

## Acesso a uma célula RAM

- I - envio de endereço da linha e coluna + ACO (Acesso à linha)
- II - pode ter vários acessos na linha mas diferentes colunas, esperar tAC ou tWR (Leitura e escrita coluna respetivamente)
- III - pre-concegar memória e esperar tRP

### Memória SDRAM

- Synchronous DRAM (flaços do relógio)
- DDR-SDRAM: Utiliza ambas os flacos do relógio.
- DDR<sub>2,3,4</sub>-SDRAM: Vários canais separados para leitura e escrita com múltiplos canais (Muitos DIMM's), alongue de banda de dados dividindo o tempo e latência entre os diferentes blocos de dados pelos sticks.

### Integridade de dados

Pode haver erros e corruptos de dados devido a falha de software ou hardware, mas ainda por mais corrompidos provocarem um bit flip.

Devido a isto é introduzido paridade e código de Hamming para proteger dados.

#### Distorcão de Hamming

XOR Bit a Bit e somar todos os 1's

XOR Fazendo

Diferente!

S1: 0010 1101

S2: 0010 0010

R: 0 000 1111

Distorcão = 1+1+1+1=4

Hamming ( $t, d$ ) -  $t = m^2$  de bits do bloco

$d = m^2$  de bits de dados

+ d = m<sup>2</sup> de bits de paridade

$$1 \text{ página} = 4096 \text{ bytes} = 4 \text{ KB}$$

MMU (Memory Management Unit) traduz endereços virtuais para físicos.

Falta de página resulta na leitura de disco (Lata!).

### Tabela de Páginas

Mutar com entradas de páginas indexadas, com o seu m<sup>3</sup> e assim se obtiver  
uma memória física o seu m<sup>2</sup> de página física.

Se não estivermos presentes estorámos SWAP

### Modos de Operação

- Modo Real
- Modo Protegido (16, 32, 64 bits)
  - Têm vários níveis de privacidade, com OS a controlar m 0, divisões m 1 e 2 e programas m 3.
  - No modo protegido é introduzido um espaço de endereçamento lógico que é visto e utilizado pelos programas.
  - Endereço lógico → Segmentação → Endereço Físico
  - 80386 (modo protegido)

Endereço lógico → Segmentação → Paginação → Endereço Físico

### Endereçamento

#### Modo Real

Modo Real  
Segmentado (Modo Real)

- 64KB (65536 bytes)
- Até 1 MiB de memória física

An. segmentado (modo protegido)

- Até 4GiB

- Memória de VM

- Endereço lógico de 16-Bits - Setores 13+1 + deslocamento 32

Bloco main

1 bloco 16 bytes 64 blocos,

$$\text{Endereço do bloco} = 12001625$$

$$\text{Nº obs} 6 \text{ blo} = 45 \cdot 64 = 11$$

### Acessos

Com insucessos de acesso a blocos:

- Still na pipeline (não é de leitura)

### Escrita na memória

#### - Write-through

- Atualiza e escreve no bloco da cache para memória principal

+ Demanda

Solução: Buffer de escrita que guarda os dados e envia na memória principal.

#### - Write-Back

- Atualiza e escreve apenas no bloco da cache e atualiza registrador dos blocos

- Caso seja necessário usar esse bloco, os dados são escritos na memória principal.

### Nível de cache

Nível 1, 2 e 3, sendo cada vez maior o tempo de acesso, e  
Sucessivamente serve os indexados da cache de cache.

### Memória Virtual

- Utiliza RAM como cache do bloco
- Cada programa recebe um espaço privado de endereçamento de memória virtual.
- CPU e OS traduzem endereços (método Virtual) em endereços físicos.
- Bloco de memória Virtual = página
- Quando um bloco de VM não está na memória física há uma falta de página (page fault)

## Tabelas de associações

- Há tabelas globais do sistema elas, que são espécies de tabelas programadas.
- Dicionários incluem informações sobre cada segmento, o seu endereço base, o seu tamanho e privilégios de acesso.

### Páginas

- Páginas de 4KB (4096 bytes) ou menos.
- Cada um deles pode conter 1 milhão de páginas, há necessidade de haver diretório de tabelas de páginas.

### Diretório de páginas

- Cada entrada endereça uma tabela de páginas.
- É uma tabela com 1024 entradas.

### Tabelas de páginas

- Cada entrada da tabela endereça uma página.
- 1024 entradas, 1MB memória.

### Cache de Endereços de páginas

- TLB - Translation Look-aside Buffer
- Cache de 16 a 512 entradas, com os endereços físicos de páginas.

## Obter(s) de um Endereço Línea

- Verificar em qual registro (atômico ou seleção).
- Obter descrição da tabela e de descriptores.
- ~~Reparar~~ Verificar se o segmento só encontra seu memória.  
Sendo selecionado, SÓ terá de conectar.
- Endereço base adicionado com deslocamento = endereço final

## Obter(s) de Endereço Físico

- Obter endereço da tabela de páginas (até da no diretório de páginas).
- Obter endereço físico da página (após a tabela de páginas).

## Interrupções

- O controle de interrupções é feito através das seguintes flags.
  - IF (Interrupt Flag)
  - TF (Trap Flag)

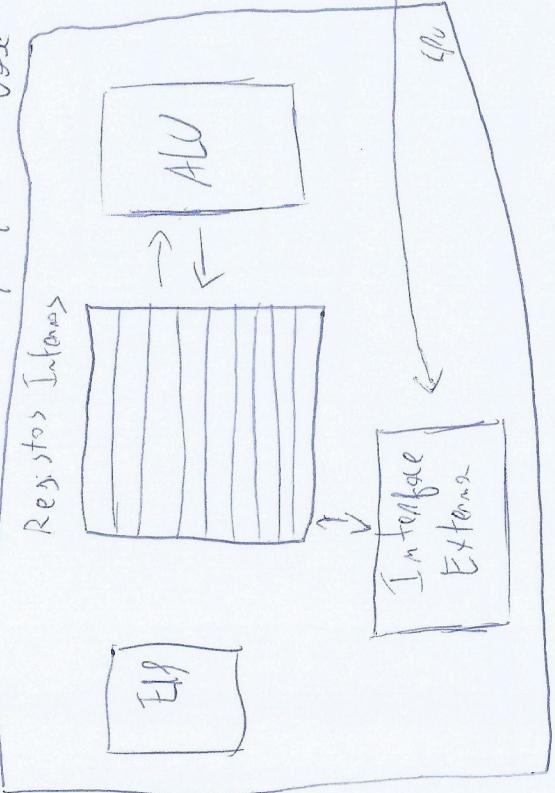
Após uma interrupção for detectada, é criado da uma tabela de descriptores de interrupções, que define o endereço da interrupção e uma resposta própria.

### Em caso de Page Fault

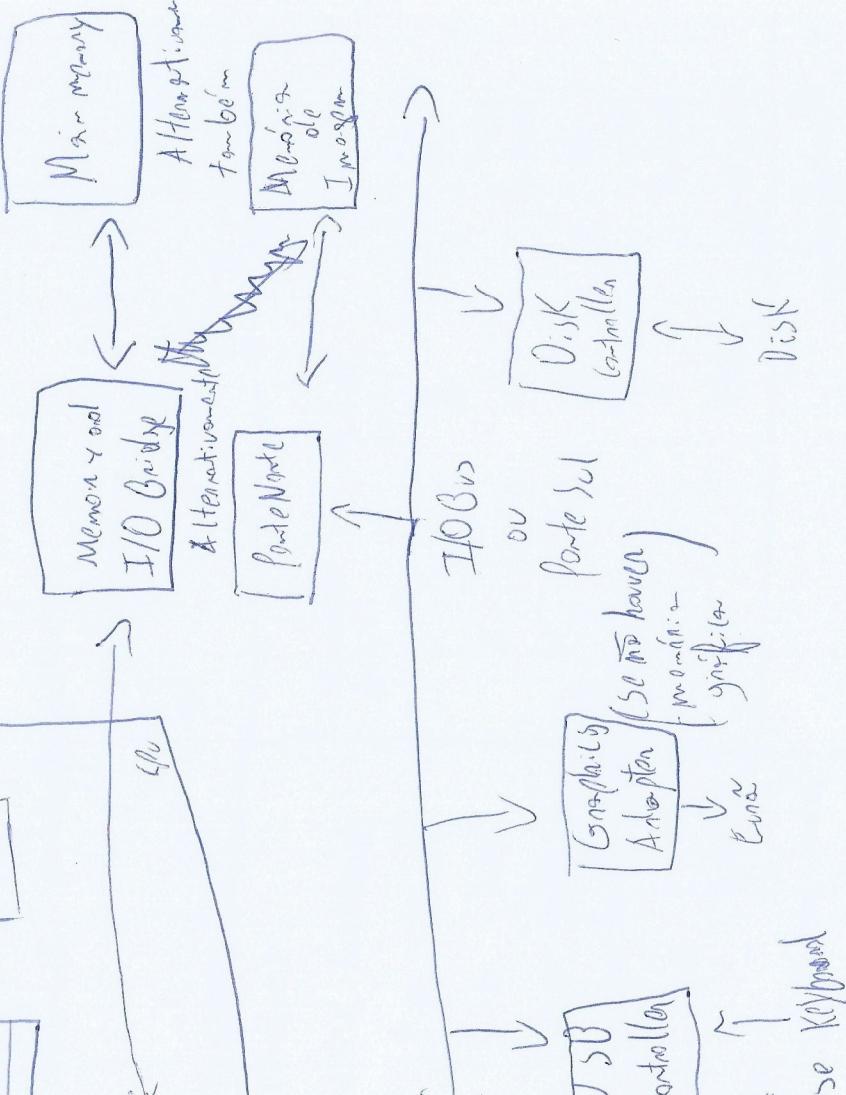
- Procura da página no Look-aside Buffer, caso não haja haverá Page Fault.  
É então produzida uma tabela de diretórios de páginas e depois o endereço é feito um varíate dele para TLB e quando é usado.

T02

## Arquitectura Box



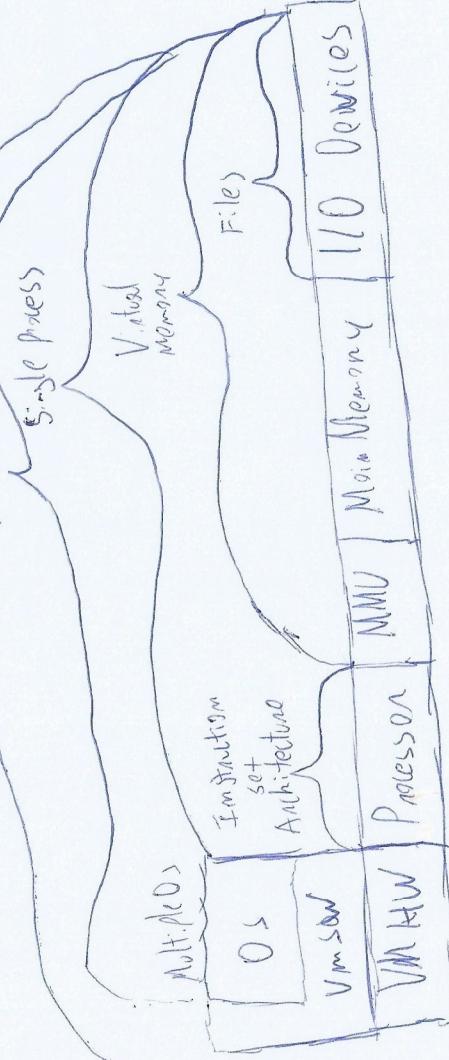
Alumnos



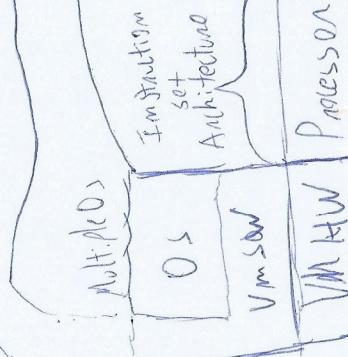
EIP → Apunta próxima instrucción o ejecutor (Extended Instruction pointer)

ALU → Arithmetic logic unit (Procesa Logicas o Aritméticas)

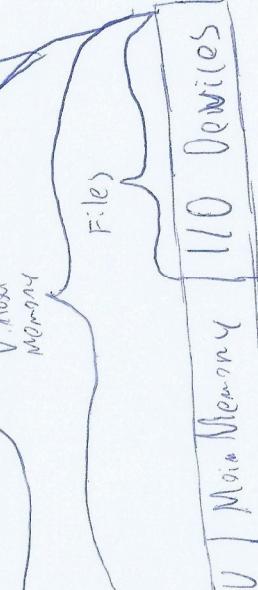
Múltiples procesos



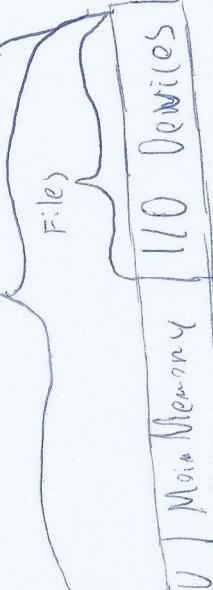
Múltiples



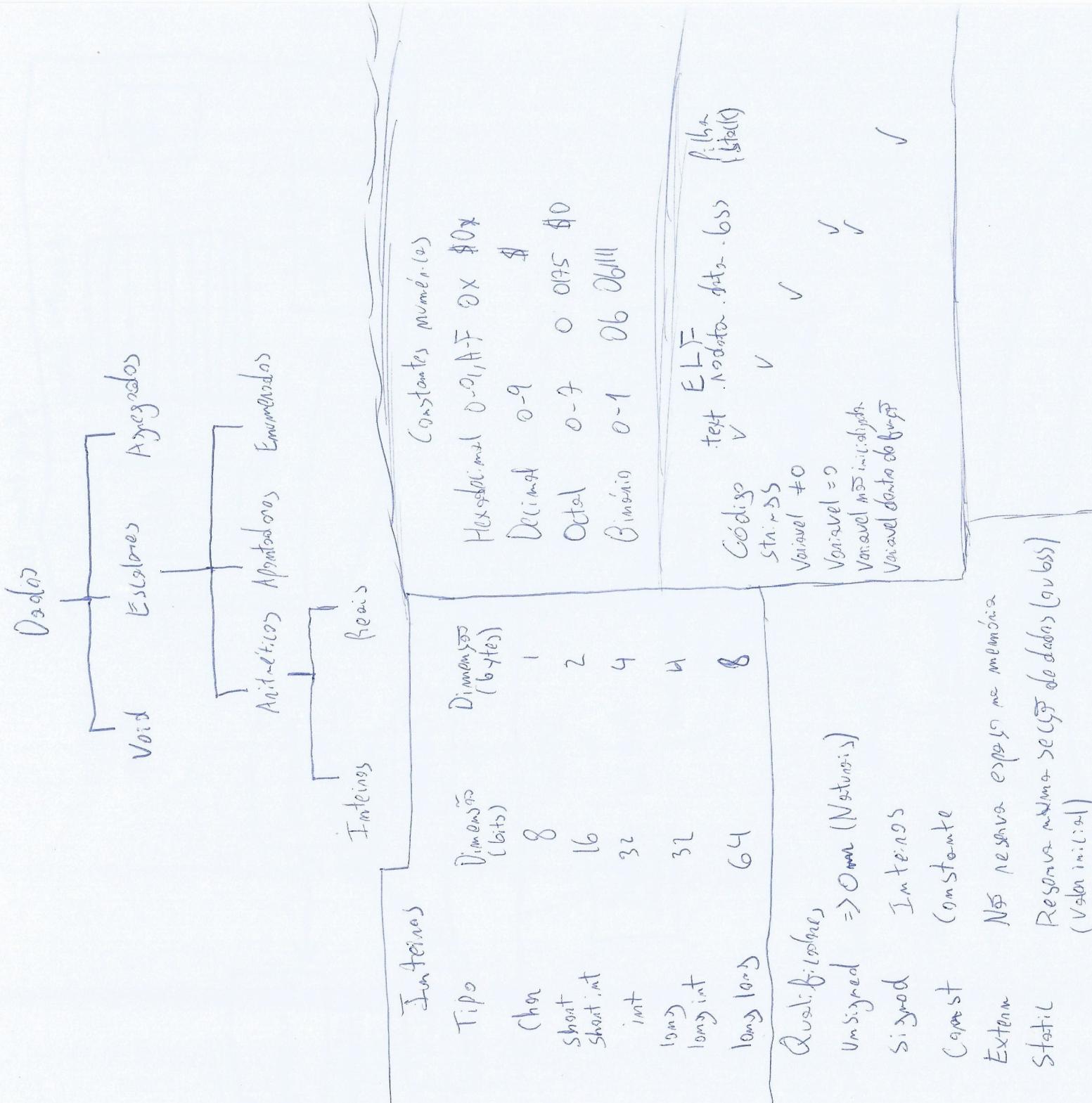
Variación



Variación



AR<sup>f</sup>-1 - C



## Flags

- CF (Carry Flag) - Transporte de 1 bit mais significativo.
- ZF (Zero Flag) - Resultado com valor zero.
- OF (Overflow Flag) - Transfere os bits de saída para o resultado.
- SF (Sign Flag) - Resultado negativo (1 1 0 0 1).
- DF (Direction Flag) - Registro de indicação de deslocamento.

## Ints

Tipo	Dimensão		Aumento de memória
	(bits)	(bytes)	
char	8	1	Desenvolvido sem penalizações no acesso à memória.
short	16	2	Exige endereços com valor múltiplo de 2.
short int			
int	32	4	Múltiplo de 4.
long	32	4	Múltiplo de 4.
long int			
long long	64	8	Múltiplo de 4.
			Modos de Endereçamento:

### Endereçamento

- Imediato (valor para registro ou memória).

- Direto:

- Registro
- Memória (endereço explícito)

- Indireto:
 

- Registro (endereço guardado num registro)
- Endereço é base

- Indiretado

### Registros vs Memória

- Registros muito mais rápidos.

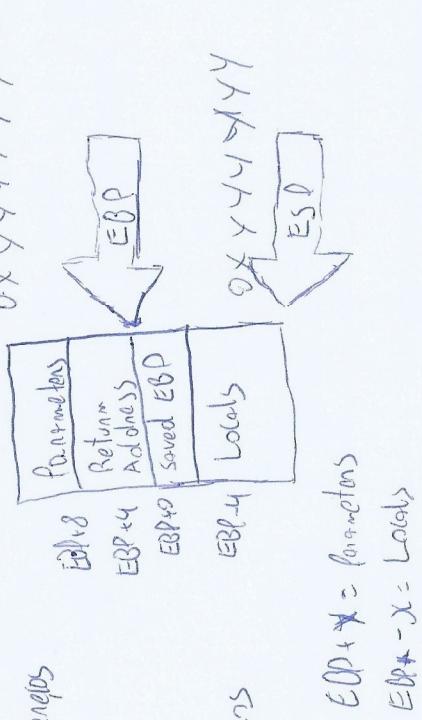
- Operações sobre dados memorizadas otimizadas.
- Com pilhas deve usar registros para variáveis.

## ARP-2(2)

### Pilha

#### LIFO (Last In First Out)

- A pilha cresce no sentido dos endereços decrescentes
- O topo da pilha é conhecido
- Pilha armazena valores e endereços



### T-03

#### ISA (Instruction Set Architecture)

Conjunto de operações, linguagem - máquina ou assembly, utilizadas pelo processador.

Diferentes CPUS = Diferentes ISA's

### Pointers

- EAX - Acumulador de operações e data de resultados
- EBX - Apontador para data no segmento DS.
- ECX - Contador de string e loop operations.
- EDX - I/O Pointer (é extenso de EAX)
- ESF - Apontador para data no segmento apontado pelo DS Register.
- EDI - Apontador para data no segmento apontado pelo DS Register.
- ESP - Stack pointer
- EBP - Apontador para data on the Stack

## T-04 Parte 1 (2)

Conversões de Binário para Octal e Hexadecimais  
Octal (Agrupamentos de 3 bits ( $2^3=8$ ))

$$\begin{array}{r}
 1011 \quad 100 \quad 110 \quad 101 \quad 011 \\
 \hline
 0 \quad 2 \quad 1 \quad 0 \quad 2 \quad 1 \quad 0 \quad 2 \quad 1 \quad 0 \\
 1 \times 2^0 + 1 \times 2^1 = 3 \quad \checkmark \quad \checkmark \quad \checkmark \\
 1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 = 5 \\
 1 \times 2^2 = 4 \quad 1 \times 2^3 + 1 \times 2^2 = 6
 \end{array}$$

Combinando: 034653

Hexadecimal (Agrupamentos de 4 bits ( $2^4=16$ ))

$$\begin{array}{r}
 1011 \quad 100 \quad 110 \quad 101 \quad 011 \\
 \hline
 3 \quad 2 \quad 1 \quad 0 \quad 3 \quad 2 \quad 1 \quad 0 \quad 3 \quad 2 \quad 1 \quad 0 \\
 1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 = 11 \\
 1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 = 10
 \end{array}$$

Combinando: B9AB

Conversão de Decimal para outras bases

Binária

$$\begin{array}{r}
 392 \quad | \quad 2 \\
 0 \quad 196 \quad | \quad 2 \\
 0 \quad 98 \quad | \quad 2 \\
 0 \quad 49 \quad | \quad 2 \\
 1 \quad 24 \quad | \quad 2 \\
 0 \quad 12 \quad | \quad 2 \\
 0 \quad 6 \quad | \quad 2 \\
 0 \quad 3 \quad | \quad 2 \\
 1 \quad 1 \quad | \quad 2 \\
 \hline
 01010010
 \end{array}$$

$$392_{10} = 1100001000_2$$

Octal

$$\begin{array}{r} 39218 \\ \times 49 \\ \hline 049 \\ 168 \\ \hline 168 \end{array}$$

↓  
PATA

$$392_{10} = 610_8$$

Hexadecimal

$$\begin{array}{r} 39216 \\ \times 16 \\ \hline 016 \\ 24 \\ \hline 016 \end{array}$$

↓  
PATA

$$392_{10} = 188_{16}$$

Float point

$$0,625_{10} = ?_2$$

$$\begin{array}{r} 1625 \\ \times 2 \\ \hline 01250 \\ \times 2 \\ \hline 01500 \\ \times 2 \\ \hline 01000 \end{array}$$

$$0,625_{10} = 0,101_2$$

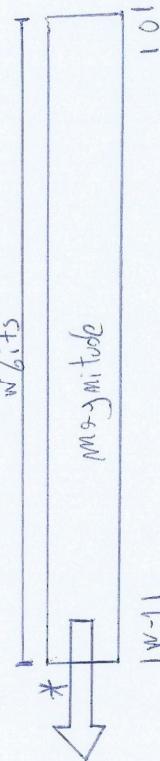
$$0,125_{10} = ?_2$$

$$\begin{array}{r} 125 \\ \times 2 \\ \hline 0125 \\ \times 2 \\ \hline 0150 \\ \times 2 \\ \hline 0100 \end{array}$$

$$0,125_{10} = 0,001_2$$

## T-04 Parte II

Números Naturais



Add

Adicção < Add (Add with borrow)

Sub

Subtração < Sub (Subtract with borrow)

Multiplicação - Mul

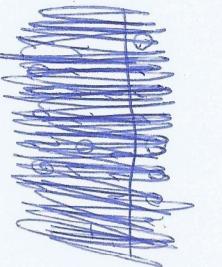
Divisão - div

Overflow

Iniciativa decimal de uma variável para representar um número  
Determinado pelo overflow de transporte do  $(w-1)$ -ésimo bit para o  $w$ -ésimo bit.  
(Transporte de 1)

SN2

Operações Aritméticas de Binário



$$\begin{array}{r}
 1010101 \\
 101111 \\
 + 11111 \\
 \hline
 10000100
 \end{array}
 \leftarrow \text{Transporte } (1+1=2=10)$$

$$\begin{array}{r}
 1010101 \\
 101111 \\
 - 111 \\
 \hline
 0110110
 \end{array}$$

## Representações de número

### ATENÇÃO:

(racional)

$$a_4 a_3 a_2 a_1 a_0 \cdot a_{-1} a_{-2} a_{-3}$$

Ou Ajustado em 4 bits

$$\boxed{a_3 a_2 a_1 a_0} \boxed{(a_0 a_1 a_2 a_3)}$$

Base N para decimal

$$\text{Número} = \sum_{i=-n}^{m-1} a_i \times 10^i$$

E.)

A 101 (Base 2)

$$\begin{array}{r} 1 \\ | \\ 1 \\ | \\ 0 \end{array}$$

$$1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 = 13$$

Conjunto de Bits

- 4 Bits: Half Byte (H)
  - 8 Bits: Byte (B)
  - 16 Bits: Word (W)
  - 32 Bits: Double-Word (DW)
  - 64 Bits: Quad Word (QW)
- Multiplos do Byte

Kilobyte ( $2^{10}$ )<sub>10</sub>

Megabyte ( $2^{20}$ )<sub>10</sub>

Gigabyte ( $2^{30}$ )<sub>10</sub>

Tebi-byte ( $2^{40}$ )<sub>10</sub>

Pebi-byte ( $2^{50}$ )<sub>10</sub>

Exabi-byte ( $2^{60}$ )<sub>10</sub>

## POP-04

Programas por módulos e compilação separada

Dado:

- 1) Um Script `modulo.c` que contém `ladochar()` calls de `funções()` para `print_hex()`, `print_bin()`
- 2) Scripts separados em 2 filhos `print_hex.c`, `print_bin.c`

É possível compilar os 2 print scripts C GCC -Wall -std=c99 -c print-x.c para ficarem objetos e depois utilizá-los no compilador do módulo.c com #GCC -Wall -std=c99 ~~funções()~~ -pedantic -O0 -o `projeto1` módulo.c `print_hex.o` ~~print\_hex.c~~ `print_bin.o`

Isso também é possível via `library`. Com os dois objetos é utilizada no comando

(nm - o) -lc libhex.a print\_hex.o  
Arquivo - ~~funções()~~ o7 -N print\_bin.o  
Lista bibliotecas - o7 -l libhex.a  
Elément - o7 -d libhex.a módulo.x

Com a biblioteca entres:

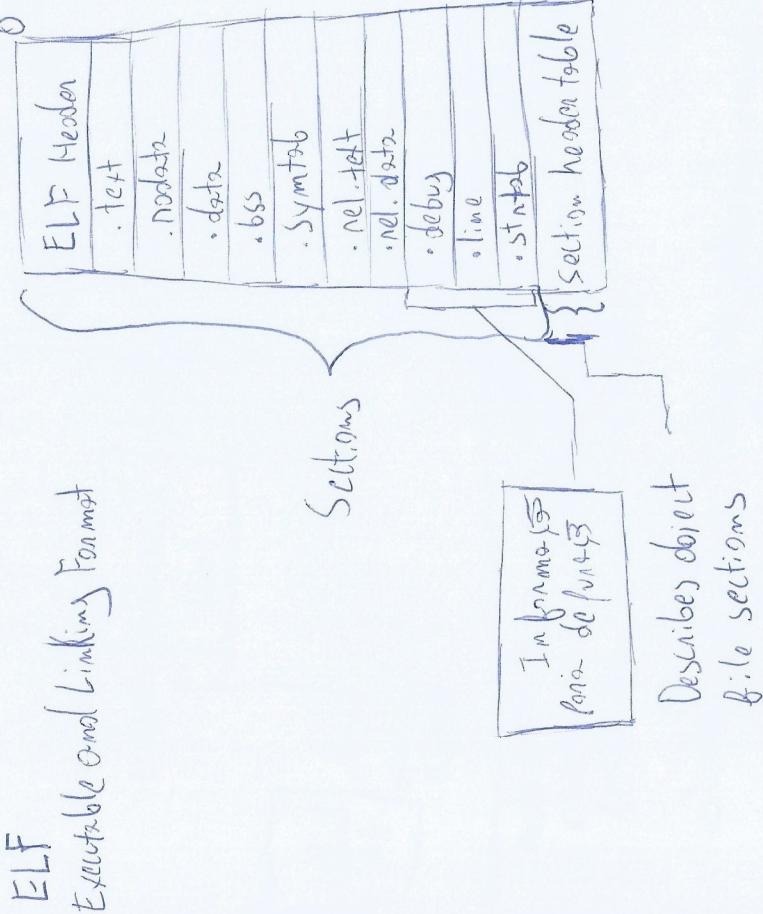
#GCC -Wall -std=c99 -pedantic -O0 -o `projeto1` módulo.c library.o

T-04 Parte 1 (1)

- Binário (Base 2)
  - Números: 0, 1
- Sistema Octal (Base 8)
  - Números: 0, 1, 2, 3, 4, 5, 6, 7
- Sistema Decimal (Base 10)
  - Números: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Sistema Hexadecimal (Base 16)
  - Números: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

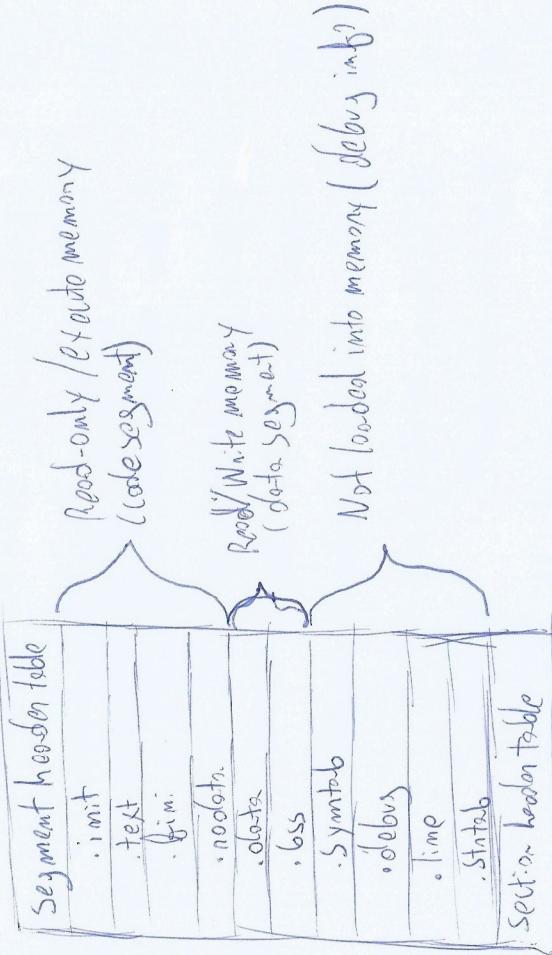
## Aula 2 (1)

Cabeçalhos de um módulo objetos



Formato de arquivos executáveis

ELF header

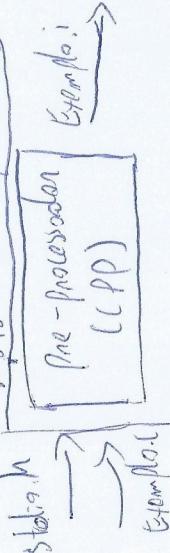


Exe.nar (from Disk to memory)

- Ler segment header para determinar formatação dos segments
- Adicionar espacos de endereçamento Virtual
- Copiar código (.text) e variáveis para a memória
- Prepara argumentos na pilha
- Inicializa registradores do processador
- Rotina de inicialização - main() — > init

## PP P-2

### Motor de Compilação da



Compilador exemplo.  
(C/C++)

Assembler exemplo.

Ficheiro fonte  
(texto)

Ficheiro Fonte  
Modificado  
(texto)

Ficheiros objetos  
Relacionados  
(Binários)

Ficheiros  
Objetos  
Relacionados  
(Binários)

Ficheiros no código de projeto e suas extensões  
(+ Comandos para obter no arquivo)

Comando

significados

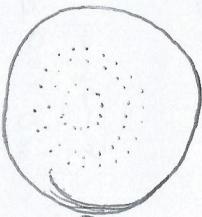
### Extensões

- .c  
Ficheiro fonte, escrito em linguagem C  
N/A
- .h  
Ficheiro de definições e protótipos intitulado por C  
-E
- .ii  
Ficheiro fonte modificado, gerado (sob codos) pelo CPP-C  
-S
- .s  
Ficheiro em linguagem assembly  
-C
- .o  
Ficheiro Objeto (um endereço relacional, pod. assembler  
N/A
- .lst  
Ficheiro listagem, gerado pelo assembler  
N/A
- .obj  
Biblioteca estática (arquivo)
- .so  
Biblioteca partilhada
- Sem Extensão  
Ficheiros executável (Windows .exe)

## T-DT Armazenamento e Contato/Solidos

### Armazenamento em disco

- Disco Rígido
  - Armazenamento não-volátil
  - Conjunto deletar/excluir que lê/sobe) magnetizado
- Disposito seletor
  - Anel de distribuição de bits (1.1)
  - Conjunto de seletores (em o mês pro Arq e file (1.1))
  - Conjunto de pistas (em o mesmo nro).



Acess & Setores é um bloco minúsculo da cabeça, p/ o fluxo de informações isoladas (leitura).

### Armazenamento Flash

- Armazenamento em Semicondutor
  - Vôô Volátil
  - 100x a 1000x + rápido
  - menor, pequeno, menor consumo e mais robusto
  - maior (nro)

### SSD

- Controlador
- memória Flash
- Cache DRAM

## RAID (Redundant Array of Independent Disks)

BY

Level 0

División de datos en N sectores (strip)

Level 1

División en 0 mas (con duplicado) o mas de strip para backup

Level 2

División de datos en nivel de bit  
Ten Hamming Code

Level 3

Mismo que 2 mas con paríodo de datos

4, 5, 6

División de bloques con periodo max 6 bloq.

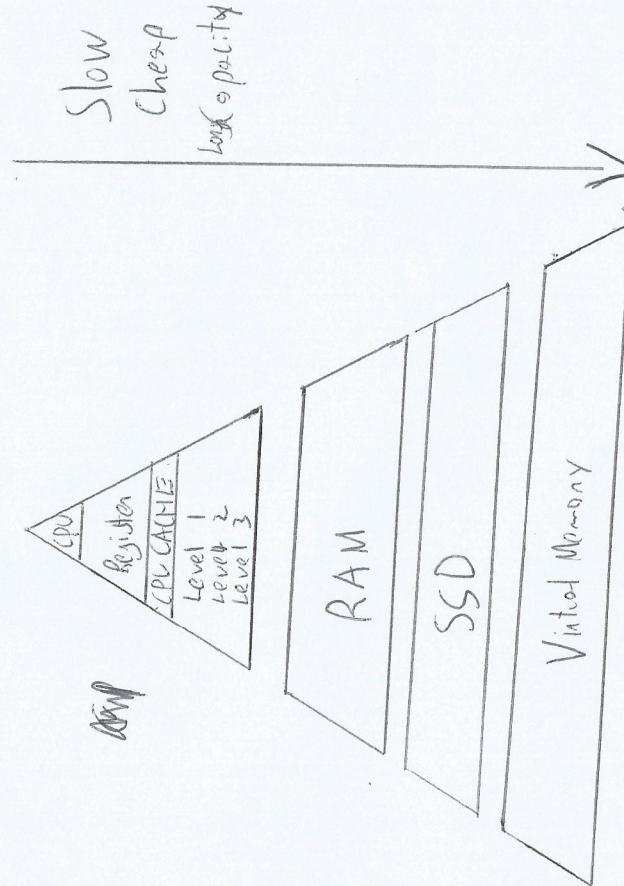
Oispositivo Els

## T-06 Memória

### Princípio da Localidade

- Num dado instante, os programas só lêem e usam poucos níveis de memória p/ gerar.
- Localidade de espaço de endereços.
- Localidade temporal ou especial
  - Localidade temporal: Se um item é referenciado, a tendência é que volte a ser referenciado de novo em breve.
  - Localidade especial: Se um item é referenciado, a tendência é que os items com endereços adjacentes sejam referenciados em breve.

### Hierarquia de Memória



### Divisão à Localidade:

- Copiar dados mais adequados para DRAM ou SRAM

### Níveis de Hierarquia de Memória

Unidade de cópia: Bloco

H.t quando os dados estão contidos no bloco, Miss quase n.s.

## Endereços de Memória

- Conjunto de células (memor quantitativo)
- Cada célula tem um nº (endereço) de referência
- células podem ter tantos nºs diferentes.

## Tecnologias de Memória

### Tipos: Volátil ou não-Volátil

Volátil - SRAM e DRAM

Não-Volátil - ROM ou PROM

### Memória Volátil

- SRAM (estática) - Bits densidade de células, não é doce, elevado consumo
- DRAM (Dinâmico) - Refreshamento de células, maior capacidade e densidade, lenta e baixa consumo.

### Sistemas de DRAM

- Bancos de Memória (4,8GB) onde se encontra os dados
- Controlada, estabelece os comandos de leitura/escrita de dados de/para vários bancos
- Comandos de endereço, dados e comando que liga os bancos ao catálogo.

### Bancos de Memória DRAM

- Compõem por linhas e colunas, endereços de memória consecutivos emlata-se na mesma linha para melhorar as hipóteses de hit.
- Linha de células flip-flop com tamanho de 8KB
- Acessos na mesma linha são escalonados primeiros dados à letaliz.

## Soma, Subtração e Subtração com IEEE

$$1,0011 \times 2^8 - 1,11 \times 2^6$$

1º passo - Por Escrita o expoente menor é menor todos os menores ordenados

$$1,11 \times 2^6 = 0,0111 \times 2^8$$

$$1,0011 \times 2^8 - 0,0111 \times 2^8$$

$$\begin{array}{r} 1,0011 \\ - 0,0111 \\ \hline 0,1100 \end{array}$$

$$0,1100 \times 2^8$$

3º passo - Não normaliza

$$1,100 \times 2^8$$

## Multiplicação e Divisão com IEEE

- 1º Adicionar ou subtrair exponents (Multiplicador +, Divisor -)
- 2º Multiplicar ou dividir mantissas
- 3º Normalizar resultado e variar signo SF, UF.
- 4º Determinar resultado final a partir dos sinais dos operandos

$$1,10011 \times 2^8 \times 1,11 \times 2^6$$

$$1^\circ \quad 8+6=14 \quad (\text{caso divisão}, -)$$

$$2^\circ \quad 1,10011 \quad (\text{caso divisão sem divisão})$$

$$\begin{array}{r} \times 1,11 \\ 1,10011 \\ 1,10011 \\ \hline 1,011100101 \end{array}$$

$$3^\circ \quad 10,110010 \times 2^{14} = 1,0110010 \times 2^{15}$$

$$4^\circ \quad + 1,0110010 \times 2^{15}$$

Através da decomposição binária, ou seja, Signed ou Unsigned 1 int, d. string -

Qual operação que usa

lhe aritmética Natural, CE serve para ver transbordo.

lhe aritmética lógica, OF serve para ver o transbordo

Aritmética Real

$$1^{\text{a}} \text{ ponto } 34,875_{10} = 100010,111_2$$

Binário

$$\begin{array}{r} 34 \frac{1}{2} \\ \times 1,875 \\ \hline 0111 \quad 2 \\ 0 \quad 4 \quad 2 \\ \times 1,750 \\ \hline 1,500 \\ \times 1,000 \\ \hline 0 \quad 1 \quad 1 \quad 2 \end{array}$$

2º Ponto

Notações

$$100010,111 \times 2^0$$

Científica

3º Ponto

Normalização

$$1,00010111 \times 2^5 \longrightarrow \text{Exp} + 127$$

$$\begin{array}{r} 4^{\text{a}} \text{ ponto Bits 1} \quad 8 \\ \text{IEEE} \quad 0 \downarrow \text{Exponente} \quad 10000100,00010111 \dots 0 \\ \text{Binário} \quad \text{Sinal} \quad \text{Parte fracionária} \\ \text{5º ponto} \quad 420B8000 \\ \text{IEEE} \quad \text{Hexa} \end{array}$$

Overflow e Underflow

Zeros	+	0	0
Infinito	+	111...1	0

No-N	+	111...1	Sequência de zeros ≠ 0
No-N	-	111...1	Sequência de zeros ≠ 0

# Impasses em condutores

Impasses impedem inicio da próxima instrução

- 1 Impasses estruturais:
  - pseudoecessão de dados
  - necessidade de obter de outras instruções
- 2 Impasses de contexto:
  - Adelistação depende da instruções anteriores.

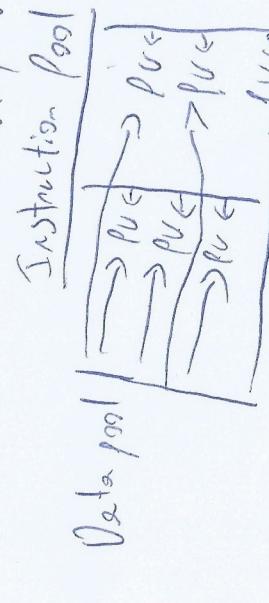
## Soluções

- 1 - Diferentes memórias/lotes
- 2 - Encaminhamento direto para write ou readback do bloco
- 3 - Previsão de saltos

## Taxonomia

- SISD (Single Instruction Single Data Stream)  


Instruction pool  
Data pool
- SIMD (Single Instruction Multiple Data Stream) processador gráfico  


Instruction pool  
Data pool 1  
Data pool 2  
Data pool n
- MIMD (Multiple Instruction Multiple Data Stream) multiprocessor  


Instruction pool 1  
Instruction pool 2  
Instruction pool n  
Data pool 1  
Data pool 2  
Data pool n

## Comutador

### 3 estágios

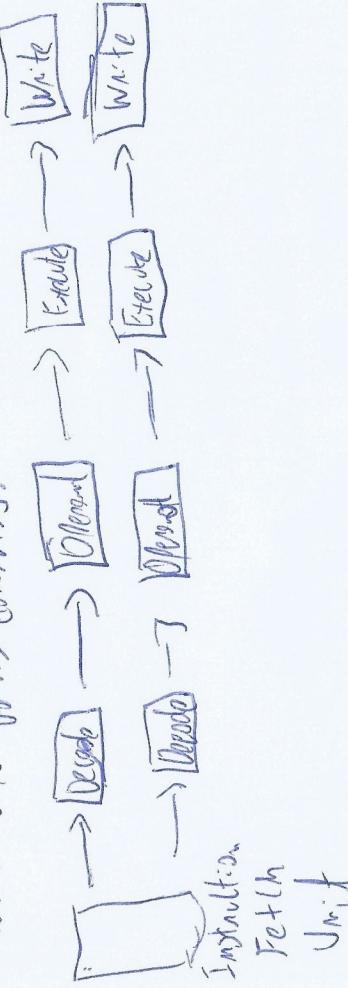
Ciclo fetch-decode-execute

Se o tempo da tarefa é dos 3 estágios não forem identicos, terá de haver delays ~~delays~~ buffers

### 5 estágios

- Ciclo Leitura, decode, leitura de operandos, execute, write  
Pode-se reforçar os condutos, o que diminui o tempo ciclo de relógio mas torna a previsão de saltos pior. Uma stall fará também um impacto maior.

Pode haver também vários condutos:



### Escolha para superescalon

Condutos longos, paralelos, diversificados, dinâmicos (retornos final de onda) permite mº de cíclos por instrução < 1

Despacho de cícle para que unidade move e executa

Com superescalon só 6 estágios

Instruction Fetch → Instruction → Buffer → Decode → Dispatch → Buffer → Complete buffer → Return

## Processador

- Realiza todos o trabalho de manipulação de dados e decisões.

### Blocos Base

- Registros
  - exx
  - edx
  - ecx
  - ebx

- ALU (Unidade Aritmética e Lógica)

- Operações Aritméticas e Lógicas

- Circuito de dados (data path)

- Suporta os Operações necessárias sobre dados e endereços.

Exemplos: ALU, registros, bancos de internos e sistemas.

- Unidade de Controles

- Controla os elementos do Circuito de Dados, entradas de comando, transmissão de dados, etc.

CISC (Complex Instruction Set Computer) vs RISC (Reduced Instruction Set Computer)

CISC

Machine Instructions

↓

Microcode Conversion

↓

MicroInstructions

↓

MicroInstructions Execution

RISC

Machine Instructions

↓

Instruction Execution

Instruções de Máquina  
São convertidas em micro-instruções que  
São executadas pelo Hardware

## RISC em Compares O-CSC

- Privilegios Operações com Registros (Fornece um número maior).
- Limita o número de operações de endereçamento mas instruções de acesso à memória.
- \* 1 ou 2 modos de endereçamento de memória.

\* Load e Store

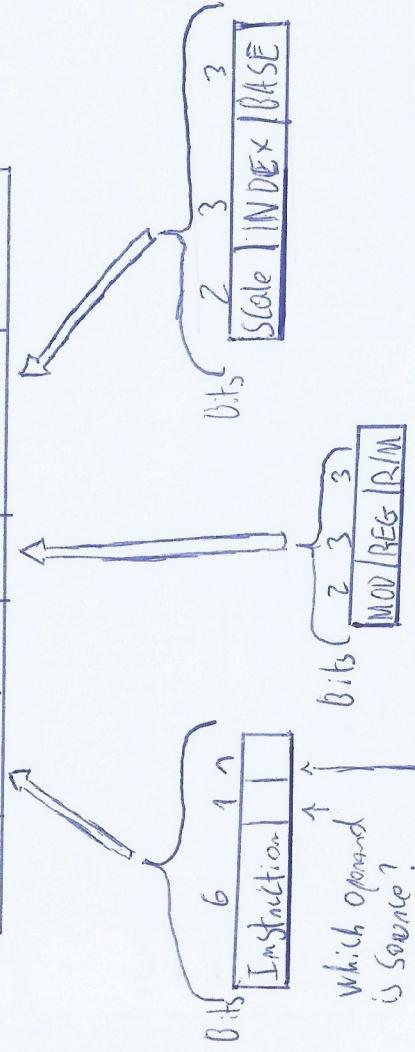
- Formatos de instruções simples e fáceis de descodificação.

## Instruções e Codificações

### CISC

- Codificação das instruções é com o tamanho variável (com campo de descodificação)
- \* Bytes de prefixo modificativo e operador
- \* Bytes de sufixo modificativo modo de endereçamento
- Bytes 0-5 1-2 0-1 0-1 0-1 0-4

Predit	OP/Code	Móde	S1B	Displacement	imediato
--------	---------	------	-----	--------------	----------



### ByteWord

### RISC

- Tamanhos fixos e formatos fixos de instruções

\* R-Format  
Operações com registos

OP - Código de operação

OP	RS	RT	RD	Shift	funct
----	----	----	----	-------	-------

Bits 6 5 5 5 5 6

\* Format

Velocidades imediatas e instruções Load/Store

OP	RS	RT	Constante ou endereço	16
----	----	----	-----------------------	----

Bits 6 5 5 5 16

RS - 1º registo fonte

RT - 2º registo fonte

RD - Registo destino

Shift - Shift value

funct - Código de função (opcode)

## Sistemas de Monitoreo e Operaciones Logísticas

## OpenOffice.org

AND (Multiplies)

$$= 0$$

一一一

110

$$0 = 0$$

四百一

OR (Beste Lernervariante pos. 1)

00 = 0

15

۱۰

卷之三

XOR (T)

XOR (Exclusiv OR) é O, diferente 1)

00 = 0

01 = 1

11

10 =

NOTICE

NOT (Contáneo)

11

## A n i m e t i c s M a t r i x N a t u r a l

Some subjects

$$\begin{array}{r}
 1100110 \\
 - 0100100 \\
 \hline
 1000010
 \end{array}$$

111-06150

110

$$\begin{array}{r}
 11091 \\
 \times 101 \\
 \hline
 11091 \\
 00000 \\
 \hline
 11091
 \end{array}$$

Digitized by

~~11/11/11~~

## Anitmet. e Intei.

### Sinal Magnitude

$$5_{10} = 00000101_2M \quad -5_{10} = 100000101$$

Em W bits, o bit w é bit do sinal, 0 = positivo, 1 = Negativo

Overflow: Soma de dois números com o mesmo Sinal

### Complemento para um (not(x))

$$5_{10} = 00000101$$

$$-5_{10} = 11111010$$

Não soma e subtração. Se sobrar bit há que fechar os bits somar ou subtrair 1 se o resultado.

### Complemento para Dois (not(x) + 1)

$$+5_{10} = 00000101_2M = 00000101CP_1 = 000000101CP_2$$

$$\begin{array}{r} -5_{10} = 11111010CP_1 = 11111011CP_2 \\ + \hline 11111011 \end{array}$$

Overflow (CP<sub>1</sub> e CP<sub>2</sub>): Soma de números (com o mesmo sinal), subtração (com Sinal diferente)

### Multiplicação em CP<sub>2</sub>

~~$$\begin{array}{r} +7 \\ \times -7 \\ \hline -49 \end{array}$$~~

~~$$\begin{array}{r} 111 \\ \times 111 \\ \hline 1001 \end{array}$$~~

~~$$\begin{array}{r} 1001 \\ \times 111 \\ \hline 1001111 \end{array}$$~~

$$\begin{array}{r} 0111 \\ \times 1001 \\ \hline 00000 \\ 00000 \\ 0111 \\ + \hline 100011 \end{array}$$

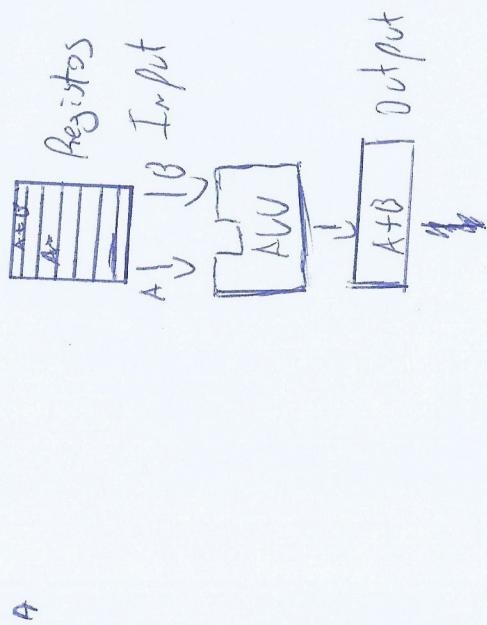
\* Extende-se os bits com bit do sinal,  
se o número de cima for negativo, estende-se com 1,  
se for positivo, estende-se com 0.

A família 4004x86 (Intel) é tradicionalmente CISC, porém a partir de 80486:

- Núcleo RISC para as instruções mais simples e mais frequentemente utilizadas
- Instruções Complexas  $\rightarrow$  CISC + XPC

## ALU

- Operações
- $X \oplus Y$
- AND, NOT, OR, XOR
- Shifts
- Pode realizar funções complexas quando fornece máscara



## Bancos de memória

Ligaçõe entre componentes.

Internas e de Sistemas.

Constituído por organizações de dados

- Bancos de Sist.
- Bancos de endereços
- Bancos de Controlos
- Relógio

## Unidade de Controlo

Ciclo Fundamental de Execução com 30 fases:

- Leitura (Fetch)
- Descodificação (Decode)
- Execução (Execute)

Este ciclo pode ser executado sequencialmente ou paralelamente

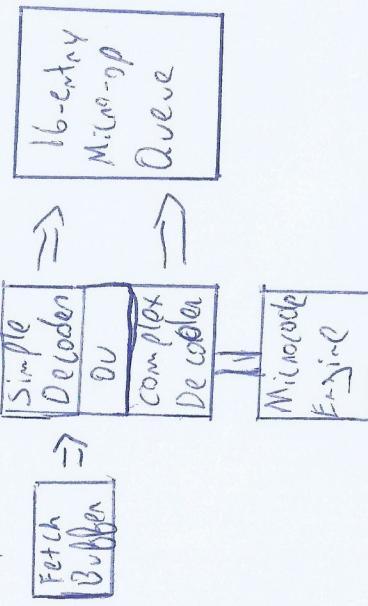
## Leitura

- Identificação endereços da próxima instrução (EIP)
- Pode involver aritmétic.
- Fetch da instrução

## Descodificação

Na descodificação há uma conversão dos bits para símbolos de controle que controlam o CPU.

Se forem instruções complexas, há ainda decodificação para milhares de dígitos e só depois para símbolos de controle.



## Execução

- ALU + Data paths
  - Cálculos aritméticos e lógicos + transferência de dados
- ALU especializada.
  - Endereços de memória para load/store
  - Endereço de memória destino de um salto
- Ajudar a execução de dadas para load/store
- atualizar EIP

Bits de paridade São potências de dois ( $1, 2, 4, 8, 16 \dots$ )  
Bit de paridade nas posições mais significativas.

### Cache de Mapas mentais Dinâmico

Posição bloco cache = (endereço do bloco) MOD (nº de blocos em cache)  
 $m^o$  do bloco =  $2^m$   
 $36\text{ bits} = 2^3 = 8$  blocos  
Feito com endereços

### Cache Associativo

- 1) Bloco pode estar em qualquer posição, necessitando de parâmetro todos os endereços e que tenha um comparsa (completamente associativo)
- 2) Há conjunto, nº do bloco indica em qual conjunto de cache pode estar, juntamente com parâmetros.

Bit de validade indica se o conteúdo está no bloco ou não.