ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 6, Issue 4, April 2018

A Protocol for Energy Efficient Dynamic Interthread Communication Using Cache Coherence

M.Sundarrajan, B Srinivasan*

School of Computing, SASTRA University, Thanjavur, Tamil Nadu, India

E-mail: sundarsoft92@gmail.com
E-mail: srinivasan@core.sastra.edu

Abstract: Threads are having different communications, which completely depend on memory hierarchy. Here the concept has different methods to find the communication pattern in shared memory. Then the method was used until the end of this paper is dynamic mapping, it is processing during the executions, which are also shown a great difference between the static and dynamic mapping. The proposed system had been used a cache coherence protocol for sending the message to all cores which have the same cache line. Here the mapping technique is done through the software and some hardware components were used to collect the information of cores. Jobs are given to the proposed model with the help of the Simics simulator. The jobs are entered or loaded into the framework with the help of the job node. The core node is responsible for processing the jobs and gives the desired output to the end user.

Keywords: Thread mapping; Communication matrix; Communication vector; Communication pattern; Coherence protocol; Invalidation message

I. INTRODUCTION

Thread mapping is used to divide the applications of the core into different task and mapping to the different cores and this could do by some scheduler. Mapping technique was completely used by software and then given to some algorithm for priority method. Which the technique is also used to reduce the migrations into the storage. Communications were collected by the hardware and software methods. There are different methods that are used.

First is communication aware thread mapping, it is done through the concept of collect the information from the core and then allocating the core. In the previous cases mapping would do by the scheduler and the map to the core in a different manner, but here the advantage is reducing the overhead of core.

Second is static and dynamic thread mapping, by this static method is do not perform during the execution because it is like 'stop and wait' method which it has to wait for a response and at the same time core do not know which core is invalidated during the execution. But dynamic mapping could do operation during the executions; this is an only method for mapping technique. Third is cache coherence protocol, which is used to send the invalidation message to the cache line and it is viewed by all the cores which have the same cache line through the ID ,cores are viewing the core is being invalidated. Through this all methods, the communication pattern was detected.

II. PROBLEM STATEMENT

2.1 Ideal Situation

With the modern technology, the systems have the dynamic mapping of the messages. But with this dynamic mapping, it introduces several challenges. This is due to the need of finding the suitable mapping and migrate the threads with a low overhead during the time of execution of various applications. Then the experts introduced a system or a mechanism to detect the communication pattern of the shared memory applications. This is done by using the shared memory applications by monitoring cache coherence protocols. But here, different cores use large amounts of memory even when the system is idle. It also increases the energy costs and communication overheads are very huge.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 6, Issue 4, April 2018

2.2 Problem

The existing systems have static communication and are not dynamic. Most of the existing methods map to the memory in a pre-defined manner. Only partial communication happens between nearby cores. This only increases the communication costs of the system. Cache solutions were introduced but writing into the cache itself caused a lot of overheads in the system.

2.3 Solution

The main solution of the problem is to design a model framework with dynamic thread mapping mechanism. It enhances the communication between the threads uniformly without any preference to the nearby cores as in the existing models. It also makes use of the cache effectively and the interconnection is very smooth when compared to the previous systems. This is done by the cache coherence protocols used in the parallel applications. It will be able to recognize the different communication patterns along the execution of the applications. The model also has the ability to migrate the threads of application with very different characteristics.

2.4 Solution-Benefits

The main advantages or the benefits associated with the proposed model are that has very negligible computational overheads. Implementation of the parallel applications has no time overheads associated with it. The thread execution is reduced by 15-20 percentages. Te model has also very low energy costs when compared to the existing systems.

2.5 System Design

All the system or the framework has their own design. The system design gives the entire description about all the parts of the model and their data flow. The data flows in the model as depicted in the system design. The data that enters the system goes through all the parts of the system in the same way as shown in the design. The system design gives the readers a diagrammatic representation of how the system works and what are the parts involve in the proposed model. Figure 1. Shows the various parallel applications that send request to the cores. The cache coherence is used to receive all the requests and execute it using various parameters. The various parameters include memory, energy, bandwidth and resources. With the help of these parameters the various outputs for the requests are given.

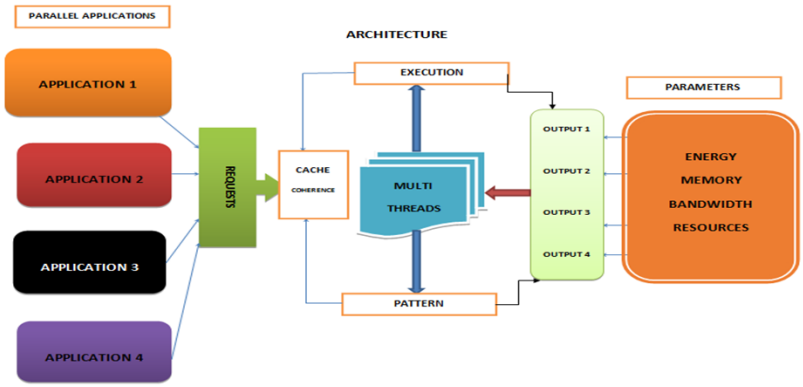

Figure 1: System architecture.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 6, Issue 4, April 2018

III. METHODOLOGIES AND APPROACHES

When a framework or a model is built it uses many methods and approaches in order give the desired output. The framework may use its own algorithms some policies or any other approaches in order to present the end user with the desired output. The various algorithms and the policies that are used designing this framework are discussed in the following section.

3.1 Algorithms

The entire framework that is designed follows some algorithms in order to get the desired output. The proposed framework also has some algorithms that follow. The algorithms give the instructions for the program of how they are to be implemented. The algorithms are used for programming which is made easier. The skeletal model of a program is an algorithm. The algorithms that are used in this framework are listed below.

3.1.1 Aging algorithm:

The aging algorithm is used in the model during the time when the communication matrix migrates all values of the core. It gives priority to that information first and then carries on with the next step. The aging algorithm makes use of a unique equation which shows that it has a previous value of aging algorithm and is added to the new value which is obtained from the communication vector. In the algorithm, we can also give factor value to the previous value. Then a good priority is set to that communication. This increases the priority value for the communication. The new and the final value can be avoided by more migrations into the communication matrix.

3.1.2 Mapping algorithm:

Mapping algorithm is the second algorithm that is used in this framework. The mapping algorithm is used to map the threads with the use of the communication filter. When the invalidation message is sent from the cache. When the communication filter gets the same communication pattern then it is mapped as they are matched. When the mapping is done then the process gets executed. The process follows entirely the graph which has vertex and edges within itself. The vertex in the graph depicts the thread and the edge in the graph depicts the amount of communications between the cores. Mapping is done in a step by step form. It will get applied with the mapping technique inside the core. There are some methods which are used in the model that plays a very vital role in designing and executing the system. The methods that are used in the models are described in the following section. All the methods that are used in this section have their own benefits for running and executing the system.

3.1.3 Communication filter:

It is used to find the communication pattern also says about whether it is a homogeneous or heterogeneous pattern. It also has some simple concept called filter which is used to divide the variance by average thereby we could get the H-factor, then we need a value of homogeneous and heterogeneous, here another value we could analyze is heterogeneous value is always greater than the homogeneous value is called H-threshold. Algorithm is used to compare the neighbour and then communicating. Perfectly communicating with a neighbour and then having the different values. It is not having the same amount of communications but it has a different amount of communications by communicating with a neighbour. It is reducing the overhead and too many migrations into the matrix, some threshold is completely adapt to the running applications and there are number of threads is communicating with the neighbour is called dynamic. After getting the information, it filters the pattern and then sends to the mapping algorithm. After the matching takes place the result will be used for the next level. The communication filter also has an equation which shows the matches of all the combinations and then gives a good example. The communication vector that is supporting will only have two threads which will have the information about the communication. It also supports the group of communications that happens in the system and also the neighbour communications that happen inside the system. It finally sends the information to the next level of the design.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 6, Issue 4, April 2018

3.1.4 Symmetric mapping filter:

It is used to delete unwanted migrations into the matrix and also after the mapping technique was progressed. No different methods were proposed to map two different threads. Then this is a final method which is to show the concept of mapping algorithm also has the concept for detecting the distance between the two neighbours. This is the main method while comparing all other because it has a way to delete migrations even it is used to clear the communication matrix if we want to go for next communications this is also a different concept proposal. Here the algorithm was used that is FLOYD-Warshall algorithm is used to get the distance the new and the previous value or mapping. Finally, it will migrate only for heterogeneous communication, not for homogeneous communication, if it homogeneous then it will not migrate the value in the matrix. This is a clear explanation about techniques.

3.1.5 Cache coherence protocol:

Which is used to send the invalidation message to cores has the same cache line; an invalidation message is used to detect the core which is being invalidated? Then we have LPCL which is used to detect the core id which is fixed with all the cores [1-4].

3.2 General Hardware Implementation

It has only three component namely communication vector, communication matrix, and c counters. All of them are perfectly done with the help of the hardware section. First, communication vector is used to store the information about the core and then it is used to get the ID of each core using cache coherence protocol. Communication pattern could be analyzed by software here the method which is simply used to detect and collect the information, it contains information with core id. Then communication is individual for the entire core which is a number of cores is equal to the number of communication vector [5]. Second is communication matrix, it is used to merge the total number of communication vector in the software layer then it can be viewed by the software, by the concept then finally it is sent to the algorithm to perform the detection mechanism, then finally algorithm will decide the whole things to perform good migrations and sharing. The third is C counter, it is also the same concept like communication vector, c is the total number of core used in this concept, storing the amount of communications of the local core including the id of the core. Then finally each core has a unique id to identify the core which is similar. By this, we can recognize the core by any situation during the executions. Cache coherence protocol will send all information about the core to communication vector and then finally it merged by using the communication matrix. Communication matrix is merged and send to mapping algorithm, then it is performing one by one process to detect the communications and the finally its detection the communications by using the cache coherence protocol. Here software which is used to map the threads [6-9].

IV. OUTPUT

The results of the design are shown with the help of two simulators. One is Proteus simulator and the other one used is the Simics simulator. Initially, the Proteus simulator is used to show the framework in a graphical representation. The entire core design is shown here. Then the Simics simulator is used to show how the jobs are scheduled and how it makes use of the benchmark programs that are given to it. The Simics simulator gives the main design of the framework and the same sequence happens in the entire framework.

4.1 Proteus Simulator

The Proteus simulator is the simulator that provides the design how the core is actually designed in the system and what are the major components of the system. The hardware that is present in the computer is depicted in the Proteus simulator. The wiring's and the execution flow of the data is shown in the simulator with the help of the simulation. The model of the framework before and after the execution is shown in the images that are presented in the following section. The simulation is shown in the VSM analyzer and the simulation log also depicts the form in which the simulation takes place inside the processor.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 6, Issue 4, April 2018

4.1.1 Before simulation:

The design shows the 8051 microcontroller that is used in the system. All the controllers are wired with each other and there is a digital analysis that shows the digital representation of the data that flows through the microcontroller as shown in Figure 2. The controllers are grounded and the voltage supply is given to the microcontroller. The panel shows the toolbar where all the operation commands are present such as a file, view tools and so on. The left panel shows the view of the microcontroller and other devices that can be used in the framework. The panel below the frame shows the buttons which trigger the simulation in the system. The image that is present in the next image gives the description where the simulation occurs. Simulation log and VSM analyzer are present which shows the actual simulation of the system using the microcontroller.

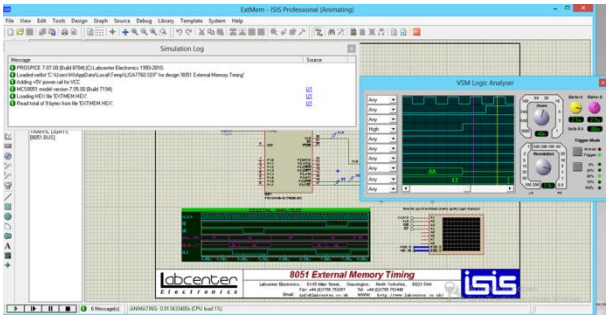

Figure 2: Proteus simulator.

4.1.2 After simulation:

The image below shows the simulation of the microcontroller 8051 which is present and connected with other devices. The simulation log and the VSM analyzer are also present which shows how the simulation occurs. The simulation log is selected from the source button that is present in the toolbar. The simulation log gives the sequential order in which the process occurs in the design. The simulation is very useful for knowing how actually the process occurs and what are the graphical representation of the process that is prepossessed in the simulator. The VSM analyzer depicts various parameters such as memory used, the cache used, bandwidth, network and all other parameters that can be measured by the simulator. The program is written in C language and is loaded into the simulator. According to the coding the simulator runs.

4.2 Simics Simulator

The Simics simulator is used to give the job to the framework designed. The framework is designed using the Java JDK tool which is used for the building any applications within itself. In the framework, the job is loaded into the design by clicking the Load button in the framework. Then the user needs to select any image from it and then give the job that needs to be performed.

4.2.1 Job node:

The job node in the framework loads any image from the system. An image is used as image processing benchmark is used in the following model. With this benchmark, a job is given to the framework. For instance, some jobs are given here such as wave, swirl and so on. When any job is selected the image processing occurs. Here before giving the job it is the performance of the system remains idle. The thread efficiency and the memory that are used are null as there is still no job given to the system.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 6, Issue 4, April 2018

4.2.2 Core Processor

The core processor is the processor that performs the thread mapping in the framework. The job that is given by the job node is executed by the core processor node. The job that is selected the is performed by the core and the output is returned to the job node itself. The job node is the node that does the thread mapping without the use of the cache. The working is given in the Figure 3. Here, before giving the job to the core the performance counters were idle as there were no job scheduled. But when the job is given and when they are processed then the thread efficiency and the memory automatically updates itself. Likewise when next job is given the thread efficiency again increases which states that each and every time a new job is given it increases the efficiency of the thread and the memory increases.

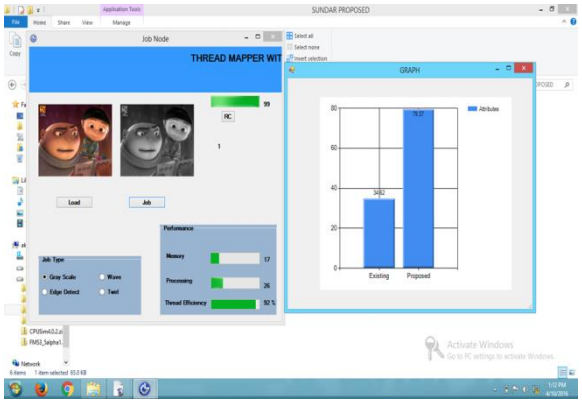

Figure 3: Output- proposed system.

V. CONCLUSION

Jobs are given to the proposed model with the help of the Simics simulator. The jobs are entered or loaded into the framework with the help of the job node. The core node is responsible for processing the jobs and gives the desired output to the end user. The coding in done with the help of Java programming and is loaded in the framework. The Java JDK tool is used in order to design the entire framework. The proposed model that is designed is very successful for the dynamic inter-thread communication. The system reduces the computational overheads significantly to a greater extent. The system improves the energy efficiency to a greater extent. It is suitable for all the types of architectures that are present. It utilizes the cache to the maximum.

VI. REFERENCES

1. CHM Eduardo, D Matthias, et al. Dynamic thread mapping of shared memory applications by exploiting cache coherence protocols. Journal of Parallel and Distributed Computing 2013; 74: 2215-2228.
2. S Lina, W Sonya, et al. Phase-guided scheduling on single-ISA heterogeneous multicore processors. IEEE Digital System Design 2011.
3. G Vishal, N Ripal, Analyzing performance asymmetric multicore processors for latency sensitive data center applications. International Conference on Power-Aware Computing and Systems 2010: 1-5.
4. ML Felipe, FC Henrique, et al. Parallel shared memory workloads performance on asymmetric multi-core architectures. IEEE International Conference on Parallel Distributed and Network-Based Processing 2010.
5. SC Juan, P Manuel, et al. A comprehensive scheduler for asymmetric multicore system. European Conference on Computer Systems 2010: 139-152
6. S Daniel, SC Juan, et al. HASS**:** a scheduler for heterogeneous multicore systems. Operating Systems Review 2009; 43:66-75.
7. F Alexandra, SC Juan, et al. Maximizing power efficiency with asymmetric multicore systems. Communications of the ACM 2009; 52: 48-57.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Vol. 6, Issue 4, April 2018

8. L Tong, B Dan, et al. Efficient operating system scheduling for performance-asymmetric multi-core architectures. IEEE conference on Supercomputing 2007.
9. K Rakesh, FI Keith, et al. Single-ISA heterogeneous multi-core architectures for multithreaded workload performance. IEEE International Symposium on Computer Architecture 2004.