A Simulation Model for Macro- and Micro-Fusion Algorithms in the CPU Core

A V Vishnevov1, E M Ivanova1, N A Stepanov1, N D Shaimov1

1School of Computer Engineering, National Research University Higher School of Economics, 34, Tallinskaya St., Moscow, 123458, Russia

emivanova@hse.ru

Abstract. The article discusses the features of modern processor’s microarchitecture, the method of instruction’s and micro-operation’s accelerated execution. The research focuses on the organization of the decoding stage in the CPU core pipeline and Macro- and Micro-fusion algorithms. The Macro- and Micro-fusion mechanisms are defined. A computer simulator has been developed to explore these mechanisms. The developed software has a user-friendly interface, is easy to use, and combines training and research options. The computer simulator demonstrates the sequence of mechanism’s implementation; the resulting macro- or micro-operations set after Macro- and Micro-fusion, and also reflects each algorithm features for different processor’s families. The software allows you to use either a pre-prepared file with Assembler (x86) code fragments as source data, or enter/change the source code fragments at your request. The main combinations of machine instructions that can be fused into a single macro-operation are considered, as well as instructions that can be decoded into fused micro-operations. The simulator can be useful both for in Computer Science & Engineering students, especially for on-line education and for researchers and General-purpose CPU cores developers.

1. Introduction

Users try to get a high-performance computing system. The growth of computation speed is the main engine and one of the main trends in the development of computer architecture. High computer performance is provided, among other things, by constant changes in the processor’s microarchitecture. One of the main trends in improving the design technology of processor chips is parallelization of computation: both between several cores and within each core. Modern processor is very complex device; therefore, there are many different speedup technologies: the pipeline processing of machine instructions [1-7], complicating the pipeline and increasing the number of its stages[1,6], multi-stage instructions decoding [1, 6, 7], combining the instruction and the micro-operation pipelines in one CPU core, principles of pipeline division into two clusters: front-end and back-end [8-11], optimization (micro-optimization) of program code during its execution by forcibly reducing the number of instructions and micro-operations executed by pipelines [1, 7, 11-13]. The last one is the most interesting technology. The program code itself does not change, but the hardware blocks of the CPU core are loaded with slightly modified instructions and operations.

The topic of this research is the analysis of methods to increase the CPU performance, namely one of them – Macro- and Micro-fusion technologies. This technology is used in Intel CPU cores and its detailed research will help all designers to improve the efficiency of processors. The simplest and most understandable way to analyse it (including for studying by University students) is to create a computer
model with configurable parameters, which will also make it possible to study both existing Macro- and Micro-fusion algorithms and create new ones. Several scientific and practical works are devoted to this issue [1, 6, 7, 13-16]. We have also investigated the existing CPU simulators [18-20] and have compared them with our solution. There are no simulators which allow to trace Macro- and Micro-fusion algorithms for existing CPU architectures and none of the solutions makes it possible to trace the procedure visually. Furthermore, some of the existing simulators are quite complicated, which makes it difficult to learn and use for educational purposes. Main ideas when creating the computer Fusion Simulator [21] are simplicity, multilanguage, visibility, training and research options.

2. The model of Macro-operation fusion

Before creating the simulator itself we had to determine simulated CPU architecture. We collected information about definitions for Macro-operation (MOP), Micro-operations (μop) and their fusion in various architectures [7]. For our purposes Intel turned out the most suitable architecture for a few reasons. First, Intel was one of the first one to implement fusion technology [13]. Secondly, there is a lot more information on Intel microarchitectures. Thirdly, Intel has more available architectures for simulation. If we look at AMD architectures, Macro-fusion is called «branch fusion» and appears only starting from Zen architectures. For RISC architectures fusion is almost completely absent. So our choice was definitely Intel.

Now a few words about Macro-fusion. Two x86 instructions, for example the most common case is CMP or TEST followed by conditional branch, are combined into a single compare and branch μop at the decoding stage [7, 14], that operation is not splitting at the execution units and executed as a single operation (Figure 1). This means that Macro-fusion reduces the amount of machine code actually executed in the CPU Back-end, saves bandwidth in all stages of the pipeline from decoding to the retirement stage, increases the processor IPC (number of Instructions executed Per Cycle), and can increase computer performance by several percent.

![Intel CPU Core Front-end](image)

Programme code example:

```c
for (unsigned int i=0; i<100000; i++) {...}
```

Corresponding instruction queue (version):

```
label:
...
push ecx
mov ecx,i
inc ecx
cmp ecx,100000
jae label
```

**Figure 1. Instruction decode/Macro-fusion**

As for Micro-fusion, it strongly depends on core microarchitecture (design). All μops can be roughly divided into four types: Address generation (AG), Store Data (SD), Data Processing (DP) and Read Data (RD) [13, 22]. The CPU can fuse only two sequential μops. It can be either: AG+SD or DP+RD operation. The older microarchitectures allow only one micro fusion per instruction while newer allow two fusions, but they must be different types [7]. Figure 2 shows a Micro-fusion model in the CPU core. The main idea of μops fusion is to improve the bandwidth after decoding\(^1\). Sequential dependent μops from a single instruction are combined into a single μop to move through the next stages of the pipeline.

---

\(^1\) Up to 6 instructions can move along the pipeline at the same time, and each of them can be decoded to several μops, which increases greatly the μop-flow along the pipeline.
In summary Micro-fusion (so the Macro-fusion do) reduces the amount μop-transfers along the pipeline, code execution time, increases the processor/computer performance.

3. The simulator description
The computer Fusion Simulator has a multi-window interface. The «Settings» tab provides the ability to switch the interface language, select one of the standard CPU microarchitectures, which automatically configure the of Micro- and Macro-fusion algorithm settings (Figure 3).

User can additionally change these settings (in related tab-areas). This way you can simulate and study your own microarchitecture.

On the «Macro level» tab you can open or create the initial data: load a set of examples from the file and select one of the x86-Assembler code fragments or input your own instructions (Figure 4). After the start of the simulation you will see a corresponding μop-list where fused ones will be highlighted with yellow.
And the last tab is a «Micro level» tab where you can trace fusion on both Macro and Micro level (Figure 5). You can see all the instruction from the initial code fragment, associated operands, associated μops each instruction is decoded to (marked «1» if present or «0» otherwise).

Here you can see four columns for each type of μop:
- READ (read data μop),
- MODIFY (data processing μop),
- ADDRESS (address generation μop),
- WRITE (store data μop).

The next two columns show how many μops execute in «Fused domain» and «Unfused domain». If there are some fused μops associated with instruction the last column specified them.
4. Conclusion
The offered Fusion Simulator allows to study the algorithms of Macro- and Micro-fusion during instruction decoding in the CPU core pipeline. This Simulator is useful for studying computer architecture. Understanding the features of the processor is necessary for both system developers and programmers. Improving the style of writing programs will let the computer speed up their execution. The Fusion Simulator applies when teaching the discipline «Modern Computer Systems» in Higher School of Economics [23]. The simulator can be useful for masters in Computer Science & Engineering, especially for on-line education. The Simulator may be of interest to researchers and General-purpose CPU cores developers both for exploration their own core architecture and for code optimization that will help speed up the processor core.

5. References
[1] Intel Corporation 2019 Intel® 64 and IA-32 Architectures Optimization Reference Manual
[2] Abeydeera M Subramanian S Jeffrey M Emer J Sanchez D 2017 SAM: Optimizing Multithreaded Cores for Speculative Parallelism Proc. 26th Int. Conf. on Parallel Architectures and Compilation Techniques (Portland: Oregon, USA) 64-78
[3] Intel Corporation 2018 Intel Analysis of Speculative Execution Side Channels
[4] Polvyvalov A 2000 In anticipation of the Willamette – the history of the IA-32 architecture and the way the P6 family processors work | iXBT.com ixbt.com
[5] Garmatyuk S 2008 The "new old" Core i7 architecture: what's more-similarities or differences? | iXBT.com ixbt.com
[6] Intel Corporation 2020 Intel 64 and IA-32 Architectures Software Developer’s Manual
[7] Fog A 2020 The microarchitecture of Intel, AMD and VIA CPUs (Lyngby: Technical University of Denmark)
[8] Swinburne R 2008 Intel Core i7 - Nehalem Architecture Dive | bit-tech Bit-tech.net
[9] Ozerov S 2005 CPU architecture Computerra CTERRA(2005)609
[10] Cutress I 2015 The Intel Skylake Mobile and Desktop Launch, with Architecture Analysis | AnandTech anandtech.com
[11] Patterson D & Hennessy J 2019 Computer Architecture: A Quantitative Approach 6th Edition (Burlington: Morgan Kaufmann)
[12] Reitbauer A 2017 Micro operations — A new operations model for the micro services age | kepnt medium.com/kepnt
[13] Bakhvalov D 2020 MicroFusion in Intel CPUs | Easyperf Easyperf.net
[14] Macro-Operation Fusion (MOP Fusion) - WikiChip En.wikichip.org
[15] Ronny R Nathaniel H 2004 System and method for using instructions (pat. US6675376B2 USA)
[16] Taram M Venkat A Tullsen D 2018 Mobilizing the Micro-Ops: Exploiting Context Sensitive Decoding for Security and Energy Efficiency Proc. ACM/IEEE 45th An. Int. Symp. on Computer Architecture (Los Angeles: CA, USA) p 624-637
[17] Moiseev I 2019 Processors of modern computing systems (SPb: Saint Petersburg State University)
[18] DrMIPS :: About DrMIPS GitHub
[19] Dr. Martin Perner M 2020 Mikrocodesimulator MikroSim 2010 Mikrocodesimulator.de
[20] Besim M 2013 YASS: A system simulator for operating system and computer architecture teaching and learning European Journal of Science and Mathematics Education 1(1)
[21] Macro_Micro_Fusion_Simulator GitHub
[22] Fog A 2020 Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs (Lyngby: Technical University of Denmark)
[23] Modern Computer Systems – Courses – Higher School of Economics National Research University Hse.ru