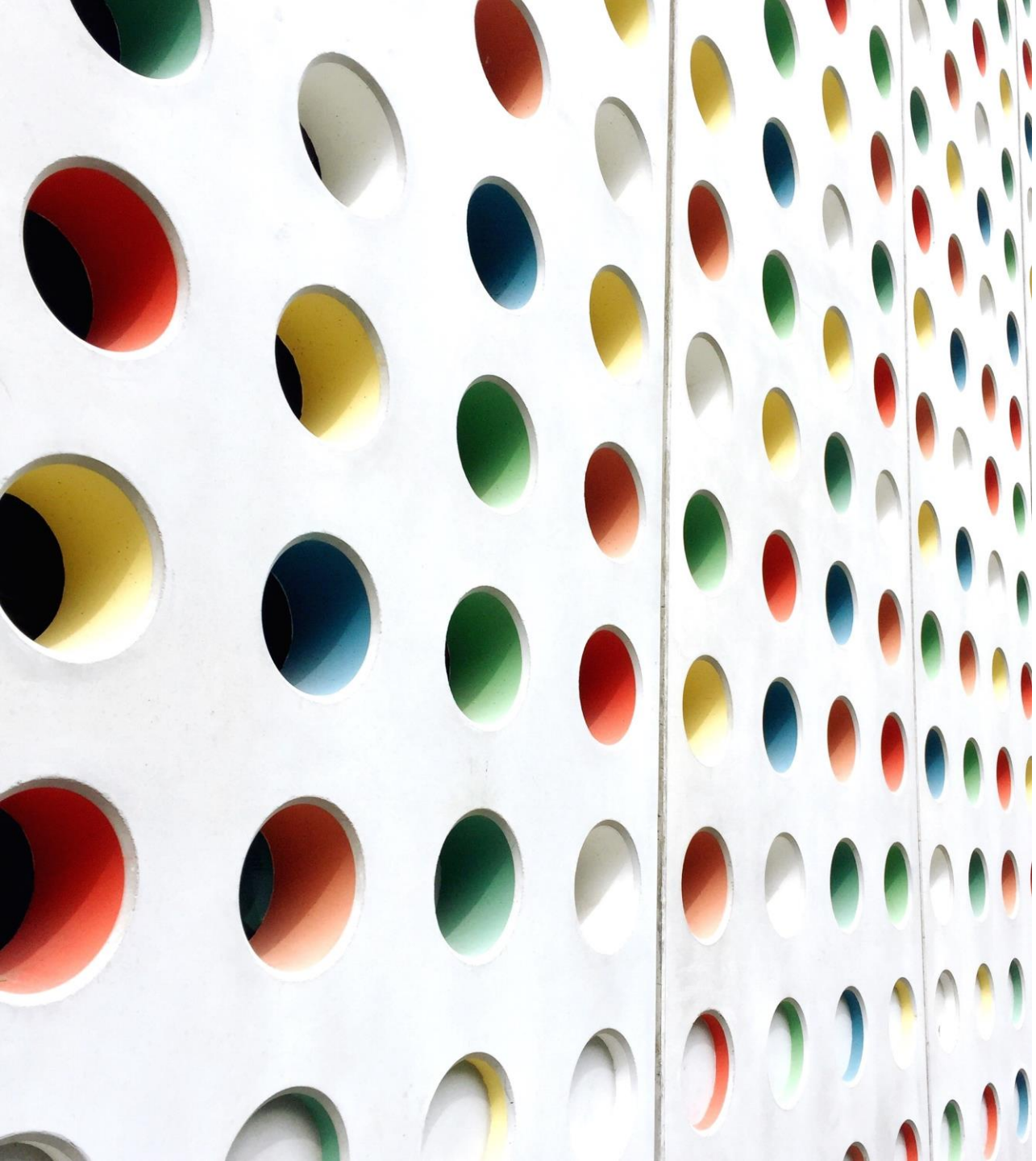
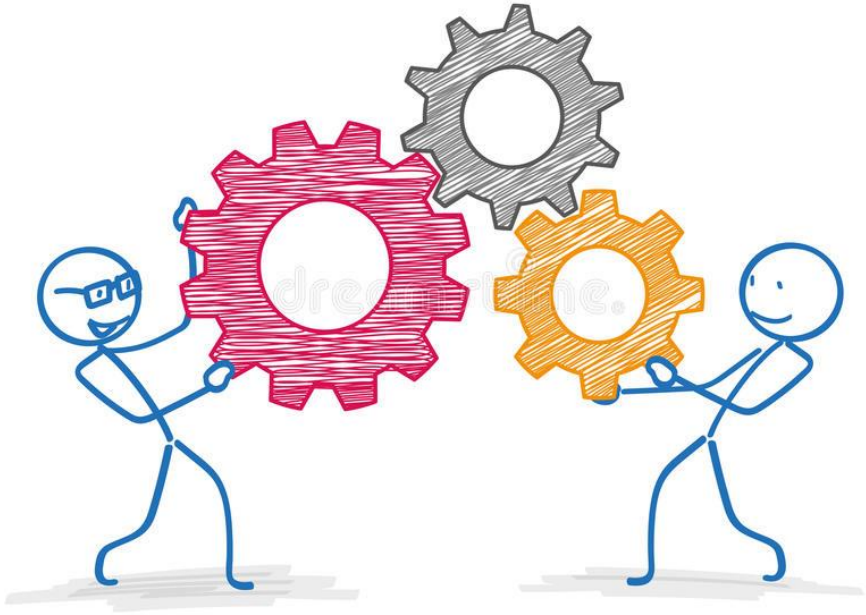


Arquiteturas Computacionais

INF028 – ARQUITETURA DE COMPUTADORES E SOFTWARE BÁSICO

FLAVIAMSN@IFBA.EDU.BR

Arquiteturas Computacionais



Arquiteturas computacionais

Duas arquiteturas computacionais são mais proeminentes:

Arquitetura Harvard (Harvard University)

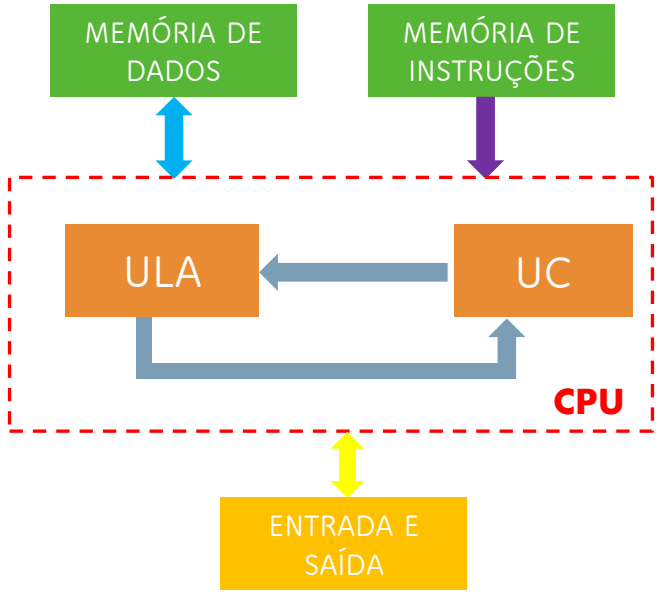
Arquitetura de Von Neumann (Princetown University)

Estes dois modelos arquiteturais formam a base para o computador moderno.

Recapitulando...

Arquitetura Harvard

- **Principal característica:** armazenar dados e instruções em diferentes locais.
- Instruções eram armazenadas em cartões perfurados
- Dados eram armazenados em componentes eletrônicos dentro da CPU

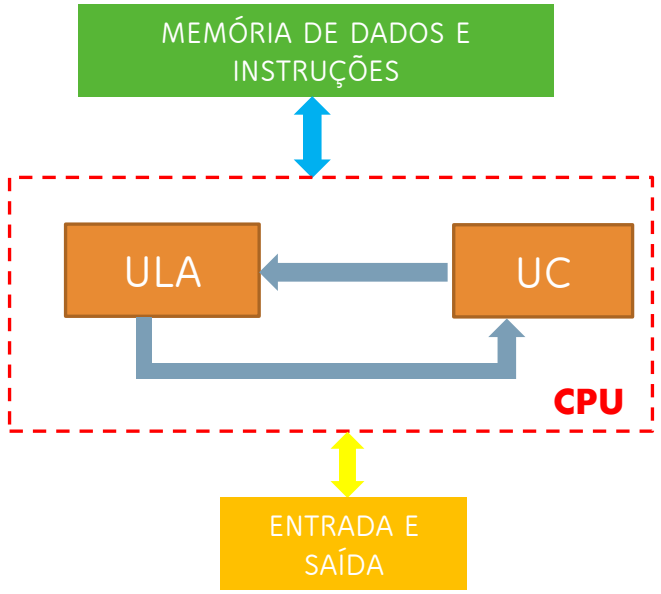


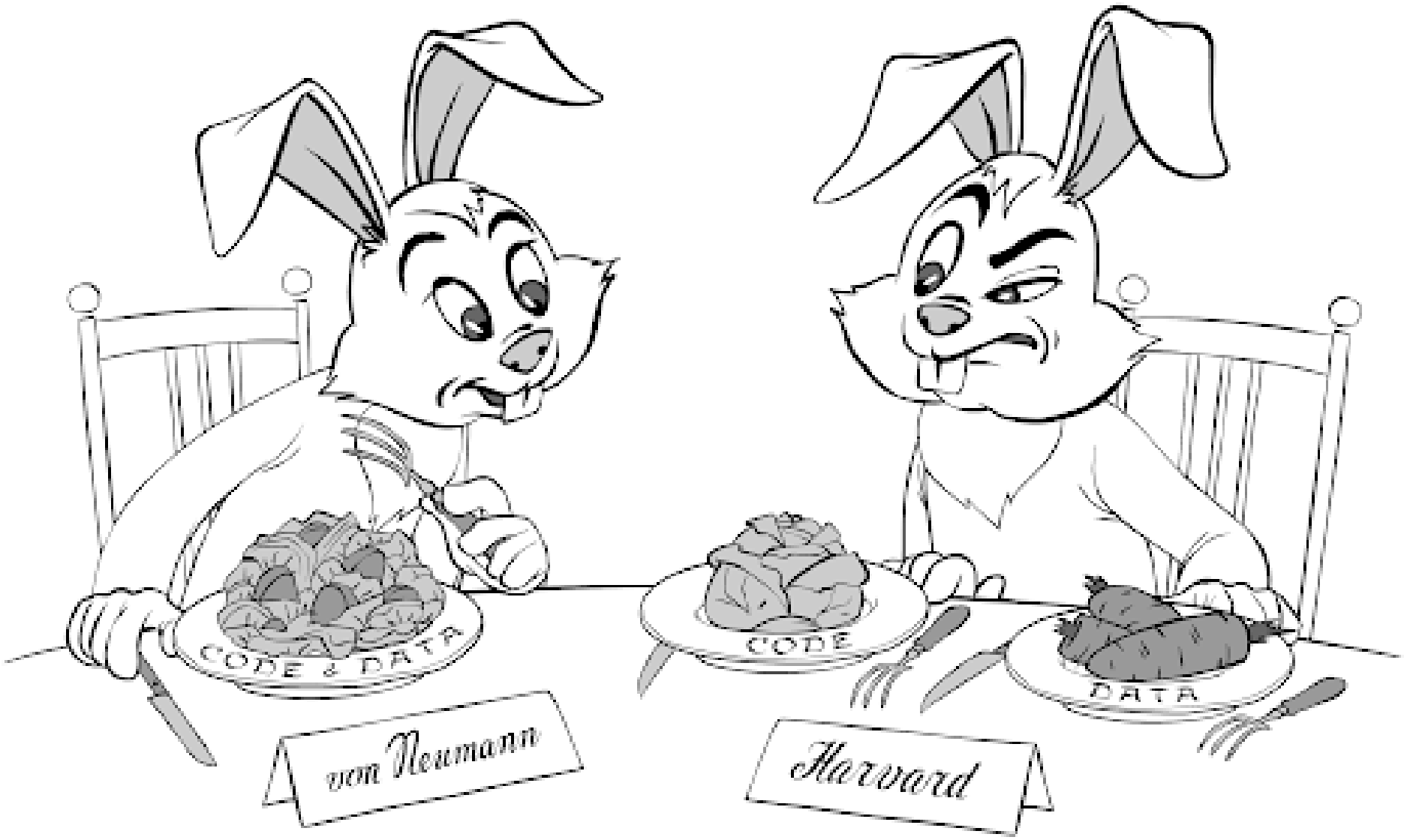
Recapitulando...

Arquitetura Von Neumann

- estabelece o conceito de programa armazenado;

- **Principal característica**: armazenar dados e instruções na memória principal

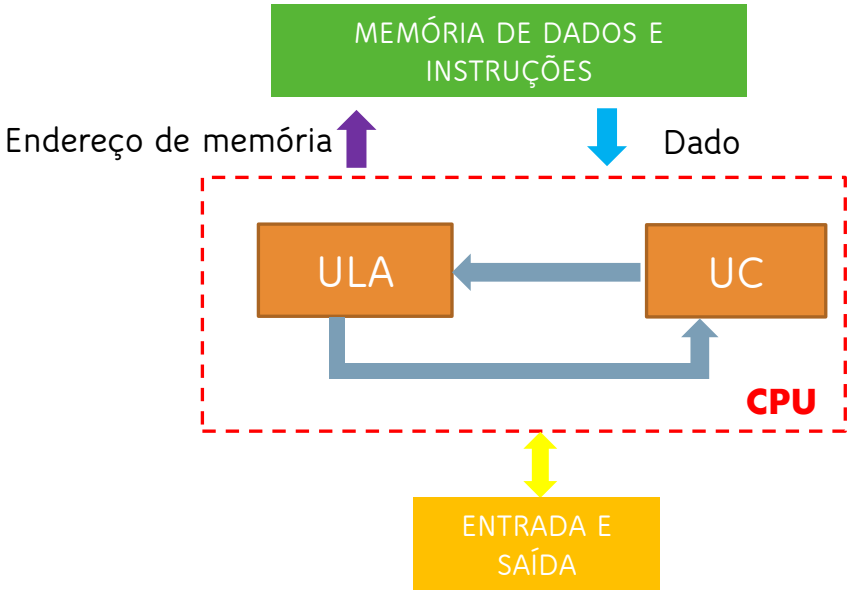



CODE & DATA
von Neumann
CODE
DATA
Harvard

Como acontecia a execução dos programas em cada uma destas arquiteturas?

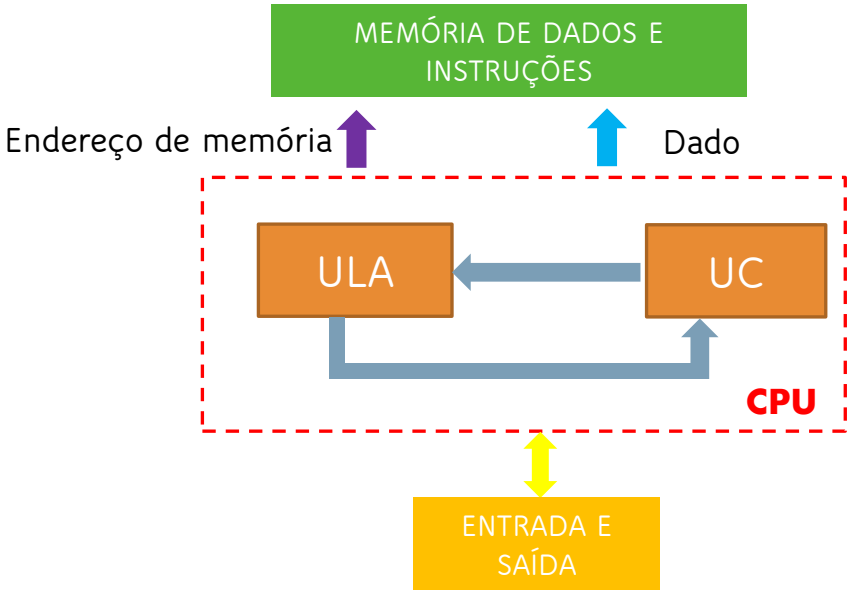
Execução de programa – Arquitetura Von Neumann

Para **recuperar** um dado da memória



Execução de programa – Arquitetura Von Neumann

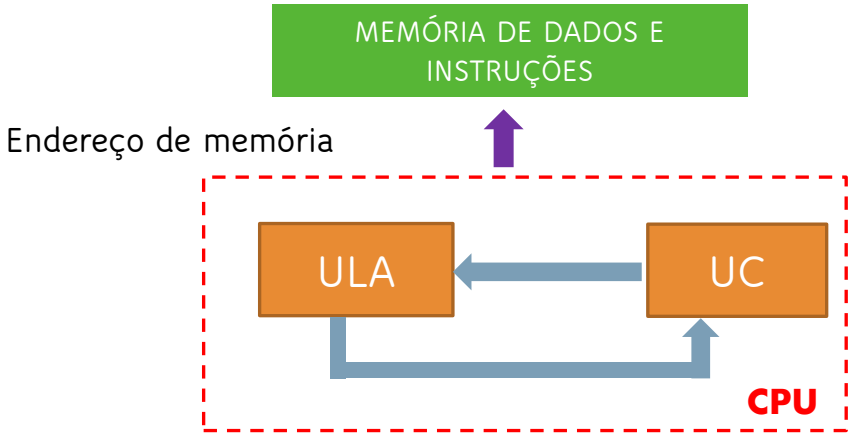
Para **armazenar** um dado da memória



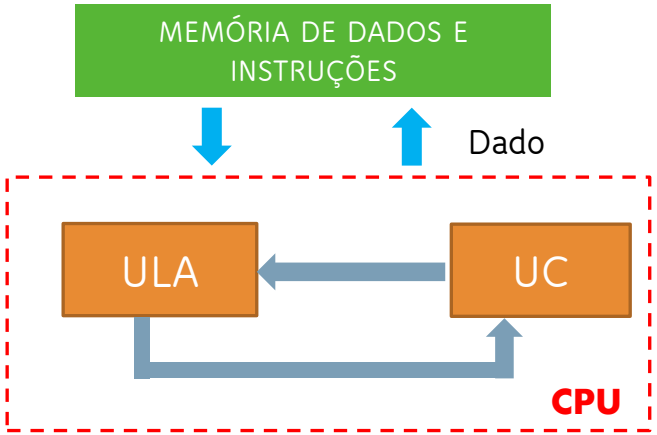

Qual o caminho percorrido pelos dados?
E pelo endereço?

Arquitetura de Von Neumann – Caminho dos dados e instruções

- Endereços trafegam em uma direção



Dados trafegam em duas direções



Arquitetura de Von Neumann – Caminho dos dados e instruções

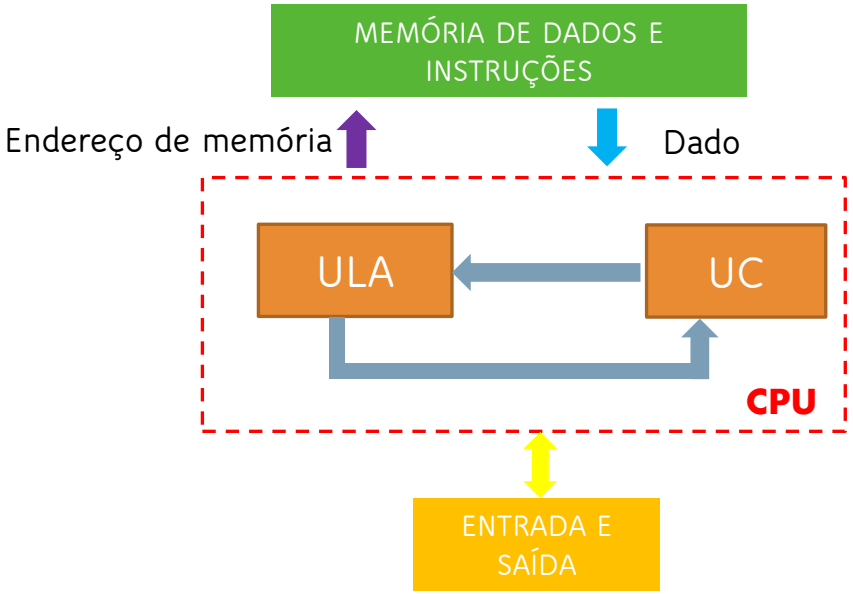
Na arquitetura de Von Neumann dados e instruções compartilham o mesmo barramento!

Qual a consequência disto??

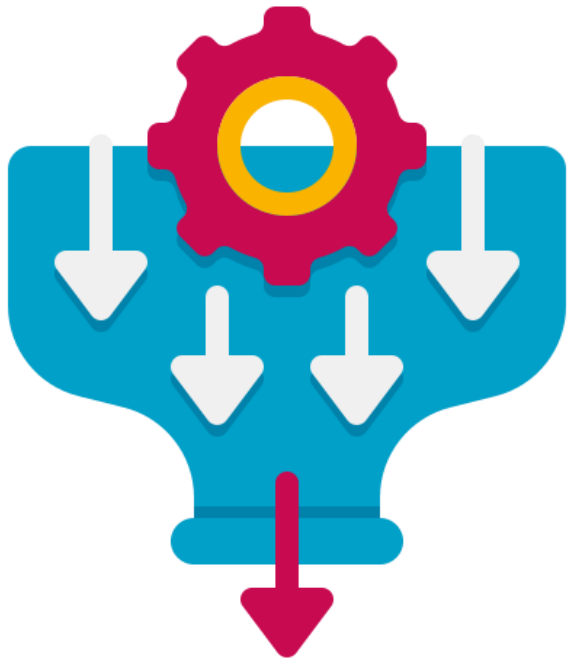
Eles precisam ser buscados separadamente!!

Exemplo – Arquitetura de Von Neumann

- Uma instrução para trazer um dado para a CPU precisa de dois ciclos.
  - Ciclo 1 → endereçamento;
  - Ciclo 2 → carregamento da instrução no processador.

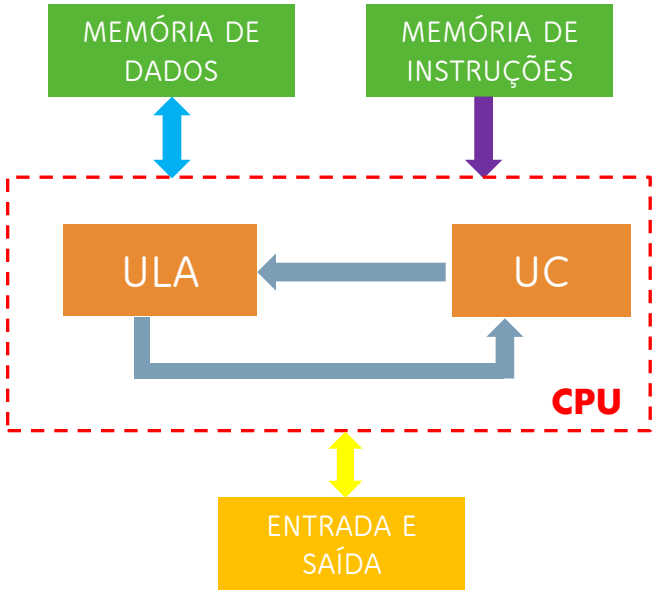


De volta ao gargalo de Von Neumann



Execução de Programa – Arquitetura Harvard

Função: recuperar um dado da memória



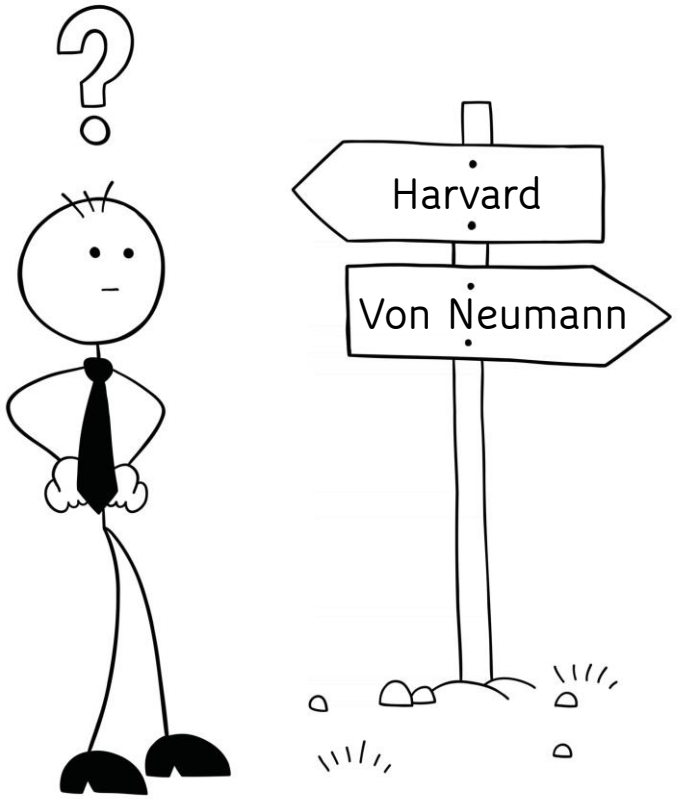
Arquitetura de Harvard **melhora** o gargalo de Von Neumann, porque faz uso de <u>duas</u> memórias: uma para os dados e outra para as instruções.

Arquitetura Harvard

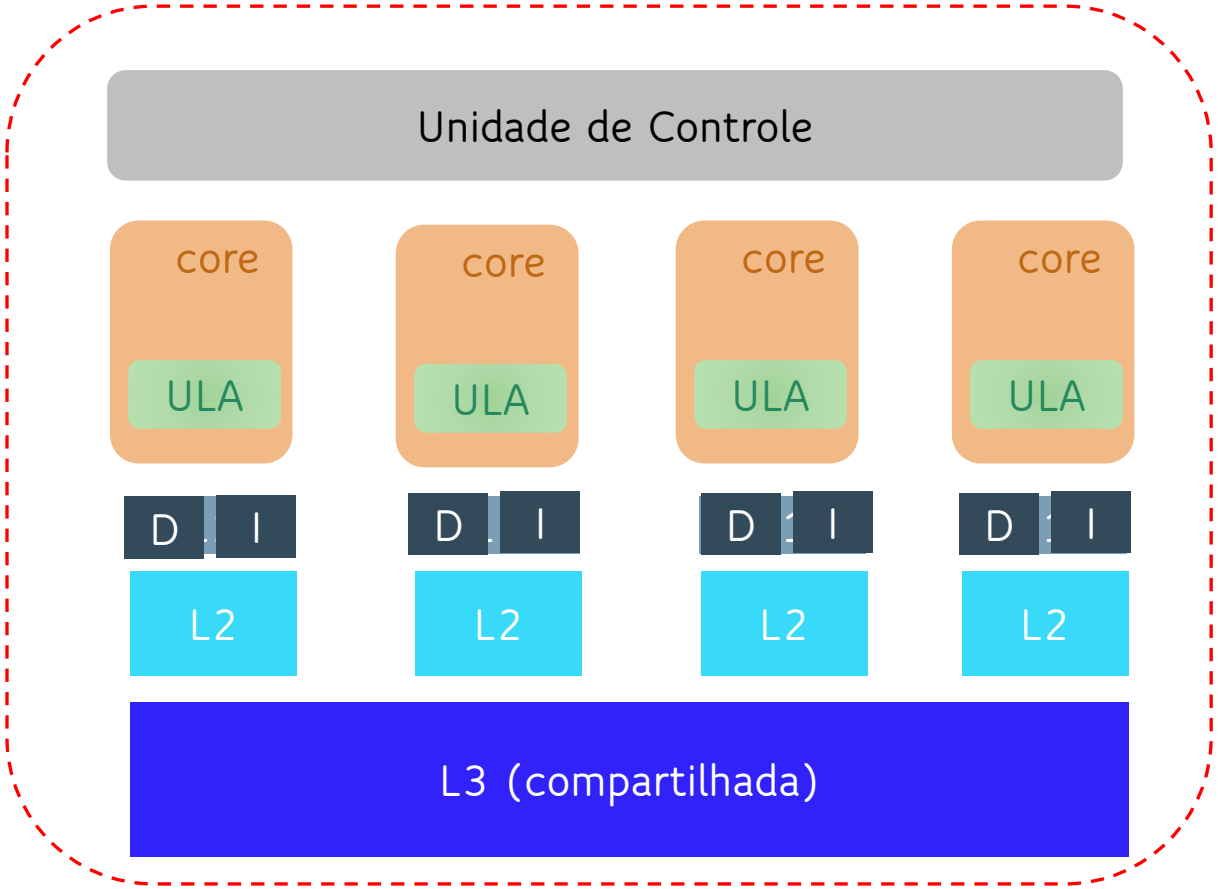
- A arquitetura fica mais flexível:
  - Permite diferentes capacidades de armazenamento e largura do barramento.

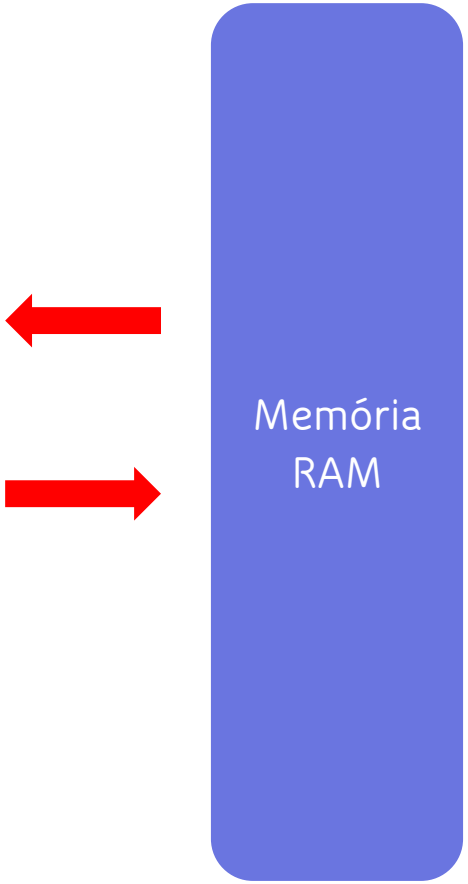
- Mais frequentemente encontrada na área de processamento digital de sinais:
  - audio e video, aplicações médicas (raio X), smart watches, aplicativos de digital assistance (Alexa Google Home);

Mas e o meu computador?



Arquitetura Moderna





Arquitetura Harvard Modificada

Arquiteturas RISC e CISC

:: CISC

Complex Instruction Set Computer

Processador que executa instruções complexas:

- Instruções longas;
- Muitas operações matemáticas;
- Diferentes operações matemáticas.

- Instruções não possuem tamanho padrão;
- Instruções requerem que o processador acesse a memória antes de sua execução

:: CISC

O tempo de processamento é maior;

Um processador pode exigir vários ciclos (clock) para executar uma única instrução.

- Ciclo de clock – medido em Hertz: quantos impulsos são realizados por segundo.
- Exemplo: um processador com 350Mhz realiza 350 milhões de impulsos por segundo.
- Quando se aumenta o ciclo do processador, se diminui a quantidade de ciclos necessários para executar uma instrução.

A família de processadores x86 da Intel é uma das mais reconhecidas usuárias da arquitetura CISC.

Arquitetura CISC predominou por anos, até que a Mac (Apple) mudou o cenário.

:: RISC

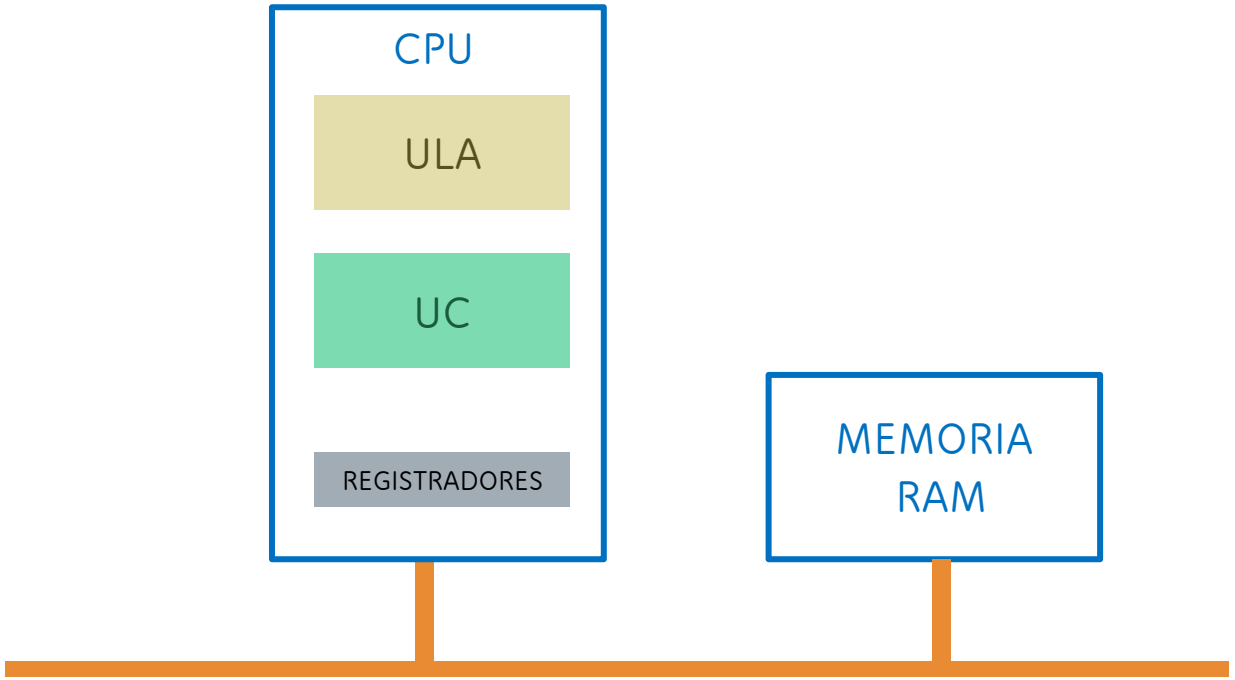
Reduced Instruction Set Computer

Processador executa instruções reduzidas:

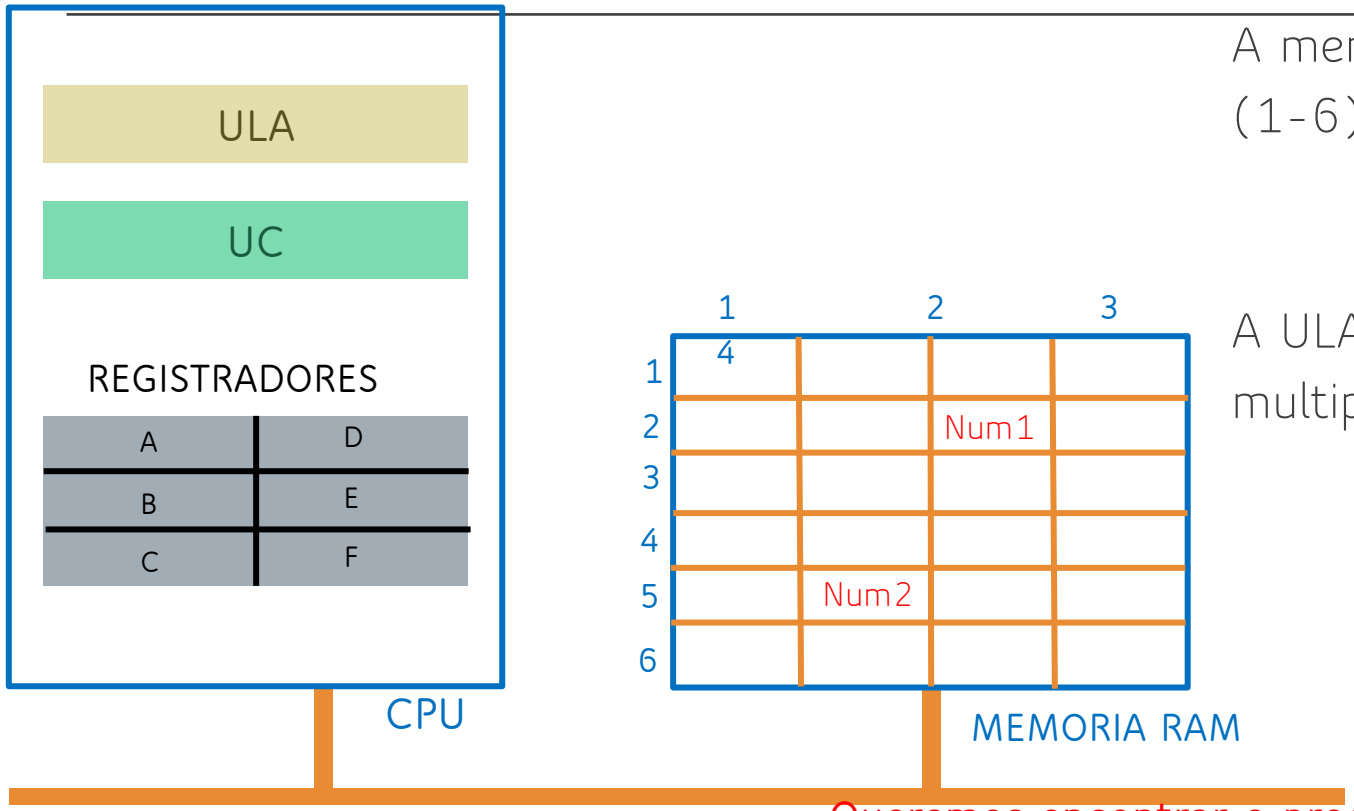
- Uma instrução é subdividida em várias instruções menores e mais simples;
- Instruções assumem um tamanho padrão;
- Instruções executadas em apenas um ciclo (clock).

Algumas máquinas com arquitetura RISC são *a Power da IBM e Sparc da Oracle*

:: Um exemplo ilustrativo



:: Multiplicando dois números na memória

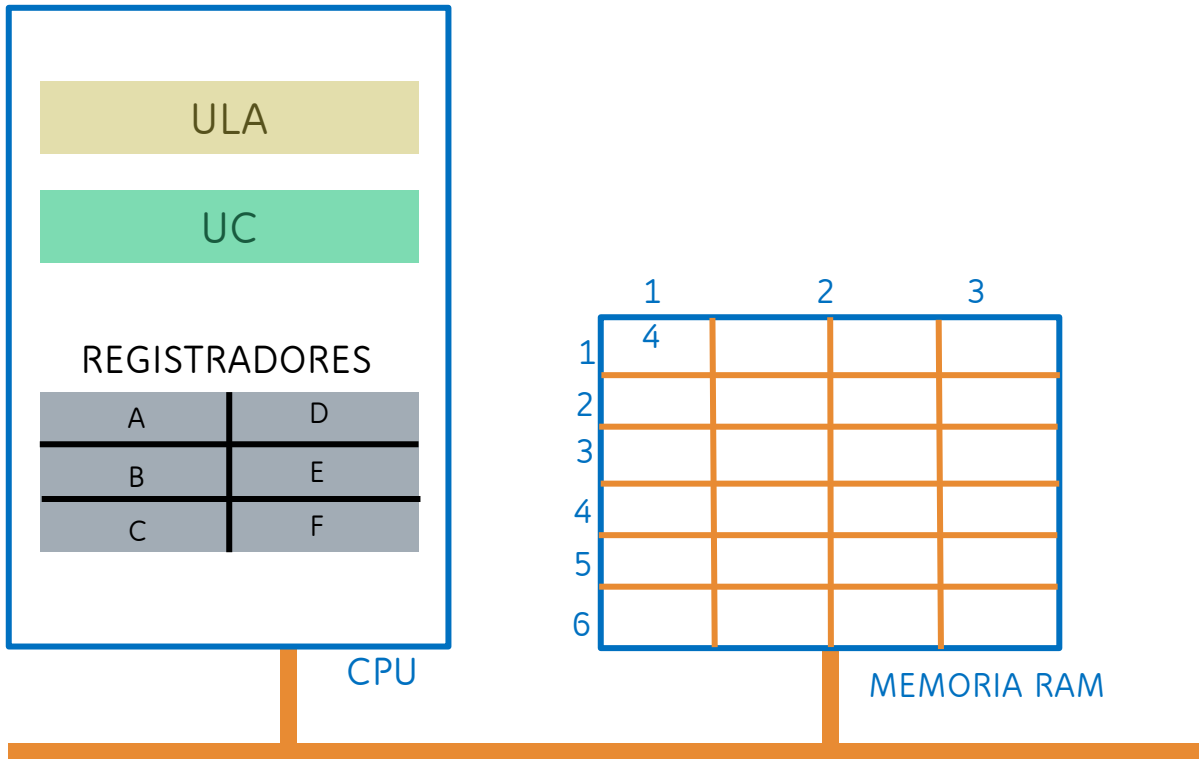


A memória principal está dividida em linhas (1-6) e colunas (1-4).

A ULA é a responsável pelo cômputo da multiplicação.

Queremos encontrar o produto de dois números:
Num1 = 2:3 e Num2= 5:2.
O resultado deve ser armazenado em Num3= 2:3.

:: CISC



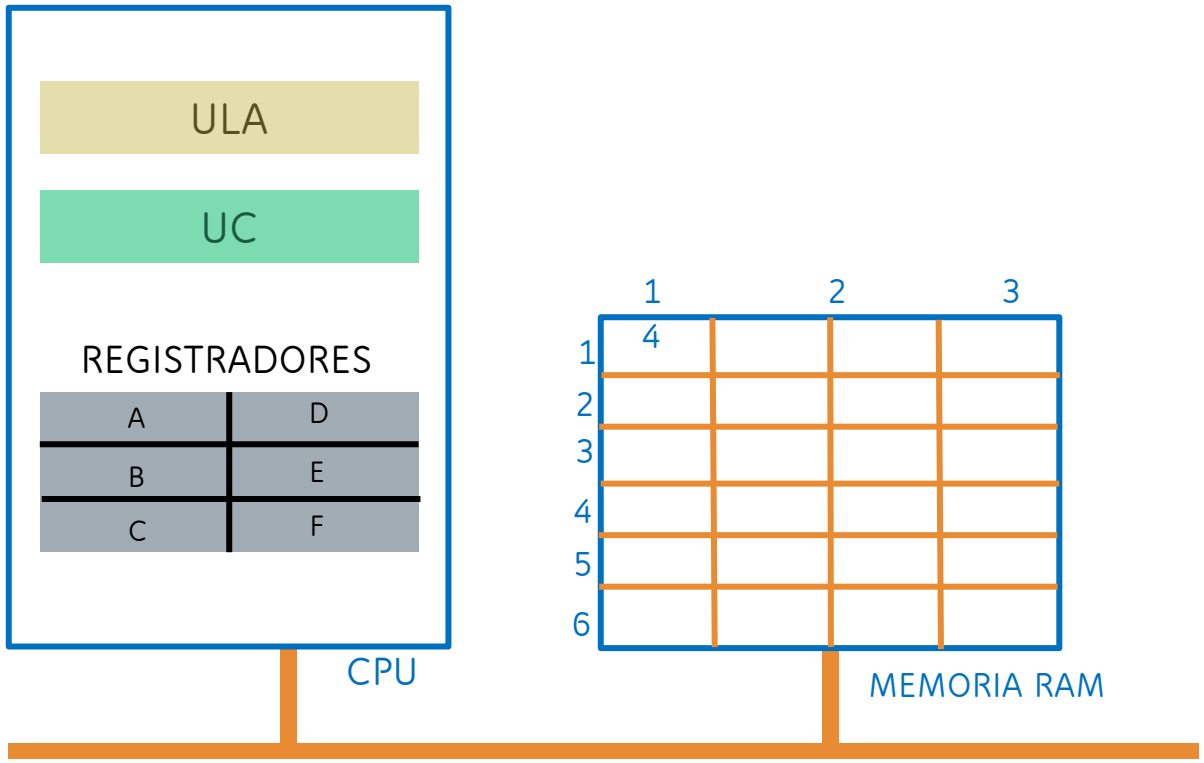
CISC – busca completar uma tarefa com o menor número possível instruções

Para isso, o hardware do processador é capaz de compreender e executar uma série de instruções "menores".

Neste caso, um processador CISC viria preparado com uma instrução específica ➔ MULT

Opera diretamente na memória do computador.

:: CISC



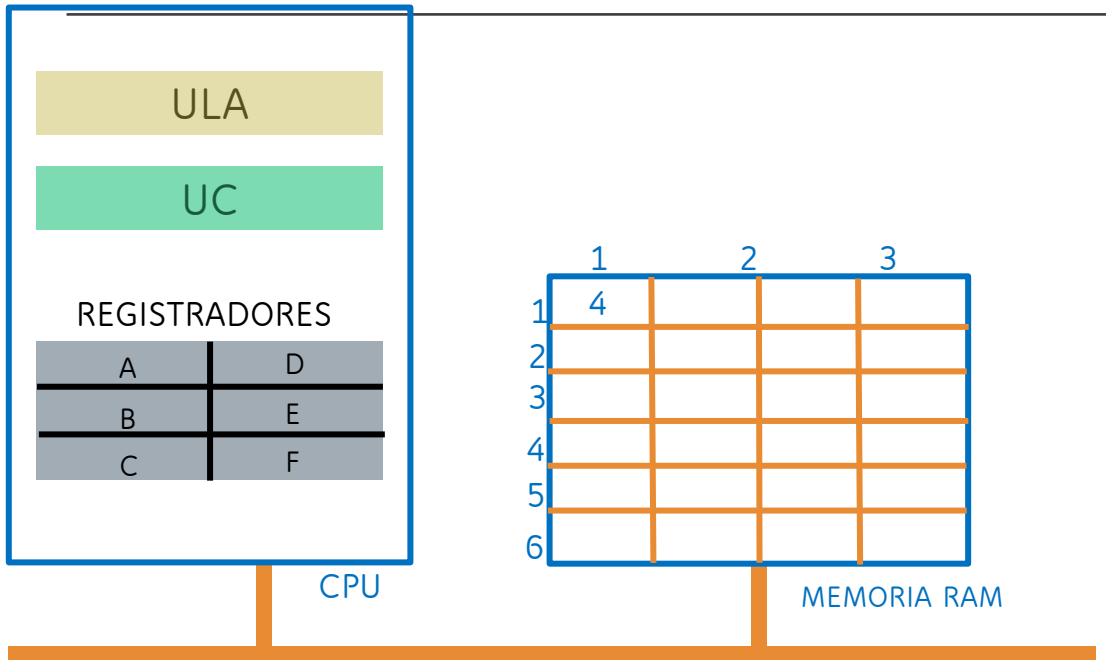
MULT é uma "instrução complexa".

Não exige que o programador utilize explicitamente nenhuma função de **load** ou **store**.

Note que a multiplicação de dois números pode ser concluída com uma instrução

Neste caso, 2:3 = MULT (2:3, 5:2)

:: RISC

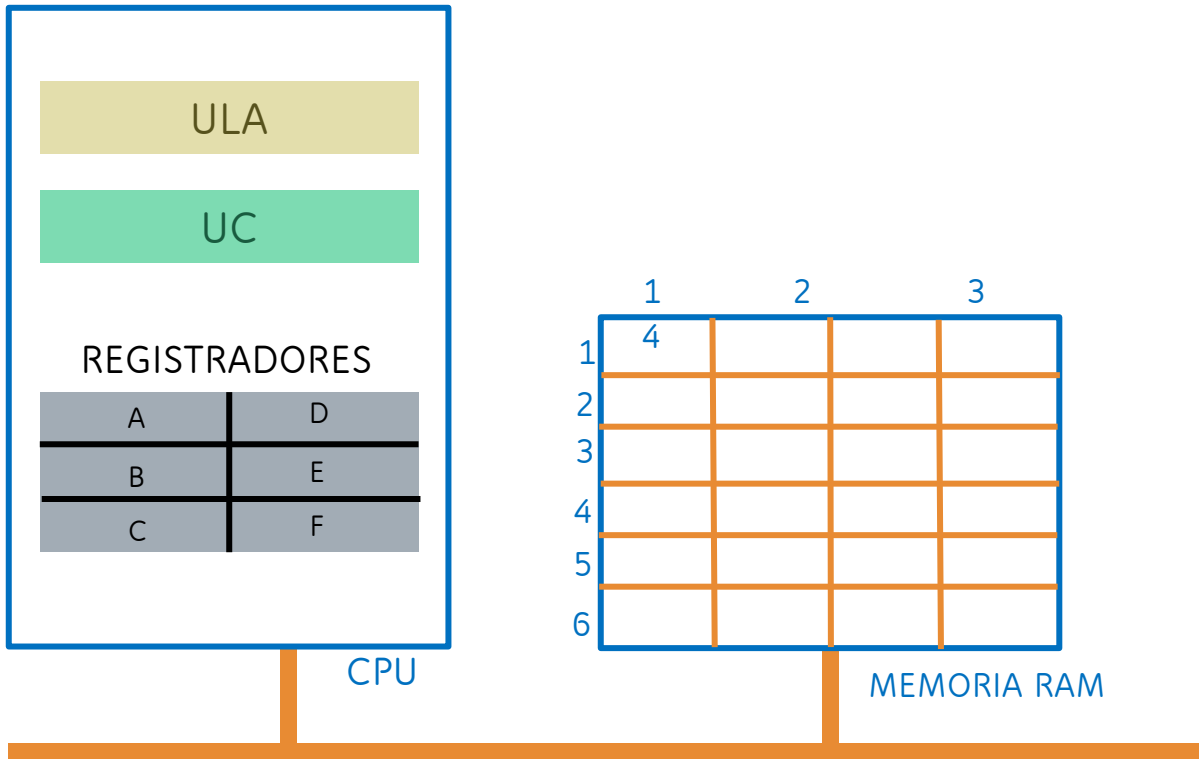


RISC - usa apenas instruções simples que podem ser executadas em um ciclo de clock.

Assim, o comando "MULT", pode ser dividido em três comandos separados:

- LOAD: que move os dados da memória para um registrador;
- PROD: encontra o produto de dois operandos (dentro dos registradores)
- STORE: move os dados de um registrador para a memória

:: RISC



Neste caso,

- LOAD A, 2: 3
- LOAD B, 5: 2
- PROD A, B
- STORE 2: 3, A

RISC vs. CISC

RISC

- Instruções são orientada a (arquitetura) da máquina;
- Instruções simples → mais linhas de código
- 1 instrução por ciclo
- Fácil de executar o pipeline

CISC

- Instruções são orientadas ao programador (software)
- Instruções complexas → menos linhas de código
- 1 instrução pode levar vários ciclos de clock
- Difícil de implementar o pipeline