Journal of King Saud University – Computer and Information Sciences (2017) 29, 410–425

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

A high level implementation and performance evaluation of level-I asynchronous cache on FPGA

Mansi Jhamb [a,*], R.K. Sharma [b], A.K. Gupta [b]

[a] University School of Information and Communication Technology, Guru Gobind Singh Indraprastha University, Dwarka Sector 16C, New Delhi 110078, India
[b] Department of Electronics and Communication Engineering, NIT Kurukshetra, India

Received 19 February 2015; revised 9 June 2015; accepted 27 June 2015
Available online 31 October 2015

KEYWORDS

Asynchronous;
Handshaking;
Cache

Abstract To bridge the ever-increasing performance gap between the processor and the main memory in a cost-effective manner, novel cache designs and implementations are indispensable. Cache is responsible for a major part of energy consumption (approx. 50%) of processors. This paper presents a high level implementation of a micropipelined asynchronous architecture of L1 cache. Due to the fact that each cache memory implementation is time consuming and error-prone process, a synthesizable and a configurable model proves out to be of immense help as it aids in generating a range of caches in a reproducible and quick fashion. The micropipelined cache, implemented using C-Elements acts as a distributed message-passing system. The RTL cache model implemented in this paper, comprising of data and instruction caches has a wide array of configurable parameters. In addition to timing robustness our implementation has high average cache throughput and low latency. The implemented architecture comprises of two direct-mapped, write-through caches for data and instruction. The architecture is implemented in a Field Programmable Gate Array (FPGA) chip using Very High Speed Integrated Circuit Hardware Description Language (VHSIC HDL) along with advanced synthesis and place-and-route tools.

1. Introduction

In technical market the majority of processors are embedded systems (Tennenhouse, 2000). The microcontrollers, DSPs, FPGAs and their combinations as SoCs provide solution for embedded system (George et al., 1999). Since FPGAs have shown significant advances in the terms of speed, density and storage capacity, they have become most commonly used embedded general purpose computing environment. Owing to the diverse functionality of FPGAs, complexity level increases thereby increasing the number of logic gates in the

* Corresponding author.
E-mail addresses: mansi.jhamb@gmail.com (M. Jhamb), mail2drrks@gmail.com (R.K. Sharma), anilg699@rediffmail.com (A.K. Gupta).
Peer review under responsibility of King Saud University.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2015.06.003
1319-1578

circuit. This results in increased energy dissipation in the system (Anderson and Najm, 2004; Shang et al., 2002). Another concern is the increasing difference in the speed of the processor and main memory which creates a bottleneck while accessing the data from the memory. While the programers want unlimited fast memory, it is too expensive to be achieved. Hence, the optimal solution is to provide a memory hierarchy in which each level is faster and more expensive per byte than the immediate higher level (Patterson and Hennessey, 2003). The cache is the first level in this memory hierarchy. Modern multi-core processors support multiple levels of cache to improve performance. In most such systems, caches (last level caches) are often shared by the processors for concurrent applications (Warrier et al., 2013). According to the principle of locality of references the data and the instructions exhibit temporal and spatial locality. This principle forms the basis of cache memory hierarchy, leading to the fact the processor performance depends on the speed of cache rather than slow main memory. 45% of energy consumption of a processor is contributed by cache hierarchy (Segars, 2001).

Research reveals that the adjustment of the cache memory parameters can contribute to reduction in energy consumption to 62% (Ross et al., 2005), leading to the performance enhancement by 30% (Ross et al., 2004). The complex task of determining an appropriate cache configuration for a dedicated application, simulation and analysis takes considerable computational time. For reduction in the area and energy dissipation without compromising the performance, designers need to develop strategies for design space exploration. There are a number of ways to implement the L1 cache and the selected design strategy impacts the complete behavior of processor system (Bahar et al., 1998; Milenkovic et al., 2003). For the implementation parameters to be freely selected, an RTL model is desired. In the logic simulation, the desired cache configuration can be employed after the cache model parameters are set. This work presents a high level implementation of a micropipelined asynchronous architecture of an L-1 cache using FPGAs.

2. Related work

In embedded-system design one has to compromise between energy-dissipation, performance and cost. The critical task of choosing the best cache architecture involves the selection of total cache size, amount of associativity, cache line size and several other architectural options. These characteristics greatly influence the hit rate and the energy consumption in cache access. The power consumption occurs when (i) the cache is accessed. (ii) The data are transferred from/to the next memory level during a cache miss and also by the idle processor when miss occurs.

Associativity segregates a cache into a number of ways, each of which is looked up concurrently during a cache access. For some programs, the cache hit rate is improved on increasing the number of ways to two or four (Patterson and Hennessey, 2003), beyond four the improvement is not significant. More ways imply more concurrent look-ups per access leading to more energy per access: a direct mapped cache uses only 30% of the energy per access as a four way set associated cache (Reinman and Jouppi, 1999). Highest possible associativity is desired in performance-oriented applications. Table 1 summarizes the features of several cache architectures in embedded microprocessors.

Table 1 Cache architectures in embedded microprocessors.

Processor	Instruction cache			Data cache		
	Size	As.	Line	Size	As.	Line
AMD-K6-IIIE	32 K	2	32	32 K	2	32
IBM PPC 750CX	32 K	8	32	32 K	8	32
IBM PPC 7603	16 K	4	32	16 K	4	32
IBM 750 FX	32 K	8	32	32 K	8	32
IBM 403GCX	16 K	2	16	8 K	2	16
IBM Power PC 405CR	16 K	2	32	8 K	2	32
Intel 960IT	16 K	2	N/A	4 K	2	N/A
Motorola MPC8240	16 K	4	32	16 K	4	32
Motorola MPC823E	16 K	4	16	8 K	4	16
Motorola MPC8540	32 K	4	32/64	32 K	4	32/64
Motorola MPC7455	32 K	8	32	32 K	8	32
Xilinx Virtex IIPro	16 K	2	32	8 K	2	32

A novel configurable cache architecture (Zhang et al., 2005) incorporates three configurable cache parameters, these are configured on setting a few bits in the configuration register. The cache can be configured in software as either direct mapped, two-way or four-way set associated while utilizing the full capacity of cache. Such configuration is attained by employing a technique called way concatenation (Zhang et al., 2003a). The technique called line concatenation (Zhang et al., 2003b) is employed to configure the cache line size. Several architectures for reducing the energy consumption in the cache are reported in the literature. One of them is the partitioning of cache into several smaller caches (Racunas and Patt, 2003). This results in the reduction of access time and the power cost per access. Another approach, filter cache (Kin et al., 1997) trades performance for power consumption by the filtration of cache references through an unusually small L1 cache. An L2 cache similar in structure and size to a L1 cache is placed after the filter cache for minimizing the performance loss. A distinct alternative, selective cache ways (Albonesi, 2000) renders the ability to disable a subset of ways in a set associative cache during the intervals of modest cache activity whereas the complete cache is operational for cache-intensive periods. In another approach the conventional unified data cache can be replaced with multi specialized caches. Each one handles different type of memory references as per their specific locality characteristics (Lee et al., 2001). These options make it possible to improve in terms of performance and power efficiency. Lastly Jin and Cho, 2006 achieves power saving in L1 cache by exploiting spacial locality. A useful state of the art design of a pipelined asynchronous cache system is as reported (Nyström et al., 2003). In their work a pipelined cache system is designed for use in an asynchronous MIPS R3000-compatible processor. This design (Nyström et al., 2003) being the first of its kind, represented a QDI (Quasi-delay insensitive) message passing implementation of L1 cache for high speed asynchronous CPU; many assumptions were made for design compatibility with now-obsolete R3000 cache architecture.

Asynchronous paradigm has potential advantages of low power, low noise, modularity, composability and ease of integration to an existing system, proving them to be an interesting alternative in the several applications. The clock distribution and alignment consume a sufficient amount of resources like wiring, power and area (Elrabaa, 2012). The asynchronous