Estudo e aplicação de reúso em uma arquitetura VLIW Multicore

Pedro Henrique Exenberger Becker
Universidade Federal do Rio Grande do Sul
Instituto de Informática
Orientado por Prof. Dr. Antônio Carlos S. Beck Filho
phebecker@inf.ufrgs.br

Motivação

- Demanda crescente dos consumidores por **maior performance** em seus dispositivos.
- Processadores **VLIW** exploram paralelismo no momento de compilação
  - São necessários mecanismos diferentes para explorar ganhos de desempenho
  - Área não é crítica: **Baixa complexidade** dos processadores VLIW não restringe adição de hardware, quando se compara com processadores *superescalares*.
- **Redundância** inerente aos códigos de aplicações
  - Porções de código podem executar repetidas vezes ao longo de uma aplicação.
  - *Modularidade* altamente utilizada por desenvolvedores.
- **Múltiplas threads** podem executar trechos semelhantes
  - Redundância entre processos pode existir quando múltiplas instâncias de aplicações executam ou quando diferentes processos usam as mesmas bibliotecas.
- **Consumo** de energia pode reduzido quando aplicações executam em menos tempo.

É possível se valer da redundância de código para extrair maior desempenho de processadores, inserindo o menor despendimento de área e consumo energético possível?

Desafios

- **Baixo consumo** de energia requerido para diferentes nichos de mercado
  - *Sistemas Embarcados*: Uso eficiente da bateria.
  - *Datacenters*: Temperatura para correta operação e manutenção do sistema.
- **Compatibilidade binária**
- Perceber a **granularidade** mais adequada para o mecanismo de reúso
  - Estudos semelhantes, em nível de simulação, indicam que *grãos maiores* potencializam *ganhos maiores*.

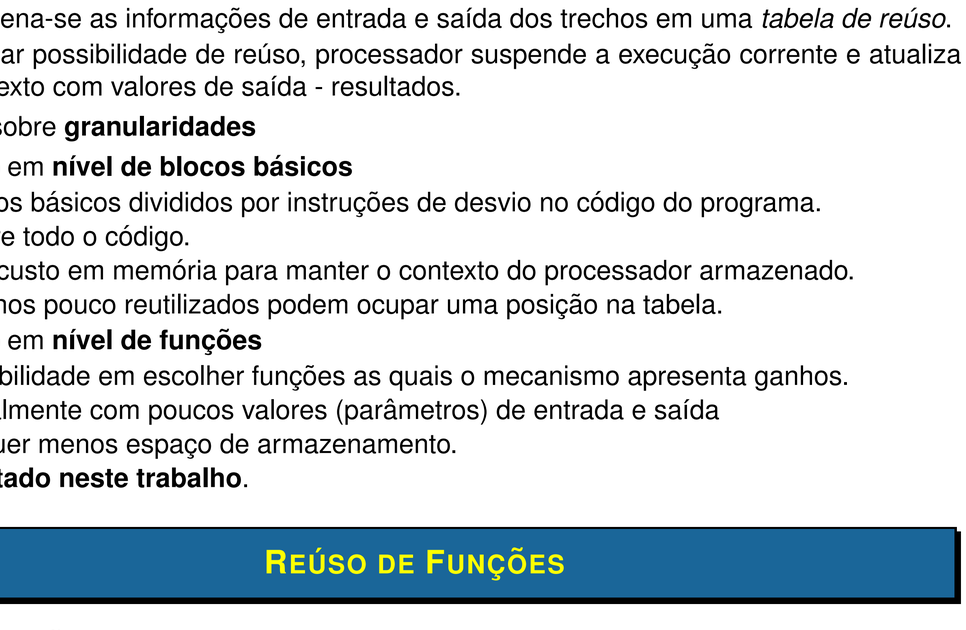
Necessidade de buscar-se novos paradigmas *microarquiteturais* que maximizem a performance provida por unidade de energia por unidade de área.

Reúso em Hardware

- Objetiva **evitar reexecução** de trechos repetidos
  - Verifica-se os valores de entrada de trechos de código.
  - Armazena-se as informações de entrada e saída dos trechos em uma *tabela de reúso*.
  - Ao notar possibilidade de reúso, processador suspende a execução corrente e atualiza o contexto com valores de saída - resultados.
- Estudo sobre **granularidades**
  - Reúso em **nível de blocos básicos**
    - Blocos básicos divididos por instruções de desvio no código do programa.
    - Cobre todo o código.
    - Alto custo em memória para manter o contexto do processador armazenado.
    - Trechos pouco reutilizados podem ocupar uma posição na tabela.
  - Reúso em **nível de funções**
    - Flexibilidade em escolher funções as quais o mecanismo apresenta ganhos.
    - Geralmente com poucos valores (parâmetros) de entrada e saída
    - Requer menos espaço de armazenamento.
    - **Adotado neste trabalho**.

Reúso de Funções

- Reúso de **grão grosso**
- **Vantagem** em relação ao reúso de blocos básicos
  - Menos memória.
  - Menos hardware - menos comparadores para verificação de reúso.
  - Funções são fortemente reutilizadas em aplicações.
  - Fatores dão a intuição de **menor complexidade** do sistema

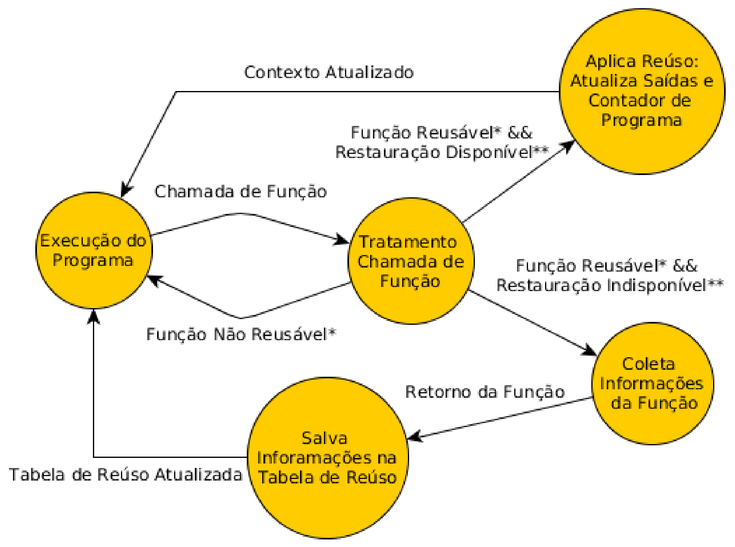


Execução de função evitada por consulta do resultado na tabela de reúso.

- **Funções potenciais** para o reúso podem ser escolhidas pelo processador
  - Alta semelhança em reexecuções
    - Maiores chances de *hits* na tabela de reúso.
  - Poucos parâmetros de entrada e saída
    - Maior facilidade para verificar igualdade de contexto
  - Funções lentas
    - Mais ciclos de execução evitados por aplicar reúso
- Exemplos de *funções potenciais*
  - Funções matemáticas como *pow*, *exp* ou *sen*.
- Caso de estudo **Processador VLIW ρvex**
- Convenção de chamadas de função no VEX
  - Até 8 parâmetros via registradores.
  - Parâmetros excedentes empilhados em memória.

Sistema Proposto

- **Desenvolver** um módulo de reúso através de *descrição de hardware*, incorporando esse módulo à descrição do processador ρvex.



Máquina de estados de monitoramento e aplicação de reúso.

* *Funções reusáveis* são funções as quais o mecanismo se aplica. São escolhidas conforme o potencial de reúso que apresentam.
** *Funções disponíveis* são funções cujos retornos estão disponíveis na tabela de reúso, dado que já foram calculados em chamadas anteriores.

- **Compartilhar** unidade de reúso entre cores do processador. Explorar reúso de funções **entre processos**.
- **Metodologia**: **Análise de potencial**.
  - Funções com bom potencial de reúso são definidas estaticamente em benchmarks escolhidos.
  - **Métrica**: Tempo de execução das aplicações ao adicionar a unidade de reúso.
  - **Variam-se as características** do processador, como issue slots, e da unidade de reúso, como tamanho da tabela de reúso ou associatividade.
  - **Speedup**: resultados normalizados com respeito ao processador original. Valores maiores que um representam ganhos do sistema proposto.
- Tempo para verificação e aplicação de reúso = **4 ciclos**.
- **Potencial** do método
  - Função com laço de repetição para calcular 3^2.
    - **Primeira execução** a função leva **23 ciclos** para retornar o valor final.
    - **Segunda execução**, reúso é aplicado e a função leva **apenas 4 ciclos** para retornar o valor final.
  - Quando o reúso consegue ser aplicado, a *complexidade* da função é constante.

Percepção inicial é de que o reúso de funções tem potencial em aumentar o desempenho de aplicações.

Sobre o Trabalho

- **Conclusões**
  - A ideia **se provou factível** no processador proposto.
  - Apresenta **potencial para continuidade** da pesquisa.
- **Trabalhos futuros**
  - **Análise** detalhada **da performance**, resultante da execução de benchmarks variados no sistema proposto.
  - Definir **dinamicamente** funções reusáveis.
  - Expandir mecanismo para versão **multicore** do processador.
  - Análise do **impacto** da adição da unidade de reúso.
    - Extração de dados de **área**.
    - Extração de dados de **energia**.
  - Otimizações do mecanismo.

SIC UFRGS 2016. XXVIII Salão de Iniciação Científica - UFRGS, 12 a 16 de setembro de 2016. Campus do Vale, Porto Alegre, Brasil.