

Exploring Comparative Computing Architectures as Means & Method of Teaching Computer Engineering Design

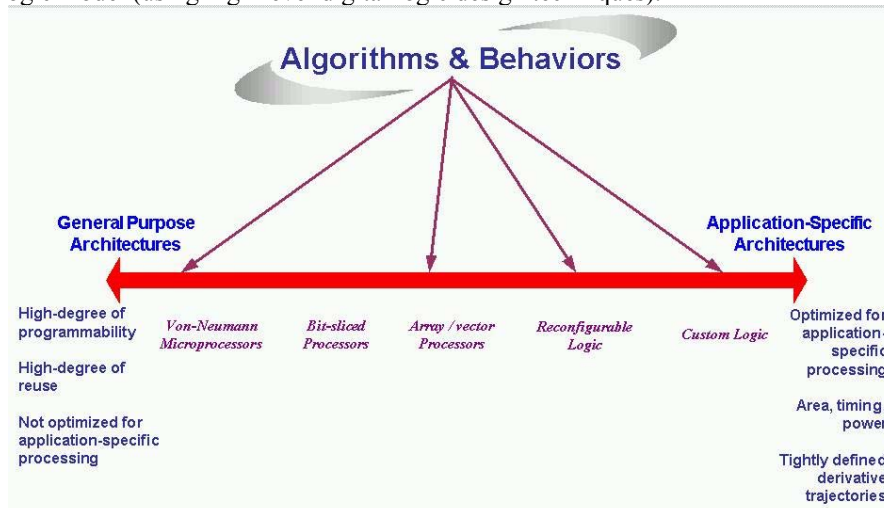
James P. Davis and Tiffany M. Mintz
Department of Computer Science and Engineering
jimdavis@cse.sc.edu

Abstract

The computing industry is rapidly changing how it designs and delivers computing solutions. The traditional ways of designing computer applications using the so-called “von Neumann” model of computing (named after the eminent mathematician John von Neumann) is giving way to the design of application-specific computing solutions implemented in so-called “reconfigurable” custom logic chips. One of the inherent differences in the characteristics of this shift in computing architecture is the high degree of concurrency and parallelism ingrained directly into the system. Our position paper presents this shift in thinking, and discusses how we are looking to develop new teaching methods to allow Computer Engineering students to explore these different styles of computing—such that, by *comparative anatomy*, if you will, the students may gain greater insight into the profound shift in computing that will be evolving over the next decade.

1. Introduction

This undergraduate research project is exploring the space between the microprocessor-based--and custom-logic based--systems design approaches that are the hallmark of the Computer Engineering hardware design discipline. As such, we are exploring a teaching method that involves immersing CE students in the different design methods, so that by direct insight, they might gain understanding of the differences in computing styles currently in use—one of the past, the other of the future. To that end, we are exploring a method based on taking a conventional microprocessor, the Motorola MC68000©, and re-architecting major portions of it for implementation as a *custom logic* model. We are attempting to explore re-architecture and design using a hybrid design methodology while also performing experiments to compare the standard microprocessor model (using assembler language programming) and the custom-logic model (using high-level digital logic design techniques).



The project objective is to gauge the effort and cognitive activity associated with creating a custom logic model of a microprocessor, whereby we take the core functionality of the processor, and re-implement it on top of a custom logic device—one whose specific functions are designed according to the specific algorithm or application. The test of a model thus created involves: (1) creating test programs, using a range of 68000 instructions, writing assembly language, compiling and loading the program’s bit patterns into a region in program memory, (2) loading these programs’ executable code into the memory for a custom logic device “emulating” the microprocessor’s instruction processing capabilities, (3) executing these program codes on this model, and (4) comparing the results of executing the program instructions on the native Motorola processor against the code executed on the custom logic “emulation” model. Thus, the experience of the students in this comparative exploration encompasses the creation of algorithms and programs in the conventional software programming sense, as well as the creation of digital logic designs that are specific to the particular algorithms created using computer chip design techniques.

The questions being asked are: (1) how difficult it is to architect, design and implement such an “emulator” model for a conventional microprocessor, and (2) can we implement a model that runs cycle for cycle as—or faster than—the native microprocessor.