




Plano de Ensino



- Sistemas de Numeração
- **Arquitetura de Computadores**
- Linguagem de Máquina
- Microcontroladores



Livro-Texto



- Livro-Texto:
 - » PEREIRA, Fabio. Microcontroladores PIC - Técnicas avançadas. 4ª ed. São Paulo: Erica, 2006.
- Bibliografia Complementar:
 - » GIMENEZ, S.P.. Microcontroladores 8051. 2ª ed. São Paulo: Pearson Education, 2005.

4. Hier. de Memória – Memória Cache



- **Cache**, em inglês: lugar seguro para esconder ou guardar algo.
- Nome usado para designar o nível de memória entre o processador e a memória principal.
- Este nome foi usado pela máquina que introduziu pioneiramente (no início dos anos 1960) este nível de memória (entre a memória principal e o processador)
- *Cache explora o princípio da localidade.*

4. Hier. de Memória – Memória Cache



- Assumamos as seguintes características de um sistema de memória extremamente simples:
 - » O processador sempre requisita uma única palavra.
 - » Existe apenas um nível de memória cache (L1).
 - » Os blocos de L1 são constituídos por somente uma palavra.

4. Hier. de Memória – Memória Cache



- Fazendo referência ao dado x_n .

antes	depois
x_4	x_4
x_1	x_1
x_{n-2}	x_{n-2}
x_{n-1}	x_{n-1}
x_2	x_2
	x_n
x_3	x_3

- » Como saber se uma informação está na cache?
- » Caso ela esteja, como encontrá-la?

4. Hier. de Memória – Memória Cache



▪ Mapeamento Direto →

- » Para cada palavra na cache, atribuir um endereço com base no endereço da palavra na memória principal.
- » A maioria das caches que usa mapeamento direto o faz usando o seguinte processo:
 - Os caches atuais estão organizados em linhas de 64 bytes (512 bits), no nosso exemplo temos uma cache de 1 palavra ou 1 byte (8 bits).
 - Portanto um cache de memória L2 de 64KB, por exemplo, será dividido em 65536 linhas.
 - $64 * 1024 \text{ bytes} = 65536 \text{ bytes}$
 - $65536 \text{ bytes} / 1 \text{ byte} = 65536 \text{ linhas}$

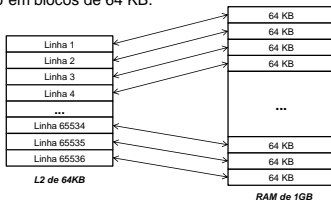
Linha 1
Linha 2
Linha 3
Linha 4
...
Linha 65534
Linha 65535
Linha 65536

L2 de 64KB

4. Hier. de Memória – Memória Cache



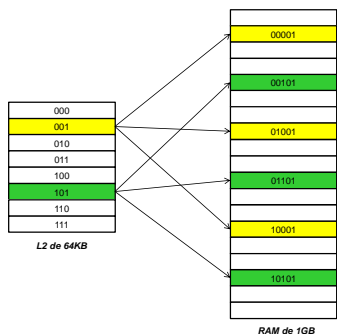
- O cache com mapeamento direto é a maneira mais simples de se criar um cache de memória.
- A memória RAM é dividida no mesmo número de linhas que existem dentro do cache de memória.
 - » No nosso exemplo, um micro com 1 GB de memória RAM seria dividido em blocos de 64 KB.



L2 de 64KB

RAM de 1GB

4. Hier. de Memória – Memória Cache



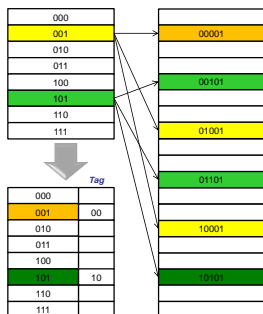
L2 de 64KB

RAM de 1GB

4. Hier. de Memória – Memória Cache



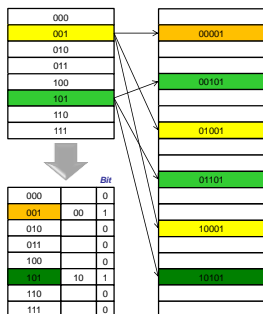
- Os **rótulos** são usados em conjunto com o endereço do mapeamento, de modo a compor o endereço completo, com relação à memória principal.



4. Hier. de Memória – Memória Cache



- O **bit de validade** valida se a informação contida naquele bloco da cache é válido ou não.
- Quando o processador é inicializado, a cache deve estar vazia.
 - Se o bit de validade = 0, a informação contida naquele bloco da cache não é válida.



4. Hier. de Memória – Memória Cache



- Exemplo: acesso a uma cache para leitura.
 - (1) Estado de inicial da cache após inicialização da máquina.

Índice	Bit	Tag	Informação
000	0		
001	0		
010	0		
011	0		
100	0		
101	0		
110	0		
111	0		

4. Hier. de Memória – Memória Cache



» (2) Referência ao endereço 10110: falta.

Índice	Bit	Tag	Informação
000	0		
001	0		
010	0		
011	0		
100	0		
101	0		
110	0		
111	0		

4. Hier. de Memória – Memória Cache



- » (2) Referência ao endereço 10110: falta.
- » Tratamento da falta: buscar no nível inferior (neste exemplo, a memória principal) o bloco com endereço 10110.

Índice	Bit	Tag	Informação
000	0		
001	0		
010	0		
011	0		
100	0		
101	0		
110	1	10	Memória (10110)
111	0		

4. Hier. de Memória – Memória Cache



» (3) Referência ao endereço 11010: falta.

Índice	Bit	Tag	Informação
000	0		
001	0		
010	0		
011	0		
100	0		
101	0		
110	1	10	Memória 10110
111	0		

4. Hier. de Memória – Memória Cache



- » (3) Referência ao endereço 11010: falta.
- » Tratamento da falta: buscar no nível inferior (neste exemplo, a memória principal) o bloco com endereço 11010.

Índice	Bit	Tag	Informação
000	0		
001	0		
010	1	11	Memória (11010)
011	0		
100	0		
101	0		
110	1	10	Memória (10110)
111	0		

4. Hier. de Memória – Memória Cache



- » (4) Referência ao endereço 10000: falta.

Índice	Bit	Tag	Informação
000	0		
001	0		
010	1	11	Memória (11010)
011	0		
100	0		
101	0		
110	1	10	Memória (10110)
111	0		

4. Hier. de Memória – Memória Cache



- » (4) Referência ao endereço 10000: falta.
- » Tratamento da falta: buscar no nível inferior (neste exemplo, a memória principal) o bloco com endereço 10000.

Índice	Bit	Tag	Informação
000	1	10	Memória (10000)
001	0		
010	1	11	Memória (11010)
011	0		
100	0		
101	0		
110	1	10	Memória (10110)
111	0		

4. Hier. de Memória – Memória Cache



» (5) Referência ao endereço 00011: falta.

Índice	Bit	Tag	Informação
000	1	10	Memória (10000)
001	0		
010	1	11	Memória (11010)
011	0		
100	0		
101	0		
110	1	10	Memória (10110)
111	0		

4. Hier. de Memória – Memória Cache



» (5) Referência ao endereço 00011: falta.

» Tratamento da falta: buscar no nível inferior (neste exemplo, a memória principal) o bloco com endereço 00011.

Índice	Bit	Tag	Informação
000	1	10	Memória (10000)
001	0		
010	1	11	Memória (11010)
011	1	00	Memória (00011)
100	0		
101	0		
110	1	10	Memória (10110)
111	0		

4. Hier. de Memória – Memória Cache



» (6) Referência ao endereço 10010: falta.

Índice	Bit	Tag	Informação
000	1	10	Memória (10000)
001	0		
010	1	11	Memória (11010)
011	1	00	Memória (00011)
100	0		
101	0		
110	1	10	Memória (10110)
111	0		

4. Hier. de Memória – Memória Cache



- » (6) Referência ao endereço 10010: falta.
- » Tratamento da falta: buscar no nível inferior (neste exemplo, a memória principal) o bloco com endereço 10010, sobrescrevendo o valor inicial.

Índice	Bit	Tag	Informação
000	1	10	Memória (10000)
001	0		
010	1	10	Memória (10010)
011	1	00	Memória (00011)
100	0		
101	0		
110	1	10	Memória (10110)
111	0		

4. Hier. de Memória – Número de Bits da Cache



- O número de entradas em uma cache deve ser potência de dois.
- O número total de bits necessários à implementação de uma cache é função do tamanho da cache e do tamanho da memória principal (*tamanho do endereço*).

Índice	Bit	Tag	Informação

4. Hier. de Memória – Número de Bits da Cache



- Uma memória principal com endereços de 32 bits referenciando bytes.
- Uma cache mapeada diretamente com 2^n palavras e blocos de uma palavra.
- Então:
 - » Tamanho dos *tags* = $32 - (n + 2)$ bits, onde 2 bits são usados para o deslocamento e n para o índice.
 - » Portanto, o número total de bits em uma cache mapeada diretamente é:
 - $2^n \times (\text{tamanho do bloco} + \text{tamanho do tag} + \text{tamanho do campo de validade})$
 - Tamanho da cache em questão será $2^n \times (32 + (32 - n - 2) + 1) = 2^n \times (63 - n)$.

$2^0 = 1$
$2^1 = 2$
$2^2 = 4$
$2^3 = 8$
$2^4 = 16$
$2^5 = 32$
$2^6 = 64$
$2^7 = 128$
$2^8 = 256$
$2^9 = 512$
$2^{10} = 1.024$
$2^{11} = 2.048$
$2^{12} = 4.096$
$2^{13} = 8.192$
$2^{14} = 16.384$
$2^{15} = 32.768$
$2^{16} = 65.536$
$2^{17} = 131.072$
$2^{18} = 262.144$
$2^{19} = 524.288$
$2^{20} = 1.048.576$

4. Hier. de Memória – Número de Bits da Cache



- *Exemplo 1:* quantos bits são necessários para implementar uma cache mapeada diretamente com 64KB entradas, e blocos de uma palavra, ligada a uma memória cujo endereço tem 32 bits?
- Solução:
 - » $64\text{ KB} = 64\text{ Kpalavras} = 2^{16}\text{ palavras}$
 - » Cada bloco tem 32 bits de informação mais o campo do tag = $32 - 16 - 2$ bits, mais o bit de validade
 - » Portanto, o tamanho da cache é
 - » $2^{16} \times (32 + (32 - 16 - 2) + 1) = 2^{16} \times 47 = 3008 \times 2^{10} = 3008\text{ Kbits}$

4. Hier. de Memória – Número de Bits da Cache



- *Exemplo 2:* quantos bits são necessários para implementar uma cache mapeada diretamente com 256KB entradas, e blocos de uma palavra, ligada a uma memória cujo endereço tem 64 bits?
- Solução:
 - » $256\text{ KB} = 256\text{ Kpalavras} = 2^{18}\text{ palavras}$
 - » Cada bloco tem 64 bits de informação mais o campo do tag = $64 - 18 - 2$ bits, mais o bit de validade
 - » Portanto, o tamanho da cache é
 - » $2^{18} \times (64 + (64 - 18 - 2) + 1) = 2^{18} \times 109 = 2^{10} \times 2^8 \times 109 = 2^{10} \times 256 \times 109 = 27904\text{ Kbits}$ ou 27,25 Mbits

4. Hier. de Memória – Bloco de Controle



- O Bloco de Controle é responsável em detectar um acerto (hit) ou processar a falta (fail); neste último caso buscando os dados na memória principal ou em outra cache imediatamente abaixo na hierarquia.
- Se a cache informar um acerto:
 - » O processamento segue como se a informação tivesse sido obtida na memória principal.

4. Hier. de Memória – Bloco de Controle



- Se a cache informar uma falta:
 - » O processador deve ser parado (congelando o conteúdo de todos os registradores).
 - » Um controlador separado ajuda no tratamento das faltas geradas no acesso à cache, comandando a busca da informação (bloco) na memória principal ou na cache de próximo nível.
 - » Uma vez que o dado tenha sido obtido, a execução é reiniciada no ciclo que gerou a falta no acesso a cache.
 - » O processamento de uma falta na cache cria uma parada no processamento similar às paradas do pipeline: todos os registradores temporários e visíveis ao programador são congelados, enquanto a informação é transferida da memória.

4. Hier. de Memória – Bloco de Controle



- Se a cache informar falta de instruções:
 - » Se o acesso a uma instrução resultar em falta, então o conteúdo do IR não é válido (uma vez que o PC é incrementado no primeiro ciclo de relógio, tanto na versão multiciclo quanto na versão pipeline).
 - » É necessário comandar uma leitura no nível inferior da hierarquia da memória, usando o PC (no caso pipeline, recursos extras de HW serão necessários: subtrator ou deslocador).
 - » Instruir a memória para realizar a leitura e esperar a resposta (uma leitura demora muitos ciclos) e então escrever a palavra (instrução) na cache.

4. Hier. de Memória – Bloco de Controle



- Se a cache informar falta de dados:
 - » Os passos para o tratamento de faltas no acesso a dados (cache de dados) são essencialmente os mesmos que os usados no tratamento de uma falta de instrução.
 - » Também é necessário parar o processador até que o dado necessário esteja disponível na cache.

4. Hier. de Memória – Bloco de Controle



- Resumo: se a cache informar falta de instruções ou dados:
 1. Enviar à memória o valor original do PC.
 2. Comandar uma leitura da unidade de memória e esperar o resultado.
 3. Escrever o resultado da leitura na entrada da cache, escrevendo também nessa entrada, no campo tag, os bits de mais alta ordem do endereço, e setando o bit de validade.
 4. Reiniciar a execução da instrução a partir do passo número 1, gerando uma nova busca da instrução na cache (mas desta vez, com a certeza de que ela será encontrada).

4. Hier. de Memória – Write-Through



- Suponha que na execução de uma instrução o dado seja escrito somente na cache de dados. Isto causará uma inconsistência: após, a escrita na cache, a memória principal terá um valor diferente daquele que foi escrito na cache.
- No esquema *write-through* não há a necessidade de se considerar se uma escrita gera uma falta ou um acerto na cache. Basta escrever a palavra na cache; e replicar sempre o valor para os níveis de memória inferiores na hierarquia.
 - » O esquema *write-through* não favorece o desempenho.
 - » Qualquer escrita na cache faz com que a memória principal seja escrita também.

4. Hier. de Memória – Write-Through com Buffer



- *Buffer* de escrita armazena o dado enquanto este aguarda para ser escrito na memória.
- Após escrever o dado na cache e no *buffer* de escrita, o processador pode continuar a execução das instruções.
- Se o *buffer* de escrita estiver cheio quando o processador tiver que executar uma instrução de escrita, o processador precisa parar, até que haja posição disponível no *buffer*, descarregando este *buffer* de escrita para a memória.
 - » Se a velocidade da memória para completar as escritas for menor que a taxa à qual o processador está gerando as escritas, nenhum *buffer* (por maior que seja) conseguirá resolver o problema.

4. Hier. de Memória – Write-Back



- No esquema *write-back*, quando ocorre uma escrita, o novo valor é escrito apenas no bloco da cache. Tal bloco somente será escrito na memória principal quando ele tiver que ser substituído na cache.
- O esquema pode aumentar bastante o desempenho, principalmente quando o processador puder gerar escritas tão rapidamente quanto estas puderem ser tratadas pela memória principal.



Sistemas Microprogramados –

Ciência da Computação

clayton.valdo@anhanguera.com