THREE LEVELS EFFECTIVE MEMORY ACCESS OPTIMIZATION ADDRESSING HIGH LATENCY ISSUES IN MODERN MEMORY DEPENDENT SYSTEMS

Muhammad Yousaf Ali Khan1, Abid Saleem2, Asif Nawaz3, Nasru Minallah4, Rehan Ali Khan5, Muneeb Sadat6, Zeeshan Najam7, Sheeraz Ahmed8

1,2Department of Electrical Engineering, Gomal University, D.I. Khan, Pakistan

3Faculty of Engineering (ETS), Dubai Women College, HCT, Dubai, U.A.E

4National Center of Big Data and Cloud Computing (NCBC), UET Peshawar

5,6Deprt of Electrical Engg, Bannu University of Science & Tech, Bannu, Pakistan

7Deprtment of Electrical Engg, MNS, UET, Multan, Pakistan

8Department of Computer Science, Iqra National University, Peshawar

Corresponding Author: Dr. Sheeraz Ahmed
Email: sheeraz.ahmad@inu.edu.pk

Abstract

The modern digital systems especially those dealing with enormous data consumption application are facing a very complicated problem of high latency in these memory access application. Latency seems to be a major hurdle in the performance of modern memory dependent systems as it experiences delay in the processing. This high latency depends upon too many factors especially applications involving memory access operation. Out of these major factors one is of the binding and allocation application. Number of different approaches in the recent past has adopted to optimize the high latency in memory access application. Yet the modern embedded system faces high latency still due to enormous data transfer. In our approach we focus to optimize the latency of modern digital system by dividing the memory into groups. Following by activating, the fourth coming commands in advance in idle slots of different memory modules. The approach is called slag time management. In our algorithm effective distribution of memory into modules activating the later command in advance is followed by the advance dynamic buffers for saving the most frequently access arrays in it. The proposed technique of dividing the memory into modules utilizing the memory management idle slot management in use of advance of dynamic buffers has significantly approved the overall of latency of
modern digital system. The result at the end of different benchmark is shown that we achieved overall 9% optimization in memory access as compared to the presence of art technique. This concludes that the approach has significant effect on memory access and dynamic buffer use.

Keywords: Array binding and allocation, Dynamic random-access memory (DRAM); effective scheduling, empty slots management, memory latency, multi-core processors on chip (MPSoC).

I. Introduction

The digital systems in the recent years were dependent on the clock speed of the computer. The increase in the speed of the clock of the computer was actually the main performance hurdle, analyzed in recent past. But it has been noticed in recent years that research needs to be initiated in the power consumption and the memory access delay as well as in the heat dissipation of the system. Though in the recent past the computer manufactures has focus on the De-Morass law and as result of that the large scale of the integration of transistors on single chip has given arise to another concept of incorporating multi-processor on single chip. The different application become too complicated that created a problem of large computation of memory access hence this compel the manufacturer to shift from a single processor to multi processors on board. The use of multi process on board solved the problem as the memory access was divided among processor on board. This multi core processor is now the back bone of computer industry.

Today’s multiprocessor computers use both the cache memory on board and the off chip memory like DRAM of the system. The cache is used for the local accesses while the off-chip DRAM is used for global accesses application. Those applications that require an enormous data computation and memory accesses are very expensive in the form of latency of the system. We also need to store them in the off-chip memories as the on chip memories are too expensive and we cannot use that in a very big size in our computer. Now the problem of accessing these on chip memory and give rise to high latency application. As the data transfer involves very high memory computation so we need to develop algorithm to reduce the overall latency of the application. Different approaches have been used recent past for access off chip memories. Different high speed off chip memories are used now a days are dual data rate DRAM first generation, dual data DRAM 2nd generation and third generation of the dual rate DRAM and dual data rate static DRAM. The latest one is used in for DDRAM four off chip memory storage. The latest version DDR four SDRAM has the maximum bandwidth as it uses both the rise and fall of the clock for the execution of the command. Hence this performance evaluation is achieved from the retain clock speed of the computer and it is important.

In the advance and modern systems, the major constraint in the performance of the digital system is affected by latency during memory access of the system by off chip memories. Once we access the off chip memories in a complicated application that are having huge data transfer are creating problem for the system. This enormous delay during radio or image processing or another complicated memory access
application is producing latency for the system. This limits the performance of modern digital system. The high latency of the system is producing another problem of power consumption hence we need to develop algorithm for video, image processing an application for high data transfer as to optimize the overall of the problem of enormous in this multi process system.

The dynamic random access memories of the present era adopted to use double data rate technique for executing application involving high data transfer. This technique uses first technique to transfer to enormous data using dual data rate. The benefit of technique is a single clock cycle data transfer (twice). During the execution of the read slot write command the data is transferred at internal core of the DRAM and at the corresponding of input slot and output fall data is received.

II. Background

Latency in accessing the memory from off chip dynamic memory in advance digital systems is the major hurdle in improves performance of modern computers. The applications that are having large number of memory access and data computations are expensive in latency as well as power consumption. Example of such applications may be videos and images. While accessing these applications the systems experiences high latency and also consume enormous power, we need to reduce the latency and power of the system. To solve the problem in these accesses we need to develop technique for multi core processor that are optimized in advance enough to solve the problem of latency of and power dissipation.

The modern DRAM’s and SRAMN’s are designed in such a way that multi processors can access the off chip memories in parallel the concept of architecture of these RAM’s are presented in [VI, X]. In our thesis we would like to present the recent related work on different methods of memory access [XV, XIII]. These techniques of accessing the off chip DRAM’s are not competent enough to resolve the issues when huge data transfer is involved in application. The model presented in [XII] to reduce the memory access latency is accurate delay modeling it works up till some level to reduce the latency in these exhaustive memory loss. Along with these many more methods are develop adopted in recent years for optimization latency and reduction of power and latency. Recent technique also focuses the increase the Band Width of the system during memory access. The page mode tech and read mode write technique are the most common and most adopted technique to increase the Band Width and optimize latency technique. Page mode technique particular are presented in [VII] and detail mode technique are presented in [VIII]. The techniques of page mode and read mode are effective to optimize the latency of the system and controlling the high data transfer in single row access clock of the memories by utilizing bidirectional clock of the system. Another technique reported in [IX] for optimization of memory accesses is also very important. The technique presented in [IX] transformation in loop by transforming the smaller loop in bigger by re-writing in loop manner. Another important and interesting technique is of loop feature [XI]. It depends on the fusing the number of loops in the bigger loop. Technique of loop morphing is also the modern technique of optimizing the latency as discussed in [IV]. Meanwhile in connection with existing technique a tech reported in [II] using loop
alignment that is also used for the reduction of power and latency optimization in high data transfer application. The page mode algorithm has limited the scopes of loops transformation technique as the whole memory module is remain active when it is assessed and one can read it entire data without entering the further command. Despite the development technique of too much memory access command for optimization by array binding memory allocation as reported in [XIV]. The researchers are also developing a technique by using dual port memory in [V] though it is too expensive to adopt. The tech is based on paralyzing of memory accessing by partition memory into module. The simple delay modeling example is shown in figure 1.

A. The Technique of Loop Fusion

![Simple delay modeling Example](image)

The important task in continuation of the designer has also considered the advantage of hiding the slow memories behind fast memory latency. This will help to decrease the overall latency but on other side we need to use more than one memory on board for achieving this optimization memory access latency.

For achieving the better performance in the modern digital systems the designers have distributed the data in different memory modules is to maximize the parallel processing for accessing the off-chip memories. Due to dividing the memories into small module though we have to achieve the parallel processing for accessing the data through these memories but on other side we have increased the memory size modules in the computer motherboard. For optimizing the overall latency, we here may introduce another technique of dividing the repeated data into cluster with in the memories. The bottleneck in these techniques processing these parallel data accesses in optimize manner. Hence the processes must be also divided into the number of processes within a single motherboard as to actively handle and manage the active memory process and at the same time. This multiple process on board techniques will handle and will keep the balance between the multiple accesses of off-chip dynamic memories and manage of this request.
The designer has though focused on using only on single basic clock cycle for the utilization of the clock in the present techniques this limits the Band Width of the memory accessed. We need to adopt the new techniques of using the borders of clock for maximize the memory band Width and also elevating the performance of modern digital system. The use of the borer of the clock is also reducing the energy consumption of system. Important transformation techniques are discussed in [IV]. This technique is fusing the smaller loop into the bigger loops. By combining these two loops into a single loop as the smaller loop is defuse into the bigger loops hides the latency of the smaller loops into bigger loop and rather the addition of two access time, the time of the smaller loop is reduced to access within the time of when we are accessing bigger loop.

The result of these fusion techniques is although we achieve the reduction of overall latency but the processing done in parallel in this required multiprocessor on board. This is done on the basic of that we have enough available slots normally we the bigger loop execution. Now we need to identify empty slots in the bigger loop. Initiate the execution of smaller loops in those ideal slots this diffusion is taken into the system depending upon the nature of data assignment access at the moment. A proper appropriate decision need to be taken at the in order to address constrain the energy, latency at the same time.

B. Technique of Loop Morphing

The Compilers are efficient enough to determine the memory access initiated in parallel the single time. The maximizing of parallel processing’s is possible at the modern day compiler execution are as thee compiler are capable to manage these accesses in a single time through multiple processor on board but it depends on load and stabbing capabilities of single memory module units are present on board.

The modern system with the modern compilers is competent enough or capable enough to schedule the parallel processing for accessing the off-chip dynamic memories. The memory request initiated by these compilers that involved used the huge data transfer in the transformer from the single memory unit is a very complex procedure. The problem arises when these initiated memory access data at a request operated in a run time on single processor installed at the motherboard computer. This problem involved multi-process on board. These stalls can be divided into multi memories and multi-processor on board but this lead to extra energy cost, and money cost in the modern system.

Another important technique may be used instead of utilizing single port memories but distribute these memories carefully with better memory access optimization technique initiated in curdle method.

By introducing the loop transformation techniques that fused the number of loops in the single loop by utilizing the empty slots available in the bigger loop in basic clock is an important solution to the problem. These empty slots can utilize the gap available in the available efficiency of the enhancement of modern digital system. The loop fusion techniques are actually increasing the local accesses and decreing the life time of the arrays initiated processor or memory accesses. The criteria of the
loop fusion are based on the combination of the appropriate loop for the increasing the efficiency be decreasing the overall latency delay of overall the memory access delay. It’s worth to note here that loops combined here are the only nested loops that are confirmable loops header other loops cannot be fused here. This constrains of combining of loops in the nested loop limits the applicability of the loop technique in broader prospects. Here another method is used that is loop morphing technique that is reduction of the latency of the system [XI].

This technique is based on fusing unconformable loop into bigger loop that is edit advantage from loop fusion. The strip mining as well as loop morphing is unconformable. We apply attractive loop morphing for the memory band width increased technique it is approved significantly for reduction of overall latency. Though loop morphing is effective technique for parallel memory access in the basic clock of the system.

III. Existing Delay Model for DRAM’s

When the modern systems are accessing the data from the off-chip dynamic Random access memories it takes too long in clock cycles in access the memory. This increase the latency of off-chip memory degrades the performance of modern system. For accurately accessing these off-chip dynamic memories for different applications the accessing should be model accurately as to neglect and reduce the conflicts of accessing of single memory module for different application for at a single time. By accurate modeling we will reduce the possibilities of conflicts of accessing these memories at as single glance by stream lines by accessing of these modules performance of the system will be enhanced and latency will reduce at minimum level.

A. Optimization of Accessing through Memory Delay Modeling

The modern computer that runs Multi-Processor on system facing the problem of high latency in execution its operation that involves in it accessing the off-chip dynamic memories, we need to reduce this access time and limit the latency of the application involving high memory transfers. The problem arises here are once we accessing the memories that having too many arrays we need to schedule these arrays and allocate these arrays appropriately memory modules as to minimize the time of accessing these arrays effectively and in parallel approach. In the modern dynamic random access memories like in the case of DDR3 SDRAM and others designing of the memory modules and allocation is in important aspects hence we use the accurate delay model in these modern DRAMS for better accessing off the off-chip memories. Now a main focus will be exploiting the memory access of operation in embedded system and scheduling them effectively. Figures 1 and 2 shows the result that actually proved the fact that scheduling the memory operation can be effectively handle by reducing the overall latency in the way that we assign and activate the commands in accurate way like proposed in accurate delay model [XII].

In the examples we have actually three arrays [X], [Y], [Z], and these arrays need to be accessed particular fashion from different memory modes. The result of addition should be such that
X[0] = Y[0] + Z[0]

Now we have different commands to be activated for Read and Write cycles that shows the complete sequence of accessing of these off-chip dynamic memories array.

First command is;

Activation1 Read1: read y[0]
Activation2 Read2: read z[0]
Activation3 Write: write x[0]

![Image](image.png)

Fig. 2: Cyclic delay Modeling Example (CADM) [XII]

IV. Three Dimensional Memory access Optimization

The most important aspects of embedded system are to utilize the chip area efficiently and reduce the overall area of the memory. The modern complexity of designing embedded system involves efficiently access the off-chip dynamic memory in its integration to the system appropriately with maximum speed and minimum chip area. This object is achieved by addressing many factors that includes scheduling of memory accesses, allocation of the modules of memory, allocation of memory to modules that can be achieved by efficient binding of memory efficient memory variables to the memory and store them appropriate memory access algorithm. These different factors need to be activated at a same time for efficiently result achieved by integrating these three dimensional effort we will globally reduce the latency of the system, these three approaches are efficient memory access memory access algorithm, appropriate allocation of memory modules and use of advance dynamic buffer at the end. The recently adopted the page mode technique for efficient access off-chip dynamic memories used in the advance system but it has some limitation. Different other techniques like allocation of memories assigning arrays to these memories efficient binding in scheduling techniques although presented by different researcher but in attempt is required to schedule these all in a single approach that has dynamic methods integrated in a single approach and that is three dimensional memory access approach is presented in this section.
A. Using Multi-Way Partition Technique for Efficient Memory Allocation

Typically, in application that involves transferring video or image to be accessed from off-chip dynamic memories the arrays become too large and it takes the latency too much to handle these arrays by incorporating them on to the RAM on to the system from off-chip dynamic RAM. These applications that are memory too much intensive during accesses the off-chip dynamic memories factor is the scheduling of the arrays access for data acquisition. By frequently accessing these large arrays the memory access latency becomes larger in the global memory realization increased. This problem can be resolved by identifying appropriate code section that lock at least to the high number of latency.

This problem needs a solution and that lays in identification of that clock in the source code that least to higher latency. These arrays need to be arranged efficiently for reduction of the overall latency. By efficiently arranging these arrays for accessing off-chip dynamic memories the overall optimization achieved. We proposed an approach that we will group the arrays together depending upon the number access made to the off-chip dynamic memories access in parallel. Now on the basis of information accessing the arrays in parallel by accessing the off-chip dynamic memories of the system the motivational example is presented in figure 3. In example of clean cubic function is presented here and in this example number of arrays are 5 in block of the code. These arrays of clock of the code are stored in DRAM of the system. Now one can analyze that accessing these memories in sequentially the memory will lead to high latency. Now if these arrays are locating in single memory modules that will restrict the parallel access to the memory and will enhance the latency now for the efficiency in the latency we would focused to increase the parallel accessed method to the off-chip memories and this can be done by allocating distinct memory modules efficiently. This will though create the problem of increasing the parallel chip area but there is always a trade between the two entities here the decreasing the latency we have to compromise on the chip area of the memory module.

```c
for (j=1;j<wz1;++j)
\
//
  Opp1: a[j]=x[j+1]-x[j];
  Opp2: b[j]=a[j-1]+a[j];
  Opp3: c[j+1]=(y[j+1])-y[j])/a[j];
  Opp4: c[j]=c[j+1]-a[j];
  Opp5: a[j]=a[j]*3;
}
```

Fig. 3: Spline Cubic Function Example
Figure 4 shows the overall latency of the example is 60 clock cycles. This 60 clock cycles latency is achieved when we allocate the four arrays [a], [b], [c], [e] to the memory module M0, memory module M2 and allocate one of the array [d] to the memory module M0. Here it is mentioned that clock cycles consume normal Write are 8 basic clock cycles of the system and normal Read consumes 5 basic clock of the system. Contrary to this the clock cycle consumed by page Write three basic clock cycles of the system and page Read consumes two basic clock cycles of the system as reported in [XI]. The table 1 represent that total area consumed by different memory cycles information is taken by module library of modern digital system.

Another data presented in table II gives the details of access information of digital system during a single alteration of the system about these arrays about these arrays that are reported in figure 3. Now one can see in the table II that the arrays are accessed more number of times are presented in second column and the first column we have shown the overall accesses of approaching these arrays by the system.

![fig:4](image-url)

Fig. 4: Memory binding and allocation example [III]
Table 1: Access Information Of Different Arrays

| Array Name | Array Elements | Number of Read Elements | Number of Write Elements | Total Number of Access |
|------------|----------------|-------------------------|-------------------------|------------------------|
| A[]        | A[j]           | 4                       | 2                       | 7                      |
|            | A[j-1]         | 1                       | 0                       |                        |
| B[]        | B[j]           | 1                       | 0                       | 2                      |
|            | B[j+1]         | 1                       | 0                       |                        |
| C[]        | C[j]           | 0                       | 1                       | 1                      |
| D[]        | D[j]           | 1                       | 0                       | 2                      |
|            | D[j+1]         | 1                       | 0                       |                        |
| E[]        | E[j]           | 1                       | 1                       | 4                      |
|            | E[j+1]         | 1                       | 1                       |                        |

Table 2: Grouping Of The Arrays

| Array groups | Number of instructions | Array groups | Number of instructions |
|--------------|------------------------|--------------|------------------------|
| {D, C}       | 2                      | {D, Y}       | 1                      |
| {D, X}       | 1                      | {C, Y}       | 1                      |
| {D, B}       | 1                      | {D, C, Y}    | 1                      |

Fig. 5: Min-cut algorithm conflict graph [V]

Copyright reserved © J. Mech. Cont. & Math. Sci.
Muhammad Yousaf Ali Khan et al
Once all the information is acquired after the complete procedure of accessing the arrays we now have to decide to store initially the arrays accessing the minimum in number and then the one accessed after this and so on access going in descending order. Now depending upon the number of accesses the arrays are accessed the order would be on the basis of information is shown in figure 4 \{A \[ \], E \[ \], B \[ \], D \[ \], C \[ \]\}. On the basis of this information now we would develop the groups so that these arrays are accessed in parallel in more efficient optimization results. On the basis of this calculation we have to develop the table III. The table III will provide the information of two different types. The grouping of arrays and the number of times these are accessed at parallel.

The conflict graph and the regular lines show the connection between the arrays that are accessed in parallel during the application. The regular lines value on the top are present that how many times these arrays are accessed in parallel during the execution of abstraction application. Now the basis on this information we should place the
arrays A and E in different memory modules as they are with conflict to each other. Now on the basis of information we can introduce there is more conflict between B, C, D arrays hence they can be put in a single memory module and they can be accessed without having conflict between them. Too many other conflicts can also be shown can also be viewed in figure 4 which need to be resolved. Now the techniques of mingle partitioning is adopted recently but we need to develop a better technique from min-cut algorithm to maximize the resolution of the conflicts between accessing arrays [III]. The technique which we proposed is max-cut algorithm of min-cut algorithm this technique is develop from min-cut algorithm of proposed technique we actually take inverse of min-cut algorithm and then adjust the values of regular line basis on information [III]. Now in the very first step, we invert the conflict graph values and then add the new regular lines the arrays that don’t have conflict of previously and in the second stage remove the existing line having the highest conflict values. The next important step is to adjust the values of the regular lines in the max-cut algorithm. By partitioning the memory modules into two groups we can resolve the conflict between array A and B as shown in the figure 6. Though it has been mention in the figure 7 that the conflicts are resolved but the added burden is the introduction of third memory module.

Now the arrays B, C can now is stored other memory modules. As an output of this technique the designer has the constrain of selecting more number of memory modules though this is the back and this is the disadvantage of max-cut algorithm but we can achieve too much parallelism in accessing this off-chip dynamic memories with the help of max-cut algorithm hence eventually it reduces the latency of the system. Now we select the problem presented in figure 4. We used the multi way partitioning technique to resolve the example and we have presented figure 5. It is better than the min-cut algorithm and it reduces the overall latency of the clock cycle from 46 clock cycle to 44 clock cycles as presented in the figure 7. The max-cut algorithm will maximize the parallel access of the off-chip dynamic memories and will reduce the overall latency of the system.

B. Managing the Latency through the Idle/Select Time Utilization

The second approach to minimize the overall latency of accessing of off-chip dynamic memories is of utilizing the stacks or idle slots available in the memory modules commands activation, we need to exploit those idle slots of the different memory modules and activate the later command in advance to optimize the overall performance of accessing off-chip dynamic memories complex application scenario like video or image processing. By analyzing the different memory allocation algorithms, we notice that the arrays are assigned to the memories that maximize the access in parallel from off-chip dynamic memories but still some slots in memory module idle and no action is performed during this idle slots. We need to exploit those idle or slag slots available in the memory modules and utilize these slots by activating the commands coming later in the sequence in advance as to reduce the overall clock cycles consumed by those applications of accessing the off-chip dynamic memories the process is shown in the figure 8 in details.

Copyright reserved © J. Mech. Cont.& Math. Sci.
Muhammad Yousaf Ali Khan et al
C. Utilization of Advance Dynamic Buffer Third Approach

The approach of partitioning the graphs in multi-way helps us are reduce conflicts almost nil. This is almost achieved by the arrays to the appropriate way memory module as to avoid parallel conflicts accessing of these arrays. Now in the last stage we need to use the advance dynamic buffers for the further reduction of memory access from off-chip dynamic memories. This will use optimize the memory accessing to the maximum extent. Now a day the scratch pad memories are used in modern digital system is on chip S-RAMS based these are due to two different reasons. The incorporation of techniques in advance dynamic buffer will solve the clock cycles in addition to the already in multi-way partitioning cycling accurate delay model. Cache using them cache off-chip memories will help us placing the array in better way for the enhancement of the performance of the system for the reduction of the overall latency of the application.

![Diagram]

**Fig. 8**: Identification of idle slots
**Table 3: Summarizing result of benchmarks**

| Benchmarks | Total number of arrays | Total no. of conflicts | Applying min-cut Partitioning | Applying our approach | Conflicts resolved (in percentage) |
|------------|------------------------|------------------------|-------------------------------|-----------------------|-----------------------------------|
|            |                        |                        | Conflict resolved | Cost | Conflict resolved | Cost |
| IBM01      | 4                      | 3                      | 1                      | 2    | 3                  | 6    | 100%                  |
| IBM02      | 6                      | 7                      | 1                      | 2    | 6                  | 15   | 85%                   |
| IBM03      | 8                      | 14                     | 4                      | 9    | 12                 | 22   | 85%                   |
| IBM15      | 15                     | 22                     | 8                      | 14   | 16                 | 28   | 80%                   |

**Table 4: Access information table**

| Benchmarks | Total number of arrays | Total Cycles in memory access | Current page-mode technique | Applying our approach of Idle time management |
|------------|------------------------|-------------------------------|----------------------------|-----------------------------------------------|
|            |                        |                               | Cycle consume | Efficiency | Cycles consume | Efficiency |
| IBM01      | 5                      | 44                            | 33            | 25%       | 21             | 36%       |
| IBM02      | 8                      | 218                           | 190           | 12%       | 172            | 9%        |
| IBM03      | 15                     | 2065                          | 1808          | 12%       | 1595           | 12%       |
| IBM15      | 18                     | 3415                          | 3180          | 9%        | 2845           | 7%        |
| Average    |                        |                               |               |           |                |           |

Clock Cycles consumed in first iteration = 42

**Fig. 9: Utilizing Idle slots**

**Copyright reserved © J. Mech. Cont. & Math. Sci.**

Muhammad Yousaf Ali Khan et al
V. Results of the Experiment

Normally we have two different ways using the scratch pad memories either by using the dynamic version or static version of the memory. We have analyzed the static or dynamic memories scratch pad memories bases on the state application we are dealing with. We have examined the application through the arrays access pattern. In the application of static memories, we actually decide about the placement of the data of the execution run time while in the case of advance dynamic buffers allocate the data as per real run time scenario. Now we are selecting the advance dynamic buffers in our thesis as this will help us execution of application in the run time while accessing the data in real time. The advance dynamic buffers using the scratch pad memories on-chip memories effectively and this will enhance the capacity of the system as the most frequent arrays access are already present on the on-chip dynamic memories and we need to access the off-chip dynamic access memories. As example we are selecting the code of in its presented in figure 8. The example having five arrays and this information available in table III. Out of these arrays the most frequent one arrays are A, B, D, and E as mention in the table I. Now we need to store only these four memories for arrays in the scratch on-pad memory that is considered to be buffered and we are not leaving to store the other array should not frequently have.

Fig. 10: Experimental Result Graph before Buffer

Fig. 11: Experimental Result Graph after Buffer

Copyright reserved © J. Mech. Cont. & Math. Sci.
Muhammad Yousaf Ali Khan et al
accessed. The remaining arrays will be stored in off-chip dynamic memory accessing these arrays will not affect the performance of the system. The figure 10 shows that advance dynamic buffer reduces to significantly to the 21 (twenty-one) clock cycles of the system.

We have decided to use the IBM benchmark for the evolution of our results and experiments the different IBM benchmarks are used for the comparison of the result. Now the results shown in table IV are of the results of benchmark used for comparison of resolving conflicts of proposed the approach and mingle algorithm. The research is focus on the result that resolve almost more than 80% (Eighty Percent) conflicts as compared to the old mingle algorithm but this is achieved through additional cost the incorporated system. We can even further optimize this scenario by incorporating more memory module into system. Now we should go for maximizing the number of module until all of conflicts are resolved. Though this will increase the chip area of memory module but the efficiency of reducing the latency of enhanced would be significantly. The integration of all three different approaches of using cyclic accurate delay models partitioning the memory into effective module and using the advance dynamic memory buffers collectively gives a better result in reducing the overall latency in the advance digital system the results presented in figure 13 which clarifies of using these three different approaches.

![Fig. 13: Experimental Results](image)

VI. Conclusion

The research is based on the need of exploring ways for minimizing the overall latency in accessing off-chip dynamic memories in modern digital systems. We have adopted and integrating approach of using cyclic accurate delay models along with multi-way partitioning memory module in the final state using advance dynamic buffer to achieve the goal of reducing latency. This triple technique approach is significant in reducing the overall latency of the modern digital system. Even more research can be initiated in this scratch on pad dynamic memories.
integrating with the advance dynamic buffers and effective memory partitioning so as to achieve the reduction in latency in modern digital system significantly. Though in this research a significant production of almost 9% (Nine Percent) is achieved as compare to the previously page mode technique adopted in the recent past.

References

I. David Tawei Wang, “Modern DRAM Memory Systems: Performance Analysis and Scheduling Algorithm”, University of Maryland libraries, 2005.

II. Fraboulet, G. Huard, A. Mignotte, “Loop Alignment for Memory Access Optimization”, 12th International Symposium on System Synthesis, 1999.

III. H. Shin, C. Kim, “A Simple Yet Effective Technique for Partitioning”, IEEE Transaction on Very Large Integration (VLSI) System, pp. 380-386, 1993.

IV. J. I. Gomez, P. Marchal, S. Verdoorlaege, L. Pinuel, F. Catthoor, “Optimizing the Memory Bandwidth with Loop Morphing”, 15th IEEE International Conference on Application-Specific Systems, Architectures and Processors (ASAP04), 2004.

V. N. Kim, R. Peng, “A memory Allocation and Assignment Method Using Multi-Way Partitioning”, IEEE International SoC Conference, 2004.

VI. Prince, Betty, “High Performance Memories: New Architecture DRAMs and SRAMs — Evolution and Function”, 1st edition, 1996.

VII. P. R. Panda, N. D. Dutt and A. Nicolau, “Incorporating DRAM Access Modes into High-Level Synthesis”, IEEE Transaction on Computer-Aided Design, Vol. 17, pp. 96-106, 1998.

VIII. P. R. Panda, N. D. Dutt, A. Nicolau, “Exploiting Off-Chip Memory Access Modes in High-Level Synthesis”, International Conference on Computer-Aided Design (ICCAD ’97), 1997.

IX. P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkarni, A. Vanderappelle, and P. G. Kjeldsberg, “Data and Memory Optimization Techniques for Embedded Systems”, ACM Transaction Design Automation Electron System, Vol. 6, no. 2, pp. 149-206, 2001.

X. P. R. Panda, “Memory Bank Customization and Assignment in Behavioral Synthesis”, ICCAD, 1999.

XI. P. Marchal, J. I. Gomez, F. Catthoor, “Optimizing the Memory Bandwidth with Loop Fusion”, IEEE International Conference on Hardware/Software Codesign and System Synthesis (CODES + ISSS’04), 2004.
XII. S. Daud, H. shin, “Cycle Accurate Memory Delay Modeling for Off-Chip DRAMs”, System on Chip Conference, 2009.

XIII. S. J. E. Wilton and N. P. Jouppi, “CACTI: An Enhanced Cache Access and Cycle Time Model,” IEEE Journal of Solid state circuits, Vol. 31, pp. 667-688, 1996.

XIV. T. Kim, J. Kim, “Integration of Code Scheduling, Memory Allocation, and Array Binding for Memory Access Optimization”, IEEE Transaction on Computer Aided Design of Integrated circuits and systems, vol. 26, no. 1, pp. 142-151, 2007.

XV. T. Wada, S. Rajan, and S. A. Przbylski, “An Analytical Access Time Model for On-Chip Cache Memories”, IEEE Journal of Solid State circuits, Vol. 27, pp. 1147-1156, 1992.