第5章 chapter 5

STM32F103的中断

本章以STM32F103为例，介绍中断的基本概念、中断系统、外部中断/事件控制器EXTI和开发实例。

5.1 中断的概念

在嵌入式系统应用中，当内部、外部紧急事件发生时，能及时响应并实时处理都是利用微控制器的中断系统实现的。

微控制器在执行程序的过程中，被内部或外界中断源打断，微控制器响应中断请求、执行中断服务程序之后，返回断点继续执行原来程序，整个过程称为“中断”，如图5-1所示。

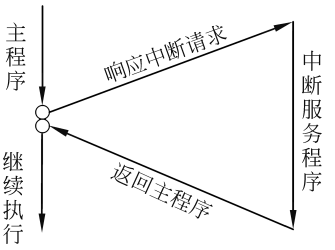

图5-1 中断响应和处理过程图

5.2 STM32F103的中断系统

中断系统是软件系统与硬件系统共同提供的功能。硬件系统包括中断源、中断通道、嵌套向量中断控制器(NVIC)、中断优先级、中断向量表，软件主要是中断服务程序。

5.2.1 中断源

中断源指能引发中断的事件。在嵌入式系统应用中，常见的中断源有定时器溢出、串口接收到数据、串口发送完数据、I2C发送完数据、I2C接收到数据、按键按下和释放

等,与此相关的中断有定时器中断、串口中断、I2C 中断和外部中断等。中断源是否有中断请求,是由它对应的中断请求标志位来表示的。

5.2.2 中断通道

STM32 采用中断通道管理中断源,一个中断通道可具有多个可以申请中断的中断源,这些中断源都能通过对应的"中断通道"向 CPU 申请中断。每个中断通道对应唯一的中断向量地址和唯一的中断服务程序。

5.2.3 嵌套向量中断控制器

嵌套向量中断控制器(NVIC),集成在 ARM Cortex-M3 内核中,与 ARM 内核逻辑紧密耦合。NVIC 最多支持 256 个异常(包括 16 个内部异常和 240 个非内核异常中断)和 256 级可编程异常优先级。而 STM32F103 微控制器的中断系统并没有使用内核 Cortex-M3 的 NVIC 全部功能,它的 NVIC 具有以下特性:

(1) 支持 84 个异常,包括 16 个内部异常和 68 个非内核异常中断。

(2) 每个中断源使用 4 位优先级设置(ARM Cortex-M3 内核定义了 8 位,STM32 微控制器只使用了其中的 4 位),具有 16 级可编程异常优先级。用户可以根据实际应用编程设定 4 位优先级中抢占优先级的位数和子优先级的位数。

(3) 中断响应时处理器状态会自动保存,无须额外指令。

(4) 中断返回时处理器状态会自动恢复,无须额外指令。

(5) 支持嵌套和向量中断。

(6) 支持中断尾链技术。

5.2.4 STM32 的中断优先级

中断优先级的概念是针对"中断通道"的。当中断通道的优先级确定后,该中断通道对应的所有中断源都享有相同的中断优先级。至于该中断通道对应的多个中断源的执行顺序,则取决于用户的中断服务程序。

NVIC 通过设置优先级,使得多个中断源同时申请,按优先级高低顺序处理。STM32 中断优先级,分为抢占优先级(preempting priority)和子优先级(sub priority)。

(1) 抢占优先级。抢占优先级又称组优先级或者占先优先级,决定了是否会有中断嵌套发生。抢占优先级编号低的中断比抢占优先级编号高的优先级高。

(2) 子优先级。子优先级又称从优先级,子优先级高的中断不会构成中断嵌套。子优先级编号低的中断比子优先级编号高的优先级高。

1. STM32 的中断优先级嵌套规则

STM32 的中断优先级嵌套规则如下:

(1) 高抢占优先级的中断(抢占优先级编号低的中断)可以打断当前正在执行的低抢占优先级的中断(抢占优先级编号高的中断)服务程序,从而执行高抢占优先级中断对应

的中断服务程序。

(2) 仅在抢占优先级相同但子优先级不同的多个中断通道同时申请服务时,STM32首先响应子优先级高(子优先级编号低)的中断。

(3) 当相同抢占优先级和相同子优先级的中断通道同时申请服务时,STM32首先响应中断向量表中地址低(中断号小)的那个中断通道。

2. STM32的中断优先级设置

STM32微控制器的中断源的优先级分为5组,分别是0～4组,每组使用4位优先级控制位设置抢占优先级和子优先级。因而具有16级可编程异常优先级。中断优先级分组和4位中断控制位的关系如表5-1所示。

表5-1 中断优先级控制位与分组方式

组 号	优先级控制位					说 明
	bit7	bit6	bit5	bit4	bit3～bit0	
0	全设置为子优先级				未用	无抢占优先级,有16个子优先级
1	抢占优先级	子优先级				有2个抢占优先级,8个子优先级
2	抢占优先级		子优先级			有4个抢占优先级,4个子优先级
3	抢占优先级			子优先级		有8个抢占优先级,2个子优先级
4	全设置为抢占优先级					有16个抢占优先级,无子优先级

由表5-1可知,4位中断优先级控制位(bit7～bit4)确定了5组优先级。从高位bit7开始,先定义抢占优先级的位,后面是子优先级的位。4位中断优先级控制位决定了抢占优先级和子优先级的数目。

(1) 0组:高4位都用于子优先级,子优先级编号可设置为0～15;

(2) 1组:最高1位用于抢占优先级,抢占优先级编号可设置为0～1,低3位用于子优先级,编号可设置为0～7;

(3) 2组:高2位用于抢占优先级,编号可设置为0～3,低2位用于子优先级,编号可设置为0～3;

(4) 3组:高3位用于抢占优先级,编号可设置为0～7,最低1位用于子优先级,编号可设置为0～1;

(5) 4组:所有4位都用于抢占优先级,编号可设置为0～15。

在STM32固件库misc.h中,分组设置数据被写入AIRCR寄存器的[10:8]内,其宏定义如下:

```
#define NVIC_PriorityGroup_0 ((u32) 0x700)          // 0组定义
#define NVIC_PriorityGroup_1 ((u32) 0x600)          // 1组定义
```

```
#define NVIC_PriorityGroup_2 ((u32) 0x500)          // 2组定义
#define NVIC_PriorityGroup_3 ((u32) 0x400)          // 3组定义
#define NVIC_PriorityGroup_4 ((u32) 0x300)          // 4组定义
```

3. 编程中优先级分组的设置

利用优先级设置函数 NVIC_PriorityGroupConfig()进行设置。

【例 5-1】 选择使用优先级第 1 组。

```
NVIC_PriorityGroupConfig(NVIC_PriorityGroup_1);
```

5.2.5 STM32F103 的中断向量表

1. STM32F103 中断向量表说明

STM32F103 微控制器的 NVIC 支持 84 个异常中断(包括 16 个内部异常和 68 个非内核异常中断),对应的中断服务程序的入口地址统一存放在 STM32F103 的中断向量表中。STM32F103 的中断向量表一般存于存储器的 0 地址处,如表 5-2 所示,说明如下。

表 5-2 STM32F103 的中断向量表

位置	中断名称	说　明	优先级	优先级类型	相对地址
—	—	栈顶地址(MSP 初值)	—	—	0x0000 0000
—	Reset	复位	−3(最高)	固定	0x0000 0004
—	NMI	不可屏蔽中断,连接 RCC 时钟安全系统(CSS)	−2	固定	0x0000 0008
—	HardFault	所有类型的失效	−1	固定	0x0000 000C
—	MemManage	存储器管理错误	0	可设置	0x0000 0010
—	BusFault	预取指令失败	1	可设置	0x0000 0014
—	UsageFault	未定义的指令或非法状态	2	可设置	0x0000 0018
—	—	保留	—	—	0x0000 001C
—	—	保留	—	—	0x0000 0020
—	—	保留	—	—	0x0000 0024
—	—	保留	—	—	0x0000 0028
—	SVCall	通过 SWI 指令的系统服务调用	3	可设置	0x0000 002C
—	DebugMonitor	调试监控器	4	可设置	0x0000 0030
—	—	保留	—	—	0x0000 0034
—	PendSV	可挂起的系统服务	5	可设置	0x0000 0038

续表

位置	中断名称	说　明	优先级	优先级类型	相对地址
—	SysTick	系统嘀嗒定时器	6	可设置	0x0000 003C
0	WWDG	窗口看门狗定时器中断	7	可设置	0x0000 0040
1	PVD	连接到 EXTI 的电源电压检测(PVD)中断	8	可设置	0x0000 0044
2	TAMPER	入侵检测中断	9	可设置	0x0000 0048
3	RTC	实时时钟全局中断	10	可设置	0x0000 004C
4	FLASH	闪存全局中断	11	可设置	0x0000 0050
5	RCC	复位和时钟控制中断	12	可设置	0x0000 0054
6	EXTI0	EXTI 线 0 中断	13	可设置	0x0000 0058
7	EXTI1	EXTI 线 1 中断	14	可设置	0x0000 005C
8	EXTI2	EXTI 线 2 中断	15	可设置	0x0000 0060
9	EXTI3	EXTI 线 3 中断	16	可设置	0x0000 0064
10	EXTI4	EXTI 线 4 中断	17	可设置	0x0000 0068
11	DMA1 通道 1	DMA1 通道 1 全局中断	18	可设置	0x0000 006C
12	DMA1 通道 2	DMA1 通道 2 全局中断	19	可设置	0x0000 0070
13	DMA1 通道 3	DMA1 通道 3 全局中断	20	可设置	0x0000 0074
14	DMA1 通道 4	DMA1 通道 4 全局中断	21	可设置	0x0000 0078
15	DMA1 通道 5	DMA1 通道 5 全局中断	22	可设置	0x0000 007C
16	DMA1 通道 6	DMA1 通道 6 全局中断	23	可设置	0x0000 0080
17	DMA1 通道 7	DMA1 通道 7 全局中断	24	可设置	0x0000 0084
18	ADC1_2	ADC1 和 ADC2 的全局中断	25	可设置	0x0000 0088
19	USB _ HP _ CAN _TX	USB 高优先级或 CAN 发送中断	26	可设置	0x0000 008C
20	USB _ LP _ CAN _RX0	USB 低优先级或 CAN 接收 0 中断	27	可设置	0x0000 0090
21	CAN_RX1	CAN 接收 1 中断	28	可设置	0x0000 0094
22	CAN_SCE	CAN 的 SCE 中断	29	可设置	0x0000 0098
23	EXTI9_5	EXTI[9:5]中断	30	可设置	0x0000 009C
24	TIM1_BRK	TIM1 刹车中断	31	可设置	0x0000 00A0
25	TIM1_UP	TIM1 更新中断	32	可设置	0x0000 00A4
26	TIM1_TRG_COM	TIM1 触发和通信中断	33	可设置	0x0000 00A8

续表

位置	中断名称	说　明	优先级	优先级类型	相对地址
27	TIM1_CC	TIM1 捕获比较中断	34	可设置	0x0000 00AC
28	TIM2	TIM2 全局中断	35	可设置	0x0000 00B0
29	TIM3	TIM3 全局中断	36	可设置	0x0000 00B4
30	TIM4	TIM4 全局中断	37	可设置	0x0000 00B8
31	I2C1_EV	I2C1 事件中断	38	可设置	0x0000 00BC
32	I2C2_ER	I2C1 错误中断	39	可设置	0x0000 00C0
33	I2C2_EV	I2C2 事件中断	40	可设置	0x0000 00C4
34	I2C2_ER	I2C2 错误中断	41	可设置	0x0000 00C8
35	SPI1	SPI1 全局中断	42	可设置	0x0000 00CC
36	SPI2	SPI2 全局中断	43	可设置	0x0000 00D0
37	USART1	USART1 全局中断	44	可设置	0x0000 00D4
38	USART2	USART2 全局中断	45	可设置	0x0000 00D8
39	USART3	USART3 全局中断	46	可设置	0x0000 00DC
40	EXTI15_10	EXTI 线[15:10]中断	47	可设置	0x0000 00E0
41	RTC Alarm	连接到 EXTI 的 RTC 闹钟中断	48	可设置	0x0000 00E4
42	USB WakeUp	连接到 EXTI 的从 USB 待机唤醒中断	49	可设置	0x0000 00E8
43	TIM8_BRK	TIM8 刹车中断	50	可设置	0x0000 00EC
44	TIM8_UP	TIM8 更新中断	51	可设置	0x0000 00F0
45	TIM8_TRG_COM	TIM8 触发和通信中断	52	可设置	0x0000 00F4
46	TIM8_CC	TIM8 捕获比较中断	53	可设置	0x0000 00F8
47	ADC3	ADC3 全局中断	54	可设置	0x0000 00FC
48	FSMC	FSMC 全局中断	55	可设置	0x0000 0100
49	SDIO	SDIO 全局中断	56	可设置	0x0000 0104
50	TIM5	TIM5 全局中断	57	可设置	0x0000 0108
51	SPI3	SPI3 全局中断	58	可设置	0x0000 010C
52	UART4	UART4 全局中断	59	可设置	0x0000 0110
53	UART5	UART5 全局中断	60	可设置	0x0000 0114
54	TIM6	TIM6 全局中断	61	可设置	0x0000 0118
55	TIM7	TIM7 全局中断	62	可设置	0x0000 011C
56	DMA2 通道 1	DMA2 通道 1 全局中断	63	可设置	0x0000 0120

续表

位置	中断名称	说　明	优先级	优先级类型	相对地址
57	DMA2通道2	DMA2通道2全局中断	64	可设置	0x0000 0124
58	DMA2通道3	DMA2通道3全局中断	65	可设置	0x0000 0128
59	DMA2通道4	DMA2通道4全局中断	66	可设置	0x0000 012C
60	DMA2通道5	DMA2通道5全局中断	67	可设置	0x0000 0130
61	ETH	以太网全局中断	68	可设置	0x0000 0134
62	ETH_WKUP	连接到EXTI的以太网唤醒中断	69	可设置	0x0000 0138
63	CAN2_TX	CAN2发送中断	70	可设置	0x0000 013C
64	CAN2_RX0	CAN2接收0中断	71	可设置	0x0000 0140
65	CAN2_RX1	CAN2接收1中断	72	可设置	0x0000 0144
66	CAN2_SCE	CAN2的SCE中断	73	可设置	0x0000 0148
67	OTG_FS	全速USBOTG全局中断	74	可设置	0x0000 014C

(1) 位置越低,优先级越小,表明优先级越高。表中除个别中断的优先级被固定,其他都具有16级可编程中断优先级可以设置。

(2) 表中最前面的16行为ARM Cortex-M3内核异常;从优先级7到优先级74,是STM32F103微控制器的NVIC支持的68个可屏蔽中断通道。

(3) 在固件库stm32f10x.h文件中,用宏定义将中断号(中断向量位置)和宏名联系起来。如表5-2中TIM3的中断号是29,USART1中断号为37,中断号宏定义分别为

```
TIM3_IRQn=29,
USART1_IRQn=37,
```

(4) 中断向量表中的地址为相对地址,如果中断向量存放在RAM中,其起始地址为0x2000 0000开始的区域。如果存放在Flash中,其起始地址为0x0800 0000。在misc.h文件中有如下声明:

```
#define NVIC_VectTab_RAM      ((uint32_t) 0x20000000)
#define NVIC_VectTab_FLASH    ((uint32_t) 0x08000000)
```

在misc.c中,定义了根据中断号到中断向量表中查找中断服务程序的函数NVIC_SetVectorTable(uint32_t NVIC_VectTab, uint32_t Offset)。

2. 中断向量表的使用

在STM32F103中断应用中,中断源(片内外设或外部设备)通过中断通道申请中断,Cortex-M3内核首先判断中断是否触发,根据中断号(中断向量位置)到中断向量表中查到中断服务程序xxx_IRQHandler(void)入口地址,然后执行中断服务程序,中断结束之后返回主程序断点处接着执行。

5.2.6 STM32F103 的中断服务函数

中断服务程序用来执行中断实际发生后的具体处理。STM32F103 的所有中断服务函数都在该微控制器产品系列的启动代码文件 startup_stm32f10x_xx.s 中有预先定义。用户在实际应用中,可根据需求在 stm32f10x_it.c 文件(或其他文件,如 main.c 文件)使用 C 语言重新定义或修改相应中断服务函数(stm32f10x_it.c 文件中已有空的中断函数),替代启动代码中默认的中断服务程序。在编译、链接生成可执行代码时,会用用户定义的同名中断服务函数代替启动代码中预设的中断服务程序。

在编写 STM32F103 中断服务程序时,必须确保定义的中断服务函数和启动文件 startup_stm32f10x_xx.s 中的中断服务程序同名,启动文件中通常以 xxx_IRQHandler 命名,其中 xxx 为中断对应的外设名。

当中断源被触发,STM32F103 微控制器响应中断请求,硬件会自动跳到固定地址的硬件中断向量表,无须编程就能读取对应地址中的中断服务程序地址(xxx_IRQHandler 程序入口地址),放到程序计数器(PC),CPU 转去执行中断服务程序,这就是 STM32F103 的中断硬件机制。

5.3 STM32F103 的外部中断/事件控制器

5.3.1 外部中断/事件控制器的硬件结构

STM32F103 的外部中断/事件控制器(EXTI)内部结构如图 5-2 所示,由 19 根外部输入线、19 个产生事件/中断请求的边沿检测器、外设接口和 APB 总线等组成。每个输入线可以独立地配置输入类型(脉冲或挂起)和对应的触发事件(上升沿或下降沿或双边沿都触发)。每个输入线都可以被独立地屏蔽,请求挂起寄存器保持着状态线的中断要求。

1. EXTI 输入线

在图 5-2 中,内部信号线上画有斜线并标有数字 19,表示这样的线有 19 条。与此对应,EXTI 输入线也有 19 根,分别为 EXTI0,EXTI1,…,EXTI18,其中 16 根外部输入线 EXTI0～EXTI15,分别对应 STM32F103 微控制器的 GPIOx_Pin0～GPIOx_Pin15(Px_Pin0～Px_Pin15),x 为 A、B、C、D、E、F、G,EXTI16 连接 PVD 输出(表 5-2 中第 1 号中断),EXTI17 连接到 RTC 闹钟事件(表 5-2 中第 41 号中断),EXTI18 连接到 USB 唤醒事件(表 5-2 中第 42 号中断)。

STM32F103 微控制器最多有 112 个引脚,分布在 PA～PG 端口 Pin15～Pin0,其中 PA0～PG0 引脚可以映射到 EXTI0 输入线上。PA1～PG1 引脚可以映射到 EXTI1 输入线上,以此类推,PA15～PG15 映射到 EXTI15 输入线,如图 5-3 所示。同一时刻,只能有一个端口(PA～PG)的 x 号引脚映射到 EXTIx 输入线上,x=0～15。

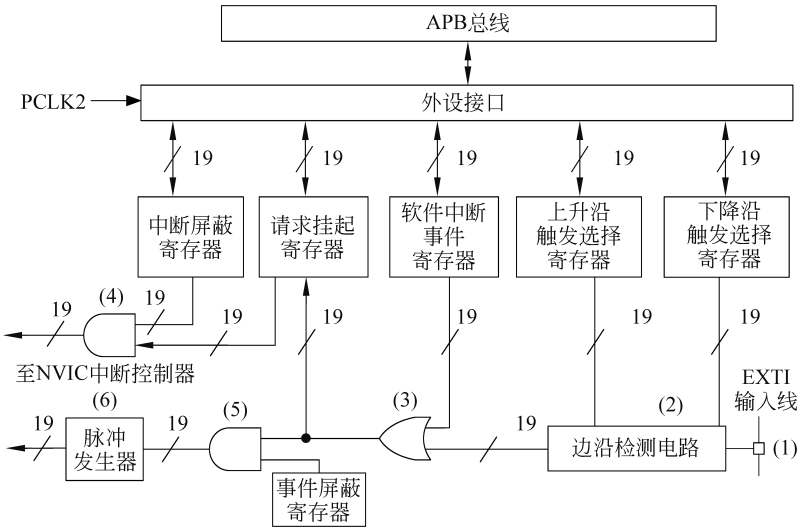

图 5-2 STM32F103 EXTI 的内部结构

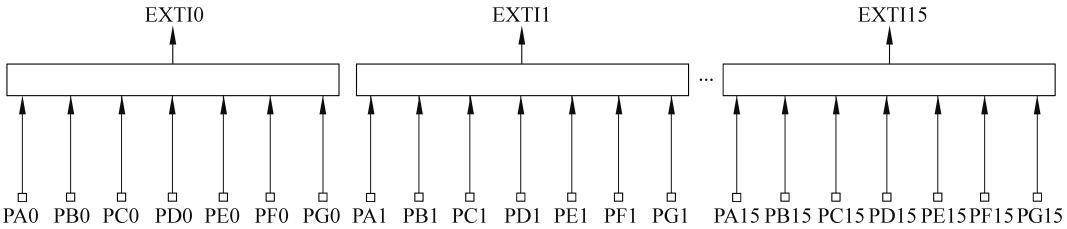

图 5-3 STM32F103 端口引脚到 EXTI 输入线的映射

编程时注意，将 PAx～PGx 端口的某个引脚映射为 EXTI 输入线时，需要将该引脚设置为输入模式。

2. 外设接口和 APB 总线

APB 总线和外设接口是每一个功能模块都有的部分，CPU 通过这样的接口访问各个功能模块。

编程时注意，将 PAx～PGx 端口的引脚映射为 EXTI 输入线时，需要同时打开 APB2 总线上该引脚对应端口时钟以及 AFIO 功能的时钟。

3. 边沿检测器

EXTI 有 19 个边沿检测器，用来连接 EXTI 输入线，产生至 NVIC 向 CPU 的中断请求以及事件请求。边沿检测器由边沿检测电路、控制寄存器、脉冲发生器和门电路等组成。

4. 工作原理

在图 5-2 中，外部信号从 EXTI 输入线进入，进入边沿检测电路，通过或门后，一路进

入请求挂起寄存器，最后通过与门输出到 NVIC 中断控制器。另一路会向其他功能模块（如定时器、USART 等）发送脉冲信号，外部中断/事件请求的产生和传输过程如下：

（1）外部信号经过边沿检测电路，受到上升沿触发选择寄存器和下降沿触发选择寄存器控制。用户可以选择上升沿、下降沿或者同时选择上升沿和下降沿。

（2）经过或门(3)输出。或门的另一个输入是通过软件在“软件中断事件寄存器”设置的中断或事件请求，这说明软件可以优先于外部信号产生一个中断或事件，将“软件中断事件寄存器”的对应位置 1，无论外部信号如何，使得编号(3)的或门输出有效信号。

（3）进入请求挂起寄存器，在请求挂起寄存器中记录外部信号的电平变化。

（4）外部请求信号与中断屏蔽寄存器的对应位相与，如果中断屏蔽寄存器的对应位为 1，则向 NVIC 中断控制器发出一个中断请求，当为 0 时，则该中断请求信号被屏蔽。以上是外部中断请求信号传输路径。

（5）如果用户希望产生“事件”，需要先配置对事件线的使能操作，并设置两个触发寄存器以完成边沿检测，同时对事件屏蔽寄存器的相应位写“1”以允许事件请求。当事件线上发生了对应的边沿信号时，经过边沿触发电路、或门(3)、与门(5)和脉冲发生器，系统将产生一个事件请求脉冲，对应的挂起位并不会被置“1”。以上是外部事件请求信号的传输路径。

5. 外部中断/事件的寄存器配置

1）硬件中断的配置

配置一个或多个 GPIO 引脚作为中断源，操作如下：

（1）在中断屏蔽寄存器(EXTI_IMR)中配置对应中断引脚的屏蔽位。

（2）在上升沿触发选择寄存器(EXTI_RTSR)和下降沿触发选择寄存器(EXTI_FTSR)中配置对应中断引脚的触发选择位。

（3）配置对应的 NVIC 中断通道的使能和屏蔽位。

2）硬件事件的配置

对于硬件事件参数的配置如下：

（1）在事件屏蔽寄存器(EXTI_EMR)配置对应事件线的屏蔽位。

（2）在上升沿触发选择寄存器(EXTI_RTSR)和下降沿触发选择寄存器(EXTI_FTSR)中配置事件线的触发选择。

3）软件中断/事件的配置

对于软件中断/事件处理的配置如下：

（1）在事件屏蔽寄存器(EXTI_EMR)或中断屏蔽寄存器(EXTI_IMR)配置对应中断/事件线的屏蔽位。

（2）在软件中断事件寄存器(EXTI_SWIER)配置对应的请求位。

5.3.2 EXTI 的寄存器

在使用 STM32 处理器的外部中断/事件、软件中断/事件时，必须按 32 位字的方式对相关 EXTI 寄存器相应位进行配置。