















# 嵌入式程序框架

CubeMx生成一个新工程后，在Vscode的插件Embedded IDE上导入工程后，基本的文件结构如下图：

	.cmsis	2023/3/31 10:16	文件夹	
	.eide	2023/3/31 10:17	文件夹	
	.vscode	2023/3/31 10:17	文件夹	
	build	2023/3/31 10:26	文件夹	
	Core	2023/3/30 16:14	文件夹	
	Drivers	2023/3/30 16:14	文件夹	
	MDK-ARM	2023/3/31 10:14	文件夹	
	Middlewares	2023/3/30 16:14	文件夹	
	.clang-format	2023/3/31 10:15	CLANG-FORMAT...	1 KB
	.eide usr.ctx.json	2023/3/31 10:35	JSON 文件	1 KB
	.gitignore	2023/3/31 10:17	文本文档	1 KB
	.mxproject	2023/3/31 9:45	MXPROJECT 文件	16 KB
	Monster.code-workspace	2023/3/31 10:17	Code Workspac...	2 KB
	MX Monster.ioc	2023/3/31 9:45	STM32CubeMX	17 KB

需要关注的有四个文件夹


- Core：主程序、中断等核心程序
- Driver：底层驱动文件，hal库
- Middlewares：中间层，如Freertos库
- MDK-ARM：keil工程文件和keil编译的中间文件

以上四个文件都是CubeMx自动生成的，能够修改而在下次CubeMx生成工程的时候不被覆盖的，只有Core文件中部分文件的特定位置，我们需要利用这些地方重构应用层和底层

关于应用层，需要在一个新的目录下进行，而main中创建的默认任务就负责创建一个应用层的初始化任务；应用层采用c++语法，在此之前我们需要弄清楚Freertos的应用

FreeRTOS - Market leading RTOS (Real Time Operating System) for embedded systems with Internet of Things extensions

Market-leading MIT licensed open source real-time operating system (RTOS) for microcontrollers and microprocessors. Includes IoT and general purpose libraries.

 <https://www.freertos.org/zh-cn-cmn-s/index.html>

这是Freertos官方的入门教程

### FreeRTOS Features - FreeRTOS

FreeRTOS Kernel Developer Docs Tasks Queues, Mutexes, Semaphores...Direct To Task Notifications Stream & Message Buffers  
Software Timers Event Groups (or "Flags") Source Code Organization FreeRTOSConfig.h Static vs Dynamic Memory Heap Memory  
Management Stack Overflow Protection Creating a New Project Co-Routines (deprecated)

 <https://www.freertos.org/zh-cn-cmn-s/features.html>

这是Freertos开发者文档

### FreeRTOS API categories

Links to FreeRTOS API function descriptions ordered by category. FreeRTOS is a portable, open source, mini Real Time kernel.A free RTOS for small embedded systems

 <https://www.freertos.org/zh-cn-cmn-s/a00106.html>

Freertos api引用

到应用层，需要实现的有任务初始化，底层初始化，任务实现

应用层头文件供main.c调用的函数有任务初始化任务函数，底层初始化函数

因为需要用到c 和 c++混编，这里贴一下相关点

`extern "C"` 指定一下内容使用c的语法编译和链接，c和c++之间调用都可以使用

#### 1. c调用c++的函数

```
//被包含的c++头文件
#ifdef __cplusplus
extern "C"
{
#endif

// 函数声明

#ifdef __cplusplus
}
#endif
```

#### 2. c++ 调用c的函数

```
//c头文件
#ifdef __cplusplus
extern "C"
{
#endif

// 头文件内容

#ifdef __cplusplus
}
#endif
```

应用层的底层初始化函数和任务初始化函数写完后，  
在main.c第二段用户代码加上底层初始化函数

```
/* USER CODE BEGIN 2 */
User_Hardware_Init();
/* USER CODE END 2 */
```

在main.c的默认任务里加上任务初始化函数并终止默认任务

```
void StartDefaultTask(void const * argument)
{
    /* USER CODE BEGIN 5 */
    Application_Task_Init();
    osThreadTerminate(defaultTaskHandle);
    /* Infinite loop */
    for(;;)
    {
        osDelay(1);
    }
    /* USER CODE END 5 */
}
```