Why Computers Are Too Slow: A Computer Architect's Perspective

First Author FIRSTAUTHOR@CSAIL.MIT.EDU
Second Author SECONDAUTHOR@CSAIL.MIT.EDU
MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge MA, 02139 USA

1. Introduction

Since their introduction, computers have been slow. When introduced, computers were supposed to make calculations run faster, but to this day, computer architects and compiler writers have failed at making all computation instantly fast. Most humans think it would be good enough to make computation fast enough that they cannot determine when the computer is computing. This is a selfish goal because it does not take into account that alien species or robots may have different notions of how fast "instantly fast" is. In this paper we advocate that all computations that ever could be done should be so fast that even fast robots won't notice that computation is going on.

2. Related Work

There have been many approaches to making computation faster. One field that has focused on making computation faster is computer architecture. There is a bunch of research on computer architecture. In these works(**?**; **?**), people have tried to make computers run faster.

3. System Overview

3.1 Hardware

The hardware that we used to implement this system is a Intel 8051. To increase the clock speed, we have cooled it with liquid Nitrogen and put it in a warp field to make it run faster. Inside of the warp field, time runs in an accelerated manner. To properly interface from non-accelerated time to accelerated time, a photon speed gasket was designed. Figure **??** shows a block diagram of our hardware setup.

3.2 Software

In order to run all computation "instantly fast" for ultra fast robots, we have developed software that is able to run in constant time. While it is well known that P==NP, polynomial time algorithms are not fast enough for large problem sizes. Thus we have designed a compiler that is able to take polynomial time algorithms and transform them into constant time algorithms. To allow for quick application development and code reuse, we use the 'C' programming language for all code development. The Constant Compiler developed in a partnership with Spanuza Research transforms all of the non-constant time algorithms to constant time equivalents all in constant time. This research required an exponential amount of time to perform.

4. Future Work

In the future we plan to make computers smaller such that all ants in the world can carry at least three of them on their back. And we will nominate you, yes you, as Imperator of Rome! Hail Caesar!