RISC-V for Real-time MCUs – Software Optimization and Microarchitectural Gap Analysis

Robert Balas, Luca Benini

BACKGROUND & MOTIVATION

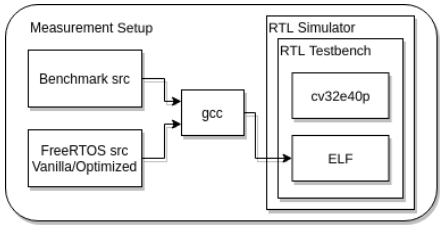
- Increasing RISC-V adoption in IoT/embedded Systems (MCU)
- Dealing with Real-time constraints
- Analyze RISC-V real-time support

NEW INSIGHTS

- Identified and quantified hardware gaps in the RISC-V Ecosystem with respect to real-time
- RISC-V solutions lag behind ARM Cortex Series

DESCRIPTION

Goal: Characterize hardware gap with respect to ARM Cortex Series



Setup

- Baseline Characterization using the `cv32e40p` and `vanilla FreeRTOS`

Software Optimizations FreeRTOS

- Use features of hardware and compiler options

Simulation and measurments

- Cycle accurate
- On synthetic and real-life (power control unit) benchmark
- Measure context switch time and interrupt latency

QUANTITATIVE IMPACT

Optimized FreeRTOS	Context Switch Time Improvement (WCET)	Int. Latency Improvement (WCET)
vs Vanilla FreeRTOS	+33%	+20%
vs ARM Cortex-M3 FreeRTOS	-26%	-50%

- After optimizations, most difference attributable to hardware

SUMMARY AND CONCLUSION

- ARM nested interrupts lower interrupt latency
- ARM automatic register stacking lower context switch time and interrupt latency
- Potential for new RISC-V extensions