CSSE 232
Computer Architecture I

Exceptions

Class Status

Reading for today

- B.7-8

Outline

- Definition
- Exception registers
- The process
- Special exception instructions
- Work on lab

Definition

What is an exception?

- Unexpected events that interrupt normal program flow
- Exception and interrupt not the same!
    - Exception - event from within the processor
    - Interrupt - event from outside the processor
- Are they necessary?
- What are the disadvantages of exceptions?

What Exceptions have you seen?

Previous programming?

- What did they look like?
- When did they happen?
- How did you solve them?

MIPS and SPIM

- What did they look like?
- When did they happen?
- How did you solve them?

Example

- Exception generated during program execution
  - What should happen?

Example

- Exception generated during program execution
  - What should happen?
  - Should OS continue?

Example

- Exception generated during program execution
    - What should happen?
    - Should OS continue?
    - Should program continue running?
- Possible ways to handle running program?

Example

- Exception generated during program execution
    - What should happen?
    - Should OS continue?
    - Should program continue running?
- Possible ways to handle running program?
    - OS terminates the program
    - Skip offending instruction, continue
    - Retry offending instruction, continue

Example

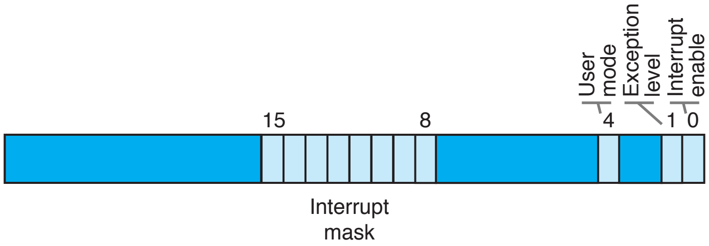
- Exception generated during program execution
  - What should happen?
  - Should OS continue?
  - Should program continue running?
- Possible ways to handle running program?
  - OS terminates the program
  - Skip offending instruction, continue
  - Retry offending instruction, continue
- What does the processor need to continue running the program?

Important Exceptions registers

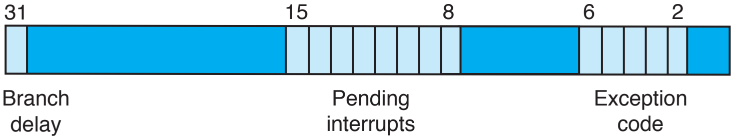
- EPC
  - Stores address of bad instruction
- Status Register
  - Stores the status of the exception
  - Exception being handled, ignore subsequent exceptions
- Cause Register
  - Stores cause of exception

Important Exceptions registers

Status register $12:



Cause register $13:



Process

- In MIPS, exceptions managed by a System Control Coprocessor (CP0) - a special location on the CPU
    - Only helps the main processor start exception handling.
    - Main processor runs the actual handler
- Save PC of offending (or interrupted) instruction
    - In MIPS: Exception Program Counter (EPC)
- Save indication of the problem
    - In MIPS: Cause register
- Jump to handler at 8000 00180
    - Not the only way to do it. Book will talk about vectored interrupts

Special Instructions

- `mfc0`
  - Move from coprocessor 0
- `mtc0`
  - Move to coprocessor 0
- `eret`
  - Return from exception (goes to EPC)

Exception Handler

- Read cause, transfer to relevant handler
- In Handler
    - Determine action required
    - If restartable
        - Take corrective action
        - use EPC to return to program
    - Otherwise
        - Terminate program
        - Report error using EPC, cause, ?

Work on Lab

- Posted on course site