# Лабораторная работа №1 : Знакомство с ESP32

## Введение в встраиваемые системы и микроконтроллеры

**Встраиваемые системы: фундамент современной электроники**

В современном мире, где технологии пронизывают все аспекты жизни, **встраиваемые системы** (ВС) выполняют **критическую** роль. Эти специализированные компьютерные системы, **интегрированные** в более **сложные устройства**, не являются общецелевыми компьютерами. Вместо этого они **оптимизированы** для выполнения **определенных функций**, обеспечивая **исключительную эффективность** и **надежность** при решении **конкретных задач**.

**Ключевые характеристики встраиваемых систем:**

* **Ограниченные ресурсы:** ВС, как правило, работают с **ограниченными вычислительными ресурсами, памятью** и **энергопотреблением**.
* **Режим реального времени:** Многие из них должны функционировать в режиме **реального времени**, **мгновенно реагируя** на события.
* **Безотказность:** **Надежность** и **устойчивость к сбоям** являются **критическими** требованиями для ВС.
* **Экономичность:** Стоимость разработки и внедрения таких систем должна быть **минимальной**.

**Примеры встраиваемых систем:**

* **Системы управления двигателем (ECU) в автомобилях:** Обеспечивают **оптимальную работу** двигателя, **контролируя** его параметры и **повышая** топливную экономичность.
* **Смартфоны:** Мощные карманные компьютеры, сочетающие в себе **множество функций**, от связи и вычислений до фото- и видеосъемки.
* **Носимые устройства:** Фитнес-браслеты, умные часы и другие гаджеты, **отслеживающие** показатели здоровья и активности пользователя.
* **Стиральные машины:** Интеллектуальные устройства, **автоматически** подбирающие режим стирки в зависимости от типа ткани и степени загрязнения.
* **Маршрутизаторы:** Обеспечивают **доступ к сети Интернет** для различных устройств в доме или офисе.

**Микроконтроллеры: сердце встраиваемых систем**

**Микроконтроллеры** (МК) - это **интегральные схемы**, объединяющие на одном чипе все основные компоненты компьютера: **процессор, память, периферийные устройства**. Благодаря своим **небольшим размерам, низкой стоимости** и **низкому энергопотреблению** они идеально подходят для создания **компактных**, **экономичных** и **энергоэффективных** встраиваемых систем.

**Структура микроконтроллера:**

* **Центральный процессор (ЦП):** Выполняет **вычисления** и **обрабатывает инструкции**.
* **Память:** Хранит **программы** и **данные**, необходимые для работы системы.
* **Периферийные устройства:** Обеспечивают **взаимодействие** МК с **внешними устройствами**, такими как датчики, дисплеи, двигатели и т.д.

**Преимущества использования микроконтроллеров:**

* **Низкая стоимость:** МК **доступны** по цене, что **снижает** общую стоимость разработки и производства встраиваемых устройств.
* **Низкое энергопотребление:** **Минимальное энергопотребление** делает их идеальными для **портативных** и **автономных** устройств.
* **Компактность:** **Небольшие размеры** МК позволяют **легко встраивать** их в различные устройства.
* **Простота использования:** Разнообразие **средств разработки** и **библиотек** упрощает процесс **программирования** МК.

## Обзор контроллера ESP32 и его возможностей

**Введение**

ESP32 - это не просто микроконтроллер, а флагман в сфере Интернета вещей (IoT). Разработанный компанией Espressif Systems, он обладает беспрецедентным набором функций, делая его идеальным выбором для широкого спектра применений, от простых датчиков до сложных систем автоматизации.

**Технические характеристики:**

* **Процессор:** 32-битный Xtensa LX106, работающий на частоте до 240 МГц
* **Беспроводная связь:** Wi-Fi 802.11 b/g/n/d/e/i/k/r, Bluetooth 4.2 BR/EDR и BLE
* **Периферия:** 18 сенсорных выводов с 12-битным АЦП, 2 ЦАП, поддержка PWM на всех GPIO, SPI, I2C, UART, PWM, SDIO
* **Питание:** 3,3 В (может работать от батареи)
* **Среды разработки:** Arduino IDE, Espressif IDF, PlatformIO

**Функциональные возможности:**

* **Сбор данных:** 18 сенсорных выводов позволяют подключать различные датчики для измерения температуры, давления, освещенности, движения и других параметров.
* **Управление устройствами:** 2 ЦАП и поддержка PWM на всех GPIO-выводах обеспечивают управление исполнительными устройствами, такими как светодиоды, реле, сервоприводы и двигатели.
* **Подключение:** Разнообразие интерфейсов (SPI, I2C, UART, PWM, SDIO) facilitates easy connection to a wide range of peripherals.
* **Энергоэффективность:** Низкое энергопотребление позволяет ESP32 работать от батареи, что делает его идеальным для автономных устройств.
* **Безопасность:** ESP32 поддерживает аппаратное шифрование AES, обеспечивая безопасность данных.

**Преимущества:**

* **Мощность и производительность:** ESP32 обеспечивает высокую вычислительную мощность, достаточную для требовательных задач.
* **Многофункциональность:** Широкий набор функций делает ESP32 универсальным решением для различных применений.
* **Простота использования:** ESP32 можно программировать с помощью различных сред разработки, что делает его доступным для пользователей с разным уровнем подготовки.
* **Доступная цена:** ESP32 является экономичным решением, что делает его привлекательным для широкого круга пользователей.

**Сферы применения:**

* **Умный дом:** ESP32 может использоваться для создания систем управления освещением, термостатов, охранных систем и других устройств умного дома.
* **Носимые устройства:** Благодаря компактным размерам и низкому энергопотреблению ESP32 идеально подходит для создания фитнес-браслетов, умных часов и других носимых устройств.
* **Промышленные решения:** ESP32 может использоваться для мониторинга и управления промышленными процессами, сбора данных с датчиков и обеспечения удаленного доступа к оборудованию.
* **Сельское хозяйство:** ESP32 может использоваться для контроля систем орошения, мониторинга состояния окружающей среды и оптимизации сельскохозяйственных операций.

**Расширенная информация:**

**1. Распиновка микроконтроллера ESP32**

| Номер GPIO | Функциональность | Описание |
| --- | --- | --- |
| 0 | GPIO0 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 1 | GPIO1 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 2 | GPIO2 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 3 | GPIO3 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 4 | GPIO4 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 5 | GPIO5 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 6 | GPIO6 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 7 | GPIO7 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 8 | GPIO8 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 9 | GPIO9 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 10 | GPIO10 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 11 | GPIO11 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 12 | GPIO12 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 13 | GPIO13 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 14 | GPIO14 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 15 | GPIO15 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 16 | GPIO16 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 17 | GPIO17 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 18 | GPIO18 | Многофункциональный вывод (вход/выход, PWM, I2C, SPI, UART) |
| 19 | Touch Sensor (T0) | Сенсорный вход |
| 20 | Touch Sensor (T1) | Сенсорный вход |
| 21 | Touch Sensor (T2) | Сенсорный вход |
| 22 | Touch Sensor (T3) | Сенсорный вход |
| 23 | Touch Sensor (T4) | Сенсорный вход |
| 24 | CLK (SPI) | Часы SPI |
| 25 | MISO (SPI) | Мастер вход последовательных данных SPI |
| 26 | MOSI (SPI) | Мастер выход последовательных данных SPI |
| 27 | CS (SPI) | Выбор чипа SPI |
| 28 | RXD0 (UART0) | Прием данных UART0 |
| 29 | TXD0 (UART0) | Передача данных UART0 |
| 30 | RXD1 (UART1) | Прием данных UART1 |
| 31 | TXD1 (UART1) | Передача данных UART1 |
| 32 | ADC1\_CH0 | Аналоговый вход 0 |
| 33 | ADC1\_CH1 | Аналоговый вход 1 |
| 34 | ADC1\_CH2 | Аналоговый вход 2 |
| 35 | ADC1\_CH3 | Аналоговый вход 3 |
| 36 | ADC1\_CH4 | Аналоговый вход 4 |
| 37 | ADC1\_CH5 | Аналоговый вход 5 |
| 38 | ADC1\_CH6 | Аналоговый вход 6 |
| 39 | ADC1\_CH7 | Аналоговый вход 7 |
| 40 | VREF | Внутреннее |

## Установка IDE и настройка для работы с ESP32

**Введение**

Для разработки под контроллер ESP32 используется несколько интегрированных сред разработки (IDE).

В этой инструкции мы рассмотрим два популярных варианта:

* **Arduino IDE:** Простая и удобная среда, идеально подходящая для начинающих пользователей.
* **Espressif IDF:** Мощная среда, предоставляющая больше возможностей и контроля над микроконтроллером.

**1. Установка Arduino IDE**

**1.1 Скачивание IDE:**

* Перейдите на сайт Arduino IDE: <https://support.arduino.cc/hc/en-us/articles/360019833020-Download-and-install-Arduino-IDE>
* Выберите операционную систему вашего компьютера (Windows, macOS или Linux) и нажмите кнопку “Скачать”.
* Запустите скачанный файл и следуйте инструкциям мастера установки.

**1.2 Настройка ESP32 в Arduino IDE:**

* Откройте Arduino IDE.
* Перейдите в меню “Инструменты” > “Плата”.
* Выберите “Менеджер плат”.
* Введите “ESP32” в поле поиска.
* Найдите “ESP32 by Espressif Systems” и нажмите кнопку “Установить”.
* Выберите “ESP32DevKitC” или другую соответствующую плату ESP32 в меню “Инструменты” > “Плата”.
* Выберите порт, к которому подключен ESP32, в меню “Инструменты” > “Порт”.

**1.3 Примеры кода для ESP32:**

* Множество примеров кода для ESP32 доступно в стандартной библиотеке Arduino: <https://docs.arduino.cc/built-in-examples>
* Дополнительные примеры можно найти на сайте Espressif: <https://github.com/espressif/arduino-esp32>

**2. Установка Espressif IDF**

**2.1 Установка Git:**

* Git - это система контроля версий, необходимая для работы с Espressif IDF.
* Скачайте и установите Git для вашей операционной системы: <https://git-scm.com/downloads>

**2.2 Настройка среды разработки:**

* Скачайте Espressif IDF: <https://www.espressif.com/en/support/download/sdks-demos>
* Распакуйте скачанный архив.
* Добавьте путь к папке IDF\_PATH в вашу системную переменную PATH.
* Установите Ninja Build System:
  + **Windows:** Скачайте и установите Ninja из <https://ninja-build.org/>.
  + **macOS:** Используйте brew: brew install ninja
  + **Linux:** Используйте ваш менеджер пакетов (apt, yum, etc.)

**2.3 Настройка ESP32 в Espressif IDF:**

* Откройте терминал и перейдите в папку с примерами ESP32 IDF: cd <IDF\_PATH>/examples
* Выберите пример, который хотите использовать (например, blink).
* Запустите команду idf.py menuconfig для настройки параметров проекта.
* Запустите команду idf.py build для компиляции проекта.
* Запустите команду idf.py flash для загрузки прошивки на ESP32.

**2.4 Дополнительные ресурсы:**

* Документация Espressif IDF: <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/index.html>
* Примеры кода ESP32 IDF: <https://github.com/espressif/esp-idf/tree/master/examples>

**3. Рекомендации:**

* Для начинающих пользователей рекомендуется начать с Arduino IDE, так как она более проста в использовании.
* Espressif IDF предлагает больше возможностей и контроля, но требует больше знаний и опыта.
* Изучите официальную документацию и примеры кода, чтобы ознакомиться с возможностями ESP32 и выбранной вами IDE.
* При возникновении проблем обратитесь к сообществам ESP32 на форумах и в социальных сетях.

## Написание и загрузка простейшей программы “Hello, ESP32!”

**Задание:**

Написать и загрузить на ESP32 простую программу, которая будет выводить на подключенный к ней Serial Monitor сообщение “Hello, ESP32!”.

**1. Подготовка:**

**1.1. Необходимое оборудование:**

* Микроконтроллер ESP32
* USB-кабель
* Компьютер

**1.2. Установка программного обеспечения:**

* **Arduino IDE:** <https://support.arduino.cc/hc/en-us/articles/360019833020-Download-and-install-Arduino-IDE>
* **Espressif IDF:** <https://dl.espressif.com/dl/esp-idf/>

**2. Реализация программы:**

**2.1. С Arduino IDE:**

**2.1.1. Подготовка:**

* Подключите ESP32 к компьютеру USB-кабелем.
* Откройте Arduino IDE.
* Выберите плату ESP32.

**2.1.2. Описание функций для Arduino IDE:**

* void setup(): Эта функция выполняется один раз при запуске программы. В ней обычно осуществляется инициализация аппаратных ресурсов, таких как GPIO-пины, датчики и исполнительные устройства. В данном случае она инициализирует последовательный порт для связи с компьютером.
* void loop(): Эта функция выполняется бесконечно после завершения функции setup(). В ней обычно размещается основной код программы, который должен выполняться в течение всего времени работы устройства. Например, здесь можно разместить код для мониторинга состояния датчиков, управления исполнительными устройствами или обработки входных данных.
* Serial.begin(115200): Эта функция инициализирует последовательный порт с заданной скоростью передачи данных (в данном случае 115200 бод). Это позволяет отправлять и принимать данные с компьютера через USB-кабель.
* Serial.println("Hello, ESP32!"): Эта функция отправляет сообщение “Hello, ESP32!” в последовательный порт. println() добавляет символ новой строки (\n) в конце сообщения.
* delay(1000): Эта функция приостанавливает выполнение программы на 1000 миллисекунд (1 секунду).

**2.2 С Espressif IDF:**

**2.2.1. Подготовка:**

* Подключите ESP32 к компьютеру USB-кабелем.
* Откруйте Espressif IDF.
* Создайте проект в Espressif IDF.

**2.2.2. Описание функций для Espressif IDF:**

* void setup(): (см. описание выше)
* void loop(): (см. описание выше)
* Serial.begin(115200): (см. описание выше)
* printf("Hello, ESP32!\n"): Эта функция отправляет сообщение “Hello, ESP32!” в последовательный порт. \n добавляет символ новой строки (\n) в конце сообщения.
* vTaskDelay(1000 / portTICK\_PERIOD\_MS): Эта функция приостанавливает выполнение задачи на 1000 миллисекунд (1 секунду). portTICK\_PERIOD\_MS - это константа, определяющая период одного тика в миллисекундах.

**3. Проверка работы:**

* Откройте Serial Monitor в Arduino IDE или используйте терминал для подключения к ESP32 по UART.
* Вы должны увидеть сообщение “Hello, ESP32!”.

**Код для Arduino IDE:**

void setup() {  
 // Инициализация Serial Monitor  
 Serial.begin(115200);  
}  
  
void loop() {  
 // Вывод сообщения "Hello, ESP32!" в Serial Monitor  
 Serial.println("Hello, ESP32!");  
  
 // Задержка 1 секунды  
 delay(1000);  
}

**Код для Espressif IDF:**

#include <stdio.h>  
#include <esp8266.h>  
  
void setup() {  
 // Инициализация Serial Monitor  
 Serial.begin(115200);  
}  
  
void loop() {  
 // Вывод сообщения "Hello, ESP32!" в Serial Monitor  
 printf("Hello, ESP32!\n");  
  
 // Задержка 1 секунды  
 vTaskDelay(1000 / portTICK\_PERIOD\_MS);  
}