中断：中止cpu当前执行，进行中断处理的过程

中断可以调高cpu的效率

中断处理流程：

1. 保存现场
2. 寻找中断入口
3. 中断处理
4. 返回断点

程序完成中断：

1. 中断发生
2. 中断相应
3. 中断处理

soc中断信号：

soc厂商设计的特殊电信号，表示硬件电路状态改变

外部中断：电信号来自soc外接电路，需要通过soc指定引脚接收，电信号可选（高电平/低电平/上升沿/下降沿/双边沿），触发时机不确定

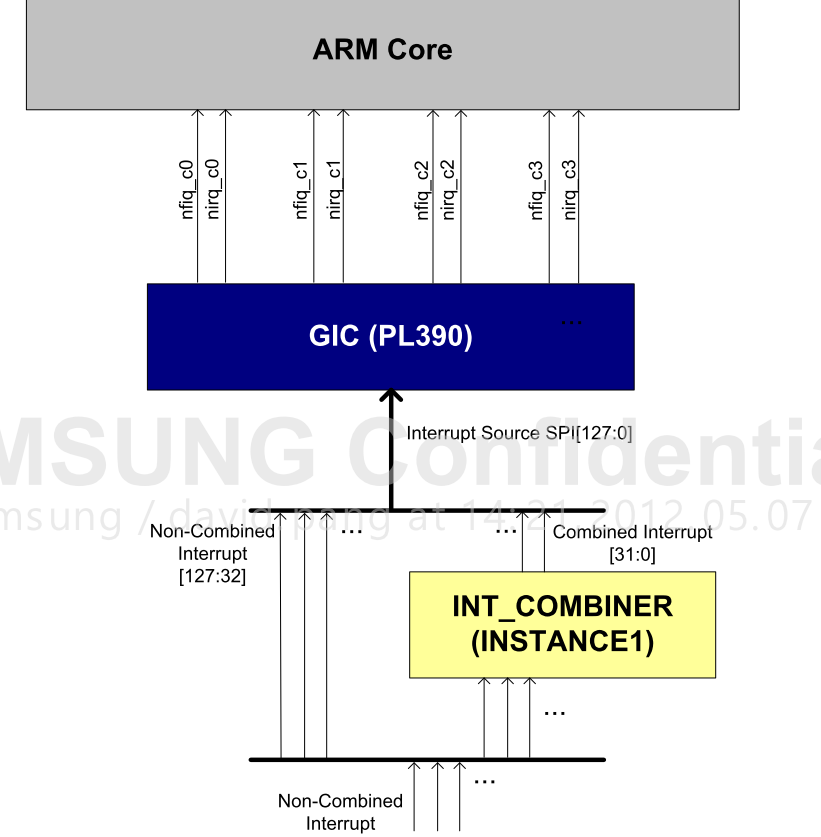
内部中断：电信号来自soc内部电路，电信号不可修改，触发时机可预测

4412中断：

4412提供200多个中断电信号，

中断控制器：负责中断信号接收/区分/传递

arm设计的GIC中断控制器（PL390）pl390设计了160个中断接口（160个中断号），来进行中断信号的接收和区分，为应对4核处理器，设计4个cpu控制器接口，每一个编号的中断都有一些列寄存器单独控制。



需求：

1. 电路使能 ICDDCR gic使能 1使能 0 禁止
2. 如何区分 ICCIAR\_CPUn cpu会自动将完成传递的中断编号写入ICCIAR\_CPUn的低10位

ICCEOIR\_CPUn cpu会将写入编号中断从激活队列清除

1. cpu控制接口选择 ICCICR\_CPUn 1使能 0禁止 cpu控制接口选择

ICCPMR\_CPUn 优先级mask

ICDISER\_CPU 具体中断使能； 每一个cpu控制接口，对160个编号中断，都有一个控制使能位

1. 目标处理器选择

ICDIPTR\_CPU 每一个cpu控制接口，对160个编号中断，都有一个8位域设置目标处理器 发送给那个处理器则需要将这个域的哪一位置1

1. 优先级如何设置

ICDIPR\_CPU 每一个cpu控制接口，对160个编号中断，都有一个8位域设置优先级

reset 0x0

udef 0x4

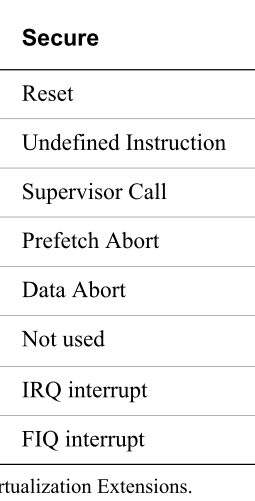
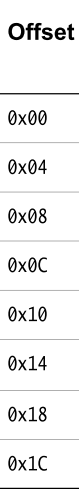
svc（软件中断） 0x8

pre\_abot 0xc

data\_abort 0x10

irq 0x18

fiq 0x1c

异常处理机制：异常发生时cpu自动跳转到异常向量表基地址+固定偏移量的位置执行代码

svc #12（swi #12） //软件中断指令，当cpu执行该指令后，自动触发软件中断

异常发生时cpu自动保存返回地址到lr寄存器，保存一场发生时cpu正在执行指令的下一条指令地址

armv7支持将异常向量表基地址指定在内存任意位置

使用cp15协处理器的c12寄存器存储异常向量表基地址

内嵌汇编：在c文件中增加一段汇编程序。

int num;

int num1 = 100;

\_\_asm\_\_(

指令域

“mov r0, #1\n” 所有指令必须被””包裹， 必须以’\n’结尾

“mov r2, #2\n”

“mov %0, r2\n” //num = r2 %0表示第一个变量

:输出变量声明域， 完成汇编传递数据到c的变量 “=&r”(num) =表示变量在汇编内是只写 &变量分配寄存器是不冲突的 r编译工具会给变量分配一个寄存器 “+&r” +表示变量在汇编内可读写

:输入变量声明域 完成c变量传递给汇编 “r”(num1) 可以在汇编中使用 %1来调用

可以定义别名来完成调用[num1]”r”(num1) 在汇编中%[num1]表示对num1变量引用

: 改变声明域；本段汇编修改的寄存器需要声明 “r0”, “r2”

);

中断处理：

1. 信号发生
2. 信号传递（接收/区分/传递）---》cpu （按照异常处理机制完成函数调用）
3. 中断处理函数设计

mask 控制中断信号传递

pending （触发标志位） 大部分手动清除

pwm定时器：

工作原理：电路按照固定频率对一个数值进行减法操作，当减为0时触发中断信号

1. 固定频率
2. 减的数值设置
3. 更新（实时/自动）
4. mask使能 TINT\_CSTAT 第0位 1使能 0禁止
5. pending 位 TINT\_CSTAT 第5位 写1 清除（中断处理后）

设置cp15协处理器的c12寄存器，完成异常向量表基地址指定

mcr p15, 0, r0, c12, c0, 0 @将r0的数值赋给cp15协处理器的c12寄存器，设置异常向量表基地值在r0

中断代码实现：

1.