INSTITUTO FEDERAL
RIO GRANDE DO SUL

Sistemas Operacionais

Memória Virtual
- Por Partições
- Por Segmentos
- Por Paginações

Apresentações Adaptadas do Material do Prof. Marcelo Paravisi

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Hardware de Memória

A memória principal é um componente fundamental em qualquer sistema de computação. Constitui o “espaço de trabalho” do sistema, no qual são mantidos os processos, threads e bibliotecas compartilhadas, além do próprio núcleo do sistema operacional, com seu código e suas estruturas de dados.

O hardware de memória pode ser bastante complexo, envolvendo diversas estruturas, como memórias **RAM**, **caches**, **unidade de gerência**, **barramentos**, etc, o que exige um esforço de gerência significativo por parte do sistema operacional.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Hardware de Memória

A memória de um computador é na verdade uma grande tabela com células sequenciais,cada célula tem seu próprio endereço que segue um padrão contínuo.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Código

```
char letra;
letra = 'B'

int num;
num = 1500;

double PI = 3.1415;
```

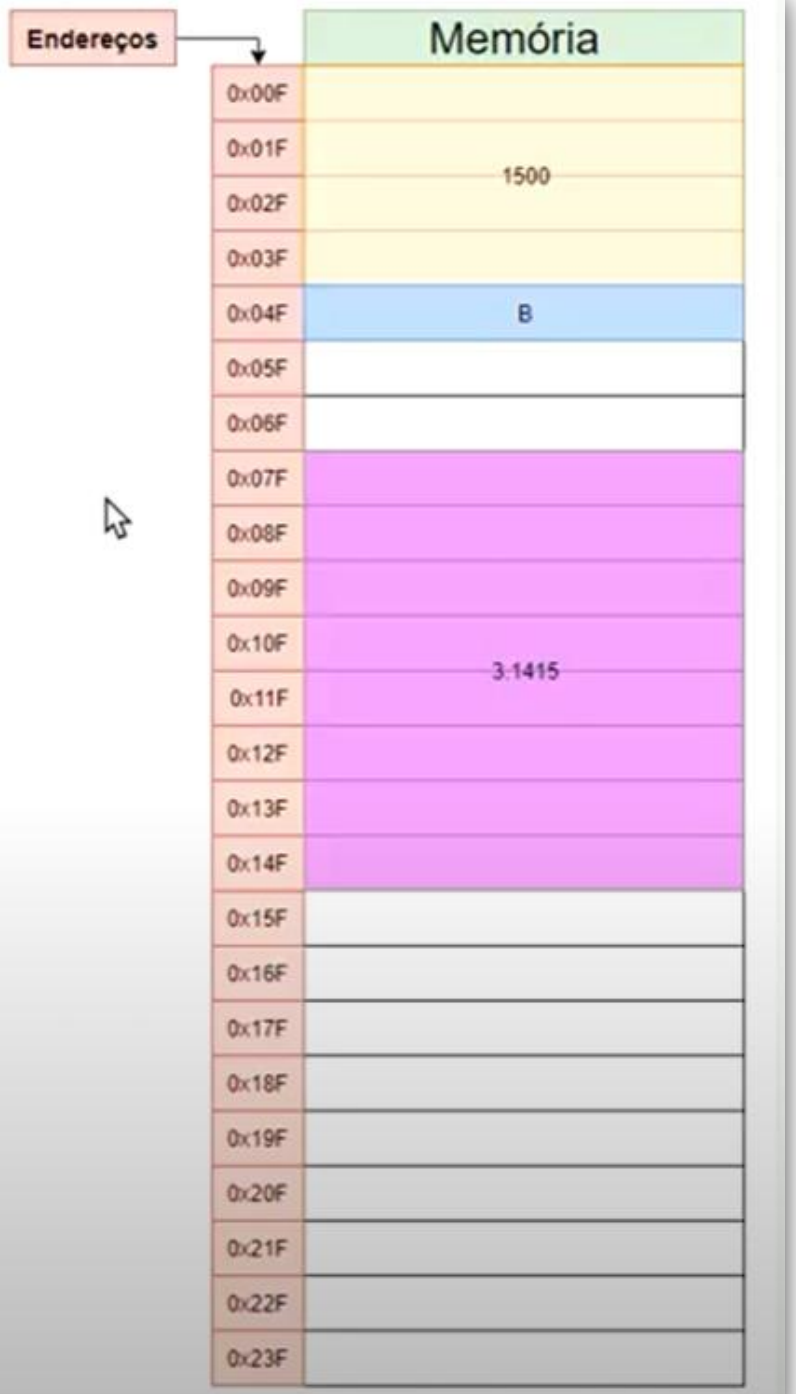
EM GERAL

Inteiros = 4 bytes

float = 4 bytes

double = 8 bytes

char = 1 byte



Hardware de Memória

Podemos considerar,

a primeira célula será	0x00F,	Conteúdo [00000000]
...a segunda	0x01F,	Conteúdo [00000000]
...a terceira	0x02F,	Conteúdo [00000000]
...a quarta	0x03F,	Conteúdo [00000000]
...a quinta	0x04F,	Conteúdo [00000000]
...a sexta	0x05F,	Conteúdo [00000000]
...a sétima	0x06F,	Conteúdo [00000000]
...a oitava	0x07F,	Conteúdo [00000000]

Hardware de Memória

Existem diversos tipos de memória em um sistema de computação, cada um com suas próprias características e particularidades, mas todos com um mesmo objetivo: armazenar informação.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Hardware de Memória

Tipicamente é possível identificar vários locais onde dados são armazenados: **os registradores e o cache interno do processador (denominado *cache L1*), o cache externo da placa mãe (*cache L2*) e a memória principal (RAM).**

Além disso, discos e unidades de armazenamento externos **(*pendrives*, CD-ROMs, DVD-ROMs, fitas magnéticas, etc.)**

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Hardware de Memória

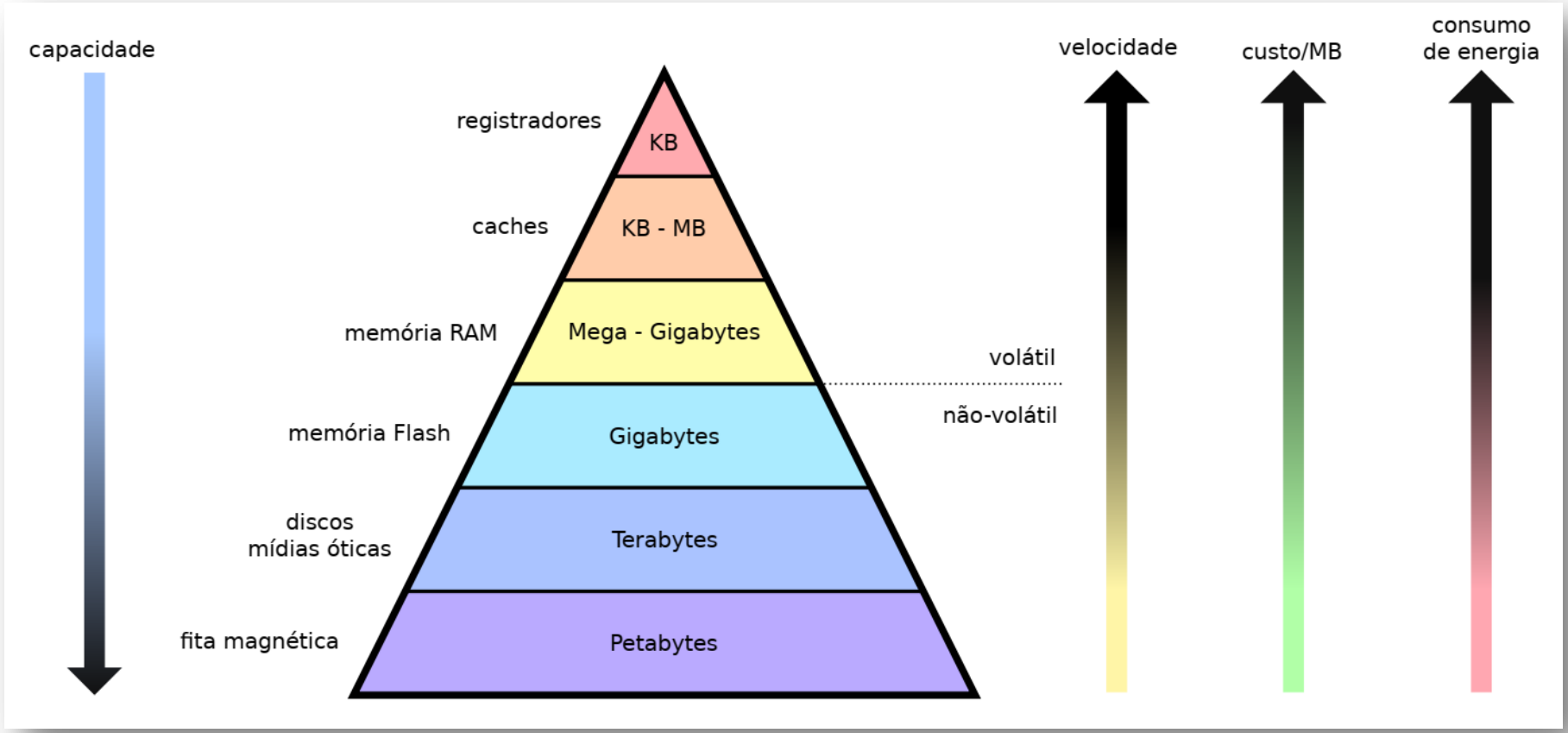
Normalmente as memórias mais rápidas, como os registradores da CPU e os caches, têm menor capacidade de armazenamento, mais caras e consomem mais energia que memórias mais lentas, como a memória principal (RAM) e os discos.

Além disso, as memórias mais rápidas são *voláteis*, ou seja, perdem seu conteúdo ao ficarem sem energia, quando o computador é desligado.

Memórias que preservam seu conteúdo mesmo quando não tiverem energia, como as unidades *Flash* e os discos rígidos, são denominadas *memórias não-voláteis*.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Hardware de Memória



INSTITUTO FEDERAL
RIO GRANDE DO SUL

Hardware de Memória

Outra característica importante das memórias é a rapidez de seu funcionamento, que pode ser traduzida em duas grandezas: o *tempo de acesso* (ou *latência*) e a *taxa de transferência*.

O tempo de acesso caracteriza o tempo necessário para iniciar uma transferência de dados de/para um determinado meio de armazenamento.

Por sua vez, a taxa de transferência indica quantos bytes por segundo podem ser lidos/escritos naquele meio, uma vez iniciada a transferência de dados .

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Hardware de Memória

Meio	Tempo de acesso	Taxa de transferência
Cache L2	1 ns	1 GB/s (1 ns por byte)
Memória RAM	60 ns	1 GB/s (1 ns por byte)
Memória *flash* (NAND)	2 ms	10 MB/s (100 ns por byte)
Disco rígido SATA	5 ms (tempo para o ajuste da cabeça de leitura e a rotação do disco até o setor desejado)	100 MB/s (10 ns por byte)
DVD-ROM	de 100 ms a vários minutos (caso a gaveta do leitor esteja aberta ou o disco não esteja no leitor)	10 MB/s (100 ns por byte)

Tempos de acesso e taxas de transferência típicas [Patterson and Henessy, 2005]

INSTITUTO FEDERAL
RIO GRANDE DO SUL

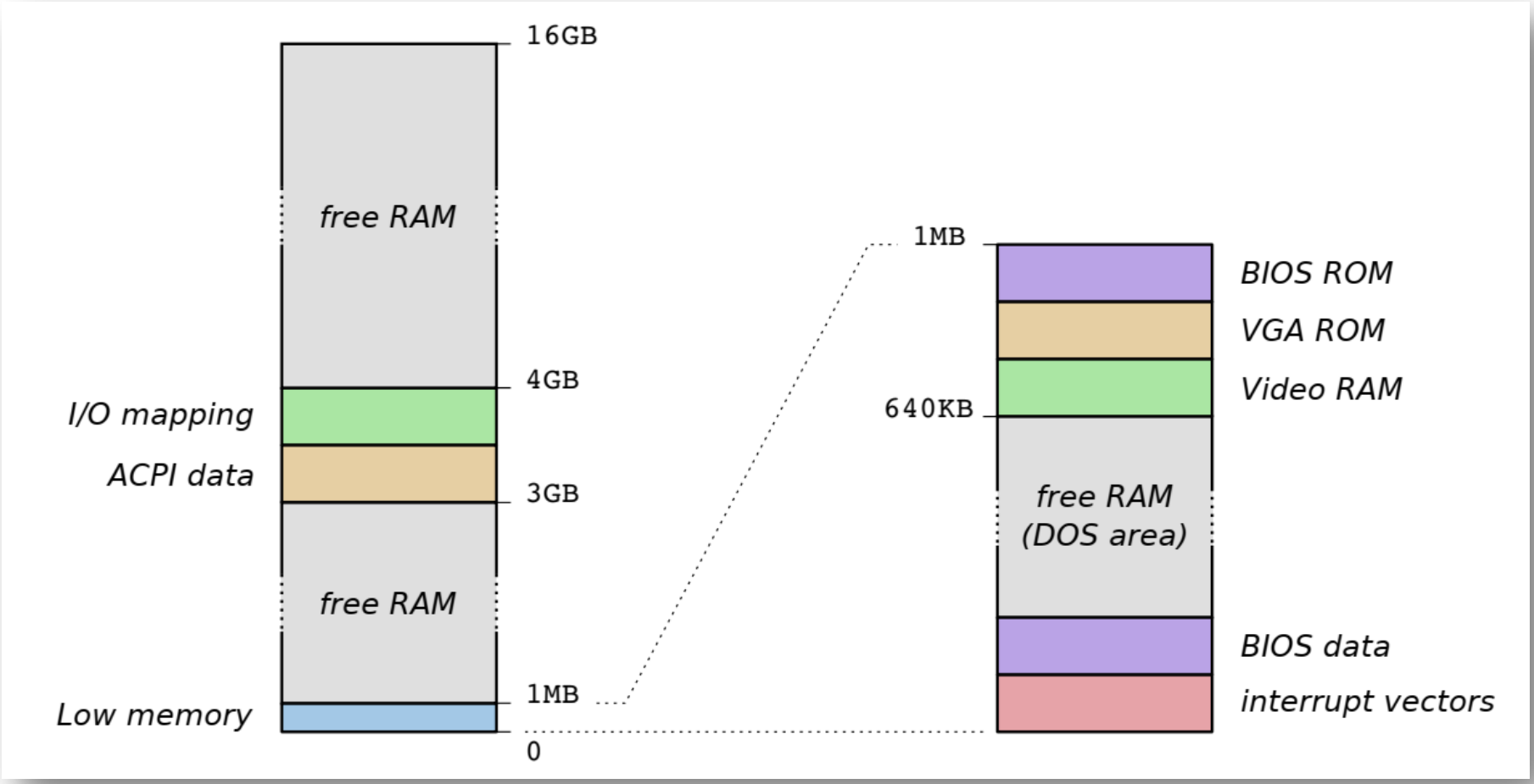
A memória Física

A memória principal do computador é uma área de RAM composta **por uma grande sequência de bytes**, que é a menor unidade de memória usada pelo processador.

Cada byte da memória RAM possui um endereço, que é usado para acessá-lo.

A quantidade de memória RAM disponível em um computador constitui seu **espaço de memória física**.

A memória física



Organização (simplificada) da memória RAM de um computador atual com 16 GB de memória RAM.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

A memória física

Os sistemas atuais mais sofisticados usam os conceitos de **espaços de endereçamento e de memória virtual**, para desacoplar a **visão da memória pelos processos da estrutura física da memória no hardware.**

Esse desacoplamento visa tornar mais simples e flexível o uso da memória pelos processos e pelo sistema operacional.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

A memória física

Espaço de endereçamento

O processador acessa a memória RAM através de barramentos de dados, de endereços e de controle.
O barramento de endereços (como os demais) possui um número fixo de vias, que define a quantidade total de endereços de memória que podem ser gerados pelo processador: um barramento de dados com *n* vias consegue gerar 2^n endereços distintos.

Ex: Intel 80386 possui 32 vias de endereços -> 2^{32} = 4GB
Intel Core i7 usa 48 vias para endereços ->2^{48} = 256TB

A memória Virtual

Para ocultar a organização complexa da memória física e simplificar os procedimentos, é implementado:

Endereços físicos (ou reais) são os endereços dos bytes de memória física do computador. Estes endereços são definidos pela quantidade de memória disponível na máquina

Endereços lógicos (ou virtuais) são os endereços de memória usados pelos processos e pelo sistema operacional e, portanto, usados pelo processador durante a execução. Estes endereços são definidos de acordo com o espaço de endereçamento do processador.

Os processos “enxergam” somente a memória virtual

A memória Virtual

Por questões de desempenho, a tradução de endereços lógicos em físicos é feita por um **componente específico do hardware** do computador, denominado **Unidade de Gerência de Memória** (MMU – *Memory Management Unit*).

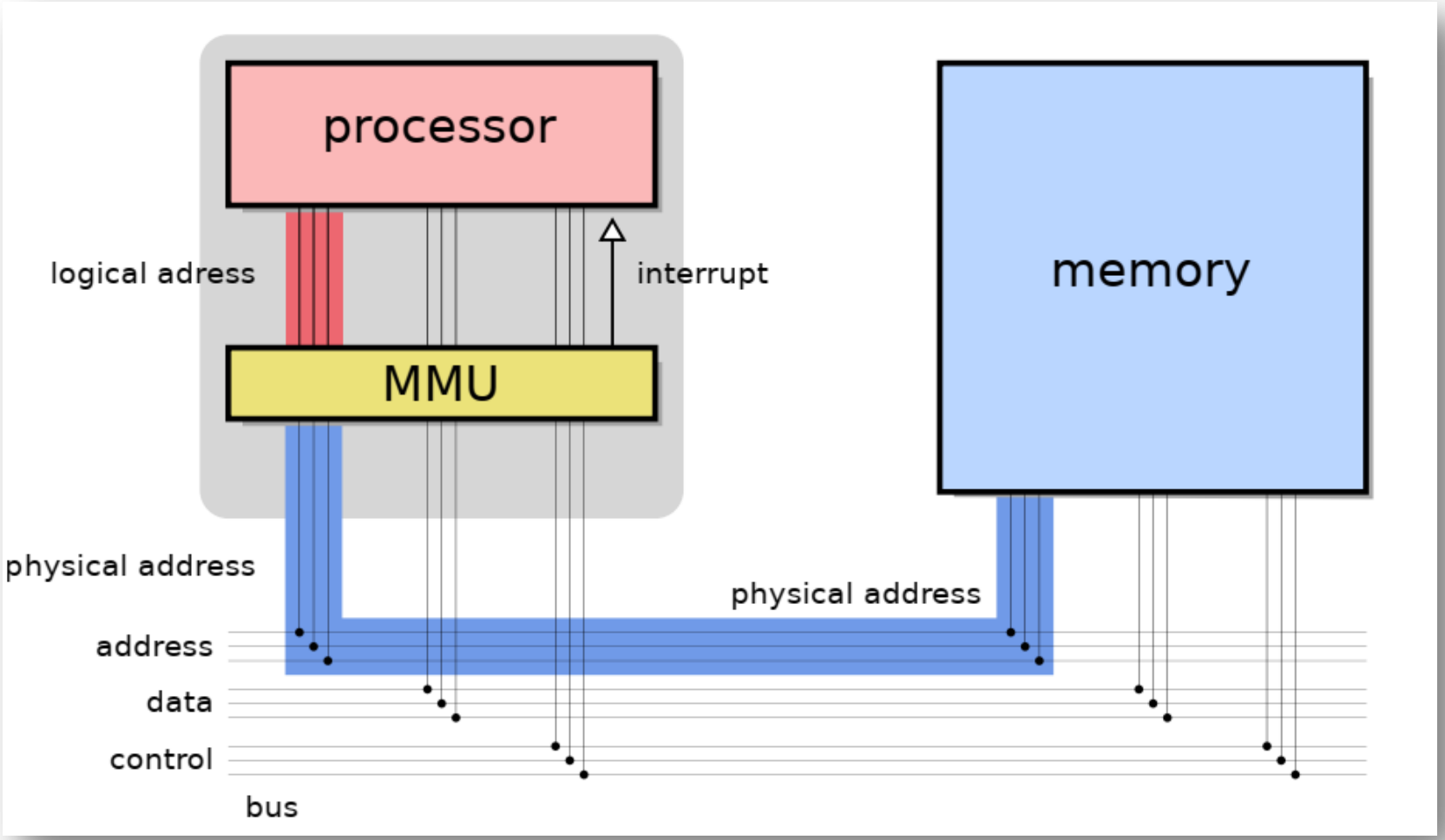
Na maioria dos processadores atuais, a MMU se encontra integrada ao chip da própria CPU.

- A MMU intercepta o acesso do processador ao barramento de endereços, recebendo os endereços lógicos gerados pelo mesmo e enviando os endereços físicos correspondentes.

- A MMU também tem acesso ao barramento de controle, para identificar operações de leitura e de escrita na memória.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

A memória Virtual



INSTITUTO FEDERAL
RIO GRANDE DO SUL

A memória Virtual

A implementação de Memória Virtual pode ser através de:

- Memória virtual por partições
- Memória virtual por segmentos
- Memória virtual por paginação

MEMÓRIA VIRTUAL POR PARTIÇÕES

Memória virtual por partições

Consiste em dividir a memória física em *N* partições (pedaços, blocos), que podem ter tamanhos iguais ou distintos, fixos ou variáveis.

Em cada partição da memória física é carregado um processo.

O processo ocupa uma partição de tamanho *N* bytes terá um espaço de endereçamento com até *N* bytes, com endereços lógicos no intervalo [0 . . . *N* - 1].

Nessa implementação, a MMU possui dois registradores:

- um ***registrador base*** (B), que define o endereço físico inicial da partição,
- e um ***registrador limite*** (L), que define o tamanho em bytes dessa partição.

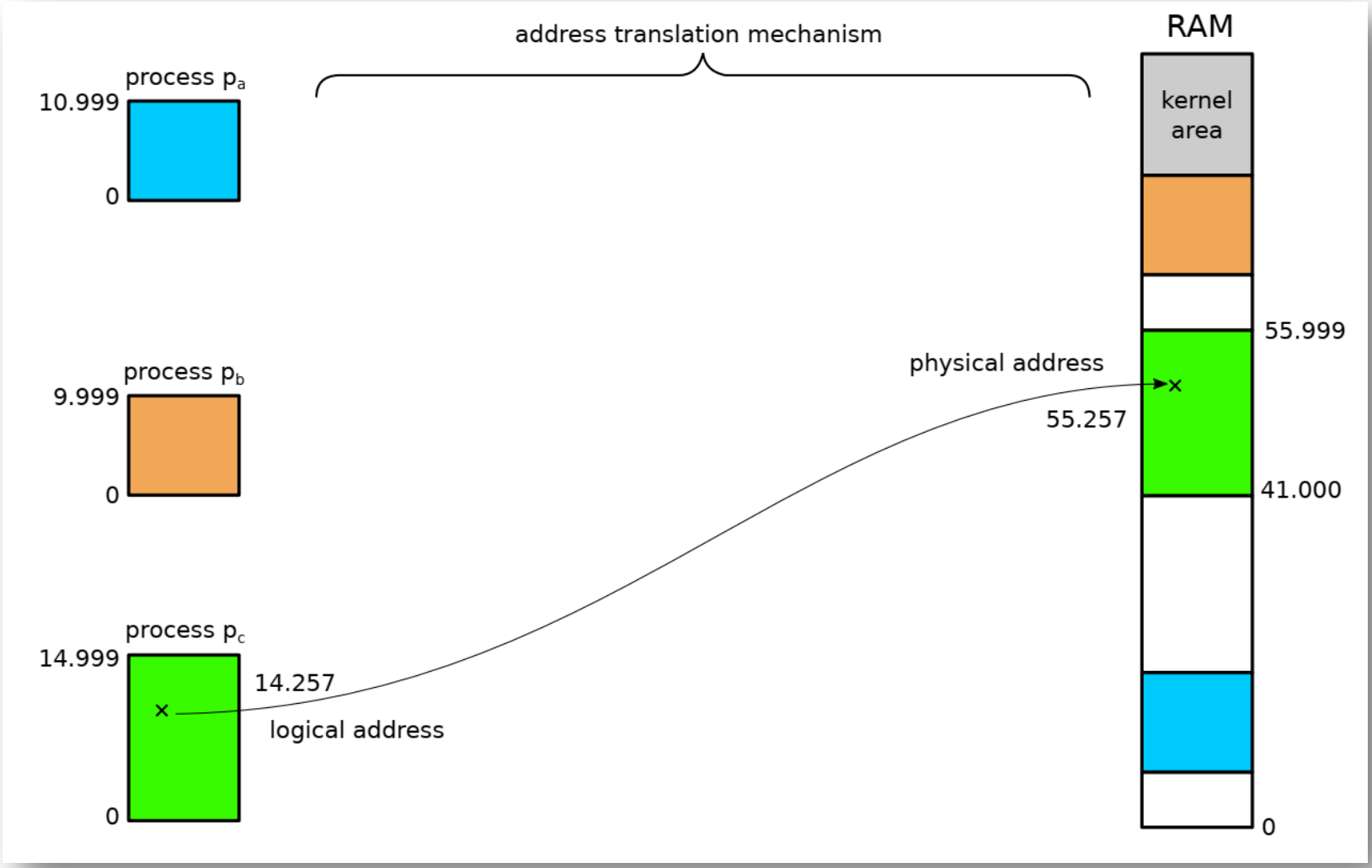
Memória virtual por partições

O Funcionamento é comparado e alterada para cada endereço lógico gerado (*el)* pelo processador, é comparado ao valor do registrador limite.

Caso seja menor que este ($el < L$), o endereço lógico é somado ao valor do registrador base, para a obtenção do endereço físico correspondente ($ef = el + B$).

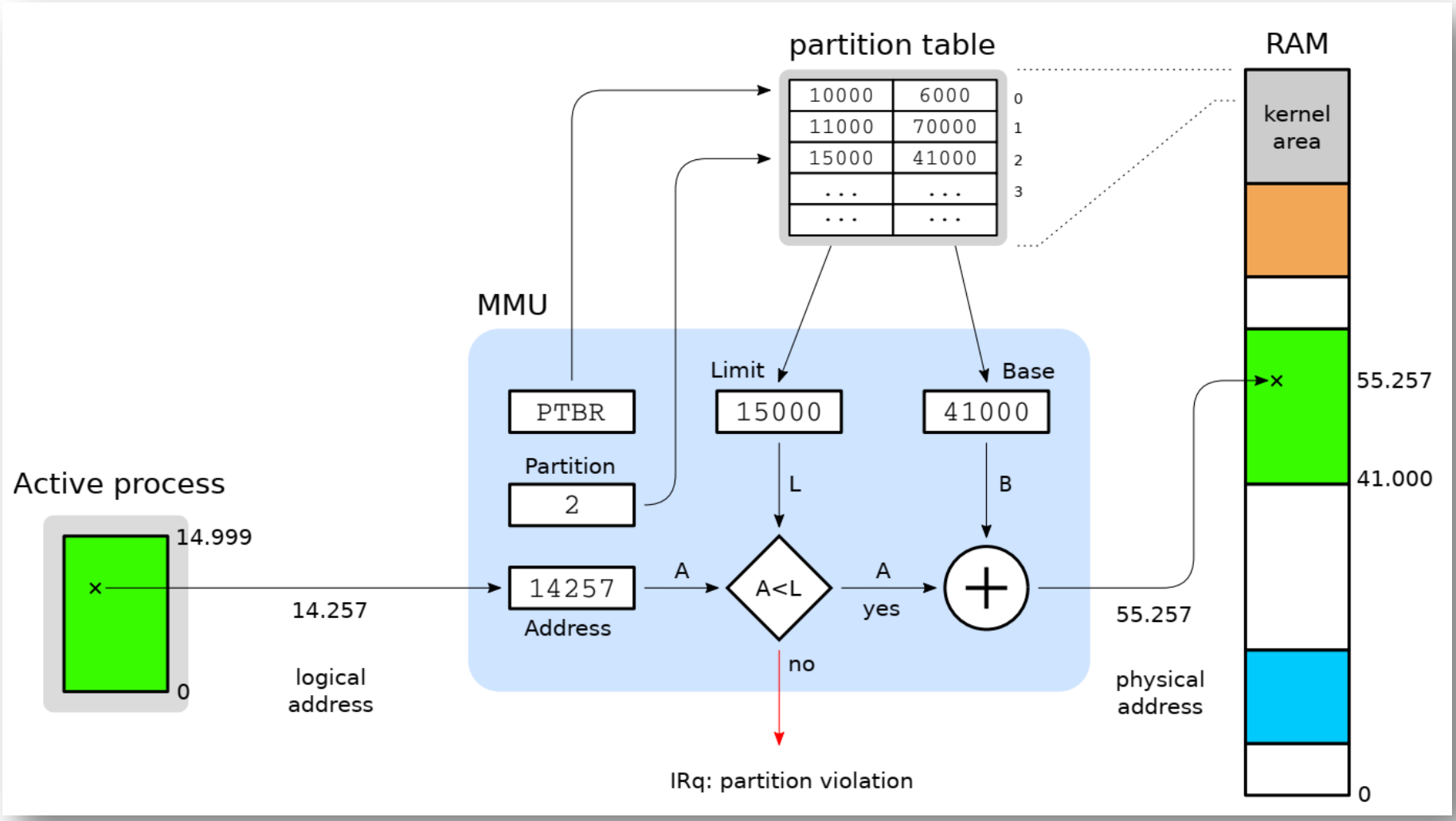
Caso contrário ($el \geq L$), uma interrupção é gerada pela MMU, indicando um endereço lógico inválido

Memória virtual por partições



O processo ativo, ocupando a partição 2, tenta acessar o endereço lógico 14.257. A MMU verifica que esse endereço é válido, pois é inferior ao limite da partição (15.000). Em seguida o endereço lógico é somado à base da partição (41.000) para obter o endereço físico correspondente (55.257).

Memória virtual por partições



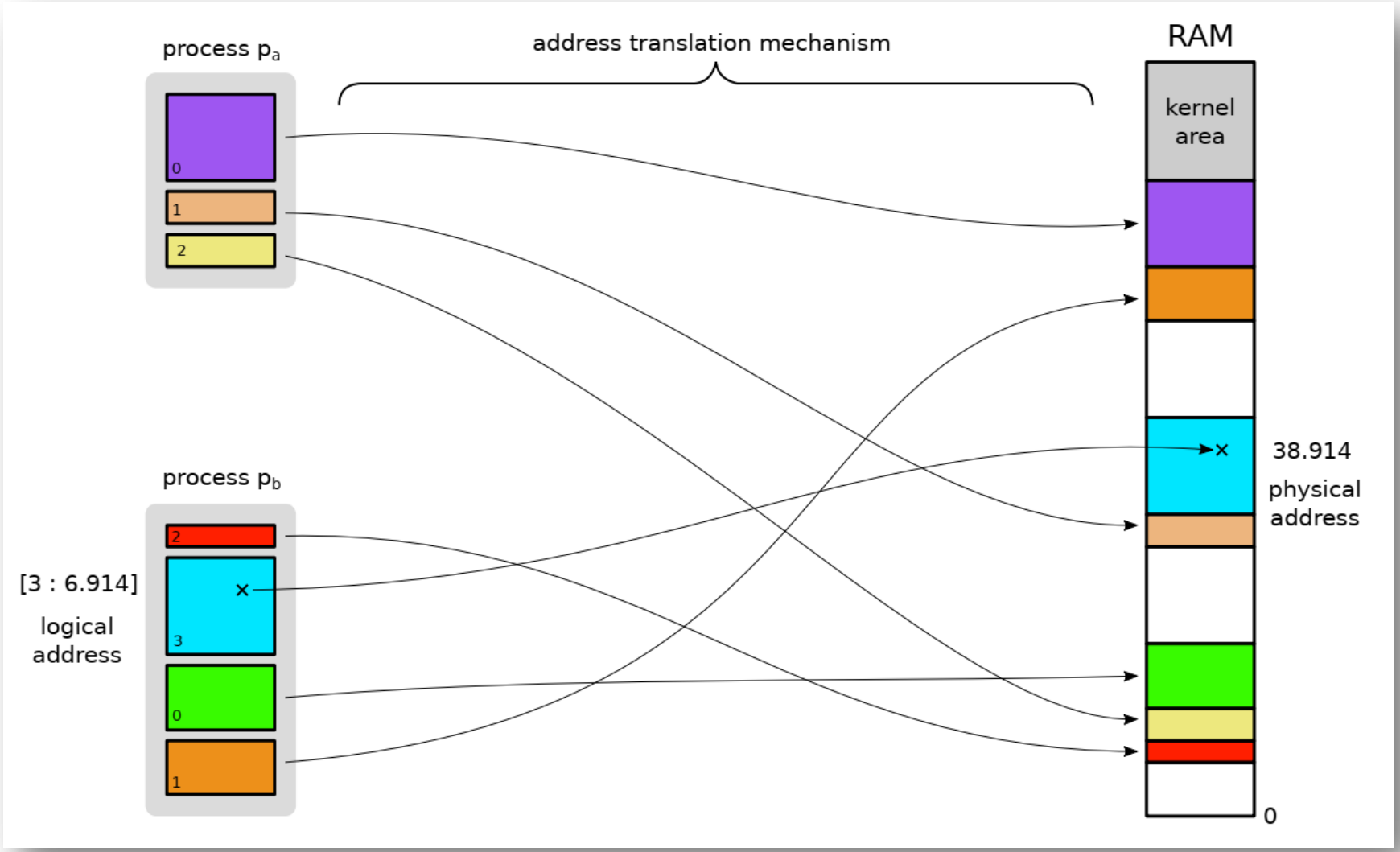
MEMÓRIA VIRTUAL POR SEGMENTOS

Memória virtual por Segementos

Nesta abordagem, o espaço de endereçamento de cada processo não é mais visto como uma sequência linear de endereços lógicos, mas como uma coleção de áreas de tamanhos diversos e políticas de acesso distintas, **denominadas *segmentos***.

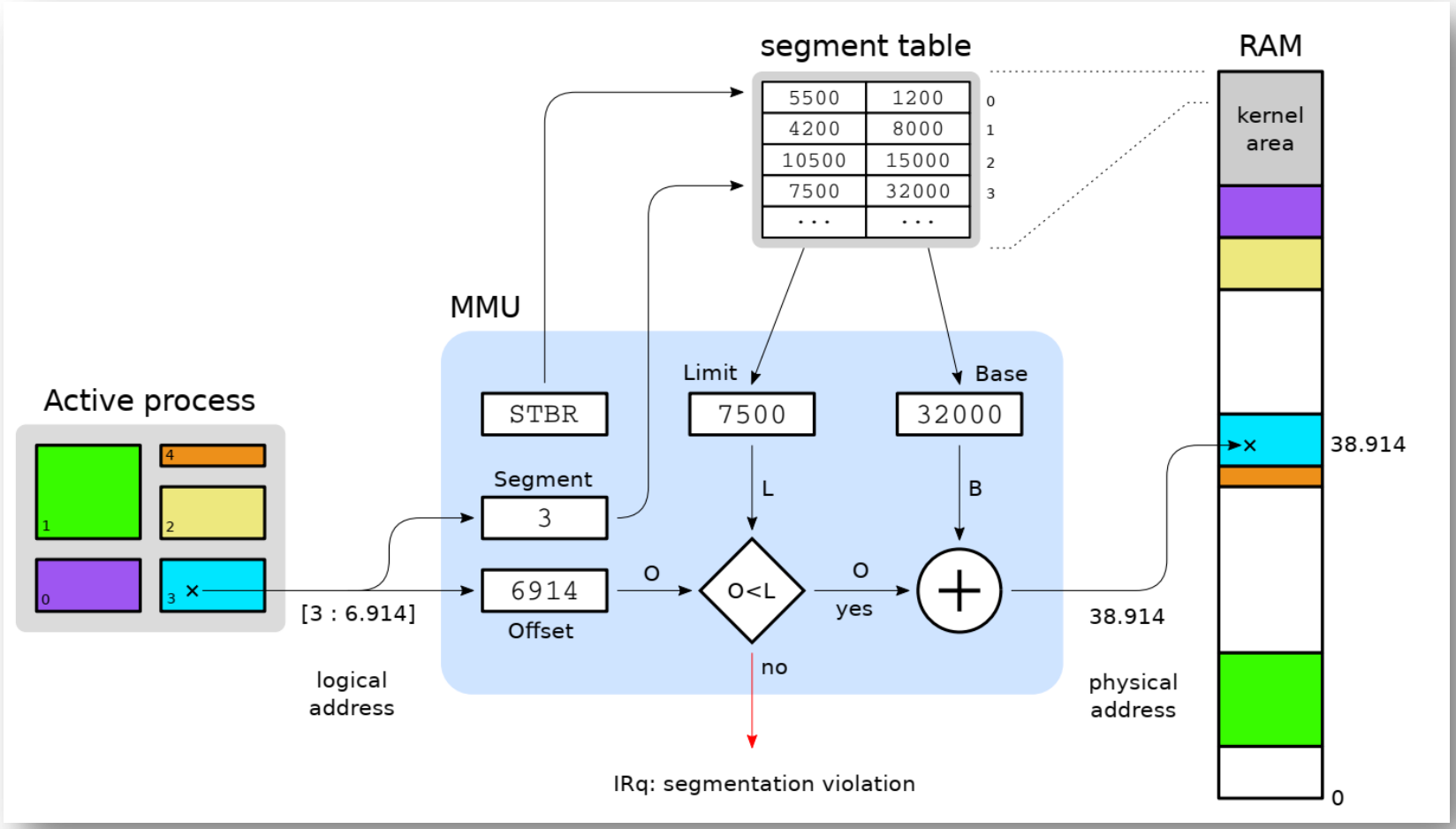
Os endereços lógicos gerados pelos processos são compostos por pares **[*segmento* : *o*ffset]**, onde *segmento* indica o número do segmento e *o*ff*set* indica a posição dentro daquele segmento.

Memória virtual por Segementos



O endereço lógico [3 : 6.914], que corresponde ao *offset* 6.914 no segmento 3 do processo p_b.
Nada impede de existir outros endereços 6.914 em outros segmentos do mesmo processo.

Memória virtual por Segementos



O modelo de memória virtual por segmentos foi muito utilizado nos anos 1970-90, sobretudo nas arquiteturas Intel e AMD de 32 bits. Pode deixar lento devido ao alto custo de acesso as tabelas de segmentos armazenados na memória RAM.

MEMÓRIA VIRTUAL POR PAGINAÇÃO

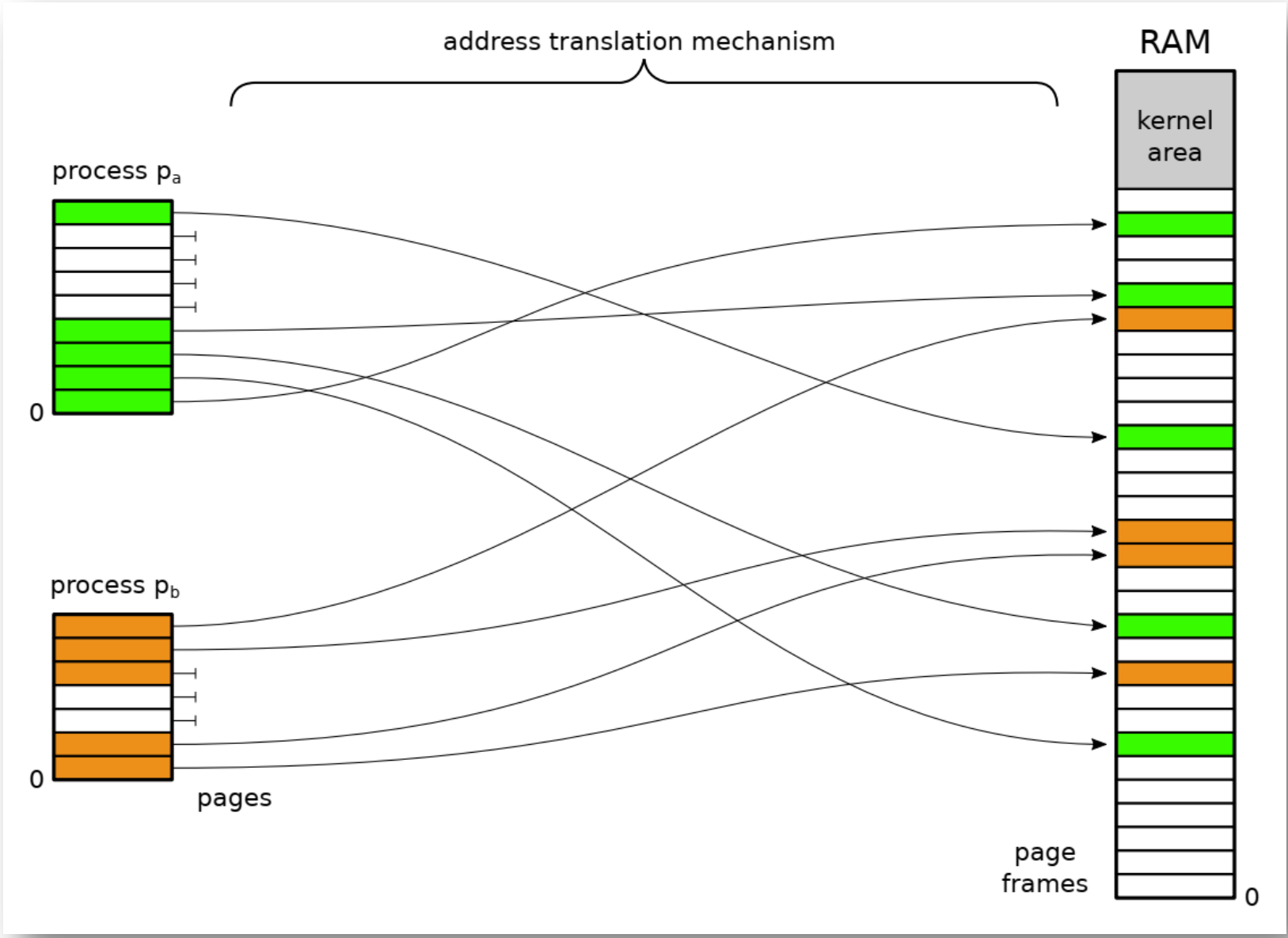
Memória virtual por Paginações

Nesse modelo, o espaço de endereçamento lógico dos processos é mantido linear e unidimensional.

Internamente, de forma transparente para o processador, o espaço de endereçamento lógico é dividido em pequenos blocos de mesmo tamanho, denominados p*áginas*.

Nas arquiteturas atuais, as páginas geralmente têm 4 KBytes (4.096 bytes)

Memória virtual por Paginações



Memória virtual por Paginações

Um processador Intel 80386 usa endereços lógicos de 32 bits e páginas com 4 KBytes;

Um endereço lógico de 32 bits é decomposto em um *offset* de 12 bits, que representa uma posição entre 0 e 4.095 dentro da página, e um número de página com 20 bits.

Dessa forma, podem ser endereçadas 2^{20} páginas com 2^{12} bytes cada (1.048.576 páginas com 4.096 bytes cada).

Memória virtual por Paginações

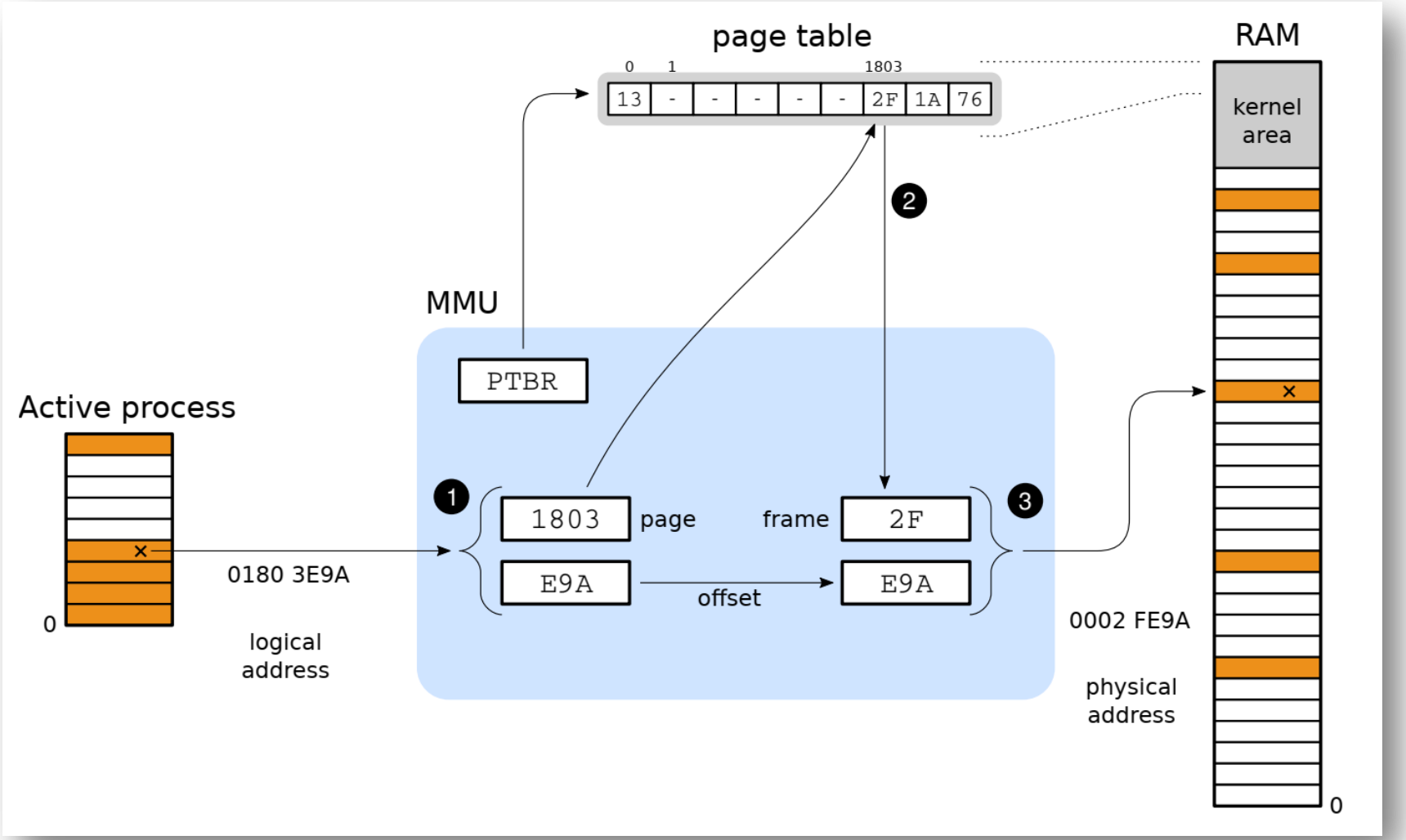
Eis um exemplo de decomposição do endereço lógico 01803*E*9*Ah* nesse sistema:

$$01803E9A_h \rightarrow \overbrace{0000\ 0001\ 1000\ 0000\ 0011}^{page:\ 20\ bits}\ \overbrace{1110\ 1001\ 1010_2}^{offset:\ 12\ bits}$$

$$\rightarrow \overbrace{0000\ 0001\ 1000\ 0000\ 0011}^{page=01803_h}\ \overbrace{1110\ 1001\ 1010_2}^{offset=E9A_h}$$

$$\rightarrow page = 01803_h \quad offset = E9A_h$$

Memória virtual por Paginações



Memória virtual por Paginações

Considerando que cada tabela de páginas tem 2^{20} páginas, uma tabela ocupará 4 MBytes de memória (4×2^{20} bytes) se for armazenada de forma linear na memória.

- No caso de processos pequenos, com muitas páginas não mapeadas, uma tabela de páginas linear poderá ocupar mais espaço na memória que o próprio processo.

Para resolver esse problema, são usadas *tabelas de páginas multiníveis*, estruturadas na forma de árvores: uma primeira tabela de páginas (ou *diretório de páginas*) contém ponteiros para tabelas de páginas secundárias e assim por diante.

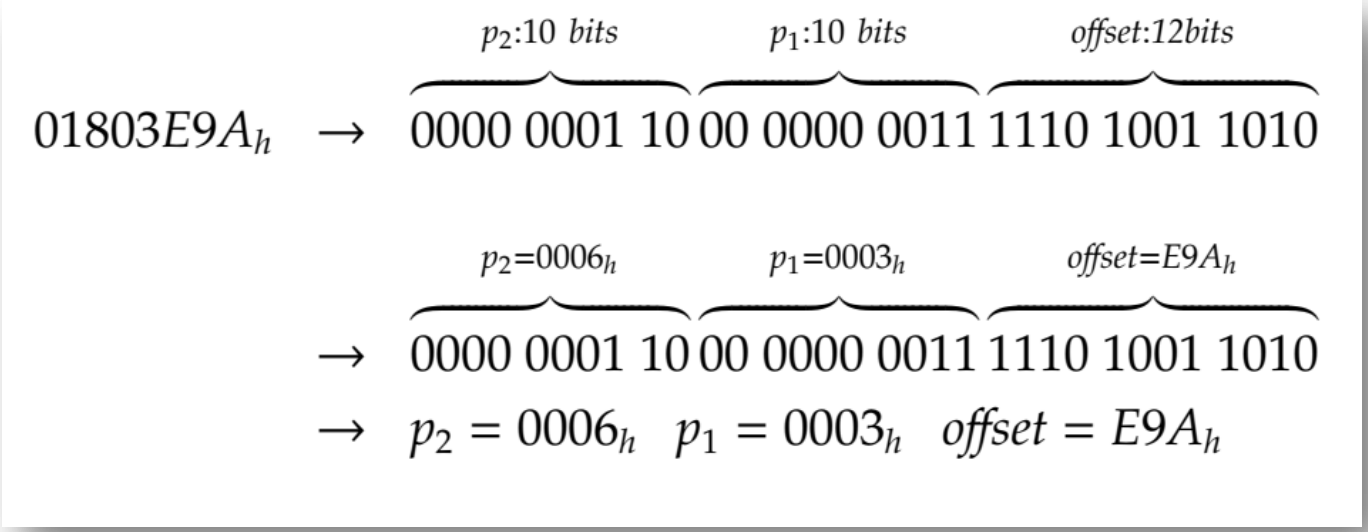
Quando uma tabela secundária não contiver entradas válidas, ela não precisa ser alocada; isso é representado por uma entrada nula na tabela principal.

Memória virtual por Paginações

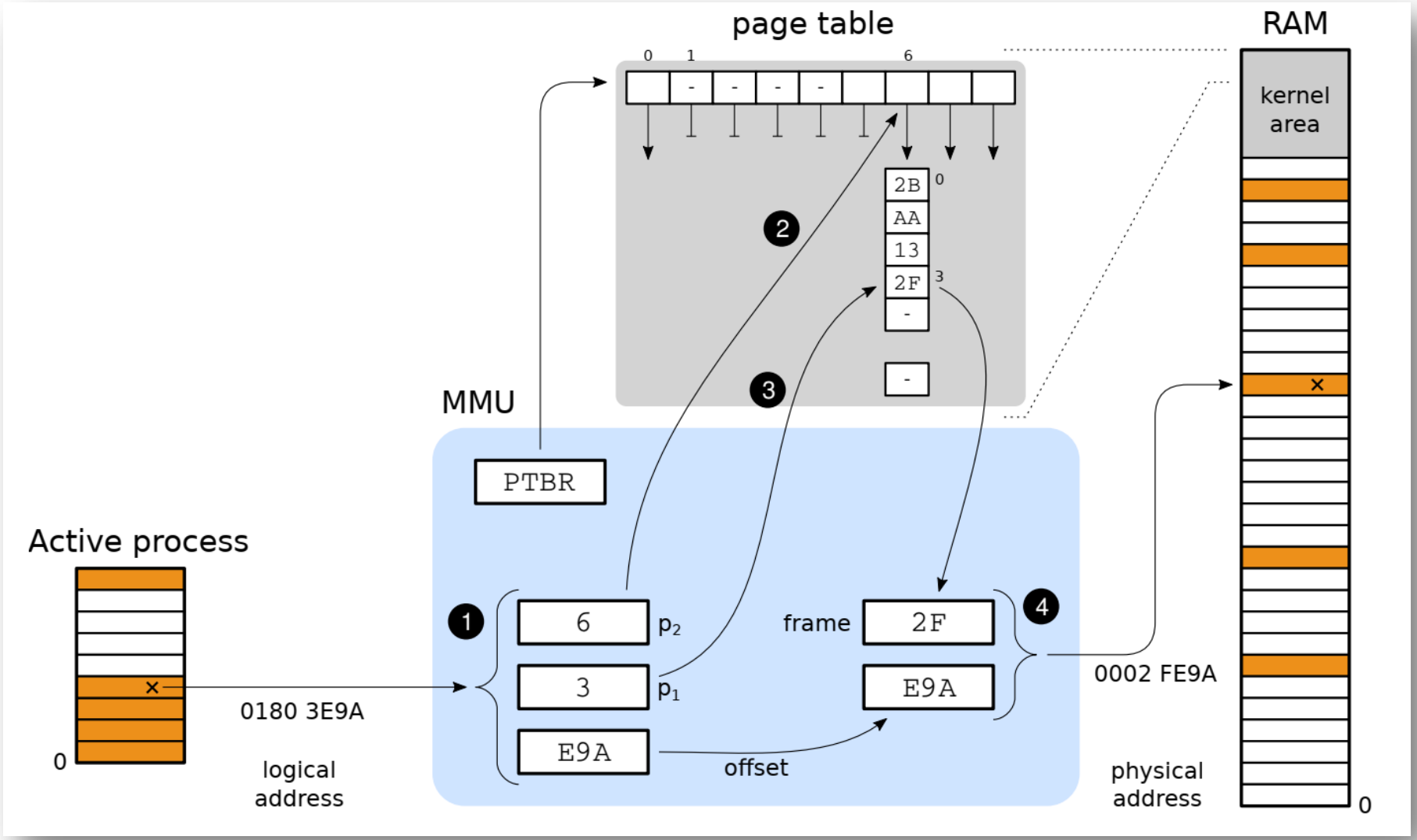
O número de página precisa ser dividido em duas ou mais partes, que são usadas como índices em cada nível de tabela, até encontrar o número de quadro desejado.

Considerando uma arquitetura de 32 bits com páginas de 4 KBytes, 20 bits são usados para acessar a tabela de páginas.

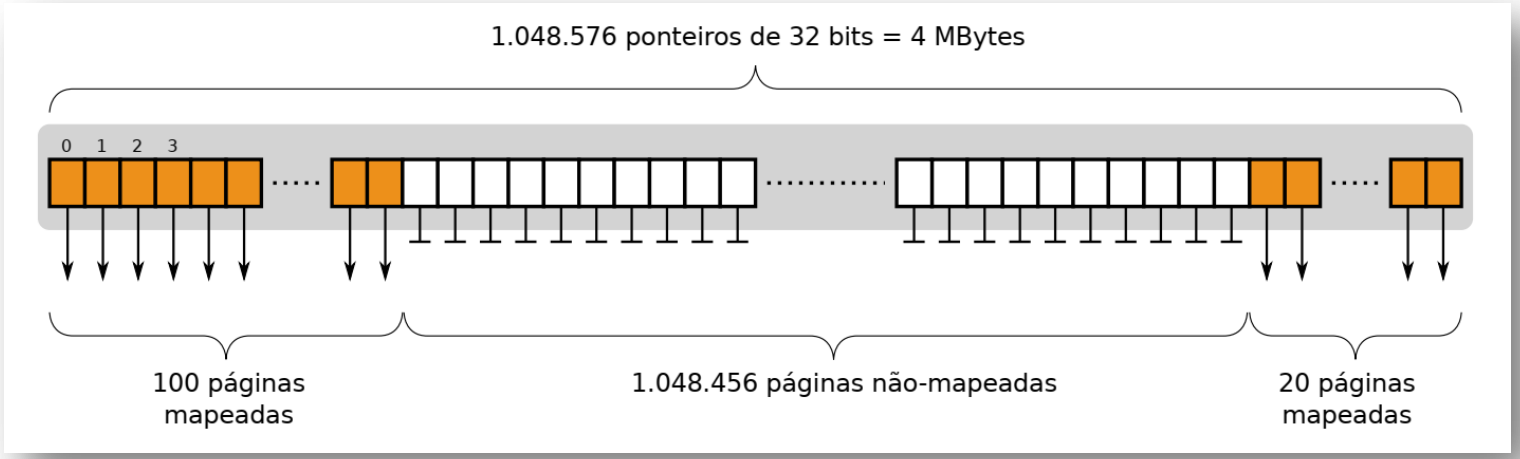
Esses 20 bits são divididos em dois grupos de 10 bits ($p1$ e $p2$) que são usados como índices em uma tabela de páginas com dois níveis:



Memória virtual por Paginações

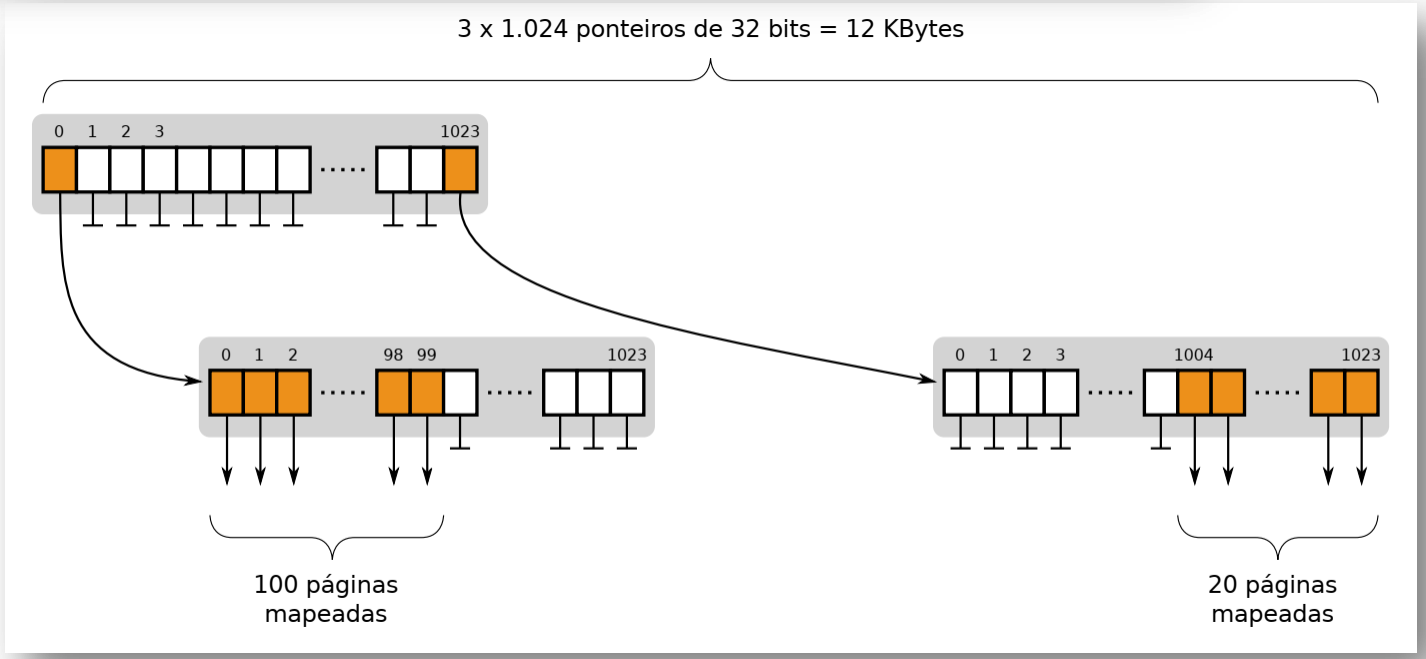


Memória virtual por Paginações



Modo Linear x Dois Níveis

Processo com 120 páginas em RAM, ou 480 KBytes .



Memória virtual por Paginações

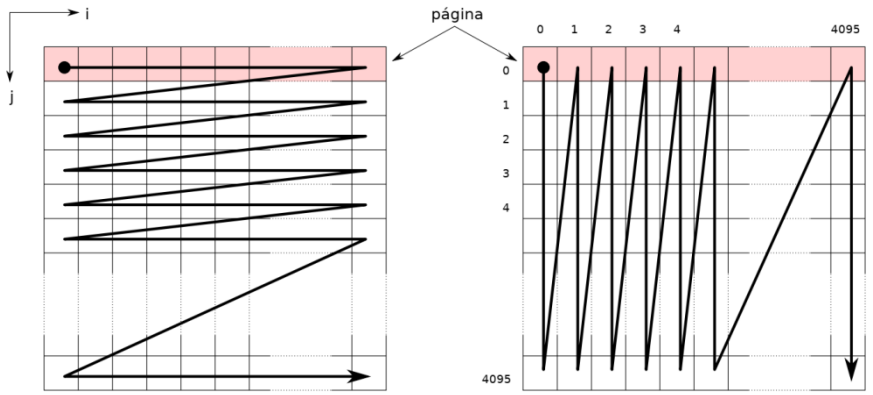
```c
#include <stdio.h>
#include <stdlib.h>

unsigned char buffer[4096][4096] ;

int main ()
 {
 int i, j ;

for (i=0; i<4096; i++)
 for (j=0; j<4096; j++)
 buffer[i][j]= (i+j) % 256 ;
 }
```

Qual é o mais rápido?




```c
#include <stdio.h>
#include <stdlib.h>

unsigned char buffer[4096][4096] ;

int main ()
 {
 int i, j ;

for (i=0; i<4096; i++)
 for (j=0; j<4096; j++)
 buffer[j][i]= (i+j) % 256 ;
 }
```

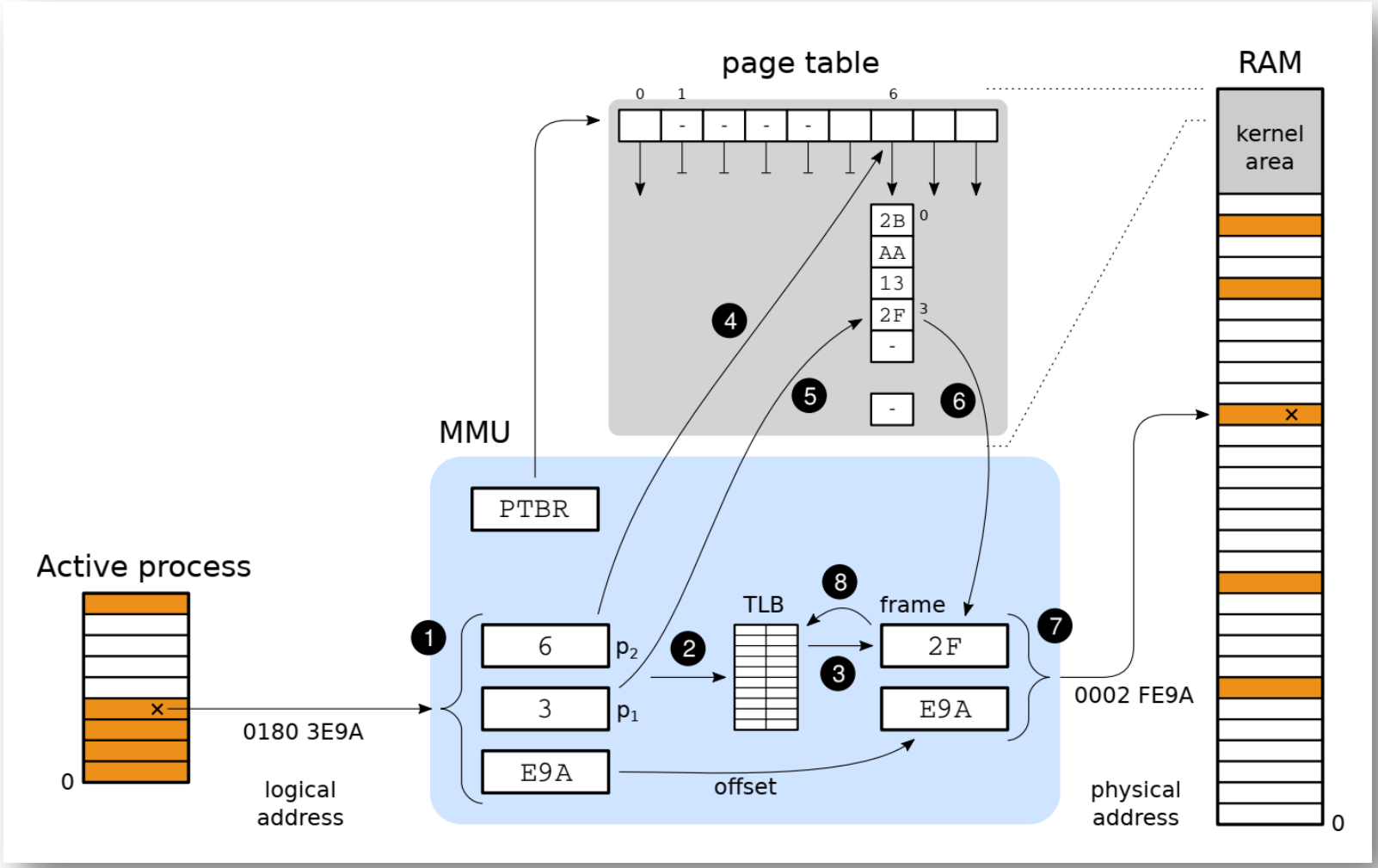
Memória virtual por Paginações

Cache da tabela de páginas

A estruturação das tabelas de páginas em vários níveis resolve o problema do espaço ocupado pelas tabelas, mas tem um efeito colateral nocivo**: aumenta fortemente o tempo de acesso à memória.**

Como as tabelas de páginas são armazenadas na memória RAM, cada acesso a um endereço de memória implica em mais acessos para percorrer a árvore de tabelas e encontrar o número de quadro desejado.

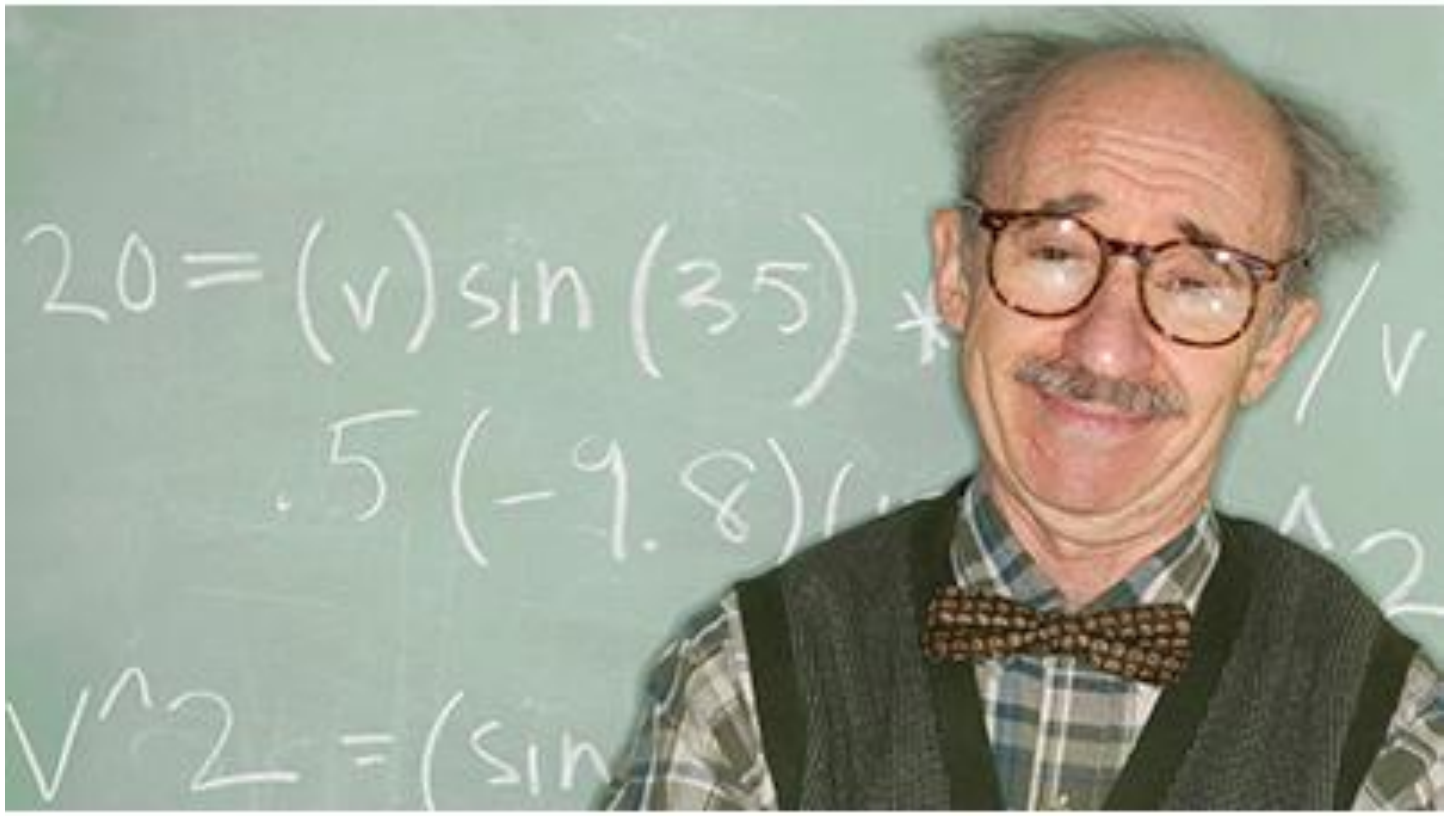
Memória virtual por Paginações



O cache de tabela de páginas na MMU, denominado **TLB (*Translation Lookaside Buffer*)** armazena as consultas recentes às tabelas de páginas do processo ativo.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Exercícios



INSTITUTO FEDERAL
RIO GRANDE DO SUL

Atividade 01

Explique a diferença entre endereços *lógicos* e endereços *físicos.*

Por que é implementado o mecanismo de memória lógica/virtual?

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Atividade 02

O que é MMU – *Memory Management Unit*?

O que é TLB - *Translation Lookaside Buffer?*

O que é um TCB – Task Control Block?

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Atividade 03

Seria possível e/ou viável implementar as conversões de endereços realizadas pela MMU em software, ao invés de usar um hardware dedicado? Justifique.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Atividade 03

Considerando a tabela de segmentos a seguir (com valores em decimal), calcule os endereços físicos correspondentes aos endereços lógicos 0:45, 1:100, 2:90, 3:1.900 e 4:200.

Segmento	0	1	2	3	4
Base	44	200	0	2.000	1.200
Limite	810	200	1.000	1.000	410

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Atividade 03

Considerando a tabela de segmentos a seguir (com valores em decimal), calcule os endereços físicos correspondentes aos endereços lógicos 0:45, 1:100, 2:90, 3:1.900 e 4:200.

Segmento	0	1	2	3	4
Base	44	200	0	2.000	1.200
Limite	810	200	1.000	1.000	410