

Laboratorio di Programmazione di Sistema

Eccezioni

Luca Forlizzi, Ph.D.

Versione 23.1



Luca Forlizzi, 2023

© 2023 by Luca Forlizzi. This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit <https://creativecommons.org/licenses/by-nc-nd/4.0/legalcode>.

Concetto di Eccezione

- In generale un *ASM-PM* può prevedere diverse situazioni anomale
- Alcuni esempi
 - Overflow
 - Operazioni di accesso a indirizzi di memoria non validi
- Una situazione anomala, viene spesso gestita tramite una procedura di emergenza
- Nella maggior parte degli *ASM-PM* moderni tale procedura è software, ovvero è una sequenza di istruzioni che si occupa di gestire la situazione
- Ciò rende programmabile la procedura di gestione delle anomalie

Concetto di Eccezione

- La procedura di gestione delle anomalie non appartiene al programma in esecuzione
- Quando una abstract machine rileva un'anomalia, interrompe la sequenza di istruzioni del programma e comincia ad eseguire la procedura di gestione delle anomalie
- La procedura di gestione delle anomalie può fare ciò che ritiene opportuno, compreso riprendere esecuzione del programma nel quale l'anomalia si è verificata

Concetto di Eccezione

- Il modo di operare appena descritto è un caso particolare di un meccanismo più generale, noto come *gestione delle eccezioni*
- Con il termine *eccezione* si intende un evento anomalo che può accadere durante l'esecuzione di un programma, che richiede, per essere gestito, l'esecuzione di una procedura apposita, chiamata *exception handler* (in italiano *gestore di eccezione*)
- Sebbene tra diversi *ASM-PM* vi siano notevoli differenze in merito alle eccezioni, si possono alcune tipologie di eccezione piuttosto comuni
 - Errori hardware (comunicazione con la memoria o altri dispositivi)
 - Errori software (overflow, divisione per 0, accesso a indirizzo non valido, codifica di istruzione non valida)
 - Eccezioni software, dette anche *trap*
 - *Interrupt*, ovvero eccezioni causate da segnali provenienti dall'esterno della abstract machine

Gestione delle Eccezioni

- Quando si verifica un'eccezione, una abstract machine svolge una serie di azioni detta *processo di gestione delle eccezioni*
- I processi di gestione delle eccezioni definiti dalla maggior parte degli *ASM-PM*, sono simili tra loro ma differiscono in molti dettagli
- Spesso i dettagli differiscono anche tra *ASM-PM* della stessa famiglia
- Tipicamente, per eseguire i processi di gestione delle eccezioni, le abstract machine entrano in un modo di funzionamento di tipo kernel, e di conseguenza hanno accesso a tutte le risorse del sistema

Gestione delle Eccezioni

- Concettualmente, una abstract machine M controlla se si è verificata un'eccezione subito prima di iniziare l'esecuzione di ciascuna istruzione
- Se si è verificata un'eccezione, allora M invece di eseguire l'istruzione il cui indirizzo è contenuto nel PC, attiva un gestore di eccezione
 - ① M entra in modo di funzionamento kernel
 - ② Il contenuto del PC e altre informazioni sullo stato di M vengono memorizzate, in modo da poterle usare, al termine del processo di gestione dell'eccezione, per riprendere l'esecuzione del programma attualmente in corso senza alterarne il funzionamento
 - ③ Se M prevede che le eccezioni abbiano una priorità, viene impostata la priorità delle eccezioni che possono interrompere il gestore di eccezione dell'eccezione attuale
 - ④ Viene avviata l'esecuzione del gestore di eccezione corrispondente al tipo dell'eccezione che si è verificata

Gestione delle Eccezioni

- Un gestore di eccezione è eseguito in modo kernel e quindi può accedere a tutte le risorse di M
- In particolare può esaminare
 - Il tipo di eccezione che si è verificata
 - L'istruzione che stava per essere eseguita quando si è verificata l'eccezione
 - L'intero contenuto della memoria (non solo quella allocata al programma in esecuzione)
- Un gestore di eccezione, deve intraprendere le azioni più opportune per rispondere all'emergenza
 - Correggere errori software
 - Raccogliere informazioni su problemi hardware
 - Rispondere a richieste di intervento

Gestione delle Eccezioni

- Una volta eseguito il suo compito, il gestore può tornare ad eseguire il programma interrotto dall'eccezione, oppure terminarne l'esecuzione
- Nel caso scelga di riprendere l'esecuzione, ciò deve essere fatto ripristinando, almeno in parte, lo stato della abstract machine al momento in cui l'eccezione si è verificata
- Una parte importante dello stato della abstract machine è costituita dai registri
- Tale azione di ripristino viene fatta, tipicamente, attraverso istruzioni apposite che leggono le informazioni salvate durante l'avvio del gestore di eccezione

Gestione delle Eccezioni

- Per attivare un gestore di eccezione, la abstract machine deve conoscerne l'indirizzo
 - In alcuni *ASM-PM*, tra cui MIPS32, i gestori di eccezione hanno indirizzi fissi
 - In altri, tra cui MC68000, gli indirizzi dei gestori di eccezione sono memorizzati in parole di memoria, le quali hanno indirizzi fissi
 - In altri, tra cui versioni successive di MIPS e di M68000, gli indirizzi dei gestori di eccezione sono memorizzati in parole di memoria, i cui indirizzi vengono calcolati usando i valori di uno o più registri specific purpose della abstract machine
- I metodi più usati per registrare quale eccezione si è verificata, tra quelle possibili, sono
 - Registrazione mediante *codice identificativo*
 - *Eccezioni vettorizzate*

Gestione delle Eccezioni

- Per effettuare la registrazione mediante *codice identificativo*, a ogni tipo di eccezione viene attribuito un codice
- Quando la abstract machine rileva un'eccezione, ne memorizza il codice del tipo in un registro o in una particolare parola di memoria nota al gestore di eccezione
- Con questo metodo è possibile utilizzare un unico gestore per tutti i tipi di eccezione
- Il gestore utilizza il codice identificativo per decidere in che modo gestire l'eccezione che si è verificata

Gestione delle Eccezioni

- Il metodo delle *eccezioni vettorizzate* prevede che ad ogni tipo di eccezione venga dedicata una parola di memoria che contiene l'indirizzo della procedura di gestione per tale tipo di eccezioni
- Quando la abstract machine rileva un'eccezione sceglie il gestore da attivare in base al tipo di eccezione verificatasi
- La causa dell'eccezione viene comunicata implicitamente
- Normalmente, quindi, ogni tipo di eccezione ha un gestore dedicato

Eccezioni in MIPS32-MARS

- Vi sono differenze nella gestione delle eccezioni tra le varie architetture MIPS
- MARS simula solo gli elementi base del processo di gestione delle eccezioni di MIPS32
- Ne risulta un processo semplificato adatto alla didattica, che descriviamo nel seguito
- Per ulteriori dettagli sulla gestione delle eccezioni, rilevanti nelle implementazioni reali di MIPS32, si rimanda a **[MIPS32]**, la documentazione tecnica dell'architettura

Eccezioni in MIPS32-MARS

- La sezione di una abstract machine MIPS32 che si occupa, tra le altre cose, della gestione delle eccezioni, prende il nome di *coprocessore 0*
- Il coprocessore 0 ha 32 registri di controllo, i cui valori determinano il funzionamento di molte caratteristiche della abstract machine
- L'accesso ai registri del coprocessore 0 avviene attraverso due istruzioni
 - `mfc0 rt,rd` copia il registro `rd` del coprocessore 0, nel registro generale `rt`
 - `mtc0 rd,rt` copia il registro generale `rt` nel registro `rd` del coprocessore 0

Eccezioni in MIPS32-MARS

- MIPS32 gestisce 2 diversi interrupt software e 6 interrupt hardware
- MIPS32 ha diverse istruzioni in grado di generare eccezioni (`syscall`, `break`, istruzioni di *trap*)
- I gestori di eccezione, come già detto in precedenza, si trovano ad indirizzi di memoria fissi
- La causa di un'eccezione viene comunicata con un metodo misto: alcune (poche) eccezioni sono vettorizzate, mentre per le rimanenti viene attivato un gestore comune che può discriminare il tipo di eccezione tramite un codice identificativo

Eccezioni in MIPS32-MARS

- MIPS32-MARS implementa solo 4 dei registri del coprocessore 0, sufficienti per simulare le basi del processo di gestione delle eccezioni

Status Controlla l'abilitazione degli interrupt e ha un bit che pone la abstract machine nello stato di esecuzione del gestore di eccezione

Cause Contiene il codice identificativo dell'eccezione e indica quali sono le richieste di interrupt

EPC Contiene una copia del valore che ha il PC al momento in cui si è verificata l'eccezione

BadVAddr Quando si verifica una eccezione a causa di un accesso non valido ad una parola di memoria, l'indirizzo di tale parola viene memorizzato in BadVAddr

Eccezioni in MIPS32-MARS

- Il processo di gestione delle eccezioni si svolge come segue
 - ① Il contenuto del PC viene salvato in EPC
 - ② Il bit 1 di Status viene posto a 1, il che provoca la disabilitazione degli interrupt e il passaggio della abstract machine in modo kernel
 - ③ Il codice identificativo dell'eccezione viene registrato nei bit da 2 a 6 di Cause
 - ④ La abstract machine inizia ad eseguire istruzioni a partire dall'indirizzo fisso del gestore di eccezione relativo al tipo di eccezione verificatasi
- Per la maggior parte delle eccezioni, viene attivato il gestore che ha indirizzo 0x80000180; tale indirizzo, si trova in un'area di memoria riservata al modo kernel

Eccezioni in MIPS32-MARS

- Il gestore di eccezione, una volta determinato il tipo di eccezione verificatasi, deve intraprendere delle azioni opportune
 - Servire la richiesta di interrupt
 - Correggere l'errore verificatosi
 - Raccogliere informazioni utili al debug
- Al termine delle sue operazioni può, se opportuno, riprendere l'esecuzione del programma interrotto dall'eccezione
- A tale scopo esegue l'istruzione `eret` che azzerà il bit 1 di `Status` e riprende l'esecuzione a partire dall'istruzione il cui indirizzo è memorizzato in `EPC`

Eccezioni in MIPS32-MARS

- I più interessanti tipi di eccezione sono
 - Errori di indirizzo in lettura/scrittura di un dato o di una istruzione da/a un indirizzo non correttamente allineato o non valido (protezione memoria)
 - Errori sul bus in accesso a istruzioni o dati (errori hardware nella comunicazione con la memoria)
 - Eccezioni provocate da istruzioni apposite (`syscall`, `break` o istruzioni *trap*)
 - Eccezioni provocate da overflow
 - L'istruzione in esecuzione è non definita o riservata ad un modo privilegiato
 - Interrupt

Interrupt in MIPS32-MARS

- MIPS32 gestisce 8 diversi interrupt, a ciascuno dei quali è dedicato un bit di richiesta in Cause
- Per effettuare una richiesta di interrupt è necessario porre a 1 il bit di richiesta corrispondente
- Ciascun interrupt può essere abilitato o meno, separatamente dagli altri, tramite un apposito bit in Status
- Status contiene anche un bit che abilita/disabilita in generale tutti gli interrupt

Interrupt in MIPS32-MARS

- Quando viene effettuata una richiesta per un determinato interrupt, se esso è abilitato viene generata un'eccezione con codice identificativo 0
- Per scoprire esattamente quale interrupt si è verificato, tra gli 8 possibili, il gestore di eccezione può leggere i bit di richiesta in Cause
- Alcuni dispositivi esterni azzerano automaticamente la richiesta non appena viene letto Cause
- In altri invece la richiesta rimane attiva (*pendente*) ed è compito del gestore di eccezione azzerare il bit di richiesta: se non lo fa, non appena termina l'eccezione, viene attivato nuovamente in quanto la richiesta risulta ancora pendente, e si crea dunque un ciclo infinito in cui il gestore viene chiamato di continuo