FPGA Accelerated Parameterized Cache Simulator

***Abstract*—Design space exploration of caches enables the architect to choose the right configuration based on metrics or constraints such as hit rates, power, area, and timing. We propose to implement an FPGA accelerated parameterized two level cache simulator with prefetching. The key motivation behind the idea is the speed with which the design space exploration can be carried out as compared to software based simulators. This tool can in turn be used to compare the efficacy of results generated by tools such as CACTI, ChampSim and so on. Our tool is expected to report cache metrics such as hit/miss rates with and without prefetching for different cache configurations, along with the timing, area and power information as implemented on the FPGA.**

***Index Terms*—Field programmable gate array, Prefetchers, Cache Simulator, Cache Metrics**

I. Motivation

Processors have been implementing different levels of caches in order to reduce the well-known Processor-Memory performance gap. Several types of cache configurations have been studied [3]. A large cache may result in more hits but will consume more area and power. There is always a trade off between area/power and cache hit rate. So, choosing an optimal cache configuration based on the metrics and desired power, area and timing constraint is essential. A cache simulator which can run several parallel configurations can speed up this task of design space exploration. A hardware based (FPGA based) cache simulator will be more accurate and also will speed up the task as compared to software/model based simulators [2].

II. Implementation and Results

We have implemented an FPGA accelerated parameterized trace driven L1 LRU cache simulator. It has been implemented in Verilog and programmed on to a Xilinx Zynq based FPGA development board: Zedboard. Zynq-7000 SOC architectures include two major sections: the Programming System (PS) which consists of the dual ARM Cortex-A9 processor and Programmable Logic (PL) – FPGA 7 series. We have implemented a Cache "simulator" which comprises of just the index and tag (without the data), since our objective is to "simulate" a cache and not to perform a hardware implementation of a cache. The parameters that can be varied in the simulator are: cache size, block size and associativity (in turn, the number of sets). We currently implement an LRU replacement policy for the set associative cache. The cache has been programmed on the PL while the address traces and cache configuration are being transferred from an SD card interface through the PS. The cache simulator on the PL in turn returns hit/miss metrics back to the PS which in turn writes the output metrics on to SD Card as shown in Figure 1.

Currently, next line prefetcher is also integrated with our cache. We have run five traces on the set-associative cache with and without prefetcher for a particular cache structure.

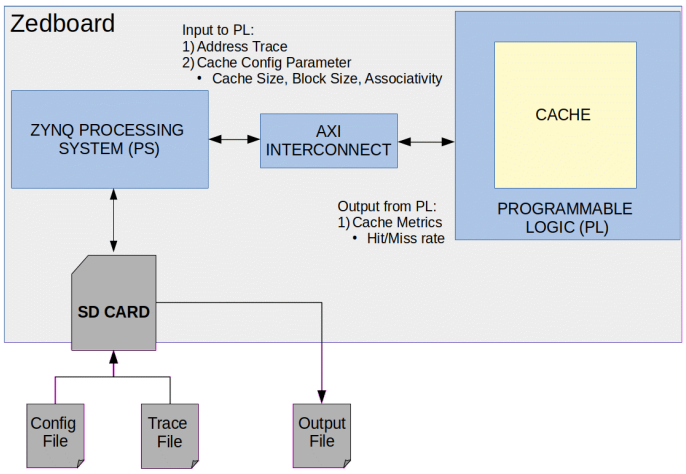

Figure 1. Framework of Cache Simulator

The hit rate for Direct Mapped and Set Associative cache configurations are shown in Table I. The resources utilised on the FPGA for the set associative configuration with prefetcher are: LUTS: 77.66%, flip-flops: 42.71%.

Table I
Hit Rate (in %) of Direct mapped (DM) and Set Associative (SA) Cache with and without Prefetcher (P). Cache Configuration – Cache Size – 32KB, Block Size – 4 words and Associativity – 4.

Trace	Trace Count	DM	SA	DM+P	SA+P
gcc	515683	97.29	97.68	97.43	98.3
gzip	481044	66.78	66.79	66.81	66.82
mcf	727230	50.502	50.503	75.22	75.23
swim	303193	95.78	96.18	96.24	97.82
twolf	482824	99.27	99.37	99.32	99.61

Implementing different prefetchers and optimising the simulator's structure is currently work in progress. The next aspect of our work is to run multiple cache configurations in parallel using inclusion property of associativity and set sizes demonstrated in [1]. We also intend to implement an L2 cache structure along with the L1 cache, which follows the same inclusion property. Finally, we intend to compare results of our implementation with software based simulators like CACTI for area, power estimations and Champsim for hit/misses, accuracy and speed.

References

[1] Josef Schneider, Jorgen Peddersen, and Sri Parameswaran. "A scorchingly fast FPGA-based Precise L1 LRU cache simulator." In 2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 412-417. IEEE, 2014.

[2] Hari Angepat, Derek Chiou, Eric S. Chung, and James C. Hoe. "Fpga-accelerated simulation of computer systems." Synthesis Lectures on Computer Architecture 9, no. 2 (2014): 1-80.

[3] Brais H, Kalayappan R, Panda PR. A Survey of Cache Simulators. ACM Computing Surveys (CSUR). 2020 Feb 5;53(1):1-32.