ISSN No. 0976-5697

Volume 4, No. 3, March 2013 (Special Issue)

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Optimal Value of Superscalar Factor in Superscalar Pipelined Processor Architecture

Suresh Kumar Jha
M. Tech. Scholar
Department of Computer Science & Engineering,
Jodhpur Institute of Engineering & Technology
Jodhpur, India
suresh.jha84@gmail.com

Deepak Jangid
M.E. Scholar
Department of Computer Science & Engineering,
MBM Engineering College, JNV University
Jodhpur, India
deepak_it_mbm@yahoo.co.in

Rajesh Purohit
Associate Professor
3Department of Computer Science & Engineering, MBM Engineering College, JNV University
Jodhpur, India
rajeshpurohitmbm@yahoo.com

Abstract: The performance of superscalar processor is affected by various parameters like superscalar factor, reservation station architecture, size of reservation station, number of different functional units etc. The superscalar factor is the primary factor which increases the IPS (instructions per cycle). Superscalar architecture supports the instruction level parallelism by issuing more than one instruction per clock cycle i.e. increases the IPC. Increasing superscalar factor results in higher IPC. This paper intent to find the optimal value of superscalar factor to which significant improvement in IPC can be achieved. The simulation performed using PSATSim v2.1 simulator with five different SPEC benchmark programs, considering both distributed and centralized reservation station architectures separately.

Keywords: Superscalar, Distributed Reservation Station, Centralized Reservation Station, IPC, Execution Time, PSATSim/EdSATSim v2.1

I. INTRODAUCTION

A superscalar processor is one in which multiple independent instruction pipelines are used. Each pipeline consists of multiple stages, so that each pipeline can handle multiple instructions at a time. Multiple pipelines introduce a new level of parallelism, enabling multiple instructions to be processed at a time as shown in figure1.
A superscalar processor exploits what is known as instruction level parallelism, which refers to the degree to which the instruction of a program can be executed in parallel.
A superscalar processor typically fetches multiple instructions at a time and then attempts to find nearby instruction that are independent of one another and can therefore be executed in parallel. If the input to one instruction depends on the output of preceding instruction, then the latter instruction cannot complete execution at the same time or before the former instruction. Once such dependencies have been identified, the processor may issue and complete instruction in an order that differs from that of the original machine code. The processor may eliminate some unnecessary dependencies by the use of additional registers and the renaming of register reference in the original code.
The superscalar processor support instruction level parallelism [3], means more than one instruction may be executed in single clock cycle which improves the performance by executing multiple instructions in parallel. The performance of superscalar processor depends on various parameters e.g. superscalar factor, size of reservation station, reservation station architecture (centralized, distributed or hybrid), number of execution units, and memory architecture.

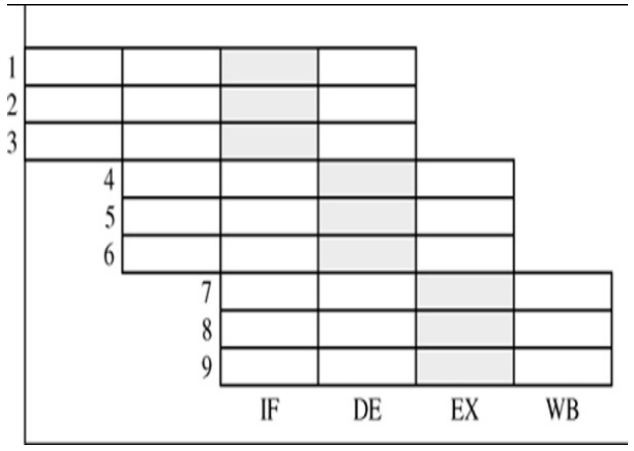

Figure1: Superscalar Pipeline

In this paper optimal value of superscalar factor at centralized and distributed reservation station architecture is computed through experimental approach using the PSATSim v2.1 simulator [1]. The distributed and

CONFERENCE PAPER
II International Conference on
"Advance Computing and Creating Entrepreneurs (ACCE2013)"
On 19-20 Feb 2013
Organized by
2nd SIG-WNs, Div IV & Udaipur Chapter , CSI , IEEE Computer Society Chapter India Council ,
IEEE Student Chapter Geetanjali Institute of Technical Studies, Udaipur, Rajasthan, India

centralized reservation station architecture were simulated for different value of superscalar factor and these two architecture were analyzed on the basis of IPC.

II. SUPERSCALAR PIPELINED PROCESSOR ARCHITECTURE

Superscalar processors have multiple functional units which provide greater concurrent processing of multiple instructions and higher instruction execution throughput. It has the ability to execute instruction in an order different from that specified by the original program. Instructions are issued from a sequential instruction stream. The central processing unit hardware dynamically checks for data dependencies between instructions at run time. The CPU accepts multiple instructions per clock cycle.

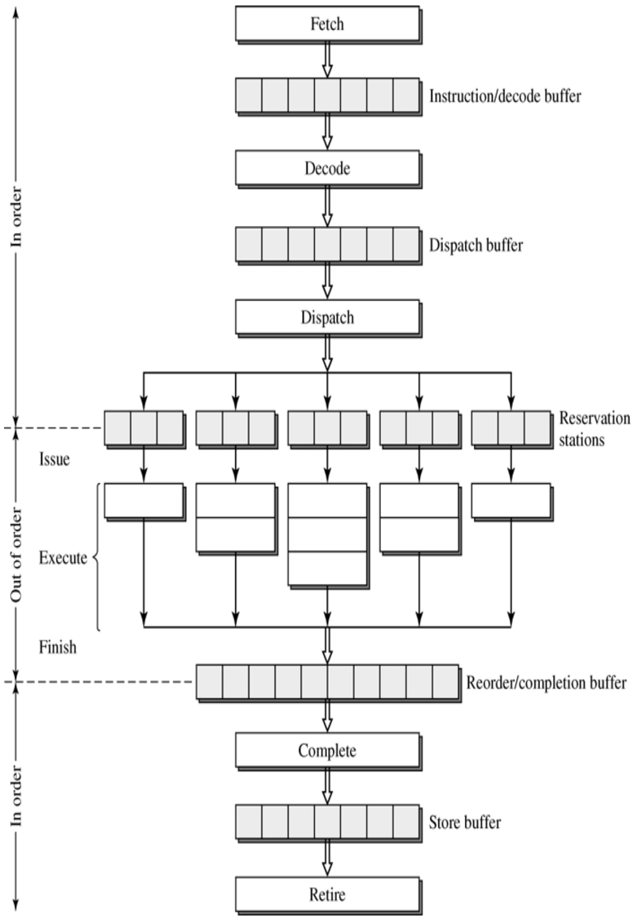

Figure 2: Superscalar Pipielined Processor Architecture

PSATSim simulator is based on design of superscalar processor shown in figure 2 proposed by shen,Lipasti [1][2]. In this architecture from *fetch* to *dispatch* stage, instruction is in-order and after the *dispatch* stage instruction is stored in *Reservation Station*.

Due to dependence [4] among the instructions, some instructions must wait for operand in *Reservation Station*. Instructions for which operands are ready will go to their correspondence *Execution Unit*. After that each executed instruction, which is in out of order, will go to in *Reorder Buffer*.

Two model architectures, were used for simulation purpose; one is centralized reservation station architecture and other is distributed reservation station architecture. To observe the effect of architecture of reservation station architectures, The other parameters are common to both models. These models were simulated on five different SPEC benchmark programs[8] .

III. ROLE OF DIFFERENT PARAMETERS IN SUPERSCALAR PIPELINED ARCHITECHTURE

Various parameters effecting performance of superscalar pipelined architectures are discussed below.

Superscalar Factor: The number of instructions that can be fetched and/or completed in one clock cycle. This value determines the width (instructions) of the fetch, decode, issue, and commit stages in the processor

Reservation Station Architecture: Every execution unit required at least one reservation station to store the source data from completing instructions. With more than one reservation station, instructions can begin execution as soon as its source operands are available even if a prior instruction has not started. Reservation stations hold the instructions that are waiting to begin execution or are currently executing. Branches and stores do not need any renaming entries since they do not change the value of any architectural registers. If instruction has all the source operands it requires, it's either executing or is ready to begin execution. Instructions remain in the reservation station until execution is complete. An instruction can begin execution when all of its source operands are available. However, only one instruction can begin execution, in any given execution unit in a single cycle. If two instructions are ready to begin execution on the same execution unit on the same cycle, the oldest instruction goes first. An integer, floating point, or branch instruction can only begin execution in the execution unit directly below the reservation station it was issued to. Memory instructions are much more complicated.

Size of Reservation Station: After fetch the opcode of instruction the instruction wait in reservation station or called also dispatch buffer in case of centralized architecture. After that each instruction goes to their corresponding execution units.

Rename Entries: Rename entries is the number of entries in the processor's register renaming buffer. This value limits the number of instructions, which potentially modify an architectural register that can simultaneously be 'in-flight'. The instruction which has issued already (left the issue stage) and has not yet to be committed (left the commit stage) is considered in-flight. The rename buffer stores information that actually shows which instruction result have been assigned to each of rename entries and whether it's result valid or not. If the instruction has not completed the result is invalid, else it has been completed and broadcasted its results, but has not yet committed (written back).

Reorder Entries: The number of entries in the processor's instruction reorder buffer. This value limits the number of instructions that can simultaneously be 'in-flight'. Any

CONFERENCE PAPER II International Conference on
"Advance Computing and Creating Entrepreneurs (ACCE2013)"
On 19-20 Feb 2013
Organized by
2nd SIG-WNs, Div IV & Udaipur Chapter , CSI , IEEE Computer Society Chapter India Council ,
IEEE Student Chapter Geetanjali Institute of Technical Studies, Udaipur, Rajasthan, India

instruction that has issued (left the issue stage) and has not committed (left the commit stage) is considered in-flight. Instructions that do not modify an architectural register, (branches and stores), do not require a rename entry, but they do require a reorder entry. The reorder buffer holds the instruction number of all in-flight instructions. The instruction can be uncompleted or completed and waiting its turn to commit and write back. As soon as it turns to commit and write back the entry will be unallocated.

IV. SIMULATION RESULTS FOR DISTRIBUTED RESERVATION ARCHITECTURE

According to Shen and Lipasti [2], the best mix of functional units for a superscalar pipeline depends on the application domain. Typical programs have 40% ALU instructions, 20% branches, and 40% load/store instructions. According to 4-2-4 rule of thumb, for every four ALU units; two branch units, and four load/store units shall be optimum. By using the above data, the superscalar factor was varied through 1 to 16 and entries per reservation station was varied through 1 to 8. The graph between Superscalar factor and IPC for distributed reservation architecture is shown in Figure 3.

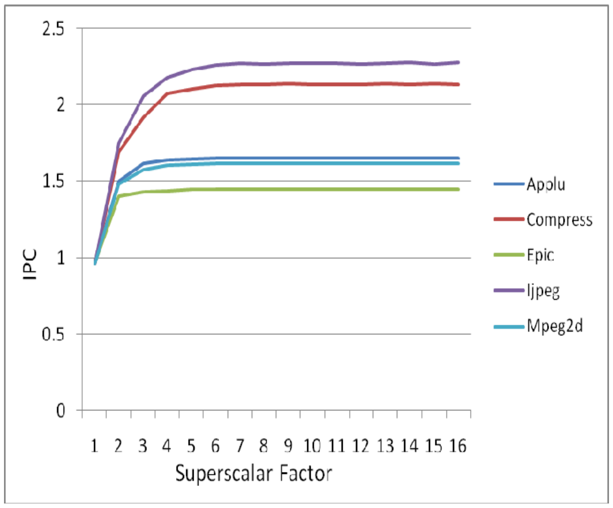

Figure 3 Superscalar factor and IPC for distributed reservation architecture

In figure 3, a graph is plotted between Superscalar factor and IPC for distributed reservation architecture at size of reservation station is 2. It is observed that, for superscalar factor 1 through 16, as the superscalar factor increases, the number of clock cycles increases significantly. After superscalar factor 4 there are no significant improvements in clock cycles. Thus we can say that optimal value for superscalar factor is four for distributed reservation station architecture.

V. SIMULATION RESULTS FOR CENTRALISED RESERVATION ARCHITECTURE

The graph between Superscalar factor and IPC for centralized reservation architecture is shown in Figure 4.

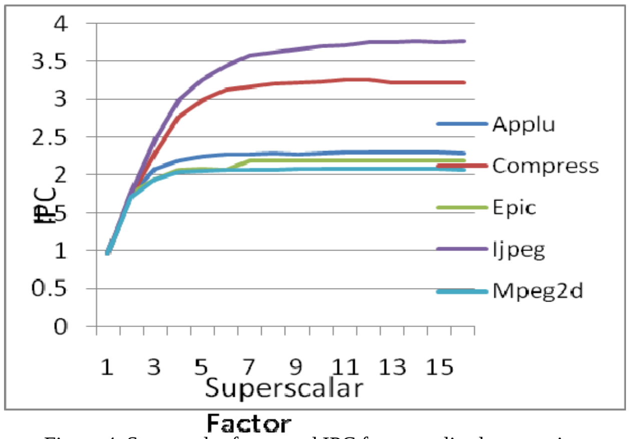

Figure 4 Superscalar factor and IPC for centralised reservation architecture

In figure 4 graph is plotted between Superscalar factor and IPC for centralized reservation architecture at size of reservation station is 4. We can see that for superscalar factor 1 through 16, as the superscalar factor increases, the IPC increases significantly. After superscalar factor 4 there are no significant improvements in clock cycles. Thus we can say that the optimal value for superscalar factor is four for centralized reservation station architecture.

VI. CONCLUSION

The simulation provides the comparison between distributed & centralized reservation station architecture at different size of reservation station, and it is found that for both architectures (distributed and centralized), the optimal value of superscalar factor is four in superscalar pipeline processor architecture.

VII. REFERENCES

[1] Smullen Clint W., Taha Tarek M. “ PSATSim: An Interactive Graphical Superscalar Architecture Simulator for Power and Performance Analysis”

[2] J. P. Shen and M. H. Lipasti. Modern Processor Design: Fundamentals of Superscalar Processors. McGraw-Hill, 2005.

[3] Sohi Gurindar S., Vajapeyam Sriram, “Instruction Issue Logic For High-Performance, Interruptable Pipelined Processors”

[4] Lazarov Vladimir, Marinova Maria “Dependencies Evaluation in Superscalar Processors” International Conference on Computer Systems and Technologies - CompSysTech’2004.

[5] Hwang Kai. “Adanced Computer Architecture”, McGraw-Hill, 2009.

[6] Seyed-Abdollah Aftabjahani, “Impact of Branch Prediction Accuracy on Superscalar Performance ”

[7] Stallinngs W., “Computer Organization & Architecture”, 7th Edition, PHI, 2006.

[8] SPEC, The SPEC benchmark programs, http://www.spec.org

“Advance Computing and Creating Entrepreneurs (ACCE2013)”