前言

本应用笔记旨在帮助您分析从现有的SXX32F107器件移植到AT32F407器件所需的步骤。本文档收集了最重要的信息，并列出了需要注意的重要事项。

要将应用程序从SXX32F107系列移植到AT32F407系列，用户需要分析硬件移植、外设移植和固件移植。

支持型号列表：

支持型号	AT32F407xxxx

目录

表目录

图目录

1 AT32F407 与 SXX32F107 异同

AT32F407 系列微控制器基本除 RCC, USB 外，其它外设基本兼容 SXX32F107 系列，同时强化许多功能的关系，有些许地方与 SXX32F107 不同。

1.1 相同点概述

- 管脚定义：相同封装管脚定义相同。为扩增的外设作管脚复用定义延伸
- 编译工具：完全相同，例如Keil, IAR

1.2 差异点概述

表 1. SXX32F107 与 AT32F407 差异概述

	AT32F407	SXX32F107
系统		
内核	Cortex-M4 并支持 DSP 指令及浮点运算单元 FPU	Cortex-M3
系统时钟	主频 240MHz，APB1 与 APB2 总线皆为 120MHz，如果使用 MCO 给 ETH PHY 供时钟，主频建议配置到 200MHz	主频 72MHz, APB1 36MHz, APB2 72MHz
启动	13 ms	2.5 ms
重置	8 ms	-
Standby 唤醒	8 ms	50 us
SRAM 容量	扩充模式可达 224KB	64KB
加挂 SPI 闪存	支持加挂 SPI flash 为 SPIM，最高达 16M Bytes	无支援
系统存储器 (System Memory)	全系列 18KB，比 SXX32F107 多支持以下功能： 1. 对 SPIM 进行 ISP 烧写 2. 对闪存内容进行 CRC 校验	18KB
选择字节 (Option Byte)	48 Bytes， 扩增以下设定功能： 1. SRAM 模式设定 2. 4 Bytes 自定义字段(例如开发商 ID) 3. 8 Bytes SPIM 加密钥匙	16 Bytes
闪存 16-bit 写入时间	50 µs	52.5 us
闪存页擦除时间	50 ms	20 ~ 40 ms
闪存整片擦除时间	0.8 s (AT32F407xC) 1.4 s (AT32F407xE) 2.8 s (AT32F407xG)	20 ~ 40 ms
外设		
安全库区保护	支持	无
RCC	无 PLL2 无 PLL3 时钟树结构不同，具体差异可参考 RM 文档	有 PLL2 有 PLL3
扩增 I2C	多一组 I2C	I2C1/2
扩增 SPI	多一组 SPI4	SPI1/2/3

	AT32F407	SXX32F107
SDIO	支持 SDIO1 和 SDIO2	无 SDIO
扩增 USART 和 UART	支持 USART6/UART7/UART8	不支持 USART6/UART7/UART8
SPI1 支援 I2S	SPI1 可支持 I^2S 功能	SPI1 仅为 SPI 功能
ETH	IEEE 1588-2008 的网路精确时钟同步标准 ETH 全局中断号：IRQ 79 ETH WKUP 中断号：IRQ 80	IEEE 1588-2002 的网路精确时钟同步标准 ETH 全局中断号：IRQ 61 ETH WKUP 中断号：IRQ 62
USB	USB FS 只支持 USB Device Full Speed	OTG_FS 支持 Host，Device 模式
CAN	支持 CAN1 和 CAN2 CAN2 过滤器独立使用	支持 CAN1 和 CAN2 CAN2 过滤器需通过 CAN1 使用
ADC	3 组 ADC	2 组 ADC
HSI 自动时钟校准 ACC	支持	不支持
XMC	支持	不支持
支持闪存 CRC 校验	支持	不支持
高速 GPIO	GPIO 挂在 AHB 总线上	GPIO 挂在 APB 总线上
32 位定时器	TMR2, TMR5 为 32 位定时器	皆为 16 位
ADC	2Msps (max ADCCLK=28MHz)	1Msps (max ADCCLK=14MHz)
ADC 触发事件	支持 TMR1, TMR8 及 TMR15	无 TMR15
温度传感器	正温度系数	负温度系数
电气		
电压范围	2.6V~3.6V	2V~3.6V
环境温度 T_A	-40°C~+105°C	-40°C~+105°C
内核电压	1.2V	1.8V
ESD 参数	HBM:5KV, CDM:1000V	HBM:2KV, CDM:500V
运行模式	39.2 mA @ 72MHz	47.3 mA @ 72MHz
睡眠功耗	33.9 mA @ 72MHz	28.2 mA @ 72MHz
停机功耗	1.4 mA	33 uA
待机功耗	5.7 uA	2.1 uA

2 快速替换 SXX32F107 芯片

2.1 快速验证 ETH 兼容性

- 步骤一：解焊SXX32F107，换成AT32F407对应型号
- 步骤二：修改之前程序的RCC倍频部分，如倍频到240MHz（请参考BSP）

```
static void SetSysClockTo240(void)
{
  __IO uint32_t StartUpCounter = 0, HSEStatus = 0;
  …
  if (HSEStatus == (uint32_t)0x01)
  {
    /* HCLK = SYSCLK */
    RCC->CFGR |= (uint32_t)RCC_CFGR_HPRE_DIV1;

    /* PCLK2 = HCLK/2 */
    RCC->CFGR |= (uint32_t)RCC_CFGR_PPRE2_DIV2;

    /* PCLK1 = HCLK/2 */
    RCC->CFGR |= (uint32_t)RCC_CFGR_PPRE1_DIV2;

    /*  PLL configuration: PLLCLK = HSE * 30 = 240 MHz */
    RCC->CFGR &= (uint32_t)((uint32_t)~(RCC_CFGR_PLLSRC | RCC_CFGR_PLLXTPRE |
                                        RCC_CFGR_PLLMULL));
    RCC->CFGR |= (uint32_t)(RCC_CFGR_PLLSRC_HSE | RCC_CFGR_PLLMULL30 |
RCC_CFGR_PLLRANGE_GT72MHZ);

    /* Enable PLL */
    RCC->CR |= RCC_CR_PLLON;

    /* Wait till PLL is ready */
    while((RCC->CR & RCC_CR_PLLRDY) == 0)
    {
    }
#if defined (STM32F10X_403A) || defined (STM32F10X_407)
    RCC_StepModeCmd(ENABLE);
#endif
    /* Select PLL as system clock source */
    RCC->CFGR &= (uint32_t)((uint32_t)~(RCC_CFGR_SW));
    RCC->CFGR |= (uint32_t)RCC_CFGR_SW_PLL;

    /* Wait till PLL is used as system clock source */
    while ((RCC->CFGR & (uint32_t)RCC_CFGR_SWS) != (uint32_t)0x08)
    {
    }
#if defined (STM32F10X_403A) || defined (STM32F10X_407)
    RCC_StepModeCmd(DISABLE);
#endif
  }
  …
}
```

- 步骤三：修改ETH PHY 时钟源，可直接将系统时钟分频到需要的时钟通过MCO输出（具体请参考BSP）

- 步骤四: 修改ETH中断号，AT32F407（IRQ 79, 80）和SXX32F107（IRQ 61, 62）中断号不同
- 步骤五：使用雅特力ICP, ISP或KEIL, IAR下载重新编译的文件或BIN文件。
- 步骤六：查看程序能否正常运行。
- 步骤七：如果系统时钟源采用HSE，为了确保量产稳定性，请修改时钟初始化源代码，方法如下：
  打开 system_at32f4xx.c 找到当前的系统时钟频率配置函数，如 240MHz 函数：
  static void SetSysClockTo240(void)
  然后按如下方式配置自动顺滑频率切换功能

```
/* Wait till PLL is ready */
      while((RCC->CR & RCC_CR_PLLRDY) == 0)
      {
      }
      *((unsigned int    *)0x40021054) |= (0x30);// 开启自动滑顺频率切换功能

      /* Select PLL as system clock source */
      RCC->CFGR &= (uint32_t)((uint32_t)~(RCC_CFGR_SW));
      RCC->CFGR |= (uint32_t)RCC_CFGR_SW_PLL;

      /* Wait till PLL is used as system clock source */
      while ((RCC->CFGR & (uint32_t)RCC_CFGR_SWS) != (uint32_t)0x08)
      {
}
//此时不需再等待 200us
*((unsigned int    *)0x40021054) &=~ (0x30); //关闭自动滑顺频率切换功能
```

- 步骤六：其他问题快速排查请参考*3.2 外设使用区别*
- 步骤七：如果经过上述步骤后程序仍无法正常运行，请参考本文件其他章节，或连络代理商及雅特力科技技术支持人员协助解决。

2.2 BSP 工程替换步骤

- 六种应用情形概述如下

表 2. BSP 替换步骤概述

序号	使用什么 BSP/Pack 开发	是否使用 AT32F407 新功能	处理方法
1	AT32F407BSP/Pack	是/否	1. 结合*3.2 外设使用区别*修改对应程序
2	SXX32F107 BSP/Pack	否	1. 参考2.1在系统时钟切换之前开启时钟顺滑模式 2. 参考2.1在系统时钟切换之后关闭时钟顺滑模式 3. 结合*3.2 外设使用区别*修改对应程序
3	SXX32F107 寄存器操作	否	1. 参考2.1在系统时钟切换之前开启时钟顺滑模式 2. 参考2.1在系统时钟切换之后关闭时钟顺滑模式 3. 结合*3.2 外设使用区别*修改对应程序
4	SXX32F107 BSP + AT32 Pack	否	1. 参考2.1在系统时钟切换之前开启时钟顺滑模式 2. 参考2.1在系统时钟切换之后关闭时钟顺滑模式 3. 修改FPU设置 4. 结合*3.2 外设使用区别*修改对应程序
5	SXX32F107 寄存器操作	是	1. 参考2.1在系统时钟切换之前开启时钟顺滑模式 2. 参考2.1在系统时钟切换之后关闭时钟顺滑模式 3. 结合*3.2 外设使用区别*修改对应程序
6	SXX32F107 BSP/Pack	是	1. 参考2.1在系统时钟切换之前开启时钟顺滑模式 2. 参考2.1在系统时钟切换之后关闭时钟顺滑模式 3. 结合*3.2 外设使用区别*修改对应程序

- 详细请参阅《AT32F407BSP和Pack应用指南》

3 AT32F407 兼容性解析

3.1 功能增强

3.1.1 FPU 的 ARM® 32 位的 Cortex®-M4F

- 描述：
  - 带存储器保护单元(MPU)
  - 内建单周期乘法和硬件除法
  - 内建浮点运算(FPU)
  - 新增支持DSP指令集
  - 例程参考
    AT32F4xx_StdPeriph_Lib_V1.x.x\Project\Examples\AT_START_F407\CortexM4\FPU
    AT32F4xx_StdPeriph_Lib_V1.x.x\Libraries\CMSIS\DSP_Lib\Examples

3.1.2 安全库区保护

- 描述
  - 目前越来越多的微控器(MCU)应用需要使用到复杂的算法及中间件解决方案(middleware solution)，因此，如何保护软件方案商开发出来的核心算法等知识产权代码(IP-Code)，便成为微控制器应用中一项很重要的课题。
    为因应这一重要的需求，AT32F407 系列提供了安全库区(sLib)的功能，以防止重要的 IP-Code 被终端用户的程序做修改或读取，进而达到保护的目的。
- 使用范例
  - 请参考《AT32F407 安全库区(SLIB) 应用指南.pdf》

3.1.3 高频 PLL 设定

- 描述：
  - AT32F407内置的PLL可输出240MHz时钟，设定略有不同
  - 该PLL支持两频段，以72MHz为分界，最高可达240MHz，须根据输出频率设定PLLRANGE寄存器
- 使用范例：
  - SXX32F107 PLL设定程序范例：
    ```
    RCC->CFGR |= (uint32_t)(RCC_CFGR_PLLSRC_HSE | RCC_CFGR_PLLMULL9);
    ```
  - AT32F407 PLL设定程序范例：

```
#define RCC_CFG_PLLMULT1 ((uint32_t)0x20000000)  /*!< PLL input clock    *  17 */
#define RCC_CFG_PLLMULT18 ((uint32_t)0x20040000) /*!< PLL input clock  *  18 */
#define RCC_CFG_PLLMULT19 ((uint32_t)0x20080000) /*!< PLL input clock  *  19 */
#define RCC_CFG_PLLMULT20 ((uint32_t)0x200C0000) /*!< PLL input clock  *  20 */
…
#define RCC_CFG_PLLMULT61 ((uint32_t)0x60300000) /*!< PLL input clock  *  61 */
#define RCC_CFG_PLLMULT62 ((uint32_t)0x60340000) /*!< PLL input clock  *  62 */
#define RCC_CFG_PLLMULT63 ((uint32_t)0x60380000) /*!< PLL input clock  *  63 */
#define RCC_CFG_PLLMULT64 ((uint32_t)0x603C0000) /*!< PLL input clock  *  64 */

#define RCC_CFG_PLLRANGE ((uint32_t)0x80000000) /*!< PLL Frequency range */
#define RCC_CFG_PLLRANGE_LE72MHZ ((uint32_t)0x00000000) /*!< When PLL frequency is
less than or equal to 72MHz */
#define   RCC_CFG_PLLRANGE_GT72MHZ   ((uint32_t)0x80000000) /*!< When PLL
frequency is greater than 72MHz */
```

以设置 72MHz 为例：

```
RCC->CFG |= (uint32_t)(RCC_CFG_PLLRC_HSE | RCC_CFG_PLLMULT9 |
RCC_CFG_PLLRANGE_LE72MHZ);
```

以设置 200MHz 为例：

```
RCC->CFG |= (uint32_t)(RCC_CFG_PLLRC_HSE | RCC_CFG_PLLMULT25 |
RCC_CFG_PLLRANGE_GT72MHZ);
```

3.1.4 预分频器扩增

- 描述：
  - 由于主频提高至240MHz，相关预分频器做出扩增
  - USB预分频器扩增支援/2, /2.5, /3, /3.5, /4输出
  - ADC预分频器扩增支持/12, /16输出
  - HSE预分频器扩增支持/3, /4, /5输出
  - 主时钟输出(CLKOUT)扩增支持CLKOUT预分频器，可以实现CLKOUT/2. CLKOUT/4…CLKOUT/512除频
  - 主时钟输出(CLKOUT)扩增支持LSE, LSI, PLLCLK/4, USB48M, ADCCLK输出
  - 请参阅AT32F407参考手册3.3.2 RCC_CFG寄存器和3.3.12 额外寄存器（RCC_MISC）叙述

3.1.5 内存容量扩展

- 描述：
  - AT32F407支持内存扩展功能，可将内存由96KB扩展为224KB
  - 若开起此模式，则具备零等待特性的闪存地址空间将限制为128KB
  - 内存扩展功能透过选择字节EOPB0[7:0] （0x1FFF F810）设定
    0xFE: 片上内存为 224K 字节
    0xFF: 片上内存为 96K 字节
    其他设置保留
  - 选择字节EOPB0更新后须重置系统方能生效
  - 请参阅AT32F407参考手册5.3.4节
  - 例程参考
    AT32F4xx_StdPeriph_Lib_V1.x.x\Project\Examples\AT_START_F407\SRAM

3.1.6 透过 SPIM 加挂 SPI 闪存

- 描述：
  - AT32F407支持加挂SPI闪存，支持下列及其兼容之型号，容量可达16M Byte。如果无法判别 SPI FLASH 型号是否兼容，可以利用AT32 ICP工具的SPIM自动侦测功能
  - 请参阅AT32F407参考手册5.2.2节与5.4.14节
  - 支持下表SPI Flash型号外，也支持兼容下表型号命令集的SPI Flash芯片

表 3. SPI 闪存支持类型表

Vendor	SPI Flash 型号
ESMT	EN25F20A/EN25QH128A
Winbond	W25Q128V
GD	GD25Q16C/GD25Q32C/GD25Q64C/GD25Q80C/GD25Q127C

  - 加挂闪存为SPIM，使用地址0x0840 0000~0x1FFF 0000
  - 加挂闪存功能在各型封装均支持。SPIM_IO0和SPIM_IO01管脚可根据EXT_SPIF_GRMP[2:0]进行remap配置。使用管脚如下表：

表 4. EXT_SPIF_GRMP[2:0]为[000]时引脚支持

SPIM	Pin	LQFP48 QFN48	LQFP 64	LQFP 100	共享脚位描述
SPIM_SCK	PB1	19	27	36	ADC12_IN9/TMR3_CH4/TMR8_CH3N
SPIM_NSS	PA8	29	41	67	TMR1_CH1/CLKOUT/USART1_CK/I2C3_SCL
SPIM_IO0	PA11	32	44	70	USB_DM/TMR1_CH4/USART1_CTS/CAN_RX
SPIM_IO1	PA12	33	45	71	USB_DP/CAN_TX/USART1_RTS /TMR1_ETR
SPIM_IO2	PB7	43	59	93	TMR4_CH2/I2C1_SDA/XMC_NADV
SPIM_IO3	PB6	42	58	92	TMR4_CH1/I2C1_SCL

表 5. EXT_SPIF_GRMP[2:0]为[001]时引脚支持

SPIM	Pin	LQFP48 QFN48	LQFP 64	LQFP 100	共享脚位描述
SPIM_SCK	PB1	19	27	36	ADC12_IN9/TMR3_CH4/TMR8_CH3N
SPIM_NSS	PA8	29	41	67	TMR1_CH1/CLKOUT/USART1_CK/I2C3_SCL
SPIM_IO0	PB10	21	29	47	USART3_TX/I2C2_SCL/I2S3_MCK/TMR2_CH3
SPIM_IO1	PB11	22	30	48	USART3_RX/I2C2_SDA/TMR2_CH4
SPIM_IO2	PB7	43	59	93	TMR4_CH2/I2C1_SDA/XMC_NADV
SPIM_IO3	PB6	42	58	92	TMR4_CH1/I2C1_SCL

  - 例程参考
    AT32F4xx_StdPeriph_Lib_V1.x.x\Project\Examples\AT_START_F407\FLASH

3.1.7 扩增 SDIO2

- 描述：
  - AT32F407扩增SDIO2

- 适用于48PIN/64PIN/100PIN封装,其中48PIN封装只支持SDIO2
- 寄存器起始地址：0x4002_3400
- 中断号：IRQ60
- DMA：使用DMA2通道5
- 管脚复用设定：于AFIO_MAP2寄存器位[20:19]扩充SDIO2复用重映像SDIO2_REMAP[1:0]（请参阅AT32F407参考手册7.4.11小节）

3.1.8 扩增 I2C3

- 描述：
  - AT32F407扩增I²C3
  - 适用于64PIN/100PIN封装
  - 寄存器起始地址：0x4001_5C00
  - 中断号：IRQ61, IRQ62
  - DMA：使用DMA1信道2(TX)与通道3(RX)
  - 管脚复用设定：于AFIO_MAP2寄存器位[18]扩充I2C3复用重映像I2C3_REMAP（请参阅AT32F407参考手册7.4.8小节）

3.1.9 扩增 SPI4

- 描述：
  - AT32F407扩增SPI4/I2S4
  - 适用于64PIN/100PIN封装
  - 寄存器起始地址：0x4000_4000
  - 中断号：IRQ63
  - DMA：使用DMA2信道3(RX)与通道4(TX)
  - 管脚复用设定：于AFIO_MAP2寄存器位[17]扩充SPI4复用重映像SPI4_REMAP（请参阅AT32F407参考手册7.4.10小节）

3.1.10 扩增全双工 I2S

- 描述：
  - AT32F407 新增两个模块（I2S2_ext，I2S3_ext）以支持I2S全双工模式
  - 适用于48PIN/64PIN/100PIN封装
  - I2Sx_ext不能独立使用，只能配合I2Sx一起用于全双工模式，且I2Sx_ext需配置为从机
  - 全双工模式下，原I2S模式未用的SPI_MISO引脚用于扩增的数据线，其他引脚不变
  - I2Sx_ext无独立的中断向量号（与I2Sx共享），但有独立的寄存器地址和DMA信道（请参阅AT32F407参考手册1.2.1及9.3.8小节）

3.1.11 扩增 USART 和 UART

- 描述：
  - AT32F407扩增USART6/UART7/UART8
  - 寄存器起始地址：USART6-0x4001_6000, UART7-0x4001_6400, UART8-0x4001_6800
  - 中断号：IRQ76, IRQ77, IRQ78

- DMA：使用弹性请求（请参阅AT32F407参考手册9.3.8小节）
- 管脚复用设定：
  于 AFIO_MAP8 寄存器位[23:20]扩充 USART6 复用重映像 USART6_GRMP（请参阅 AT32F407 参考手册 7.4.7 小节）
  于 AFIO_MAP8 寄存器位[27:24]扩充 UART7 复用重映像 UART7_GRMP（请参阅 AT32F407 参考手册 7.4.7 小节）
  于 AFIO_MAP8 寄存器位[31:28]扩充 UART8 复用重映像 UART8_GRMP（请参阅 AT32F407 参考手册 7.4.7 小节）

3.1.12 32 位定时器

- 描述：
  - AT32F407 TMR2/TMR5 可配置成32位定时器
- 使用范例：
  - TMR2->CTRL1|=0X0400; //开启TMR2的32位模式
  - TMR5->CTRL1|=0X0400; //开启TMR5的32位模式

（相关变量须改为 32 位，请参考 AT32F407 库文件 at32f4xx_tim.c , at32f4xx_tim.h）

3.1.13 SPI1 复用为 I2S1

- 描述：
  - AT32F407 SPI1可复用为I^2S1
  - 适用于48PIN/64PIN/100PIN封装
  - 增加I^2S1_MCK引脚
  - 其他引脚，寄存器空间和中断号，DMA通道同SPI1

3.1.14 USBDEV 缓冲区

- 描述：
  - AT32F407 USB设备(USBDEV)缓冲区最大可扩充为768 Bytes
  - 于USB768B(RCC_MISC[24])设定（请参阅AT32F407参考手册3.3.11小节）
  - 512Byte模式：缓冲区地址范围0x4000 6000~0x4000 63FF
  - 768Byte模式：当USB768B 为1 时，SRAM 的大小为768~1280BYTE，地址范围为0x4000 7800~0x4000 81FF，缓冲区的大小与CAN1和CAN2 是否使能有关，当CAN1 和CAN2 都不使能时，USB 分组缓冲区最大可设定到1280 字节，当CAN1 或CAN2 任意一个使能，USB 分组缓冲区最大可设定到1024 字节，当CAN1 和CAN2 都使能时，USB 分组缓冲区最大可设定到768 字节。
- 768 Byte模式使用范例：
  - RCC->MISC |= 0x00000001 << 24; /*打开USB768Byte模式*/
  - #define PMAAddr 0x40007800 /*使用768Byte模式，一定要修改缓冲区地址*/

3.1.15 扩增 48MHz HSI 支持 USB 外设

- 描述：
  - AT32F407支持48MHz时钟供USB使用
  - 适用于系统时钟和USB时钟同步情形，如系统时钟跑200MHz（PLL from HSE）,USB外设仍需要使用情况。

- 详情请参考请参阅AT32F407参考手册3.2.2 HSI时钟

- 使用范例：
  - 请参考以下案例AT32F4xx_StdPeriph_Lib_V1.x.x/Project/AT_START_F407/Examples/USB_Device

3.1.16 HSI 自动时钟校准 ACC

- 描述：
  - AT32F407新增ACC模块

    HSI 自动时钟校准器（HSI ACC）利用USB 模块产生的SOF 信号（周期为1 毫秒）作为参考信号，实现对HSI 时钟的采样和校准。
  - 本模块主要功能就是实现对USB 设备提供48MHz±0.25%精度的时钟
  - 适用于所有封装
  - 寄存器起始地址：0x4001_5800
  - 中断号：IRQ72
- 使用范例：
  - 请参考以下范例：
    AT32F4xx_StdPeriph_Lib_V1.x.x\Project\AT_START_F407\Examples\ACC

3.1.17 支持闪存 CRC 校验

- 描述：
  - AT32F407支持闪存CRC校验，通过配置闪存CRC校验控制寄存器（CRC_DR），即通过闪存CRC校验结果寄存器（CRC_OUTR）校验闪存内容。该功能在在读保护或者sLib情况下同样适用。
  - 详情请参考《AT32F407系列 技术手册》中5.4.26 闪存CRC校验控制寄存器（CRC_DR）和5.4.27 闪存CRC校验结果寄存器（CRC_OUTR）

3.1.18 高速 GPIO

- 描述：
  - AT32F407对GPIO进行了升级，将GPIO时钟挂在AHB总线上，而SXX32F107的GPIO挂在APB总线上。
  - 在做GPIO翻转测试或者使用GPIO模拟SPI/USART/I2C等外设时，存在AT32F407GPIO翻转速度与SXX32F107相比，频率会更快。对比如下表

表 6. SXX32F107/407 GPIO toggle 最大速度性能测试

时钟频率配置(MHz)	AHB=192; APB2=96	AHB=72;APB2=72	AHB=36;APB2=36
SXX32F107 I/O 翻转(MHz)	不支持	18	9
AT32F407 I/O 翻转(MHz)	96	36	18

图 1. 407 在 192MHz 主频下 GPIO 最快翻转速度波型

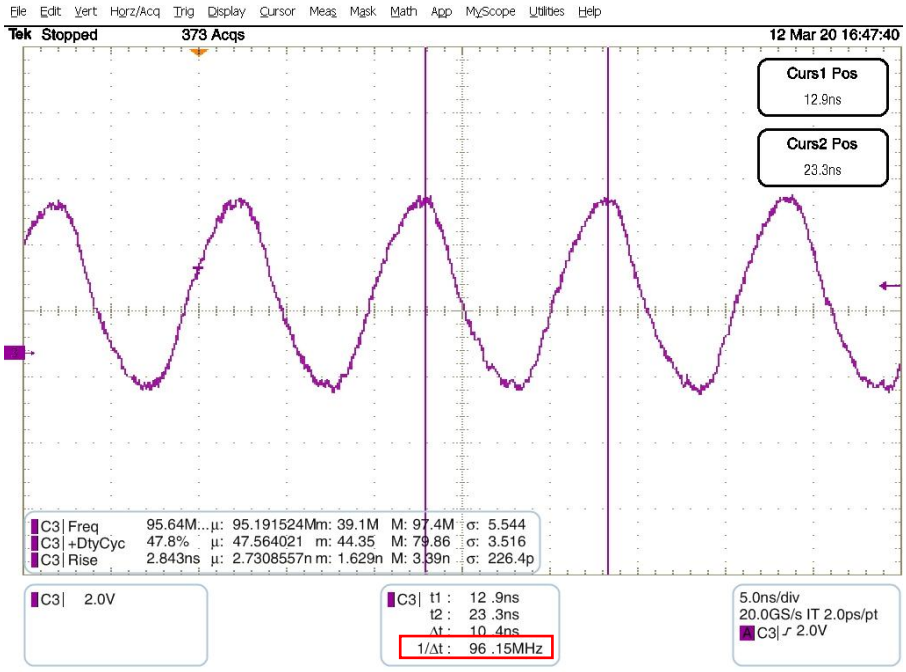


图 2. 407 在 72MHz 主频下 GPIO 最快翻转速度波型

图 3. SXX32F107 在 72MHz 主频下 GPIO 最快翻转速度波型

3.1.19 扩增 DMA 弹性映像请求功能

- 描述：
  - AT32F407的DMA1/DMA2新增弹性映像请求功能
  - 适用于64PIN/100PIN封装
  - 弹性映射与固定映射同时只能开启其一，二者不能同时开启
  - 弹性映射请求无独立的中断向量号（与固定映射请求共享），但有独立的配置寄存器地址（请参阅AT32F407参考手册）
- 使用范例：
  - 请参考以下范例：
    AT32F4xx_StdPeriph_Lib_V1.x.x\Project\AT_START_F407\Examples\DMA\SPI_RAM_FLEXIBLE。

3.2 外设使用区别

3.2.1 高频下 PLL 切换导致程序挂起或 hard fault

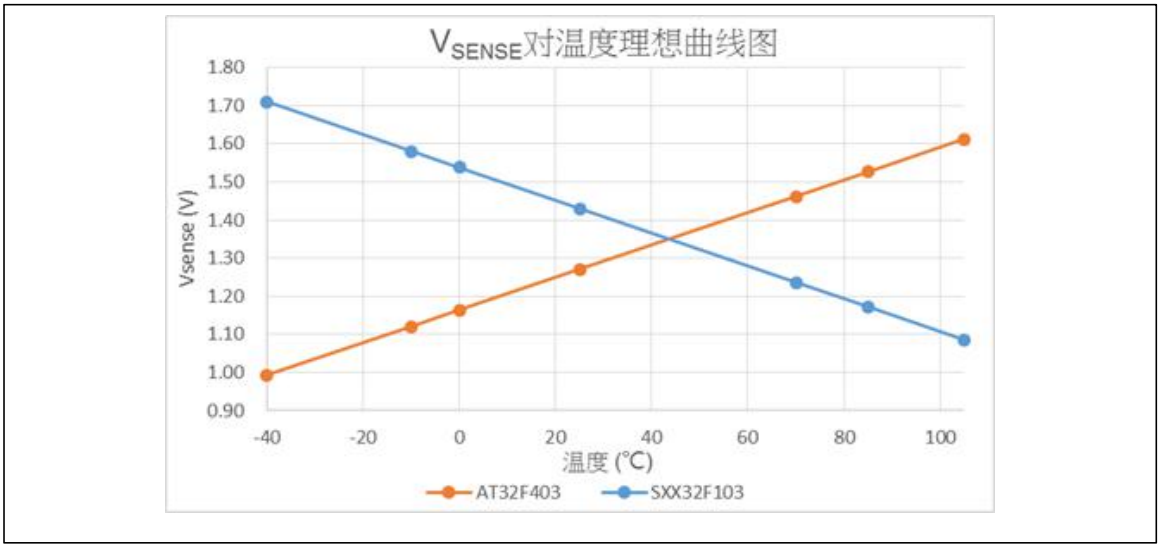
- 描述：
  初始化RCC时，当系统时钟SYSCLK从低频状态(HSE/HSI)切换到高频PLL状态(PLL >108MHz以上)可能导致程序挂起或进入hard fault。
- 需要特别注意的：
  - 当AT32F407内置的PLL为108MHz以上时钟时，PLL设定略有不同，需要操作自动滑顺频率切换功能
- 168MHz PLL使用范例：
  打开 system_sxx32f10x.c 找到当前的系统时钟频率配置函数（需经过上述 PLL 配置），如 168MHz 函数：
  static void SetSysClockTo168(void)
  增加如下斜黑体部分：

```
/* Wait till PLL is ready */
while((RCC->CR & RCC_CR_PLLRDY) == 0)
{
}
*((unsigned int    *)0x40021054) |= (0x30);//  开启自动滑顺频率切换功能

/* Select PLL as system clock source */
RCC->CFGR &= (uint32_t)((uint32_t)~(RCC_CFGR_SW));
RCC->CFGR |= (uint32_t)RCC_CFGR_SW_PLL;

/* Wait till PLL is used as system clock source */
while ((RCC->CFGR & (uint32_t)RCC_CFGR_SWS) != (uint32_t)0x08)
{
}
 *((unsigned int    *)0x40021054) &=~ (0x30); //关闭自动滑顺频率切换功能
```

3.2.2 内部温度传感器

- 描述：
  AT32F407 温度传感器为正温度系数，SXX32F107 为负温度系数
- 解决方法：
  按照数据手册中的值并利用下列公式得出温度：
  温度(℃) = {(V_{25} - V_{SENSE}) / Avg_Slope} + 25
  这里：
  V_{25} = V_{SENSE} 在 25°C 时的数值
  Avg_Slope ＝ 温度与 V_{SENSE} 曲线的平均斜率 (单位为 mV/℃).
  V_{25} 和 Avg_Slope 必须根据数据手册中的典型值参与运算, AT32F407 与 SXX32F107 不同。

图 4. V_{SENSE} 对温度理想曲线图



- 例程参考
  AT32F4xx_StdPeriph_Lib_V1.x.x\Project\Examples\AT_START_F407\ADC\Temperature

3.2.3 ADC 要求较长的采样时间

- 描述：
  ADC 采样电路为适应采样率提升至 2Msps，内部等效 R_{ADC} 和 C_{ADC} 值较大，要求较长的采样时间。特别在 ADC 输入源阻抗较大时，需满足足够的采样时间以获得准确的转换数据并消除不同 ADC 输入通道转换时的 cross-talk。
- 解决方法：
  使用时若遇到转换值不如预期，可先尝试设置采样时间至最大 239.5 个 ADC 时钟，再逐步减小采样时间至合适设置。若可接受较长采样间隔，在不同通道转换间插入 V_{REFINT} 转换也有让转换数据准确的效果。

3.2.4 GPIO 5V 容忍管脚兼容

- 描述：
  各封装的 PA11, PA12 与 64/48 管脚封装的 PD0, PD1 不属于 5V 电压输入容忍管脚，故这些管脚输入电平不可超过 VDD + 0.3V。
- 解决方法：
  使用时请留意此限制。

3.2.5 BOOT0 自带下拉电阻

- 描述：
  使用 BOOT0 引脚时，由于 BOOT0 内部自带阻值约为 90KΩ的下拉电阻（不可禁用），因此，在客户使用时，无需再额外外接下拉。

3.2.6 Standby 模式下自动使能 PA0 引脚下拉电阻

- 描述：
  当芯片进入 standby 模式下时，PA0 引脚的下拉电阻会由芯片内控制线路自动使能，以避免引脚浮空漏电。
- 解决方法：
- 请留意让PA0外部线路维持对此两引脚0V输入电平，否则系统会有微量额外耗电。

3.2.7 使用 USB 模块时系统时钟频率须不小于 12MHz

- 描述：
  系统时钟频率小于 12MHz 时，USB 无法正常收发数据。
- 解决方法：
  为满足 USB 正常收发数据的需求，系统时钟频率必须不小于 12MHz。

3.2.8 USB 双缓冲区 SW_BUF 和 DTOG 区别

- 描述：
  SXX32F107：当软件翻转SW_BUF，就认为释放一个缓冲，而不管SW_BUF和DTOG是否冲突，可以继续接收或发送数据，SXX32F107的设计与文档上说明不符。
  AT32：软件翻转SW_BUF，当SW_BUF和DTOG相等时，此时状态为NAK状态，不能接收或发送数据。完全按照文档说明设计。
- 解决方法：
  按照AT32F407参考手册说明使用

图 5. USB 双缓冲区使用说明

端点类型	DTOG 位	SW_BUF 位	USB 模块使用的缓冲区	应用程序使用的缓冲区
IN 端点	0	1	ADRn_TX_0 / CNTn_TX_0	ADRn_TX_1 / CNTn_TX_1
	1	0	ADRn_TX_1 / CNTn_TX_1	ADRn_TX_0 / CNTn_TX_0
	0	0	无(1)	ADRn_TX_0 / CNTn_TX_0
	1	1	无(1)	ADRn_TX_0 / CNTn_TX_0
OUT 端点	0	1	ADRn_RX_0 / CNTn_RX_0	ADRn_RX_1 / CNTn_RX_1
	1	0	ADRn_RX_1 / CNTn_RX_1	ADRn_RX_0 / CNTn_RX_0
	0	0	无(1)	ADRn_RX_0 / CNTn_RX_0
	1	1	无(1)	ADRn_RX_1 / CNTn_RX_1

3.2.9 DAC 输出有毛刺

- 描述：
  在使用Timer触发DAC输出波形的过程中去改变BFF位的状态来改变驱动能力，会导致毛刺的产生。
- 解决方法：
  在触发 DAC 输出之前配置好 BFF 位的状态。

3.2.10 Flash sector erase 更长

- 描述：
  因为 AT32F407 的 Flash sector erase 时间为 50ms 左右比 SXX32F107 的 22ms 更长。这会导致一些应用上的时间要求不同。如在开启 WWDG 后立即进入 Flash sector erase，则可能导致来不及喂狗而发生复位。
- 解决方法：
  针对上述应用，可以在擦除前先关闭 WWDG，擦除完成之后在开启 WWDG。

3.2.11 SDIO 在发生 RX over run 之后接收数据异常

- 描述：
  如果使用的FIFO相关标志，例如RX FIFO half full来做接收数据判断，则接收数据会出现异常现象。其原因在于，在发生了RX OVER RUN的时候，AT32会直接进入IDLE状态，FIFO相关标志都会被硬件清零，

SXXX32F107则保持在接收状态，不会清除FIFO相关标志。这里AT32设计更为合理。

- 解决方法：
  开 RX over run 中断，中断里重启 DPSM。

3.2.12 I2S 从机在特定条件下使能会导致通讯数据错误且无法自动恢复

- 描述：
  若从机使能在主机发送时钟的第 2 个 CLK 与第 3 个 CLK 之间时，会导致当前及之后的数据传输出现乱序。
- 解决方法：
  确保从机使能必须要在主机 CK 引脚上的时钟到达之前完成。

3.2.13 GPIO 在 50MHz 配置下高频输出时存在过冲现象

- 描述：
  对于 AT32F407，当 GPIO 端口配置为输出模式，选择最大输出频率为 50MHz 时，会存在输出高频信号存在过冲现象，有可能干扰电路并影响应用。
- 解决方法：
  - 情景一：对于常规使用，代码修改为GPIO_Speed_2MHz配置。
  - 情景二：对于强电流驱动，用户可根据AT32F407数据手册5.3.12的描述，依需求可配置成不同的速度参数MDEx。
- 情景一使用范例：
  - 针对使用 SXX32F107 BSP/Pack情形，修改方法如下黑体字部分

```
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_5;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
/* 以下修改为针对使用 SXX32F107 BSP/Pack 情形 */
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_2MHz;
GPIO_Init(GPIOB, &GPIO_InitStructure);
```

- 情景二使用范例：
  - 略。

3.2.14 使用 ADC 双模式时，程序会死在 ADC 校准函数

- 描述：
  - 在使用ADC双模式时，程序会死在ADC校准函数while(ADC_GetCalibrationStatus(ADC1));
- 解决方法：
  - 修改ADC使能和校准函数顺序，如原ADC初始化函数

```
ADC_Configuration()
{
…
ADC_Ctrl(ADC1, ENABLE);
…
while(ADC_GetCalibrationStatus(ADC1));//程序会死在这里

…
ADC_Ctrl(ADC2, ENABLE);
…
while(ADC_GetCalibrationStatus(ADC2));
…
}
```

- 修改方式一

```
ADC_Configuration()
{
…
ADC_Ctrl(ADC2, ENABLE);
…
while(ADC_GetCalibrationStatus(ADC2));

…
ADC_Ctrl(ADC1, ENABLE);
…
while(ADC_GetCalibrationStatus(ADC1));
…
}
```

- 修改方式二

```
ADC_Configuration()
{
…
ADC_Ctrl(ADC1, ENABLE);
…
ADC_Ctrl(ADC2, ENABLE);
…
while(ADC_GetCalibrationStatus(ADC1));
while(ADC_GetCalibrationStatus(ADC2));
…
…
}
```

3.2.15 使用 WWDG 中断时，无法清除 EWIF 标志

- 描述：
  - 在使用WWDG的中断时，在进入中断程序中WWDG_CNT = 0x40时无法清除EWIF标志，所以进入中断后需要先进行喂狗操作，再清除EWIF标志。

- 解决方法：
  - 在WWDG中断处理函数中，先进行喂狗操作，再清除EWIF标志。

```
void WWDG_IRQHandler(void)
{
  WWDG_SetCounter(127);
  WWDG_ClearFlag();
}
```

3.2.16 STOP Mode 下无法停掉 MCO 输出

- 描述：

  进入STOP Mode后，MCO还会有clk输出（LSICLK时钟频率），增加功耗

- 解决方法：

  在进入STOP模式前，通过软件关闭MCO输出，STOP唤醒后再重新配置

```
/********进 Stop 前先关闭 MCO 输出********/
RCC_CLKOUTConfig(RCC_CLKOUT_NOCLK, /*用户原始期望的分频配置*/);
PWR_EnterSTOPMode(PWR_Regulator_ON, PWR_STOPEntry_WFI);
/****** Stop 唤醒后重新配置 MCO 输出******/
RCC_CLKOUTConfig(/*用户原始期望的 MCO 时钟源*/, /*用户原始期望的分频配置*/);
```

3.2.17 Systick 中断异常唤醒执行 WFE 进入的 Stop Mode

- 描述：

  进入STOP Mode后，Systick未停止计数，当发生计数到0中断时，会误唤醒STOP

- 解决方法：

  在进Stop Mode之前先关闭Systick，待退出Stop Mode时再重新打开Systick

```
/********进 Stop 前先关闭 Systick********/
SysTick->CTRL &= 0xFFFFFFFE;
PWR_EnterSTOPMode(PWR_Regulator_ON, PWR_STOPEntry_WFE);
/****** Stop 唤醒后重新开启 Systick ******/
SysTick->CTRL |= 0x1;
```

3.2.18 ETH 中断号差异

- 描述：

  AT32F407 ETH全局中断号和唤醒中断号分别是IRQ79,IRQ80，与SXX32F107的中断号IRQ61，IRQ62有差异，如果直接使用SXX32F107代码移到AT32F407时要注意ETH中断号和中断向量表的修改。

- 解决方法：

  1. 修改ETH全局中断号和唤醒中断号为IRQ79，IRQ80。
  2. 修改中断向量表,中断向量表在.s 文件中定义，将ETH全局中断和唤醒中断入口放到新的位置。

3.2.19 USART 智能卡模式下接收数据异常

- 描述：

  初始化USART的智能卡模式时，若先使能USART再使能智能卡模式，USART 智能卡模式使能后的第一帧数据长度时段内无法接收数据。

- 解决方法：

  使能USART之前先使能智能卡模式。

```
/********先使能智能卡模式********/
    /* Enable the NACK Transmission */
    USART_SmartCardNACKCmd(USARTx, ENABLE);
    /* Enable the Smartcard Interface */
    USART_SmartCardCmd(USARTx, ENABLE);

/******配置完成后再使能 USART ******/
    USART_Cmd(USARTx, ENABLE);
```

3.2.20 PWR 以 WFE 进入 low-power 模式的差别

- AT32F407系列采用Cortex-M4内核，WFE操作有所区别，需要按如下使用范例操作。
- 使用范例：

```
/* Request Wait For Event */
__SEV();
__WFE();
__WFE();
```

3.2.21 CAN接收数据域期间出现位填充错误时导致下一帧数据错位

- 描述：
  CAN在接收数据域期间若出现“位填充错误”时，将导致收到的下一帧数据出现错位，但后续帧又自动恢复正常的现象。
- 解决方法：
  开启CAN的上次错误中断号对应的错误中断，在CAN错误中断的中断函数内检测到出现位填充错误时，复位CAN，并重新调用CAN初始化函数。以CAN1为例，其典型示例代码如下：

```
__IO uint32_t err_index = 0;
void CAN1_SCE_IRQHandler(void)
{
    if (CAN_GetINTStatus (CANx,CAN_INT_LEC) == SET )
    {
        err_index = CAN1->ESTS;
        CAN_ClearINTPendingBit (CANx, CAN_INT_LEC);
        if(err_index & 0x00000010) ///<判定是否出现位填充错误
        {
            CAN_Reset(CAN1);
            /*调用 CAN 初始化函数*/;
        }
    }
}
```

4 版本历史

表 7. 文档版本历史

日期	版本	变更
2020.03.16	1.0.0	最初版本
2020.05.17	1.0.1	修改中断向量简写，如#60修改为IRQ60
2020.06.11	1.0.2	增加*STOP Mode 下无法停掉MCO输出* 增加*Systick 中断异常唤醒执行WFE进入的Stop Mode*
2020.06.30	1.0.3	增加AT32F407 ETH中断号与SXX32F107 ETH中断号的差异描述
2020.08.11	1.0.4	增加*USART智能卡模式下接收数据异常*
2021.03.04	1.0.5	新增*PWR以WFE进入low-powe模式的差别*
2021.05.06	1.0.6	增加CAN接收数据域期间出现位填充错误时导致下一帧数据错位

重要通知 - 请仔细阅读

买方自行负责对本文所述雅特力产品和服务的选择和使用，雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为雅特力授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明，否则，雅特力对雅特力产品的使用和 / 或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途（及其依据任何司法管辖区的法律的对应情况），或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力产品并非设计或专门用于下列用途的产品：(A) 对安全性有特别要求的应用，如：生命支持、主动植入设备或对产品功能安全有要求的系统；(B) 航空应用；(C) 汽车应用或汽车环境；(D) 航天应用或航天环境，且/或(E) 武器。因雅特力产品不是为前述应用设计的，而采购商擅自将其用于前述应用，即使采购商向雅特力发出了书面通知，风险由购买者单独承担，并且独力负责在此类相关使用中满足所有法律和法规要求。

经销的雅特力产品如有不同于本文档中提出的声明和 / 或技术特点的规定，将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大雅特力的任何责任。