Internship Project Report

January 14, 2016

1 Motivation

The main motivation was to establish an alternate flow to instruction tracing to characterize highly threaded benchmarks and its simulations on next-gen SoCs. We have instruction traces of SPEC Benchmark suits, which are used for core simulations. But simulating SoCs with 32 or higher core counts require replicating core simulators multiple times. This is very slow and practically impossible for 8 or higher cores. Thus we need an alternate simulation methodology to predict the performance of upcoming SoCs.

2 Goal

Our team was building an Abstract CPU simulator, which is more like a memory trace player and injects memory traffic to the northbridge (sometimes termed as Data Fabric). Our goal was to capture memory transactions at post L1 boundary on current gen processors. This is called as bus trace, which will be used as an input to the abstract CPU simulator. For each benchmark, we have to collect bus traces and validate statistics from it with respect to silicon data to ensure that the captured bus trace actually represents the benchmark behaviour.

3 What we did

We captured good representative bus traces which will be used as an alternative to instruction trace and will be needed to drive the abstract CPU model. Later, we also focused on the simulator itself for simulation glitches and making it more robust and more correlated wrt cycle accurate simulators. The bus tracing work is divided into two parts: Trace collection and trace validation.

3.1 Trace Collection

- We collected the transactions routed through the northbridge using amd tools. It contains the transaction details eg type (RdBlkE/M/S etc) along with northbridge timestamps.
- This is done on a Kaveri system (Steamroller architecture) making L2 direct mapped (as we can't turn off L2 completely).
- As memory is limited we can trace a very small portion of the benchmark at once. (8GB buffer can capture around 1.2 secs of execution). Monitoring the benchmark using other amdtools we can find region of interests, having high DRAM b/w and RdBlkE/M/S/VicBlk transactions.

3.2 Trace Validation

- In this phase we compared the RdBlkE/M/S/VicBlk PTI values we got from collected bus trace with those stats over the entire benchmark collected from amd tools (referred as Silicon data). This is to ensure that bus trace is representative of the benchmark.
- For a single benchmark, we collected many traces. Using trace reduction, we assigned weights to each of them and minimized the error between trace data and silicon data.
- Lastly we went down to both flows, instruction trace+core cycle accurate simulator and bus trace+abstract CPU simulator to correlate how two models behave with respect to actual silicon execution.

3.3 Enhancing Abstract model

The last correlation study showed there are possible enhancements which can make Abstract model more accurate to actual execution. We ruled out some simulation bugs in cache hierarchy and categorized the abstract cache model misses into different types of transactions. Later, we tried to attach the cycle accurate cache model (ASPEN) to the abstract CPU model to increase it's simulation accuracy.