International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Implementation of Enhanced Cache Controller with Multi-Sized Outputs

Sweety M Pinjani[1], Prof. V. B. Baru[2]

[1]PG Student, E&TC Department, Sinhgad College of Engineering, Pune, Maharashtra, India

[2]Associate Professor, E&TC Department, Sinhgad College of Engineering, Pune, Maharashtra, India

Abstract: ***The major role of cache controller is reduction in the data transfer access time between the CPU and cache. The controller which is capable of handling a system with many cores is designed. It is also capable of giving multi-sized output like 1, 2, 4, 8 and 16 bytes. The operation of this controller is defined by 6 different states like Fetch Data, Read Cache, Write Cache, Read Memory, Write Memory and Give Data. The controller can be used for different ways of caches. The design is developed using Verilog HDL. A test bench is also developed to test the functionality of the design.***

Keywords: Cache, cache controller, system performance, hit, miss.

1. Introduction

The technology is evolving at a fast rate. As a result, there is a requirement for faster microprocessors. One of the major components inside the microprocessors is the memory which is used to store the data. Unfortunately, the memories are slow as compared to processors. Due to this fact, it is difficult to utilize the microprocessor efficiently. The memory which is closest to the processor is the cache. It is also the fastest and the most expensive of all the memories. So, it is a major concern to utilize this memory most effectively.

Cache controller is used to control the read and write operation to this cache memory. It is also used to control the data transfer operations between cache and processor as well as cache and the main memory. Also, the controller needs to be fast enough to handle the massive data transfers between the processor, the cache and the main memory.

Many ways have been proposed for this cache controller design. One of the ways is to design a pipelined controller. But, this solution results in the increased complexity of the design. In the multi-core system, the cache memory is shared between different cores. So, the controller needs to be capable of handling multiple applications simultaneously. Another way is to increase the cache memory size. However, there is a tradeoff, where the cache access time increases with the size. Applying multiple ways and block sizes to the cache reduces the miss penalty, but it results in complex design. Thus, a simple cache controller is needed. One such cache controller is proposed. This controller can work with 2 different agents or processor cores. It also gives the user a facility to give multi-sized output data as per user requirements.

2. Proposed Methodology

A. Cache Memory

Fig 1 shows the basic architecture of the cache memory. The cache memory shown has 4 different ways. This memory has a 32 bit address, which is divided into various parts like tag, set, block and byte. The byte offset is used such that the cache can provide variable data like 1, 2, 4, 8 and 16 byte. The block select address is used to select any word inside the cache. The set select address is used to select the set in every ways inside the cache.

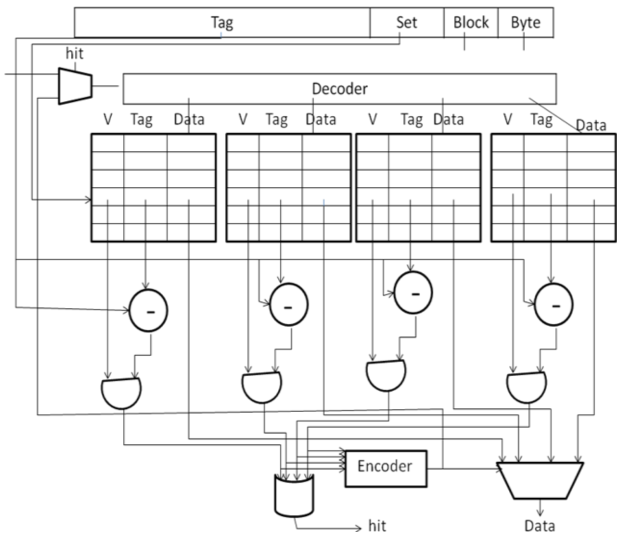

Figure 1: Block Diagram for Cache Architecture [1]

Cache operations are explained below:

1) Write operation: Any one way will be enabled to write the data inside the cache. Any empty way or any way with the same input tag register will be selected for cache write. The set of each ways will be selected according to the set select address.
2) Read operation: All cache ways will be enabled for the read operation. The set select address will decide which set will be used. The tag address provided by the processor will be compared with the tag register inside the cache. If the tag address matches and the valid bit are set, the data is present inside the cache. This scenario is termed as cache hit.

B. Cache Controller

Cache controller is used to control the operations of the cache. These operations can be explained with the help of state machine diagram as shown in Fig 2. Table 1 shows the various states involved in the controller operation.

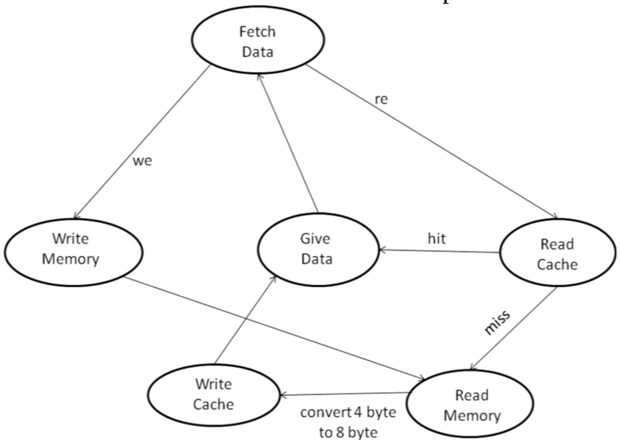

Figure 2: State diagram for the Cache Controller [1].

The various states involved are explained below:

1) Fetch Data: This state will check whether the processor request is a read or a write operation. Within this stage, the controller will keep the busy flag until it goes to the other stage.
2) Read Cache & Give Data: In this stage the cache will be checked. If it is a hit, the data in the cache will be sent to the CPU. If it is a miss, the controller will go to the next state which is the Read Memory State.
3) Read Memory: The data which is not found inside the cache is read from the main memory in this state.
4) Write Cache: this state will use the previously stored data in the temporary register and write it to the cache. If there is a cache miss, it will return to Read Cache & Give Data State. Else, it will go to Fetch Data State waiting for the next instruction.
5) Write Memory: Whenever there is a write request from the processor, the data is writing into main memory using this state.

Table 1: Controller States.

State used in design (state)	*Cache State*
000	Fetch Data
001	Read Cache
010	Write Memory
011	Read Memory
100	Write Cache
101	Give Data

In the design, user has the ability to select the output data of multiple sizes like 1, 2, 4, 8 or 16 byte. The size for the output is selected depending on the size select input as shown in the Table 2.

Table 2: Multiple sized outputs in the design [1]

Size Select Input (szsel)	*Selected Output Size*
000	1 byte
001	2 byte
010	4 byte
011	8 byte
100	16 byte

The design gives the capability to allow read of data by multiple core simultaneously. The various signals involved in multi-core operation are shown in Table 3.

Table 3: Signals in multi core operation.

Signal used for multi core	*Meaning of signal*
read_core1	Read request from core1
read_core2	Read request from core2
cache_out	Output data from cache
core1_out	Output data from core1
core2_out	Output data from core2

3. Results

The simulation results of the controller designed are divided into 3 parts. The first part shows the state changes within the cache controller when the data is input using a write request. The second part shows the multiple sized outputs obtained from the controller as per the user request. The third part shows that the cache can be accessed by multiple cores simultaneously. The simulation results showing cache state changes are shown in Fig 3. Whenever the data is input it is first written into the main memory. The data from main memory is then read and stored into the cache. This data can then be read from cache and given to the user.

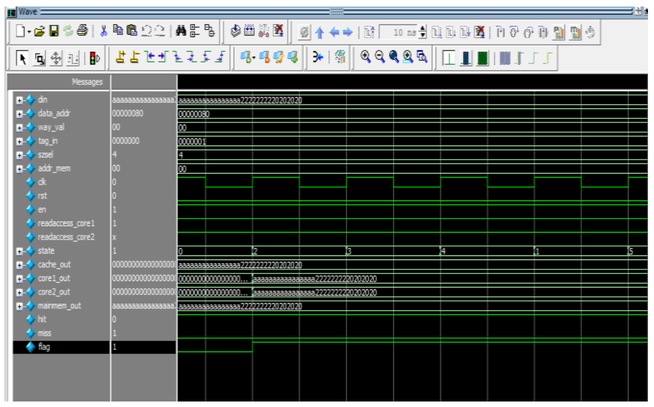

Figure 3: Simulation results showing state changes

The results for multiple sized output data are depicted in Fig 4 below. As shown in Fig 4, when the user gives szsel=1, the output data size is 2 bytes. Similarly, for szsel=2, the output data is of 4 byte. For szsel=3 and szsel=4, the output size of 8 byte and 16 byte are obtained respectively.

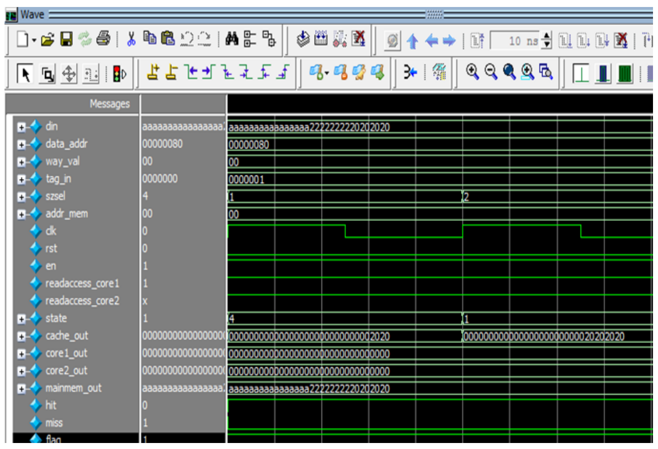

Figure 4: Simulation results showing multi sized outputs

The design is capable of handling multiple cores simultaneously. This is shown in Fig 5. When both core1 and core2 request for accessing the data from cache by asserting their read signals to 1, the output data is available at the outputs core1_out and core2_out of the core1 and core2 respectively.

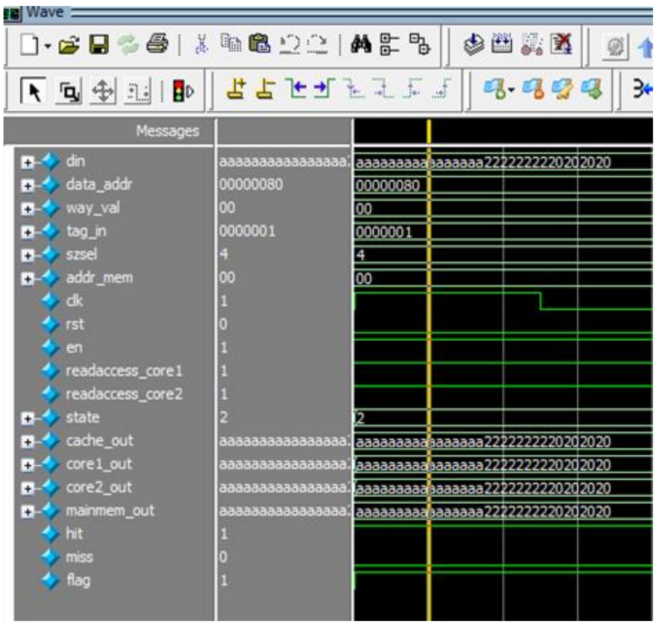

Figure 5: Simulation results showing multiple core access.

4. Conclusion

The controller designed is capable of handling a system with many cores. Thus, it supports multi-core architectures. It also gives multi-sized output like 1,2,4,8 or 16 bytes. It can also be used for caches with different ways. This controller reduces the number of instruction cycles required for providing the data requested by the processor to 1 instruction cycle. This is due to the fact that the data from the cache can be accessed in one processor instruction cycle. The design can be used for various different types of cache with various different sizes of data like 1, 2, 4, 8 and 16 byte.

References

[1] Siti Lailatul Mohd Hassan, “Multi-Sized Output Cache Controller”, *2013 International Conference on Technology, Informatics, Management, Engineering & Environment (TIME-E 2013) Bandung, Indonesia, June 23-26,2013.*

[2] David Money Harris, Sarah L. Harris, *Digital Design and Computer Architecture*, Morgan Kaufmann, March 2007, pp. 476.

[3] Alokika Dash, Peter Petrov, "Energy-Efficient Cache Coherence for Embedded Multi-Processor Systems through Application-Driven Snoop Filtering", *IEEE Journal of Solid State Circuits, 2006.*

[4] David E. Culler, Jaswinder Pal Singh, Anoop Gupta, *Parallel Computer Architecture: A Hardware/Software Approach*, Gulf Professional Publishing, 1999, pp. 381.

[5] Kunle Olukotun, "Multilevel Optimization of Pipelined Caches", *IEEE Joumal of Solid-State Circuits, October 1997.*

[6] Apoorv Srivastava, "1900-MHz CMOS 4-Kbyte Pipelined cache", in *IEEE Joumal of Solid-State Circuits*, 1995, pp.1053.

[7] Roy W. Badeau, "A 100-MHz Macropipelined V AX Microprocessor", in *IEEE Joumal of Solid-State Circuits*, Vol. 27, No. II, November, 1992.

[8] Daniel W. Dobberpuhl, "A 200-MHz 64-bit Dual Issue CMOS Microprocessor", in *IEEE Joumal of Solid-State Circuits*, Vol. 27, No. II, November, 1992.

[9] Vipin S. Bhure, Praveen R. Chakole, "Design of Cache controller for Multi-core Processor System", in *Intemational Joumal of Electronics and Computer Science Engineering*, pp.520.

[10] Badri Ram, *Advanced Microprocessors and Interfacing*, Tata McGraw-Hill Education, Sept 200 I, pp.289.

Author Profile

Ms. Sweety M. Pinjani has done BE in Electronics and Telecommunications from Sant Gadge Baba Amravati University, Amravati. She is currently pursuing ME in VLSI & Embedded Systems (E&TC) from Sinhgad College of Engineering, Pune. Her areas of interest are ASIC, SOC and IP Verification.

Prof. V. B. Baru is an Associate Professor in Electronics and Telecommunication Department at Sinhgad College of Engineering, Pune. He has done BE in 1993 from College of Engineering, Pune and completed ME in 1999 in Electronics and Telecommunications. He is pursuing Ph. D from College of Engineering, Pune. He has 20 years of Teaching Experience and published more than 50 papers in national and International level journals. He is author of two books ‘Electronic Product Design’ by Wiley Publication and ‘Basic Electronics’ by Dreamtech Publication. He has guided more than 100 UG students and about 25 PG students for their Dissertations.