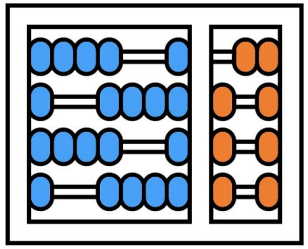

Instituto de Computação

UNIVERSIDADE ESTADUAL DE CAMPINAS

Organização Básica de computadores e linguagem de montagem

Exceções e Interrupções por Software

Prof. Edson Borin

https://www.ic.unicamp.br/~edson

Institute of Computing - UNICAMP

Agenda

- **Modos de privilégio e proteção do sistema**
- Exceções
- Interrupções por SW
- Proteção de sistemas RISC-V

Modos de privilégio e proteção do sistema

- Sistemas computacionais com múltiplas aplicações precisam ser protegidos contra *software* defeituoso ou malicioso.
  - Falha em uma aplicação de usuário não deveria comprometer a segurança ou o funcionamento do restante do sistema!

Modos de privilégio e proteção do sistema

- Sistemas computacionais com múltiplas aplicações precisam ser protegidos contra *software* defeituoso ou malicioso.
  - Falha em uma aplicação de usuário não deveria comprometer a segurança ou o funcionamento do restante do sistema!
- Dois tipos de *software*:
  - ***software* de sistema**: Gerencia o *hardware* (incluindo periféricos) e o compartilhamento de recursos.
  - ***software* de usuário**: Manipula dados na memória e nos registradores e invoca o *software* de sistema para outras atividades, como realizar entrada e saída.

Modos de privilégio e proteção do sistema

is com múltiplas
protegidos contra
malicioso.

e usuário não deveria
ou o funcionamento do

Precisa **ter privilégio** suficiente para interagir com os periféricos, tratar interrupções, carregar e executar programas de usuário, *etc.*
Ex: Sistema operacional.

- ***software* de sistema**: Gerencia o *hardware* (incluindo periféricos) e o compartilhamento de recursos.
- ***software* de usuário**: Manipula dados na memória e nos registradores e invoca o *software* de sistema para outras atividades, como realizar entrada e saída.

Modos de privilégio e proteção do sistema

Precisa **ter privilégio** suficiente para interagir com os periféricos, tratar interrupções, carregar e executar programas de usuário, *etc.*
Ex: Sistema operacional.

Precisam ser **monitorados e controlados** de forma a impedir que falhas no *software* ou código malicioso danifique o restante do sistema.
Ex: Editor de texto.

- ***software* de sistema**: Gere[illegible] *hardware* (incluindo periféricos) e o compartil[illegible]mento de recursos.
- ***software* de usuário**: Manipula dados na memória e nos registradores e invoca o *software* de sistema para outras atividades, como realizar entrada e saída.

Modos de privilégio e proteção do sistema

Estratégia: *Hardware* com dois modos de privilégio: superusuário e usuário.

- **Superusuário** (supervisor, modo privilegiado, …)
  - Neste modo, a CPU não limita o *software* que está sendo executado - O *software* tem privilégio para interagir com periféricos, acessar endereços de memória que contém outros *softwares* e configurar o *hardware*, incluindo mudar o modo de privilégio!

Modos de privilégio e proteção do sistema

Estratégia: *Hardware* com dois modos de privilégio: superusuário e usuário.

- **Usuário** (modo não-privilegiado, …)
  - Neste modo, o *hardware* da CPU monitora a execução de instruções e caso um instrução tente realizar uma ação privilegiada, como acessar um CSR ou um endereço que foi marcado como protegido pelo código de sistema, a CPU interrompe a execução do *software* e invoca o código do sistema.

Modos de privilégio e proteção do sistema

Configurando o sistema

- Durante o *boot*, o *hardware* coloca, automaticamente, o sistema em modo superusuário e invoca o código de inicialização. Este código carrega e executa o código de sistema com privilégio de superusuário.
  - Com isso, o código de sistema pode configurar todo o *hardware*, incluindo configurar os periféricos, registrar rotinas de tratamento de interrupção e configurar os mecanismos de proteção do sistema (p.ex: endereços válidos)

Modos de privilégio e proteção do sistema

Executando código de usuário

- Com o *hardware* configurado, o código de sistema pode carregar aplicações de usuário na memória e executá-las. Para isso, o código de sistema:
  a) muda o modo de privilégio para "Usuário"; e
  b) salta para o endereço de entrada da aplicação.

OBS: Para garantir que o código de sistema retomará o controle da CPU, é comum o código de sistema configurar um temporizador em *hardware* para gerar interrupções periódicas.

Modos de privilégio e proteção do sistema

Tratando operações ilegais

- Como o código de usuário está executando com privilégio de Usuário, caso a aplicação tente realizar uma operação privilegiada, o *hardware* gera uma **exceção**, o que **interrompe a execução do código de usuário e invoca o código de sistema** para tratar a exceção.
  a) Ao tratar a exceção, o código de sistema pode optar por interromper a execução da aplicação!

Modos de privilégio e proteção do sistema

Chamando o *software* de sistema (p.ex: SO)

- O código de usuário executa com privilégio de Usuário.
  a) O *hardware* deve oferecer um mecanismo que permita o código de usuário invocar o código de sistema e mudar o modo de privilégio para superusuário;
  b) Este mecanismo não deve permitir que o código de usuário mude o modo de privilégio para superusuário e invoque seu próprio código!

Modos de privilégio e proteção do sistema

Chamando o *software* de sistema (p.ex: SO)

- O código de usuário executa com privilégio de Usuário.
  a) O *hardware* deve oferecer um **mecanismo que permita o código de usuário invocar o código de sistema e mudar o modo de privilégio para superusuário**;
  b) Este mecanismo não deve permitir que o código de usuário mude o modo de privilégio para super

Mecanismo de
interrupções por *software*

Modos de privilégio e proteção do sistema

Tratando interrupções externas

- Quando uma interrupção externa é gerada, o *hardware* muda o modo de privilégio para "superusuário".
  a) Dessa forma, o código de sistema tem permissão para realizar as operações necessárias para tratar a interrupção (p.ex: interagir com o periférico)

Agenda

- Modos de privilégio e proteção do sistema
- **Exceções**
- Interrupções por SW
- Proteção de sistemas RISC-V

Exceções

Exceções são eventos gerados pela CPU em resposta a situações excepcionais durante a execução de uma instrução.

Exemplos:

- Instrução com código de operação inválido;
- Acesso a endereço protegido;
- Salto para endereço que não é múltiplo de 4;
- Divisão por zero (Depende da ISA - não ocorre no RISC-V)

Exceções

Exceções são geralmente tratadas pela CPU de forma parecida com as interrupções externas:

- A CPU salva parte do contexto atual. Ex: PC e modo de operação atual;
- Muda o modo de operação para superusuário.
- Desvia o fluxo de execução para uma rotina que tratará a exceção.
  - Esta rotina é definida pelo código de sistema e registrada na CPU durante o processo de configuração do sistema.

Exceções

Exceções são geralmente tratadas pela CPU de forma **parecida** com as interrupções externas:

- Exceções geralmente não são mascaradas (filtradas) pela CPU.
- Dependendo do tipo da exceção, o código de sistema pode decidir interromper a execução do software.
  - Neste caso, basta não recuperar o contexto.

Agenda

- Modos de privilégio e proteção do sistema
- Exceções
- **Interrupções por SW**
- Proteção de sistemas RISC-V

Interrupções por *Software*

Interrupções por *Software* são eventos gerados pela CPU em resposta à execução de instruções especiais.

Podem ser usadas pelo código de usuário para invocar o código de sistema.

Exemplo:

- Instrução "`ecall`" no RISC-V;

Interrupções por *Software*

Interrupções por *Software* são geralmente tratadas pela CPU de forma parecida com as exceções. Em geral:

- A CPU salva parte do contexto atual. Ex: PC e modo de operação atual;
- Muda o modo de operação para superusuário.
- Desvia o fluxo de execução para uma rotina que tratará a interrupção por *software*.
  - Esta rotina é definida pelo código de sistema e registrada na CPU durante o processo de configuração do sistema.

Interrupções por *Software*

Inte
pel
ger

Garante que o código do sistema terá privilégio para realizar a operação requisitada!

- A CPU salva parte do contexto PC e modo de operação atual;
- Muda o modo de operação para **superusuário**.
- Desvia o fluxo de execução para uma rotina que tratará a interrupção por *software*.
  - Esta rotina é definida pelo código de sistema e registrada na CPU durante o processo de configuração do sistema.

Interrupções por *Software*

Inte das
pel n
ger

- A
  modo de operação atual;
- Muda o modo de operação p perusuário.
- Desvia o fluxo de execução p uma rotina que tratará a interrupção por *softw are*.
  - **Esta rotina é definida pelo código de sistema** e registrada na CPU durante o processo de configuração do sistema.

Garante que o código do usuário não conseguirá mudar o privilégio para superusuário e executar o próprio código!

Agenda

- Modos de privilégio e proteção do sistema
- Exceções
- Interrupções por SW
- **Proteção de sistemas RISC-V**
  - **Modos de privilégio no RISC-V**
  - **Execução de programas de usuário**
  - **Configuração dos mecanismos de exceção e interrupções por *software***
  - **Tratamento de exceções**
  - **Tratamento de interrupções por *software***

RISC-V: Modos de privilégio no RISC-V

O RISC-V define 3 modos de privilégio:

- Machine (Maior privilégio)
- Supervisor
- User/Application (Menor privilégio)

Processadores RISC-V podem implementar 1, 2 ou 3 modos de privilégio.

- Microcontroladores simples podem implementar somente o modo Machine.
- Processadores de computadores de mesa implementam dois ou mais modos.

RISC-V: Modos de privilégio no RISC-V

Em nossos exemplos suporemos que a CPU implementa dois modos de privilégio:

- Machine (Maior privilégio)
- User/Application (Menor privilégio)

Level	Encoding	Name	Abbreviation
0	00	User/Application	U
1	01	Supervisor	S
2	10	*Reserved*	
3	11	Machine	M

RISC-V: Execução de programas de usuário

Antes de saltar para o código do usuário, o código de sistema deve mudar o modo de privilégio para "Usuário" (User/Application).

No RISC-V, o modo de privilégio atual é armazenado por um dispositivo interno que não pode ser acessado diretamente com instruções de escrita e leitura na memória e em registradores.

RISC-V: Execução de programas de usuário

Para se mudar o modo de privilégio, devemos escrever no campo `mstatus.MPP` e executar a instrução de retorno de interrupções (`mret`).
Esta instrução:

- Modifica o campo `mstatus.MIE` usando o valor do campo `mstatus.MPIE`.
- Ajusta o modo de privilégio atual com o valor do campo MPP (Machine Previous Privilege);
- Salta para o endereço no registrador `MEPC`;

RISC-V: Execução de programas de usuário

Para mudar o modo de privilégio, devemos escrever no campo `mstatus.MPP` e executar a instrução de retorno de interrupções (`mret`).
Exemplo:

```
csrr t1, mstatus      # Update the mstatus.MPP
li t2, ~0x1800        #  field (bits 11 and 12)
and t1, t1, t2        #  with value 00 (U-mode)
csrw mstatus, t1

la t0, user_main      # Loads the user software
csrw mepc, t0         # entry point into mepc

mret                  # PC <= MEPC; mode <= MPP;
```

RISC-V: Configuração de exceções e int. por *software*

O mecanismo de exceções e interrupções por *software* é configurado de forma similar ao mecanismo de interrupções externas.

- As rotinas de tratamento das exceções e interrupção por *software* devem ser registradas com o auxílio do registrador `mtvec`.

RISC-V: Configuração de exceções e int. por *software*

Registrando rotinas de tratamento de exceções e interrupção por *software* com o auxílio do registrador `mtvec`.

- Modo *direct*: Uma única rotina é registrada. Ela é responsável por inspecionar o registrador `mcause` para determinar o tipo de evento (interrupção externa, exceção ou interrupção por *software*) e a causa do evento (instrução inválida, …).
- Modo *vectored*: As rotinas de tratamento de interrupções externas, exceções e de interrupções por *software* são registradas na tabela de interrupções.

RISC-V: Configuração de exceções e int. por *software*

O mecanismo de exceções e interrupções por *software* é configurado de forma similar ao mecanismo de interrupções externas.

- As rotinas de tratamento das exceções e interrupção por *software* devem ser registradas com o auxílio do registrador `mtvec`.
- O sistema deve reservar espaço para salvar o contexto durante o tratamento das exceções e interrupções por *software*. O registrador `mscratch` pode ser usado para auxiliar neste processo (similar ao tratamento de interrupções).

RISC-V: Configuração de exceções e int. por *software*

OBS: Sistemas que possuem unidade de proteção (ou gerenciamento) de memória devem ser configurados para proteger endereços que estão mapeados em periféricos e palavras de memória ocupadas pelo código de sistema (ou de outros processos) antes de invocar o código de usuário.

- O funcionamento deste dispositivo é coberto na disciplina de sistemas operacionais.

RISC-V: Tratamento de exceções

No RISC-V, exceções são tratadas pela CPU de forma parecida com as interrupções externas. A CPU:

- Salva parte do contexto atual
  - `mstatus.MPP <= modo de privilégio`
  - `mepc <= pc`
  - `mstatus.MPIE <= mstatus.MIE`
- Muda o modo de operação para Machine mode
  - `modo de privilégio <= "11" (Machine)`
- Desabilita interrupções
  - `mstatus.MIE <= "0" (Machine)`
- Desvia o fluxo de execução para uma rotina que tratará a exceção.
  - `pc <= mtvec.BASE  # (Modo direct)`

RISC-V: Tratamento de exceções

A rotina de tratamento de exceções deve:

- Salvar o restante do contexto;
- Tratar a exceção; e
- Recuperar o contexto
  - Este último passo não é necessário nos casos onde a aplicação deve ser interrompida!

RISC-V: Tratamento de exceções

Salvando o contexto:

- Na inicialização do sistema: Fazer `mscratch` apontar para uma posição de memória que possa ser usada pela rotina de tratamento de exceções para salvar o contexto.
- Na rotina de tratamento de exceções: Trocar o conteúdo de `mscratch` por um registrador de propósito geral no início e no final da rotina.
  - Em nosso exemplo suporemos que `mscratch` aponta para uma pilha dedicada às rotinas de tratamento de interrupções externas, exceções e interrupções por sw.

RISC-V: Tratamento de exceções

Usando o `mscratch` para auxiliar a rotina a salvar e recuperar o contexto

```
exception_handling:
  # Salvar o contexto
  csrrw sp, mscratch, sp  # Troca sp com mscratch
  addi  sp, sp, -64       # Aloca espaço na pilha
  sw a0, 0(sp)            # Salva a0
  sw a1, 4(sp)            # Salva a1
  ...
  # Trata a exceção
  ...
  # Recupera o contexto
  ...
  lw a1, 4(sp)            # Recupera a1
  lw a0, 0(sp)            # Recupera a0
  addi  sp, sp, 64        # Desaloca espaço da pilha
  csrrw sp, mscratch, sp # Troca sp com mscratch novamente
  mret                    # Recupera o restante do contexto
```

Fazer SP apontar para a pilha das rotinas de tratamento e salvar o contexto nesta pilha!

RISC-V: Tratamento de exceções

Usando o `mscratch` para auxiliar a rotina a salvar e recuperar o contexto

```
exception_handling:
  # Salvar o contexto
  csrrw sp, mscratch, sp  # Troca sp com mscratch
  addi  sp, sp, -64       # Aloca espaço na pilha
  sw a0, 0(sp)            # Salva a0
  sw a1, 4(sp)            # Salva a1
  ...
  # Trata a exceção
  ...
  # Recupera o contexto
  ...
  lw a1, 4(sp)            # Recupera a1
  lw a0, 0(sp)            # Recupera a0
  addi  sp, sp, 64        # Desaloca espaço da pilha
  csrrw sp, mscratch, sp # Troca sp com mscratch novamente
  mret                    # Recupera o restante do contexto
```

Tratar a interrupção.

RISC-V: Tratamento de exceções

Usando o `mscratch` para auxiliar a rotina a salvar e recuperar o contexto

```
exception_handling:
  # Salvar o contexto
  csrrw sp, mscratch, sp  # Troca sp com mscratch
  addi  sp, sp, -64       # Aloc
  sw a0, 0(sp)            # Sal
  sw a1, 4(sp)            # Sal
  ...
  # Trata a exceção
  ...
  # Recupera o contexto
  ...
  lw a1, 4(sp)            # Recupera a1
  lw a0, 0(sp)            # Recupera a0
  addi  sp, sp, 64        # Desaloca espaço da pilha
  csrrw sp, mscratch, sp # Troca sp com mscratch novamente
  mret                    # Recupera o restante do contexto
```

Recuperar o contexto do programa anterior:

1 - Recuperar os registradores de propósito geral e desalocar o espaço da pilha.

RISC-V: Tratamento de exceções

Usando o `mscratch` para auxiliar a rotina a salvar e recuperar o contexto

```
exception_handling:
  # Salvar o contexto
  csrrw sp, mscratch, sp  # Troca sp com mscratch
  addi  sp, sp, -64       # Aloc
  sw a0, 0(sp)            # Sal
  sw a1, 4(sp)            # Sal
  ...
  # Trata a exceção
  ...
  # Recupera o contexto
  ...
  lw a1, 4(sp)            # Recu      a1
  lw a0, 0(sp)            # Recupera a0
  addi  sp, sp, 64        # Desaloca espaço da pilha
  csrrw sp, mscratch, sp  # Troca sp com mscratch novamente
  mret                    # Recupera o restante do contexto
```

Recuperar o contexto do programa anterior:
2 - Recuperar o SP do programa

RISC-V: Tratamento de exceções

Usando o `mscratch` para auxiliar a rotina a salvar e recuperar o contexto

```
exception_handling:
  # Salvar o contexto
  csrrw sp, mscratch, sp  # Troca sp com mscratch
  addi  sp, sp, -64       # Aloc
  sw a0, 0(sp)            # Sal
  sw a1, 4(sp)            # Sal
  ...
  # Trata a exceção
  ...
  # Recupera o contexto
  ...
  lw a1, 4(sp)            # Recupera a1
  lw a0, 0(sp)            # Recupera a0
  addi  sp, sp, 64        # Desaloca espaço da pilha
  csrrw sp, mscratch, sp  # Troca sp com mscratch novamente
  mret                    # Recupera o restante do contexto
```

Recuperar o contexto do programa anterior:

3 - Recuperar o contexto que foi salvo automaticamente pela CPU (mstatus.MIE, pc, …)

RISC-V: Tratamento de exceções

A instrução `mret`:

- Instrução especial utilizada para retornar de rotinas de tratamento de interrupções externas, exceções e interrupções por *software*.
- Recupera o estado que foi salvo automaticamente pela CPU quando a interrupção foi tratada.
  - `pc <= mepc`
  - `mstatus.MIE <= mstatus.MPIE`
  - `modo de privilégio <= mstatus.MPP`

RISC-V: Tratamento de interrupções por *software*

No RISC-V, interrupções por *software* são tratadas pela CPU de forma idêntica às exceções.

- A instrução "ecall" gera uma exceção!
- O registrador mcause pode ser usado para identificar a causa da exceção (p.ex: instrução inválida, *environment call*, …)

RISC-V: Tratamento de interrupções por *software*

A rotina de tratamento de interrupções por *software* deve:

- Terminar de salvar o contexto;
- Tratar a interrupção por *software*; e
- Recuperar o contexto.

RISC-V: Tratamento de interrupções por *software*

A rotina de tratamento de interrupções por *software* deve:

- Terminar de salvar o contexto;
- Tratar a interrupção por *software*; e
- Recuperar o contexto.
  - Este último passo também pode ser opcional. Por exemplo, a chamada de sistema *exit* solicita a interrupção do processo, logo, o contexto não deve ser recuperado.

RISC-V: Tratamento de interrupções por *software*

A rotina de tratamento de interrupções por *software* deve:

- Terminar de salvar o contexto;
- Tratar a interrupção por *software*; e
- Recuperar o contexto.
  - Este último passo também pode ser opcional. Por exemplo, a chamada de sistema *exit* solicita a interrupção do processo, logo, o contexto não deve ser recuperado.
  - O endereço salvo em `MEPC` é o endereço da instrução que gerou a interrupção por *software*, no entanto, o tratador de interrupções deve retornar para instrução subsequente.
    - O endereço em MEPC deve ser ajustado!

RISC-V: Tratamento de interrupções por *software*

Ajustando o endereço em MEPC antes de retornar da rotina de tratamento da interrupção por *software*.

```
# Ajustando MEPC para retornar de uma chamada de sistema
csrr a1, mepc  # carrega endereço de retorno (endereço
               #  da instrução que invocou a syscall)
addi a1, a1, 4 # soma 4 no endereço de retorno
               #  (para retornar após a ecall)
csrw mepc, a1  # armazena endereço de retorno
               #  de volta no mepc
```