

# 从 **STM32F4xx** 移植到 **GD32F4xx** 的移植说明

---

## 目录

1、	本文简介.....	2
2、	硬件资源对比.....	2
3、	系统及外设资源对比.....	2
4、	开发集成环境和烧录调试工具兼容说明.....	3
5、	<b>GD32F4xx 系列 MCU 移植步骤.....</b>	<b>3</b>
5.1.	工程选型配置.....	3
5.2.	切换系统时钟注意事项.....	5
5.3.	SPI 模块使用时修改代码.....	5
5.4.	ADC 模块使用时修改代码.....	5
5.5.	USART 模块使用时修改代码.....	5
5.6.	ENET 模块使用时修改代码.....	6
5.7.	USBFS 模块使用时修改代码.....	6

## 1、 本文简介

GD32F4xx 系列 MCU 是基于 ARM® Cortex™-M4 处理器的 32 位通用微控制器，主频高达 200MHz，内部 Flash 高达 3MB，SRAM 高达 512KB，为 GD32 系列高性能 MCU。该系列 MCU 与 STM32F4xx 系列产品保持较高兼容性，总体来说：硬件上，严格意义来说 GD32F4xx 和 STM32F4xx 并不完全兼容，但是有差异的一般只有两个引脚，Vcap\_1 和 Vcap\_2，这两个脚在 STM32F4xx 上是有实际使用意义的，在 GD32F4xx 上，这两个脚是 NC，如果用户之前使用 STM32F4xx 开发的硬件电路，不管这两个引脚怎么接，都不影响替换，所以，可以说 GD32F4xx 和 STM32F4xx 硬件兼容；软件上，GD32F4xx 与 STM32F4xx 寄存器兼容，由于芯片设计及工艺不同，有些寄存器配置或时序配置需要修改，具体见本文说明，若用户之前使用 STM32F4xx 进行的软件开发，修改后可实现软件兼容。

## 2、 硬件资源对比

GD32F4xx 和 STM32F4xx 硬件引脚对比表 2.1 所示，由该表可知，GD32F4xx 可完全兼容 STM32F4xx 的硬件引脚定义。

表 2.1 GD32F4xx 和 STM32F4xx pin 对比表

Type	Pin_to_pin	Pinouts 差异					
LQFP64	GD32F4xx	31	NC	47	NC		
	STM32F4xx		VCAP_1		VCAP_2		
LQFP100	GD32F4xx	49	NC	73	NC		
	STM32F4xx		VCAP_1		VCAP_2		
LQFP144	GD32F4xx	71	NC	106	NC		
	STM32F4xx		VCAP_1		VCAP_2		
BGA176	GD32F4xx	M10	NC	F13	NC	L4	NC
	STM32F4xx		VCAP_1		VCAP_2		BYPASS_REG

注：

- (1) NC 代表可接高、可接地、可不接。
- (2) STM32F4xx 的 VCAP\_1/2 引脚一般是通过阻容接地，若采用 GD32F4xx 替代，建议可直接通过电阻接地，电容可省略。
- (3) STM32F4xx 的 BYPASS\_REG 引脚一般接地或接高，不影响替换。

## 3、 系统及外设资源对比

GD32F4xx 外设资源较丰富，可实现对 STM32F4xx 外设资源的覆盖，具体系统及外设资源对比如表

3.1 所示。

表 3.1 GD32F4xx 和 STM32F4xx 系统及外设资源对比表

系统及外设资源	GD32F4xx	STM32F4xx
主频	200MHz	180MHz
内核	M4F	M4F
Flash	Up to 3MB	2MB
SRAM	Up to 512KB	256KB
供电范围	2.6V-3.6V	1.8V-3.6V
active 功耗	<a href="#">99mA@200MHz 所有外设使能</a>	<a href="#">98mA@180MHz 所有外设使能</a>
温度范围	-40°C~85°C/负 40 度-105 度	-40°C~85°C/-40°C~105°C
外设资源	支持高达 14 个定时器、8 个串口、3 路 IIC、6 路 SPI、2 路 CAN、USBFS、USBHS、2 路 IIS、SDIO、LCD_TFT、摄像头接口、以太网 MAC、IPA、EXMC、3 个 ADC、2 个 DAC	支持高达 14 个定时器、8 路串口、6 路 SPI、3 路 IIC、USBFS、USBHS、2 路 CAN、1 路 SAI(IIS)、SDIO、摄像头接口、LCD_TFT、Accelerator(IPA)、3 个 ADC、2 个 DAC、以太网 MAC

## 4、开发集成环境和烧录调试工具兼容说明

GD32F4xx 和 STM32F4xx 均为 ARM M4 内核 MCU，可采用相同的集成开发环境和烧录调试工具，一般集成开发环境为 IAR、KEIL 和 eclipse 等，烧录和调试工具可选用 ULINK、JLINK、STLINK、GDLink(仅 GD32F4xx 支持)。

## 5、GD32F4xx 系列 MCU 移植步骤

### 5.1. 工程选型配置

打开 KEIL 或 IAR 工程后，工程选型可选择 STM32F4xx 或 GD32F4xx 选型，建议选择 GD32F4xx 选型，首先安装选型 pack 包，选型 pack 包可通过 <https://pan.baidu.com/s/1mhQsNpu> 网盘或 <http://gd32mcu.21ic.com/documents/官网下载>，若采用 KEIL5 开发，也可通过 Pack Installer 进行在线更新。下载后解压 PACK 包如图 5.1 所示，其中 GigaDevice.GD32F4xx\_Addon.1.0.2.exe 为 KEIL4 的 Pack 包、GigaDevice.GD32F4xx\_DFP.1.0.3.pack 为 KEIL5 的 Pack 包、IAR\_GD32F4xx\_ADDON.1.0.0.exe 为 IAR 的 Pack 包，请根据不同的软件开发环境及版本进行安装，安装路径默认。




 GigaDevice.GD32F4xx_Addon.1.0.2.exe	25/04/2017 17:13	应用程序	2,420 KB
 GigaDevice.GD32F4xx_DFP.1.0.3.pack	07/06/2017 16:28	uVision Software...	1,034 KB
 IAR_GD32F4xx_ADDON.1.0.0.exe	24/04/2017 20:12	应用程序	3,963 KB

图 5.1 pack 包

安装 Pack 包后，可在具体软件工程中进行切换选型，KEIL5 的选型配置如图 5.2 所示。选型后，先关闭 Option 配置窗口，然后再打开 Option 配置窗口，

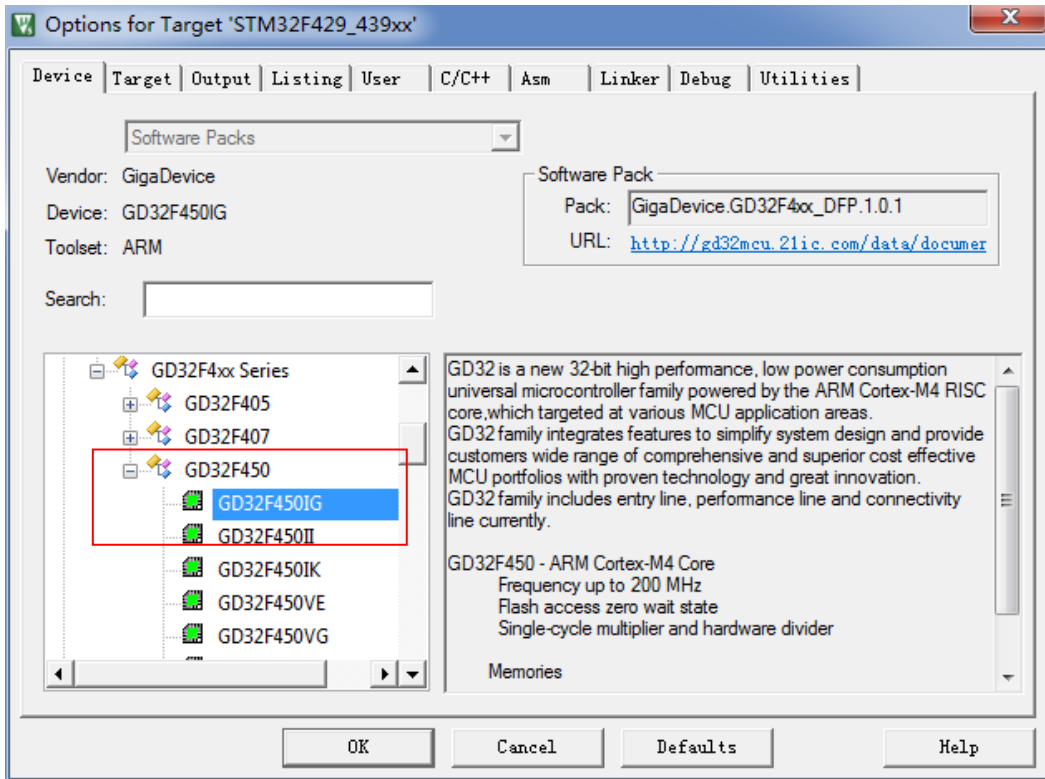


图 5.2 KEIL5 下选型配置

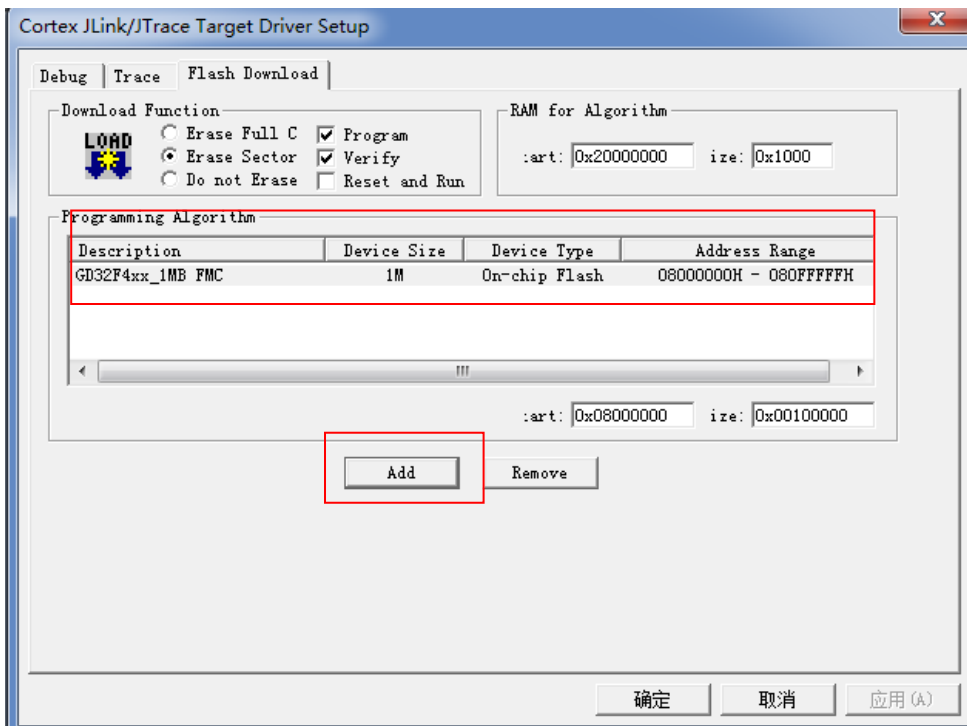


图 5.3 下载算法选择窗口

## 5.2. 切换系统时钟注意事项

GD32F4xx 系统时钟从高频时钟切换至低频时钟（IRC16M）时，可能导致 IRC16M、HXTAL、PLL 时钟消失，使得程序跑飞，常见于 IAP 的 BOOT 跳转 APP 时。软件上需要切换时钟前将 HCLK 时钟 4 分频，延迟后，再进行时钟切换，修改代码如下例代码 2.1 所示。

示例代码 2.1

```
__IO uint32_t i_delay = 0;
RCC->CFGR |= RCC_CFGR_HPRE_DIV4;
for(i_delay=0xffff;i_delay>0;i_delay--);
```

## 5.3. SPI 模块使用时修改代码

若用户在使用时切换 SPI 配置，重配 SPI 后，SPI 时钟改变，建议在重配 SPI 之前先关闭 SPI 模块，配置完成后，再使能 SPI。修改代码如下例代码 2.2 所示。

示例代码 2.2

```
SPI_Cmd(SPIx, DISABLE);
SPI_Cmd(SPIx, ENABLE);
```

## 5.4. ADC 模块使用时修改代码

ADC 在设置为 8bit 模式右对齐时，GD32F4xx 是取 12bit 数据中的高 8bit，使用时请注意，如图 5.4 所示。可采用左对齐，读取高字节数据，如图 5.5 所示。

X	X	X	X	Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	X	X	X	X
---	---	---	---	------	------	------	------	------	------	------	------	---	---	---	---

图 5.4 右对齐时 ADC 采样数据寄存器

Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	X	X	X	X	X	X	X	X
------	------	------	------	------	------	------	------	---	---	---	---	---	---	---	---

图 5.5 左对齐时 ADC 采样数据寄存器

在 ADC IDLE 的时候，软件写一下 `swstart`，硬件会检测到 `swstart` 的上升沿，然后开始采样，并在采样开始的时候将 `swstart` 清 0。如果在 ADC 正在转换时写 `swstart`，此时 ADC 无法检测上升沿，当 ADC 转换完当前通道后，`swstart` 虽为高电平，但检测不到上升沿，ADC 也无法启动转换，因此置位 `swstart` 之前需要等待 EOC 标志置位，即 ADC 转换完成。

## 5.5. USART 模块使用时修改代码

GD32F4xx MCU 的 USART 使用 DMA 发送，如果先使能 DMA，后使能串口的 Tx 功能，会导致由于 DMA 先于 USART 打开的时间差，造成在 USART 没准备好发送的情况下 DMA 事先传输数据，进而导致数

---

据丢失，软件上可修改 DMA 和串口配置顺序，先配置串口，然后配置 DMA。

## 5.6. ENET 模块使用时修改代码

若出现以太网 ping 不通的问题，若排除硬件问题，软件有以下两种可能：（1）由于 GD32F4xx 芯片主频较高，在代码端，应该保证将 ENET\_DMA\_CTL 寄存器的第 20 位 FTF 置 1，清空发送 FIFO 后，必须软件等待该位被硬件清 0 后适当延迟再进行其他操作。否则，有概率性导致 ENET 发送异常，从而出现 PING 不通的情况，修改代码如示例代码 2.3 所示。

示例代码 2.3

```
void ETH_FlushTransmitFIFO(void)
{
    ETH->DMAOMR |= ETH_DMAOMR_FTF;
    while((ETH->DMAOMR & ETH_DMAOMR_FTF) != 0);
    for(uint32_t i=0; i<0xffff; i++ );
}
```

（2）若客户以太网采用半双工通信，若打开载波侦听，会导致发送异常，按照 802.1 以太网协议，需关闭载波监听功能，修改代码如示例代码 2.4 所示。

示例代码 2.4

```
ETH_InitStruct->ETH_CarrierSense = ETH_CarrierSense_Disable;// 半双工通信模式下修改
```

## 5.7. USBFS 模块使用时修改代码

若出现 USB 端点发送数据偶尔出错的情况，请排查 DCD\_EP\_Flush();函数使用情况，该函数仅需在 USB 初始化中端点缓冲区配置完成后，进行 Flush，其他地方 Flush 缓冲区会造成 USB 缓冲区异常，因而只需在 USB 初始化时进行一次缓冲区 Flush 操作，其他地方的 Flush 操作可屏蔽。