NV32F100x SIM 模块编程示例

第一章 库函数简介

1.1 库函数列表

void SIM_Init(SIM_ConfigType *pConfig)

通过结构体 SIM_ConfigType 初始化 SIM 模块

SIM_SetClockGating(uint32_t u32PeripheralMask, uint8_t u8GateOn)

SIM 系统时钟选通控制函数

uint32_t SIM_GetStatus(uint32_t u32StatusMask)

读取系统复位状态函数

uint8_t SIM_ReadID(IDType sID)

读取 ID 寄存器值函数

寄存器操作的内联函数，调用内联函数和直接操作寄存器的效率一样

void SIM_DelayETM2Trig2ADC(uint8_t u8Delay)

ETM2 触发延时

void SIM_EnableClockOutput(void)

使能总线时钟输出

void SIM_DisableClockOutput(void)

禁止总线时钟输出

void SIM_SetClockOutputDivide(uint8_t u8Divide)

设置总线时钟输出分频

void SIM_EnableUART0RXDConnectETMOCH1(void)

UART0_RX 输入信号连接到 UART0 模块和 ETM0 通道 1

void SIM_EnableUART0Modulation(void)

UART0_TX 输出在映射到引出线前由 ETM0 通道 0 调制

void SIM_DisableUART0Modulation(void)

UART0_TX 输出直接连接到输出引线

void SIM_GenerateSoftwareTrig2ETM2(void)

生成 ETM2 模块的同步触发

void SIM_RemapETM2CH3Pin(void)

ETM2_CH3 通道输出映射到 PD1 上

void SIM_RemapETM2CH2Pin(void)

ETM2_CH2 通道输出映射到 PD0 上

void SIM_RemapETM0CH1Pin(void)

ETM0_CH1 通道输出映射到 PB3 上

void SIM_RemapETM1CH1Pin(void)

ETM1_CH1 通道映射到 PE7 上

void SIM_RemapETM0CH0Pin(void)

ETM0_CH0 通道映射到 PB2 上

void SIM_RemapUART0Pin(void)

UART0_RX 和 UART0_TX 映射到 PA2 和 PA3 上

void SIM_RemapSPI0Pin(void)

SPI0_SCK、SPI0_MOSI、SPI0_MISO 和 SPI0_PCS 映射到 PE0、PE1、PE2 和 PE3 上

void SIM_RemapI2CPin(void)

I2C0_SCL 和 I2C0_SDA 分别映射到 PB7 和 PB6 上

void SIM_EnableUART0Filter(void)

RXD0 输入信号由 ACMP0 滤波，然后注入 UART0

void SIM_DisableUART0Filter(void)

RXD0 输入信号直接连接到 UART0 模块

void SIM_TriggerADCByRTC(void)

用作 ADC 硬件触发的 RTC 溢出

void SIM_TriggerADCByPIT(void)

用作 ADC 硬件触发的 PIT 通道溢出

void SIM_TriggerADCByETM2Init(void)

带 8 位可编程计数器延迟的 ETM2 初始触发

void SIM_TriggerADCByETM2Match(void)

带 8 位可编程计数器延迟的 ETM2 匹配触发

void SIM_EnableRTCCapture(void)

RTC 溢出连接到 ETM1 输入通道 1

void SIM_DisableRTCCapture(void)

RTC 溢出未连接到 ETM1 输入通道 1

void SIM_EnableACMP0InputCapture(void)

ACMP0 输出连接到 ETM1 输入通道 0

void SIM_DisableACMP0InputCapture(void)

ACMP0 输出未连接到 ETM1 输入通道 0

void SIM_RemapRTCPin(void)

RTC0 映射到 PC5 上

void SIM_SetBusDivide(uint8_t u8Divide)

设置总线时钟分频

void SIM_RemapETM2CH1Pin(void)

ETM2_CH1 通道输出映射到 PH1 上

void SIM_RemapETM2CH0Pin(void)

ETM2_CH0 通道输出映射到 PH0 上

void SIM_RemapETM1CH0Pin(void)

ETM1_CH0 通道映射到 PH2 上

1.2 SIM 模块特性说明

*复位状态和器件 ID 信息

*系统互连配置和特殊引脚的启用

*引脚再映射控制

*系统时钟选通控制和时钟分频

1.3 SIM 模块使用说明

*当使用某一模块时，系统时钟选通控制寄存器的对应为置 1，默认状态下开启 SWD 和 FLASH 模块系统时钟

*对引脚选通寄存器配置选择外设输出映射引脚，默认状态引脚选通寄存器每一位值都为 0

*设置系统选项寄存器，默认状态使能 SWD、RESET、NMI 引脚。

*SIM 模块结构体在系统初始化函数中配置，主要是对系统选项寄存器的配置。

关于引脚选通、系统时钟等控制具体参阅参考手册

第二章 模块初始化

在 SIM 初始化函数函数中，主要配置系统选项寄存器和使能 SWD、FLASH 模块的总线时钟。

2.1 SIM 系统选项寄存器(SIM_SOPT)

位	描述
31－24 DELAY	ETM2 触发延迟 指定将 1 写入 ADHWT 时从 ETM2 初始或匹配触发到 ADC 硬件触发的延迟。该 8 位模数值允许 0 到 255 的延迟，具体取决于 BUSREF 时钟设置。这是一个一次性计数器，当触发到达时开始计数，当计数器值达到所定义的模数值时停止计数。
23 DLYACT	ETM2 触发延迟有效 该只读字段指定有关 ETM2 初始或匹配延迟是否有效的状态。该字段在 ETM2 触发到达且

	延迟计数器正在计数时置位，否则，该字段清零。 0 延迟无效。 1 延迟有效。
22－21 保留	此字段为保留字段。 此只读字段为保留字段且值始终为 0。
20 FLASHDP	Flash Deep Sleep 使能控制 0 Flash 在 STOP 模式下不进入 Deep Sleep 1 Flash 在 STOP 模式下进入 Deep Sleep
19 CLKOE	总线时钟输出使能 0 总线时钟输出在 PTH2 上禁用。 1 总线时钟输出在 PTH2 上使能。
18－16 BUSREF	总线时钟输出选择 通过可选预分频器使能总线时钟输出。 000 总线 001 总线 2 分频 010 总线 4 分频 011 总线 8 分频 100 总线 16 分频 101 总线 32 分频 110 总线 64 分频 111 总线 128 分频
15 TXDME	UART0_TX 调制选择 使能由 ETM0 通道 0 调制的 UART0_TX 输出。 0 UART0_TX 输出直接连接到引出线。 1 UART0_TX 输出在映射到引出线前由 ETM0 通道 0 调制。
14 ETMSYNC	ETM2 同步选择 将 1 写入该字段时生成 ETM2 模块的 PWM 同步触发。 0 未触发任何同步。 1 生成 ETM2 模块的 PWM 同步触发。
13 RXDFE	UART0 RxD 滤波器选择 使能 UART0 RxD 输入由 ACMP 滤波。该功能使能时，任何具有 ACMP 输入标记的信号都可被视作 UART0。 0 RXD0 输入信号直接连接到 UART0 模块。 1 RXD0 输入信号由 ACMP0 滤波，然后注入 UART0。
12 RXDCE	UART0_RX 捕捉选择 使能 UART0_RX 由 ETM0 通道 1 捕捉。 0 UART0_RX 输入信号仅连接到 UART0 模块。 1 UART0_RX 输入信号连接到 UART0 模块和 ETM0 通道 1。
11 ACIC	模拟比较器至输入捕捉使能 将 ACMP0 的输出连接到 ETM1 输入通道 0。 0 ACMP0 输出未连接到 ETM1 输入通道 0。 1 ACMP0 输出连接到 ETM1 输入通道 0。
	实时计数器捕捉

10 RTCC	允许实时计数器(RTC)溢出由 ETM1 通道 1 捕捉。 0 RTC 溢出未连接到 ETM1 输入通道 1。 1 RTC 溢出连接到 ETM1 输入通道 1。
9－8 ADHWT	ADC 硬件触发源 选择 ADC 硬件触发源，所有触发源都是在上升沿开始 ADC 转换。 00 用作 ADC 硬件触发的 RTC 溢出 01 用作 ADC 硬件触发的 PIT 通道溢出 10 带 8 位可编程计数器延迟的 ETM2 初始触发 11 带 8 位可编程计数器延迟的 ETM2 匹配触发
7-4 保留	此字段为保留字段。
3 SWDE	单线调试端口引脚使能 使能 PA4/ACMP0_OUT/SWD_DIO 引脚用作 SWD_DIO，使能 PC4/RTC_CLKOUT/ETM1_CH0/ACMP0_IN2/SWD_CLK 引脚用作 SWD_CLK。清零时，两个引脚用作 PA4 和 PC4。该引脚在任何 MCU 复位之后默认用作 SWD_DIO 和 SWD_CLK。 0 PA4/ACMP0_OUT/SWD_DIO 作为 PA4 或 ACMP0_OUT 功能，PC4/RTC_CLKOUT/ETM1_CH0/ACMP0_IN2/SWD_CLK 用作 PC4、RTC_CLKOUT、ETM1_CH0、OR ACMP0_IN2 功能。 1 PA4/ACMP0_OUT/SWD_DIO 用作 SWD_DIO 功能，PC4/RTC_CLKOUT/ETM1_CH0/ACMP0_IN2/SWD_CLK 用作 SWD_CLK 功能。
2 RSTPE	RESET 引脚使能 在任何复位后都可对该一次性写入字段进行写操作。RSTPE 置位时，PA5/IRQ/TCLK0/RESET引脚用作RESET。清零时，该引脚用作备用功能之一。该引脚在 MCU POR 之后默认用作RESET。其他复位不会影响该字段。RSTPE 置位时，RESET上的内部上拉器件使能。 0 PA5/IRQ/TCLK0/RESET引脚用作 PA5/IRQ/TCLK0。 1 PA5/IRQ/TCLK0/RESET引脚用作RESET。
1 NMIE	NMI引脚使能 在任何复位后都可对该一次性写入字段进行写操作。NMI 置位时，PB4/ETM2_CH4/SPI0_MISO/ACMP1_IN2/NMI引脚用作NMI。清零时，该引脚用作备用功能之一。该引脚在 MCU POR 之后默认用作NMI。其他复位不会影响该位。NMI置位时，NMI上的内部上拉器件使能。 0 PB4/ETM2_CH4/SPI0_MISO/ACMP1_IN2/NMI引脚用作 PB4、ETM2_CH4、SPI0_MISO 或 ACMP1_IN2。 1 PB4/ETM2_CH4/SPI0_MISO/ACMP1_IN2/NMI引脚用作NMI。
0 保留	此字段为保留字段。 此只读字段为保留字段且值始终为 0。

2.2 系统时钟选通控制寄存器(SIM_SCGC)

系统选通寄存器完整信息请参看参考手册

17 I2C	I2C 时钟选通控制 控制 I2C 模块的时钟选通。 0 I2C 模块的总线时钟禁用。 1 I2C 模块的总线时钟使能。
16-14 保留	此字段为保留字段。 此只读字段为保留字段且值始终为 0。
13 SWD	SWD（单线调试器）时钟选通控制 控制 SWD 模块的时钟选通。 0 SWD 模块的总线时钟禁用。 1 SWD 模块的总线时钟使能。
12 FLASH	Flash 时钟选通控制 控制 Flash 模块的时钟选通。 0 Flash 模块的总线时钟禁用。 1 Flash 模块的总线时钟使能。

2.3 系统引脚寄存器(SIM_PINSEL)

引脚选择寄存器完整信息请参看参考手册

5 I2C0PS	I2C0 端口引脚选择 选择 I2C0 端口引脚。 0 I2C0_SCL 和 I2C0_SDA 分别映射到 PA3 和 PA2 上。 1 I2C0_SCL 和 I2C0_SDA 分别映射到 PB7 和 PB6 上。

函数名	SIM_Init
函数原形	SIM_Init(SIM_ConfigType *pConfig)
功能描述	配置结构体 pConfig 来初始化 SIM
输入参数	配置结构体 SIM_ConfigType
输出参数	无
返回值	无
先决条件	无
函数使用实例	先设置配置结构体，SIM_Init(&sSIMConfig);

```
/*****************************************************************************//*!
    *
    * @ 概要  初始化 SIM 寄存器
    *
    * @ 参数【输入】 pConfig     指向 SIM 配置结构体.
    *
    * @ 无返回
    *
    *****************************************************************************/
     void SIM_Init(SIM_ConfigType *pConfig)
     {
```

```
    uint32_t    u32Sopt;
    uint32_t    u32PinSel;
    uint32_t    u32Scgc;
    uint32_t    u32BusDiv;
    /* 初始化 SIM 寄存器，设置写入引脚选通寄存器的数值来控制引脚映射，设置写入系统时钟选通寄存器的值来控制各模块的时钟选通。在系统初始化函数中设置 SIM 结构体，实现对系统选项寄存器配值 */
    u32Sopt     = 0x0010000E;                   /* 使能 SWD、RESET、NMI 引脚 */
    u32PinSel   = 0;
    u32Scgc     = 0x00003000;                   /* 使能 SWD、FLASH 模块的总线时钟 */
    u32BusDiv   = 0;
    u32BusDiv = pConfig->sBits.bBusDiv;     /*总线时钟分频值*/
    if(pConfig->sBits.bDisableNMI)          /*禁用 NMI 引脚*/
    {
        u32Sopt &= ~SIM_SOPT_NMIE_MASK;
    }
    if(pConfig->sBits.bDisableRESET)        /*禁用 RSTPE 引脚*/
    {
        u32Sopt &= ~SIM_SOPT_RSTPE_MASK;
    }
    if(pConfig->sBits.bDisableSWD)          /*禁用 SWDE 引脚*/
    {
        u32Sopt &= ~SIM_SOPT_SWDE_MASK;
    }
    if(pConfig->sBits.bEnableCLKOUT)        /*使能总线时钟输出*/
    {
        u32Sopt |= SIM_SOPT_CLKOE_MASK;
    }
    if(pConfig->sBits.bETMSYNC)             /*ETM2 同步选择*/
    {
        u32Sopt |= SIM_SOPT_ETMSYNC_MASK;   /*生成 ETM2 模块的 PWM 同步触发*/
    }
    if(pConfig->sBits.bRXDCE)               /*UAT0_RX 捕捉选择*/
    {

       /*UAT0_RX 输入信号接到 UART0 模块和 ETM0 通道 1*/
        u32Sopt |= SIM_SOPT_RXDCE_MASK;
    }
    if(pConfig->sBits.bTXDME)               /*URAT0_TX 捕捉选择*/
    {
        /*URAT0_TX 输出映射到引出线前由 ETM0 通道调制*/
        u32Sopt |= SIM_SOPT_TXDME_MASK;
    }
    if(pConfig->sBits.bACIC)                /*模拟比较器至输入捕获使能*/
    {
```

```
            u32Sopt |= SIM_SOPT_ACIC_MASK;      /* ACMP0 输出连接到 ETM1 输出通道 0*/
        }
            if(pConfig->sBits.bRTCC)
        {
            u32Sopt |= SIM_SOPT_RTCC_MASK;      /*RTC 溢出连接到 ETM1 输入通道*/
        }
            if(pConfig->sBits.bRXDFE)                /*URT0 RxD 滤波器选择*/
        {
          /*RXD0 输入信号由 ACMP0 滤波，然后注入 UART0*/
            u32Sopt |= SIM_SOPT_RXDFE_MASK;
        }
        u32Sopt |= ((pConfig->u8BusRef & 0x07) << 16);      /*总线时钟 128 分频*/
        u32Sopt |= ((pConfig->u8Delay) << 24);           /*ETM2 延迟触发*/

       /*ADC 的触发源为带 8 位可编程计数延迟的 ETM2 匹配触发*/
        u32Sopt |= ((pConfig->sBits.u8ADHWT & 0x03) << 8);
        u32PinSel = pConfig->u32PinSel;
        u32Scgc = pConfig->u32SCGC;
       /*写数据到 SIM 寄存器*/
        SIM->SOPT = u32Sopt;
        SIM->PINSEL = u32PinSel;
        SIM->SCGC = u32Scgc;
        SIM->BUSDIV = u32BusDiv;
    }
```

注：

1. SIM 模块结构体在系统初始化函数中配置，对系统选项寄存器配置，在系统初始化函数中进行。详细内容请参看系统初始化函数。

2. SIM 初始化函数中，只控制了 SWD 和 FLASH 两个模块的时钟选通，其他模块的总线时钟没有选通。如果要选通其他模块的总线时钟有三种方法；

方法 1:：更改写入时钟选通寄存器数值，模块对应位写 1，可参看 SWD 和 FLASH 模块时钟选通控制。

方法 2：直接调用 SIM 系统时钟选通控制函数。

方法 3：在对应 模块初始化时直接对时钟选通寄存器进行操作。向寄存器中该模块的对应位写 1（常用方法）

例：使用方法 3 使能 I2C 总线时钟，只需在 I2C 初始化函数中加入；

```
SIM->SCGC |= SIM_SCGC_IIC_MASK;
```

3. SIM 初始化函数中，对引脚寄存器每一位写入的数值均为 0，要改变模块的引脚输出映射有三种方法；

方法 1：在 SIM 初始化函数中更该写入引脚寄存器的值，

方法 2：调用对应内联函数

方法 3：直接操作引脚选择寄存器；

例：使用方法 3 选择 I2C 引脚输出，当系统初始化后 I2C 的 I2C0_SCL 和 I2C0_SDA 分别映射到 PA3 和 PA2 上，当要使 I2C0_SCL 和 I2C0_SDA 分别映射到 PB7 和 PB6 上，只需要在引脚控制寄存器对应为写 1。

```
SIM->PINSEL |=SIM_PINSEL_IICPS_MASK
```