Extending FirePerf to Userspace

Raghav Gupta, Alex Hao, Reza Sajadiany

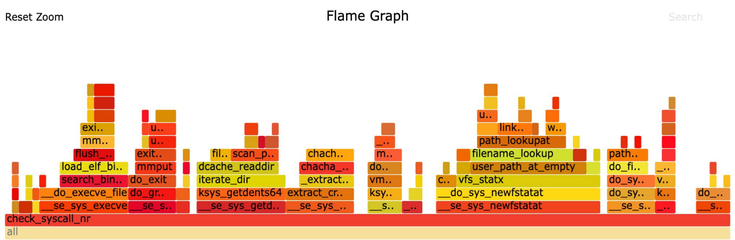
Full-Fidelity Full-Stack Performance Profiling using Realtime Call-Stack Reconstruction on Cloud FPGAs

Overview

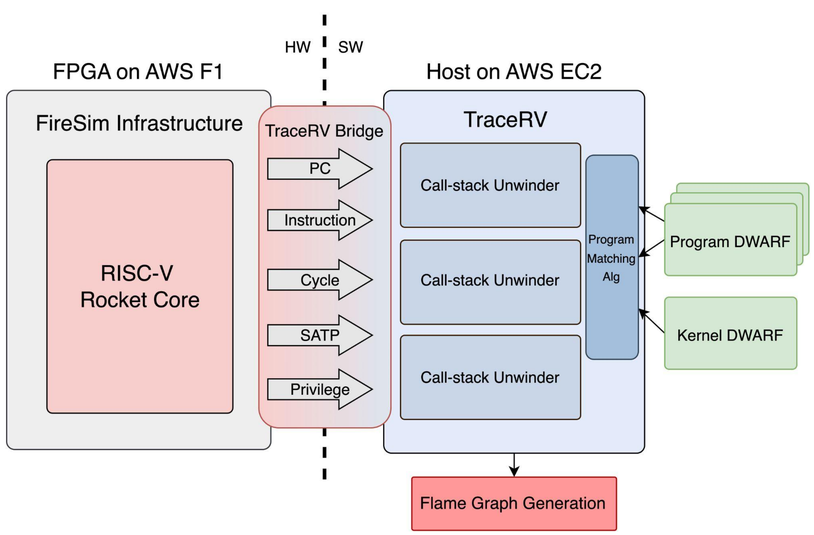
- High-fidelity introspection into HW/SW behavior via call-stack reconstruction and flame graphs
- Simply expose performance issues not discoverable by traditional software profilers
- Original FirePerf only supports the kernel, we aim to provide full-stack support

Motivation

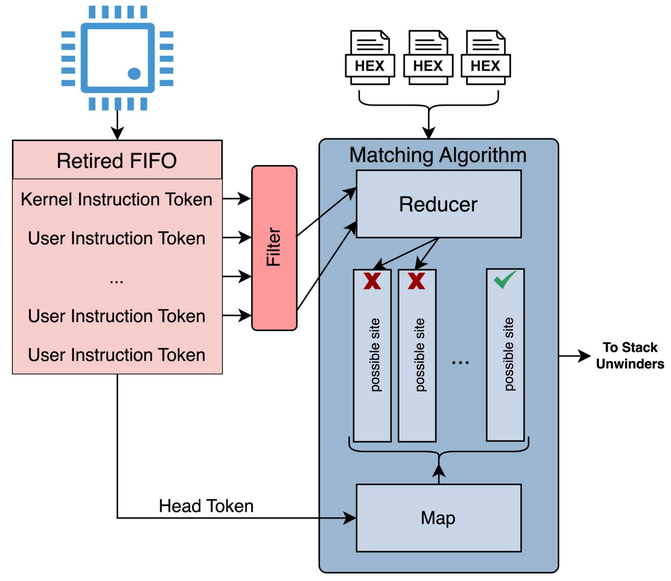
- Conventional profilers (perf, strace) are in-band
  - Low sampling frequency
  - Perturb target system at high frequency
- Out-of-band profilers completely decouple tool and target ⇒ minimal overhead but often slow
- Our final goal looks like this, but for user+kernel modes:



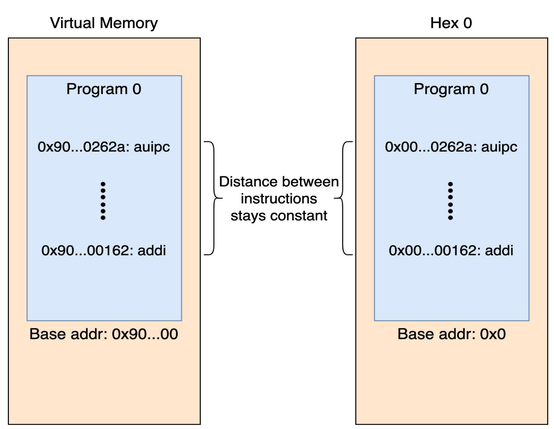
System Architecture



Matching Approach

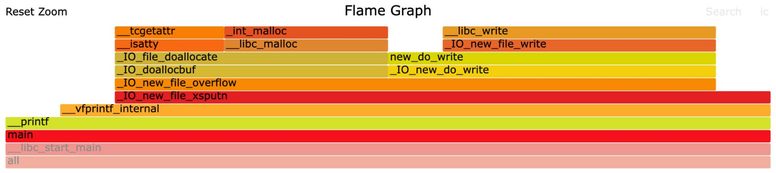


- Need to match traced instructions to function names in source code
- Gather trace tokens from Rocket Core
  - Instrumented trace port to provide {pc, instr, cycle, satp, priv}
- Buffer tokens in a queue ⇒ generate local history
- Match instructions per process with history + hexdumps
- Assumption: instruction offsets preserved under Virtual Memory
- For each instruction:
  - Find possible sites (binary + page) ⇒ map stage
  - Calculate offsets to other instructions
  - Eliminate possible sites by checking matches at offsets ⇒ reduce stage

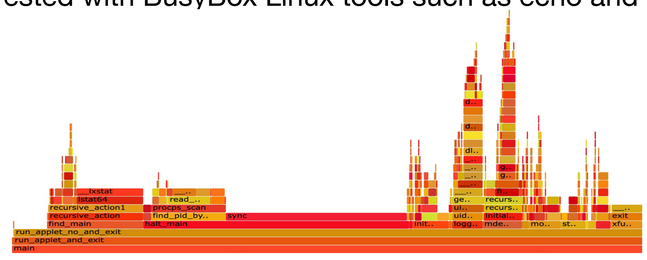


Proof-of-Concept

- Implementation tested with simple C program
- Successfully generated flame graph
- Core spends significant time in library write and malloc functions



- Tested with BusyBox Linux tools such as echo and cat



- Detailed report available at: https://people.eecs.berkeley.edu/~kubitron/courses/cs262a-F22/projects/reports/project9_report_ver2.pdf

Future Work

- Optimizations
  - Cache satp ⇔ binary mappings, reduce possible sites
- Multithreading / Multi-Core Processor
  - Threads share satp/PID
  - Instrument target to differentiate
- Dynamic Linking and Position Independent Code
  - DWARF/hexdump doesn't contain actual instruction addresses and jump offsets
  - Instrument target + process linker for relative offsets and base addresses

Acknowledgements

We would like to acknowledge mentor Sagar Karandikar, SLICE Lab, FireSim and Chipyard Developers.
This work is based on "Fireperf: Fpga-accelerated full-system hardware/software performance profiling and co-design" by S. Karandikar, A. Ou, A. Amid, H. Mao, R. Katz, B. Nikolic´, and K. Asanovic. Available: https://doi.org/10.1145/3373376.3378455