Project Proposal: Comparative Analysis of GPS Grandmaster Using Raspberry Pi 5 and Compute Module 4

Objective

The project's goal is to conduct a comprehensive investigation into the feasibility and efficiency of utilizing Raspberry Pi 5 and Compute Module 4, equipped with u-blox GPS receivers, as cost-effective GPS Grandmasters for precise time synchronization. By applying and comparing various optimization techniques from high-frequency trading (HFT), this study aims to enhance the accuracy and reliability of time synchronization across both platforms, providing critical insights into their comparative performances in time-sensitive applications like financial trading.

Background

In the high-stakes world of high-frequency trading (HFT), nanoseconds can dictate the difference between significant profits and losses. The essence of HFT lies in executing a large number of transactions at extremely high speeds, which necessitates the utmost precision in time synchronization across trading systems. This precision ensures that trade orders are executed in the intended order, maintaining fairness and efficiency in the markets. Current methodologies for achieving this level of precision typically involve sophisticated and costly GPS Grandmaster clocks. These systems utilize GPS signals to provide a time source that networked devices can reference, ensuring their internal clocks are accurately synchronized to a universal standard time.

However, these traditional GPS Grandmaster solutions present several challenges:

- **Cost and Accessibility:** The high cost and complexity of professional-grade GPS Grandmaster clocks limit accessibility for smaller institutions and individual traders, potentially putting them at a disadvantage in the competitive trading environment.
- **Scalability and Flexibility:** Traditional solutions may not offer the scalability and flexibility needed for customized setups or for application in fields beyond finance, such as telecommunications, distributed networks, and scientific research, where precise time synchronization is also crucial.

Recognizing these challenges, our project seeks to explore a cost-effective, scalable alternative by harnessing the capabilities of widely available and affordable hardware, specifically Raspberry Pi 5 or Compute Module 4, equipped with u-blox GPS receivers. This approach aims to democratize access to precise time synchronization technology, making it more accessible to a broader range of users and applications.

Impact and Relevance

Achieving high-precision time synchronization with Raspberry Pi-based systems could significantly lower the barrier to entry for high-frequency trading firms and other industries requiring synchronized time-stamping, such as telecommunications and scientific research. This research aims to extend the application of consumer-grade hardware to professional and critical infrastructures, potentially transforming how these sectors approach the challenge of precise timekeeping.

Approach

The methodology involves a parallel analysis of Raspberry Pi 5 and Compute Module 4 units, focusing on:
- Initial hardware setup with u-blox GPS receivers.
- Implementation of various software and hardware optimization techniques on both platforms.
- Systematic tracking of performance metrics, including time synchronization offset and other relevant measures, before and after optimizations.
- Comparative analysis of performance improvements and optimization effects on Raspberry Pi 5 vs. Compute Module 4.

Comparative Analysis Framework

Hardware and Initial Setup

- Acquisition and configuration of Raspberry Pi 5 and Compute Module 4 units with u-blox GPS receivers.
- Establishment of baseline time synchronization performance metrics for both platforms.

Optimization Techniques Implementation

Sequential application of optimization techniques derived from high-frequency trading practices to both Raspberry Pi 5 and Compute Module 4, including but not limited to:

- **Dedicated CPU Core for PPS Signal Polling**: By using the `isolcpus` kernel parameter, we will isolate a CPU core exclusively for PPS signal polling. This minimizes interruptions from other processes, ensuring precise timing for critical operations.
- **Interrupt Steering for PPS Interrupt Requests**: We will configure interrupt request (IRQ) affinity to direct PPS signal interrupts to the isolated CPU core, reducing latency and enhancing time synchronization accuracy by ensuring prioritized and efficient interrupt handling.
- **Busy Poll Detection**: Implementing Busy Polling in network device drivers to reduce latency in packet processing, especially beneficial in environments where immediate processing of incoming data is critical. This approach ensures faster response times by reducing the wait time for packet processing, making it especially useful in time-sensitive applications like high-frequency trading where every microsecond counts. Careful tuning will be necessary to balance CPU utilization and response time, ensuring the dedicated core does not become a bottleneck due to excessive polling.
- **DMA and SPI Protocols**: Leveraging Direct Memory Access (DMA) and Serial Peripheral Interface (SPI) to improve communication efficiency between the CPU and peripheral devices. These protocols provide speed and efficiency advantages over traditional UART communication, crucial for rapid and reliable data processing.
- **Thermal Regulation**: Implementing heat sinks and potentially thermistors to actively monitor and manage device temperature, preventing overheating and ensuring stable performance within optimal thermal conditions.

Each optimization phase will be followed by a thorough evaluation of its impact on time synchronization accuracy and overall system performance.

Comparative Performance Analysis

To ensure a robust comparative analysis between Raspberry Pi 5 and Compute Module 4 as GPS Grandmasters, we will systematically collect and analyze the following metrics:

1. **Time Synchronization Offset (Accuracy)**: This measures the deviation from a reference time, providing a direct assessment of the system's ability to maintain precise time synchronization.
2. **Jitter**: By quantifying the variability in time synchronization accuracy, jitter gives insights into the system's stability over time.
3. **System Latency**: This measures the delay between GPS signal reception and processing, offering an evaluation of the system's responsiveness.
4. **CPU Utilization**: Monitoring CPU resources consumed by the time synchronization process, especially on the dedicated core for PPS signal polling, to understand the optimization's impact on system performance.
5. **Interrupt Handling Time**: This tracks the time taken for PPS signal interrupt processing, measuring the efficiency of interrupt steering and related optimizations.
6. **Temperature**: Recording system temperature to assess thermal management effectiveness and its impact on stable performance.
7. **Power Consumption**: Measuring the system's energy efficiency during the synchronization process to evaluate deployment feasibility in power-constrained settings.

Collecting these metrics will provide comprehensive insights into each platform's performance, enabling a detailed understanding of the applied optimization techniques' effectiveness. This approach will highlight performance enhancements and potential trade-offs, guiding the optimization process and system configuration for precise time synchronization.

We plan on collecting these metrics at intervals determined based on their expected variability, the analysis duration, and the applied optimization techniques:

- **High-Frequency Metrics** (Time Synchronization Offset, Jitter, System Latency, Interrupt Handling Time): Collected every 1 to 5 seconds to capture rapid fluctuations and the immediate impacts of optimizations.
- **Medium-Frequency Metrics** (CPU Utilization, Temperature): Collected every 30 seconds to 1 minute, suitable for metrics exhibiting slower changes.
- **Low-Frequency Metrics** (Power Consumption): Collected every 5 to 10 minutes, reflecting longer-term changes.

Adjustments to the collection intervals of these metrics will be informed by both initial findings and the dynamic insights provided by real-time monitoring throughout each optimization phase. This ensures a tailored approach to effectively capture the nuanced effects of applied optimizations on system performance. Real-time monitoring, integral to our methodology, allows for the continuous observation of system performance. Based on this ongoing analysis, we will dynamically adjust the collection frequencies in response to observed periods of high variability or stability within any given metric. For example, should real-time monitoring identify a significant shift in Time Synchronization Offset subsequent to an optimization, the frequency of collection for this metric may be refined to more accurately document both the immediate impacts and the stabilization process that follows. Conversely, if minimal variation is noted in CPU Utilization over a longer span, the interval between collections for this metric may be extended, thereby optimizing our data collection strategy to concentrate resources on areas exhibiting greater volatility.

Project Goals

- **MVP**: Establish baseline time synchronization capabilities on both Raspberry Pi 5 and Compute Module 4 and conduct basic optimization techniques.
- **Expected Completion Goals**: Achieve significant improvements in time synchronization precision on both platforms through optimization techniques.
- **"Reach" Goals**: Demonstrate comparative or superior performance of Raspberry Pi-based systems against traditional GPS Grandmaster solutions.

Timeline and Milestones

Hardware Setup and Initial Configuration (February 28 - March 13)

- Tasks: Acquire hardware, set up GPS Grandmaster and clients, establish initial synchronization.
- Milestone: Hardware setup and initial synchronization.

Basic Techniques Implementation (March 11 - March 24)

- Tasks: Implement basic optimization techniques, document setup, preliminary performance analysis.
- Milestone: Basic HFT techniques setup.

Intermediate Techniques Implementation (March 25 - April 7)

- Tasks: Replicate basic techniques on CM4, implement and test intermediate techniques, update documentation.
- Milestone: Intermediate HFT techniques setup.

Advanced Techniques and Analysis (April 8 - April 21)

- Tasks: Implement advanced techniques, comprehensive data collection and analysis.
- Milestone: Advanced techniques applied and data collected.

Final Analysis and Project Wrap-Up (April 22 - April 30)

- Tasks: Statistical performance analysis, finalize report and documentation.
- Milestone: Project completion with final report and documentation.

Technology Stack and Resources

Hardware

- **Raspberry Pi 5 and Compute Module 4**: These are the primary hardware platforms for the project, chosen for their performance, community support, and compatibility with a wide range of peripherals.
- **u-blox GPS Receievers**: LEA-5T-0-003 and EVK-F9T models will be used to provide accurate time signals. These receivers are known for their accuracy and reliability in time-sensitive applications.
- **GPS Antenna with SMA Connection**: To ensure optimal signal reception and enhance the accuracy of the GPS receivers, we will use a GPS antenna that offers a 28db gain through an SMA connection.

Time Synchronization

- **PTP (Precision Time Protocol) and NTP (Network Time Protocol)**: Both protocols will be integral to our time synchronization strategy. PTP will be employed for its high-precision synchronization capabilities, essential for the project's aim of achieving nanosecond accuracy. NTP will serve as a broadly compatible and reliable method for initial clock synchronization across our devices, ensuring that our system is accurately aligned with global time standards before applying more refined synchronization techniques.
- **GPSD**: A daemon that provides a uniform interface between GPS receivers and applications. It translates and interfaces GPS data to multiple applications simultaneously, allowing for efficient and standardized access to GPS information without needing to directly interact with the hardware.
- **Chrony**: An efficient implementation of the Network Time Protocol (NTP) that will be used for synchronizing the system clocks of our Raspberry Pi units with accurate time sources. Chrony is chosen for its ability to maintain accuracy even under adverse conditions such as intermittent network connections or variable latencies.
- **linuxptp**: An implementation of the Precision Time Protocol (PTP) in Linux, adhering to the IEEE 1588 standard. It will be used for precise time synchronization of networked computers to a sub-microsecond level.
- **pps-tools**: A collection of utilities designed for managing Pulse Per Second (PPS) signals from the GPS receivers. These utilities are crucial for processing PPS signals, providing an accurate timing reference necessary for the high-level precision our project aims for.
- **python-gps**: To facilitate seamless access to GPS data, we will use the python-gps library. This Python interface allows for easy interaction with GPS receivers, enabling efficient data acquisition and processing within our system.

Monitoring and Diagnostic Tools

- **Prometheus with Grafana**: Prometheus is an open-source system monitoring and alerting toolkit, ideal for collecting and storing metrics as time series data. Grafana is a visualization platform that integrates with Prometheus to create customizable dashboards. This combination allows for real-time monitoring and visualization of system performance metrics, including CPU utilization, system latency, and temperature, enhancing the ability to identify trends and issues.
- **perf**: A performance analyzing tool in Linux for detailed profiling of CPU usage and memory performance diagnostics. It is useful for analyzing CPU cycles consumed by synchronization processes, system call performance, and hardware cache misses, helping to identify bottlenecks and optimize CPU core utilization.
- **sysdig**: An open-source tool for system-level exploration and troubleshooting. It captures system state and activity, offering insights into system calls, file and network operations, and performance metrics, valuable for diagnosing system behavior during GPS signal reception and processing.

Networking

- **tcpdump/Wireshark**: Network protocol analyzers for capturing and inspecting packets on a network. These tools can diagnose communication issues and monitor packet flow between GPS receivers and Raspberry Pi units or for NTP/PTP protocol implementations, aiding in identifying delays or errors affecting synchronization.

Data Collection and Analysis

- **Python**: Python will serve as the primary language for data analysis due to its versatility, readability, and the extensive support of libraries for statistical analysis, data manipulation, and visualization.
  - **pandas and numpy**: Pandas and NumPy will be used for data manipulation and analyzing raw data points.
  - **Plotly, matplotlib, and Seaborn**: All three are powerful for creating static, animated, and interactive vizualizations in Python. This will be used to visualize the collected metrics and any analysis.

Team Contributions

Laxmi Vijayan, *Team Lead*

- **Strategic Planning and Coordination**: Laxmi will spearhead the overall project strategy, aligning technical efforts with the project's goals at every phase, from hardware setup to final analysis.
- **Hardware Setup Contribution**: She will contribute to the hardware setup, particularly focusing on ensuring that software and hardware integrations are optimized for initial configurations and beyond.

- **Optimization Techniques Development and Implementation**: Throughout the project, Laxmi will lead the development and implementation of both basic and advanced optimization techniques, working closely with Haoyuan to ensure hardware compatibility.
- **Data Analysis and Reporting**: In collaboration with Yiqing, Laxmi will assist in preliminary data analysis, ensuring the optimizations' effectiveness is accurately measured and reported.
- **Final Reporting**: She will oversee the creation of the final report, ensuring it comprehensively covers findings, methodologies, and recommendations for future work.

Yiqing Huang, *Data Analyst*

- **Data Collection and Error Analysis**: Yiqing will be responsible for designing and implementing the data collection framework, focusing on capturing performance metrics accurately across all phases of the project.
- **Comparative Studies on Performance Metrics**: She will conduct comparative analyses to evaluate the effectiveness of optimization techniques, providing insights into the comparative performance of Raspberry Pi 5 versus Compute Module 4.
- **Performance Metrics Analysis**: In the advanced techniques and analysis phase, Yiqing will deepen her analysis, identifying trends, pinpointing errors, and deriving conclusive insights from the data.
- **Final Analysis and Insights Reporting**: Yiqing will play a critical role in the final analysis, preparing detailed reports on the project's outcomes, contributing data-driven insights to the final project documentation.

Haoyuan You, *Hardware Specialist*

- **Initial Hardware Setup and Configuration**: Haoyuan will lead the initial setup of the Raspberry Pi units and GPS modules, ensuring the hardware is primed for high precision synchronization from the outset.
- **GNSS Latency Optimization**: He will apply his expertise to optimize GNSS latency, crucial for achieving the project's time synchronization accuracy goals.
- **Linux Kernel Configuration**: Throughout the project, Haoyuan will manage Linux kernel configurations to ensure the operating system is optimized for the project's needs.
- **Advanced Hardware Optimization**: In later phases, he will implement advanced hardware optimization techniques, adjusting configurations based on data-driven insights to enhance performance further.
- **Technical Contributions to Reporting**: Haoyuan will provide detailed technical insights into the hardware setup and optimization processes for inclusion in the final project report.

Victor Li, *Network Specialist*

- **Network Infrastructure Setup and Optimization**: Victor will ensure the network infrastructure is robust, scalable, and optimized for precise time synchronization, implementing necessary adjustments to minimize latency from the outset.
- **Protocol Implementation and Adjustment**: He will manage the implementation of networking protocols and make adjustments as needed to support the precision requirements of the project.
- **Network Performance Optimization**: Victor will focus on refining network configurations and protocols during the intermediate and advanced phases, based on the ongoing analysis of system performance.
- **Contribution to Final Reporting**: He will contribute insights into the network setup, challenges, and optimizations to the final project documentation, ensuring a comprehensive overview of the network components' role in achieving project goals.

Final Deliverables

The project will culminate in the delivery of:

- **Comparative Analysis Report**: A detailed document presenting the findings from the comparative analysis, including methodology, data analysis, conclusions, and recommendations for future research or application developments.

- **Code Repository**: A GitLab repository containing all developed scripts, configuration files, and documentation necessary to replicate the study or further extend the research.