VOL. 10, NO. 9, MAY 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

ERROR FREE FILE PLACEMENT IN LARGE DATABASE MANAGEMENT SYSTEM

Evan Nicholas Dawson and G. Merlin Sheeba
Department of Electronics and Tele-Communication Engineering, Faculty of Electrical and Electronics, Sathyabama University, Chennai, India
E-Mail: dawson.evan@gmail.com

ABSTRACT

In growing need of large database handling and analyzing becomes a greater challenge. There are technologies like parallel computing and shared memory technology to split up the processing task and execute it at much ease and convenience. However, there are situations while specific large files which are under query process and the same file cannot be updated or placed from server as there will be a conflict on read/write operation in memory system. Therefore, current technique suggest with priority assignment to handle the issue of concurrency. However, this proposes a methodology to handle up the file placement more successfully without priority assigned but on approach called conditional basis, using this technique we will be able to implement both write and read operation simultaneously without affecting presently running files and we will also be able to ensure the data packets are not lost due to waiting on queue.

Keywords: DBMS (Database Management), file placement, parallel computing, shared memory architecture, CREW.

1. INTRODUCTION

Database Management has become most dominant domain in every business sector. Starting with early manual system, revolution began in the 1950's with magnetic tapes for storage which was purely sequential access system. Late 60's and 70's came up with file system technologies to handle the database management. The most annoying disadvantage is the duplication of records. To overcome these several new ideas proposed and one of the most popular one is relational database management system proposed by E.F Codd. Today DB2, Oracle and SQL server are the most prominent commercial DBMS products based on relation model. However, several technologies are existing and are upgrading in software level, but so far lesser work have been identified appreciably at hardware level for the database management system effectiveness. This project proposes hardware based novel method to improve the reliability of the data being transferred and processed by any respective DBMS application software. Since concept is involved in hardware level it becomes an application independent system.

2. RELATED WORK

Parallel computing as space-multiplexed memories with many memory chips connected in parallel where architecture normally requires a large number of interconnects with potentially large routing delay and consumes massive area [1]. The other works depicts on server level throughput with focus on latency and bandwidth. This optimal data placement on Multicore system is focused on hardware level but it has been viewed to work on throughput of the large data by improving the affinity between the Kernel Space and Application Space and this does not cover up concurrency of the read and write operation [3]. The data distribution works emphasis on decisions making regarding the placement of data across a cluster of database servers [2].Workload as a Sequence is designed with a focus on sequential operation. Sequence exploitation technique is adopted by heuristic algorithm and suggested to work as "Query by day, update at night" this may not possible for global users due to conflict possible of user working at different time zone [5]

3. PROBLEM STATEMENT

Concurrency is not possible with the current system but priority technique is adopted which still may leads to file placement error occurrence. Today, DBMS technologies use different applications to query the database which has been downloaded from the remote servers. Irrespective of applications being used, appreciably no isolation is identified existing between newly arrived files from the server and the file which is being used or read by the CPU. In other words independent action between write and read operation is not found.

4. FUNCTIONAL DIAGRAM OF HARDWARE SET UP

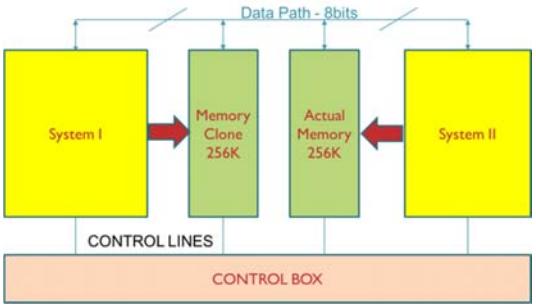

Figure-1. Functional diagram of hardware.

5. OPERATIONS

The memory module is defined to function as single memory logically but physically, it works as a separate memory as per given conditions below. CREW –

Conditions	WR1	OE1	WR2	OE2	LOAD	DIS	HLD	HLDA	ADDR1	ADDR2
S1 WR,S2 RD	0	1	1	0	1	1	0	0	1	1
S1 WR, S2 IDLE	0	1	1	1	1	1	0	0	1	X
S1 IDLE, S2 RD	1	1	1	0	1	1	0	0	X	1
S1 IDLE, M1 TO M2(DMA)	1	0	0	1	0	0	1	1	X	X

Figure-2. Truth table.

Concurrent Read and Exclusive Write working with priority can be avoided by this method [1]. Clock signal are generated from separate Systems 1 and 2. The read/write operations take place at clock rising edge. Each port also has its own address bus (ADDR), data in bus (DATA_IN) and data out bus (DATA_OUT). DATA_OUT pins are registered which gives the content of current memory location specified by ADDR ports only at rising clock edge. Each port has a port enable signal (OE) and write enable signal (W E). When OE of a port is not active, access at thisport is ignored, and its DATA_OUT become high Z (high impedance). Access to ports are only valid when their OE signals are active and write access must turn WE signals active low. DMA is used to monitor memory transaction

A. Block diagram

System 1 refers to the server and System 2 refers to User machine or destination machine. Two separate memories used in this model are each exclusively deployed for the respective CPUs. Memory - Memory Interaction takes place during the idle state of CPUs. Memory interactions are monitored by DMA (Control Box) as per logic provided in the truth table.

B. Truth table– description

The Notation 1 and 2 used on the pins refer to the System 1 and System respectively. All control signals are designed to work on active low conditions. For an instance S1 WR2, S2 RD refers the isolated operation between two CPUs. The second row refers to the System 1 writing in to memory 1. The third row explains the condition of System 2 read operation. And finally, the last row in the table defines the memory interaction. Refer Figure-2.

C. Circuit diagram

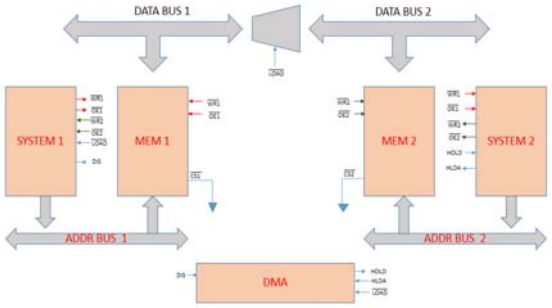

Figure-3. Circuit diagram.

The schematics shown above unique data bus for these two memory segments. Both System 1 and 2 are communicated with each other on their current status to ensure the memory interactions are effective as expected to operate. Based on the system status whether idle or active, DMA is given authorization of bus utilization.

6. MEMORY MODULE SIMULATION RESULT

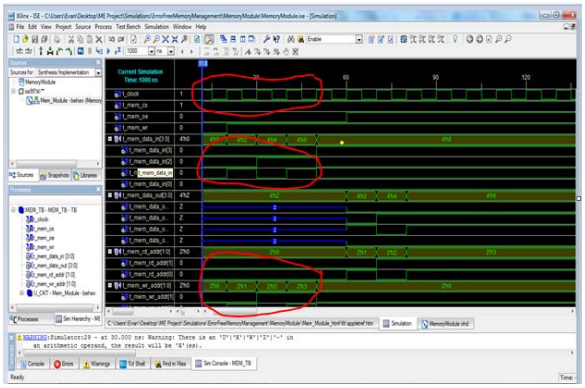

Figure-4. Memory module result.

The memory module and result depicted in Figure-4 shows the data write operation and data read operation are concurrent with respect to the time. The design of this proposal has been verified using a VHDL code and Xilinx 9.2i tool to simulate the result.

7. CONCLUSIONS

The problem of file placement error occurring due to read and write operation is extremely critical when

working with large database system and when handling big data. We are presenting a solution for this problem of file placement error in the memory by implementing the memory isolation. This finding is absolutely application independent as it is designed on elemental level. This ensures the process of updating is under the control of the individual user and at their convenience. Since all other works describes to improve performance in various aspect, this project's primary focus is to eradicate the file placement error due to lack of restriction on concurrency operation.

This project has been successfully tested on Xilinx 9.2i. This paper thus confirms the effective data management ensuring both reading and writing happens simultaneously and mutually exclusively.

REFERENCES

[1] Tran Due Linh, Tony de Souza-Daw, Thang Manh Hoang, Nguyen Tien Dzung. "Parallel Random Access Memory in a Shared Memory Architecture".

[2] Carlos Garcia-Alvarado, Venkatesh Raghavan Sivaramakrishnan Narayanan, Florian M. Waas. "Automatic Data Placement in MPP Databases".

[3] Stelios Mavridis1, Yannis Sfakianakis1, Anastasios Papagiannis, ManolisMarazakis and Angelos Bilas. "Achieving Scalability through Optimal Data Placement on Multicore Systems".

[4] C. Lameter, "Numa (non-uniform memory access): An overview," ACM Queue.

[5] Sanjay Agrawal, VivekNarasayya. "Automatic Physical Design Tuning: Workload as a Sequence".

[6] Nicolas Bruno, Surajit Chaudhuri. "An Online Approach to Physical Design Tuning".

[7] S. Agrawal, S. Chaudhuri, and V. Narasayya. "Automated selectionof materialized views and indexes in SQL databases. In: Proceedings of the Internatio.