Relatório Prática 1 – Raspberry Pi 3

Reinaldo Kaminski Neto e Matheus Berbel Barusso

I. Introdução

O contínuo desenvolvimento de sistemas embarcados tem impulsionado a adoção de equipamentos acessíveis e de alto desempenho, como o *Raspberry Pi*. Nesse contexto, surge a necessidade de analisar a capacidade dessas plataformas em executar processos demandantes, principalmente em cenários que exploram o paralelismo de núcleos de processador, característica central de aplicações *multithread*.

Este trabalho tem como objetivo analisar o uso de múltiplas threads (*multithreading*) no contexto do *Raspberry Pi 3*, comparando seu desempenho com o de um notebook convencional. Para isso, foram realizados diversos testes de benchmarking utilizando ferramentas já estabelecidas, como o *John the Ripper*, além do desenvolvimento e execução de aplicações com paralelismo por meio da biblioteca OpenMP.

A proposta visa avaliar as capacidades do *Raspberry Pi 3* em explorar o paralelismo de hardware disponível, bem como identificar possíveis limitações em comparação à um sistema de maior desempenho, como o notebook convencional.

.

II. Materiais e Métodos

Antes da realização dos testes de desempenho, é essencial compreender as características de hardware dos dois dispositivos comparados: o Raspberry Pi 3 e o notebook DCM4A-4.

Para isso, utilizou-se o comando `lshw` [1] em ambos os sistemas, o qual fornece informações detalhadas sobre os componentes físicos, como CPU, memória e arquitetura. Os parâmetros mais relevantes para a análise estão resumidas abaixo:

TABLE I
Características de hardware do Raspberry Pi 3

Especificação	Raspberry Pi 3B
Arquitetura	ARMv8 64 bits
Processador	BCM2837 (4 núcleos, 1.4 GHz)
Memória RAM	1 GB LPDDR2
Armazenamento	Cartão microSD
Sistema Operacional	Raspberry Pi OS (Debian 12)
Capacidade Multithread	4 threads
Energia consumida (estimada)	5–7 W

TABLE II
Características de hardware do Notebook DCM4A-4

Especificação	Notebook DCM4A-4
Arquitetura	x86_64 64 bits
Processador	AMD Ryzen 5 (4 núcleos, 2.5 GHz)
Memória RAM	8 GB DDR4
Armazenamento	SSD/HDD
Sistema Operacional	Ubuntu 22.04
Capacidade Multithread	4+ threads com Hyper-Threading
Energia consumida (estimada)	60 W

A. Testes de Desempenho

Para avaliar o desempenho dos dois dispositivos, foram realizados dois experimentos distintos: o uso da ferramenta *John the Ripper* [2] e a multiplicação de matrizes de dimensão 1000×1000 implementada com OpenMP [3]. Ambos os testes foram executados em modos *single-thread* e *multi-thread*, permitindo uma análise comparativa da capacidade de paralelização e da eficiência de cada plataforma.

A ferramenta *John the Ripper* pode ser instalada diretamente pelo gerenciador de pacotes com o comando `sudo apt install john`. Para configurar a execução com múltiplas threads, utiliza-se o parâmetro `--fork=N`, conforme descrito na documentação oficial [2]. Neste trabalho, optou-se por utilizar 3 threads, respeitando o limite de 4 núcleos físicos disponíveis no Raspberry Pi 3, de modo a evitar problemas que serão detalhados durante este relatório.

Utilizando o algoritmo `md5crypt` como modelo de base, podemos avaliar o desempenho dos dispositivos:

Durante os testes com a ferramenta *John the Ripper* no notebook, foram apresentados diferentes modos de benchmarking, como *Many salts* e *Only one salt*, devido a otimizações específicas da arquitetura x86_64 com suporte a instruções AVX2. Para efeito de comparação com o Raspberry Pi 3, optou-se por utilizar o resultado da opção *Only one salt*, pois esse modo representa um cenário comum de quebra de senha com um único valor de *salt* e é mais compatível com os testes realizados no Raspberry Pi, que não conta com otimizações como AVX2. Dessa forma, garante-se uma comparação mais real entre os dois dispositivos.

B. Testes de Desempenho com OpenMP

Para avaliar o desempenho dos dispositivos com e sem o OpenMP [4], foi desenvolvido um programa em `C++` que realiza as seguintes operações:

- Gera duas matrizes 1000x1000 com números aleatórios
- Multiplica essas duas matrizes
- Calcula a inversa da matriz resultante
- Mede somatória de tempo gasto em cada uma dessas operações utilizando a biblioteca Chrono [5].

O uso do OpenMP permite distribuir o trabalho entre os núcleos de processamento disponíveis em cada dispositivo, possibilitando a execução concorrente das tarefas que são facilmente paralelizáveis, como a multiplicação de matrizes.

O código fonte, assim como as instruções de compilação, podem ser encontradas no GitHub [3].

III. Resultados e Discussão

Os experimentos foram realizados em ambos os hardwares descritos pelas Tabelas I e II utilizando os testes de desempenho apresentados na seção II, Materiais e Métodos.

Inicialmente era questionado se os resultados apresentariam ganhos relevantes de performance e tempo de execução, porém a melhora obtida pode ser considera significativa no cenário de testes de Benchmarking.

Para o teste de desempenho realizado utilizando a ferramenta *John the Ripper* [2] foi notável o ganho de 3,8x de performance para o teste utilizando o Raspberry Pi 3, enquanto o ganho de performance para o Notebook DCM4-A, por mais que existente, foi proporcionalmente menor, cerca de 1,78x.

TABLE III
DESEMPENHO DO RASPBERRY PI 3B

Modo de Execução	**md5crypt (c/s real)**
Single-thread	3.504 c/s
Multi-thread (3 threads)	13.356 c/s
Ganho com multithreading	3,8×

TABLE IV
DESEMPENHO DO NOTEBOOK DCM4-A

Modo de Execução	**md5crypt (c/s real)**
Single-thread	112.728 c/s
Multi-thread (3 threads)	201.361 c/s
Ganho com multithreading	1,78×

Além dos testes utilizando o Benchmark *John the Ripper* [2], os testes foram realizados também para a checagem de aumento de desempenho e diminuição de tempo de execução de um algoritmo desenvolvido pela equipe com o objetivo de demonstrar os possíveis ganhos quando paralelismo de tarefas é aplicada, tornando um algoritmo complexo que antes utilizava apenas um Thread em um novo algoritmo que tem o potencial de beneficiar-se da biblioteca OpenMP [4], utilizando mais Threads para a execução do código.

O teste utilizando a biblioteca OpenMP também obteve resultados favoráveis a paralelização de processos, com um ganho de tempo de execução de 3.8x para o Notebook DCM4-A. Para o teste realizado no Raspberry Pi o tempo de execução do teste decaiu de 29.704 segundos para 21.298, apresentando um ganho de 1,39x ao optarmos por utilizar o multi Threading para a operação.

Vale lembrar que apenas 3 dos 4 threads do Raspberry Pi foram utilizados para o benchmarking do OpenMP, visto que ao executarmos o programa compilado definindo 4 threads para o uso a placa Raspberry Pi desligava subitamente e reiniciava após isso. Inúmeras tentativas foram feitas mas o resultado foi o mesmo, ao limitarmos o uso para 3 threads o problema não se repetiu, e portanto, foi definido para os testes em ambos os Hardwares o limite de 3 threads para a paralelização dos cálculos.

TABLE V
DESEMPENHO DO RASPBERRY PI 3 B

Modo de Execução	**tempo (s)**
Single-thread	29.704
Multi-thread (3 threads)	21.298
Ganho com multithreading	1,39×

O gráfico a seguir representa a relação de ganhos para cada um dos Benchmarks e Hardwares testados, comprovando o proposto benefício da paralelização de tarefas em processadores que possuem a tecnologia de *Multi Threading*, demonstrando que o ganho proveniente do uso desta tecnologia existe em processadores de ambas arquiteturas [6], como a *RISC(Reduced Instruction Set Computer)*, presente no processador ARMv8 64 bits do Raspberry Pi 3 e a arquitetura *CISC(Complex Instruction Set Computer)*, presente no processador AMD Ryzen 5 do Notebook DCM4-A.

TABLE VI
DESEMPENHO DO NOTEBOOK DCM4-A

Modo de Execução	**tempo (s)**
Single-thread	0.867
Multi-thread (3 threads)	0.228
Ganho com multithreading	3,8×

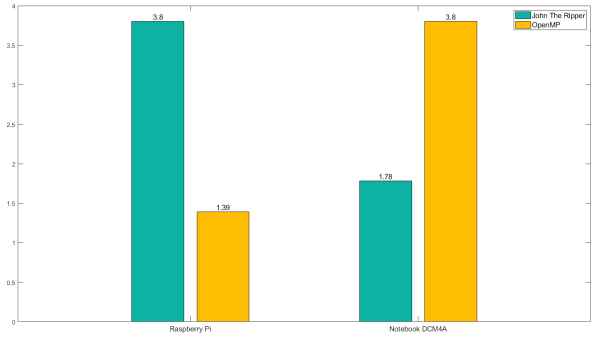

Fig. 1. Relação de Ganhos

REFERENCES

[1] Linux.die.net. (2024) lshw(1) - linux man page. Acesso em: 18 mai. 2025. [Online]. Available: https://linux.die.net/man/1/lshw
[2] Openwall Project. (2024) John the ripper - command line options. Acesso em: 18 mai. 2025. [Online]. Available: https://www.openwall.com/john/doc/OPTIONS.shtml
[3] M. Barusso. (2025) Matriz 1000x1000. Repositório no GitHub. Acesso em: 18 mai. 2025. [Online]. Available: https://github.com/MatheusBarusso/organizacao_aquitetura_de_computadores
[4] (2024) Syntax reference guide. Guia de Referências OpenMP API 6.0. Acesso em: 18 mai. 2025. [Online]. Available: https://www.openmp.org/wp-content/uploads/OpenMP-RefGuide-6.0-OMP60SC24-web.pdf
[5] M. Learn. (2023) cpp standard-library chrono. Documentação Biblioteca Chrono. Acesso em: 18 mai. 2025. [Online]. Available: https://learn.microsoft.com/pt-br/cpp/standard-library/chrono?view=msvc-170
[6] W. Stallings. (2010) Arquitetura e organização de computadores. 8. ed. São Paulo: Prentice-Hall, 2010. 624 p. ISBN 978-85-7605-564-8.