Unit 0. Introduction to the computer architecture

Computer Architecture

Area of Computer Architecture and Technology
Department of Computer Science and Engineering
University of Oviedo

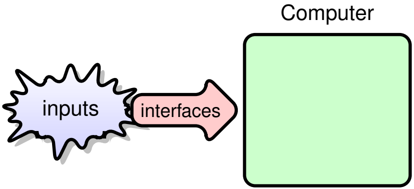
Fall, 2015

Computer

Definition

Machine which is able to execute algorithms

- receives inputs
- a program processes data
- produces outputs



Examples

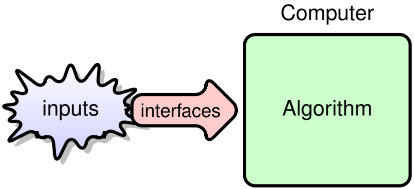
- desktop computers
- laptops, tablets
- smartphones

Computer

Definition

Machine which is able to execute algorithms

- receives inputs
- a program processes data
- produces outputs



Examples

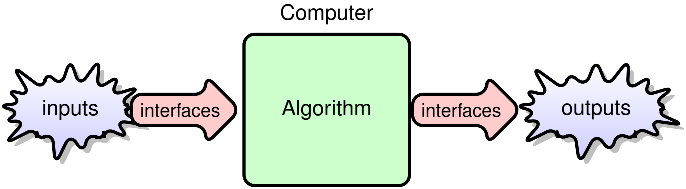
- desktop computers
- laptops, tablets
- smartphones

Computer

Definition

Machine which is able to execute algorithms

- receives inputs
- a program processes data
- produces outputs



Examples

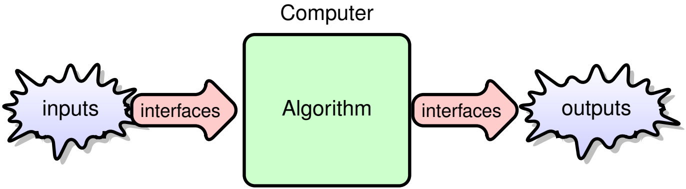
- desktop computers
- laptops, tablets
- smartphones

Computer

Definition

Machine which is able to execute algorithms

- receives inputs
- a program processes data
- produces outputs



Examples

- desktop computers
- laptops, tablets
- smartphones

Computer design

Requirements

- store (and remove) the program to be executed
- store data
- load and execution of the program
- input/output mechanism

Solution: several designs

- memory
- processor (CPU)
- peripherals

Computer design

Requirements

- store (and remove) the program to be executed
- store data
- load and execution of the program
- input/output mechanism

Solution: several designs

- memory
- processor (CPU)
- peripherals

Computer design

Requirements

- store (and remove) the program to be executed
- store data
- load and execution of the program
- input/output mechanism

Solution: several designs

- memory
- processor (CPU)
- peripherals

Computer design

Requirements

- store (and remove) the program to be executed
- store data
- load and execution of the program
- input/output mechanism

Solution: several designs

- memory
- processor (CPU)
- peripherals

Computer design

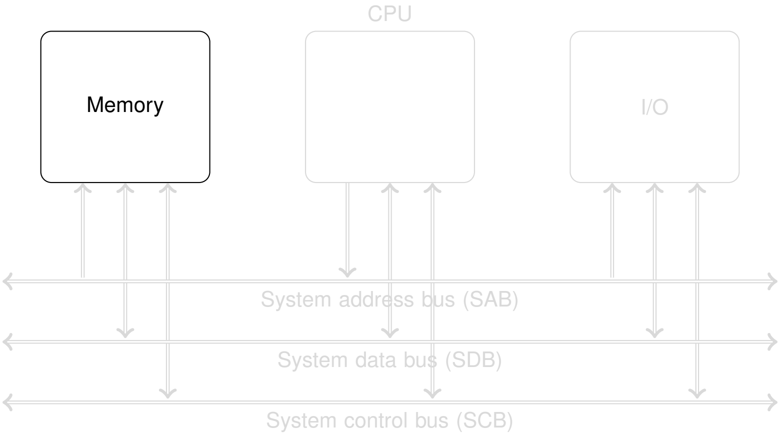
Requirements

- store (and remove) the program to be executed
- store data
- load and execution of the program
- input/output mechanism

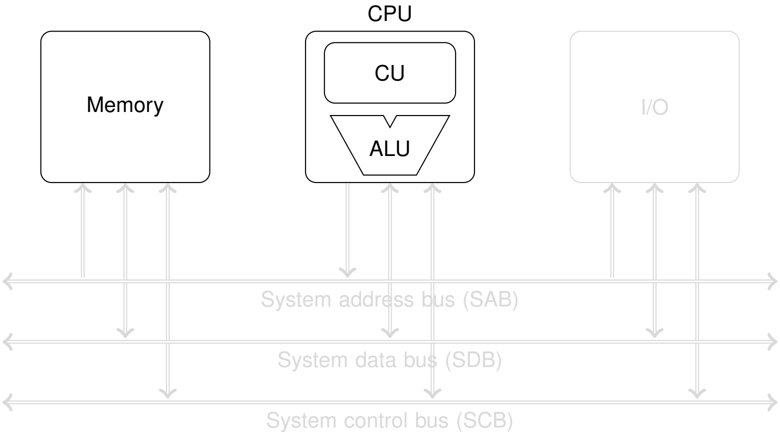
Solution: several designs

- memory
- processor (CPU)
- peripherals

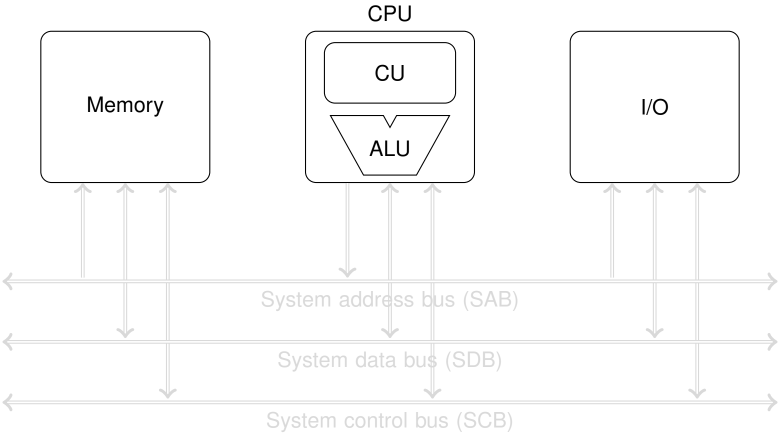
Von Neumann architecture



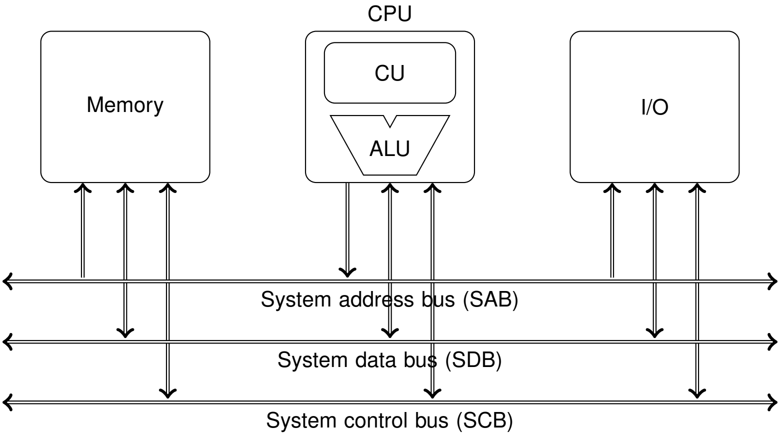
Von Neumann architecture



Von Neumann architecture



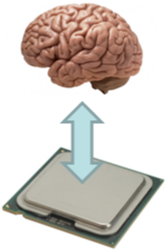
Von Neumann architecture



Conceptual distance

Problem

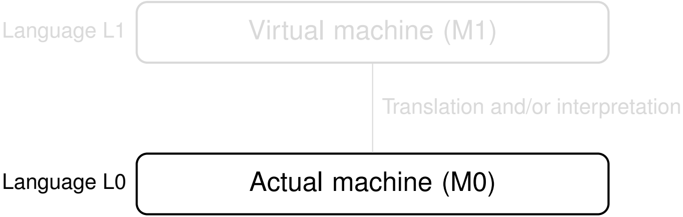
- programmer interests ⇒ natural language
- machine constraints ⇒ simple machine code



Virtual machine

Solution

- define a new language (L1)
- closer to the natural language
- built upon machine language (L0)



Language L1

Virtual machine (M1)

Translation and/or interpretation

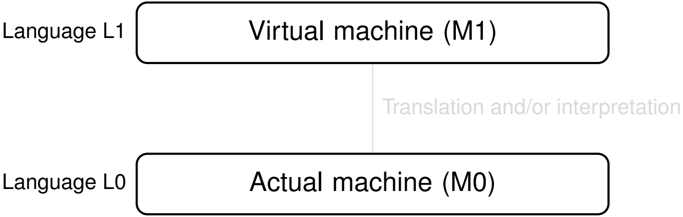
Language L0

Actual machine (M0)

Virtual machine

Solution

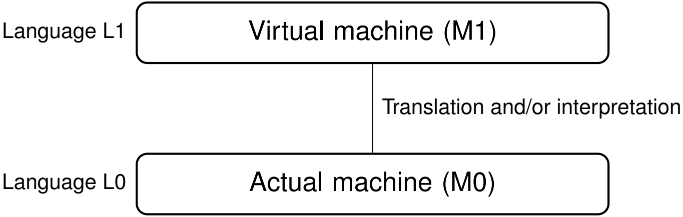
- define a new language (L1)
- closer to the natural language
- built upon machine language (L0)



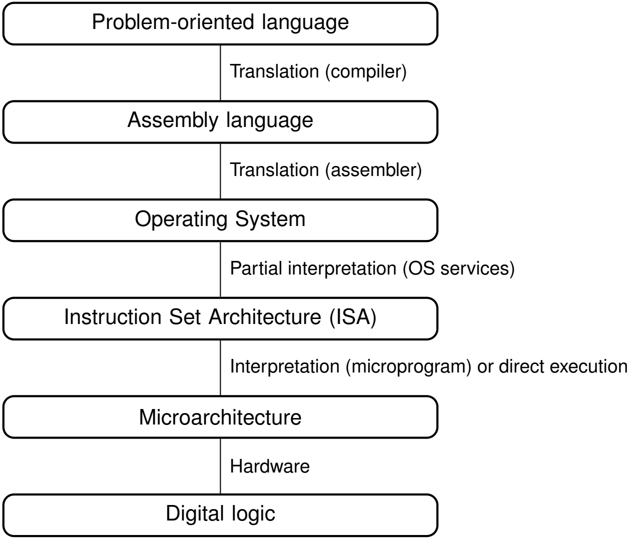
Virtual machine

Solution

- define a new language (L1)
- closer to the natural language
- built upon machine language (L0)



Multi-level machine



Problem-oriented language

Translation (compiler)

Assembly language

Translation (assembler)

Operating System

Partial interpretation (OS services)

Instruction Set Architecture (ISA)

Interpretation (microprogram) or direct execution

Microarchitecture

Hardware

Digital logic

Computer architecture

Definition

- functional behavior
- interconnection of components
- I/O mechanisms
- etc.

Example

- Von Neumann architecture
- Harvard architecture

Two levels

- Instruction set architecture (ISA)
- Microarchitecture

Instruction set architecture

Visible to the programmer

- data types
- registers
- instruction set
- address format

RISC (*Reduced Instruction Set Computer*)

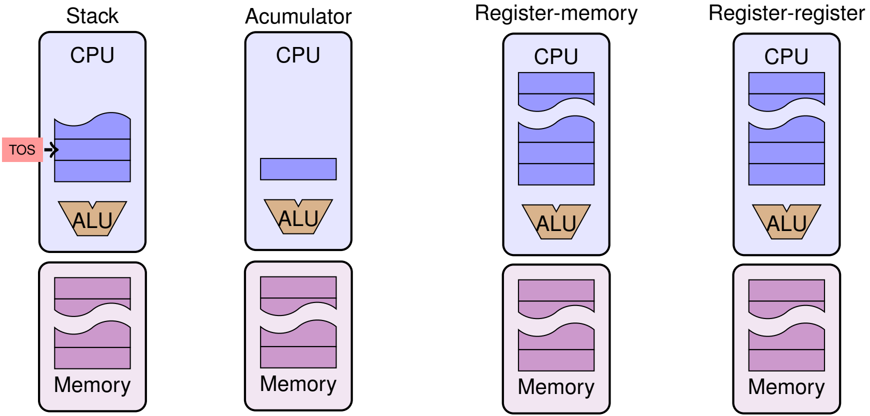
- very simple instructions ⇒ quick execution
- Ex.: PowerPC, IA-64, SPARC, MIPS, ARM

CISC (*Complex Instruction Set Computer*)

- complex instructions ⇒ high level operations
- Ex.: x86, Motorola 68 000

Instruction set architecture

ISA types: machine models (`C = A + B`)



```
Push A
Push B
```

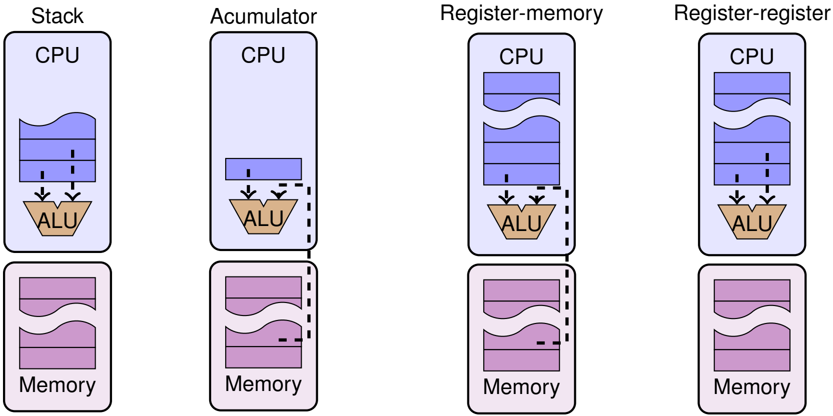
```
Load A
```

```
Load R1, A
```

```
Load R1, A
Load R2, B
```

Instruction set architecture

ISA types: machine models (`C = A + B`)



```
Push A
Push B
Add
```

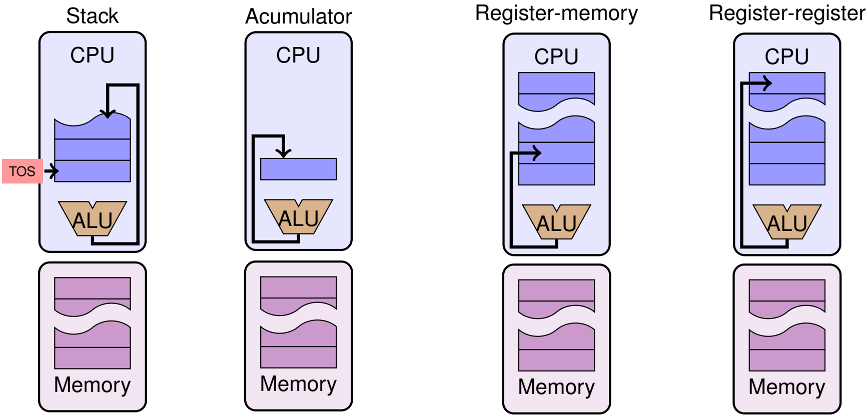
```
Load A
Add B
```

```
Load R1, A
Add R3, R1, B
```

```
Load R1, A
Load R2, B
Add R3, R1, R2
```

Instruction set architecture

ISA types: machine models (`C = A + B`)



```
Push A
Push B
Add
```

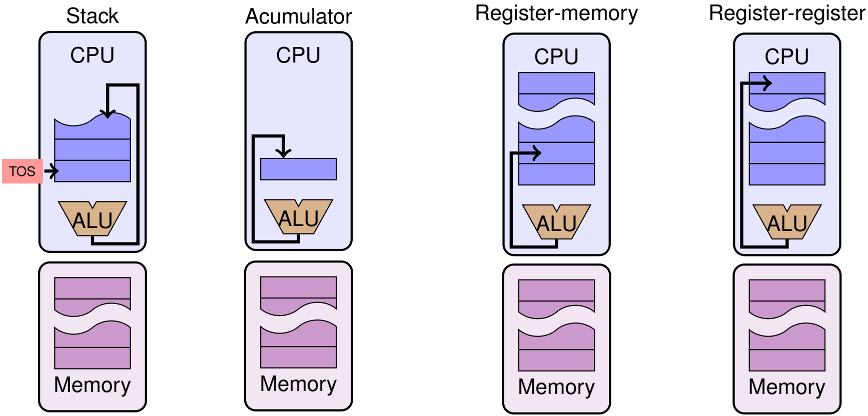
```
Load A
Add B
```

```
Load R1, A
Add R3, R1, B
```

```
Load R1, A
Load R2, B
Add R3, R1, R2
```

Instruction set architecture

ISA types: machine models (`C = A + B`)



```
Push A
Push B
Add
Pop C
```

```
Load A
Add B
Store C
```

```
Load R1, A
Add R3, R1, B
Store R3, C
```

```
Load R1, A
Load R2, B
Add R3, R1, R2
Store R3, C
```

Microarchitecture

Implementation of an ISA

- functional units
- cache (type and level count)
- bus width

Internal organization of the CPU

- segmented
- superscalar
- multicore
- VLIW (*Very Long Instruction Word*)
- vectorial

Examples of x86 microarchitectures

- Intel Nehalem (Core i5), AMD K10 (Athlon II)

Unit 0. Introduction to the computer architecture

Computer Architecture

Area of Computer Architecture and Technology
Department of Computer Science and Engineering
University of Oviedo

Fall, 2015