William Stallings
Computer Organization and Architecture

Chapter 11
CPU Structure and Function

CPU Structure

- CPU must:
  - Fetch instructions
  - Interpret instructions
  - Fetch data
  - Process data
  - Write data

Registers

- CPU must have some working space (temporary storage)
- Called registers
- Number and function vary between processor designs
- One of the major design decisions
- Top level of memory hierarchy

User Visible Registers

- General Purpose
- Data
- Address
- Condition Codes

General Purpose Registers (1)

- May be true general purpose
- May be restricted
- May be used for data or addressing
- Data
  - Accumulator
- Addressing
  - Segment

General Purpose Registers (2)

- Make them general purpose
  - Increase flexibility and programmer options
  - Increase instruction size & complexity
- Make them specialized
  - Smaller (faster) instructions
  - Less flexibility

How Many GP Registers?

- Between 8 - 32
- Fewer = more memory references
- More does not reduce memory references and takes up processor real estate
- See also RISC

How big?

- Large enough to hold full address
- Large enough to hold full word
- Often possible to combine two data registers
  - C programming
  - double int a;
  - long int a;

Condition Code Registers

- Sets of individual bits
  - e.g. result of last operation was zero
- Can be read (implicitly) by programs
  - e.g. Jump if zero
- Can not (usually) be set by programs

Control & Status Registers

- Program Counter
- Instruction Decoding Register
- Memory Address Register
- Memory Buffer Register

- Revision: what do these all do?

Program Status Word

- A set of bits
- Includes Condition Codes
- Sign of last result
- Zero
- Carry
- Equal
- Overflow
- Interrupt enable/disable
- Supervisor

Supervisor Mode

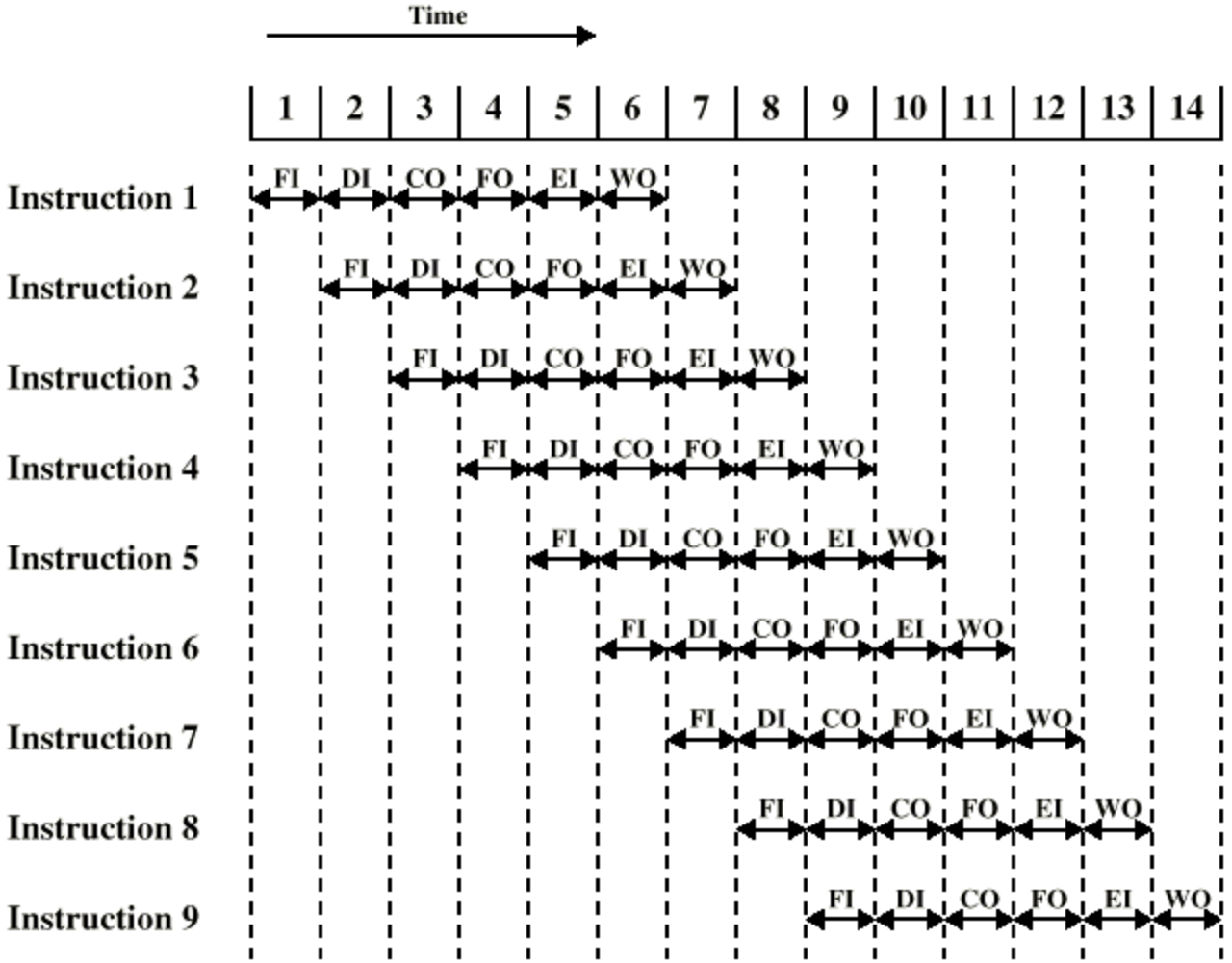
- Intel ring zero
- Kernel mode
- Allows privileged instructions to execute
- Used by operating system
- Not available to user programs

Pipelining

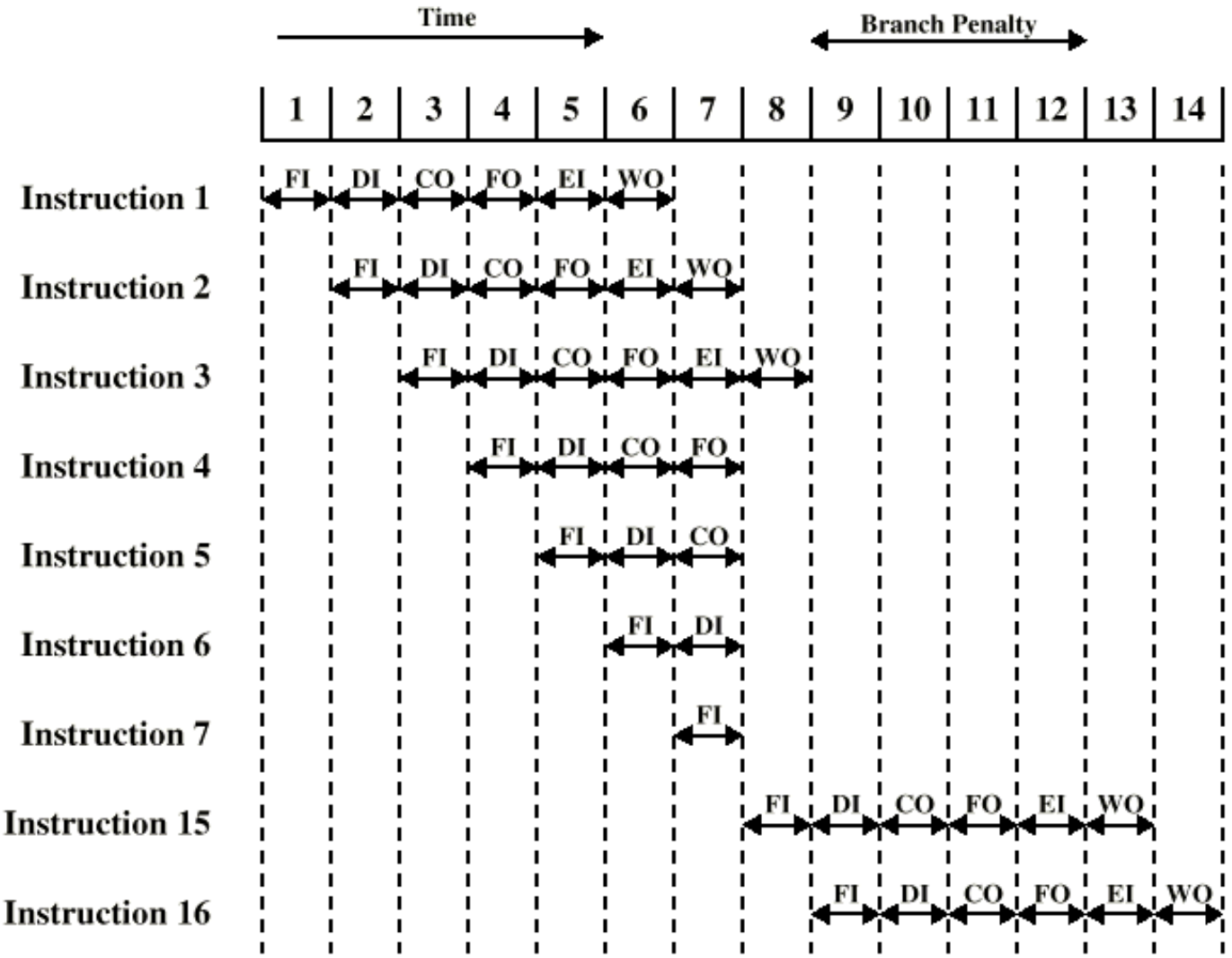
- Fetch instruction
- Decode instruction
- Calculate operands (i.e. EAs)
- Fetch operands
- Execute instructions
- Write result

- Overlap these operations

Timing of Pipeline



Branch in a Pipeline



Dealing with Branches

- Multiple Streams (Fluxos)
- Prefetch Branch Target
- Loop buffer
- Branch prediction
- Delayed branching

Multiple Streams

- Have two pipelines
- Prefetch each branch into a separate pipeline
- Use appropriate pipeline

- Leads to bus & register contention
- Multiple branches lead to further pipelines being needed

Prefetch Branch Target

- Target of branch is prefetched in addition to instructions following branch
- Keep target until branch is executed
- Used by IBM 360/91

Loop Buffer

- Very fast memory
- Maintained by fetch stage of pipeline
- Check buffer before fetching from memory
- Very good for small loops or jumps
- c.f. cache
- Used by CRAY-1

Execução Especulativa

- Execução especulativa consiste em um conjunto de técnicas para antecipar a execução de instruções antes que todas as dependências tenham sido resolvidas.
- A não resolução de todas as dependências pode significar inclusive que instruções são executadas desnecessaria ou erroneamente. Todavia, o ganho representado pela não paralisação do pipeline é comumente maior que o custo de possíveis execuções equivocadas.

Execução Especulativa

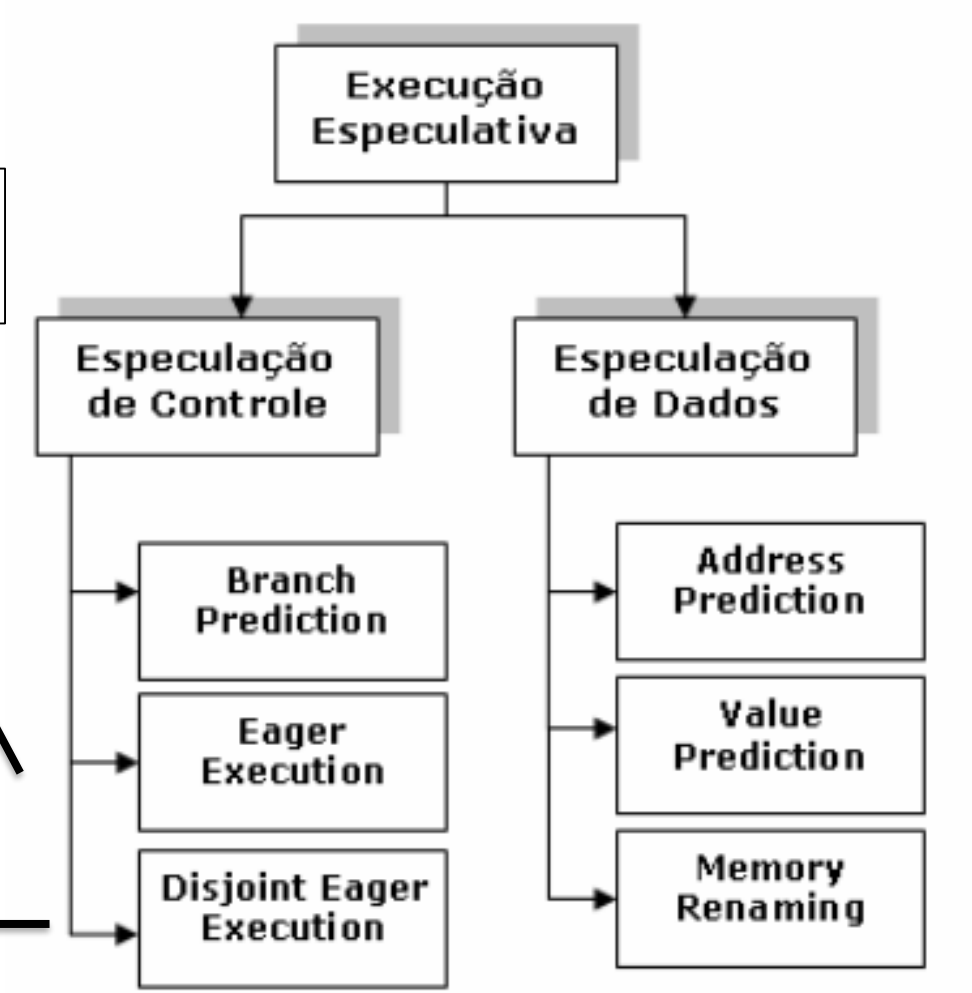
- A antecipação proporcionada pela execução especulativa, na maioria dos casos, é baseada em análise estatística dos resultados previamente obtidos. Essa análise é realizada explorando-se a localidade de valor, espaço e tempo, que é observada na imensa maioria dos softwares desenvolvidos.

Execução Especulativa

- As Dependências de Dados ocorrem quando uma instrução depende de um dado que ainda não foi obtido ou calculado por outra instrução precedente. Principalmente em virtude da latência da instrução load, as dependências de dados consistem num importante gargalo para os processadores com pipeline.
- As seguintes técnicas de execução especulativa procuram reduzir esse problema:
  - Address Prediction: Técnicas que tentam predizer em qual posição de memória dados ou instruções estão armazenados.
  - Value Prediction:Técnicas que procuram predizer qual o valor está armazenado em um registrador ou em um endereço de memória.
  - Memory Renaming:Técnicas que procuram comunicar valores já armazenados para instruções loads.

Execução Especulativa

Nessa técnica todos os possíveis caminhos de execução são testados



Variação da técnica
Eager Execution em que as restrições de disponibilidade de recursos são levadas em consideração para determinação de quais caminhos, selecionados de acordo com a probabilidade de serem tomados, serão executados.

Meltdown & Spectre

- Spectre: As brechas chamadas de Spectre quebram a barreira de isolamento entre aplicações. Potencialmente permitem que uma aplicação maliciosa ”force” outra aplicação a acessar de forma arbitrária segmentos de informações em sua memória, que podem ser visualizadas por um canal paralelo, vazando informações. A vulnerabilidade é baseada em uma falha de design, que explora a técnica de execução especulativa.
- Meltdown: A brecha chamada de Meltdown quebra a barreira de isolamento entre aplicação e sistema operacional. Permite que qualquer aplicação executada pelo usuário acesse a memória central do computador. A palavra meltdown significa derreter, em inglês, em alusão ao fato de que a aplicação pode ignorar a barreira de segurança do processador, que impediria o acesso à memória do kernel dos sistemas operacionais (seja Windows, Linux ou macOS). O kernel possui acesso a praticamente todos os processos de um dispositivo.

Branch Prediction (1)

- Predict never taken
  - Assume that jump will not happen
  - Always fetch next instruction
  - 68020 & VAX 11/780
  - VAX will not prefetch after branch if a page fault would result (O/S v CPU design)
- Predict always taken
  - Assume that jump will happen
  - Always fetch target instruction

Branch Prediction (2)

- Predict by Opcode
  - Some instructions are more likely to result in a jump than thers
  - Can get up to 75% success
- Taken/Not taken switch
  - Based on previous history
  - Good for loops

Branch Prediction (3)

- Delayed Branch
  - Do not take jump until you have to
  - Rearrange instructions