Лабораторная работа №4.

Тема: "Знакомство с Arduino и Proteus". Таймеры.

Цель работы: освоить базовые навыки работы с Arduino IDE и Proteus.

Теоретическая часть

1. Аппаратные таймеры в Arduino

Arduino Uno основан на микроконтроллере ATmega328P, который имеет три аппаратных таймера:

Timer0 (8-битный): Используется для функций времени (millis() и micros()), а также для генерации ШИМ на пинах 5 и 6.

Timer1 (16-битный): Более точный таймер, используется для сложных задач, таких как управление сервоприводами или высокоточные временные интервалы. Генерирует ШИМ на пинах 9 и 10.

Timer2 (8-битный): Может работать в асинхронном режиме и используется для генерации ШИМ на пинах 3 и 11.

Аппаратные таймеры работают независимо от основного цикла программы и могут быть настроены через регистры микроконтроллера. Однако для большинства задач достаточно использовать программные таймеры, такие как millis() и micros().

2. Программные таймеры: millis() и micros().

millis():

Возвращает количество миллисекунд, прошедших с момента запуска программы (до переполнения через ~50 дней).

Тип данных: unsigned long.

Пример использования:

unsigned long currentTime = millis();

Преимущества:

Не блокирует выполнение программы.

Позволяет управлять несколькими процессами одновременно.

micros():

Возвращает количество микросекунд, прошедших с момента запуска программы (до переполнения через ~70 минут).

Тип данных: unsigned long.

Пример использования:

unsigned long currentTime = micros();

Преимущества:

Высокая точность (разрешение 4 мкс на Arduino Uno).

Подходит для задач, требующих точного измерения времени.

Различия между millis() и micros() :

millis() работает с шагом 1 мс, что достаточно для большинства задач.

micros() работает с шагом 4 мкс, что необходимо для задач, требующих высокой точности.

3. Почему millis() лучше delay()?

delay() приостанавливает выполнение программы на указанное количество миллисекунд. Блокирует выполнение других задач, что делает его непригодным для многозадачных систем.

millis() не блокирует выполнение программы. Позволяет выполнять другие задачи во время ожидания.

Пример:

```
// С delay()
digitalWrite(LED_PIN, HIGH);
delay(500); // Программа "зависает" на 500 мс
digitalWrite(LED_PIN, LOW);
delay(500);

// С millis()
if (currentMillis - previousMillis >= interval) {
  previousMillis = currentMillis;
  digitalWrite(LED_PIN, !digitalRead(LED_PIN));
}
```

Задание 1. Мигание светодиода с использованием таймера
Цель: научиться использовать таймеры для создания временных интервалов.
Компоненты Proteus:

- Arduino Uno.
- LED (светодиод).
- резистор, например, 220 Ом

Описание:

- Анод светодиода подключите к цифровому пину D8.
- Катод светодиода подключите к GND через резистор 220 Ом.

Напишите программу, которая использует таймер (millis()):

- Включает светодиод на 500 мс.
- Выключает светодиод на 500 мс.
- Повторяет этот процесс бесконечно.

Задание 2. Мигающий светодиод с переменной частотой
Цель: научиться настраивать таймер для создания ШИМ-сигнала с изменяемой частотой.
Компоненты:

- Arduino UNO (в библиотеке Proteus)
- Светодиод (LED-RED)
- Резистор 220 Ом
- Потенциометр 10 кОм

Задача:

Создать схему, где светодиод мигает с частотой, зависящей от положения потенциометра. Использовать Timer1 Arduino для генерации прерываний. Частота мигания должна изменяться от 0.5 Гц до 5 Гц.

Задание 3. Управление несколькими светодиодами с разными интервалами
Цель: научиться управлять несколькими устройствами с разными временными интервалами.
Компоненты Proteus:

- Arduino Uno
- три светодиода
- три резистора, например, по 220 Ом

Описание:

- Первый светодиод подключите к D9 через резистор 220 Ом.
- Второй светодиод подключите к D10 через резистор 220 Ом.
- Третий светодиод подключите к D11 через резистор 220 Ом.

Напишите программу, которая:

- Включает первый светодиод каждые 1000 мс.
- Включает второй светодиод каждые 1500 мс.
- Включает третий светодиод каждые 2000 мс.

Используйте millis() для управления временными интервалами.

Задание 4. Секундомер.
Цель: научиться создавать простой секундомер с использованием таймера.
Компоненты Proteus:

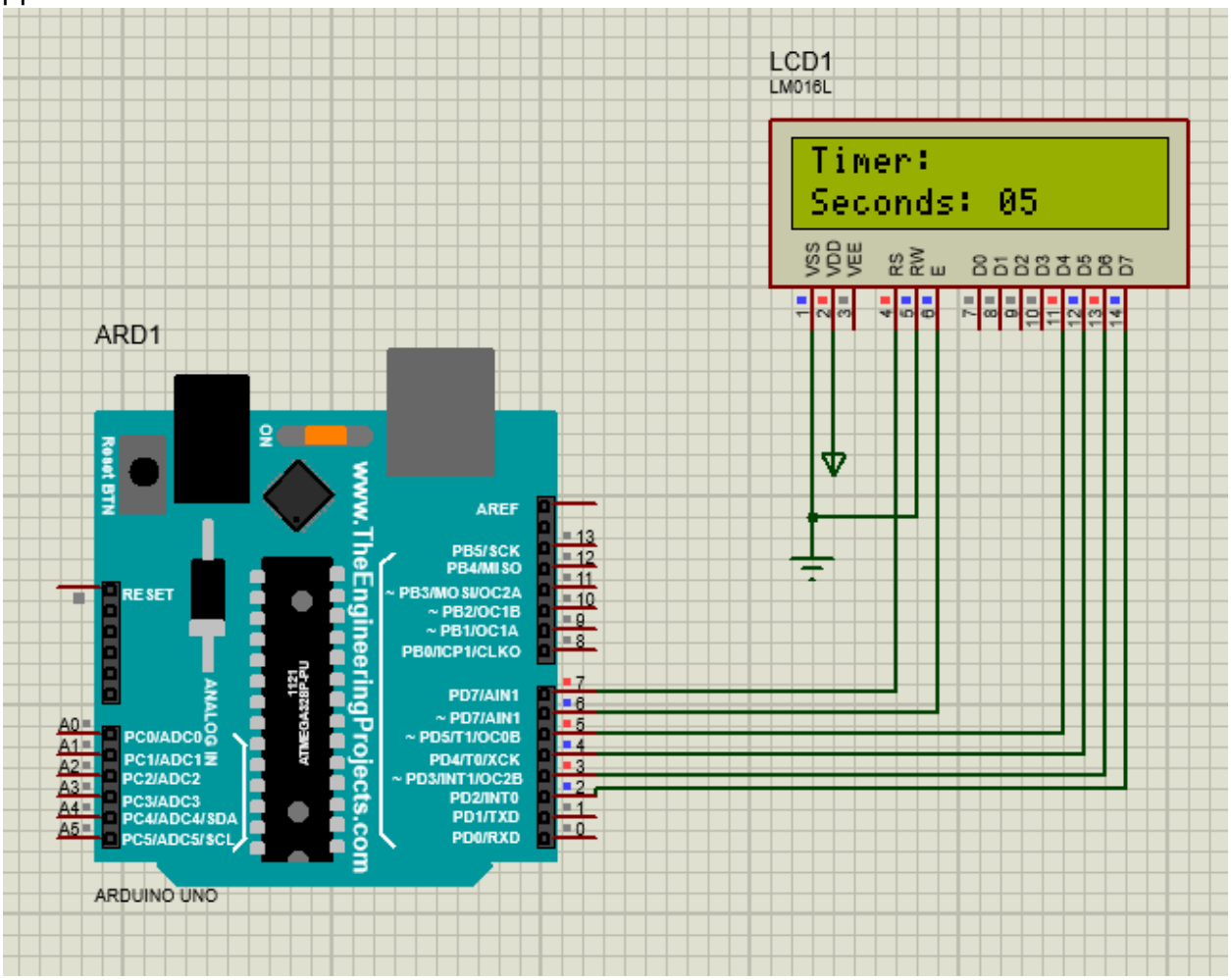
- Arduino Uno .
- LCD Display (например, LM016L).

Соедините компоненты:

- LCD (LM016L) к Arduino Uno:
- RS (Pin 4) LCD -> Цифровой пин D7 Arduino
- E (Pin 6) LCD -> Цифровой пин D6 Arduino
- D4 (Pin 11) LCD -> Цифровой пин D5 Arduino
- D5 (Pin 12) LCD -> Цифровой пин D4 Arduino
- D6 (Pin 13) LCD -> Цифровой пин D3 Arduino
- D7 (Pin 14) LCD -> Цифровой пин D2 Arduino
- VSS (Pin 1) LCD -> GND Arduino
- VDD (Pin 2) LCD -> +5V (VCC) Arduino
- R/W (Pin 5) LCD -> GND Arduino (режим записи)

Напишите программу, которая:
Отображает счетчик секунд на LCD дисплее.
Обновляет значение каждую секунду, используя millis().

При достижении 60 секунд обнуляет счетчик.
Примечание: Используйте библиотеку LiquidCrystal для работы с LCD дисплеем.



Задание 5. Управление сервоприводом с помощью таймера
Цель: научиться управлять сервоприводом с использованием таймера.
Компоненты Proteus:

- Arduino Uno.
- MOTOR-PWMSERVO из библиотеки (MOTORS) (сервопривод).

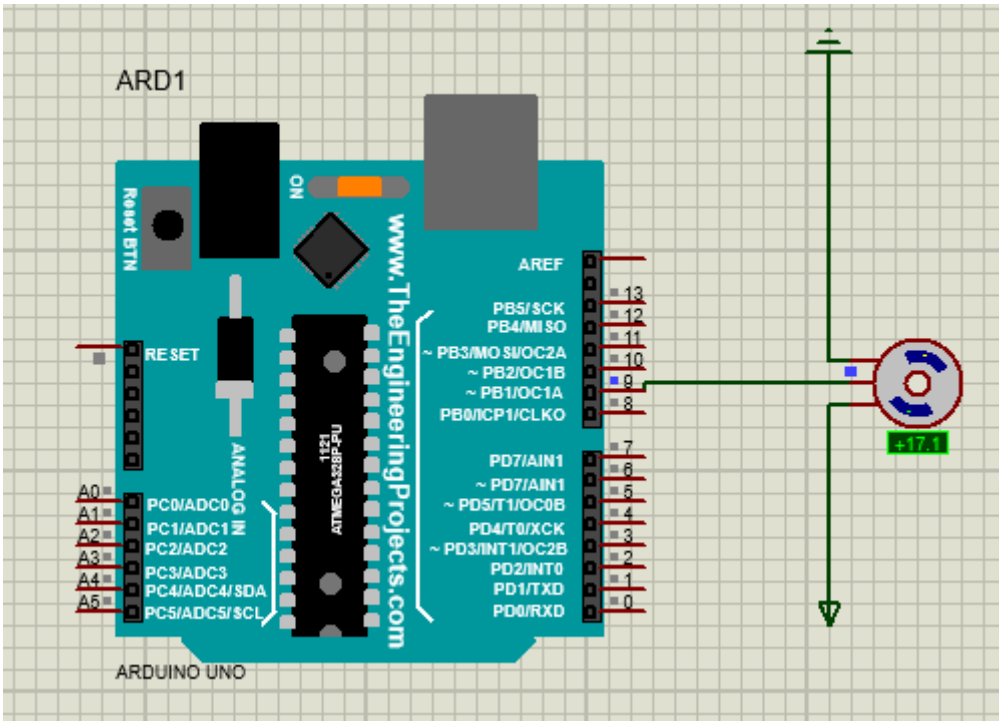
Описание:

- Сигнальный провод сервопривода подключите к цифровому пину D9.
- Питание (+5V) подключите к +5V.
- Землю (GND) подключите к GND.

Напишите программу, которая:

- Поворачивает сервопривод от 0° до 180° за 2 секунды.
- Затем поворачивает его обратно от 180° до 0° за 2 секунды.

Используйте millis() для управления временными интервалами.
Примечание: Используйте библиотеку Servo для управления сервоприводом.



Задание 6. Создание таймера с кнопкой
Цель: научиться создавать таймер, который запускается и останавливается с помощью кнопки.
Компоненты Proteus:

- Arduino Uno
- кнопка
- резистор, например, 10 кОм — для подтяжки кнопки.
- светодиод)
- резистор, например, 220 Ом

Описание:

- Один вывод кнопки подключите к цифровому пину D2.
- Второй вывод кнопки подключите к GND.
- Добавьте подтягивающий резистор (10 кОм) между D2 и +5V.
- Анод светодиод подключите к цифровому пину D8.
- Катод подключите к GND через резистор 220 Ом.

Напишите программу, которая:

- При нажатии кнопки запускает таймер.
- Через 5 секунд включает светодиод.

Используйте millis() для управления временными интервалами.

Задание 7. Многозадачность с таймерами
Цель: научиться выполнять несколько задач одновременно с использованием таймеров.
Компоненты Proteus:

- Arduino Uno.
- два светодиода

- два резистора, по 220 Ом
- кнопка (button)
- резистор, 10 кОм — для подтяжки кнопки.

Описание:

- Первый светодиод подключите к D9 через резистор 220 Ом.
- Второй светодиод подключите к D10 через резистор 220 Ом.
- Один вывод кнопки подключите к цифровому пину D2.
- Второй вывод кнопки подключите к GND.
- Добавьте подтягивающий резистор (10 кОм) между D2 и +5V.

Напишите программу, которая:

- Мигает первым светодиодом каждые 500 мс.
- При нажатии кнопки включает второй светодиод на 3 секунды.

Используйте millis() для управления временными интервалами.

Задание 8. Реализация часов реального времени
Цель: научиться создавать простые часы реального времени с использованием таймера.
Компоненты Proteus:

- Arduino Uno.
- LCD Display (например, LM016L).

Описание:
Подключите LCD дисплей (как в задании 4).
Напишите программу, которая:

- Отображает текущее время в формате "Часы:Минуты:Секунды" на LCD дисплее.
- Обновляет значения каждую секунду, используя millis().

Примечание: Используйте переменные для хранения часов, минут и секунд. При достижении 60 секунд увеличивайте минуты, а при достижении 60 минут — часы.