Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Toledo
Engenharia de Computação

Relatório Técnico da prática utilizando o Raspberry Pi – Organização e Arquitetura de Computadores

Acadêmicos: **RA:**
Gabriel Duarte Santos (2382954)
Gustavo d'Avila (2370506)
William Nathan Pereira (1910450)

Data: 25 de Abril de 2024

Sumário

1. Introdução 1
2. Materiais e Métodos 1
3. Resultados e Discussão 2
Raspberry Pi 3 Model B: 2
Ryzen 5 5600: 2
Gráficos de Desempenho: 3
4. Considerações Finais: 4
5. Referências Bibliográficas 4

1. Introdução

A computação paralela é um campo crucial no desenvolvimento de sistemas de alto desempenho e eficiência energética. Uma das técnicas mais comuns para explorar a paralelização de tarefas é o uso de multithreading, que permite a execução simultânea de múltiplas tarefas em um único processador. Este estudo visa investigar o ganho de desempenho ao utilizar multithreading em comparação com um único thread em sistemas baseados no Raspberry Pi 3 Model B e em computadores convencionais.

2. Materiais e Métodos

Para este estudo, utilizamos um Raspberry Pi 3 Model B com o sistema operacional Raspbian versão completa, além de um computador convencional com Windows. O Raspberry Pi possui uma arquitetura ARM A72, enquanto o computador possui uma arquitetura Zen3.

Para o desenvolvimento do código, empregamos o Visual Studio Code como ambiente de programação e o compilador GCC para ambas as plataformas. Para implementar a técnica de multithreading, utilizamos a biblioteca OpenMP.

As configurações dos dispositivos são as seguintes: o Raspberry Pi possui um processador Quad Core 1.2GHz Broadcom BCM2837 64bit e 1GB de RAM, enquanto o computador possui um processador Ryzen 5 5600 4.65GHz Zen3 e 32GB de RAM.

O procedimento experimental consistiu em preparar o ambiente de desenvolvimento

em ambos os sistemas, desenvolver o código-fonte utilizando a biblioteca OpenMP, compilar o código com o GCC e executar testes de desempenho comparando a execução com múltiplos threads e um único thread. Registramos os resultados obtidos, incluindo tempos de execução e uso de CPU, e realizamos uma análise para avaliar o ganho de desempenho obtido com a utilização de multithreading em cada sistema.

3. Resultados e Discussão

Os resultados obtidos dos testes de desempenho utilizando multithreading em diferentes configurações de hardware e software são apresentados e discutidos abaixo.

Raspberry Pi 3 Model B:

Para o Raspberry Pi, os testes foram conduzidos utilizando dois programas diferentes e o aplicativo x264 para codificação de vídeo.

No primeiro programa (**finalOpenMP.c**), que consiste em operações em um vetor de tamanho 1000, observamos um aumento significativo de desempenho com o aumento do número de threads. Por exemplo, a execução com 4 threads foi cerca de 4.6 vezes mais rápida do que a execução com 1 thread. Isso demonstra a eficácia do multithreading em melhorar o desempenho em sistemas com recursos limitados, como o Raspberry Pi.

No segundo programa (**exampleOpenMP.c**), que envolve operações em um vetor de tamanho 80000, novamente observamos uma melhoria no desempenho com o aumento de threads, embora em menor escala. A execução com 4 threads foi cerca de 2.5 vezes mais rápida do que a execução com 1 thread.

No teste com o aplicativo x264, utilizado para codificação de vídeo, também observamos um aumento linear na taxa de frames por segundo (fps) com o aumento do número de threads. A execução com 4 threads foi aproximadamente 2.5 vezes mais rápida do que a execução com 1 thread.

Ryzen 5 5600:

Para o Ryzen 5 5600, os testes foram conduzidos com o mesmo conjunto de programas, porém em um hardware mais potente.

No primeiro programa (**finalOpenMP.c**), operando em um vetor de tamanho 1000, notamos uma melhoria ainda mais significativa no desempenho com o aumento de threads. Por exemplo, a execução com 4 threads foi aproximadamente 8 vezes mais rápida do que a execução com 1 thread. Isso demonstra como o processador mais poderoso consegue lidar de forma mais eficiente com tarefas paralelas.

No segundo programa (**exampleOpenMP.c**), operando em um vetor de tamanho 50000, também observamos uma melhoria considerável no desempenho com o aumento de threads. A execução com 4 threads foi aproximadamente 3.5 vezes mais rápida do que a execução com 1 thread.

Gráficos de Desempenho:

Gráfico representando o aumento da performance utilizando do código final (**finalOpenMP.c**).

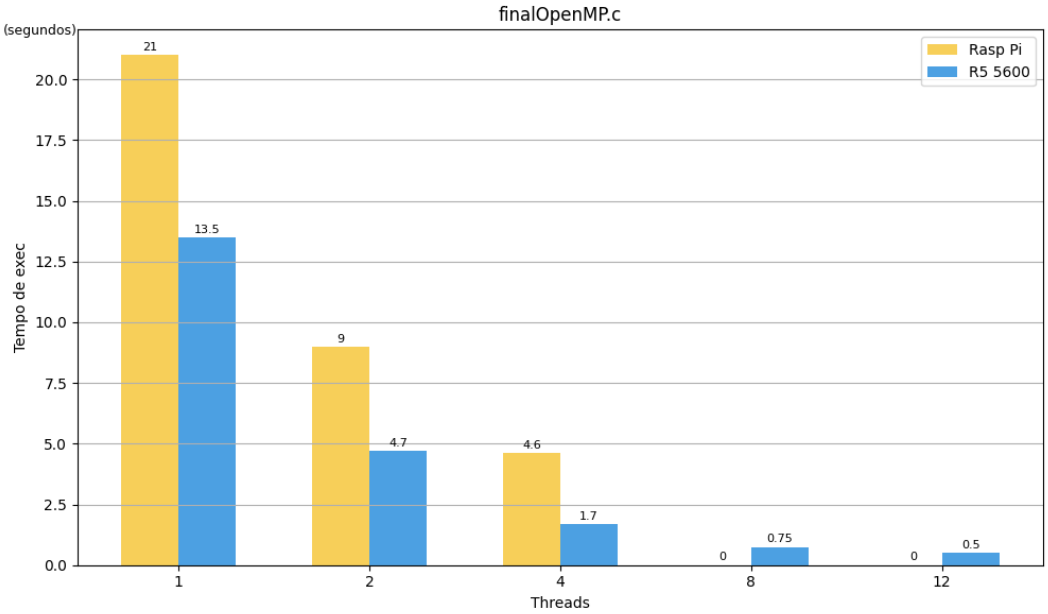


Gráfico representando o aumento da performance utilizando de uma versão modificada do exemplo apresentado em aula (**exampleOpenMP.c**).

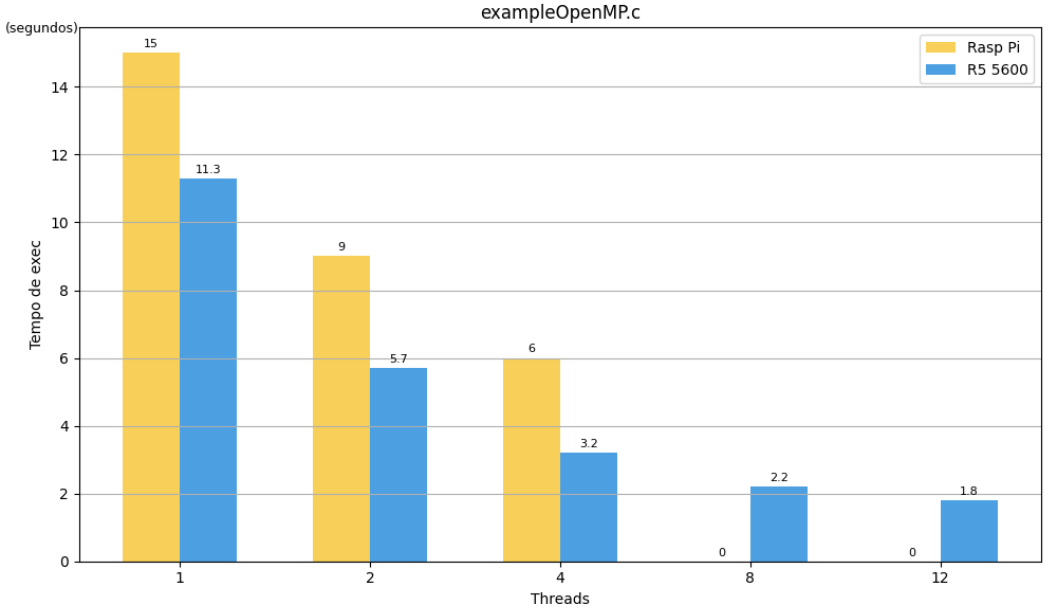
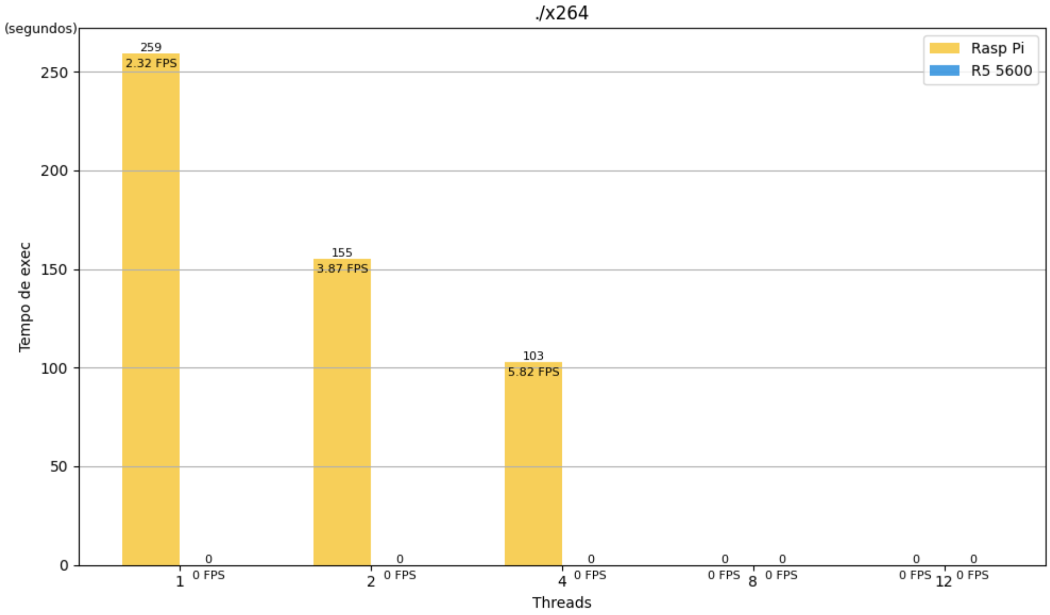


Gráfico representando o aumento da performance utilizando a ferramenta de Benchmark do **x264**.



4. Considerações Finais:

Os resultados obtidos corroboram a eficácia do multithreading em melhorar o desempenho em sistemas computacionais, especialmente em ambientes com recursos limitados. A escalabilidade do desempenho com o aumento de threads foi observada em ambas as plataformas, destacando a importância do uso eficiente de recursos em aplicações modernas. Além disso, os resultados também evidenciam a influência do hardware na eficácia do multithreading, com processadores mais potentes apresentando ganhos de desempenho mais expressivos.

5. Referências Bibliográficas

Arquitetura e organização de computadores /. William Stallings. — 8. ed. — São Paulo: Pearson Pratice Hall, 2010. Disponível em: < https://www.telecom.uff.br/orgarqcomp/arq/arquitetura-e-organizacao-computadores-8a.pdf >.

Mattson, Tim. "A 'Hands-on' Introduction to OpenMP." 2019. Disponível em: < https://www.openmp.org/wp-content/uploads/Intro_To_OpenMP_Mattson.pdf >.