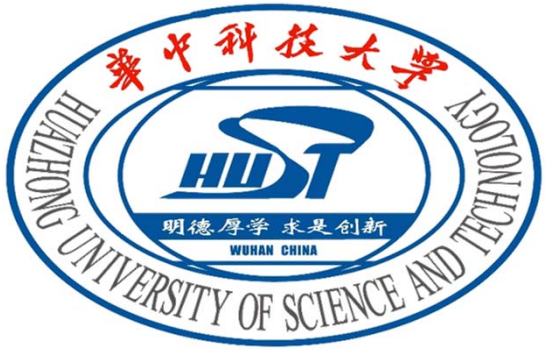
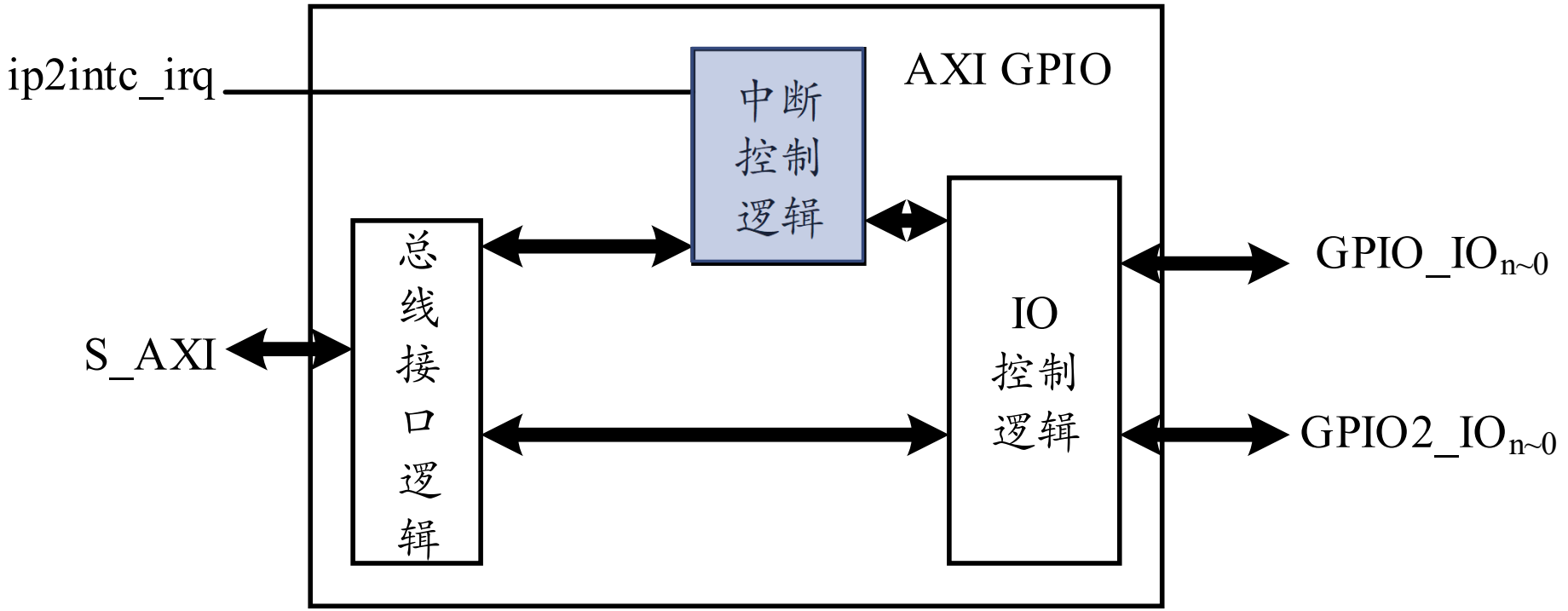
微机原理与接口技术

GPIO中断应用示例

华中科技大学　左冬红



GPIO结构框图



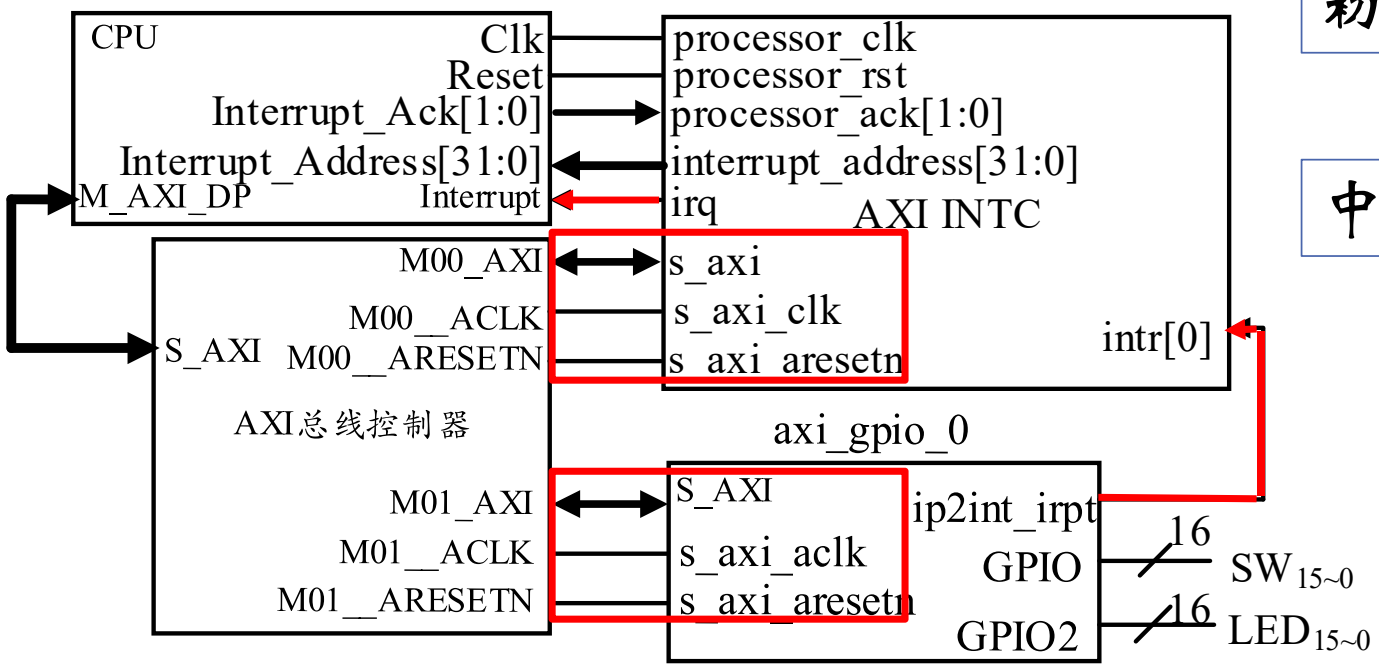
GPIOx_IO引脚输入时出现信号电平跳变，则可以产生中断

两个通道都可以产生中断，最后形成一个中断输出，类似INTC

GPIO中断相关寄存器

名称	偏移地址	含义
GIER	0x11c	D_{31}=1使能中断信号ip2intc_irpt输出
IPIER	0x128	D_0=1使能通道GPIO中断；D_1=1使能通道GPIO2中断，
IPISR	0x120	读：获取通道中断状态，写：清除中断状态 读：D_0=1GPIO产生了中断；D_1=1GPIO2产生了中断 写：D_0=1清除GPIO中断状态；D_1=1清除GPIO2中断状态

应用示例1



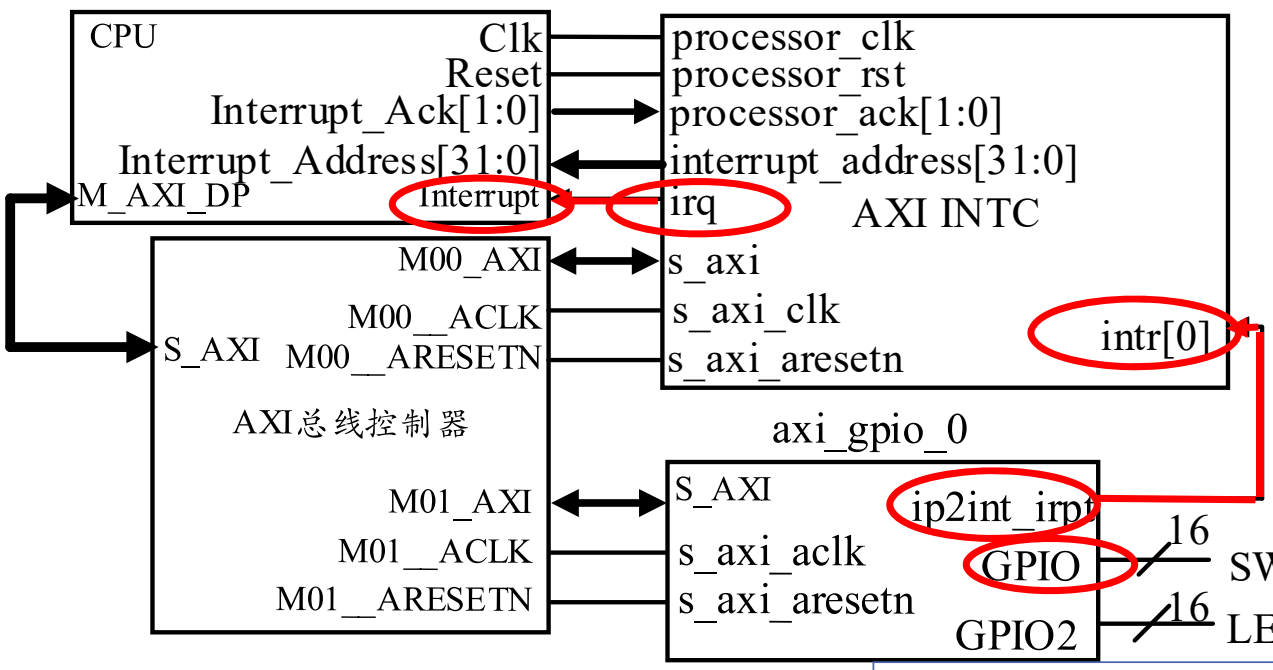
初始化程序

中断服务程序

GPIO 开关拨动产生中断

中断方式读取开关状态并输出到LED

应用示例1



开放中断

- gpio_0 GPIO通道
  - IPIER
  - GIER
- intc intr[0]
  - IER
  - MER
- MicroBlaze
  - MSR

初始化程序

填中断向量表

MB-GCC __attribute__((interrupt_handler))

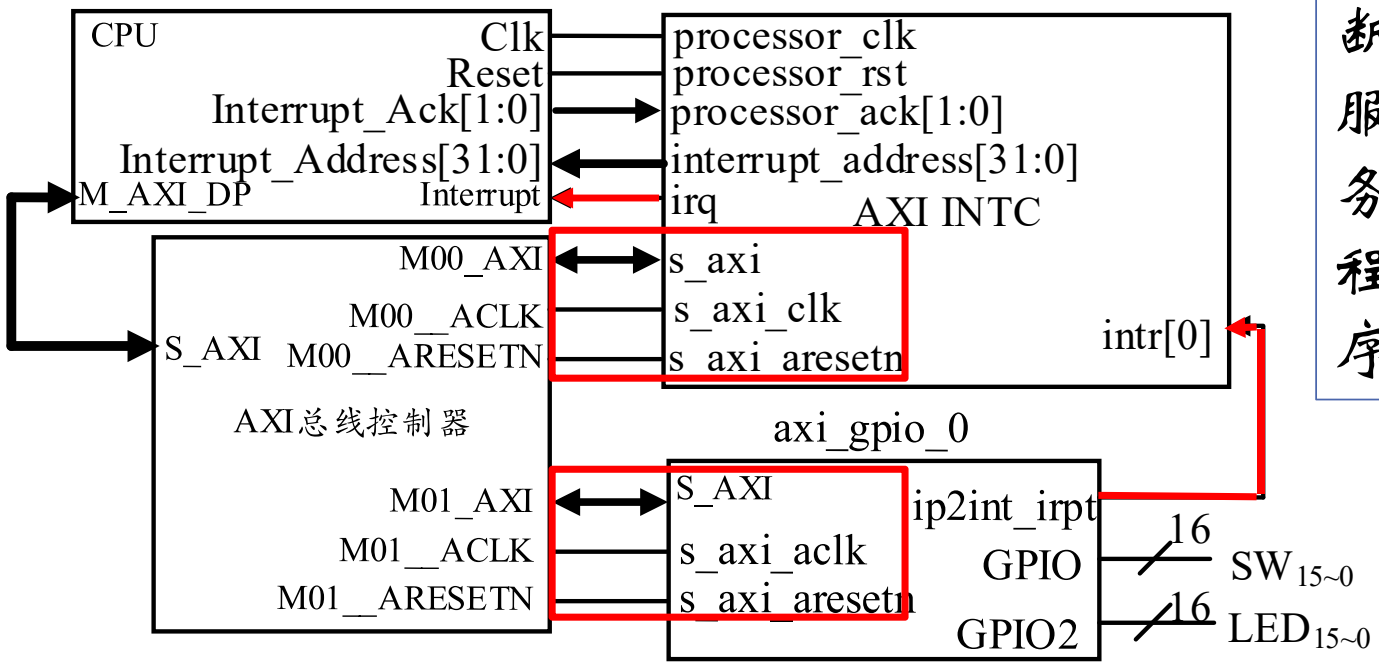
API microblaze_register_handler(,)

IO命令

GPIO工作方式

两者任选其一

应用示例1



中断服务程序

- 中断事务处理
  - 读一次开关
  - 写一次LED
- 清除中断状态
  - INTC intr[0]
    - IAR
  - Gpio_0 GPIO
    - IPISR

中断方式读取开关状态并输出到LED

应用示例1-初始化程序结构

开放中断

- gpio_0 GPIO通道 IPIER GIER
- intc intr[0] IER MER
- MicroBlaze MSR

Xil_Out32(addr,value)

填中断向量表

microblaze_enable_interrupts();

MB-GCC __attribute__(interrupt_handler)

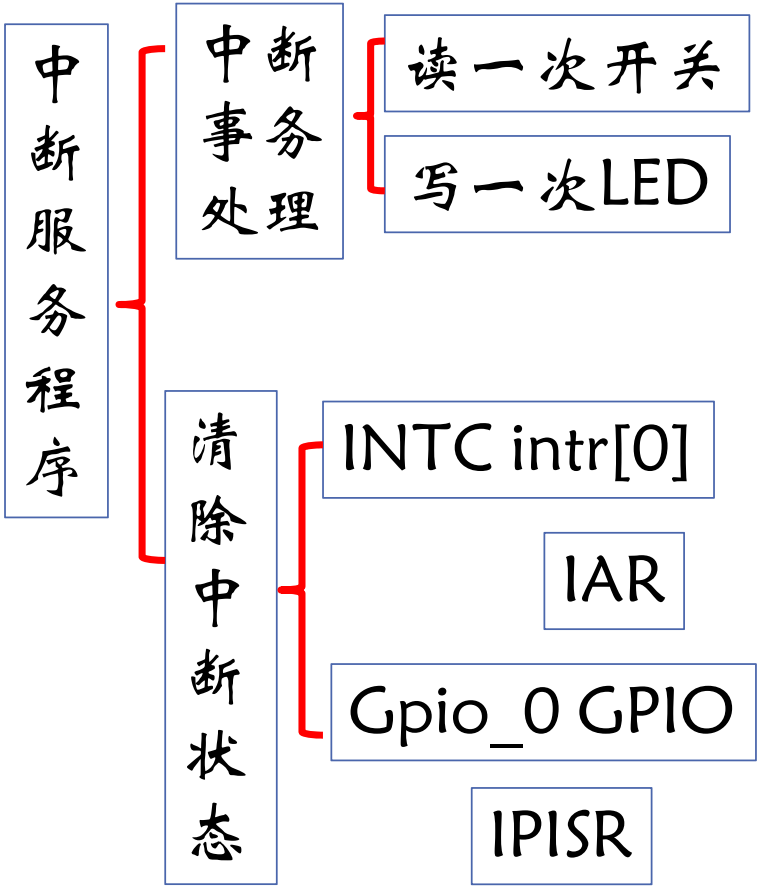
IO命令

GPIO工作方式

应用示例1-初始化程序

```
void main(void)
{
    Xil_Out32(0x40000120,0x1);//写AXI_GPIO_0 IPISR 清除中断
    Xil_Out32(0x40000128,0x1);//写AXI_GPIO_0 IPIER
    Xil_Out32(0x4000011C,0x80000000);//写AXI_GPIO_0 GIER
    Xil_Out32(0x4120000c,0xffffffff);//写INTC IAR清除中断状态
    Xil_Out32(0x41200008,0x1);//写INTC IER,使能intr[0]中断
    Xil_Out32(0x4120001c,0x3);//写INTC MER,使能硬件中断irq输出
    microblaze_enable_interrupts();//使能微处理器中断
    Xil_Out32(0x40000004,0xffff);//写GPIO_TRI通道GPIO输入
    Xil_Out32(0x4000000C,0x0); //写GPIO2_TRI通道GPIO2输出
}
```

应用示例1-中断服务程序



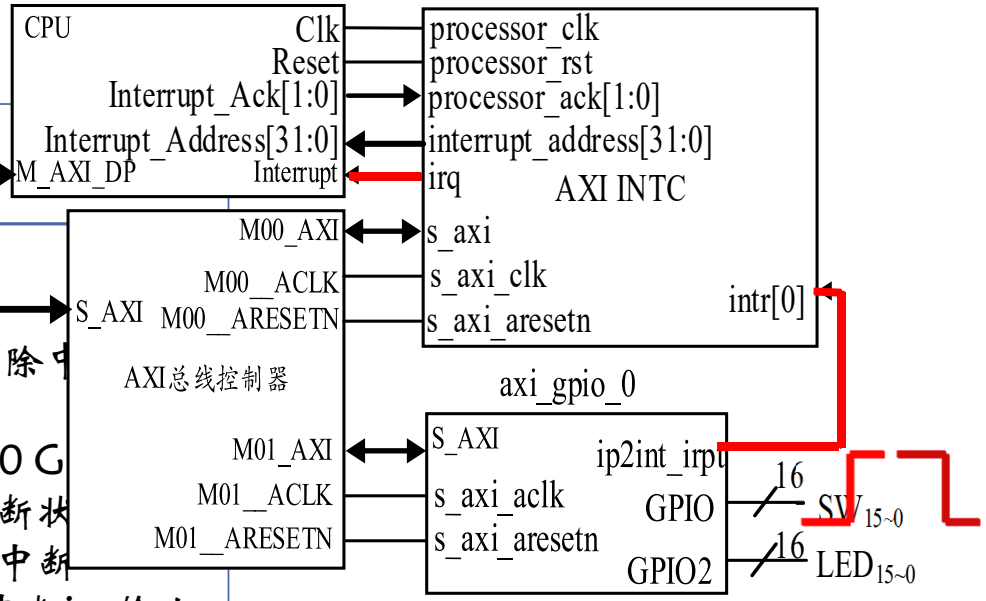

```
void swHandler(void) __attribute__((interrupt_handler));
```

```
void swHandler(void)
{
    int sw;
    sw=Xil_In32(0x40000000);
    Xil_Out32(0x40000008,sw)
    Xil_Out32(0x40000120,0x1);
    Xil_Out32(0x4120000c,0x1);
}
```

应用示例1

```
#include “xil_io.h”
void swHandler(void) __attribute__((interrupt_handler);

void main(void)
{
    Xil_Out32(0x40000120,0x1);//写AXI_GPIO_0 IPISR 清除中
    Xil_Out32(0x40000128,0x1);//写AXI_GPIO_0 IPIER
    Xil_Out32(0x4000011C,0x80000000);//写AXI_GPIO_0 G
    Xil_Out32(0x4120000c,0xffffffff);//写INTC IAR清除中断状
    Xil_Out32(0x41200008,0x1);//写INTC IER,使能intr[0]中断
    Xil_Out32(0x4120001c,0x3);//写INTC MER,使能硬件中断irq输出
    microblaze_enable_interrupts();//使能微处
    Xil_Out32(0x40000004,0xffff);//写GPIO_
    Xil_Out32(0x4000000C,0x0); //写GPIO2
}
```




```
void swHandler(void)
{
    int sw;
    sw=Xil_In32(0x40000000);
    Xil_Out32(0x40000008,sw)
    Xil_Out32(0x40000120,0x1);//写GPIO ISR
    Xil_Out32(0x4120000c,0x1);//写INTC IAR
}
```

小结

- GPIO产生中断的原因
  - GPIO输入引脚电平跳变
- 中断程序设计
  - 初始化程序
    - GPIO中断开放
    - GPIO工作方式设置
    - INTC开中断
    - MicroBlaze开中断
    - 填写中断向量表
  - 中断服务程序
    - 中断事务处理
    - 清除中断
      - GPIO\INTC都需清除

下一讲：定时器中断