

## CHAPTER I

### INTRODUCTION

As the computing industry continues to push advancements in microprocessor technology, a major effort for improving performance has focused on Instruction Level Parallelism (ILP). ILP is simply a set of techniques, which the hardware uses to execute more than one instruction in a single clock cycle. Superscalar (which will soon be discussed) and pipelined implementations are two of the most common methods for accomplishing ILP. However, hardware for these processors has grown so complex that it is presenting major challenges to computer architects to make further advancements in computer performance<sup>1</sup>. In order to continue the pursuit of “the need for speed” in microprocessors, a new architectural approach is becoming increasingly popular. This new approach is referred to as Explicitly Parallel Instruction Computing (EPIC). EPIC enables high levels of parallelism without adding additional complexity to the hardware.

Advancement of some of the more popular architectures (Intel x86, Motorola 68xx, PowerPC, and MIPS) have been accomplished without much rewriting of programs, without changing algorithms or languages, and sometimes without even recompiling existing programs. Instruction-level parallelism has so far been the only practical approach to increasing performance without changes to the way software is written.

## Instruction-Level Parallelism

When it comes to improving current microprocessor technology, improvements in semiconductor technology might first come to mind. While semiconductor improvements do increase both circuit speed and density, these improvements do have limits. Other performance improvements stem from some sort of parallelism, or a processor being able to execute more than one instruction in a single clock cycle.

Instruction-level parallelism is a set of techniques implemented by either the hardware or the compiler. Systems, which exploit ILP, take programs written in sequential form in some high-level language, and use the compiler and/or hardware to schedule instructions in parallel. ILP techniques have been, for the most part, invisible to the programmer (except in multiprocessor parallel systems). This means that programmers can enjoy the benefits of ILP without changing the style in which they write their programs. Due to this fact, ILP has been the only parallelism solution available to continue to push forward performance without re-writing programs<sup>2</sup>.

Two fundamental ILP architectures most often seen are superscalar processors and very-long instruction word (VLIW) processors. Superscalar processors, described more thoroughly in the next chapter, make use of parallelism purely using hardware. This means that no information is conveyed from the compiler to the hardware about a plan of action for executing instructions in parallel. This type of architecture is becoming too complex to make further improvements of performance practical.

VLIW processors, on the other hand, rely on explicit information from the compiler to schedule instructions to execute in parallel. By taking this burden off the

processor, it becomes possible to make further performance advancements without making the hardware overly complex. EPIC is based on the VLIW architecture as the compiler is responsible for conveying information to the hardware about many details, such as a plan of execution (POE) for instructions to execute in parallel, and speculative information to help improve branch performance<sup>1</sup>.

In this paper, the fundamentals of EPIC are discussed. Included is an investigation of the three major EPIC features: predication, speculation, and parallelism. Also, a description of the Intel IA-64 (Itanium), and EPIC based processor, is presented. This will allow the reader to not only realize the major differences in the EPIC architecture from others (such as Pentium), but also to understand the major differences between the assembly languages for the Itanium and others (such as MIPS).