

## Практическое занятие №

### Тема: «Таймеры счетчики МК Работа с модулем МК в программе»

**Цель работы:** приобрести практические навыки по подключению и программированию ЖК-дисплея LCD1602A на платформе Arduino.

#### Последовательность выполнения работы:

- Собрать схемы на макетной плате, иначе при отсутствии набора Arduino в web-приложениях (<https://wokwi.com/projects/new/arduino-uno> или <https://www.tinkercad.com/>) для приведенных примеров.
- Запрограммировать микроконтроллер согласно заданию в примере.
- Выполнить задание для самостоятельной работы.

#### Содержание отчета:

- Название практического занятия, его цель.
- Фото или скриншоты собранной схемы.
- Написанный программный код вставить текстом, Courier New, 12 кегль, одинарный отступ без абзацев.
- Вывод о проделанной работе.
- Файл Fritzing с принципиальной и монтажной схемой.

## ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Таймеры – это фундаментальная концепция в программировании микроконтроллеров, позволяющая отслеживать временные интервалы и выполнять действия через определенные промежутки времени. В Arduino существует несколько подходов к работе с таймерами.

## ОСНОВНЫЕ МЕТОДЫ РАБОТЫ С ВРЕМЕНЕМ

#### Функция *delay()*

```
void loop() {  
    digitalWrite(LED_PIN, HIGH);  
    delay(1000); // Блокирует выполнение на 1 секунду  
    digitalWrite(LED_PIN, LOW);  
    delay(1000);  
}
```

- Блокирует выполнение всей программы
- Невозможно выполнять другие задачи во время ожидания
- Не подходит для реактивных систем

## Функция *millis()*

```
unsigned long previousMillis = 0;
const long interval = 1000; // 1 секунда

void loop() {
  unsigned long currentMillis = millis();

  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    // Выполняем действие здесь
    digitalWrite(LED_PIN, !digitalRead(LED_PIN));
  }

  // Другие задачи выполняются здесь без блокировки
}
```

- Не блокирует выполнение программы
- Позволяет выполнять *multiple tasks* одновременно
- Точное отслеживание времени

## Функция *micros()*

```
unsigned long previousMicros = 0;
const long interval = 500; // 500 микросекунд

void loop() {
  unsigned long currentMicros = micros();

  if (currentMicros - previousMicros >= interval) {
    previousMicros = currentMicros;
    // Действия с высокой точностью
  }
}
```

## Аппаратные таймеры Arduino

Arduino содержит несколько аппаратных таймеров, которые можно использовать для точного управления временем:

### **Arduino Uno/Nano (ATmega328P):**

- Timer0 - 8-битный (используется для *delay()*, *millis()*, *micros()*)
- Timer1 - 16-битный
- Timer2 - 8-битный

### **Arduino Mega (ATmega2560):**

- 6 таймеров (Timer0-Timer5)

# ИСПОЛЬЗОВАНИЕ ПРЕРЫВАНИЙ ПО ТАЙМЕРУ

## *Прерывания по времени с использованием библиотек*

```
#include <TimerOne.h>

void setup() {
  Timer1.initialize(100000); // 100 ms
  Timer1.attachInterrupt(blinkLED); // Функция для вызова
  pinMode(LED_PIN, OUTPUT);
}

void blinkLED() {
  digitalWrite(LED_PIN, !digitalRead(LED_PIN));
}

void loop() {
  // Основной код выполняется свободно
}
```

Таблица 1 – Сравнение методов работы с таймерами

Метод	Точность	Блокирует выполнение	Сложность	Применение
delay()	низкая	да	очень низкая	Простые проекты
millis()	высокая	нет	низкая	Большинство проектов
micros()	очень высокая	нет	средняя	Высокоточные задачи
Аппаратные таймеры	максимальная	нет	высокая	Профессиональные решения

Понимание и правильное использование таймеров - ключевой навык для создания эффективных и отзывчивых проектов на Arduino. Метод на основе millis() является наиболее универсальным и подходит для большинства применений, в то время как аппаратные таймеры следует использовать для задач, требующих максимальной точности и производительности.

## ЗАДАНИЕ

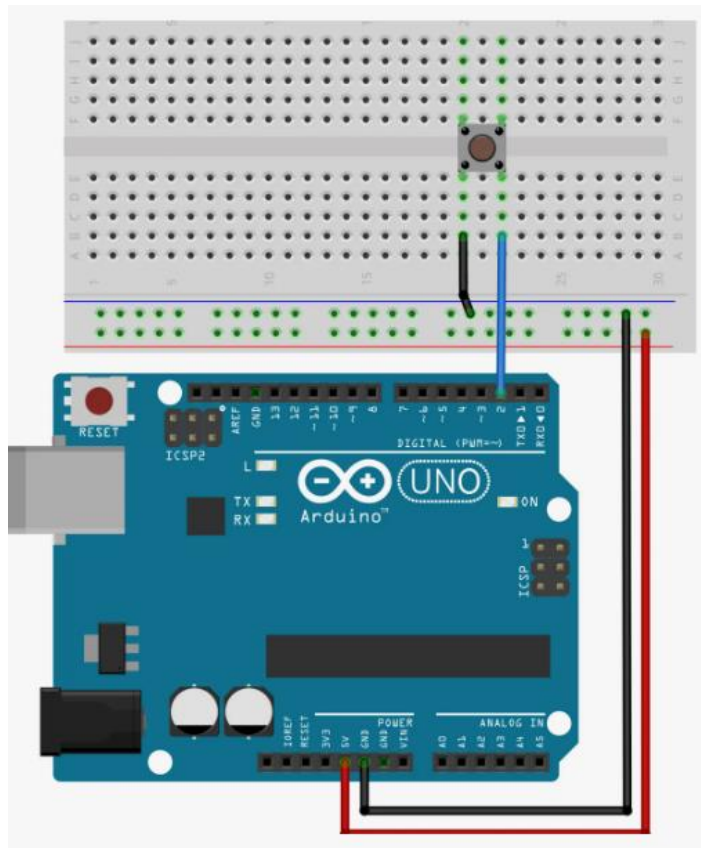


Рисунок 1 – Подключение кнопки

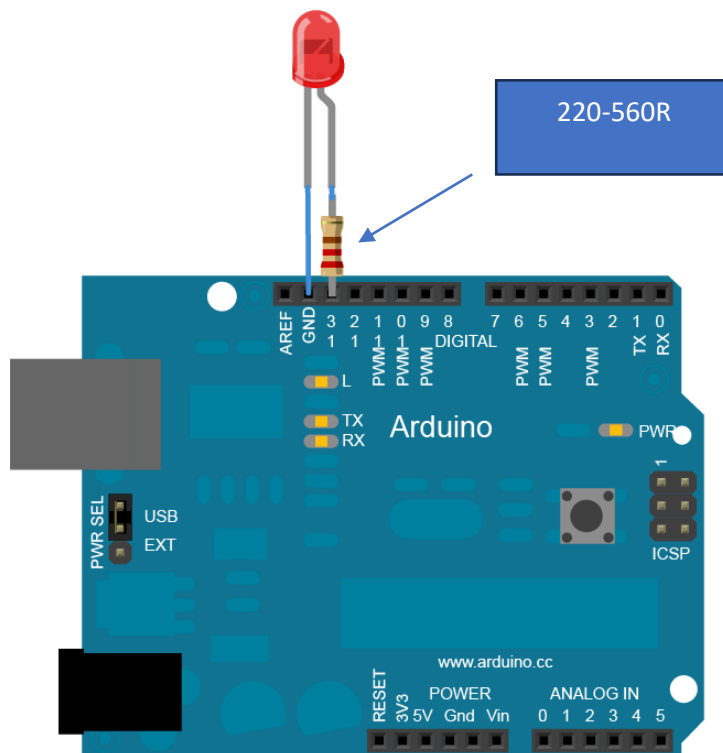


Рисунок 2 – Подключение светодиода

## Задание 1: Регулятор скорости мигания для 5 светодиодов

**Описание:** Две кнопки регулируют скорость мигания 5 светодиодов одновременно

```
const int LEDS[] = {2, 3, 4, 5, 6};
const int BUTTON_UP = 7;
const int BUTTON_DOWN = 8;
unsigned long previousMillis = 0;
int interval = 500;
const int MIN_INTERVAL = 50;
const int MAX_INTERVAL = 2000;
const int INTERVAL_STEP = 50;

void setup() {
  for (int i = 0; i < 5; i++) {
    pinMode(LEDS[i], OUTPUT);
  }
  pinMode(BUTTON_UP, INPUT_PULLUP);
  pinMode(BUTTON_DOWN, INPUT_PULLUP);
}

void loop() {
  unsigned long currentMillis = millis();

  // Обработка кнопок
  if (digitalRead(BUTTON_UP) == LOW) {
    interval = constrain(interval - INTERVAL_STEP,
MIN_INTERVAL, MAX_INTERVAL);
    delay(200);
  }
  if (digitalRead(BUTTON_DOWN) == LOW) {
    interval = constrain(interval + INTERVAL_STEP,
MIN_INTERVAL, MAX_INTERVAL);
    delay(200);
  }

  // Мигание светодиодов
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    for (int i = 0; i < 5; i++) {
      digitalWrite(LEDS[i], !digitalRead(LEDS[i]));
    }
  }
}
```

## Задание 2: Выбор светодиода по времени нажатия кнопки

**Описание:** Время удержания кнопки определяет, какой светодиод включится (1-5 секунд = светодиод 1-5)

```
const int LEDES[] = {2, 3, 4, 5, 6};
const int BUTTON = 7;
unsigned long pressStartTime = 0;
bool buttonPressed = false;

void setup() {
  for (int i = 0; i < 5; i++) {
    pinMode(LEDES[i], OUTPUT);
  }
  pinMode(BUTTON, INPUT_PULLUP);
}

void loop() {
  if (digitalRead(BUTTON) == LOW && !buttonPressed) {
    pressStartTime = millis();
    buttonPressed = true;
  }

  if (digitalRead(BUTTON) == HIGH && buttonPressed) {
    buttonPressed = false;
    unsigned long pressDuration = millis() - pressStartTime;
    int ledIndex = constrain(pressDuration / 1000, 1, 5) - 1;

    // Выключаем все светодиоды
    for (int i = 0; i < 5; i++) {
      digitalWrite(LEDES[i], LOW);
    }

    // Включаем выбранный светодиод
    digitalWrite(LEDES[ledIndex], HIGH);
  }
}
```

### Задание 3: Индивидуальная регулировка таймеров для каждого светодиода

**Описание:** Каждый светодиод имеет свою скорость мигания, регулируемую кнопками

```
const int LEDES[] = {2, 3, 4, 5, 6};
const int BUTTONS_UP[] = {7, 8, 9, 10, 11};
const int BUTTONS_DOWN[] = {12, 13, A0, A1, A2};
unsigned long previousMillis[5] = {0};
int intervals[5] = {500, 600, 700, 800, 900};

void setup() {
  for (int i = 0; i < 5; i++) {
    pinMode(LEDES[i], OUTPUT);
    pinMode(BUTTONS_UP[i], INPUT_PULLUP);
    pinMode(BUTTONS_DOWN[i], INPUT_PULLUP);
  }
}

void loop() {
  unsigned long currentMillis = millis();

  // Обработка кнопок для каждого светодиода
  for (int i = 0; i < 5; i++) {
    if (digitalRead(BUTTONS_UP[i]) == LOW) {
      intervals[i] = constrain(intervals[i] - 50, 100, 2000);
      delay(200);
    }
    if (digitalRead(BUTTONS_DOWN[i]) == LOW) {
      intervals[i] = constrain(intervals[i] + 50, 100, 2000);
      delay(200);
    }

    // Мигание светодиодов
    if (currentMillis - previousMillis[i] >= intervals[i]) {
      previousMillis[i] = currentMillis;
      digitalWrite(LEDES[i], !digitalRead(LEDES[i]));
    }
  }
}
```

#### Задание 4: Таймер обратного отсчета со светодиодной индикацией

**Описание:** Кнопка запускает таймер, светодиоды показывают оставшееся время

```
const int LEDS[] = {2, 3, 4, 5, 6};
const int BUTTON = 7;
unsigned long
  startTime = 0;
const long COUNTDOWN_TIME = 10000; // 10 секунд
bool countdownActive = false;

void setup() {
  for (int i = 0; i < 5; i++) {
    pinMode(LEDS[i], OUTPUT);
  }
  pinMode(BUTTON, INPUT_PULLUP);
}

void loop() {
  if (digitalRead(BUTTON) == LOW && !countdownActive) {
    startTime = millis();
    countdownActive = true;
    delay(200);
  }

  if (countdownActive) {
    unsigned long elapsed = millis() - startTime;
    unsigned long remaining = COUNTDOWN_TIME - elapsed;

    // Определяем, какие светодиоды включать
    int ledsToLight = map(remaining, 0, COUNTDOWN_TIME, 0, 5);

    // Обновляем светодиоды
    for (int i = 0; i < 5; i++) {
      digitalWrite(LEDS[i], i < ledsToLight ? HIGH : LOW);
    }

    // Проверяем завершение таймера
    if (remaining <= 0) {
      countdownActive = false;
      // Мигание по завершении
      for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 5; j++) digitalWrite(LEDS[j], HIGH);
        delay(200);
        for (int j = 0; j < 5; j++) digitalWrite(LEDS[j], LOW);
        delay(200);
      }
    }
  }
}
```