

# **Histórico da Evolução dos Computadores**

[Introdução à Organização de Computadores]

Adriano J. Holanda

31 de agosto de 2020

# Introdução

## Introdução

Charles Babbage e a máquina analítica

Alan Turing e a máquina de Turing

Z3

ENIAC

Arquitetura de von Neumann

Mark 1

## Texto

Padrão ASCII

Padrão Unicode

## Organização

O Processador

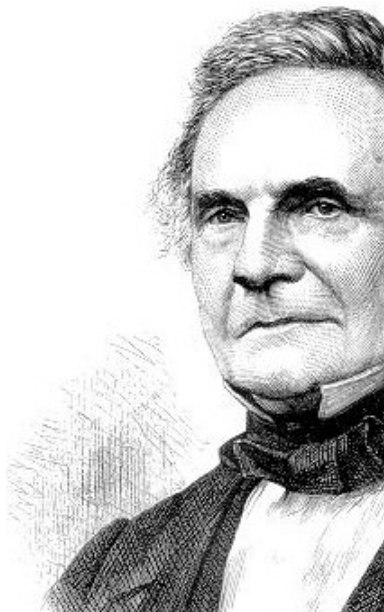
Arquiteturas: RISC x CISC

- ▶ História da evolução do computador

# Charles Babbage

1791–1871

- ▶ Matemático, Filósofo, Inventor e Engenheiro Mecânico.
- ▶ Projetou a “Máquina Análítica”, “computador” mecânico de propósito geral que possuía:
  - ▶ unidade aritmética
  - ▶ controle de fluxo
  - ▶ desvio condicional
  - ▶ laços
  - ▶ memória integrada com funções
- ▶ Primeira descrição da máquina: **1837**.

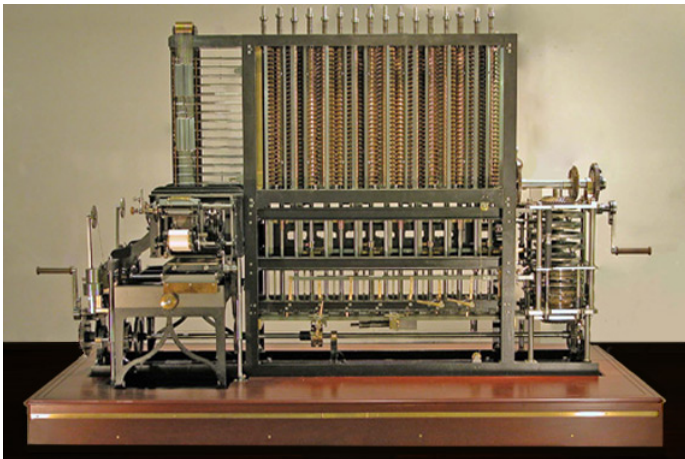


# Máquina Análitica

## Máquina diferencial nº 1

No começo de 1820, Babbage trabalhou em um protótipo da máquina diferencial nº 1, que apesar de contar com recursos financeiros nunca foi terminada. Algumas partes estão no Museu da História da Ciência em Oxford.

Abaixo o fragmento aperfeiçoado por Henry Babbage (filho de C. Babbage) em 1910 que se encontra no Museu da Ciência em Londres.



# Máquina Análitica

## Descrição

### Máquina diferencial nº 1

- ▶ Componentes: 25.000;
- ▶ Peso: 13,6 toneladas;
- ▶ Altura: 2,5 metros;
- ▶ Nunca foi terminada.

A Máquina diferencial nº 2, projetada por Babbage posteriormente, foi reconstruída em 1991 com a tolerância e materiais disponíveis na época do projeto e em testes realizou cálculos com até 31 dígitos, número maior que a maioria das calculadoras modernas.

# Alan Turing

1912–1954

- ▶ Matemático, Logicista, “Criptoanalista” e Cientista da Computação;
- ▶ Idealizou um dispositivo computacional no clássico artigo “On Computable Numbers, with an Application to the Entscheidungsproblem” (1937) para analisar as limitações sobre o que pode ser computado.



# Máquina de Turing

Há somente 6 operações atômicas que uma máquina de Turing realiza no decorrer de um computação:

1. lê (identifica) o símbolo sobre o cabeçote
2. escreve um símbolo no quadrado abaixo do cabeçote (depois de apagar o símbolo existente, se houver)
3. move a fita para o quadrado esquerdo
4. move a fita para o quadrado direito
5. muda o estado
6. para (*halt*)



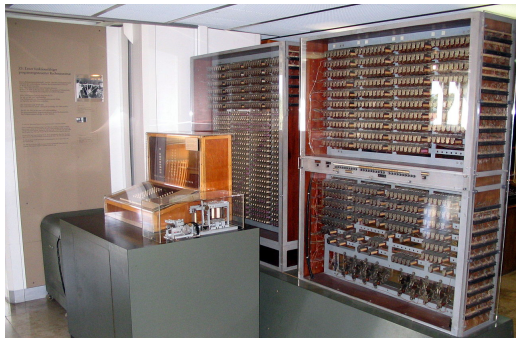
# Máquina de Turing

vídeo

[Vídeo no YouTube](#)



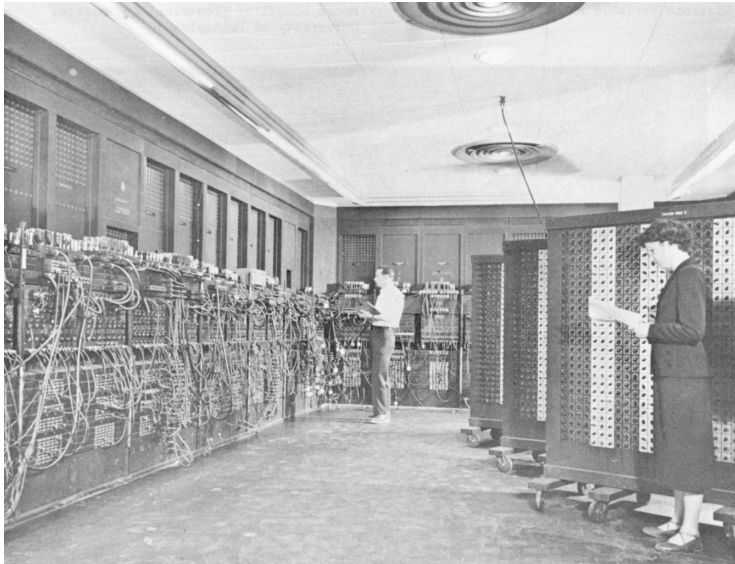
- ▶ Primeiro computador eletromecânico projetado por **Konrad Zuse** (1910–1995) programável e totalmente automático;
- ▶ Terminado em **1941** e usado pela Força Aérea Alemã para realizar análises estatísticas de fluxo de vento nas asas de aviões;
- ▶ Contruída com 2.000 relês, palavra de 20 bits e frequência de clock entre 5–10 Hz;
- ▶ O original foi destruído em 1943 durante um bombardeio dos Aliados em Berlin.



← Réplica do Z3 no Museu Deutsches em Munique

# ENIAC

## Electronic Numerical Integrator And Computer



# Projeto ENIAC

1943-1946

Produzido com o objetivo de realizar complexos cálculos de balística que passaram a ser realizados em 30 segundos, ao invés das 12 horas que costumavam demorar com as calculadoras manuais.

## Alguns dados:

- ▶ Projetistas: [John Mauchly](#) and [J. Presper Eckert](#);
- ▶ Pesava mais de 30 toneladas;
- ▶ Operava na base 10 e não em binária;
- ▶ Ocupava 270 m<sup>2</sup>;
- ▶ Realizava 5.000 operações por segundo;
- ▶ Possuía 18.000 válvulas de 160 KW cada;
- ▶ Utilizava base 10.

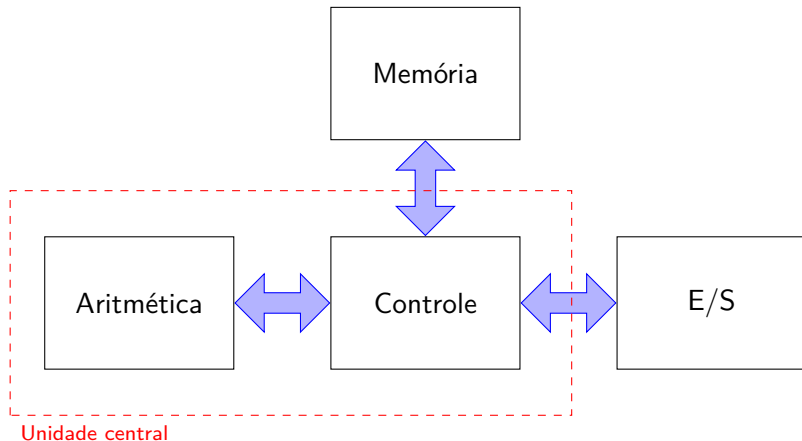
# John von Neumann

1903–1957

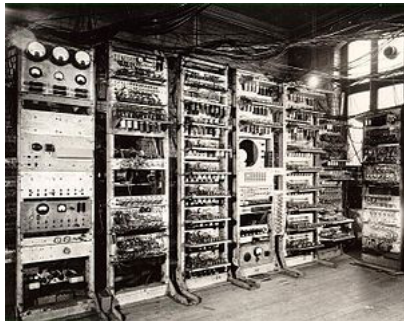
- ▶ Matemático
- ▶ Escreveu, em 1945, *"First Draft of a Report on the EDVAC"*, durante uma consultoria para a Universidade da Pensilvânia, em que descreve uma arquitetura de computador em que os programas e dados são armazenados juntos na memória. Esta arquitetura teve influência do projeto ENIAC.



# Arquitetura de von Neumann



# Manchester Mark 1



- ▶ Projetistas: **Frederic C. Williams** e **Tom Kilburn**;
- ▶ Desenvolvido na Universidade de Vitória em Manchester tornando-se operacional em **1949**;
- ▶ Continha inovações como programas armazenados em memória e registradores para armazenar índices das instruções;
- ▶ Possuía palavra de 40 bits com byte de 5 bits.
- ▶ Contou com a consultoria de Alan Turing que posteriormente produziria o **manual para o Mark II**.



# **Representação de caracteres no computador**

[Introdução à Organização de Computadores]

Adriano J. Holanda

31 de agosto de 2020

# Padrão ASCII

O padrão **ASCII** (*American Standard Code for Information Interchange*) foi desenvolvido na década de 1960 para padronizar a comunicação de mensagens.

- ▶ 8 bits: 7 para codificação e 1 para detecção de erro
- ▶ 128 códigos: 95 gráfico e 33 de controle

# Código gráfico

## ASCII

Parte da tabela ASCII para representação de algumas letras.

bin	oct	dec	hex	letra
0100 0001	101	65	41	A
0100 0010	102	66	42	B
0100 0011	103	67	43	C
0100 0100	104	68	44	D
0100 0101	105	69	45	E
0100 0110	106	70	46	F
0100 0111	107	71	47	G
0100 1000	110	72	48	H
0100 1001	111	73	49	I

bin	oct	dec	hex	letra
0110 0001	141	97	61	a
0110 0010	142	98	62	b
0110 0011	143	99	63	c
0110 0100	144	100	64	d
0110 0101	145	101	65	e
0110 0110	146	102	66	f
0110 0111	147	103	67	g
0110 1000	150	104	68	h
0110 1001	151	105	69	i

# Dígitos

ASCII

Representação dos dígitos:

<b>bin</b>	<b>oct</b>	<b>dec</b>	<b>hex</b>	<b>dígito</b>
0011 0001	061	49	31	1
0011 0010	062	50	32	2
0011 0011	063	51	33	3
0011 0100	064	52	34	4
0011 0101	065	53	35	5
0011 0110	066	54	36	6
0011 0111	067	55	37	7
0011 1000	070	56	38	8
0011 1001	071	57	39	9

# Códigos de controle

## ASCII

Alguns códigos de controle:

<b>bin</b>	<b>oct</b>	<b>dec</b>	<b>hex</b>	<b>código em C</b>	<b>função</b>
0000 0000	000	00	00	<code>\0</code>	nulo
0000 1001	011	09	09	<code>\t</code>	tabulação horizontal
0000 1010	012	10	0A	<code>\n</code>	nova linha
0001 1011	033	27	1B		escapar (Esc)
0111 1111	177	127	7F		apagar (Del)

# Padrão Unicode

O Padrão **Unicode** é usado para representação de texto no computador, englobando a maioria das línguas existentes.

O padrão ISO/IEC 10646 suporta 3 formatos de codificação:

- UTF-8:** Os caracteres possuem tamanho variável e correspondência com o ASCII.

- UTF-16** Representa a maioria dos caracteres com 16 bits.

- UTF-32** Possui tamanho fixo de 32 bits, usado quando não houver preocupação com armazenamento.

Todas as codificações usam 4 bytes no máximo.

# Codificação UTF-8

## Unicode

A codificação UTF-8 foi desenvolvida por Ken Thompson e Rob Pike em 1992, com o objetivo de ser compatível com o padrão ASCII e evitar problemas de portabilidade devido à representação de bits.

O padrão é recomendado pelo consórcio W3C e *Internet Mail* para representação dos caracteres.

Exemplo: *tag* HTML indicando o uso da codificação UTF-8

```
<meta http-equiv="Content-Type" content="text/html;  
charset=utf-8" />
```

# Codificação UTF-8

## Unicode

Boa parte dos sistemas operacionais e linguagens de programação atuais utilizam UTF-8 para codificação dos caracteres.

O número de bytes utilizados varia de acordo com o conjunto de caracteres a serem representados:

- 1 byte Caracteres ASCII.
- 2 bytes Alfabetos latino, grego, cirílico, árabe, hebreu, armênio.
- 3 bytes Restante das línguas.
- 4 bytes Símbolos matemáticos, símbolos pictóricos.



# **PROCESSADORES: Organização e Operação Geral.**

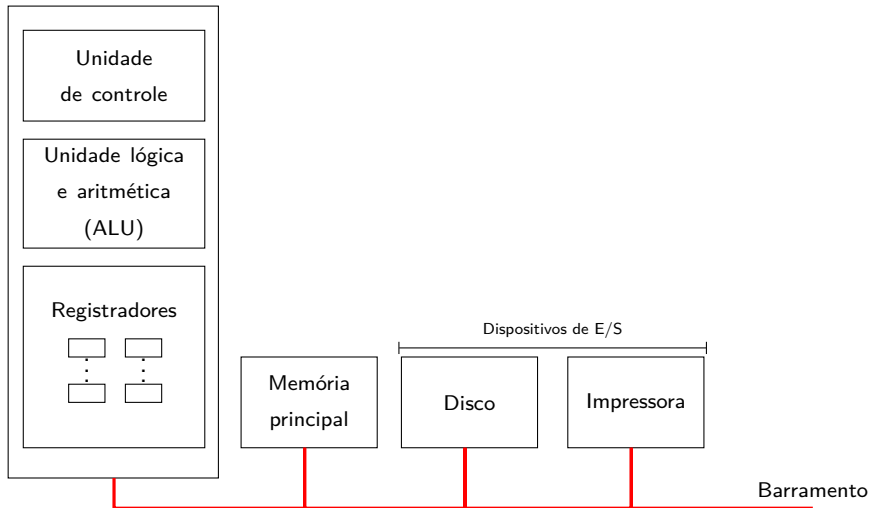
[Introdução à Organização de Computadores]

Adriano J. Holanda

31 de agosto de 2020

# Organização de um computador simples<sup>1</sup>

Unidade central de processamento (CPU)



<sup>1</sup> Figura adaptada de Tanenbaum, 2007, pg 29.

# 0 Processador

# Tarefas de um processador

Principais tarefas de um processador MIPS divididas por estágios.

*Instruction fetch stage – IF* → Recuperar as instruções da memória.

*Instruction decode/register file read stage – ID* → Ler os registradores enquanto decodifica a instrução.

*Execution stage – EX* → Execução da instrução ou cálculo de um endereço.

*Memory access stage – MEM* → Acessar um operando na memória de dados.

*Write back stage – WB* → Escreve o resultado no registrador.

# Tarefas de um processador

Principais tarefas de um processador MIPS divididas por estágios.

*Instruction fetch stage – IF* → Recuperar as instruções da memória.

*Instruction decode/register file read stage – ID* → Ler os registradores enquanto decodifica a instrução.

*Execution stage – EX* → Execução da instrução ou cálculo de um endereço.

*Memory access stage – MEM* → Acessar um operando na memória de dados.

*Write back stage – WB* → Escreve o resultado no registrador.

# Tarefas de um processador

Principais tarefas de um processador MIPS divididas por estágios.

*Instruction fetch stage – IF* → Recuperar as instruções da memória.

*Instruction decode/register file read stage – ID* → Ler os registradores enquanto decodifica a instrução.

*Execution stage – EX* → Execução da instrução ou cálculo de um endereço.

*Memory access stage – MEM* → Acessar um operando na memória de dados.

*Write back stage – WB* → Escreve o resultado no registrador.

# Tarefas de um processador

Principais tarefas de um processador MIPS divididas por estágios.

*Instruction fetch stage – IF* → Recuperar as instruções da memória.

*Instruction decode/register file read stage – ID* → Ler os registradores enquanto decodifica a instrução.

*Execution stage – EX* → Execução da instrução ou cálculo de um endereço.

*Memory access stage – MEM* → Acessar um operando na memória de dados.

*Write back stage – WB* → Escreve o resultado no registrador.

# Tarefas de um processador

Principais tarefas de um processador MIPS divididas por estágios.

*Instruction fetch stage – IF* → Recuperar as instruções da memória.

*Instruction decode/register file read stage – ID* → Ler os registradores enquanto decodifica a instrução.

*Execution stage – EX* → Execução da instrução ou cálculo de um endereço.

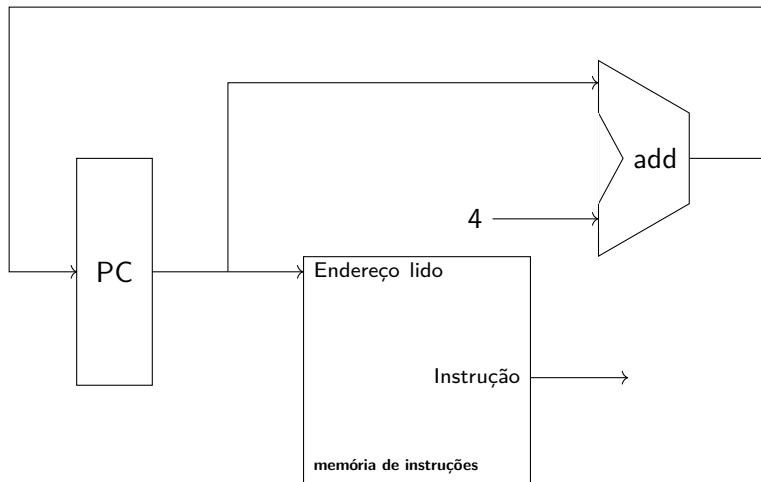
*Memory access stage – MEM* → Acessar um operando na memória de dados.

*Write back stage – WB* → Escreve o resultado no registrador.



# Primeiro estágio da via de dados

Via de dados de busca de instruções e incremento do “contador de programa”(PC)



# Processador MIPS32 M4K Core

FMT – Fixed mapping translation

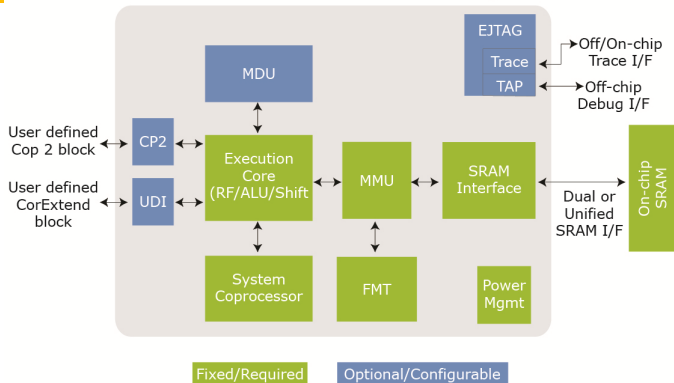
CP – CoProcessor

SRAM – Static RAM

MDU – Integer multiply/divide unit

MMU - memory management unit

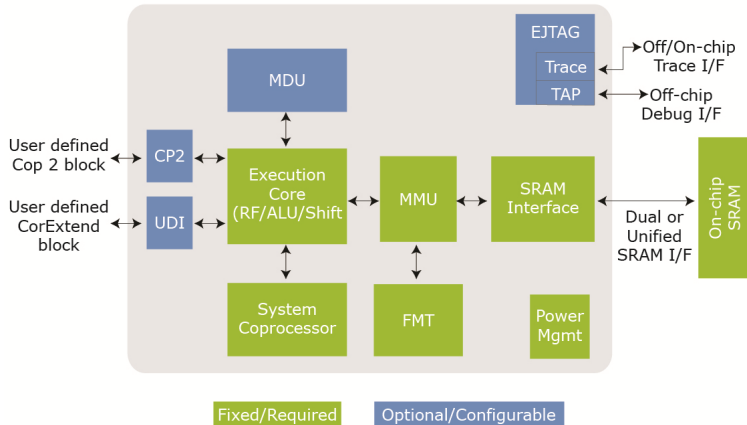
## MIPS32® M4K™ Core Block Diagram



Fonte: MIPS Technologies, Inc. Reproduzida com permissão.

# Processador MIPS

MIPS32® M4K™ Core Block Diagram



# Outros componentes

*Clock*: pulso uniforme e com frequência fixa que permite a sincronização dos eventos;

*Flip-flop*: circuito com 2 estados estáveis usado para armazenar informações de estado;

*Latch*: célula de memória estática;

Temporizador: tipo especializado de *clock*;

Deslocadores: deslocamento de bits;

Registradores memória do processador.

# Arquiteturas: RISC x CISC

# RISC

## Reduced Instruction Set Computer

### Características

- ▶ Conjunto pequeno e simples de instruções;
- ▶ Instruções executadas no hardware, não havendo microprogramação;
- ▶ As instruções levam aproximadamente a mesma quantidade de tempo para serem executadas;
- ▶ Maior número de registradores para reduzir o número de acessos à memória principal.

### Características:

- ▶ Possui um conjunto grande e complexo de instruções, algumas armazenadas como código de baixo nível no processador, que fornece ao programador de linguagem de montagem uma camada adicional chamada microprogramação.

### Vantagens e desvantagens com relação à RISC:

- ▶ Uma vantagem seria fornecer ao programador um número maior de operações que na arquitetura RISC deveriam ser codificadas;
- ▶ O outro lado da moeda é que algumas otimizações não podem ser feitas na arquitetura CISC devido à impossibilidade de alterar a instrução composta com o objetivo de melhorar o desempenho.




### Características:

- ▶ Possui um conjunto grande e complexo de instruções, algumas armazenadas como código de baixo nível no processador, que fornece ao programador de linguagem de montagem uma camada adicional chamada microprogramação.

### Vantagens e desvantagens com relação à RISC:

- ▶ Uma vantagem seria fornecer ao programador um número maior de operações que na arquitetura RISC deveriam ser codificadas;
- ▶ O outro lado da moeda é que algumas otimizações não podem ser feitas na arquitetura CISC devido à impossibilidade de alterar a instrução composta com o objetivo de melhorar o desempenho.



-  [Tanenbaum, 2007] Andrew S. Tanenbaum, Organização Estruturada de Computadores, Editora LTC, 2007,
-  [Stallings, 2006] William Stallings, Arquitetura e Organização de Computadores, Makron Books, 2006
-  [Patterson, 2008] David A. Patterson; John L. Hennessy, Computer Organization and Design, Morgan Kaufmann Publisher, 2008,