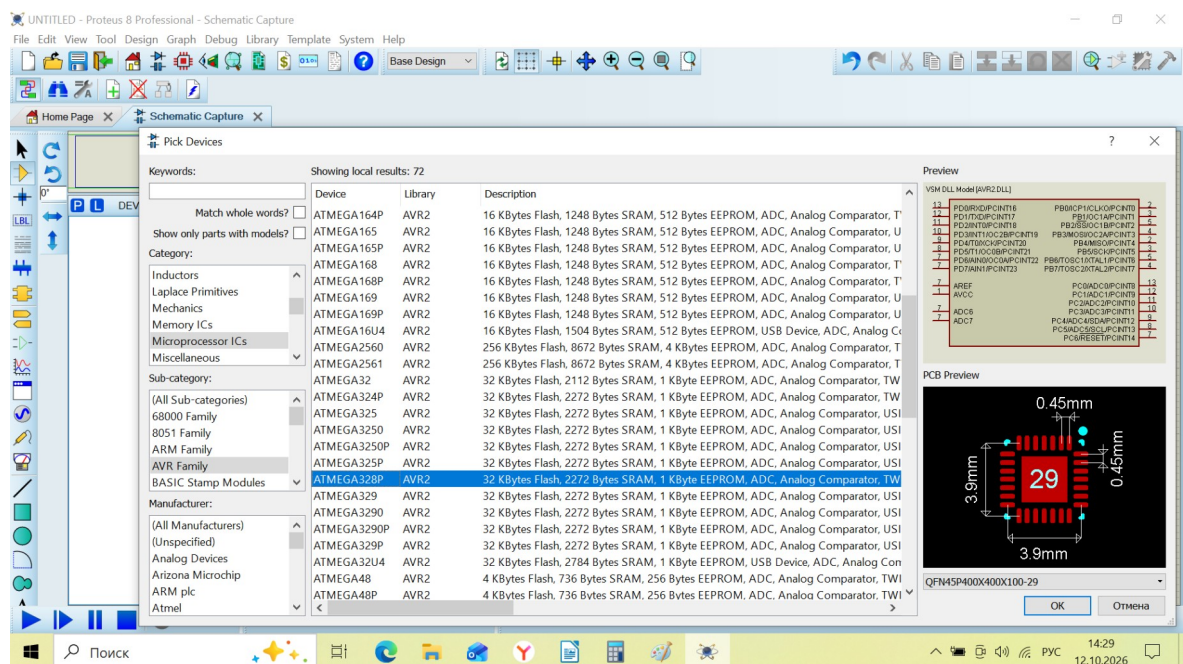


Рис. 2. Библиотека микроконтроллеров семейства AVR программы Proteus

Для того чтобы разместить выбранный микроконтроллер на схеме, необходимо в окне **Pick Devices** нажать на кнопку **OK**. После чего данное окно будет закрыто, а символ компонента будет прикреплен к курсору мыши, при помощи которого необходимо поместить символ в нужное место на схеме (щелкнуть в конкретном месте схемы левой кнопкой мыши).

Кварцевый резонатор, конденсаторы и элементы питания в данном случае в схему можно не добавлять, так как они эмулируются программно. Однако если вы будете разрабатывать проект до его логического конца, то есть до разработки конструкции и изготовления печатной платы, элементы придется добавить.

Обратите внимание, что выводы микроконтроллера в Proteus для удобства объединены в отдельные группы по портам. Однако это не соответствует порядку расположению их в реальном корпусе МК. Кроме того отсутствуют выводы, к которым подается напряжение для питания МК. Эта функция установлена по умолчанию.

Параметры размещенного на схеме микроконтроллера при необходимости можно редактировать в окне **Edit Component** (рис. 3). Данное окно открывается путем двойного щелчка левой кнопки мыши по уже размещенному в рабочем поле программы символу компонента. Для того чтобы добавить необходимые компоненты в рабочее поле проекта, необходимо вызвать окно **Pick Devices** (рис. 4) и в поле Category выбрать из списка соответствующую библиотеку, в поле Sub-category выбрать соответствующий компонент, а в поле Results — строку нужного компонента, затем нажать кнопку **OK**. В результате окно **Pick Devices** будет закрыто, а символ выбранного компонента будет прикреплен к курсору мыши, при помощи которого необходимо поместить символ в нужное место на схеме — щелкнуть в определенном месте схемы левой кнопкой мыши столько раз, сколько компонентов требуется разместить. Например, для добавления резисторов используйте библиотеку Resistors. Сопротивление резисторов можно установить в поле Resistance окна настроек **Edit Component** (рис. 5), вызвав его двойным щелчком левой кнопки мыши по выбранному резистору на схеме. В нашем случае это значение 330 Ом.

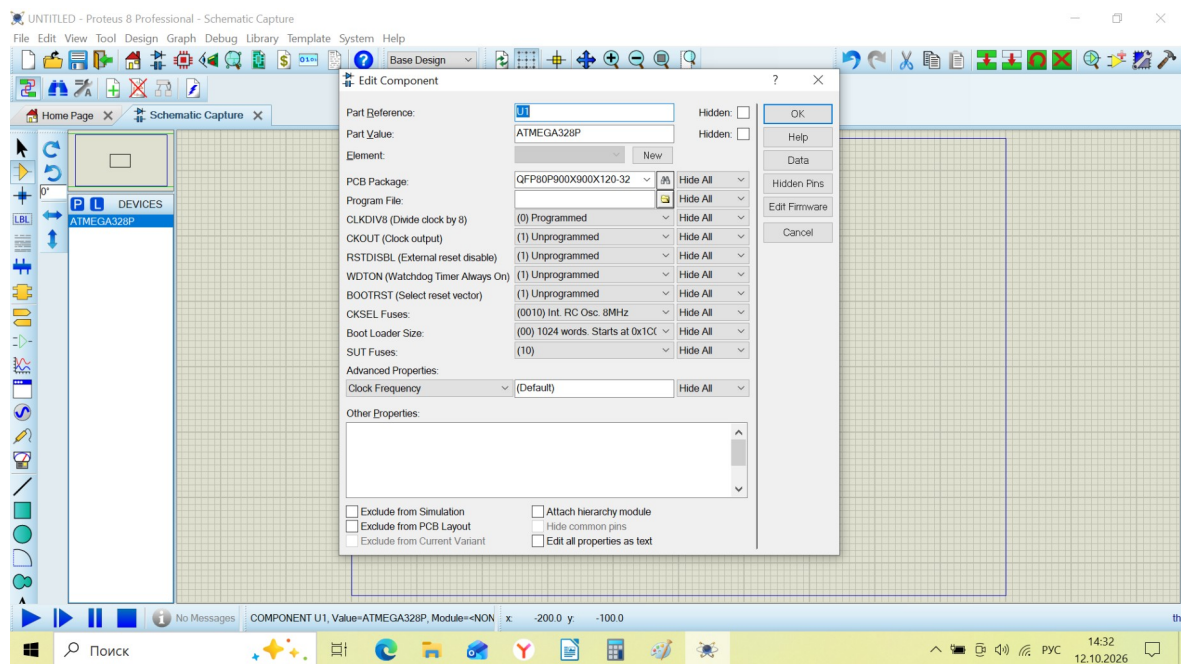


Рис. 3. Окно Edit Component

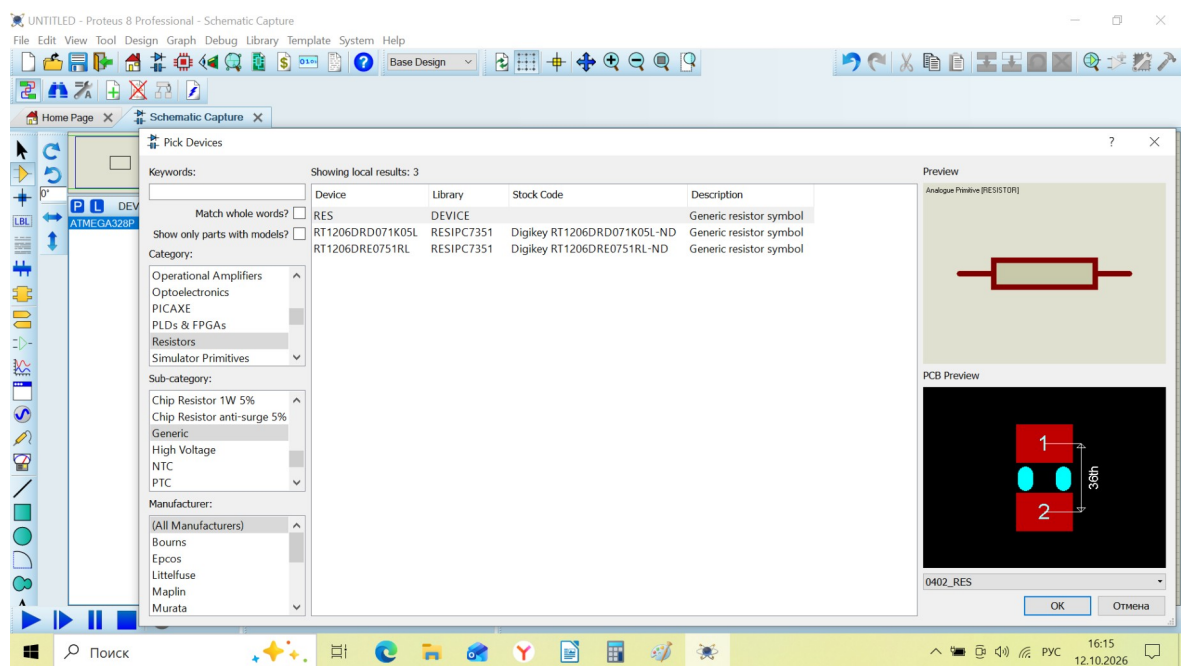


Рис. 4. Выбор резистора в окне Pick Devices

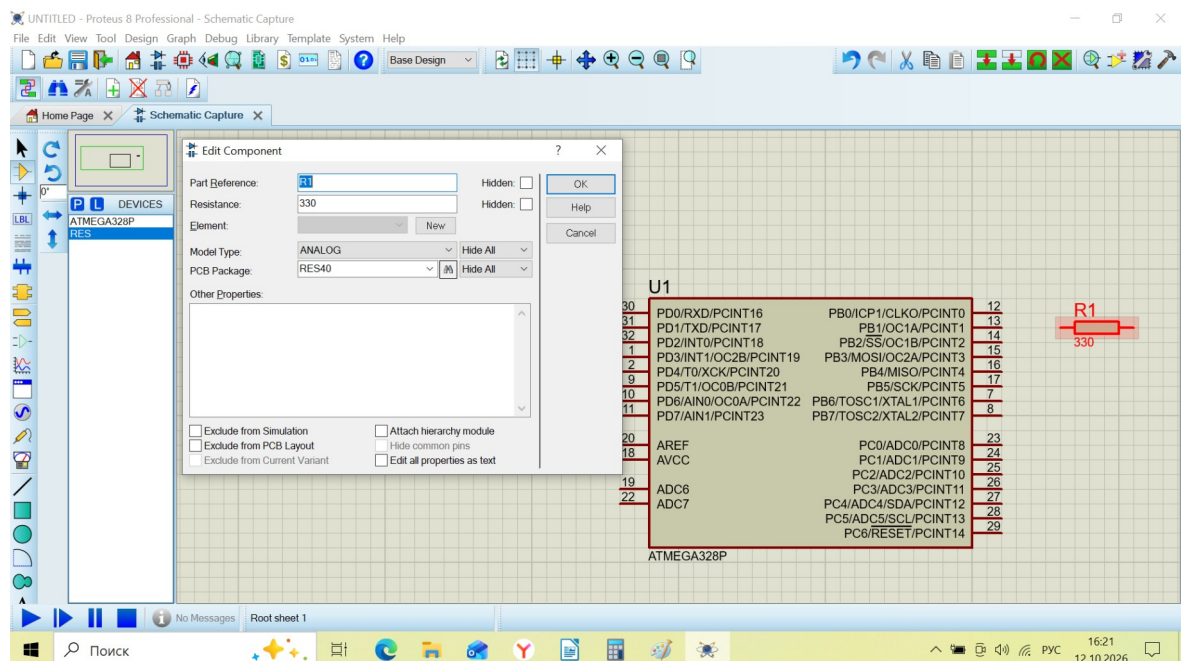


Рис. 5. Окно настроек резистора Edit Component

Аналогичным образом добавляем на схему светодиод анимированного зелёного свечения (рис. 6). Светодиоды находятся в категории *Optoelectronics* (Оптоэлектроника) и далее в подкатегории *LEDs*. В данном примере он выбран зеленого цвета *LED-GREEN* с анимацией.

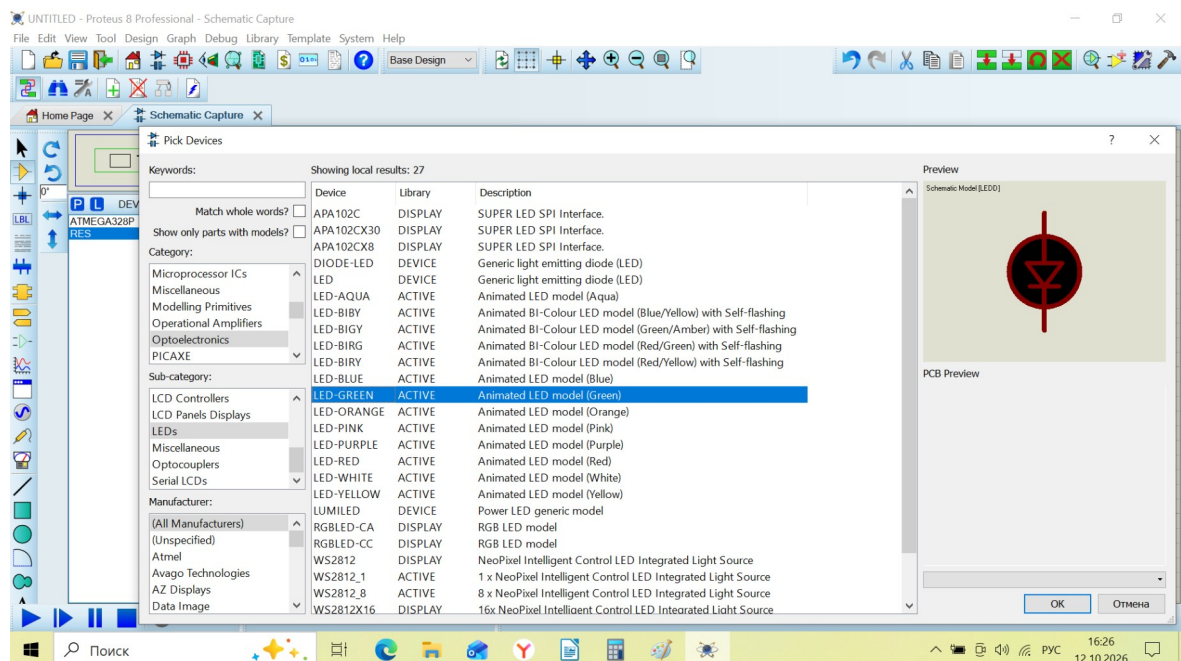


Рис. 6.

Если в схему необходимо добавить условное графическое обозначение (УГО) «земли». Сделать это можно посредством выбора при помощи левой кнопки мыши на панели **TERMINALS** (рис. 7) строки с названием **GROUND**. Для того чтобы открыть панель **TERMINALS**, нужно на левой панели редактора **ISIS** выбрать кнопку **Terminals Mode**.

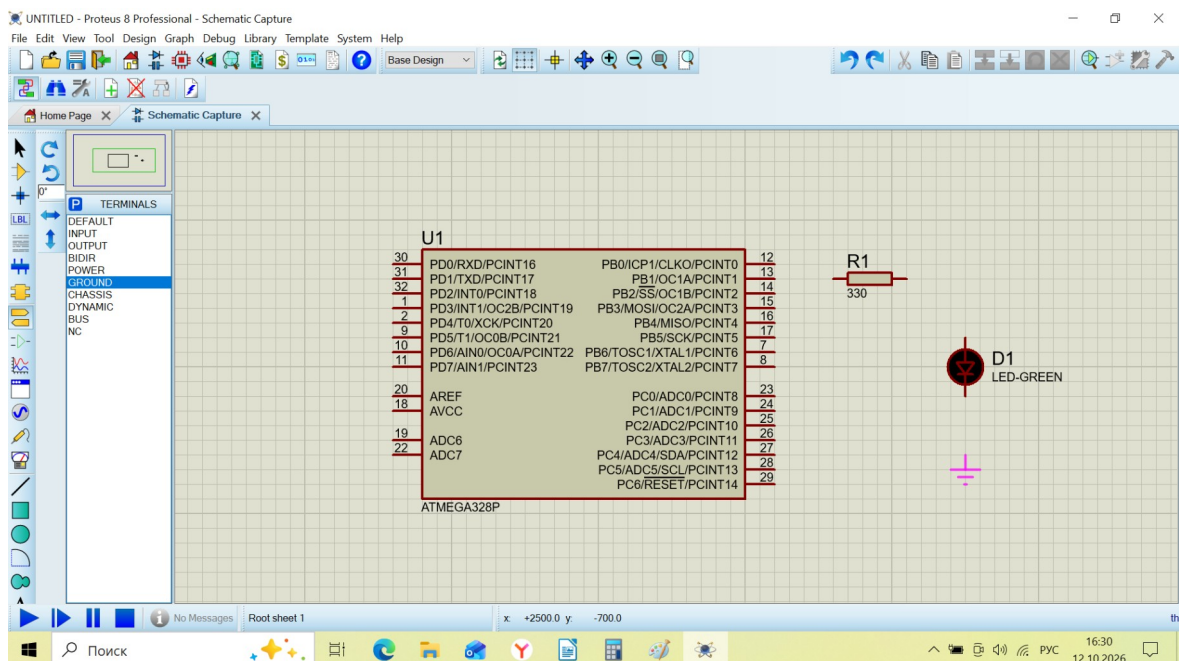


Рис. 7. Панель TERMINALS

Теперь собираем схему, как показано на рисунке 8. Для соединения элементов проводником необходимо левой кнопкой мыши (ЛКМ) указать исходную точку вывода элемента, а затем конечную точку вывода подключаемого элемента. К выводу МК PB1 подсоединяем резистор R1, который соединяем с анодом светодиода D1. Катод светодиода соединяем с «землей». Элемент «земля» находится в меню вкладки *Terminals Mode*.

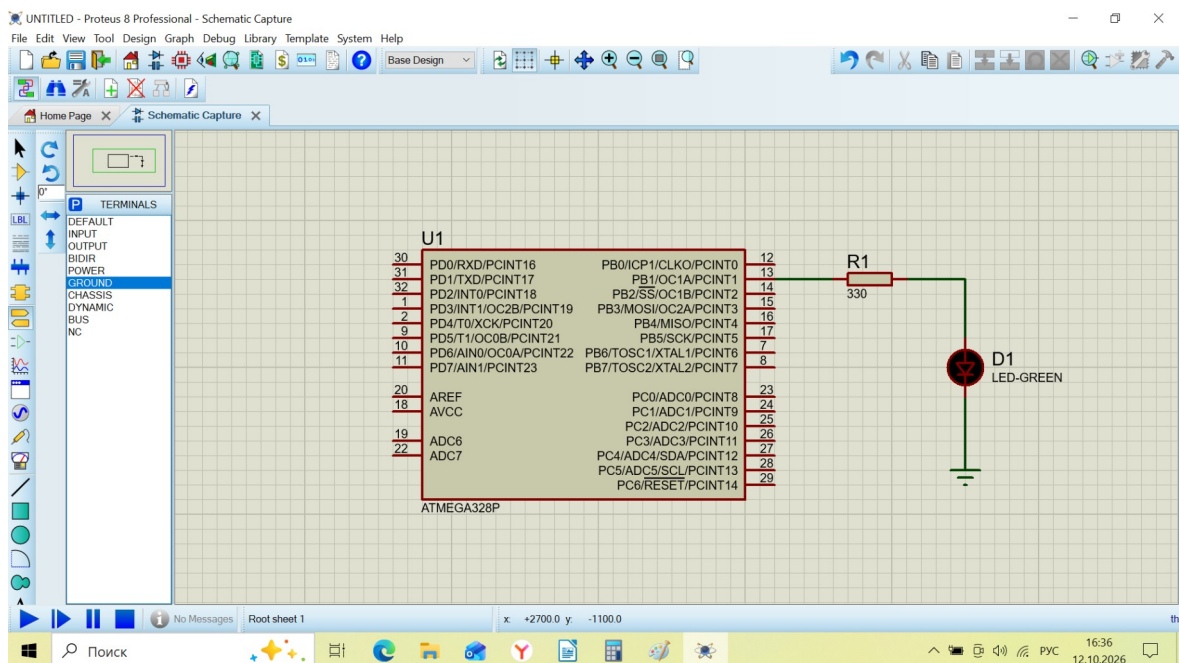


Рис. 8.

Теперь схема готова к симуляции, но пока нет управляющей программы в микроконтроллере. Её тоже можно написать и скомпилировать в ИСР Протеус.

Разработка и компиляция программного кода

Открыть вкладку, на которой в процессе проектирования будет вноситься код программы микроконтроллера, можно следующим образом. Выделите при помощи

левой кнопки мыши символ микроконтроллера в рабочее поле проекта, при помощи правой кнопки мыши вызовите контекстное меню и выберите в нем пункт Edit Source Code (рис. 9).

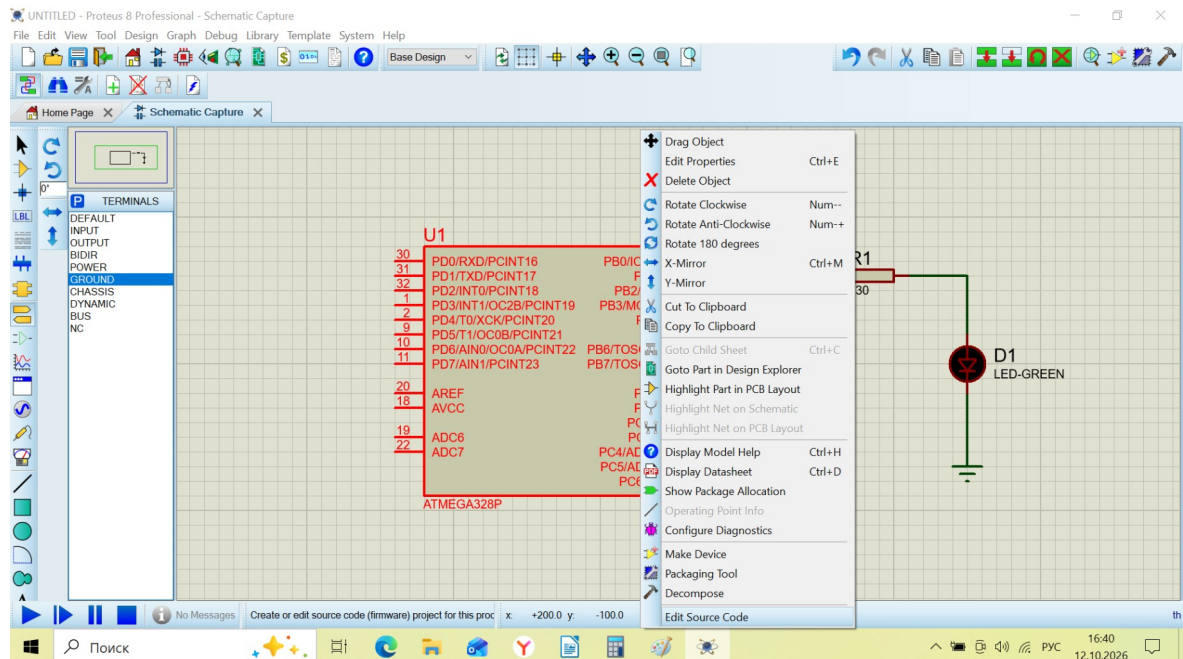


Рис. 9.

В результате в проект будет добавлена вкладка Source Code и открыто окно выбора микроконтроллера **New Firmware Project** (рис. 10), в котором устанавливаются следующие параметры:

- Family — семейство микроконтроллера;
- Controller — модель микроконтроллера;
- Compiler — компилятор;
- Create Quick Start Files — автоматическое создание заготовки программного кода для микроконтроллера.

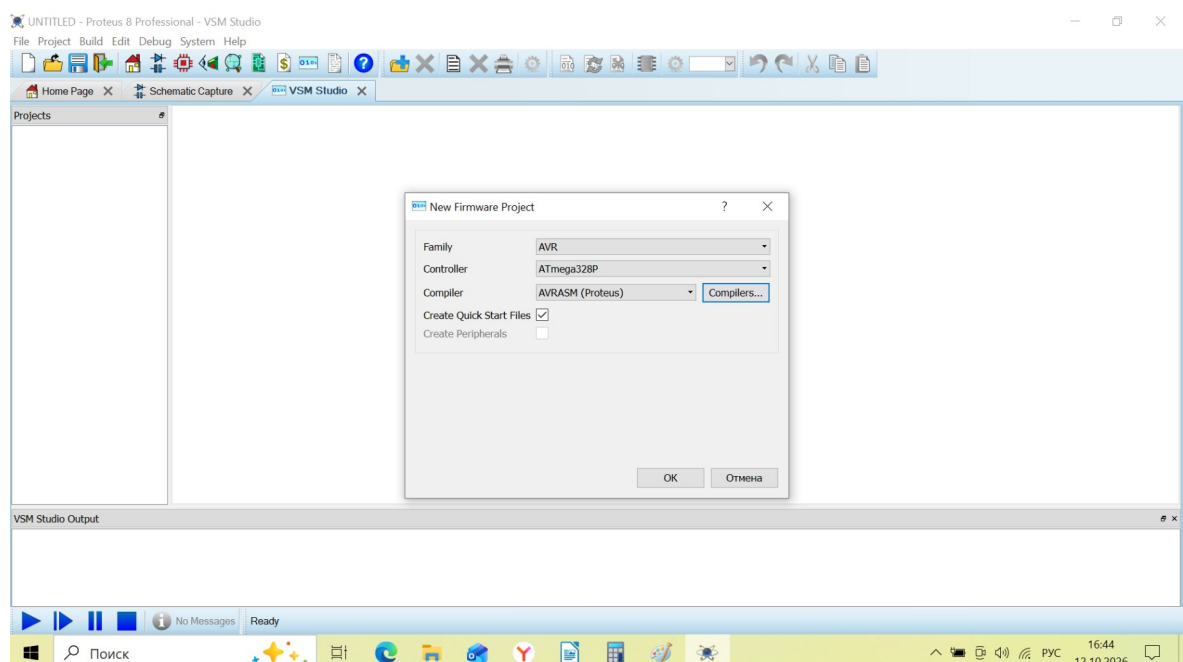


Рис. 10. Окно New Firmware Project

В результате откроется окно Source Code текстового редактора кода (рис. 11).

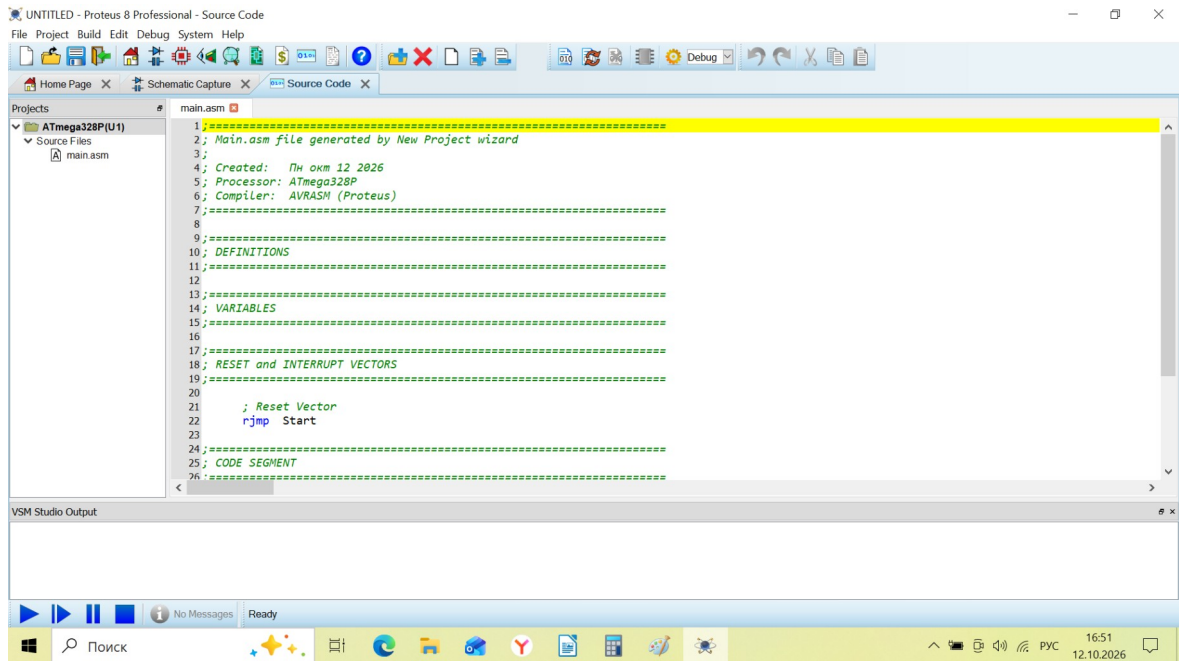


Рис. 11.

В проект можно также добавить и уже подготовленный ранее .asm— или .c-файл. Для этого на вкладке **Source Code** на панели дерева проектов **Projects** выберите при помощи левой кнопки мыши проект, к которому необходимо добавить .asm— или .c-файл, при помощи правой кнопки мыши вызовите контекстное меню и выберите в нем пункт Add Files. В результате выполненных действий откроется окно файлового менеджера, в котором необходимо выбрать нужный файл, после чего нажать кнопку «Открыть».

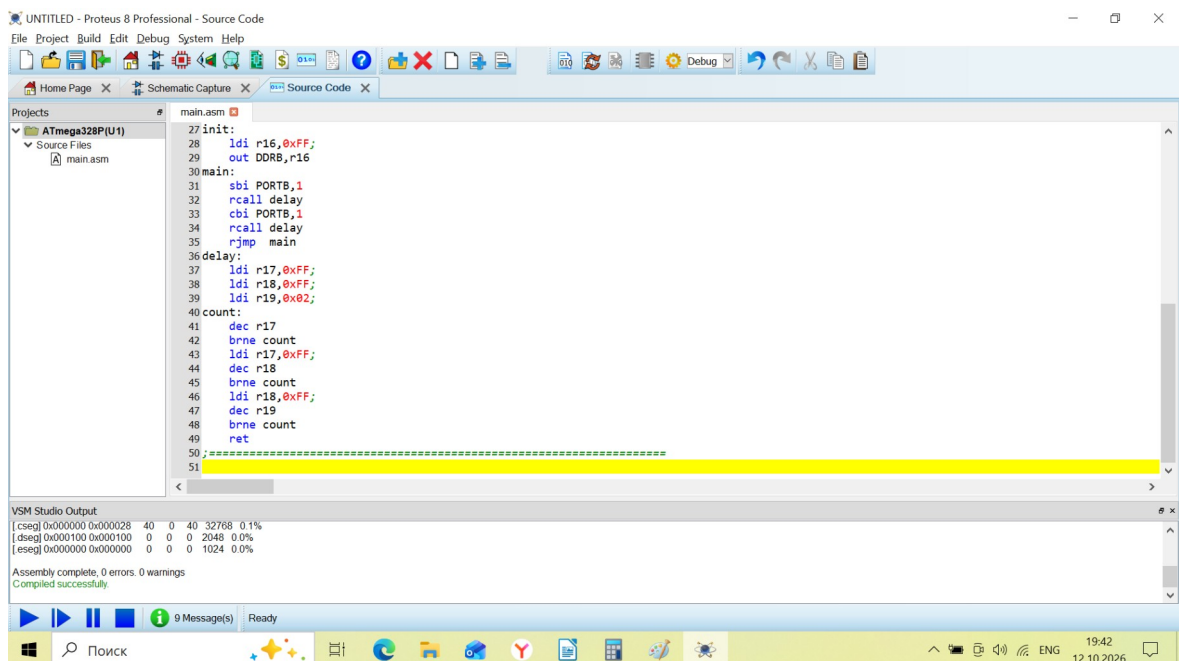


Рис. 12. Вкладка AVR Source Code программы, написанной на языке ассемблер

Компиляция и отладка программы выполняется на вкладке AVR Source Code и инициализируется командой основного меню редактора ISIS — Debug/Start VSM Debugging

Если компилятор в листинге программы при компиляции ассемблерного кода с

помощью компилятора AVRASM не обнаружит ошибок, то на диске вашего компьютера в рабочей папке проекта будут созданы *.elf, *.hex—, *.obj— и *.asm-файлы.

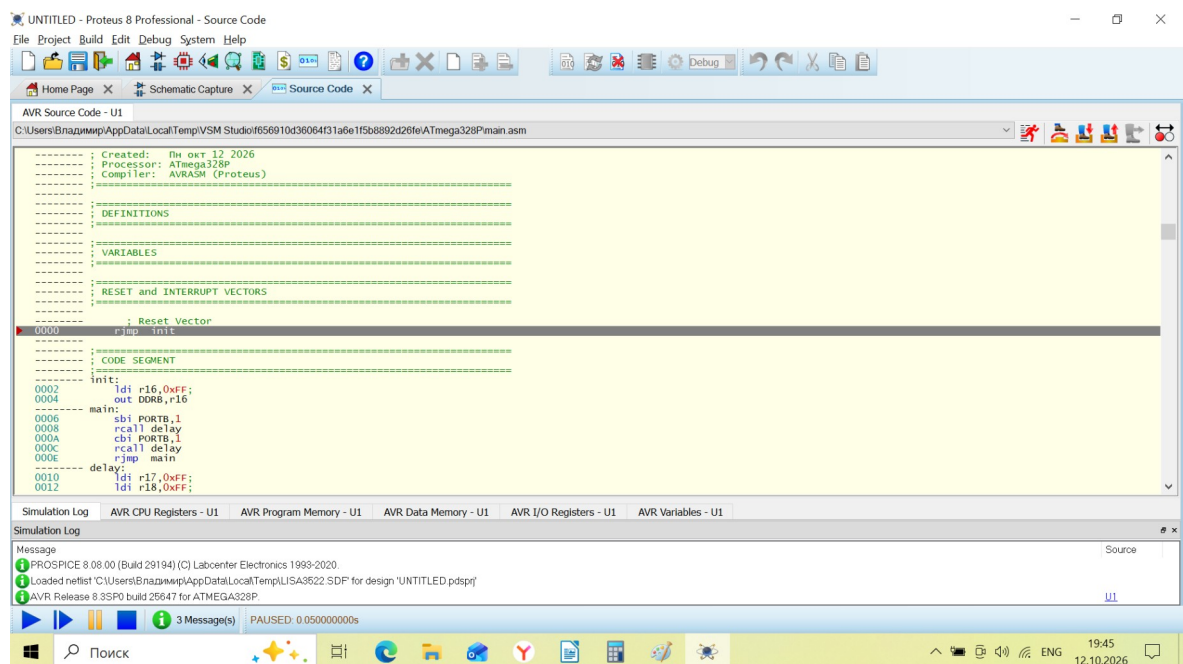


Рис. 13. Окно отладки программы, написанной на языке ассемблер.

Отчет о результате компиляции программного кода, отображается в нижней части вкладки Source Code на панели VSM Studio Output (см. рис. 12).

Состояние памяти и регистров микроконтроллера можно просмотреть с помощью выбора опций Debug/AVR основного меню редактора ISIS (рис. 14).

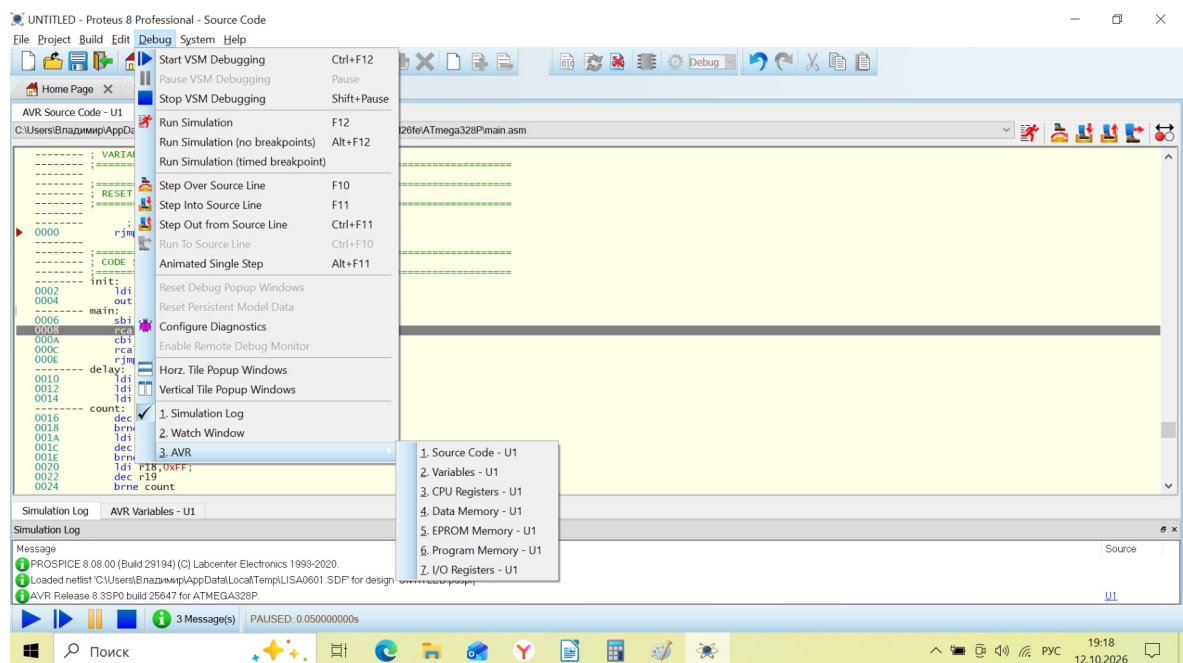


Рис. 14.

Моделирование работы электрической схемы

После того как в рабочей области проекта собрана схема (см.рис. 8), а на вкладке Source Code введен код программы (рис. 12), можно запускать моделирование.

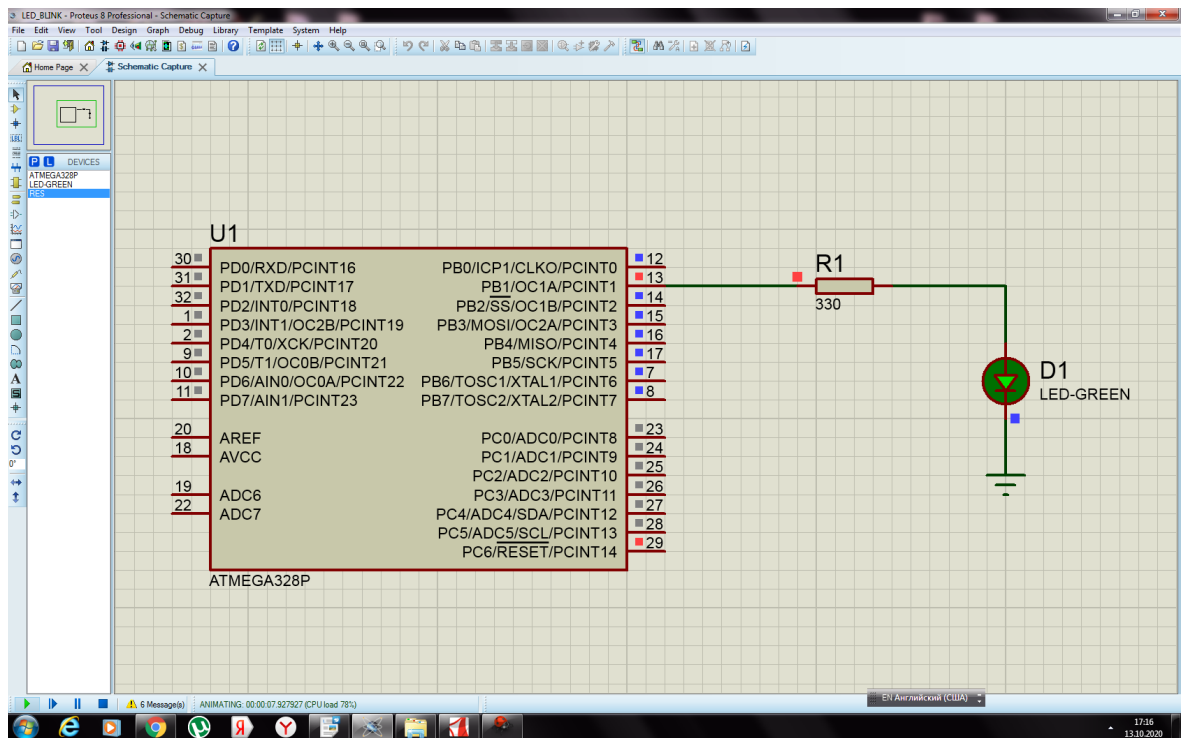


Рис. 15. Демонстрационная схема с использованием микроконтроллера ATmega328P

Процесс моделирования запускается при помощи кнопки **Run the simulation**, которая находится в левом нижнем углу окна программы. Для того чтобы временно приостановить процесс симуляции, используйте кнопку **Pause the simulation, or start up at time 0 if stopped** (кнопка находится в левом нижнем углу окна программы). Остановить моделирование можно при помощи кнопки **Stop the simulation**.

Для отладки можно с помощью меню использовать и пошаговый режим, а также устанавливать точки остановки выполнения программы (рис. 16).

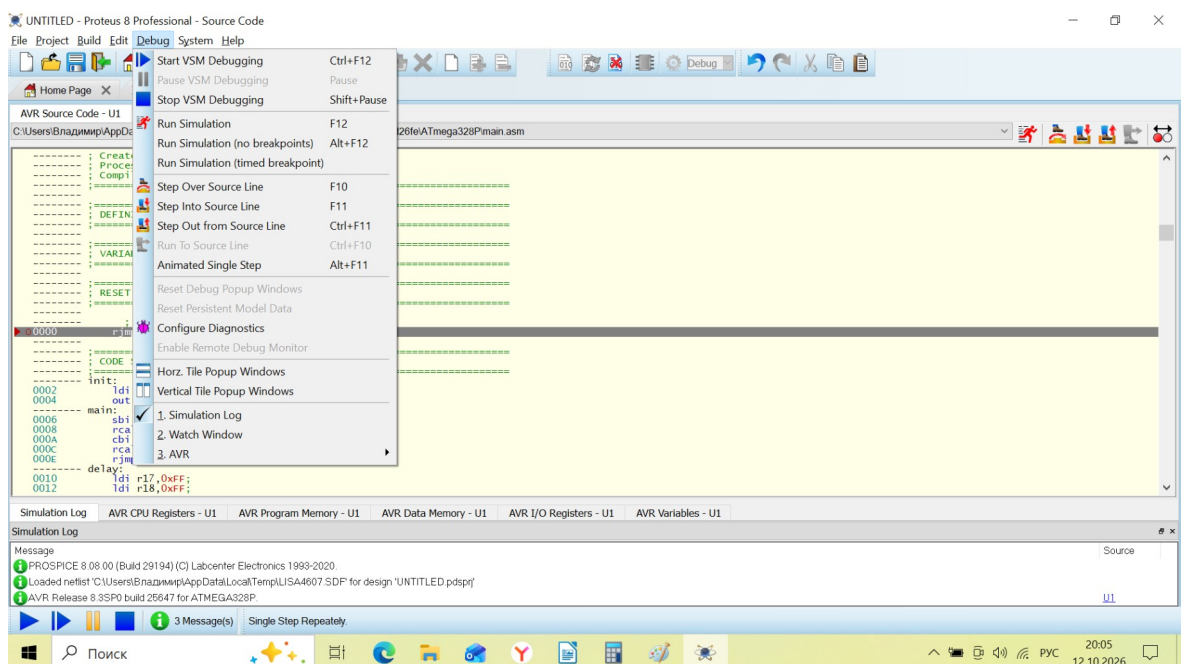


Рис. 16.

Красный цвет маленького квадратика на рис. 15 возле вывода 13 микроконтроллера, указывает на то, что на выводе присутствует высокий потенциал,

т. е. + 5В. Низкий потенциал отображается синим цветом. Обратите внимание, высокий потенциал всегда имеет 29-й вывод RESET. Поэтому, в случае необходимости, для сброса или перезапуска программы, вывод RESET нужно кратковременно соединить с землей.

Если программа для микроконтроллера разрабатывалась с помощью других средств, например, AVR Studio, то её тоже можно загрузить в модель микроконтроллера ИСР Proteus. Для этого в окне редактора схем Schematic Capture дважды кликаем левой кнопкой мыши по схематическому изображению микроконтроллера и в появившемся новом окне указываем путь к файлу с кодом. Место расположения файла находим кликнув по значку в виде открытой папки в строке *Program File*.

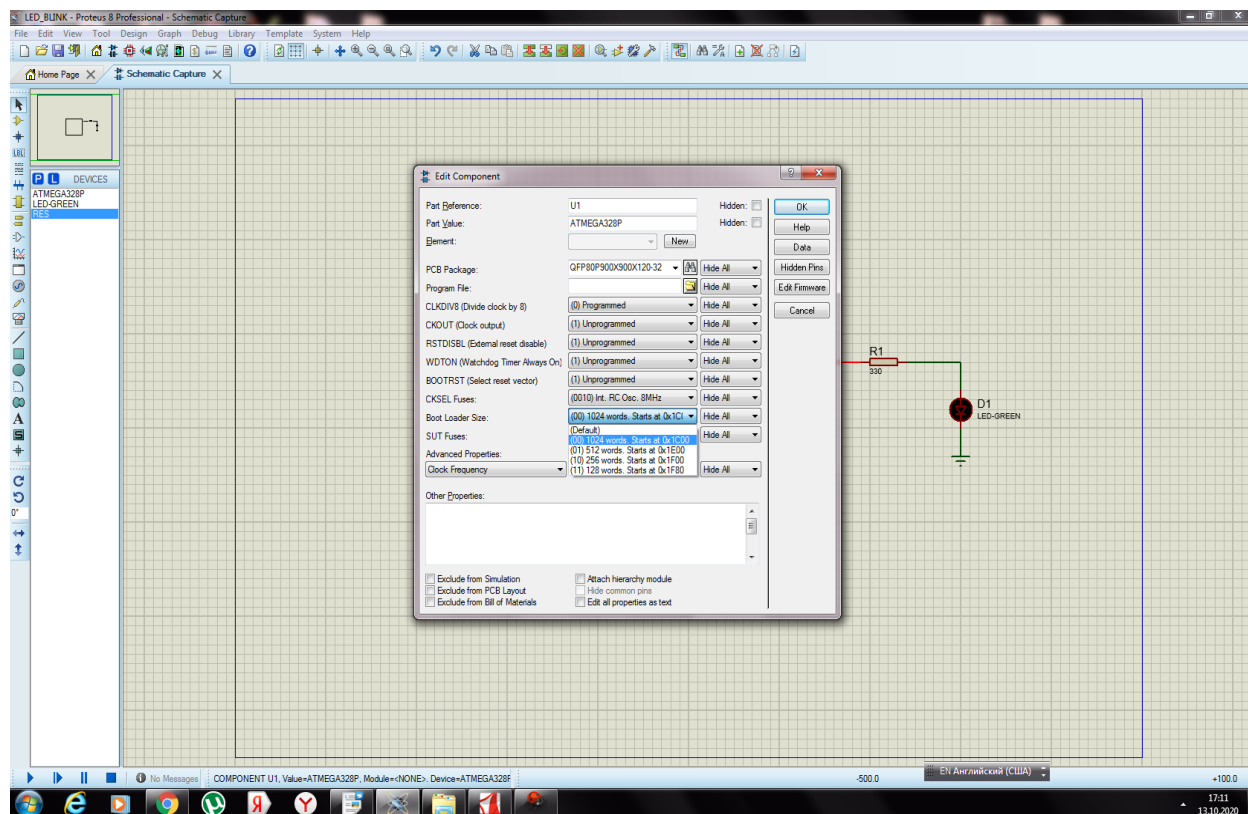


Рис. 17.

В папке с проектом находим папку *Debug* и в ней выбираем файл с расширением *HEX* (рис. 18). После этого нажимаем кнопку *Открыть*.

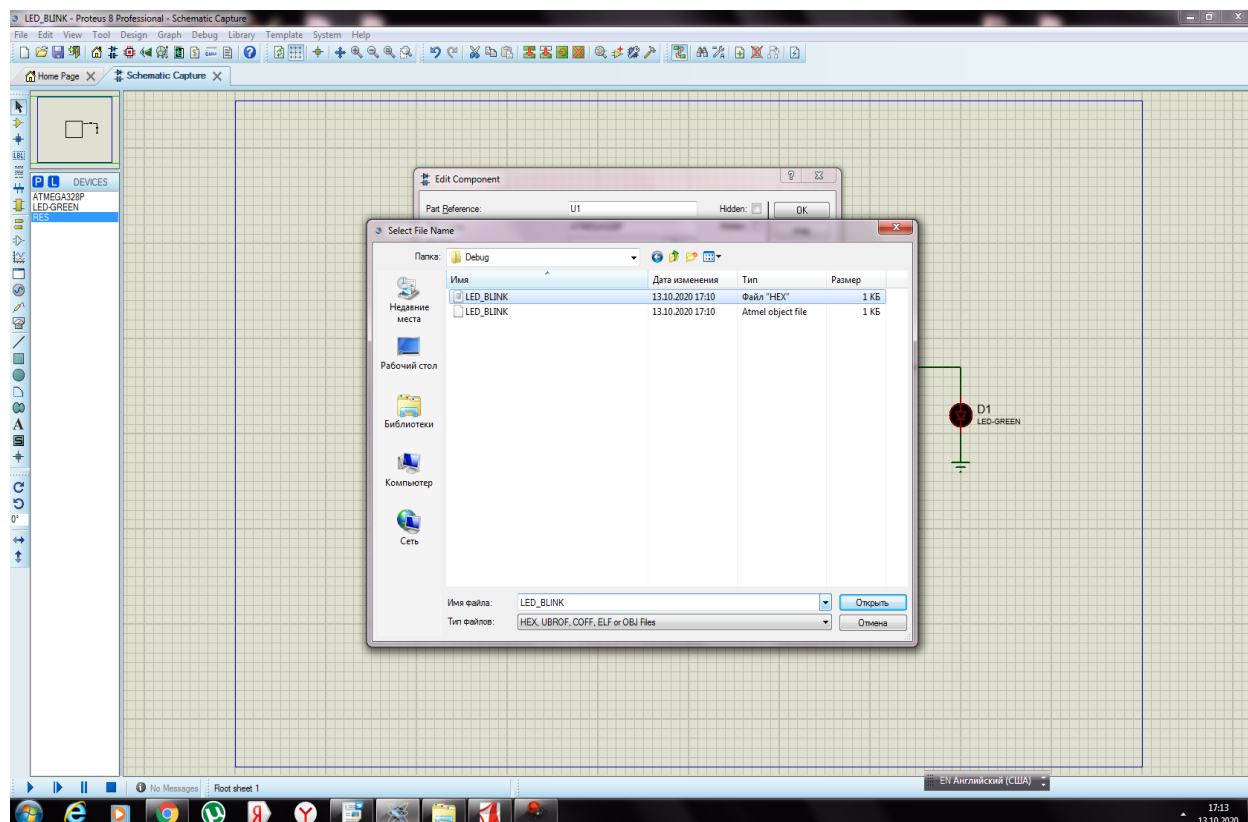


Рис. 18.

Теперь все готово. Для начала симуляции нажимаем кнопку *Run the simulation* треугольной формы в левом нижнем углу окна программы.

Задание

Выполните действия по разработке и загрузке в память МК ATmega 328P программы, реализующей сигнал ШИМ на выводе микроконтроллера OC2B с частотой $PWM_FREQ = 16 \text{ кГц}$ со статически заданным коэффициентом заполнения $PULSE_WIDTH_percent = 0.30$. Тактовая системная частота микроконтроллера $clk_{I/O} = 8 \text{ МГц}$.

Проверьте работу программы ШИМ на примере схемы повышающего ШИМ-преобразователя путём симуляции её совместно с микроконтроллером в ИСР Proteus 8.

Пример реализации.

Для инициализации работы таймера в требуемом режиме необходимо записать в регистры управления соответствующего таймера требуемые коды. Управляющими регистрами таймера T0 МК ATmega-328P, как можно увидеть из соответствующего Datasheet являются регистры TCCR0A и TCCR0B. Рассмотрим их побитно и определим какие разряды за что отвечают.

TCCR0A – Timer/Counter Control Register A

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Итак, биты WGM01, WGM00 задают режим работы таймера (TIMER_MODE) в соответствии со следующей таблицей

Режим таймера	WGM01	WGM00
---------------	-------	-------

<i>Normal</i>	0	0
<i>Phase correct PWM</i>	0	1
<i>CTC</i>	1	0
<i>Fast PWM</i>	1	1

Мы будем использовать режим фазокорректный ШИМ. Поэтому в соответствующие разряды регистра TCCR0A запишем 0 и 1.

Биты COM0A1, COM0A0 задают режим вывода сигнала сравнения OC0A канала А таймера T0 , а COM0B1, COM0B0 соответственно канала В в соответствии с таблицей

Режим вывода OC0	COM0x1	COM0x0
Таймер отключён от вывода OC0	0	0
Вывод OC0 переключается в противоположное состояние (Toggle)	0	1
Вывод OC0 переключается в состояние «0» (Инверсный режим)	1	0
Вывод OC0 переключается в состояние «1» (Прямой режим)	1	1

Мы будем использовать только канал А в инверсном режиме вывода сигнала ШИМ на вывод OC0A микроконтроллера (вывод корпуса микросхемы 10, вывод 6 порта PD). Поэтому в соответствующие разряды COM0A1, COM0A0 регистра TCCR0A запишем 1 и 0, а вывод OC0B отключим, записав в соответствующие разряды нули.

TCCR0B – Timer/Counter Control Register B

Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Биты CS02, CS01, CS00 задают режим работы прескалера, т.е. коэффициент деления N исходной частоты $clk_{I/O}$ в соответствии с таблицей

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped)
0	0	1	$clk_{I/O}$ /(No prescaling)
0	1	0	$clk_{I/O}/8$ (From prescaler)
0	1	1	$clk_{I/O}/64$ (From prescaler)
1	0	0	$clk_{I/O}/256$ (From prescaler)
1	0	1	$clk_{I/O}/1024$ (From prescaler)
1	1	0	External clock source on T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

Определить необходимый коэффициент N прескалера можно из расчётной формулы для результирующей частоты ШИМ в заданном режиме, в нашем случае это фазокорректный ШИМ, поэтому

$$f_{OCnxPCPWM} = \frac{f_{clk\ I/O}}{N \cdot 510}$$

откуда

$$N = \frac{f_{clk/I/O}}{510 \cdot f_{PWM}} = \frac{8 \cdot 10^6}{510 \cdot 16 \cdot 10^3} \approx 1$$

Тогда в соответствии с приведённой выше таблицей в разряды CS02, CS01, CS00 записываем 001. Остальные биты в регистре заполняем нулями, поскольку в режиме фазокорректной ШИМ они должны быть нулями.

Рассчитываем абсолютное значение коэффициента заполнения PWM, которое необходимо загрузить в регистр PWM, чтобы получить PULSE_WIDTH_percent = 0.30

$$PWM = 0,30 \cdot 256 = 77$$

Создаём с помощью ИСР Proteus 8 принципиальную схему повышающего ШИМ-преобразователя с управлением от МК Atmega-328P (Рис. 19).

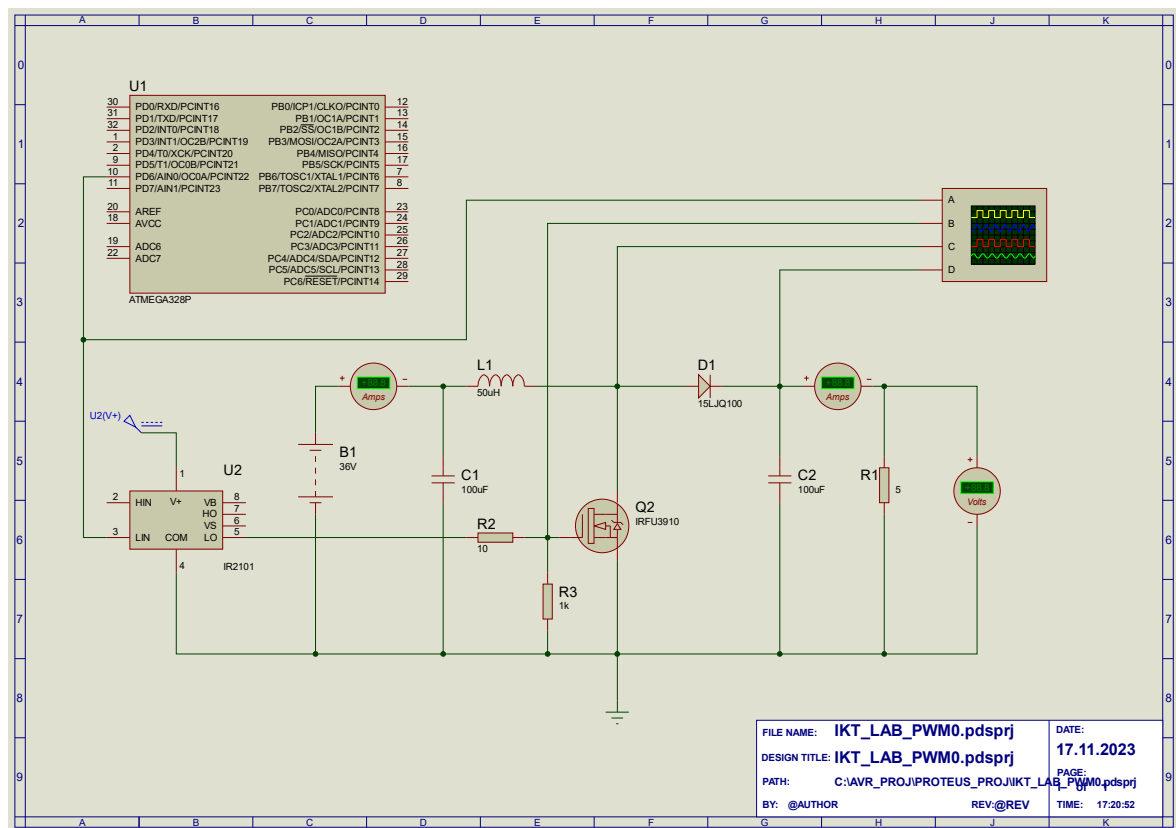


Рис. 19.

Далее открываем программный редактор ИСР Proteus и начинаем писать программу, комментируя основные действия.

Пример текста программы выполнения задания:

```
/*
 * Project: IKT_LAB_PWM_0
 *
 * FILE NAME: IKT_LAB_PWM0.asm
 * PROGRAMM LANGUAGE: AVRASM2
 */
```

```

* Author:          V.G.Komarov
* Company:         TransElectric
* Version: 1 - 25.10.2022
*
* Назначение: ИЗУЧЕНИЕ РАБОТЫ ТАЙМЕРОВ
*
* Используемые аппаратные ресурсы:
* Платформа: ALGUS_WE_PICO
* Микроконтроллер: ATmega328P
*
* Описание:
*
*/
#include "m328Pdef.inc"
.device ATmega328P
// КОНСТАНТЫ
.equ XTAL = 16000000 // Частота МК

//=====
// ПЕРЕМЕННЫЕ
.def temp =r16 //регистр временного хранения
.def PWM =r24 //заполнение ШИМ

.CSEG
//ВЕКТОРА ФУНКЦИЙ ОБРАБОТКИ ПРЕРЫВАНИЙ
.org 0x000
rjmp setup
.org 0C0Aaddr
rjmp TIM0A_comp //Переход на функцию обработки прерывания по совпадению
с OSR0A
//=====
//ФУНКЦИИ

.org INT_VECTORS_SIZE

//функция обработки прерывания TIM0 по совпадению с OCR0A
TIM0A_comp:
cli // Общий запрет прерываний
ldi PWM, 180
out OCR0A, PWM
sei
reti

//Функция сброса и начальной установки
setup:

// Установка значений в ОЗУ и регистры R0-R31
clr PWM

//настройка указателя стека
ldi temp, low(RAMEND)
out SPL, temp
ldi temp, high(RAMEND)
out SPH, temp

/*Настройка таймера 0:
запись кода режима в регистры управления TCCR0
(пределитель N=1, режим обратный Phase Correct PWM,
при Fclk=16 МГц частота квантования ШИМ 31,25 кГц)
*/
ldi temp, 0b10000001
out TCCR0A, temp
ldi temp, 0b00000001
out TCCR0B, temp

```

```

//установка вывода сигнала ШИМ на OC0A (PD6)
sbi DDRD, 6

//разрешение прерываний от таймера 0
ldi temp, (1<<OCIE0A)
sts TIMSK0, temp
clr temp
out TCNT0, temp

//общее разрешение всех прерываний
sei

main:
//главный цикл программы
rjmp main

```

Далее компилируем и проводим отладку программы в ICP Proteus 8. Загружаем скомпилированную программу в модель МК симулятора Proteus и проверяем её работу с помощью виртуального осциллографа (Рис. 20).

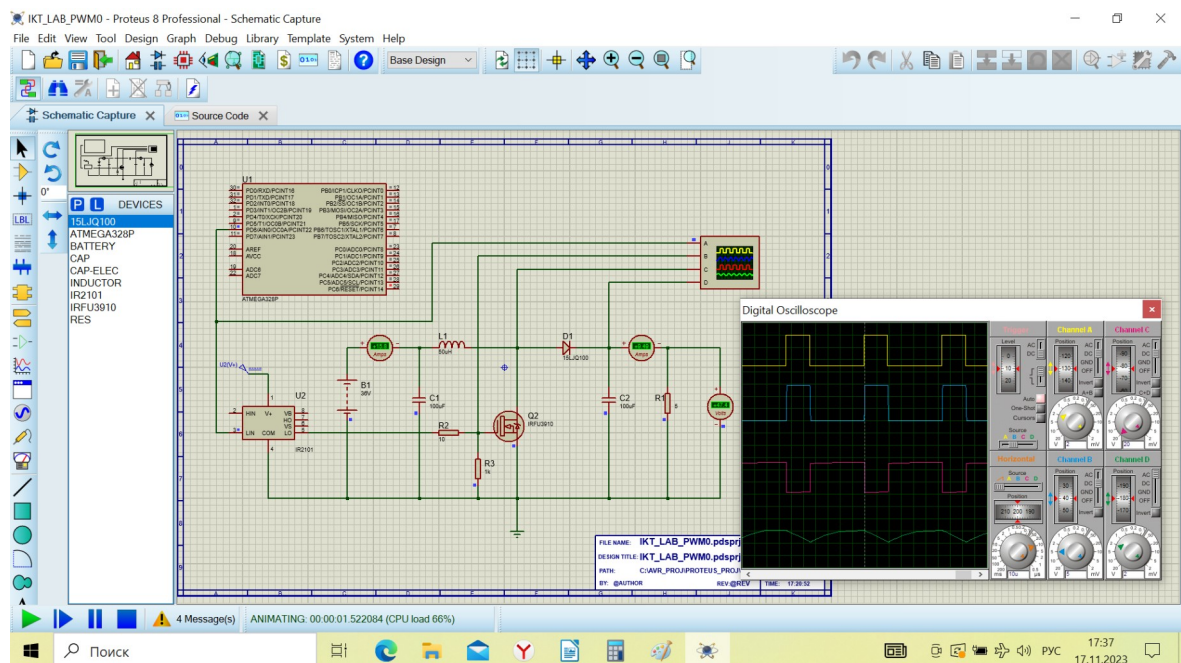


Рис. 20.

ОТЧЕТ О ВЫПОЛНЕНИИ ЗАДАНИЯ

Отчет о выполнении задания должен содержать:

1. Титульный лист, оформленный по образцу, приведенному в Приложении 1.
2. Формулировку задания.
3. Схему соединений, описание алгоритма решения и блок-схему алгоритма решения.
4. Код программы с подробными комментариями.
5. Описание и результаты симуляции работы программы.
6. Описание результатов аппаратного тестирования.
7. Выводы.
8. Список использованных источников.

Пример оформления отчета о выполнении задания приведен в Приложении 1.

Приложение 1

Титульный лист отчёта по лабораторной работе

ФГБОУ ВО
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
"МЭИ"
ИНСТИТУТ ЭЛЕКТРОТЕХНИКИ



КАФЕДРА ЭКАО И ЭТ

Направление подготовки: 13.04.02 Электроэнергетика и электротехника

Профиль подготовки: Теория движения электроподвижного состава и проблема оптимизации тягового оборудования и устройств электроснабжения транспортных систем

Квалификация (степень) выпускника: магистр

Форма обучения: очная

ОТЧЁТ
по лабораторной работе № 6
по дисциплине ИЭТ; Б1.Б.3 "Информационные и
компьютерные технологии в электротехнике"

**Тема: Широтно-импульсная модуляция с
помощью таймеров микроконтроллера**

Студент _____

Научный
руководитель _____

Фамилия, и., о. группа подпись

доцент__к.т.н._Комаров_В.Г.
должность, звание, фамилия, и., о.

Допущен	Принято	Оценка

Москва

2026 г.