Arquitetura e Organização de Computadores

Capítulo 14

Paralelismo em nível de instruções e processadores superescalares

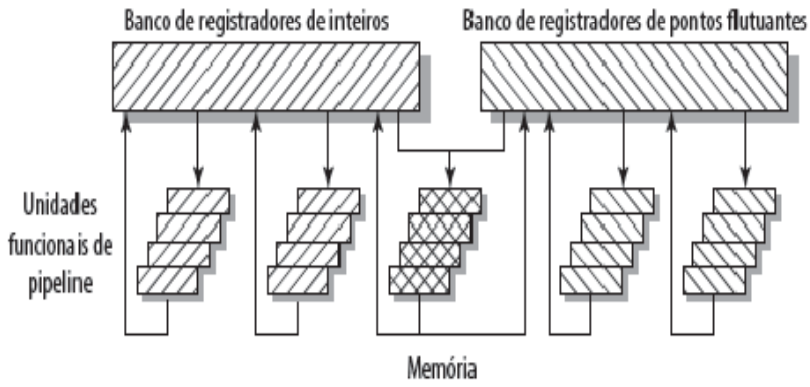
O que é superescalar?

- Instruções comuns (aritméticas, load/store, desvio condicional) podem ser iniciadas e executadas independentemente.
- Igualmente aplicável a RISC e CISC.
- Na prática, normalmente RISC.

Por que superescalar?

- Maioria das operações é sobre quantidades escalares (ver notas sobre RISC).
- Melhore essas operações para obter uma melhoria geral.

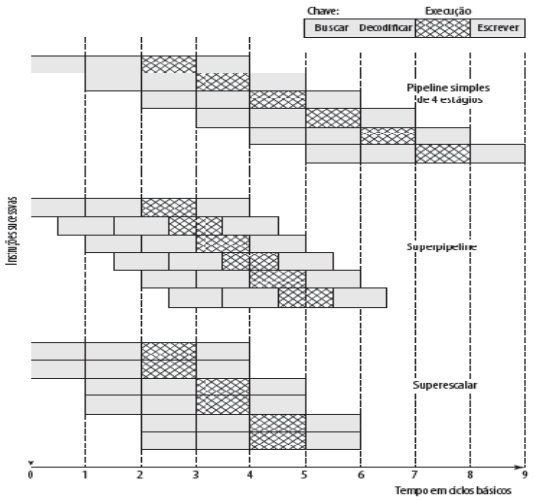
Organização superescalar geral



Superpipeline

- Muitos estágios de pipeline precisam de menos da metade de um ciclo de clock.
- Dupla velocidade de clock interno recebe duas tarefas por ciclo de clock externo.
- Superescalar permite busca/execução paralela.

Superescalar *versus* superpipeline



Limitações

- Paralelismo em nível de instrução.
- Otimização baseada em compilador.
- Técnicas de hardware.
- Limitado por:
  - Verdadeira dependência de dados.
  - Dependência procedural.
  - Conflitos de recursos.
  - Dependência de saída.
  - Antidependência.

Dependência de dados verdadeira

- ADD r1, r2 (r1 := r1+r2;).
- MOVE r3,r1 (r3 := r1;).
- Pode ler e decodificar segunda instrução em paralelo com a primeira.
- NÃO pode executar segunda instrução até que a primeira termine.

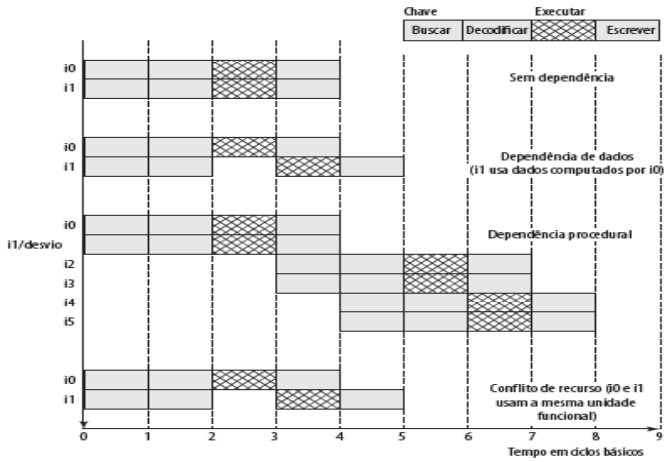
Dependência procedural

- Não pode executar instruções após um desvio em paralelo com instruções antes de um desvio.
- Além disso, se o tamanho da instrução não for fixo, as instruções precisam ser decodificadas para descobrir quantas leituras são necessárias.
- Isso impede leituras simultâneas.

Conflito de recursos

- Duas ou mais instruções exigindo acesso ao mesmo recurso ao mesmo tempo.
  - P.e., duas instruções aritméticas.
- Pode duplicar recursos.
  - P.e., têm duas unidades aritméticas.

Efeito das dependências



Questões de projeto

- Paralelismo em nível de instruções:
  - Instruções em uma sequência são independentes.
  - Execução pode ser sobreposta.
  - Governado por dependência de dados e procedural.
- Paralelismo de máquina:
  - Habilidade de tirar proveito do paralelismo em nível de instrução.
  - Controlado pelo número de pipelines paralelos.

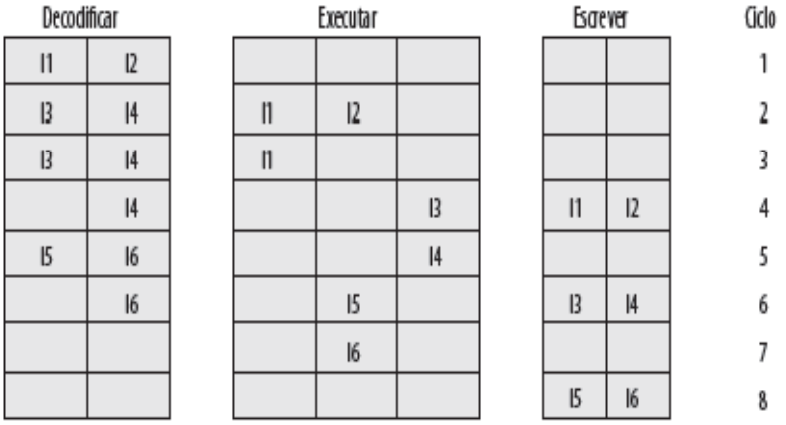
Política de emissão de instruções

- Ordem em que as instruções são lidas.
- Ordem em que as instruções são executadas.
- Ordem em que as instruções mudam de registradores e memória.

Emissão em-ordem, conclusão em-ordem

- Emite instruções na ordem em que ocorrem.
- Não muito eficiente.
- Pode ler mais de uma instrução.
- Instruções precisam parar, se necessário.

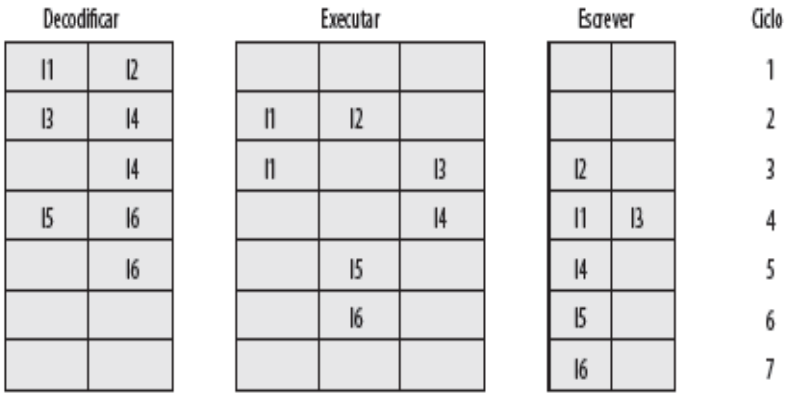
Emissão em-ordem, conclusão em-ordem (diagrama)



Emissão em ordem, conclusão fora-de-ordem

- Dependência de saída:
  - R3:= R3 + R5 (I1).
  - R4:= R3 + 1 (I2).
  - R3:= R5 + 1 (I3).
  - I2 depende do resultado de I1 – dependência de dados.
  - Se I3 conclui antes de I1, o resultado de I1 estará errado – dependência de saída (leitura-escrita).

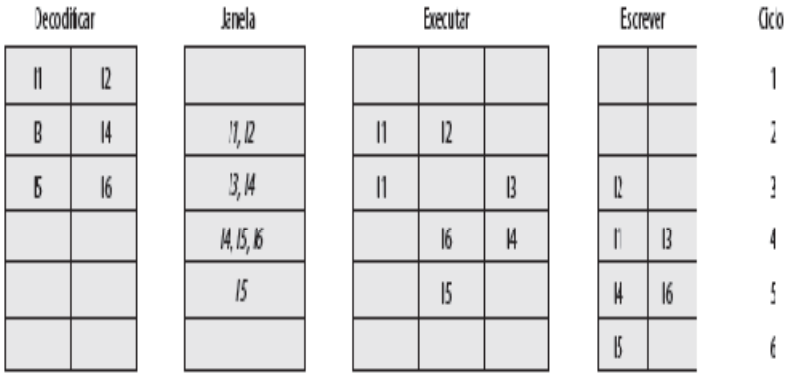
Emissão em-ordem, conclusão fora-de-ordem (diagrama)



Emissão fora-de-ordem, conclusão fora-de-ordem

- Desacopla o pipeline de decodificação do pipeline de execução.
- Pode continuar a buscar e decodificar até que o pipeline esteja cheio.
- Quando uma unidade funcional se tornar disponível, uma instrução poderá ser executada.
- Como as instruções foram decodificadas, o processador pode antecipar.

Emissão fora-de-ordem, conclusão fora-de-ordem (diagrama)



Antidependência

- Dependência escrita-escrita:
  - R3:=R3 + R5 (I1).
  - R4:=R3 + 1 (I2).
  - R3:=R5 + 1 (I3).
  - R7:=R3 + R4 (I4).
  - I3 não pode concluir antes que I2 inicie, enquanto I2 precisa de um valor em R3 e I3 muda R3.

Buffer de reordenação

- Armazenamento temporário para resultados.
- Colocados no banco de registradores na ordem do programa.

Renomeação de registradores

- Ocorrem saída e antidependências porque conteúdo do registrador pode não refletir a ordenação correta do programa.
- Pode resultar em uma parada de pipeline.
- Registradores alocados dinamicamente.
  - Ou seja, registradores não são nomeados especificamente.

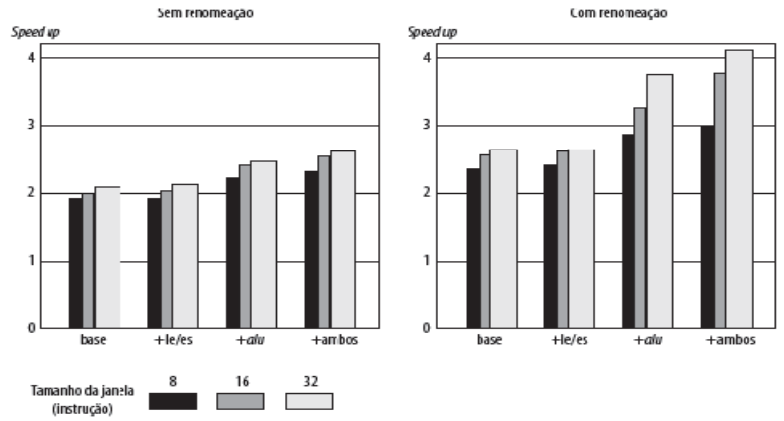
Exemplo de renomeação de registrador

- R3b:=R3a + R5a (I1).
- R4b:=R3b + 1 (I2).
- R3c:=R5a + 1 (I3).
- R7b:=R3c + R4b (I4).
- Sem subscrito refere-se a registrador lógico na instrução.
- Com subscrito é registrador de hardware alocado.
- Observe R3a R3b R3c.
- Alternativa: *Scoreboarding*.
  - Técnica de *bookkeeping.*
  - Permite execução da instrução sempre que não depende das instruções anteriores e sem hazards estruturais.

Paralelismo de máquina

- Duplicação de recursos.
- Emissão fora de-ordem.
- Renomeação:
- Não vale a pena duplicar funções sem renomear registrador.
- Precisa de janela de instrução grande o suficiente (mais de 8).

Acelerações de organizações de máquina sem dependências procedurais



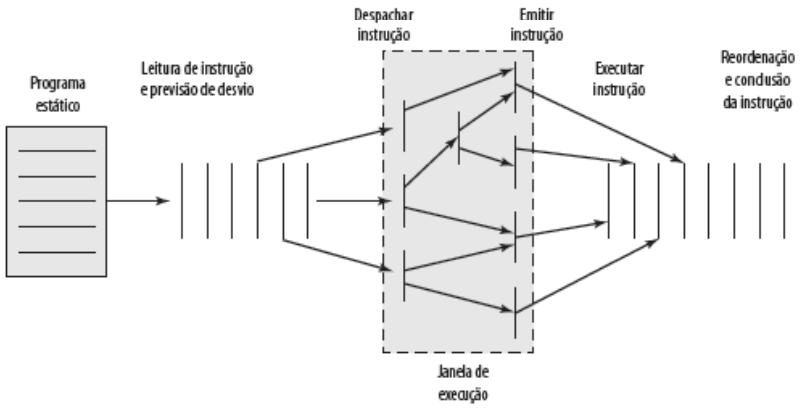
Previsão de desvio

- 80486 busca próxima instrução sequencial após instrução de desvio e alvo de desvio.
- Provoca atraso de dois ciclos se desvio for tomado.

RISC – Desvio atrasado

- Calcule resultado do desvio antes que instruções não usáveis sejam pré-buscadas.
- Sempre executa instrução única imediatamente após desvio.
- Mantém pipeline cheio enquanto busca novo fluxo de instruções.
- Não muito bom para superescalar.
  - Múltiplas instruções precisam ser executadas no *delay slot.*
  - Problemas de dependência de instrução.
- Reverter para previsão de desvio.

Execução superescalar



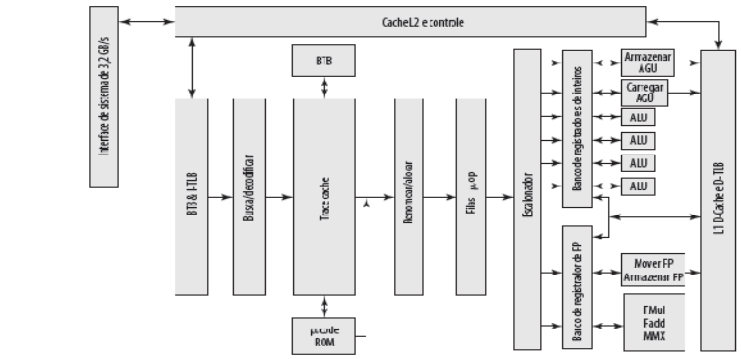
Implementação superescalar

- Busca simultaneamente múltiplas instruções.
- Lógica para determinar dependências verdadeiras envolvendo valores de registrador.
- Mecanismos para comunicar esses valores.
- Mecanismos para iniciar múltiplas instruções em paralelo.
- Recursos para execução paralela de múltiplas instruções.
- Mecanismos para executar estado do processo na ordem correta.

Pentium 4

- 80486 − CISC
- Pentium − alguns componentes superescalares.
  - Duas unidades de execução inteiras separadas.
- Pentium Pro – superescalar completo.
- Modelos subsequentes refinam e melhoram o projeto superescalar.

Diagrama de blocos do Pentium 4

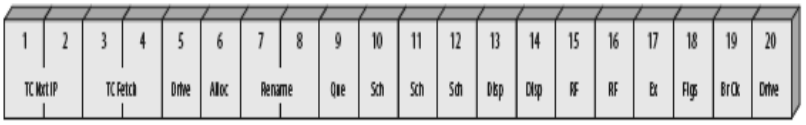


AGU = unidade de geração de endereço (do inglês *address generation unit*)
BTB = buffer de alvo de desvio (do inglês *branch target buffer*)
FP = pontos flutuantes (do inglês *floating points*)
D-TLB = TLB de dados
I-TLB = TLB de instruções

Operação do Pentium 4

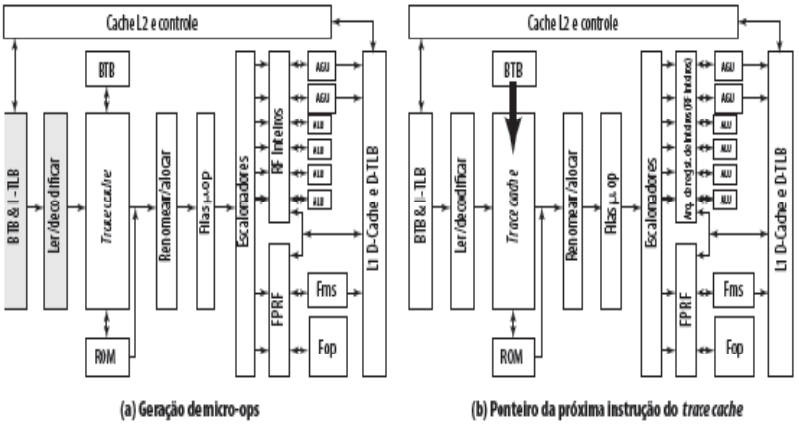
- Lê instruções da memória na ordem do programa estático.
- Traduz instrução para uma ou mais instruções RISC de tamanho fixo (micro-operações).
- Executa micro-operações.
- Executa micro-operações em pipeline superescalar.
  - Micro-operações podem ser executadas fora da ordem.
- Coloca resultados das micro-perações no conjunto de registradores na ordem original do fluxo do programa.
- Shell CISC mais externo com núcleo RISC mais interno.
- Pipeline do núcleo RISC mais interno com pelo menos 20 estágios.
  - Algumas micro-operações exigem múltiplos estágios de execução.
    - Pipeline mais longa.
  - Compare com pipeline de cinco estágios no x86 até o Pentium.

Pipeline do Pentium 4



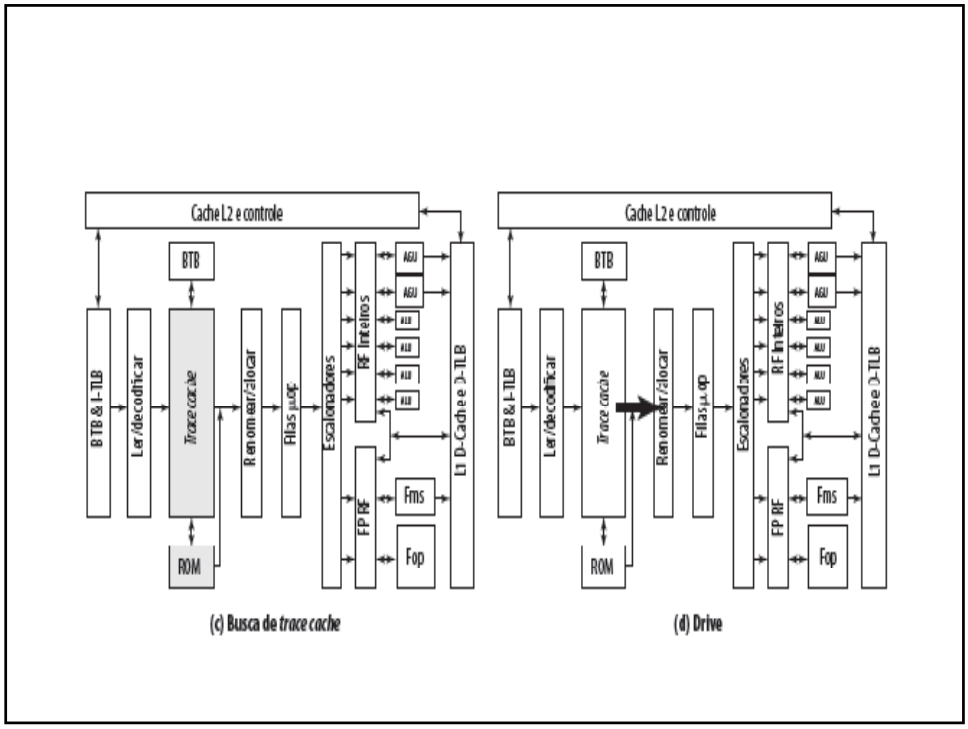
TC Next IP = ponteiro da próxima instrução da trace cache
TC Fetch = busca na trace cache
Alloc = alocar
Rename = renomeação de registrador
Que = fila micro-op
Sch = escalonamento micro-op
Disp = despachar
RF = banco de registradores
Ex = executar
Flgs = flags
Br Ck = verificar desvio

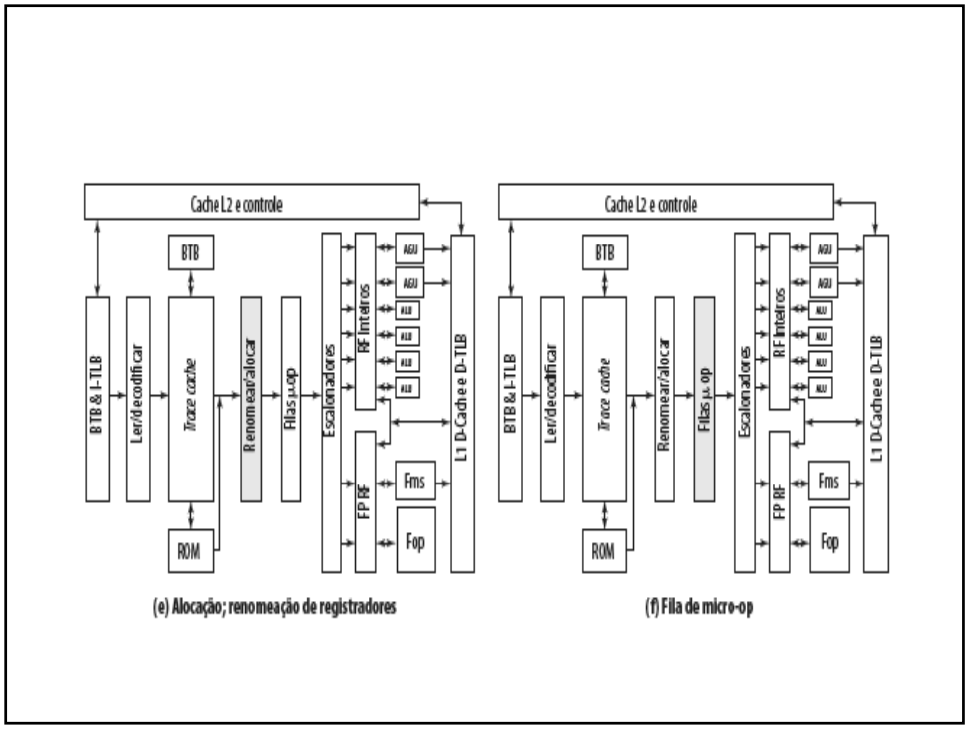
Operação do pipeline do Pentium 4

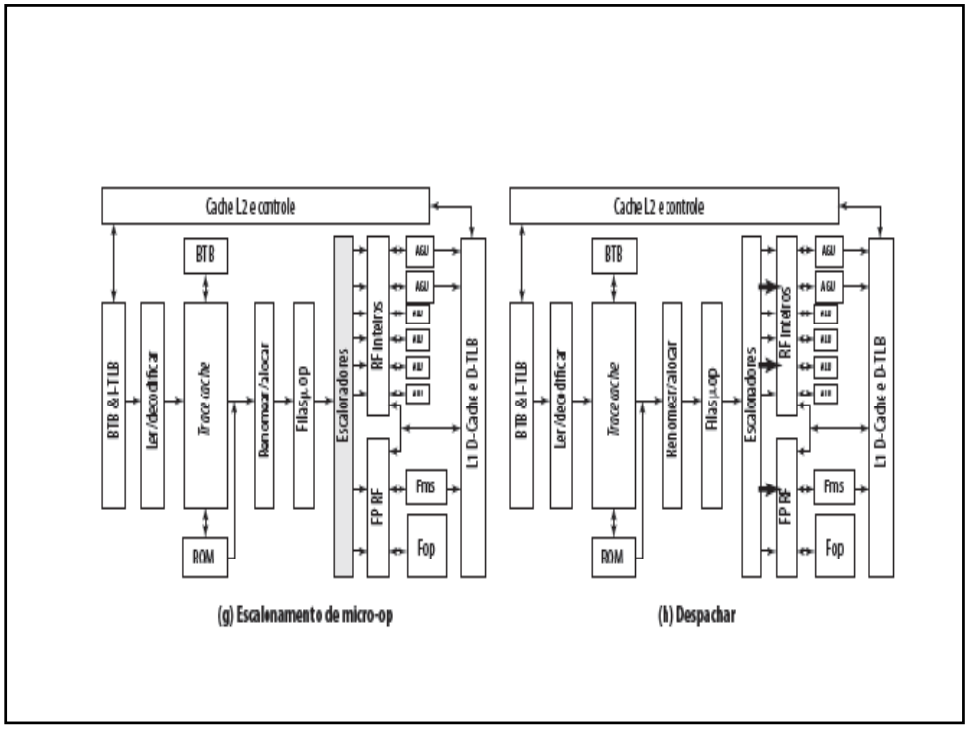


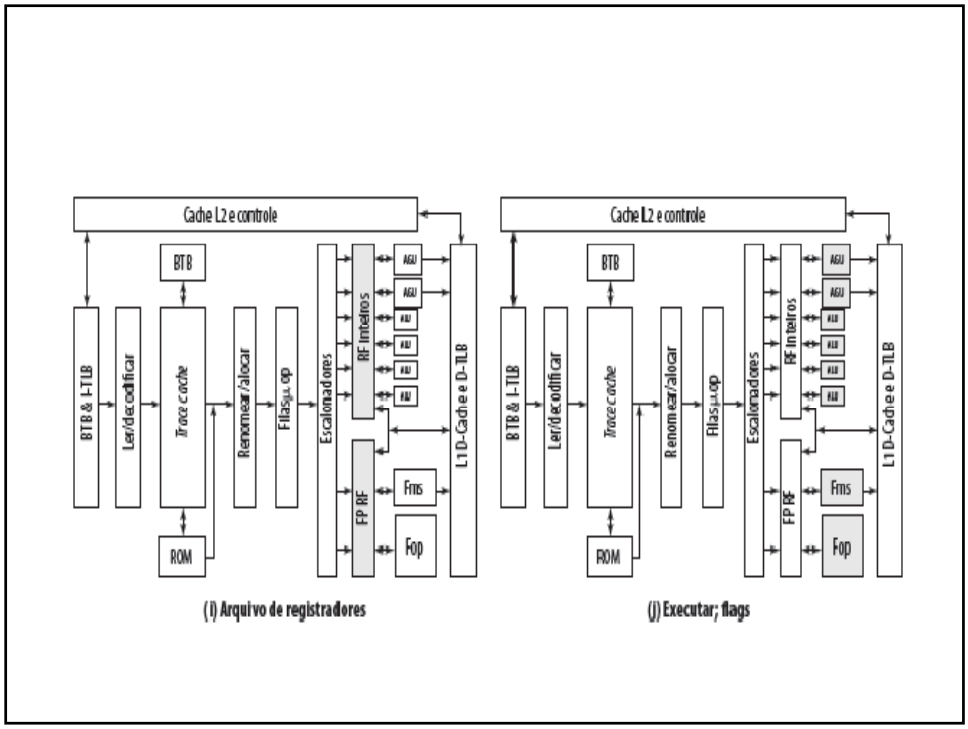
(a) Geração de micro-ops

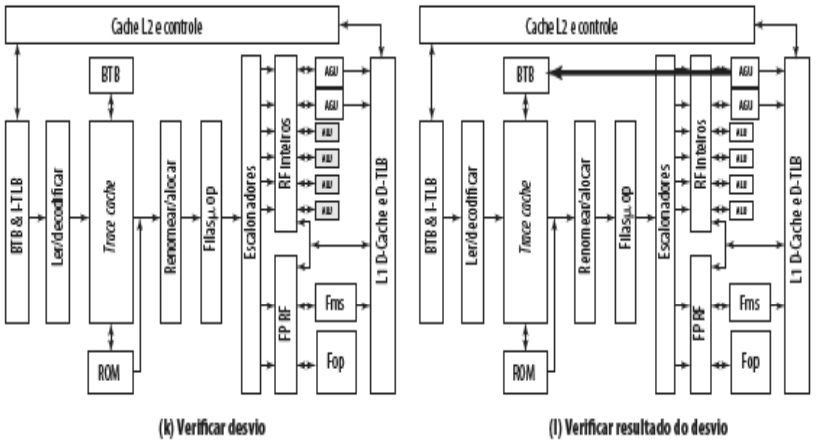
(b) Ponteiro da próxima instrução do *trace cache*


Cache L2 e controle
BTB
BTB & I-TLB
Ler/decodificar
Trace cache
Renomear/alocar
Filas µop
Escalonadores
RF Inteiros
FP RF
AGU
ALU
MUL
Fmx
Fop
L1 D-Cache e D-TLB
ROM
(c) Busca de trace cache
(d) Drive


Cache L2 e controle
BTB
BTB & I-TLB
Ler/decodificar
Trace cache
Renomear/alocar
Filas µop
Escalonadores
RF Inteiros
FP RF
AGU
ALU
MUL
Fmx
Fop
L1 D-Cache e D-TLB
ROM
(e) Alocação; renomeação de registradores
(f) Fila de micro-op


Cache L2 e controle
BTB
BTB & I-TLB
Ler/decodificar
Trace cache
Renomear/alocar
Filas μop
Escaloradores
RF inteiros
FP RF
AGU
ALU
Fms
Fop
L1 D-Cache e D-TLB
ROM
(g) Escalonamento de micro-op
Cache L2 e controle
BTB
BTB & I-TLB
Ler/decodificar
Trace cache
Renomear/alocar
Filas μop
Escaloradores
RF inteiros
FP RF
AGU
ALU
Fms
Fop
L1 D-Cache e D-TLB
ROM
(h) Despachar

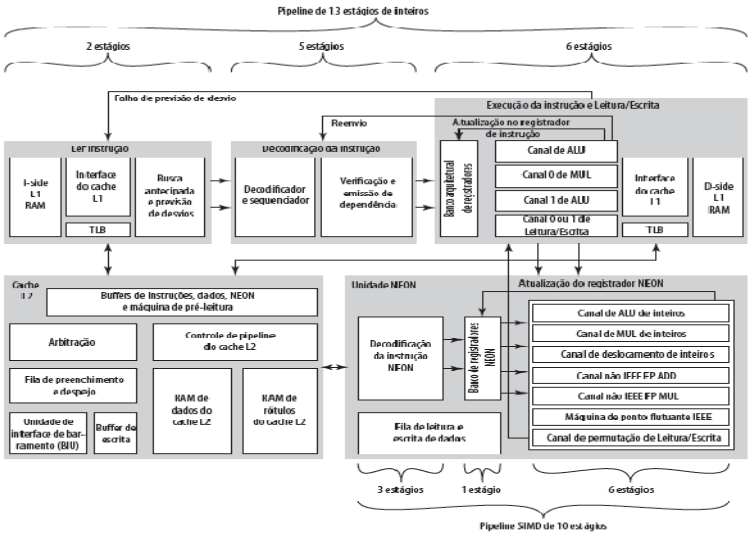

Cache L2 e controle
BTB
BTB & I-TLB
Ler/decodificar
Trace cache
Renomear/alocar
Filas μop
Escaloradores
RF inteiros
FP RF
AGU
ALU
Fms
Fop
L1 D-Cache e D-TLB
ROM
(i) Arquivo de registradores
Cache L2 e controle
BTB
BTB & I-TLB
Ler/decodificar
Trace cache
Renomear/alocar
Filas μop
Escaloradores
RF inteiros
FP RF
AGU
ALU
Fms
Fop
L1 D-Cache e D-TLB
ROM
(j) Executar; flags



ARM CORTEX-A8

- ARM refere-se ao Cortex-A8 como processadores de aplicação.
- Processador embutido rodando sistema operacional complexo.
  - Aplicações sem fio, de eletrônica de consumo e imagens.
  - Telefones móveis, caixas set-top, consoles de jogos e navegação de automóveis/sistemas de entretenimento.
- Três unidades funcionais.
- Dual, emissão em ordem, pipeline de 13 estágios.
  - Mantém requisito de potência mínimo.
  - Emissão fora de-ordem precisa de lógica extra, consumindo potência extra.
- Figura 14.11 mostra os detalhes do pipeline principal do Cortex-A8.
- Separa unidade SIMD (Single-Instruction-Multiple-Data).
  - Pipeline de 10 estágios.

Diagrama de blocos do ARM Cortex-A8



Unidade de busca de instruções

- Prevê fluxo de instruções.
- Lê instruções da cache de instruções L1.
  - Até quatro instruções por ciclo.
- Para buffer para pipeline de decodificação.
- Unidade de busca inclui cache de instruções L1.
- Buscas de instruções especulativas.
- Instrução de desvio ou excepcional causa esvaziamento do pipeline.
- Estágios:
- F0 Unidade de geração de endereço gera endereço virtual.
  - Normalmente, o próximo na sequência.
  - Também pode ser endereço de alvo do desvio.
- F1 Usado para ler instruções da cache de instruções L1.
  - Na leitura paralela, endereço usado para acessar arrays de previsão de desvio.
- F3 Dados de instrução são colocados na fila de instruções.
  - Se há previsão de desvio, novo endereço de destino enviado à unidade de geração de endereço.
- Previsor de desvios com histórico global de dois níveis:
  - Branch Target Buffer (BTB) e Global History Buffer (GHB).
- Pilha de retorno usada para prever endereços de retorno de sub-rotinas.
- Pode obter e enfileirar até 12 instruções.
- Emite duas instruções de cada vez.

Unidade de decodificação de instruções

- Decodifica e sequencia todas as instruções.
- Estrutura de pipeline dual pipeline, *pipe0* e *pipe1.*
  - Duas instruções podem prosseguir ao mesmo tempo.
  - Pipe0 contém instrução mais antiga na ordem do programa.
  - Se instrução no pipe0 não puder ser emitida, pipe1 não emitirá.
- Instruções prosseguem em ordem.
- Resultados escritos no banco de registradores ao final do pipeline de execução.
  - Previne hazards WAR.
  - Mantém diretos o acompanhamento de hazards WAW e a recuperação de condições de esvaziamento.
  - Preocupação principal do pipeline de decodificação é prevenção de hazards RAW.

Estágios de processamento de instrução

- D0− Instrução Thumb descompactads e decodificação preliminar efetuada.
- D1− Decodificação de instrução é concluída.
- D2− Instrução de escrita de e para instruções da fila pendente/*replay.*
- D3− Contém a lógica de escalonamento de instrução.
  - Scoreboard prevê disponibilidade de registrador usando escalonamento estático.
  - Verificação de hazard.
- D4− Decodificação final para sinais de controle de unidades de execução de inteiros e leitura/escrita.

Unidade de execução de inteiros

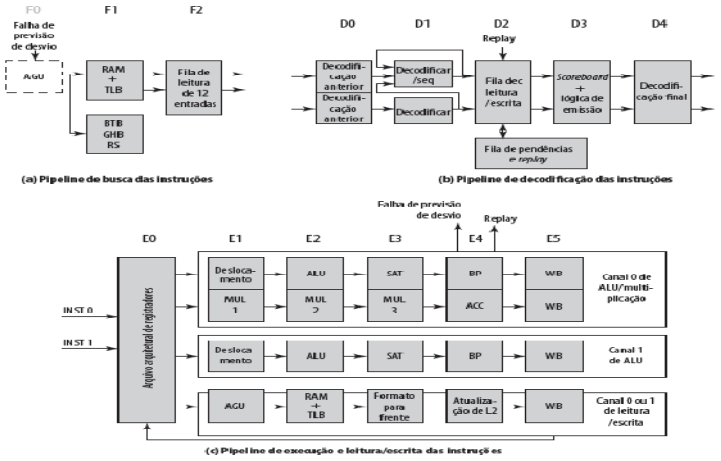
- Dois pipelines simétricos (ALU), um gerador de endereço para instruções de leitura e escrita, e pipeline de multiplicação.
- Estágios de pipeline:
- E0− Acessar banco de registradores.
  - Até seis registradores para duas instruções.
- E1− Deslocador rápido, se necessário.
- E2− Função da ALU.
- E3 −Se necessário, completa saturação aritmética.
- E4− Alteração no fluxo de controle priorizada e processada.
- E5− Resultados escritos no banco de registradores.

- Instruções que invocam unidade de multiplicação levadas para pipe0.
  - Realizadas nos estágios de E1 até E3.
  - Operação de acumulação da multiplicação feita em E4.

Pipeline de leitura/escrita

- Paralelo ao pipeline de inteiros.
- E1− Endereço de memória gerado a partir do registrador de base e índice.
- E2− Endereço aplicado a arrays de cache.
- E3− leitura, dados retornados e formatados.
- E4− escrita, dados formatados e prontos para serem escritos na cache.
- E5− Atualiza cache L2, se necessário.
- E6− Resultados são escritos de volta no banco de registradores.

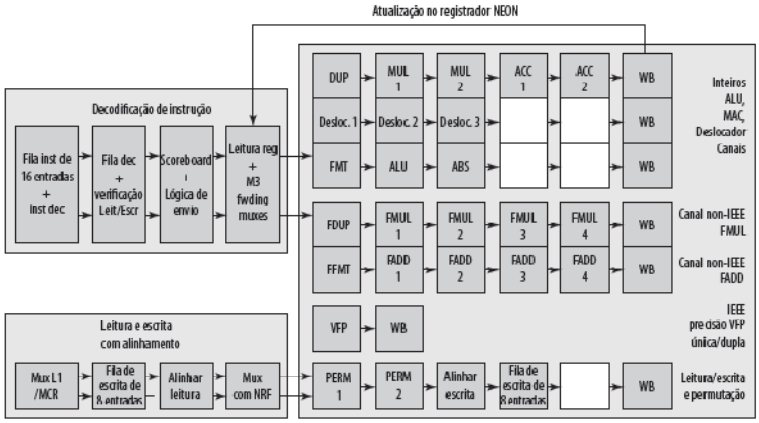
Pipeline de inteiros do ARM Cortex-A8



Pipeline SIMD e de ponto flutuante

- Instruções SIMD e de ponto flutuante passam pelo pipeline de inteiros.
- Processadas em pipeline de 10 estágios separados.
  - Unidade NEON.
  - Trata de instruções SIMD empacotadas.
  - Oferece dois tipos de suporte para ponto flutuante.
- Se implementado, coprocessador vetorial de ponto flutuante (VFP) efetua operações de ponto flutuante no padrão IEEE 754.
  - Se não, pipelines de multiplicação e adição separados implementam operações de ponto flutuante.

ARM Cortex-A8 NEON & Pipeline de ponto flutuante



Leitura recomendada

- Stallings, Capítulo 14.
- *Sites Web* dos fabricantes.
- *Site Web* IMPACT:
  - Procure por “predicated execution”.