Estructura de Computadors (EC)

2023 - 2024 Q2

Professors grup 20

Adrià Armejach - adria.armejach@upc.edu

- Consultes
  - Via e-mail - dubtes ràpids o per concretar hora presencial

Laboratori:

- 21/22: Toni Cortes
- 23: David Alvarez

Pàgina web

- Tota la informació de l'assignatura està a:

  https://docencia.ac.upc.edu/FIB/grau/EC/

  - Apunts teoria i bibliografia
  - Quaderns de problemes i enunciats de les sessions de laboratori
  - Calendari de classes de laboratori
  - Exàmens d'anys anteriors
- Bibliografia
  - D. Patterson and J. L. Hennessy. "Estructura y Diseño de Computadores: La Interfaz Hardware/Software", 4a edició, 2011

Teoria, Laboratori i Exàmens

- 27 sessions de teoria i problemes
  - Dimecres de 10:00 a 12:00 i Dijous de 12:00 a 14:00 (A5E01)
  - Problemes intercalats amb teoria -> porteu la col·lecció de problemes (impresa o virtual)
- 6 sessions de laboratori
  - Sessió 0: Introducció
  - Sessió 1: Ensamblador MIPS y tipos de datos básicos
  - Sessió 2: Traducción de programas
  - Sessió 3: Tipos de datos estructurados
  - Sessió 4: Codificación en coma flotante
  - Sessió 5: Memoria cache
- Exàmens
  - Exàmen parcial: 9 d'Abril 2024 de 10:30 a 12:30
  - Exàmen de laboratori: Horari de classe, mirar calendari web EC
  - Exàmen final: 17 de Juny 2024 de 15:00 a 18:00

Avaluació

- NotaFinal = **0.2** * max(EP,EF) + **0.6** * EF + **0.2** * (EL * 0.85 + EC * 0.15)
  - EP = Exàmen Parcial
  - EF = Exàmen Final
  - EL = Exàmen de Laboratori
  - AC = Avaluació Continuada de Laboratori

- Avaluació continua a cada sessió de laboratori
  - Estudi previ individual
  - Per parelles durant la sessió presencial

Objectius

- Conèixer la jerarquia de nivells de un computador
- Conèixer l'arquitectura (ISA) de un procesador RISC
- Saber representar i operar amb números reals i enters
- Saber com s'emmagatzemen i com s'accedeix a tipus de dades estructurats
- Saber traduir programes d'alt nivell (p.e., en *c*) a llenguatge assemblador
- Conèixer l'estructura i el funcionament de les memòries cache
- Entendre el funcionament bàsic de la memòria virtual
- Conèixer els conceptes de excepció i interrupció

Temari

1. Introducció
2. Assemblador i tipus de dades bàsics
3. Traducció de programes
4. Matrius
5. Aritmética d'enters i coma flotant
6. Memòria cache
7. Memòria virtual
8. Excepcions i interrupcions

Transparències

- Les transparencies que fem servir a classe es podran trobar després de cada sessió aqui:

  http://personals.ac.upc.edu/adria/EC-2324Q2

Tema 1
Introducció (Part 1)

Estructura de Computadors (EC)

Enginyeria de Computadors

ENIAC
Primer computador digital programable
Ocupava 170 m^2
Pesava 30 tones

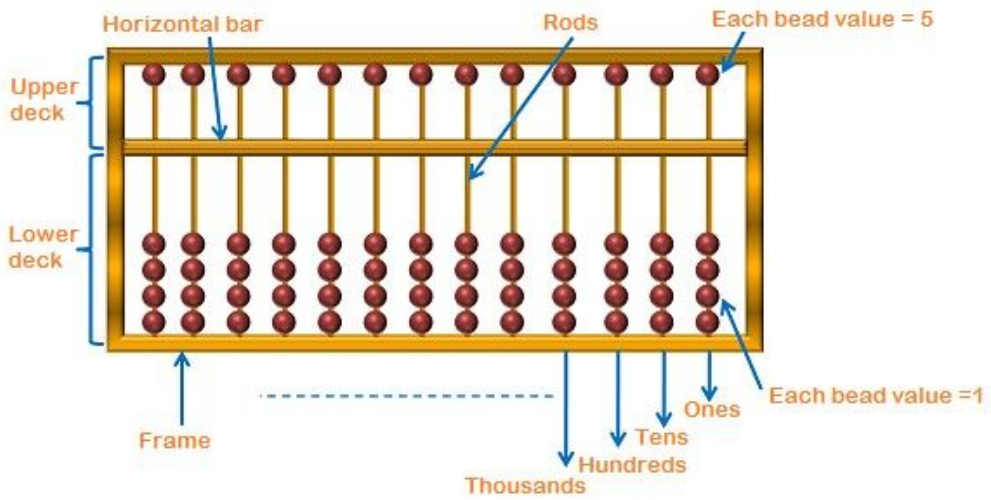
IBM Power4
Primer processador amb dos cores
174.000.000 transistors

Intel 8088
Successor del 8086
Processador de 16 bits que s'integra a tots els IBM PCs
29.000 transistors

AMD EPYC Rome
64 cores (128 threads)
39.000.000.000 transistors

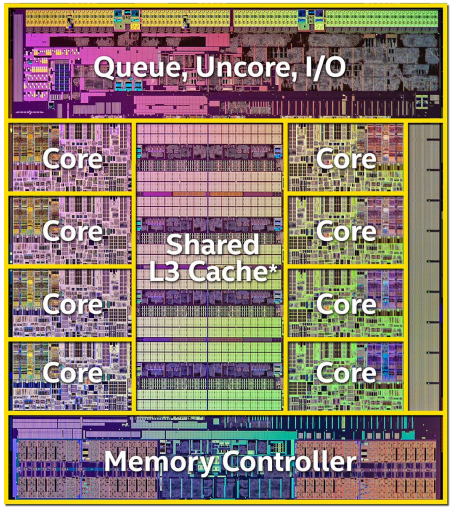
El Primer Dispositiu de Còmput

L'àbac



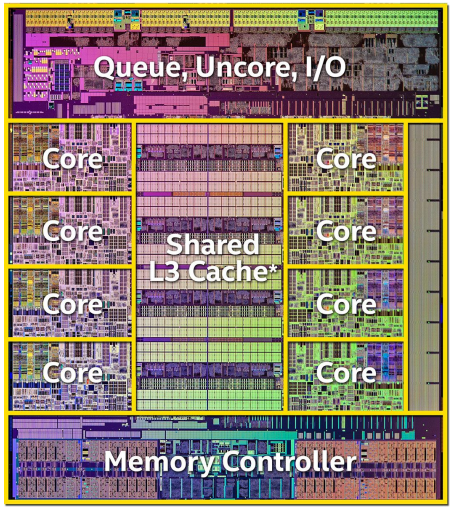
1. Mecànic
2. Manual
3. Poc emmagatzematge
4. Lent

Computador Digital



1. Electronic
2. Automatic / **Programable**
3. Molt emmagatzematge (TBytes)
4. Molt ràpid (TFLOPS)

Computador Digital



1. Electronic
2. Automàtic / **Programable**
3. Molt emmagatzematge (TBytes)
4. Molt ràpid (TFLOPS)

- Dispositiu electronic capaç d'emmagatzemar i processar informació de manera automática
- Com el programem? No podem interactuar amb electrons de manera directa

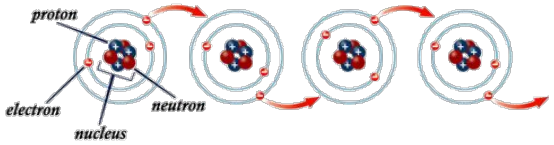
Nivells d'abstracció

Interfície d'usuari
menus, icones, botons, …

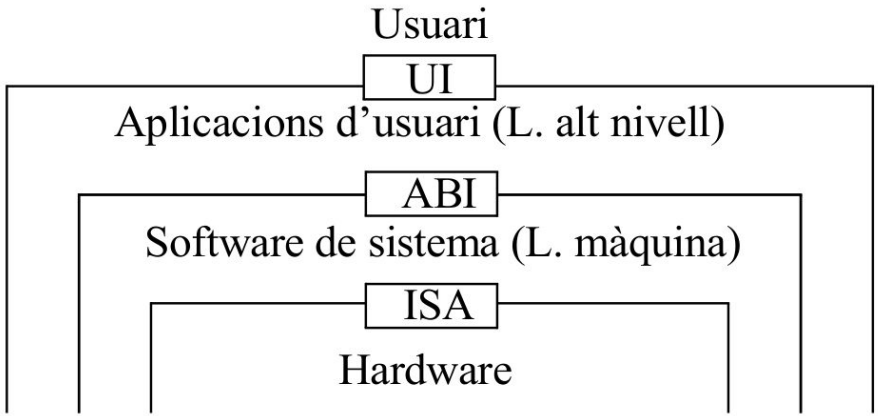
Llenguatge alt nivell
C/C++, java, c#, python, … (Pro2)

Llenguatge màquina
MIPS, x86_64, ARM, RISC-V, … (EC)
(instruccions, registres, modes d'adreçament, model memòria…)

Hardware
Portes lògiques, multiplexors, biestables, … (IC)

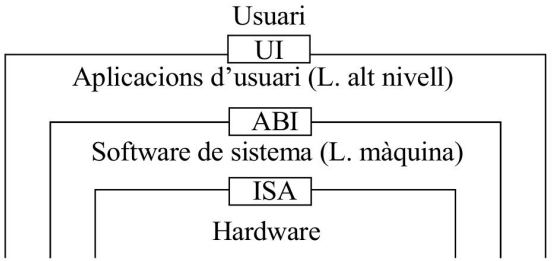


Interfícies entre nivells



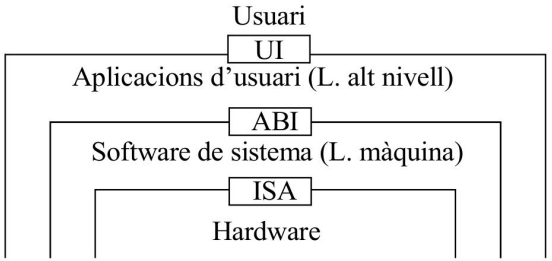
- Cada nivell d'abstracció estableix una interfície
  - El nivell superior pot interactuar amb el computador sense conèixer cap més detall de la implementació (caixa negra)

Instruction Set Architecture (ISA)



- Interfície entre el hardware i el software
- Descriu els aspectes del processador que son visibles al programador de llenguatge màquina
  - Repertori d'instruccions - sintaxi, format
  - Registres disponibles
  - Modes de direccionament
  - Model de memoria
  - Excepcions
- Exemples: **MIPS**, x86_64, ARM, RISC-V

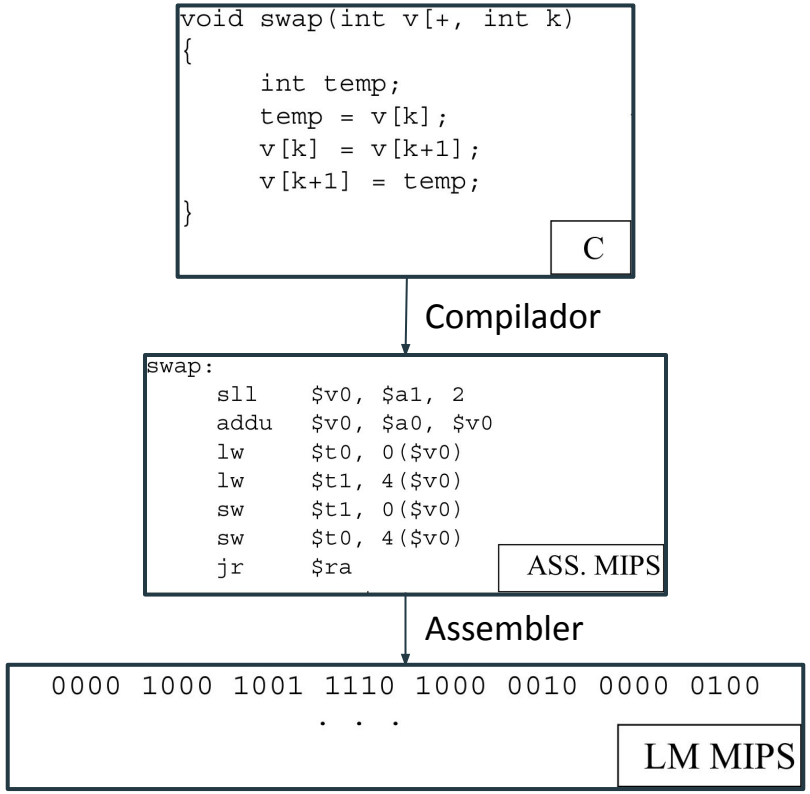
Application Binary Interface (ABI)



- Interfície entre el software de sistema (SO) i les aplicacions d'usuari
- Descriu les funcions del SO:
  - Com es gestiona la memòria
  - Com s'emmagatzemen els programes i les dades
  - L'accés als serveis del SO per mitjà de crides al sistema
  - Estableix convenis de crida i retorn de funcions, pas de paràmetres, …
- El compilador és l'encarregat d'assegurar que es compleix l'ABI

Traducció entre nivells

- Llenguatge d'alt nivell
  - Abstracte
  - Portable
  - Ràpid de escriure
- Llenguatge assemblador
  - Representació textual de les instruccions
- Llenguatge màquina
  - Representació hardware (bits)
  - Mateix nivell que assemblador (traducció directe)



Compilació vs Interpretació

- Compilació
  - Traducció del programa sencer una única vegada (estàtic)
  - \+ Programa generat optimitzat i ràpid
  - − Requereix recompilar per cada ISA i/o ABI (no és portable)
  - Exemples: C/C++, fortran, pascal, …
- Interpretació
  - Traducció dinàmica en temps d'execució (és tradueix a cada execució)
  - − Execució més lenta
  - \+ Portable
  - Exemples: java, python, …
- Llenguatges interpretats més productius, portables i segurs
  - Pero son massa lents en situacions on el rendiment és crític

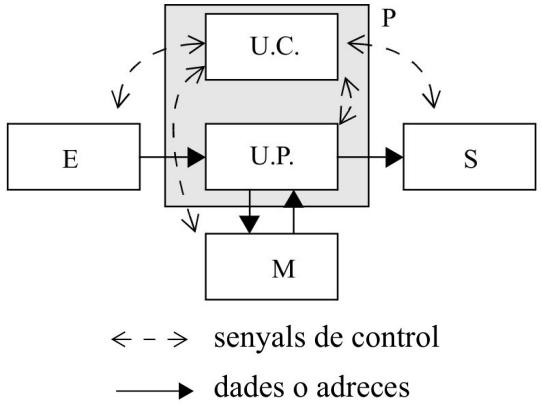
Exemple: Compilació - Interpretació

EXEMPLE 1: Tenim un computador MIPS ¿Com podem executar-hi un programa escrit en Java si disposem del següent software?

- Compilador de C a x86 (escrit en x86)
- Traductor binari de x86 a MIPS (escrit en MIPS)
- Intèrpret de Java (escrit en C)

Von Neumann Architecture

- Enginyer que va treballar amb l'ENIAC
- Al 1945 descriu el que avui s'anomena "L'arquitectura Von Neumann"
- El model es compon de 5 parts
  - Input (E)
  - Output (S)
  - Memory (M)
  - Central arithmetic (coneguda com Unitat de Procés)
  - Central control (coneguda com Unitat de Control).



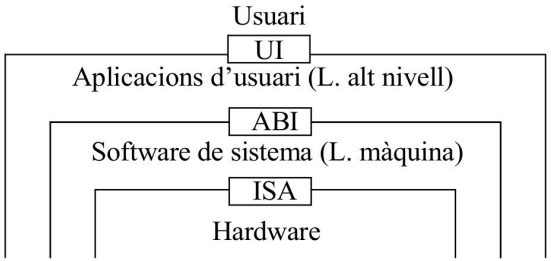
- La unitat de procés conjuntament amb la unitat de control formen el Processador (o CPU)

Tema 2
Assemblador i tipus de dades bàsics

Estructura de Computadors (EC)

Context

- Necessitem un joc d'instruccions per donar ordres al processador



- ISA: Instruction Set Architecture
  - Les instruccions que el processador implementa
  - Els tipus de dades
  - Els registres
  - L'organització de la memòria

CISC vs RISC

- Complex Instruction Set Computer (CISC)
  - Joc d'instruccions gran i complex
  - Instruccions de longitud variable
  - Cada instrucció es descodifica en múltiples micro-operacions
  - Exemples: x86

- Reduced Instruction Set Computer (RISC)
  - Joc d'instruccions petit i senzill
  - Instruccions de longitud fixa
  - Format de les instruccions i modes de direccionament senzills
  - Executades directament per el hardware
  - Exemples: **MIPS**, ARM, RISC-V

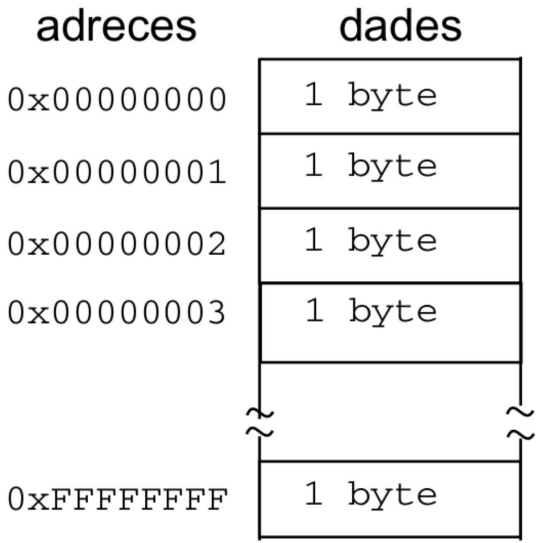
MIPS

- Microprocessor without Interlocked Pipeline Stages
  - Dissenyat entre 1981-1985 per Henessy & Patterson
  - Joc d'instruccions senzill (RISC)
  - Implementacions comercials
    - Routers i modems
    - Playstation (PSX, PS2, PSP)
    - Nintendo 64

- Ampliament usada per la docència
  - Conceptes similars a altres ISAs RISC
  - MIPS32: ISA utilitzada a EC (teoria i laboratori)

La memòria

- Vector de bytes
  - Cada byte s'identifica per una adreça de 32 bits
  - Tots els bytes de la memòria tenen una adreça associada



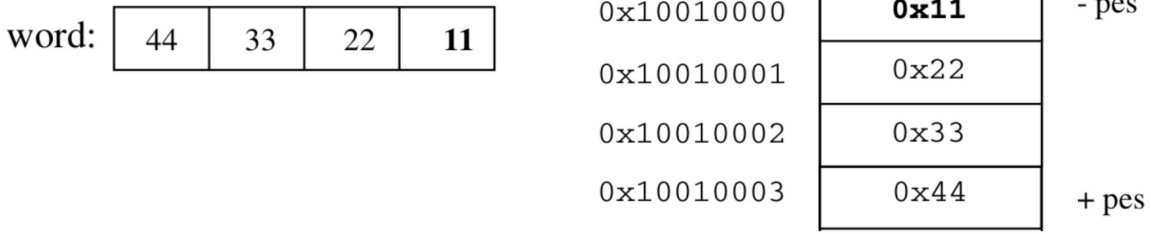
Tipus de dades: Paraula (word)

- Tipus de dades que té el tamany natiu de la arquitectura
  - MIPS32: 32 bits (4 bytes)
- Normalment: tamany paraula = tamany dels registres
- Exemples: 0xAABBCCDD, 0x44332211

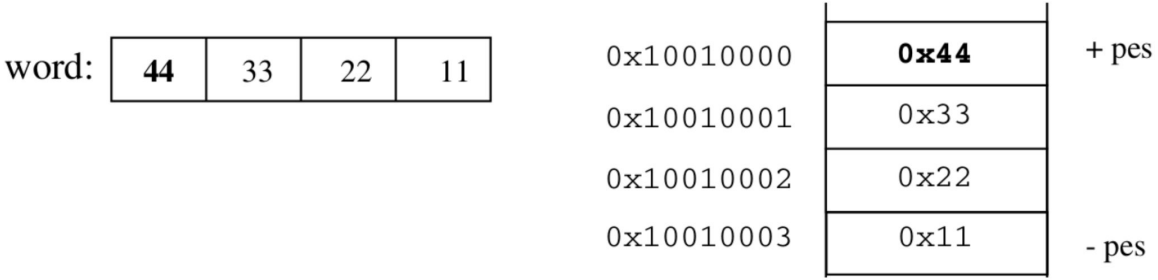
Com s'emmagatzemen els bytes d'una paraula a memòria?

Organització de la memòria: Endianness

- Little-endian: byte de **menor** pes a l'adreça més baixa



- Big-endian: byte de **major** pes a l'adreça més baixa



Declaració de variables

- Variables **globals**
  - Accessibles des de qualsevol funció/rutina
  - Emmagatzemades a memòria durant tota la execució del programa
    - A una adreça de memòria fixa

- Variables **locals**
  - Accessibles només dintre del bloc on han estat declarades
  - Es creen a l'inici de l'execució del bloc y deixen d'existir quan finalitza
  - Es reserva espai d'emmagatzematge de manera dinàmica (memòria o registre)

Exemple: Variables globals i locals

- Llenguatge C

```
int a = 0x44332211;

int main(void) {
    int i = 7;
}
```

- Assemblador MIPS

```
.data
a: .word 0x44332211

.text
main:
    li $t0, 7
```

Tamany dels tipus bàsics

Tamany	Llenguatge C	Assemblador MIPS
1 byte	char / unsigned char	.byte
2 bytes	short / unsigned short	.half
4 bytes	int / unsigned int	.word
8 bytes	long long / unsigned long long	.dword

Exemple: Declaració i alineació de variables

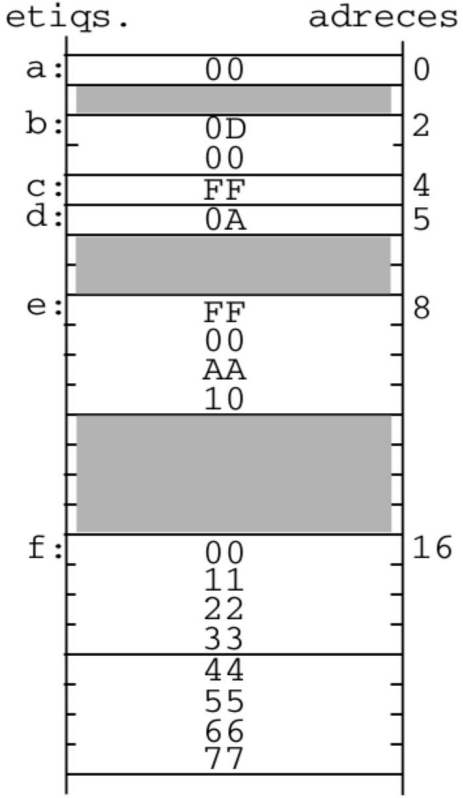
```
unsigned char a;
short b = 13;
char c = -1, d = 10;
int e = 0x10AA00FF;
long long f = 0x7766554433221100;
```

```
    .data
a:  .byte 0
b:  .half 13
c:  .byte -1
d:  .byte 10
e:  .word 0x10AA00FF
f:  .dword 0x7766554433221100
```

Exemple: Alineament en memòria

```
unsigned char a;
short b = 13;
char c = -1, d = 10;
int e = 0x10AA00FF;
long long f = 0x7766554433221100;
```

```
    .data
a:  .byte 0
b:  .half 13
c:  .byte -1
d:  .byte 10
e:  .word 0x10AA00FF
f:  .dword 0x7766554433221100
```



Exercici

Tradueix a MIPS la següent declaració de variables globals en C i indica el contingut de la memòria.

```
char a = 0xFF;
char b = 0xEE;
char c = 0xDD;
unsigned long long d = 0x7766554433221100;
short e = 0xABCD;
unsigned int f = 0x40302010;
```

Declaració de vectors

- Declaració amb inicialització

```
short vec[5] = {2, -1, 3, 5, 0};
```

```
    .data
vec: .half 2, -1, 3, 5, 0
```

- Declaració sense inicialització (alineament explicit)

```
char a;
int v[100];
```

```
    .data
a:  .byte 0
    .align 2     # Alinear a multiplo de 4
v:  .space 400   # Vector de 100 enteros
```

Resum de conceptes

- Conèixer la jerarquia de nivells de un computador
  - Nivells d'abstracció i interficies entre nivells (ABI, ISA)
- Conèixer l'arquitectura (ISA) de un procesador RISC
  - La memòria
    - Declaració de variables globals i locals
    - Alineament en memòria