I n terrupt

1. 中断的作用：

异常　硬件或软件错误

I/O　响应外部事件，进行人机交互

并发　ＣＰＵ与Ｉ/O并发，多任务并发

服务　提供系统服务和保护的机制

２．时钟中断 维持系统时间，多任务分时共享CPU

３．异常处理模式　将控制权交给OS的异常处理程序　可恢复中断和不可恢复中断

4．x86

外部中断:硬中断　• 可屏蔽中断,不可屏蔽中断NMI

内部中断:软中断　• 程序异常,系统调用 INT n,指令断点(int 3调试)

RISC

外部事件(interrupts):I/O中断

异常Exceptions:软(硬)件故障

陷阱Traps:　syscall，断点break，自陷TEQ

5．中断发生与CPU响应的时机与条件

为了保证程序执行的顺序性, 没有执行完的指令必须记录它们执行到哪一个阶段

中断发生与CPU响应时间

　外部中断:异步(延迟响应) 指令周期结束(顺序执行模型!)　内部中断:同步(“马上”响应)

响应中断的条件

　外部中断　可禁止(PSW的中断允许位),可屏蔽(中断屏蔽寄存器)　内部中断:不可禁止

６．中断周期

• CPU与中断控制器通信,响应外部事件　– 有中断请求?识别中断源,获得中断向量

• 动作:存PC和PSW,关中断,识别,转ISR

７．中断机构组成

　CPU中断禁止/允许控制:IF@PSW

　CPU中断请求/响应控制:INTR、INTA

　中断响应/返回:中断隐指令

　断点/现场保存:MEM(stack)

　中断服务　– 中断源识别/判优:中断控制器　– ISR入口:向量方式、非向量方式

８．MIPS中的异常

External events中断或读总线错等

Memory translation exceptions缺页或越界等

Other unusual program conditions for the kernel to fix须内核处理的不常见程序状态

Program or hardware-detected errors非法指令、溢出、对齐等

Data integrity problems(Checksum etc.)

９．MIPS的异常(软中断)处理

• 设计控制部件的难点在于异常处理　检查异常和采取相关的动作通常在关键路径上进行

• 讨论两种异常:非法指令和算术溢出

• 异常处理

– 断点保存:将受干扰的指令的地址保存在EPC寄存器中

– 异常识别:根据状态寄存器cause中的异常原因分别处理异常

• 非法指令:为用户程序提供某些服务

• 溢出:对溢出进行响应

– 异常服务程序:停止异常程序的执行并报告错误等　异常处理程序地址在c00000000H

１０．非流水线的中断和异常语义

• 顺序语义模型

– 之前的指令都已执行完成　已经提交其状态

– 之后的指令还没有启动　没有改变任何机器状态

• 断点精确:= PC/nPC　– 中断:异步响应　– 指令异常:同步响应，包括软中断

• 现场简明

１１．流水线中的异常与中断

• 多个流水段,多条指令,多种异常并发　– 异常:访存、译码、溢出　– 中断

１２．流水线异常处理的挑战

顺序执行”只是一种抽象:断点和状态难以确定

– 转移预测失误可能取消其后出现了异常的指令

– 后续指令在产生异常指令完成之前改变了系统的部分状态

如 ld 指令在MEM段产生异常,而其后的R-type指令设置了0标志

– 异常发生顺序与指令执行顺序不一定相同

１３．非精确异常　允许已进入流水线中的指令执行完再转去执行中断处理

• 无论在第i条指令的哪一流水段上发生异常,都不再允许后继指令进入流水线

– 断点:最后进入流水线的那条指令的地址

• 非精确(不是“当前指令”),且可变(不同段发生异常,EPC增量不同)

• 优点:硬件比较简单

１４．ＤＬＸ实现非精确　将异常视为一种控制依赖

暂停指令流中导致异常的指令，执行完之前的所有指令，清除之后的所有指令，记录异常原因，保存最后进入流水线的指令的地址（断点），转异常处理程序。

不允许后续指令继续执行（与”非精确不同“）

１５．非精确异常的问题

异常响应时间较长(抖动):不同段异常时　可能会导致程序出错: 异常

– 程序调试不便:程序员在第i条指令设置断点，但程序不能准确中断在所设置的断点处。

• 多个异常?

１６．MIPS实现精确异常

• 实现“精确”开销大　要保证异常指令“可重启”，安全停止流水线,并完整保存当前状态，需要大量后援寄存器保存流水线中各指令的现场，包括RegFile和流水段寄存器(含各段的控制寄存器)!

• MIPS采用提交点技术实现精确异常,简化了设计

– 提交点:M段 (可否其他段?)　多个异常:先发生的异常并不立即处理,只是被标记

EXCn寄存器:保存异常类型　流水线中最深的指令引起的异常最优先

• 中断:在M段检测(符合“异步”语义)

– 断点:EPC = PC or nPC?　中断断点 = MEM段的当前指令(返回后要重新执行!)

１７．各段产生的异常及处理:MIPS的策略

保持流水线的异常标记直到提交点，早期流水段的异常抑制后来的异常，提交点引入外部异常（抑制其他异常），如果提交点有异常，则更新cause和ＥＰＣ，清除所有流水段，恢复PC恶到fetch段。

１８．中断系统

中断的概念

暂停当前程序的执行,转而执行其他程序,在它们执行完成后再恢复被中断的程序执行。

中断源:外部中断、内部中断(软中断、异常)

中断源识别：查询法，向量法。　中断判优　中断服务程序（ＩＳＲ)：入口地址

中断到达与响应时机

中断响应

断点：nPC，系统关键状态，隐指令完成　　现场：regs，PSW，中断服务相关

中断返回（清中断）　　中断周期

中断嵌套

中断机构　为了实现中断,计算机系统中必须有相应的中断系统。

中断响应时机:程序的当前状态

断点、现场?

指令周期

基于总线周期的计算机?流水线实际上是总线周期计算机

• 同步/异步:到达/发生时刻与指令周期的关系

– 中断:异步,在中断周期响应;返回下一条指令　• 何时不允许中断?

– 异常:同步,无中断周期;返回当前指令或无法返回

– 陷阱:同步,无中断周期;返回下一条指令

– 过程调用:同步,无中断周期;返回下一条指令

• 中断/异常/陷阱改变系统状态,Call不改变系统状态

１１．中断的产生　整机效率　突发事件　实时控制

１２．中断I/O用途　CPU与I/O设备并行工作　硬件故障处理:故障-->中断-->自动恢复

人机通信　多任务切换　实时处理　多处理机通信

１３．中断请求触发器(INTR):用于保存各中断源的请求。每个中断源一位。可以在CPU内部,也可以在外部

1	2	3	4	5			n
掉电	过热	内存读写校验错	阶上溢	非法除法		光电输入机	打印输出机

中断请求标记寄存器

１４．中断服务程序入口地址。不同的中断(类型)有各自的服务程序。

硬件向量法(中断向量法)中断向量:设备接口向CPU提供的自身标识，CPU根据中断向量检索中断向量表,得到中断服务程序的入口地址

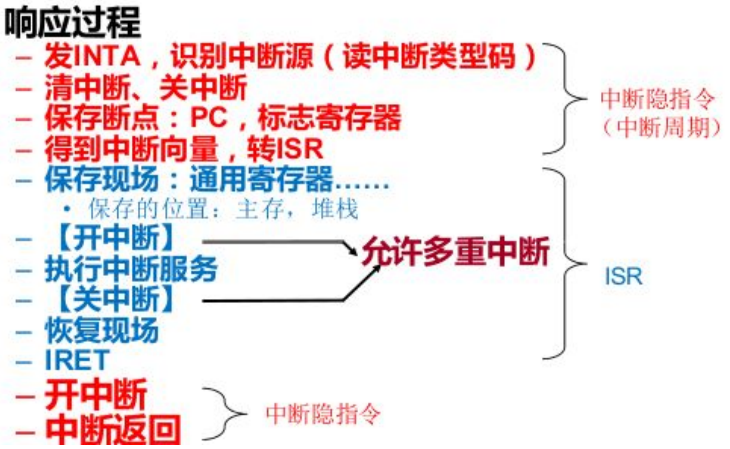
软件查询法:用软件寻找中断服务程序入口地址的方法。由程序员(或系统)事先确定中断源对应的入口地址,不涉及硬件设备,但查询时间比较长。

１５．中断隐指令:在机器指令系统中没有的指令,它是CPU在中断周期内由硬件自动完成的一条指令。主要功能:中断响应。

• 保护程序断点:将当前程序计数器PC的内容(程序段点)保存到存储器中,或者存在存储器的特定单元(如0号地址),或者入栈。

• 寻找中断服务程序的入口地址

• 关中断:确保CPU响应中断后的所需作的一序列操作不至于有受到新的中断请求的干扰



１６．如果中断频繁,则效率低，高速、批量数据传输不适用，某些任务必须使用(如调度)，ISR必须短小