**各系列MCU芯片基于的内核，不同的内核有不同的汇编**

# 一．MCU芯片编译开发环境及烧录工具

## 1.stc系列

**STC和STC32都是STC（宏晶科技）公司推出的单片机系列**，但它们有显著的不同。STC系列主要基于8051架构，而STC32系列基于ARM Cortex-M0/M3内核。

由于STC32是基于ARM内核，所以也支持GNU交叉编译工具链

GNU交叉工具链官方下载地址：<https://developer.arm.com/downloads/-/gnu-rm>

### STC系列芯片有哪些

STC（宏晶科技）系列芯片主要基于8051架构，涵盖了多种型号，广泛应用于嵌入式系统中。以下是一些主要的STC系列芯片及其特点：

#### 1. STC89系列

**特点**：

* 基于经典的8051架构。
* 频率范围：一般为12 MHz到35 MHz。
* 低功耗设计。
* 适合普通的控制应用。

**常见型号**：

* STC89C51RC/RD+：增强型51单片机，带有8K/16K的Flash存储器。
* STC89LE52：低功耗型号，带有52K的Flash存储器。

#### 2. STC12系列

**特点**：

* 基于8051架构，但增强了许多功能。
* 频率范围：最高可达35 MHz。
* 更高的处理能力和更多的I/O口。
* 支持ISP（In-System Programming）和IAP（In-Application Programming）。

**常见型号**：

* STC12C5A60S2：带有60K Flash存储器，丰富的I/O口。
* STC12LE5A60S2：低功耗版本，适合电池供电的应用。

#### 3. STC15系列

**特点**：

* 基于增强型8051架构，性能进一步提升。
* 频率范围：最高可达35 MHz。
* 支持ISP和IAP功能。
* 集成更多的外设模块，如ADC、PWM、UART等。

**常见型号**：

* STC15F204EA：带有4K Flash存储器，丰富的外围接口。
* STC15L2K60S2：带有60K Flash存储器，低功耗版本。

#### 4. STC8系列

**特点**：

* 基于高性能的8051架构，性能显著提升。
* 频率范围：最高可达60 MHz。
* 支持ISP和IAP功能。
* 集成更多高级外设，如增强型PWM、比较器等。

**常见型号**：

* STC8A8K64S4A12：带有64K Flash存储器，支持高速运行。
* STC8G1K08-38I：低功耗版本，适合电池供电的应用。

#### 5. STC10系列

**特点**：

* 基于增强型8051架构，性能中等。
* 频率范围：最高可达40 MHz。
* 支持ISP和IAP功能。
* 集成基本的外设模块。

**常见型号**：

* STC10F04XE：带有4K Flash存储器，适合简单控制应用。
* STC10L8XE：低功耗版本，带有8K Flash存储器。

#### 6. STC11系列

**特点**：

* 基于增强型8051架构，性能适中。
* 频率范围：最高可达40 MHz。
* 支持ISP和IAP功能。
* 集成基本的外设模块。

**常见型号**：

* STC11F32XE：带有32K Flash存储器，适合较复杂控制应用。
* STC11L16XE：低功耗版本，带有16K Flash存储器。

#### 7. STC32系列

**特点**：

* 基于ARM Cortex-M3内核，性能大幅提升。
* 频率范围：最高可达72 MHz。
* 具有更高的处理能力和丰富的外设支持。
* 适合高性能需求的应用，如物联网设备、工业控制等。

**常见型号**：

* STC32G12K128：带有128K Flash存储器，高性能型号。
* STC32A8K64：带有64K Flash存储器，适用于较高性能需求的应用。

#### 总结

STC系列芯片涵盖了从经典的8051架构到增强型8051架构，再到高性能的ARM Cortex-M3架构，满足不同应用需求。开发者可以根据具体项目的性能要求、功耗需求和成本预算选择合适的STC系列芯片。

### STC开发编译环境

STC系列芯片开发编译环境有多种选择，涵盖了从入门到专业级别的工具。以下是一些常见的开发编译环境：

#### 1. Keil C51(免费评估版，收费版)

**特点**：

* **流行度**：Keil C51是最流行的8051系列单片机开发环境之一。
* **功能强大**：支持丰富的调试功能和优化选项。
* **集成度高**：IDE中集成了编辑、编译和调试功能。

**适用范围**：广泛用于各种基于8051架构的MCU，包括STC89、STC12、STC15等。

#### 2. STC-ISP（官方免费）

**特点**：

* **官方工具**：STC官方提供的烧录软件，用于将编译后的程序下载到STC系列芯片中。
* **易于使用**：界面简单，操作方便。
* **功能专一**：专注于程序下载和芯片配置。

**适用范围**：所有STC系列单片机的程序烧录。

#### 3. SDCC (Small Device C Compiler)（开源）

**特点**：

* **开源免费**：一个开源的C编译器，支持8051系列单片机。
* **跨平台**：支持多种操作系统，包括Windows、Linux和macOS。
* **灵活性高**：支持标准C编程，兼容性强。

**适用范围**：可以用于STC89、STC12、STC15等系列的开发。

#### 4. Arduino IDE(免费)

**特点**：

* **易于上手**：Arduino IDE简洁易用，适合初学者。
* **支持STC**：通过第三方库和插件，可以支持STC系列单片机的开发。
* **跨平台**：支持Windows、Linux和macOS。

**适用范围**：适合入门级项目和快速原型开发。

#### 5. PlatformIO（免费）

**特点**：

* **现代化环境**：支持多种开发板和平台，功能强大且灵活。
* **集成度高**：集成了编辑、编译、调试和上传功能。
* **支持STC**：通过插件和第三方库，PlatformIO也可以支持STC系列单片机。

**适用范围**：适合专业开发和复杂项目。

#### 6. MIDE-51(免费)

**特点**：

* **专为8051设计**：一个简单易用的8051系列单片机开发环境。
* **轻量级**：占用资源少，适合中小型项目。
* **集成度**：集成了编辑、编译和调试功能。

**适用范围**：适合STC系列的初学者和简单项目开发。

#### 7. IAR Embedded Workbench(收费)

**特点**：

* **专业级工具**：支持多种微控制器，功能强大。
* **优化性能**：提供高级优化选项，提高代码效率。
* **广泛支持**：支持8051架构的开发，适用于STC系列单片机。

**适用范围**：适合高要求和专业级的嵌入式开发。

#### 8. μVision IDE (Keil)（免费评估版，收费版）

**特点**：

* **全面集成**：一个全面的开发环境，集成了编辑、编译、调试和仿真功能。
* **高级调试**：支持多种调试接口和硬件仿真器。
* **广泛支持**：支持8051架构，包括STC系列单片机。

**适用范围**：适合复杂项目和专业开发。

#### 总结

开发STC系列芯片的编译环境有很多选择，从经典的Keil C51到现代的PlatformIO和Arduino IDE，开发者可以根据具体项目需求和个人偏好选择合适的开发工具。无论是专业级开发还是初学者学习，都能找到适合的环境来进行STC系列单片机的开发。

### STC烧写软件及硬件

STC-ISP

**stc官网是哪个**

STC的官网有几个不同的网址，每个都提供关于STC单片机的不同信息。以下是STC的主要官网：

1. **STC MCU官方资源网** (www.stcmcudata.com): 这个网站提供STC单片机的各种资源，包括手册、程序、原理图和库函数等资料。它强调STC作为32位8051单片机的创新者和ISP/IAP技术的创新者 **1** 。
2. **STC MCU全球第一品牌官网** (www.stcmcu.com): 这个网站宣称STC是全球最大的8051单片机设计公司，提供抗干扰性强、加密功能强大的产品。网站上有关于STC单片机的最新资料、选型指南、开发工具和免费样品等信息 **2** 。
3. **STC Microcontroller官网** (www.stcmicro.com): 这个网站提供STC单片机的最新动态、产品推荐、数据手册等。它还介绍了STC的各类产品，如STC8S003芯片、STC25Q40系列宽电压SPI NOR FLASH芯片等 **3** 。

这些网站都是STC的官方资源，提供详细的产品信息和相关服务。您可以根据自己的需要访问这些网站以获取更多关于STC单片机的详细信息。

## 2.stm系列

STMicroelectronics（ST）是一家知名的半导体制造商，它生产了两类不同的产品系列：STM（Standard Microsystems）和STM32。

**STM（Standard Microsystems）**

STM系列产品主要是集成了各种标准接口和功能的微控制器（Microcontrollers）。它们通常用于相对简单的控制任务，例如传感器接口、通信接口等。STM系列微控制器广泛应用于各种嵌入式系统中，如家电、消费电子、工业自动化等领域。

**STM32**

STM32则是STMicroelectronics推出的一系列高性能微控制器，主要基于ARM Cortex-M处理器核心。STM32系列微控制器提供了更高的计算性能、更多的外设接口以及更丰富的功能特性，适用于需要复杂算法和高性能处理的应用场景。STM32系列被广泛用于工业控制、智能家居、汽车电子、医疗设备等需要高度可靠性和性能的领域。

### Stm系列芯片有哪些

STM系列芯片是由STMicroelectronics（意法半导体）公司生产的一系列微控制器（MCU），涵盖了多种不同的系列和型号，适用于各种嵌入式应用。以下是一些主要的STM系列芯片及其特点：

#### 1. STM32系列

**特点**：

* **基于ARM Cortex-M内核**：主要包括Cortex-M0, M0+, M3, M4, M7等版本。
* **丰富的外设**：包括GPIO、ADC、DAC、UART、SPI、I2C、PWM等。
* **支持多种通信协议**：如CAN、Ethernet、USB等。
* **广泛的内存配置**：包括Flash存储器和SRAM。
* **适用于高性能和低功耗需求**。

**常见系列**：

* **STM32F0**：低成本、低功耗，适合简单应用。
* **STM32F1**：适用于性价比高的应用。
* **STM32F2**：高性能，带有外设加速器。
* **STM32F3**：集成了数字信号处理功能。
* **STM32F4**：高性能，带有DSP和FPU支持。
* **STM32F7**：高性能、高集成度。
* **STM32L0**：极低功耗，适用于电池供电应用。
* **STM32L1**：低功耗，带有射频和USB支持。
* **STM32L4**：低功耗、高性能，支持数字信号处理和FPU。

#### 2. STM8系列

**特点**：

* **基于8位STM8内核**。
* **低成本、高性能**。
* **丰富的外设**：包括ADC、UART、SPI、I2C等。
* **适用于成本敏感和资源有限的应用**。

**常见系列**：

* **STM8S**：低成本、中等性能，适用于基本控制应用。
* **STM8L**：低功耗，适用于电池供电应用。

#### 3. STM32MP系列

**特点**：

* **基于多核处理器**：结合了Cortex-A和Cortex-M内核。
* **适用于复杂的嵌入式应用**。
* **丰富的外设和通信接口**。

**常见系列**：

* **STM32MP1**：集成了Cortex-A7和Cortex-M4内核，适合多媒体和处理能力要求高的应用。

#### 4. STM32WL系列

**特点**：

* **集成LoRa无线通信功能**。
* **低功耗**。
* **适用于长距离低功耗通信应用**。

#### 5. STM32H系列

**特点**：

* **高性能**：支持高达550 MHz的运行频率。
* **高集成度**：内置DSP和FPU支持。
* **广泛的外设和通信接口**。

#### 6. STM32G系列

**特点**：

* **中高性能**：适用于性能要求较高的应用。
* **支持数字信号处理和FPU**。

#### 总结

STM系列芯片涵盖了从低成本低功耗到高性能高集成度的多种需求。选择合适的STM系列芯片取决于项目的具体需求，包括性能要求、功耗预算、外设和通信接口需求等。STMicroelectronics提供了丰富的开发工具和支持，使得开发者能够有效地利用STM系列芯片开发各种嵌入式应用。

### Stm开发编译环境

当涉及到开发STM系列芯片时，可以使用多种开发和编译环境，这些环境通常根据开发者的需求和技术栈选择。以下是一些常见的STM芯片开发编译环境：

#### 1. STM32CubeIDE(官方免费)

* **特点**：官方推荐的集成开发环境（IDE），基于Eclipse平台。
* **功能**：包括代码编辑、编译、调试、仿真和固件更新等。
* **优势**：全面支持STM32系列芯片，提供STM32CubeMX的集成，能够快速生成初始化代码和配置文件。

#### 2. Keil MDK (Microcontroller Development Kit)（免费，商业）

* **特点**：流行的ARM开发工具，包括支持STM32系列的编译器、调试器和仿真器。
* **功能**：强大的优化和调试功能，适合专业开发和复杂项目。
* **适用**：广泛支持STM32系列的各种芯片和外设。

#### 3. IAR Embedded Workbench for ARM(收费)

* **特点**：专业级的嵌入式开发工具，支持多种ARM系列芯片，包括STM32。
* **功能**：高级优化和调试功能，适合需要高效率和稳定性的项目开发。

#### 4. GNU ARM Embedded Toolchain (GCC)（免费）

* **特点**：开源的ARM交叉编译工具链，兼容性强。
* **功能**：适合裸机开发和自定义构建环境，支持多种STM32系列芯片。
* **GNU交叉工具链官方下载地址**：<https://developer.arm.com/downloads/-/gnu-rm>

#### 5. STM32CubeMX + Third-party IDEs（免费）

* **特点**：STM32CubeMX是STM32系列的配置工具，可生成初始化代码和配置文件。
* **集成**：可以与其他第三方IDE如Eclipse、Visual Studio等集成使用，方便开发和调试。

#### 6. PlatformIO（免费）

* **特点**：现代化的开发平台，支持多种开发板和MCU，包括STM32系列。
* **功能**：集成了编辑、编译、调试和上传功能，支持多种硬件平台和操作系统。

#### 7. Arduino IDE with STM32 Core（免费）

* **特点**：简易上手的IDE，通过安装STM32 Core插件可以支持STM32系列芯片。
* **适用**：适合初学者和快速原型开发，支持STM32的基本功能和库。

#### 8. SEGGER Embedded Studio（免费版，收费版）

* **特点**：专业的嵌入式开发工具，支持多种调试接口和硬件仿真器。
* **功能**：适合复杂项目的开发和调试，支持多种ARM系列芯片，包括STM32。

#### 9. Atollic TrueSTUDIO (现已并入STM32CubeIDE)（免费）

* **特点**：基于Eclipse的开发环境，提供丰富的插件和功能支持，适合STM32系列开发。

#### 10. 裸机开发工具链

* **特点**：如Makefile、GCC工具链等，适合需要定制化开发环境和更精简控制的项目。
* **适用**：可以结合STM32提供的标准外设库（STM32 Standard Peripheral Library）或HAL库进行裸机开发。

选择适合的开发环境取决于项目的具体需求，包括开发团队的技术栈、项目的性能和功能需求、硬件资源等因素。STMicroelectronics提供了多种官方推荐的开发工具和支持，开发者可以根据自身需求选择最合适的工具和环境进行STM系列芯片的开发。

### STM烧写软件及硬件

**Stm官网是哪个**

STM的官网主要是指意法半导体（STMicroelectronics）的官方网站。意法半导体是一家全球性的半导体公司，提供各种半导体产品和解决方案。以下是意法半导体的一些主要官网：

1. **意法半导体主官网** (www.st.com.cn): 这个网站提供意法半导体的全面信息，包括他们的产品、工具与软件、资源、视频、解决方案、应用、出版刊物等。它涵盖了公司的各种半导体产品，如基于Arm的MCU（微控制器单元）等，并强调公司在智能出行、电源与能源、云连接的自主化设备等领域的创新 **1** 。
2. **意法半导体英文官网** (www.st.com): 这个网站与中文官网类似，提供意法半导体的全面信息，包括产品、工具与软件、资源、视频、解决方案、应用、出版刊物等，但内容以英文为主 **2** 。
3. **STM32 MCU单片机官网** (www.stmcu.com.cn): 这个网站专注于意法半导体的STM32系列32位微控制器，这些微控制器基于Arm® Cortex®-M处理器。网站提供了关于STM32系列微控制器的详细信息，包括产品特点、选型手册、技术支持等 **3** 。

这些网站都是意法半导体的官方资源，提供详细的产品信息和相关服务。您可以根据自己的需要访问这些网站以获取更多关于意法半导体的产品和技术信息。

## 3.esp芯片

### Esp系列芯片有哪些

ESP系列芯片是由乐鑫（Espressif Systems）公司推出的一系列低功耗、高集成度的Wi-Fi和蓝牙解决方案。这些芯片广泛用于物联网（IoT）设备、智能家居、工业控制等领域。以下是一些主要的ESP系列芯片：

#### 1. ESP8266系列

* **特点**：最早的ESP系列芯片，集成了Wi-Fi功能。
* **适用**：适合对成本和功耗要求较低的简单物联网应用。

**常见型号**：

* ESP8266-01、ESP8266-12E、ESP8266-12F等。

#### 2. ESP32系列

* **特点**：升级版的ESP系列，集成了Wi-Fi和蓝牙功能，性能更强大。
* **适用**：适合需要更复杂功能和更高性能的物联网和嵌入式应用。

**常见型号**：

* ESP32-WROOM系列：如ESP32-WROOM-32、ESP32-WROOM-32D等。
* ESP32-WROVER系列：如ESP32-WROVER、ESP32-WROVER-B等。
* ESP32-S系列：如ESP32-S2、ESP32-S3等。

#### 3. ESP8285

* **特点**：类似于ESP8266，但集成了闪存存储器。
* **适用**：适合对PCB空间和成本要求较高的设计。

**常见型号**：

* ESP8285。

#### 4. ESP32-C3

* **特点**：基于RISC-V架构的Wi-Fi和蓝牙解决方案，功耗更低。
* **适用**：适合需要低功耗和高性能的物联网设备。

**常见型号**：

* ESP32-C3。

#### 5. ESP32-S2

* **特点**：集成了Wi-Fi功能，性能较ESP8266和ESP32-C3更强。
* **适用**：适合对功耗要求较高的物联网设备和应用。

**常见型号**：

* ESP32-S2。

#### 6. ESP32-S3

* **特点**：升级版的ESP32-S2，集成了更多的外设和功能。
* **适用**：适合对性能和功能有较高要求的物联网和嵌入式设备。

**常见型号**：

* ESP32-S3。

#### 总结

ESP系列芯片以其低功耗、高集成度和丰富的功能特性在物联网领域中得到广泛应用。不同型号的ESP芯片适用于不同的应用场景和需求，开发者可以根据项目的具体要求选择合适的ESP芯片型号进行开发。

### Esp开发编译环境

开发ESP系列芯片的编译环境可以选择多种工具和平台，以下是一些常见的开发编译环境：

#### 1. Espressif IoT Development Framework (ESP-IDF)（官方免费）

* **特点**：
  + **官方支持**：由乐鑫公司提供的官方开发框架。
  + **高度定制化**：支持灵活的配置和定制，适合复杂的应用开发。
  + **丰富的示例和文档**：提供详细的示例代码和文档。

#### 2. Arduino IDE with ESP8266/ESP32 Core（免费）

* **特点**：
  + **易于上手**：适合初学者和快速原型开发。
  + **支持多种ESP系列芯片**：通过安装对应的ESP8266或ESP32 Core插件，支持不同型号的ESP芯片开发。
  + **大量的社区支持和扩展库**。

#### 3. PlatformIO（免费）

* **特点**：
  + **现代化的开发平台**：支持多种开发板和MCU，包括ESP系列。
  + **集成度高**：提供了编辑、编译、调试和上传等功能。
  + **跨平台支持**：适用于Windows、Linux和macOS等操作系统。

#### 4. Espressif Visual Studio Code Extension (Espressif VS Code Extension)

* **特点**：
  + **基于Visual Studio Code**：支持丰富的插件和扩展。
  + **集成ESP-IDF和Arduino Core**：可以集成ESP-IDF或Arduino Core进行开发。
  + **提供丰富的功能和调试支持**。

#### 5. Eclipse IDE with ESP8266/ESP32 Plugin

* **特点**：
  + **传统的开发环境**：适合习惯使用Eclipse的开发者。
  + **插件支持**：通过安装对应的ESP8266或ESP32插件，实现对ESP系列芯片的开发和调试。
  + **强大的编辑和项目管理功能**。

#### 6. Mongoose OS

* **特点**：
  + **基于JavaScript和C/C++**：支持使用JavaScript进行开发。
  + **简化的固件开发**：提供了一套简化的开发流程和API。
  + **适用于快速原型和商业项目**。

#### 7. MicroPython

* **特点**：
  + **Python语言支持**：支持使用Python语言进行开发和编程。
  + **适用于教育和快速原型开发**：提供了简单而强大的开发体验。

#### 8. AWS IoT Core、Google Cloud IoT Core等云服务平台

* **特点**：
  + **云端集成**：与云服务平台集成，支持在ESP系列芯片上开发物联网应用。
  + **提供丰富的SDK和工具**：支持设备管理、数据传输和分析等功能。

#### 9. 裸机开发工具链

* **特点**：
  + **低级别的开发**：如使用Makefile、GCC等工具链进行裸机编程。
  + **适用于需要定制和控制的应用场景**。

选择适合的开发环境取决于开发者的技术栈、项目需求以及个人偏好。每种开发环境都有其独特的优势和适用场景，开发者可以根据具体情况选择最适合的工具来开发ESP系列芯片的应用。

### Esp烧写软件及硬件

Esp官网是哪个

ESP的官网主要是指乐鑫科技（Espressif Systems）的官方网站，特别是针对其著名的ESP32系列。乐鑫科技是一家专注于无线通信和物联网技术的公司，其官网提供了关于ESP32和其他相关产品的详细信息。以下是乐鑫科技的主要官网：

1. **乐鑫科技官网** (www.espressif.com): 这个网站专注于乐鑫科技的产品，特别是ESP32系列。ESP32是一款集成了Wi-Fi和蓝牙功能的系统级芯片（SoC），适用于各种物联网应用。网站提供了关于ESP32的详细信息，包括产品特点、文档、应用案例等 **1** 。

乐鑫科技的官网是获取其产品信息和相关资源的官方渠道。如果您对ESP32或其他乐鑫科技的产品感兴趣，可以访问这个网站以获取更多详细信息。

## 4.AVR

**开发编译环境**

AVR是一种由Atmel（现在被Microchip收购）推出的8位微控制器系列，常见的开发编译环境和工具如下：

### 1. Atmel Studio

**Atmel Studio** 是由Microchip官方提供的集成开发环境（IDE），专门用于AVR和ARM微控制器的开发。

* **特点**：
  + 提供完整的开发工具链，包括编译器、调试器和仿真器。
  + 集成了Atmel START工具，用于快速配置和初始化代码生成。
  + 支持AVR Studio 4/5/6的项目导入和转换。

### 2. MPLAB X IDE + AVR Plugin

**MPLAB X IDE** 是Microchip官方推出的开源跨平台IDE，通过安装AVR插件，可以支持AVR系列微控制器的开发。

* **特点**：
  + 支持多种Microchip微控制器的开发，包括AVR系列。
  + 集成了丰富的插件和工具支持。
  + 提供了强大的调试功能和性能分析工具。

### 3. CodeVisionAVR

**CodeVisionAVR** 是一款商业的AVR开发环境，提供了专业的开发工具和支持。

* **特点**：
  + 提供了高度优化的AVR编译器和调试器。
  + 集成了图形化配置工具，支持快速开发和调试。
  + 提供了丰富的例程和库文件支持。

### 4. AVR-GCC

**AVR-GCC** 是GNU Compiler Collection (GCC) 的一部分，提供了开源的C/C++编译器，支持AVR系列微控制器。

* **特点**：
  + 免费开源，支持跨平台运行。
  + 提供了高度优化的编译器。
  + 可以与多种IDE集成，如Eclipse、Visual Studio Code等。

### 5. Eclipse + AVR Plugin

**Eclipse** 是一款流行的开源跨平台IDE，通过安装AVR插件，可以支持AVR微控制器的开发。

* **特点**：
  + 强大的插件生态系统，支持丰富的功能扩展。
  + 可以集成AVR-GCC和调试器，支持单步调试和性能分析。
  + 支持多种版本控制系统和项目管理工具。

### 6. Arduino IDE with AVR Core

**Arduino IDE** 是一个简单易用的开发环境，通过安装AVR Core扩展，可以支持AVR系列微控制器的开发。

* **特点**：
  + 集成了Arduino风格的编程语言和库。
  + 支持通过简单的图形界面和编程接口进行开发。
  + 社区支持丰富，有大量的示例和库可用。

### 7. 其他工具和环境

除了上述列出的主流开发环境外，还有一些其他工具和环境可以用于AVR系列微控制器的开发，具体选择取决于开发者的经验、项目需求和技术栈。

总体来说，Atmel Studio和MPLAB X IDE是官方推荐的主要开发环境，提供了完整的工具链和支持。AVR-GCC作为开源的选择，可以与多种IDE集成使用，适合喜欢自定义和跨平台开发的开发者。Arduino IDE则适合初学者和快速原型开发。

**烧写工具**

## 5.PIC

**开发编译环境**

PIC（Peripheral Interface Controller）是由Microchip Technology推出的一系列8位和16位微控制器。常见的PIC开发编译环境和工具如下：

### 1. MPLAB X IDE

**MPLAB X IDE** 是Microchip官方推出的开源跨平台IDE，支持多种Microchip微控制器，包括PIC系列。

* **特点**：
  + 提供完整的开发工具链，包括编译器、调试器和仿真器。
  + 集成了丰富的插件和工具支持。
  + 提供了强大的调试功能和性能分析工具。

### 2. MPLAB XC Compiler

**MPLAB XC Compiler** 是Microchip官方提供的一系列专业的编译器，支持各种PIC系列微控制器的开发。

* **特点**：
  + 提供了针对PIC架构优化的C编译器。
  + 支持高级优化和代码生成。
  + 集成了与MPLAB X IDE无缝连接的功能。

### 3. CCS C Compiler

**CCS C Compiler** 是一款商业的C编译器，专门用于PIC微控制器的开发。

* **特点**：
  + 提供了高度优化的PIC编译器。
  + 支持丰富的PIC特定库和组件。
  + 适合快速开发和原型验证。

### 4. Hi-Tech C Compiler

**Hi-Tech C Compiler** 是Microchip旗下的一款C编译器，专门用于PIC系列微控制器的开发。

* **特点**：
  + 提供了高效的编译器和代码生成。
  + 支持多种PIC系列微控制器。
  + 集成了丰富的标准C库和PIC特定库。

### 5. Proteus Design Suite

**Proteus Design Suite** 是一款集成的电子设计自动化软件，也是一种仿真环境，可以用于PIC微控制器的软硬件开发。

* **特点**：
  + 提供了完整的电路设计和仿真功能。
  + 支持PIC微控制器的模拟和调试。
  + 可以与MPLAB X IDE等工具集成。

### 6. Arduino IDE with PIC32 Core

**Arduino IDE** 是一个简单易用的开发环境，通过安装PIC32 Core扩展，可以支持PIC32系列微控制器的开发。

* **特点**：
  + 集成了Arduino风格的编程语言和库。
  + 支持通过简单的图形界面和编程接口进行开发。
  + 社区支持丰富，有大量的示例和库可用。

### 7. 其他工具和环境

除了上述列出的主流开发环境外，还有一些其他工具和环境可以用于PIC微控制器的开发，具体选择取决于开发者的经验、项目需求和技术栈。

总体来说，MPLAB X IDE和相关的MPLAB XC Compiler是Microchip官方推荐的主要开发工具，适合从初学者到专业开发者的不同需求。其他的商业编译器和仿真工具也提供了丰富的功能和支持，可以根据具体需求进行选择和使用。

**烧写工具**

## 6.XXXX

# 二．项目准备

1. **单片机DIV常见封装形式**

DIP

SIP

TO

SOIC

1. **元器件购买平台**

嘉立创

<https://www.szlcsc.com/>

云汉芯城

<https://www.ickey.cn/>

华强商城

<https://www.hqbuy.com/>

远程通过esp控制家里服务器

单片机小项目(待定)

1. **完成需求**

**Esp32控制家里电脑开关电源，stm单片机小项目**

1. **Xxx**

# 三．常见计算机设计模型有哪些

常见的计算机设计模型可以根据其核心特性和应用领域分类如下：

## 1. 冯·诺依曼模型（Von Neumann Architecture）

* **特点**：
  + 单一存储器结构：指令和数据存储在同一存储器中。
  + 顺序执行：指令按照顺序逐条执行。
  + 存储程序概念：程序以二进制形式存储在存储器中。
  + 控制流由程序计数器（PC）控制。
* **应用**：通用计算机系统，如个人电脑、服务器等。

## 2. 哈佛模型（Harvard Architecture）

* **特点**：
  + 分离存储器结构：指令存储器（程序存储器）和数据存储器分开。
  + 独立总线：指令和数据通过不同的总线传输。
  + 并行处理：可以并行提取指令和数据，提高效率。
* **应用**：嵌入式系统、数字信号处理器（DSP）、某些高性能计算系统。

## 3. 指令集架构（ISA）

* **CISC（Complex Instruction Set Computer）**：
  + 特点：指令集复杂，一条指令可完成多个操作。
  + 应用：x86架构，用于通用计算机。
* **RISC（Reduced Instruction Set Computer）**：
  + 特点：指令集精简，每条指令执行单一操作。
  + 应用：ARM架构，广泛应用于嵌入式系统和移动设备。

## 4. 并行处理模型

* **超标量架构（Superscalar Architecture）**：
  + 特点：多个执行单元可以同时执行多条指令，提高指令级并行性。
  + 应用：现代CPU，如Intel Core系列和AMD Ryzen系列。
* **多核处理（Multi-core Processing）**：
  + 特点：在单个处理器芯片上集成多个处理核心，增加并行处理能力。
  + 应用：高性能计算、服务器等。
* **超线程技术（Hyper-Threading Technology）**：
  + 特点：在一个物理处理器核心上模拟多个逻辑处理器核心，增加并行度。
  + 应用：Intel的超线程技术（HT）用于提高多任务处理性能。

## 5. 内存访问模型

* **SMP（Symmetric Multiprocessing）**：
  + 特点：多个处理器核心共享同一物理内存和系统总线。
  + 应用：多处理器服务器和工作站。
* **NUMA（Non-Uniform Memory Access）**：
  + 特点：每个处理器核心有自己的本地内存，但可以访问其他处理器的内存，速度不一致。
  + 应用：大型多处理器系统，用于提高内存访问效率。

## 6. 并行计算模型

* **SIMD（Single Instruction Multiple Data）**：
  + 特点：单条指令同时操作多个数据元素，用于数据并行计算。
  + 应用：科学计算、图形处理等。
* **MIMD（Multiple Instruction Multiple Data）**：
  + 特点：多个处理器核心同时执行不同的指令流和数据流。
  + 应用：分布式计算、大规模并行处理等。

## 7. 专用计算模型

* **GPU（Graphics Processing Unit）**：
  + 特点：大规模并行处理单元，用于图形渲染和通用并行计算。
  + 应用：图形处理、科学计算、深度学习等。
* **FPGA（Field-Programmable Gate Array）**：
  + 特点：可编程逻辑门阵列，灵活可配置，适用于特定硬件加速需求。
  + 应用：数据加速、嵌入式系统、通信设备等。
* **ASIC（Application-Specific Integrated Circuit）**：
  + 特点：为特定应用定制的集成电路，提供高性能和低功耗。
  + 应用：特定领域的高性能计算需求，如网络设备、传感器控制等。
* **SoC（System on Chip）**：
  + 特点：将处理器核心、存储器、I/O接口等集成在一个芯片上，用于嵌入式系统和移动设备。
  + 应用：智能手机、平板电脑、物联网设备等。

## 总结

这些计算机设计模型在不同的应用场景中各具特点，选择合适的设计模型取决于应用的性能需求、资源约束和系统设计的复杂性。现代计算机系统往往会结合多种模型，以实现最佳的性能和效率。

# 四．Harvard与冯·诺依曼的关系

哈佛架构（Harvard Architecture）和冯·诺依曼架构（Von Neumann Architecture）是两种不同的计算机体系结构设计模型，它们在数据存储和处理方式上有显著的区别。

## 冯·诺依曼架构

冯·诺依曼架构是由约翰·冯·诺依曼在1945年提出的计算机设计模型，其主要特点包括：

* **统一存储器**：程序指令和数据存储在同一个存储器（内存）中。
* **顺序执行**：指令按照顺序逐条执行。
* **存储程序概念**：程序以二进制形式存储在内存中，并由程序计数器（PC）控制执行顺序。
* **单一总线**：指令和数据共享同一条总线进行传输。

冯·诺依曼架构的设计简单、通用性强，适用于大多数通用计算机系统。但由于指令和数据共享同一条总线，可能会导致数据传输瓶颈，限制系统的性能。

## 哈佛架构

哈佛架构最初应用于哈佛大学的Mark I计算机，与冯·诺依曼架构相比有明显的不同点：

* **分离存储器**：指令存储器（程序存储器）和数据存储器分别存储在不同的存储器中。
* **独立总线**：指令和数据使用不同的总线进行传输。
* **并行传输**：指令和数据可以同时进行传输，提高了系统的吞吐量和效率。

哈佛架构的设计复杂度相比冯·诺依曼架构略高，但能够避免冯·诺依曼架构中由于单一总线带来的性能瓶颈问题。这使得哈佛架构特别适用于需要高速数据处理和并行传输的应用，如数字信号处理（DSP）、嵌入式系统等。

## 关系与区别

1. **存储器结构**：
   * **冯·诺依曼架构**使用统一存储器，即指令和数据共享同一个内存空间。
   * **哈佛架构**使用分离存储器，即指令和数据分别存储在不同的存储器中。
2. **总线结构**：
   * **冯·诺依曼架构**使用单一总线进行指令和数据的传输。
   * **哈佛架构**使用独立的指令总线和数据总线，支持并行传输。
3. **应用领域**：
   * **冯·诺依曼架构**广泛应用于通用计算领域，如个人电脑、服务器等。
   * **哈佛架构**更常见于需要高速数据处理和并行传输的专用领域，如嵌入式系统、DSP等。
4. **发展趋势**：
   * 在实际应用中，有时候会结合哈佛架构和冯·诺依曼架构的优点，形成混合架构，以满足特定需求的计算机系统。

## 总结

哈佛架构和冯·诺依曼架构代表了两种主要的计算机设计范式，各自在特定的应用场景中展现出优势。选择合适的架构取决于具体的应用需求，如性能要求、数据处理方式以及系统的复杂度和成本考量。

**参考：**

<https://www.bilibili.com/video/BV12v411b7xg>