Sensibilidade a erros em aplicações na arquitetura RISC-V

João Fabrício Filho[1,2], Isaías B. Felzmann[2], Lucas F. Wanner[2]

[1]Universidade Tecnológica Federal do Paraná – Câmpus Campo Mourão
Via Rosalina M Santos, 1233 – 87301-899 – Campo Mourão – PR – Brasil

[2]Instituto de Computação – Universidade Estadual de Campinas
Av Albert Einstein, 1251 – 13083-852 – Campinas – SP – Brasil

joaof@utfpr.edu.br, {isaias.felzmann,lucas}@ic.unicamp.br

Resumo. *Arquiteturas que implementam o conjunto de instruções RISC-V são adequadas para o contexto de sistemas embarcados. A demanda por menor consumo energético e maior desempenho nesse contexto é crescente, e a aproximação de elementos de memória tem potencial para alcançar ambos os benefícios. Contudo, a sensibilidade a erros de cada aplicação pode impedir a obtenção de maiores benefícios, por meio de quebras de execução ou menor qualidade dos resultados. Neste trabalho, propomos a avaliação da sensibilidade de aplicações a falhas em dados armazenados em memória na arquitetura RISC-V. Expondo toda a memória de dados a um modelo de erro em um simulador, é possível verificar a correlação entre o aumento das quebras de execução e a diminuição da qualidade dos resultados. Para um requisito de qualidade de 90%, as 3 aplicações avaliadas toleraram diferentes níveis de aproximação em escala logarítmica, chegando na ordem da taxa de erro de* 10^{-7}.

1. Introdução

É crescente a demanda por desempenho e economia de energia em sistemas computacionais embarcados. Nesses sistemas, elementos de memória podem representar até 60% do consumo de energia [Konstantakos et al. 2008], nos quais o acesso também representa um gargalo para se obter desempenho. É possível diminuir o consumo de energia e aumentar o desempenho por meio de alterações nos parâmetros de funcionamento da memória, ao custo de eventuais erros nos dados da aplicação [Koppula et al. 2019]. Contudo, cada aplicação pode ter sensibilidade diferente a esses erros, traduzida em maior ou menor qualidade dos resultados finais [Raha et al. 2017].

As pesquisas em sistemas embarcados modernos demandam por escalabilidade e extensibilidade, requisitos atendidos pelas arquiteturas RISC-V, que implementam um conjunto de instruções de código aberto [RISC-V Foundation 2017]. A implementação de um modelo de aproximação em uma arquitetura depende de diversos fatores, como o tipo de memória e características do ambiente. Modelos de erro da literatura implementados em um ambiente simulado podem ter características gerais de aproximação em memória que permitem maior aplicabilidade da técnica.

O objetivo deste trabalho é avaliar a suscetibilidade a falhas de aplicações em um ambiente de memória aproximada em arquiteturas que implementem o conjunto

Este trabalho tem apoio do CNPq, da CAPES e da FAPESP (#2018/24177-0).

de instruções RISC-V. Em nosso ambiente, toda a memória de dados da aplicação é exposta a erros para evitar alterações no código em relação à execução em arquiteturas não-aproximadas. Além disso, há um controle de nível de aproximação que define a taxa de erros nos dados da aplicação.

Nossa implementação utiliza o simulador RISC-V de referência Spike para substituir as operações de memória por um modelo de erro e avaliar 3 aplicações nesse ambiente. Nossos resultados evidenciam diferentes tolerância a erros de cada aplicação e a forte influência de erros em dados críticos na qualidade das saídas. Quanto maior a taxa de erro tolerada, maiores os benefícios em desempenho ou energia proporcionados pela aproximação. Para um requisito de qualidade de 90% as aplicações avaliadas toleram taxas de erro de até 10^{-7}.

2. Memórias Aproximadas

Estruturas de memória são altamente sensíveis a variabilidade do circuito, devido à utilização tipicamente dos *layouts* mais densos entre os componentes de *hardware* [Gottscho et al. 2015]. Além disso, elementos de memória representam uma parte significativa do total de energia de um sistema embarcado, podendo chegar a 60% [Konstantakos et al. 2008]. A aproximação dos elementos de memória tem potencial para obter ganhos em desempenho e energia [Koppula et al. 2019]. Essa aproximação pode ser realizada por alterações no circuito da memória ou nos parâmetros de funcionamento.

Parâmetros de energia, como a tensão de alimentação, e parâmetros de latência, como a taxa de atualização ou *delay* de ativação, podem ser ajustados para obter benefícios ao custo de eventuais erros nos dados armazenados [Koppula et al. 2019, Raha et al. 2017]. Aumentar o nível de aproximação expõe o sistema a maiores taxas de erro, mas também traz maiores benefícios em energia e desempenho, a depender da técnica de aproximação utilizada [Fabrício Filho et al. 2019, Gottscho et al. 2017].

3. Sensibilidade das Aplicações a Erros

A aproximação de elementos de memória utilizada permite ganhos em desempenho ou energia. Contudo, a sensibilidade de cada aplicação vai traduzir o impacto dos erros na qualidade das saídas de cada execução. Além disso, há dados críticos da aplicação que não admitem erros. Erros em dados críticos podem ocasionar interrompimento prematuro da execução ou ainda impedir que a execução termine [Felzmann et al. 2018]. Isso resulta em saídas não produzidas, reduzindo a qualidade média dos resultados obtidos.

Nossa proposta é avaliar a sensibilidade a aproximação de memória de diferentes tipos de aplicação em um ambiente que implementa o conjunto de instruções RISC-V. As aplicações devem possuir algum grau de tolerância a erros para permitir a aproximação. O ambiente em questão deve expor toda a memória de dados a erros, independentemente da fase de execução. Dessa forma, controles de aproximação são simplificados para permitir a execução de aplicações sem modificações no código. Além disso, as mudanças arquiteturais são mínimas para permitir esse tipo de aproximação, já que basta o controle do nível de aproximação em memória antes da execução da aplicação, e.g. controle de tensão, taxa de atualização e parâmetros de latência. Quanto maior o nível de aproximação e a taxa de erro ao qual a aplicação está exposta, maiores são os benefícios obtidos em energia ou desempenho.

Um ambiente simulado permite diversas alternativas para implementação dos erros em uma arquitetura. Os erros em um ambiente de memória aproximada dependem de diversos fatores, como componentes, temperatura, arquitetura e tipo da memória. O trabalho de [Koppula et al. 2019] comparou diferentes tipos de erro e evidenciou a probabilidade uniforme de inversão de *bit* como um modelo de erro robusto para aproximações por meio de ajuste de tensão e de parâmetros de latência.

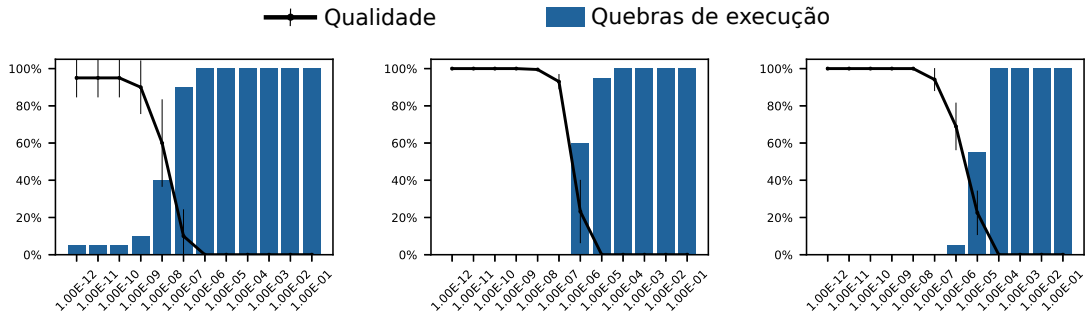
4. Avaliação Experimental

4.1. Configuração

Nosso ambiente experimental substitui as operações de memória por modelos de *software* que expõem a palavra de dados em uma probabilidade uniforme a uma inversão de *bit*. Esse modelo foi implementado em um simulador Spike [Waterman and Lee] em taxas de erro de 10^{-12} a 10^{-1}, em 12 intervalos logarítmicos. Os intervalos logarítmicos podem representar faixas de erros em alterações de parâmetros de memória, como a escala de tensão [Fabrício Filho et al. 2019, Felzmann et al. 2018] ou tempos de latência [Koppula et al. 2019]. Como o erro é não-determinístico, diferentes execuções podem apresentar resultados diferentes em uma mesma taxa de erro. Dessa forma, para obter dados relevantes executamos cada aplicação 20 vezes em cada taxa de erro. Analisamos os resultados por meio de duas métricas para medir a sensibilidade das aplicações a erros: (1) quebras de execução e (2) qualidade das saídas. Quebras de execução são interrompimentos da execução antes da produção da saída. A qualidade das saídas mede o impacto dos erros nos resultados finais de cada execução.

A métrica de qualidade é a diferença entre a saída aproximada e a saída acurada, e depende do contexto de cada aplicação. Para melhor visualização de dados, escolhemos métricas normalizadas. As aplicações executadas e suas respectivas métricas de qualidade são: (1) JPEG: similaridade estrutural entre duas imagens; (2) QSORT: a fração de elementos iguais em cada posição de um vetor; e (3) NBODY: erro relativo médio entre todos os elementos da saída.

Alguns erros podem impedir que a aplicação convirja, fazendo com que a execução dure infinitamente. Para evitar esse problema, atribuímos como tempo máximo de execução o dobro de uma execução precisa. Assim, quando esse tempo é excedido, a execução é finalizada mesmo que não tenha produzido uma saída.



(a) JPEG (b) QSORT (c) NBODY

Figura 1. Resultados de qualidade e quebras de execução para cada aplicação.

4.2. Resultados

A figura 1 apresenta os resultados obtidos em relação ao número de quebras de execução e à qualidade com intervalo de confiança de 95%. Para as três aplicações, observa-se uma forte correlação entre o aumento das quebras de execução e a diminuição da qualidade dos resultados. Nos pontos de inflexão, nos quais há quebras e também saídas, a variabilidade da qualidade aumenta devido às qualidades nulas em execuções interrompidas. A aplicação JPEG manipula imagens, tipos de dados que comumente são tolerantes a falhas. Contudo, essa aplicação se mostrou mais sensível a erros, permitindo apenas uma taxa de erro de até 10^{-10} para um requisito de qualidade de 90%, enquanto NBODY e QSORT permitem até 10^{-7}. Isso se deve, principalmente, à sua maior utilização de memória e ao formato dos dados de entrada, nos quais um erro poderia causar uma quebra de execução.

5. Considerações Finais

Memórias aproximadas podem reduzir o consumo de energia e aumentar o desempenho de sistemas computacionais. Todavia, mostramos que cada aplicação pode reagir de maneira diferente a esse tipo de aproximação, e essa reação depende não só dos tipos de dados mas também das estruturas utilizadas. Nossos resultados evidenciaram a correlação entre o aumento de quebras de execução e a diminuição da qualidade dos resultados em maiores taxas de erro nos dados da aplicação. No geral, as aplicações avaliadas toleram taxas de erro de até 10^{-7} para um requisito de qualidade de 90%. Nossos trabalhos futuros incluem a investigação de potenciais ganhos de desempenho e energia para cada taxa de erro, a avaliação experimental do modelo arquitetural proposto e a identificação automatizada de regiões aproximáveis de aplicações.

Referências

Fabrício Filho, J., Felzmann, I., Azevedo, R., and Wanner, L. (2019). A resilient interface for approximate data access. In *SBESC*.

Felzmann, I. B., Fabrício Filho, J., Azevedo, R. J., and Wanner, L. F. (2018). Impacto de memórias aproximadas na eficiência energética. In *WSCAD*.

Gottscho, M., BanaiyanMofrad, A., Dutt, N., Nicolau, A., and Gupta, P. (2015). DPCS: Dynamic Power/Capacity Scaling for SRAM Caches in the Nanoscale Era. *TACO*.

Gottscho, M., Shoaib, M., Govindan, S., Sharma, B., Wang, D., and Gupta, P. (2017). Measuring the Impact of Memory Errors on Application Performance. *IEEE CAL*.

Konstantakos, V., Chatzigeorgiou, A., Nikolaidis, S., and Laopoulos, T. (2008). Energy consumption estimation in embedded systems. *IEEE TIM*.

Koppula, S., Orosa, L., Yaglikci, A. G., Azizi, R., Shahroodi, T., Kanellopoulos, K., and Mutlu, O. (2019). EDEN: Enabling energy-efficient, high-performance deep neural network inference using approximate DRAM. In *MICRO*.

Raha, A., Sutar, S., Jayakumar, H., and Raghunathan, V. (2017). Quality Configurable Approximate DRAM. *IEEE TC*.

RISC-V Foundation (2017). The RISC-V Instruction Set Manual. `https://content.riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf`.

Waterman, A. and Lee, Y. Spike, a RISC-V ISA Simulator. `https://github.com/riscv/riscv-isa-sim`.