Multicore Avionics Operational Analysis

Lightning Talk 5: Design Part 2

Team: sddec24-09

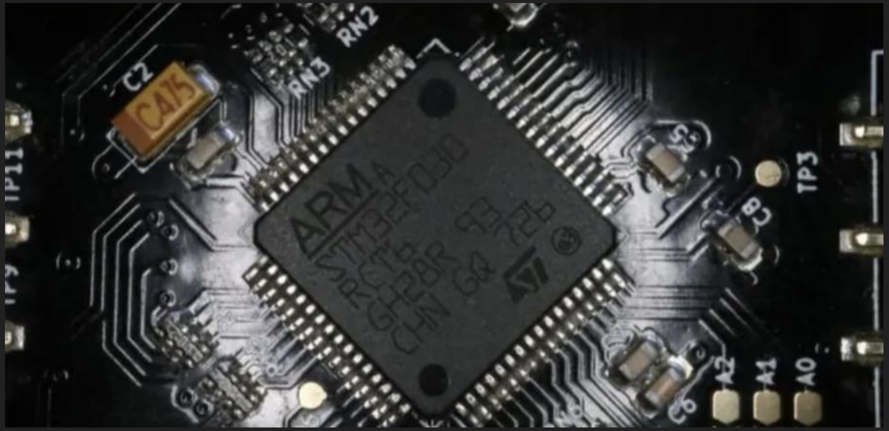
Team members: Alex Bashara, Joe Dicklin, Hankel Haldin, Anthony Manschula

Faculty advisors: Dr. Zambreno & Dr. Jones

Client: Boeing

Project Overview

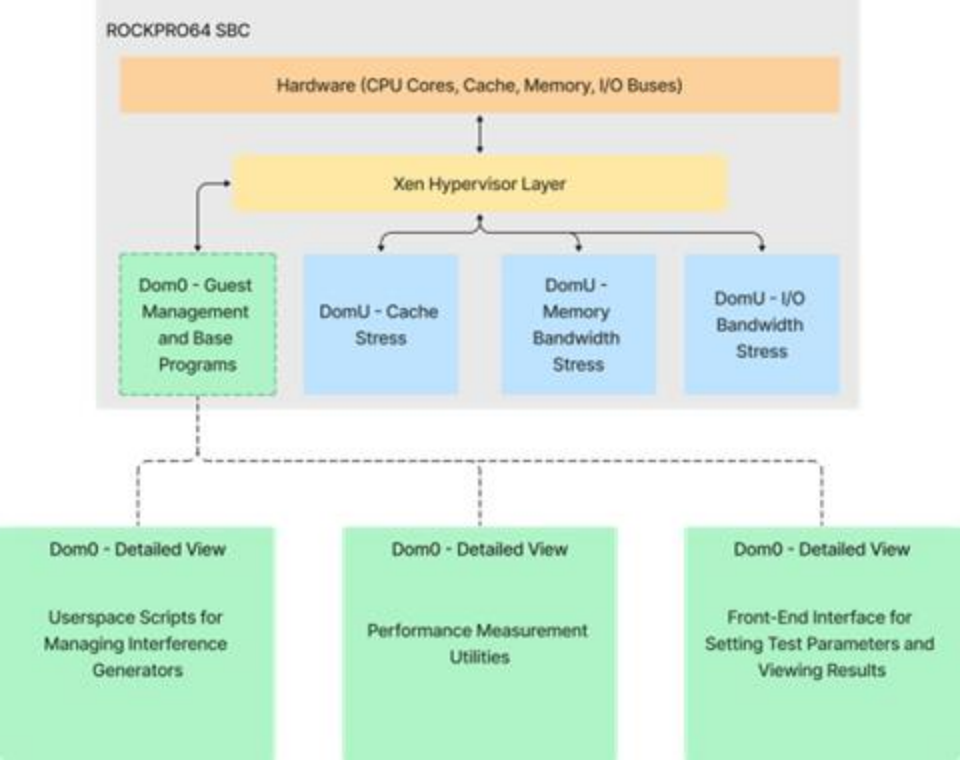
- Multicore avionics systems
  - Meet the increasing compute demand of modern avionics software with concurrent execution of programs
  - Concurrent programs competing for shared resources
    - Introduce interference & negatively affect execution timing behavior
    - Ability to examine and verify the effects of interference is critical for FAA certification
- Hardware: ARM-based SBC and bare-metal hypervisor
  - Hypervisor allows more granular control of resource allocation to programs
  - Run control applications with the system under extreme load
    - Collect and analyze data on worst-case execution time (WCET)



Problem Statement

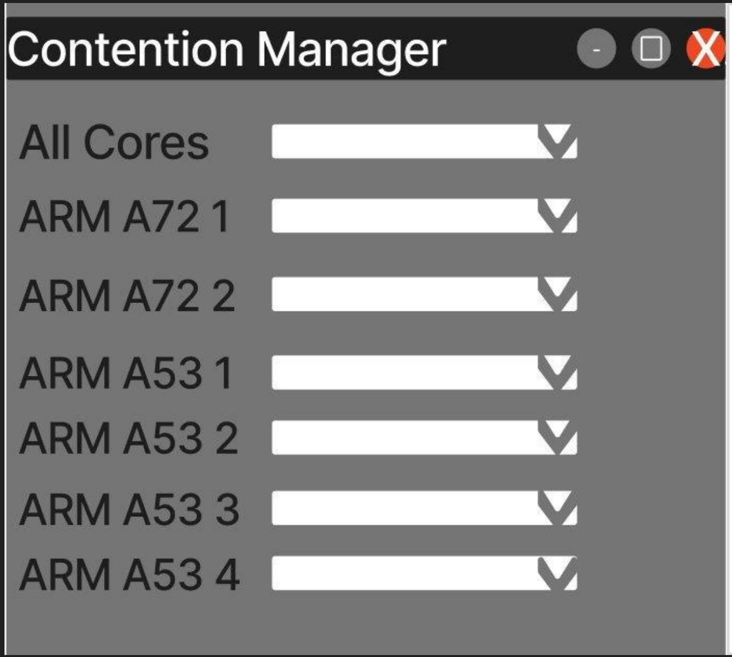
- Our project addresses a need for a suite of open-source tools to characterize interference modes in multicore avionics systems
    - Identify potential interference channels on a multicore platform
        - "Control tests" as a baseline that target each channel for use in analysis
    - Set of tools to apply stress and contention to the identified subsystems in a controlled manner
    - Set of tools/methods to demonstrate mitigation of interference channels
    - Integrate testing and analysis tools into unified suite

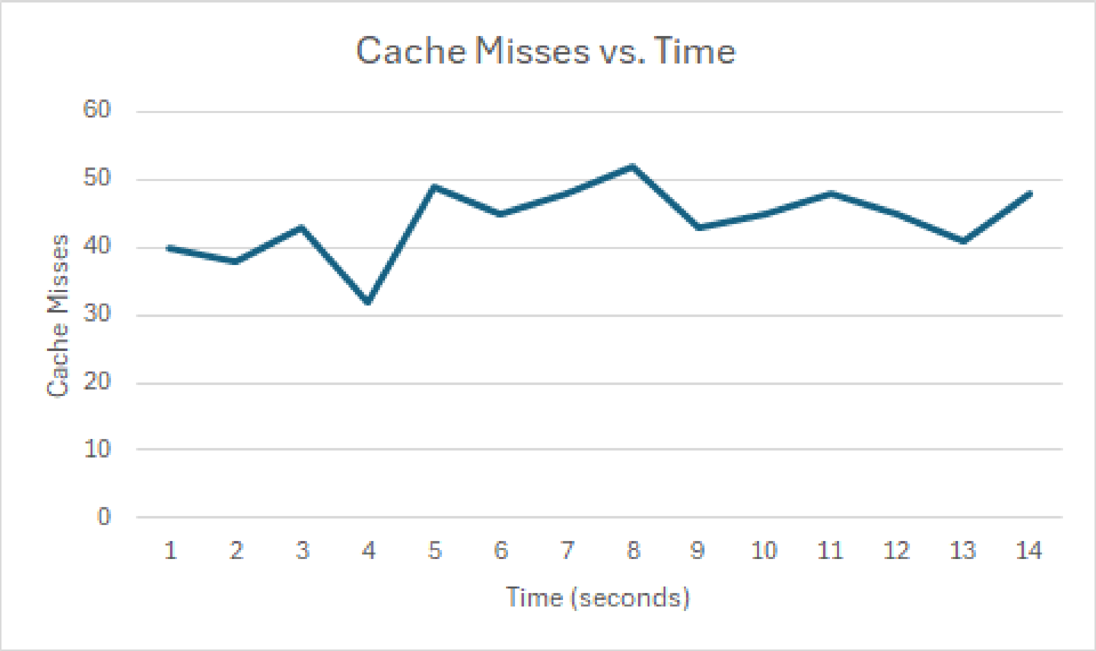
High Level Design Block Diagram


ROCKPRO64 SBC
Hardware (CPU Cores, Cache, Memory, I/O Buses)
Xen Hypervisor Layer
Dom0 - Guest Management and Base Programs
DomU - Cache Stress
DomU - Memory Bandwidth Stress
DomU - I/O Bandwidth Stress
Dom0 - Detailed View
Userspace Scripts for Managing Interference Generators
Dom0 - Detailed View
Performance Measurement Utilities
Dom0 - Detailed View
Front-End Interface for Setting Test Parameters and Viewing Results

Visuals

- Our final design will have a GUI interface that allows users to control contention on specific cores
- The application gathers real-time metrics on system performance
  - This allows the user to monitor performance with and without interference





Functionality

- How does our solution operate?
  - Front end: controlled via a command-line interface
    - User specifies which test to run, and what type of resource contention to apply (if any)
  - Back-end: Perform hypervisor setup and data aggregation
    - Base test is started in Dom0
    - Resource contention generators started in one or more DomU environments
    - Performance data collection is done via Linux PERF framework
      - Can grab data like program execution time, hardware performance counters

Technology Considerations

- RockPro64 Single Board Computer
  - Cheap, easily available
  - Hardware documentation readily available
  - On the higher end of ARMv8 SOCs in terms of performance
  - Some alternatives
    - FPGA-basesd solution (such as Xilinx UltraScale+ MPSoC)
      - Combines an ARM SoC with FPGA development board hardware
      - Very Expensive and complex

Technology Considerations

- Xen Hypervisor
  - De facto hypervisor for Embedded Linux systems
  - Highly configurable and adaptable to many hardware platforms
    - Ability to run bare-metal programs (i.e., run assembly code directly on hardware without overhead of a guest operating system)
  - Limited documentation on configuring and building for ARM-based Linux systems (like ours)
  - Complexity added by the configurable nature leads to difficulty in debugging setup issues
  - Alternatives
    - None – Xen is the only type 1 (bare-metal) hypervisor available that supports ARM on Linux

Areas of Concern and Development

- Primary Concerns
  - Hypervisor configuration
    - Configuring the hypervisor to behave exactly how we need it to is proving to be difficult
  - Performance data collection
    - Given that our hypervisor setup is still ongoing, it will be difficult to determine how effective our solution is in determining a bound on WCET
    - This could leave us with less time than anticipated to implement thorough performance measurement

For the Audience

- Conclusion
  - We are developing a solution guided by our client's input and our own research. The complex nature of our project requires us to quickly iterate on our ideas and adapt the ones that work. Ultimately, we hope to produce an open-source library that addresses industry needs.
- Questions?