CSC 565 - Computer System Architecture: Paper Summary #1

Due on January 27, 2023 @ 11:59 PM, EST

Decisive Aspects in the Evolution of Microprocessors

Cason Konzer

January 27, 2023

Summary

In this paper summary we will discuss the work of [1] up until section **IV.A**. A condensed summary is provided by the journal's prelog [2].

The paper is acronym (and variable) heavy, so let us first set the nomenclature.

General:

- ILP: Instruction Level Parallelism.
- ISA: Instruction Set Architecture.
- SIMD: Single Instruction Multiple Data.

. . . . . .

Technical:

- f_c: Clock Frequency.
  - *Clock Cycles per Second, measured in †Hz where † is a large metric prefix such as (G)iga.*
- c_i: Issue Cycle i.
  - *Clock Cycle with at least 1 Instruction Issued.*
- II (s_i): Issue Interval i.
  - *Time Between Issue Cycle i and $i+1$.*
- IPII ($\overline{n}_{\text{IPII}}$): Average # Instructions Issued per Issue Interval.
- CPII ($\overline{n}_{\text{CPII}}$): Average # Clock Cycles per Issue Interval.
- IPC ($\overline{n}_{\text{IPC}}$): Average # of Instructions Issued per Clock Cycle.
  - *(Per Cycle) Throughput of the Processor.*
  - *Execution Width of the Processor.*
- P_{pa}: Absolute Processor Performance.
  - *Usually Represented in Instructions per Second.*
  - *Sometimes Represented in Data Operations per Second.*
- OPI: Average # Data Operations per Instruction.
- OPC: Average # of Data Operations Processed per Clock Cycle.

With the terminology in place, let us dive into the sections.

Abstract

In brevity, Sima highlights that performance demands in the computing space have driven key technological advancements in clock frequencies and microarchitecture. With the paper focusing on the side of microarchitecture, he elaborates on the historical development of ILP, introducing *temporal, issue, & intrainstruction* parallelism. Specifically noted is that advancements within one domain of parallelism prompted similar innovations along others, due to their arising bottlenecks imposed. In conclusion, it is reiterated, the content is based on historical evolution (prior to 2004) of microarchitecture.

I: *Introduction*

As a result of the advancements in microarchitectures, comes performance increases. At the time of writing, IPC is considered to be approaching limits, driving techniques to leverage ILP at the compiler level, as well as use of parallelism via multiprocessing and multithreading. This is stated and the use ILP in addition to use of higher-than-instruction parallelism [1]. In general, advancements in microarchitecture drive P_{pa}, and the article takes a clear focus on ILP. The introduction rounds itself off with an outline of the remaining sections.

II: *Design Space of Increasing Processor Performance*

Given the well defined layout of the content, Sima now dives into the nomenclature we mention at the onset, along with two key ways of measuring performance. First *relative performance measures* are recalled, with reference to those commonly used, such as Hennessy and Patterson's CPU time [3], which measures running time for a specific program. The benefit of this kind of measure is the ability to compare across ISAs, in comparison to *absolute performance measures* such as P_{pa} which provide a more isolated view of performance potential within a select ISA [1], of which the later is the focus within the work. Foundational to understanding the many variables aforementioned is to know how they are related, laid out in the first five equations.

$$P_{pa} = f_c \times \text{IPC} \tag{1}$$

$$\text{IPC} = \frac{\text{IPII}}{\text{CPII}} \tag{2}$$

$$\text{OPC} = \text{IPC} \times \text{OPI} \tag{3}$$

$$\text{OPC} = \frac{\text{IPII} \times \text{OPI}}{\text{CPII}} \tag{4}$$

$$P_{pa} = \frac{f_c \times \text{IPII} \times \text{OPI}}{\text{CPII}} \tag{5}$$

(1) provides a high level picture of what is meant by *absolute processor performance*, the average instructions per second. IIs are detailed with the subtlety that not all clock cycles issue instructions, thus allowing a decomposition of IPC into its base components CPII <*temporal parallelism*> and IPII <*issue parallelism*> (2). Notes are made here that in processors issuing an instruction in every clock cycle CPII = 1, and IPII > 1 is the distinguishing factor between superscalers and scalers such that IPII = 1.

Discussion now shifts into one of focus on execution of data operations rather than instructions alone. It is explained that often it is the case instructions perform exactly one data operation, but in the context of control or SIMD instructions none or many operations can be executed respectively [1]. This triggers further decomposition of IPC into the ratio OPC/OPI, expressed similar to (3), which is substituted into (2) and solved for OPC in (4). Here OPI is revealed as *intrainstruction parallelism* and OPC combines the three aspects microarchitectures can exploit to leverage ILP. A brief aside is made to comment on the chronology of microarchitecture evolutions such that the move from sequential to pipelined processors later pushed the advances from scaler to superscalers last followed by increases in average data operations per instruction. In addition, f_c is classified as a dimension outside of ILP exploitations, interestingly driving its own set of evolutions into multi-core processors when hitting a 'power wall' [4]. Last P_{pa} is redefined as $f_c \times OPC$, exposing one formula for *absolute processor performance* along a total of four dimensions. In all cases but when $OPC = 1$ this definition contradicts the prior, and thus a distinguishing notation such as P_{pa}^{I} and P_{pa}^{O} would bring clarity to the matter.

III: *Increasing the Clock Frequency and its Implications*

Starting in **III.A**, f_c is given an intrinsic upper bound determined by *worst case path length* which in turn describes the advancements in increased frequencies via minimization of such length. This is primarily achieved through increased pipeline lengths, which in turn bring their own challenges with increased *misprediction penalties* and *execution unit latency* [1]. This idea is well described in [3] as increases in the number of pipeline stages most often increases the execution of a single instruction while increasing instruction throughput. In designing longer pipelines balancing stage latency minimized the *worst case path length* and hence the maximum stage latency driving further increases in throughput. Due to the engineering overhead, and diminishing return on investment, an *optimal design point* is sought for maximizing profits by limiting throughput. Following discussion of increased clock frequencies in Intel processors the article dives into resulting bottlenecks in **III.B**.

Three subsystems are identified as the key bottlenecks requiring advancements: *bus, memory, & I/O*. Advancements in the *bus* subsystems focus on increased data width and frequency; In the *memory* subsystem evolution was driven by new components, cache enhancement via their organization, levels, capacity, and latency; The *I/O* subsystem increased storage capacity, and in terms of displays, resolution, while decreasing access times [1]. These advancements are mentioned in passing and limited detail before switching gears to the article's more technical content.

IV.A: *Introduction and Evolution of Utilizing Temporal Parallelism : Overview of Possible Approaches to Introduce Temporal Parallelism*

This subsection alone reemphasizes the definition of *temporal parallelism* as defined in **II**, followed by breaking it's advancements into methodological processes. This evolution is presented again in chronological order, via means of *prefetching, pipelining execution units, & pipelining processors*. The first advancement to *sequential* processing came from the overlap of write and fetch stages in processor stages, as not until the decode stage were dependencies from the write stage retrieved (*prefetching*). Evolution followed with the introduction of pipelining, first with the execution units, and second the processor as a whole. Both first and in parallel, word length extensions allowed for the development of new ISAs building in backwards compatibility for their legacy counterparts [1]. Pipelined processors containing pipelined execution units set the convention for today's microarchitectures.

Conclusion

Sima's work provides an elegant overview within the evolution of microarchitectures. Truly there is more than can be covered within one journal article, and thus many details are left to the reader's independent study. Of the paper's focus, explanation is provided in depth, focusing on ILP along three main dimensions: *temporal, issue, & intrainstruction* parallelism. The foundational concepts of microarchitecture analysis are presented, and regards are given with respect to increases in clock frequencies and the separate evolutions they drive. The summary we provide leaves off at the evolution of *temporal parallelism*, while the work itself continues to explore *issue & intrainstruction* dimensions.

References

[1] D. Sima, "Decisive aspects in the evolution of microprocessors," *Proceedings of the IEEE*, vol. 92, no. 12, pp. 1896–1926, 2004.

[2] H. Falk, "Prelog to decisive aspects in the evolution of microprocessors," *Proceedings of the IEEE*, vol. 92, no. 12, pp. 1895–1895, 2004.

[3] J. L. Hennessy and D. A. Patterson, *Computer Architecture, Sixth Edition: A Quantitative Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 6th ed., 2017.

[4] R. E. Bryant and D. R. O'Hallaron, *Computer Systems: A Programmer's Perspective*. Pearson, 3rd, global ed., 2019.