Organização e Arquitetura de Processadores

Processador x Modelos de Memória

Baseado em slides do Prof. Dr. Ney Calazans

Prof. Dr. César Marcon

Organização x Arquitetura

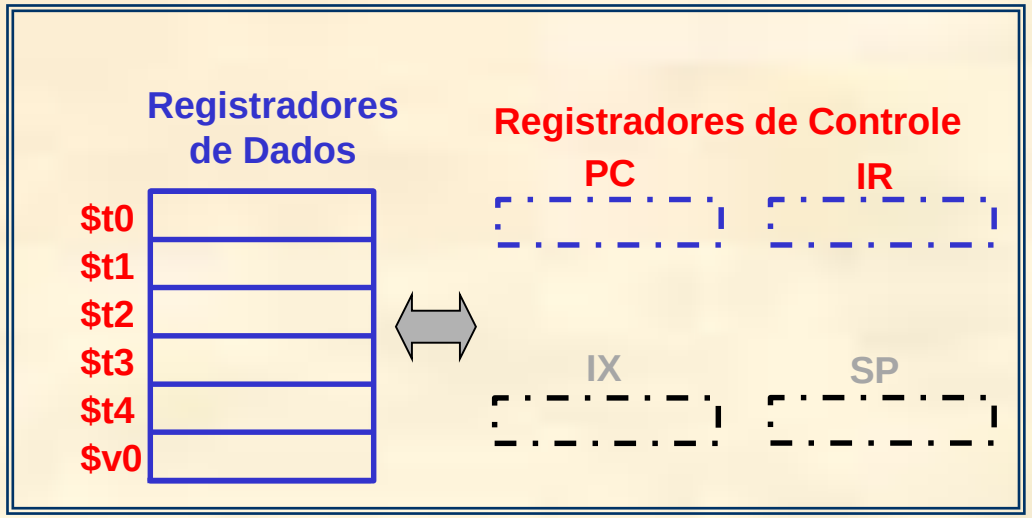
- Arquitetura é o conjunto de informações que permite ***utilizar*** o processador
  - A visão abstrata do programador de baixo nível (linguagem de montagem, em inglês *assembly language*)
  - Informações sobre, por exemplo:
    - instruções que o processador executa
    - registradores que armazenam dados

- Organização é o conjunto de informações quer permite ***implementar*** o processador
  - A visão abstrata do engenheiro (elétrico / de computação) sobre um computador
  - Informações, por exemplo:
    - Do posicionamento e interligação de portas lógicas
    - Das unidades lógico-aritméticas usadas para implementar as instruções
    - Das conexões de multiplexadores e fios

- Alguns autores utilizam os termos **arquitetura externa** para **arquitetura** e **arquitetura interna** para **organização**

Arquitetura de Processadores

Abstração do Processador



Arquitetura =
visão do programador
em linguagem de montagem

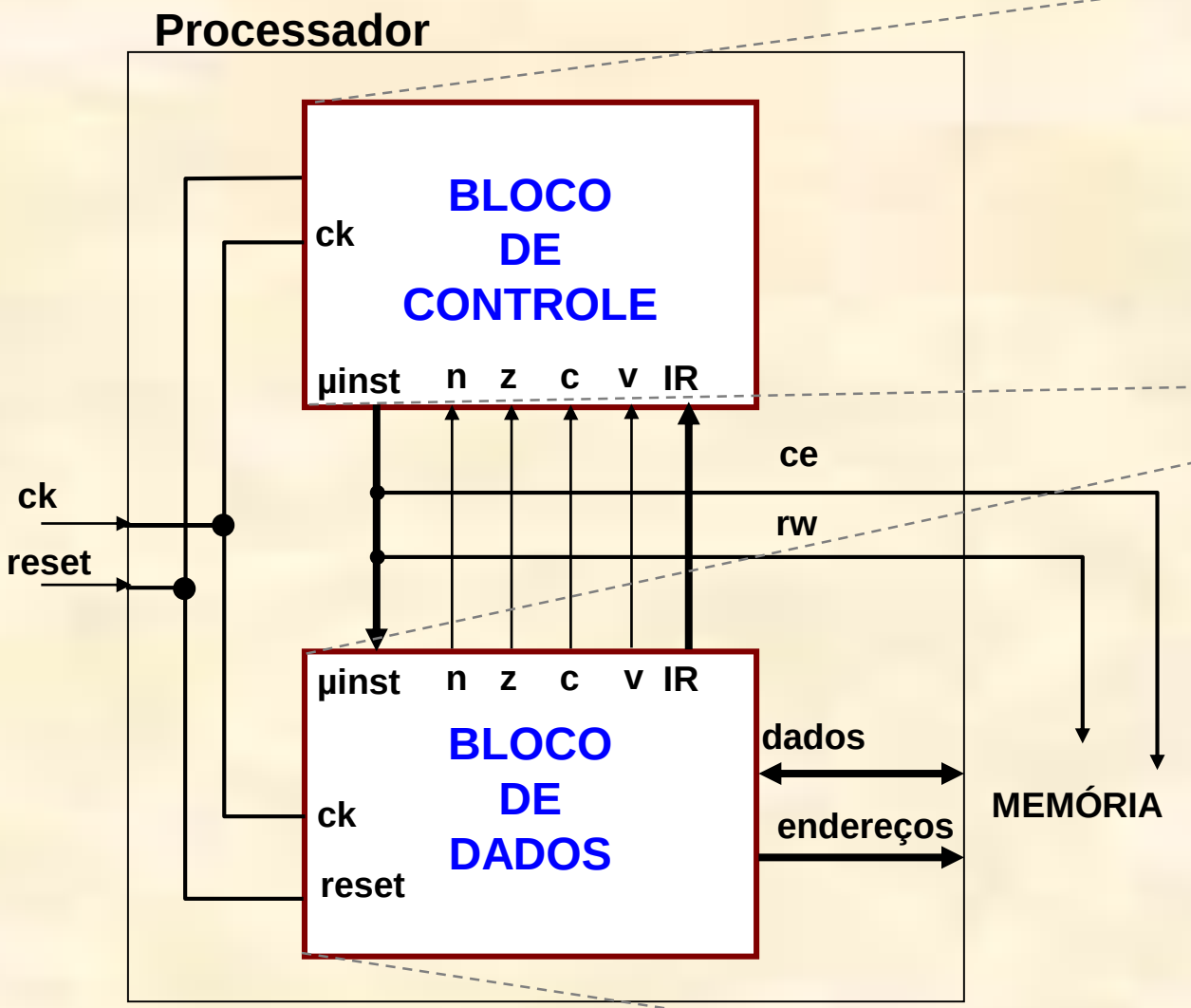
Memória de Programas

```
        .text
        .globl main
main:   la $t0,array
        la $t1,size
        lw $t1,0($t1)
        la $t2,const
        lw $t2,0($t2)
loop:   blez        $t1,end
        lw $t3,0($t0)
        addu        $t3,$t3,$t2
        sw $t3,0($t0)
        addiu       $t0,$t0,4
        addiu       $t1,$t1,-1
        j  loop
end:    li     $v0,10
        syscall
```

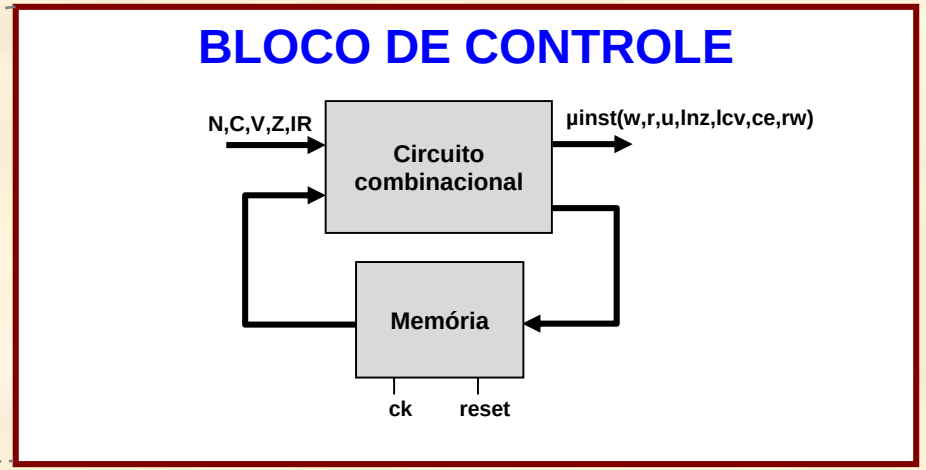
Memória de Dados

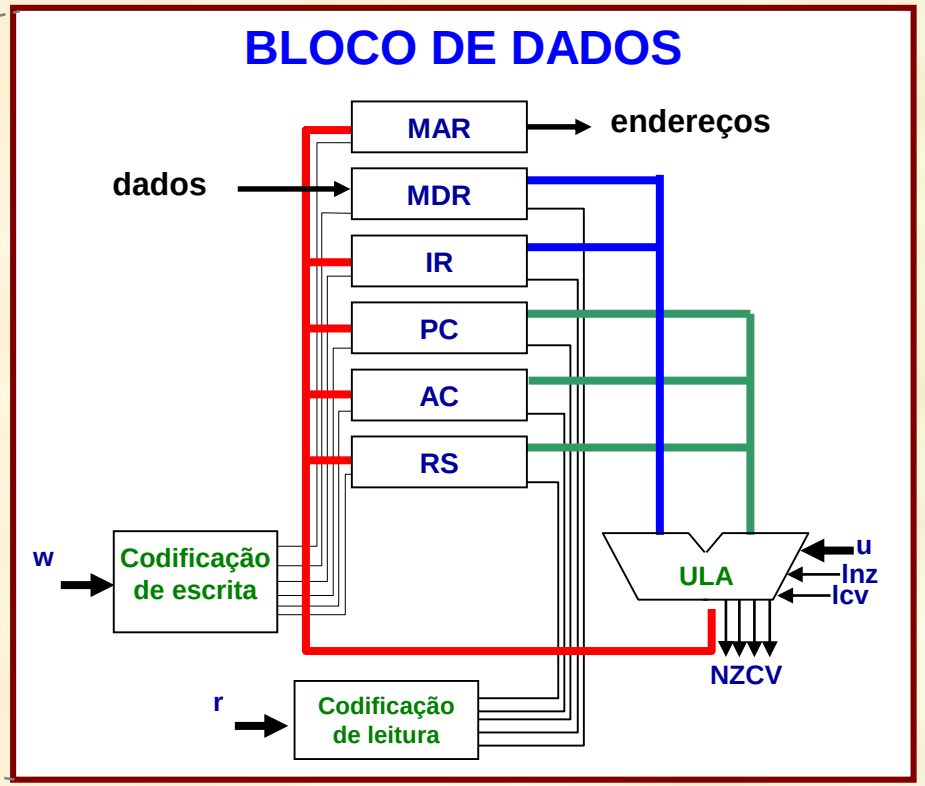
```
.data
array:   .word   12 0xff 3 14 878 31
size:    .word   6
const:   .word   0x100
```

Organização de Processadores



Organização =
visão do engenheiro elétrico / de computação





Organização x Arquitetura

- **Casos para analisar**

1. Implementação de uma instrução de multiplicação com somas sucessivas ou unidade multiplicativa. Onde entra a arquitetura e a organização?

2. Como analisar processadores x86 implementados por diversos fabricantes? É uma questão de arquitetura ou organização?

Modelos von Neumann e Harvard

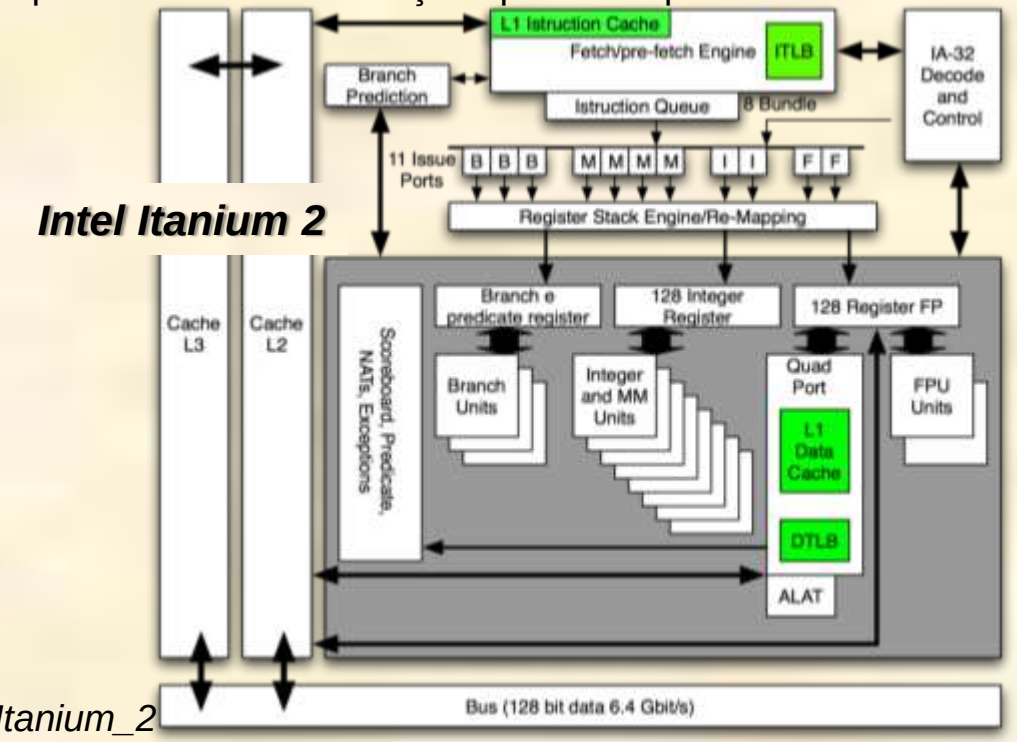
- **Existem modelos que estabelecem formas de implementar um computador sobre o ponto de vista processador/memória**

- **Estes modelos servem como classificação do processador ou hierarquia de memória**
  - Modelo von Neumann – dados e código compartilham um meio de armazenamento único
    - Portas de endereço e informação compartilhadas
      - Processador multiplexa dados e instruções
    - Mais simples
      - Processador implementa apenas uma interface de entrada e saída
    - Operação apenas de forma serial
      - Processador gera endereço de dados ou código, mas não simultaneamente; instruções que têm acesso a dados requerem mais de um ciclo de execução
    - Dissipa menos potência do processador
      - Processador executa menos tarefas simultaneamente, requerendo menos processamento instantâneo; contudo, pode ser que consuma mais energia para executar a tarefa por precisar de mais ciclos de execução
    - Modelo de memória dos primeiros computadores com memória programável

Modelos von Neumann e Harvard

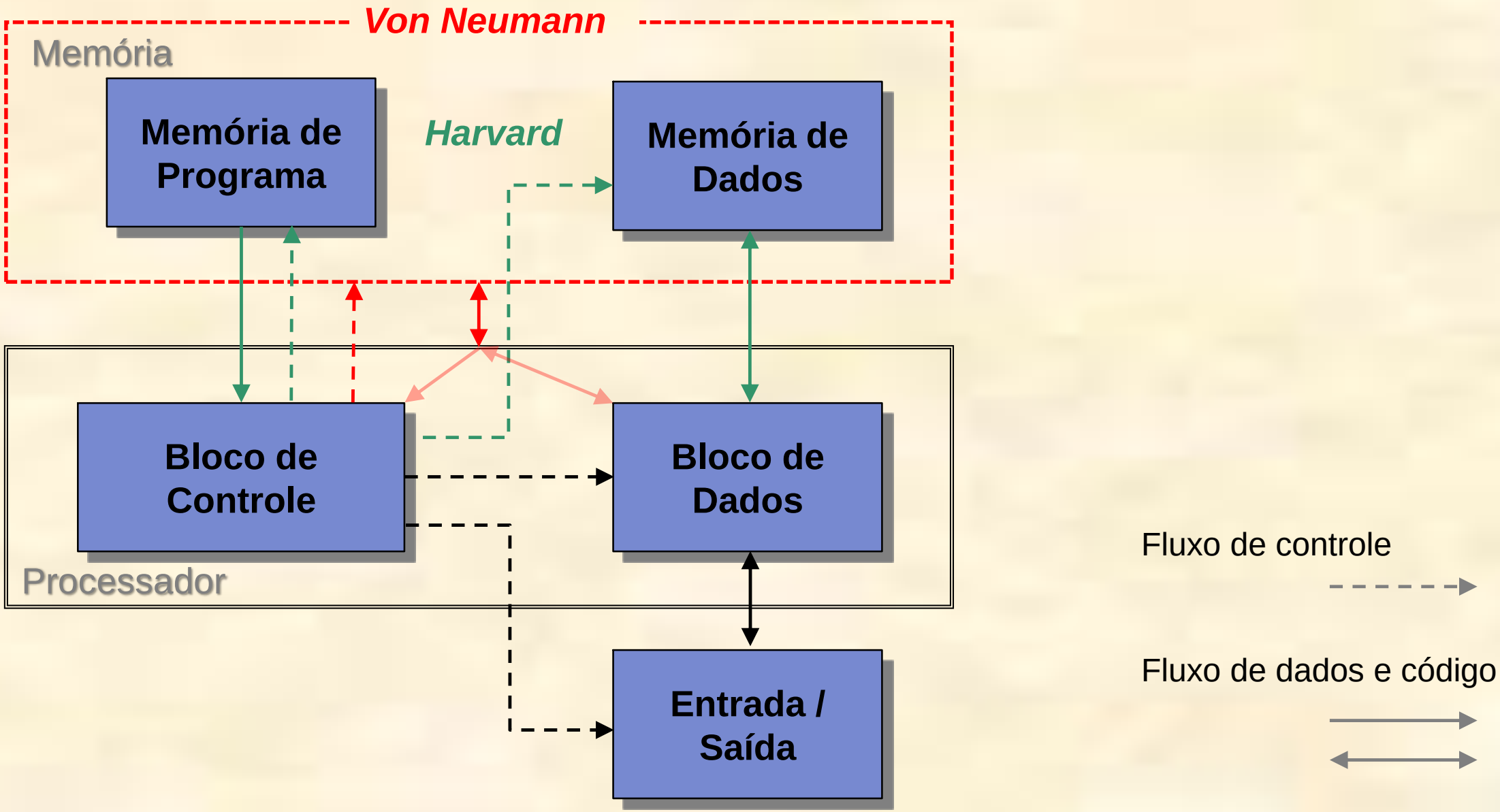
- Modelo Harvard – dados e programas mantidos em meios de armazenamento distintos
  - Mais adequado para suportar paralelismo
    - Dados e código separados permite, em paralelo, capturar uma instrução e escrever/ler um dado
      - » Leitura de uma instrução e leitura/escrita de dado de outra instrução
  - Mais complexo
    - Processador tem que implementar duas interfaces de entrada e saída e potencialmente trabalhar com elas simultaneamente
  - Maior previsibilidade de acesso à código
    - Instruções podem ou não acessar dados; com duas memórias, o tempo de acesso às instruções pode independer do acesso ao dado
  - Maior vazão de dados/instruções
  - Modelo da maioria dos sistemas computacionais modernos (desktops, celulares, câmeras digitais,…)

- **Uma hierarquia de memória pode conter ambos os modelos de memória**
  - Tipicamente os níveis próximos ao processador são modelos Harvard e os mais próximos à memória principal são von Neumann



https://it.wikipedia.org/wiki/Architettura_Itanium_2

Modelo Geral Processador + Memória + E/S



Definição Simples de um Processador

- **Um processador pode ser definido como uma máquina com capacidade de acessar meios que armazenam dados e código e operar estas informações**
  - Dados são as informações a serem processadas
  - Código contém as informações de como processar os dados

- **Código contém sequências de instruções reconhecidas pelo processador**

- **O funcionamento básico de um processador é repetir infinitamente a sequência de 3 ações:**
  1. Buscar instrução
  2. Identificar a instrução buscada
  3. Executar a instrução buscada

- **A execução de instruções pode incorrer em acesso a memórias e dispositivos de entrada e saída (eventualmente, também mapeados em memória)**

Definição Precisa da Arquitetura de um Processador

- **Arquitetura do Conjunto de Instruções (do inglês, Instruction Set Architecture ou ISA) define a funcionalidade de um processador**
- **Uma definição completa e unívoca de uma arquitetura de um processador pode ser alcançada por um ISA que contempla seis elementos:**

Definição Precisa da Arquitetura de um Processador

- **Arquitetura do Conjunto de Instruções (do inglês, Instruction Set Architecture ou ISA) define a funcionalidade de um processador**
- **Uma definição completa e unívoca de uma arquitetura de um processador pode ser alcançada por um ISA que contempla seis elementos:**
  1. Linguagem de Montagem (AL) – Conjunto de regras léxicas, sintáticas e semânticas que permite descrever textualmente programas executáveis no processador alvo

```
.text
.globl main
                    ...
.main:
        MOVE    $a1, $v0
                    ...
        LA      $a0, $TxtFact
        LI      $v0, 4
        SYSCALL
                    ...

.rdata
$TxtFact:
        .ascii “O fatorial de 10 é:”
                    ...
```

Definição Precisa da Arquitetura de um Processador

- **Arquitetura do Conjunto de Instruções (do inglês, Instruction Set Architecture ou ISA) define a funcionalidade de um processador**
- **Uma definição completa e unívoca de uma arquitetura de um processador pode ser alcançada por um ISA que contempla seis elementos:**
  1. Linguagem de Montagem (AL) – Conjunto de regras léxicas, sintáticas e semânticas que permite descrever textualmente programas executáveis no processador alvo
  2. Conjunto de Registradores (R) – Elementos de memória internos ao processador acessíveis ao programador através da AL da ISA

Número	Nome	Significado
0	$zero	constante 0
1	$at	reservado para o assembly
2, 3	$v0, $v1	resultado de função
4 - 7	$a0 - $a3	argumento para função
8 - 15	$t0 - $t7	temporário
16 - 23	$s0 - $s7	temporário (salvo nas chamadas de função)
24, 25	$t8, $t9	temporário
26, 27	$k0, $k1	reservado para o SO
28	$gp	apontador de área global
29	$sp	stack pointer
30	$fp	frame pointer
31	$ra	registrador de endereço de retorno

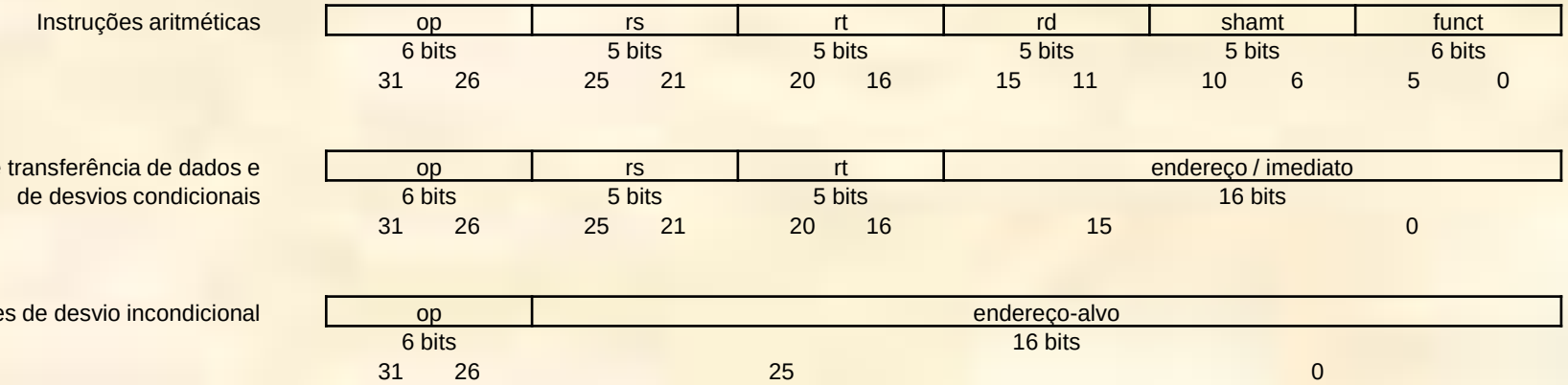
Definição Precisa da Arquitetura de um Processador

- **Arquitetura do Conjunto de Instruções (do inglês, Instruction Set Architecture ou ISA) define a funcionalidade de um processador**
- **Uma definição completa e unívoca de uma arquitetura de um processador pode ser alcançada por um ISA que contempla seis elementos:**
  1. Linguagem de Montagem (AL) – Conjunto de regras léxicas, sintáticas e semânticas que permite descrever textualmente programas executáveis no processador alvo
  2. Conjunto de Registradores (R) – Elementos de memória internos ao processador acessíveis ao programador através da AL da ISA
  3. Conjunto de Instruções (I) – Todas as instruções que fazem parte do alfabeto de entrada do processador e implementam o código

Instrução	Operação
LA rDest, endereço	rDest ← endereço
LI rDest, imm	rDest ← imm
LB rt, endereço (byte) e LW rt, endereço (word)	rt ← M[endereço]
SB rt, endereço (byte) e SW rt, endereço (word)	M[endereço] ← rt
MOVE rDest, Rsrc	rDest ← Rsrc
ADD rd, rs, rt	rd ← rs + rt
SUB rd, rs, rt	rd ← rs - rt
OR rd, rs, rt	rd ← rs OR rt
SLT rd, rs, rt	Se rs menor que rt rd ← 1, senão rd ← 0
J endereço	$pc ← endereço (endereço de 26 bits + 2 de deslocamento = 2^{28})
JAL endereço (chamada de função)	$ra ← $pc + 4 e $pc ← endereço
J $ra (retorno de função)	$pc ← $ra (Salto a registrador = 2^{32})
BEQ rs, rt, label (label tem 16 bits, permite salto de +/- 2^{15})	Se igual então $pc ← label, senão $pc ← $pc + 4
BGT rs, rt, label	Se maior então $pc ← label, senão $pc ← $pc + 4
BGEZ rs, label	Se maior ou igual a zero $pc ← label, senão $pc ← $pc + 4
BGTZ rs, label	Se maior que zero $pc ← label, senão $pc ← $pc + 4

Definição Precisa da Arquitetura de um Processador

- **Arquitetura do Conjunto de Instruções (do inglês, Instruction Set Architecture ou ISA) define a funcionalidade de um processador**
- **Uma definição completa e unívoca de uma arquitetura de um processador pode ser alcançada por um ISA que contempla seis elementos:**
  1. Linguagem de Montagem (AL) – Conjunto de regras léxicas, sintáticas e semânticas que permite descrever textualmente programas executáveis no processador alvo
  2. Conjunto de Registradores (R) – Elementos de memória internos ao processador acessíveis ao programador através da AL da ISA
  3. Conjunto de Instruções (I) – Todas as instruções que fazem parte do alfabeto de entrada do processador e implementam o código
  4. Formato de Instrução (F) – Regra de organização física e lógica dos bits de I



Definição Precisa da Arquitetura de um Processador

- **Arquitetura do Conjunto de Instruções (do inglês, Instruction Set Architecture ou ISA) define a funcionalidade de um processador**
- **Uma definição completa e unívoca de uma arquitetura de um processador pode ser alcançada por um ISA que contempla seis elementos:**
  1. Linguagem de Montagem (AL) – Conjunto de regras léxicas, sintáticas e semânticas que permite descrever textualmente programas executáveis no processador alvo
  2. Conjunto de Registradores (R) – Elementos de memória internos ao processador acessíveis ao programador através da AL da ISA
  3. Conjunto de Instruções (I) – Todas as instruções que fazem parte do alfabeto de entrada do processador e implementam o código
  4. Formato de Instrução (F) – Regra de organização física e lógica dos bits de I
  5. Modo de Endereçamento (AM) – Forma de especificar operandos de I para cada F, juntamente com a forma de alcançar os operandos

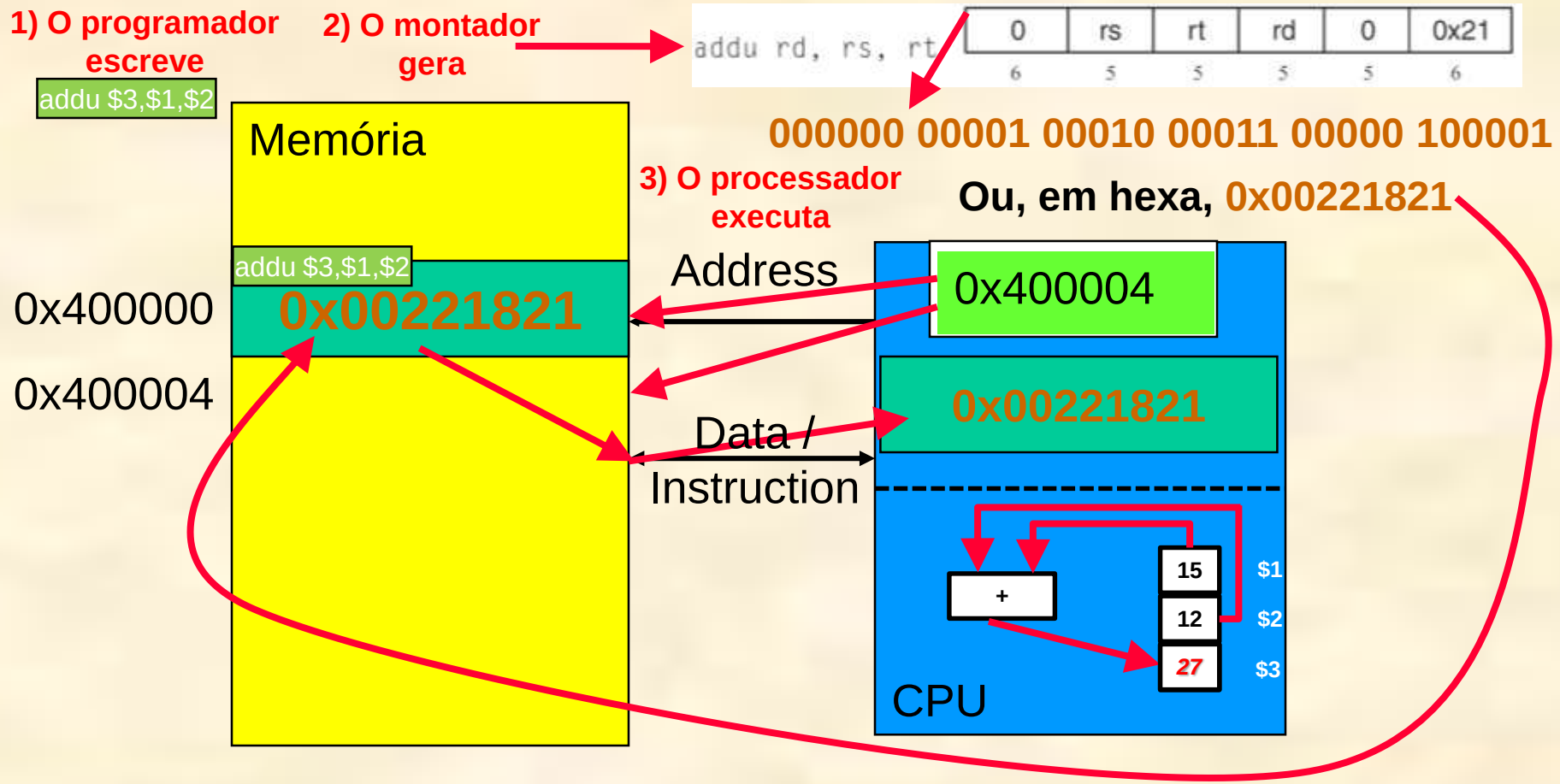
ADD rd, rs, rt	rd ← rs + rt
ADDIU rd, rs, CTE	rd ← rs + CTE

Definição Precisa da Arquitetura de um Processador

- **Arquitetura do Conjunto de Instruções (do inglês, Instruction Set Architecture ou ISA) define a funcionalidade de um processador**
- **Uma definição completa e unívoca de uma arquitetura de um processador pode ser alcançada por um ISA que contempla seis elementos:**
  1. Linguagem de Montagem (AL) – Conjunto de regras léxicas, sintáticas e semânticas que permite descrever textualmente programas executáveis no processador alvo
  2. Conjunto de Registradores (R) – Elementos de memória internos ao processador acessíveis ao programador através da AL da ISA
  3. Conjunto de Instruções (I) – Todas as instruções que fazem parte do alfabeto de entrada do processador e implementam o código
  4. Formato de Instrução (F) – Regra de organização física e lógica dos bits de I
  5. Modo de Endereçamento (AM) – Forma de especificar operandos de I para cada F, juntamente com a forma de alcançar os operandos
  6. Modelo de Acesso à Memória (M) – Regras que o ISA usa para interagir com o ambiente onde o processador está inserido

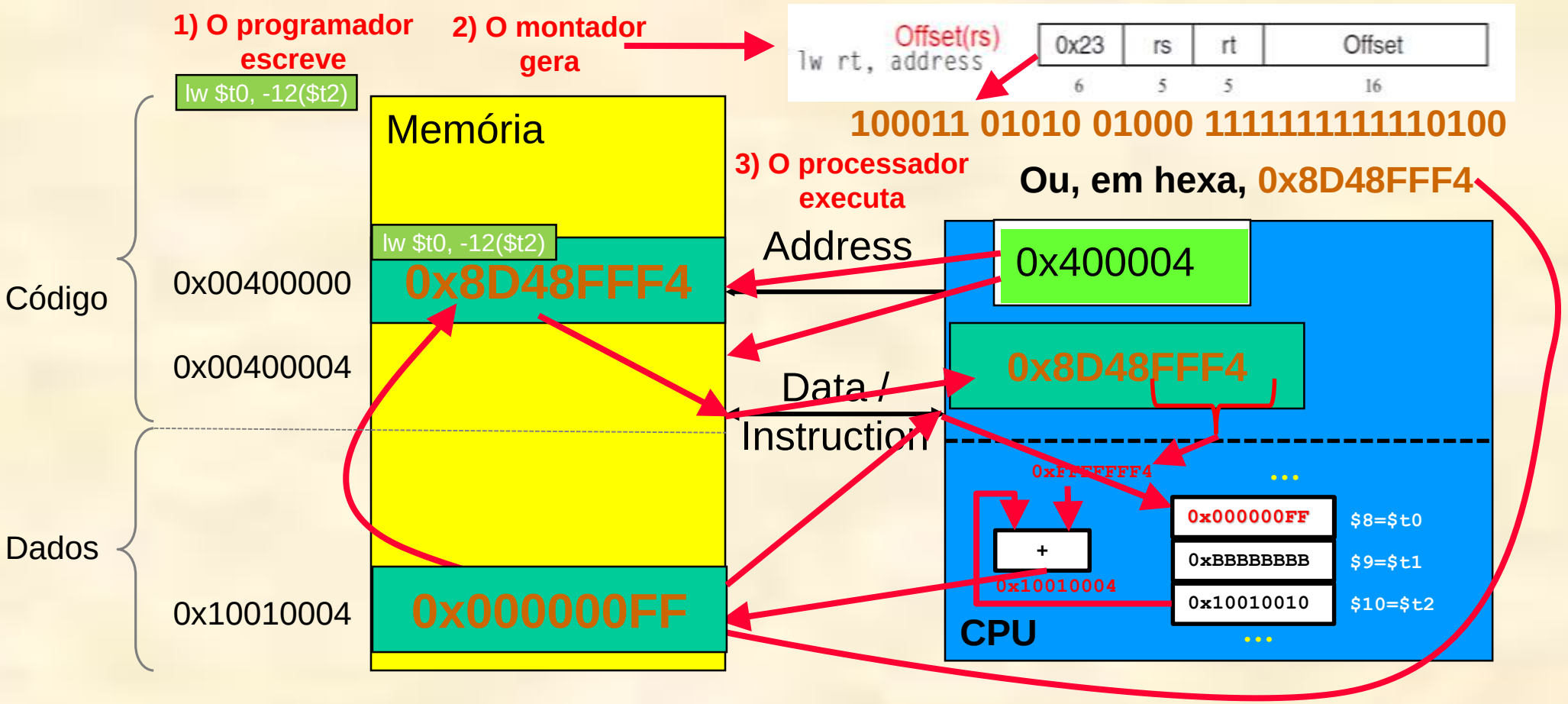
Busca, Decodificação e Execução de uma Instrução (MIPS) - 1

Interface Processador/Memória com modelo von Neumann
Exemplo de execução completa de uma instrução



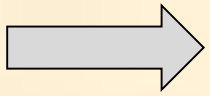
Busca, Decodificação e Execução de uma Instrução (MIPS) - 2

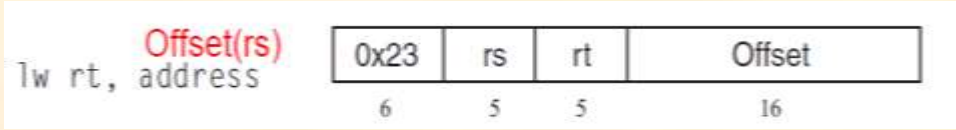
Modelo von Neumann: Exemplo de execução completa de instrução que faz acesso à memória de dados



Entendendo a Codificação da Instrução

```
lw $t0, -12($t2)
```






```
Código   RS    RT        Offset
100011 01010 01000 1111111111110100   ➔
```

```
Em Binário (agrupados em 4 bits)
1000 1101 0100 1000 1111 1111 1111 0100
Em Hexadecimal (conforme agrupamento)
   8    D    4    8    F    F    F    4
```

```
100011: 0010 0011
           2    3 -> 0x23 = 35

01010:  0000 1010
           0    A -> 0x0A = 10

01000:  0000 1000
           0    8 -> 0x08 =  8

1111111111110100: 1111 1111 1111 0100
                     F    F    F    4 -> 0xFFF4
                                    -> 0xFFFFFFF4 = -12 (extensão de sinal)
```

Busca, Decodificação e Execução de uma Instrução (MIPS) - 3

Modelo Harvard: Exemplo de execução completa de instrução que faz acesso à memória de dados