What Makes HPEC Applications Challenging?
Understanding Application/Architecture Interactions

David Koester, Ph.D
The MITRE Corporation
dkoester@mitre.org

Introduction

Frequently we develop and run benchmarks on computer architectures to examine performance and often to compare systems for either research or for procurements. Too often those benchmarks are run as "black-boxes" with little or no understanding of the theoretical reasons why particular performance levels are observed for various architectures. High performance embedded computer (HPEC) architectures are very complicated today, with systems having a wide variety of processors and often being multicomputer architectures. As embedded applications move beyond traditional signal processing and include data-driven signal processing and knowledge formation algorithms, HPEC application developers may encounter significant degradation in expected performance when compared to traditional embedded kernel benchmarks.

Challenges

It is critical for the HPEC community to understand the interactions between applications and architectures and be able to articulate *"What makes HPEC applications challenging?"* The *"challenging"* part has a lot to do with the "*High Performance*" in *HPE*C. We want to get the most out of a system and the HPEC community traditionally asks for more in terms of performance than traditional high performance computing (HPC). The HPEC community is often driven by factors like power consumption — due to the communities that embedded computing support. Power consumption and heat extraction are critical system design issues in embedded systems and not just budgetary concerns for the electrical power to run and cool a high performance computer.

"Application Challenges" create system *"bottlenecks"* and the performance of any system *"in series"* is obviously limited by the lowest performance component. We need to understand both HPEC applications and architectures so that we can choose the architecture with the *"right"* resources for the application. Identifying bottlenecks requires that we be able to identify where is performance lost when an application is run on an architecture. Understanding application challenges can also assist in making an informed decision when deciding on what architecture characteristics would make worthwhile investments to improve application performance. For many application/architecture combinations, memory latency and bandwidth limitations — often referred to as the *memory wall* — cause substantial bottlenecks. For other application/ architecture combinations, limited concurrency causes significant bottlenecks.

Performance beyond a single compute engine is possible only as a result of concurrency (parallelism) in applications. For extensive improvements in performance, applications must exhibit the characteristic of hierarchical concurrency — pipelining on a processor, multi-core chips, symmetric multiprocessing nodes, and constellations of nodes. This hierarchy can be viewed as "in the small" (i.e., "locally") or "in the large" (i.e., "system-wide"). Concurrency can take the form of task or data decomposition or data flow.

In the presentation, we will address the following application characteristics that can cause HPEC applications to be "*challenging*".

- Memory access patterns
  - Locally — Spatial and Temporal locality
  - System-wide — Data Decomposition
- Computational intensity
  - Locally
    - Arithmetic density
    - Instruction mix
      Floating point versus integer
    - Branch dependencies
    - Data dependencies
  - System-wide — Load balance
- Synchronization
- I/O

Example *Challenges*

The effects of branch and data dependencies on memory performance for IBM Power3 PowerPC and Intel Itanium microprocessors are presented in the two figures below. Effective memory bandwidth is

substantially reduced for both branch and floating point data dependencies when compared to stride 1 memory access. This data has been collected by the Performance Modeling and Characterization (PMaC) team at the San Diego Supercomputing Center (SDSC).

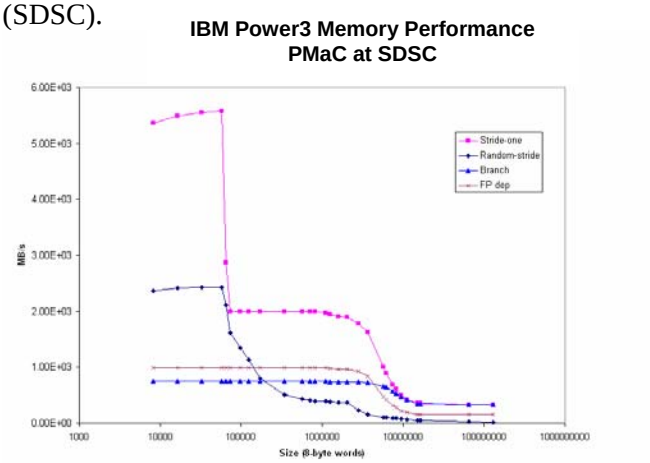

Figure 1: Effective Memory Bandwidth for the IBM Power3 PowerPC Microprocessor [1]

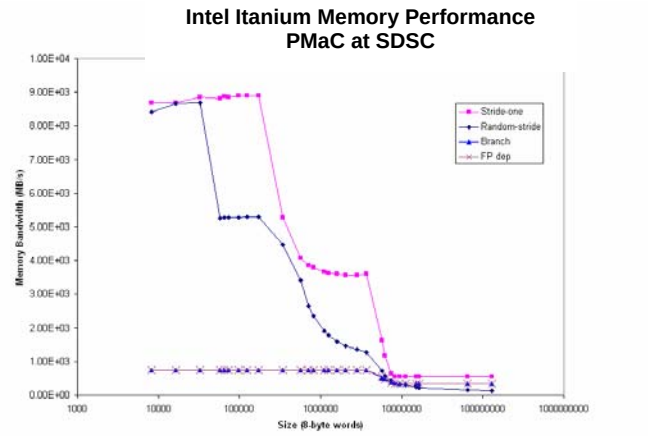

Figure 2: Effective Memory Bandwidth for the Intel Itanium Microprocessor [1]

Summary

In the HPEC conference presentation, we will examine those issues on the aforementioned list in detail, discuss effective metrics to understand the characteristics of an application, and propose a spectrum of benchmarks for users to examine application "*challenges*". We will compare traditional signal processing applications with new data driven signal processing and knowledge formation applications. Many of the issues that make applications "*challenging*" are related to the memory wall and not to Moore's Law. The memory wall has existed for many years and efforts to minimize the effects of the memory wall in microprocessors have consumed a large percentage of the transistors made available as the transistor density has increased. Meanwhile, the degraded performance in the figure is a result of disruptions in the computational pipeline — an incidence of currency within the microprocessor. Consequently, HPEC application programmers need to understand thoroughly the "*challenges*" in applications if there is any possibility for software to save Moore's Law.

References

[1] A. Snavely and L. Carrington, Improvements in Performance Prediction Methodology, Performance Modeling and Characterization Lab at the San Diego Supercomputing Center.