Compiler Support for Application Migration in Heterogeneous-ISA Platforms

Rob Lyerly Christopher Jelesnianski Antonio Barabalace Binoy Ravindran
Bradley Department of Electrical and Computer Engineering, Virginia Tech
{rlyerly, bielsk1, antoniob, binoy}@vt.edu

Abstract

In recent years there has been a proliferation of heterogeneous-ISA systems ranging from mobile devices to warehouse-scale computers. These systems, which couple together processors with unique execution characteristics, seek to accommodate many different types of applications. Additionally, many of these architectures are OS-capable, meaning they are able to run an operating system. With this ability comes a number of benefits (e.g. I/O capabilities, scheduling, etc.). However, current OSes are unable to fully exploit these systems as individual architectures within the system are loosely coupled, preventing the OS from being able to manage the entire system. Additionally, programming these systems is challenging, as the developer must write inflexible and tedious boilerplate code that handles low-level device and memory management. As these systems become increasingly ubiquitous, new OS and compiler tools must reduce the barrier to entry in order to maximize full-system performance and enable developer productivity.

In this work we seek to exploit emerging heterogeneous-ISA platforms to gain energy efficiency and high performance with minimal programmer effort. In particular, our approach lets developers utilize existing applications written using the shared-memory programming model in a heterogeneous-ISA context. This is in contrast to existing programming practices, which require refactoring portions of applications into offloadable sections. With our approach, migration execution can occur at arbitrary function boundaries, allowing applications to utilize the OS scheduler for several types of benefits (e.g. fairness, isolation). Indeed, the Linux scheduler migrates execution between cores at arbitrary points to balance load and improve performance. OS schedulability provides benefits over the queue-and-execute approach favored in current GPU programming models, which prevents OS scheduling optimization.

Our approach uses a combination of OS, runtime and compiler support to handle architectural differences (ABI, ISA, and micro-architecture) and data management. We build upon Popcorn Linux [1], a replicated-kernel OS that provides inter-architecture migration and distributed shared memory (DSM). DSM allows the developer to focus on application logic rather than marshaling memory; Popcorn manages memory coherency between discrete memory regions transparently. Alternatively, offloading requires developers to manually pack data structures for streaming between devices, which can be problematic for many types of data structures (e.g. trees). Because there is a single virtual memory space, minimal developer effort is required to manage memory.

We introduce a toolchain that analyzes and refactors arbitrary applications with the ability to migrate execution between architectures, built explicitly for Popcorn Linux. Our toolchain utilizes a source-to-source compiler built on top of LLVM to automatically insert migration code into the original application source. Additionally, our toolchain generates an aligned binary per architecture, where data and functions are loaded at runtime at aligned addresses across architectures. This enables Popcorn to manage inter-architecture memory consistency, and allows for functions to become inter-architecture migration points. With this support we enable intelligent mapping of applications to architectures for varying goals such as performance, energy savings, etc. Through this work, we seek to understand what benefits are obtainable and what overheads exist for migration on heterogeneous-ISA platforms. Additionally, we seek to understand what types of programming models are best for heterogeneous-ISA platforms, both in terms of developer productivity and performance. To the best of our knowledge, this is the first OS and compiler toolchain to enable application execution across heterogeneous-ISA architectures using a shared memory programming model.

Acknowledgments

This work is supported by the US Office of Naval Research under Contract N00014-12-1-0880.

References

[1] Barbalace, A., Sadini, M., Ansary, S., Jelesnianski, C., Ravichandran, A., Kendir, C., Murray, A., and Ravindran, B., "Popcorn: Bridging the Programmability Gap in Heterogeneous-ISA Platforms," *EuroSys 2015*, April 2015, Bordeaux, France.

http://ssrg.ece.vt.edu

Systems Software Research Group

VirginiaTech Invent the Future®

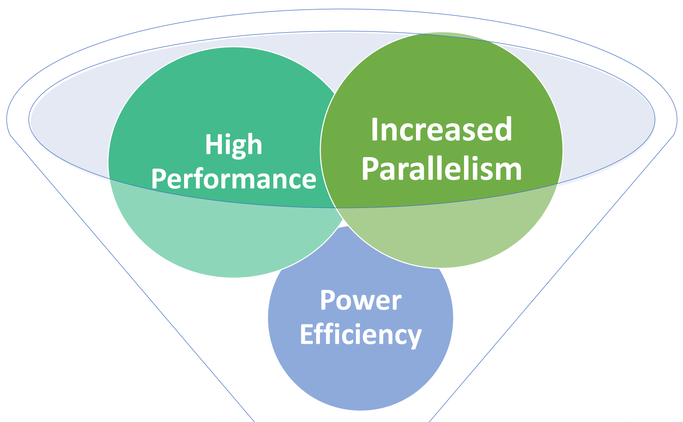
Compiler Support for Application Migration in Heterogeneous-ISA Platforms

Rob Lyerly (student, presenter), Christopher Jelesnianski (student), Antonio Barbalace and Binoy Ravindran
Systems Software Research Group, Bradley Department of Electrical and Computer Engineering, Virginia Tech, Virginia, USA
{rlyerly, bielsk1 antoniob, binoy}@vt.edu

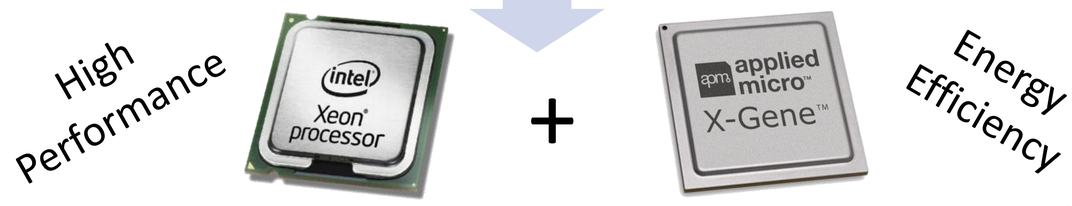
Supported by
DEPARTMENT OF THE NAVY · USN · ONR · Science & Technology

Emerging Hardware Trends

- Specialization of hardware
- Hardware evolving to be OS-capable and fully general purpose
- Integration of heterogeneous-ISA architectures on the same chip



Fully **OS-capable Het-ISA** platforms

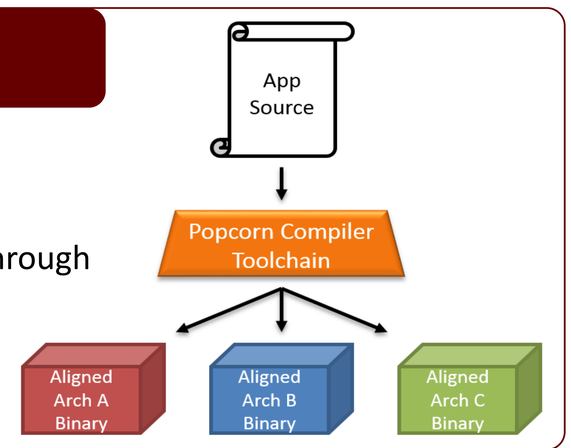


Goals

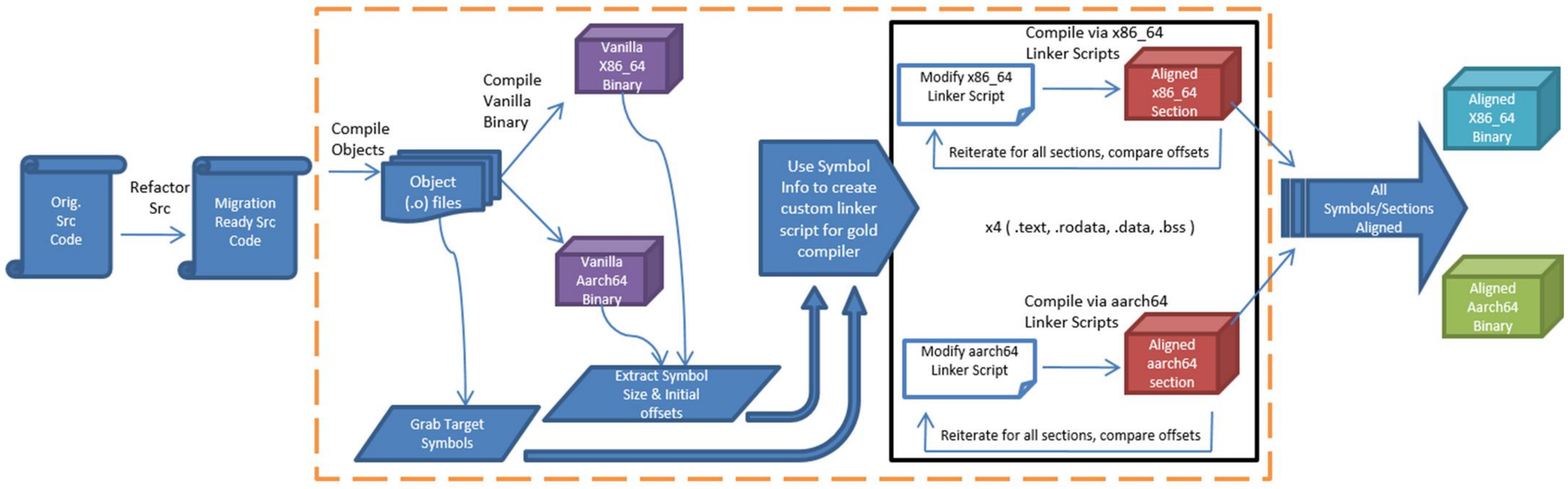
- Gain benefits of heterogeneous-ISA platforms using **existing programming models**
- Enable cross-arch **execution migration** of **arbitrary applications** with **minimal developer effort**
- Use **OS**, **runtime** & **compiler support** to handle arch (ISA, ABI, micro-arch) & data management
- Enable intelligent **mapping of applications to architectures** for performance, energy, etc.
- Built on **Popcorn Linux**, a replicated-kernel OS that provides inter-architecture migration & distributed shared memory (DSM) across heterogeneous-ISA platforms
  - Popcorn boots one Linux kernel per ISA island (e.g. 2 kernels for Xeon-Xeon Phi platform)
  - Kernels communicate via message passing to provide single OS interface to applications
  - Provides **transparency** to applications, **load-sharing** across architectures & allows applications to **exploit hardware asymmetries**

Multi-Binary Applications

- Generate binary per arch to leverage arch-specific optimizations
- Align global memory and function addresses
  - Popcorn handles inter-arch memory consistency through DSM
  - Functions become inter-arch **migration points**
    - Popcorn loads arch-specific version of function upon migration

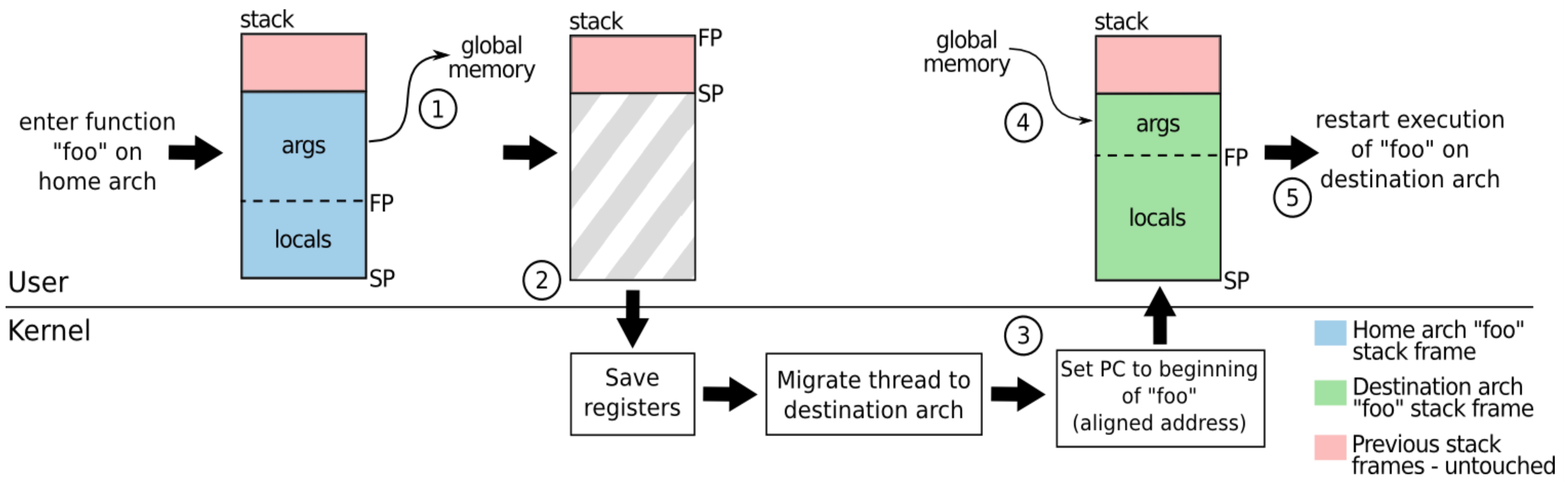


Linking



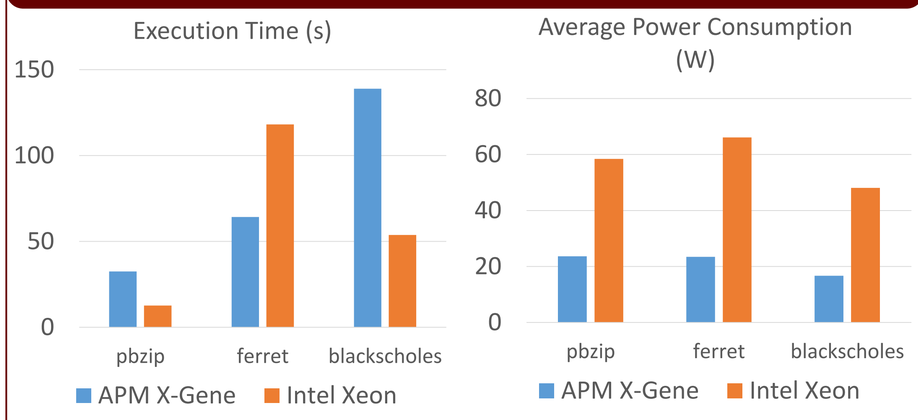
- Generate object files, extract application specific symbols, and sort according to type (.text, .rodata, .data, .bss)
  - Compile for each ISA without alignment
  - Align one section at a time
    - Gather alignment of every symbol for each ISA
    - Align section header & offsets
    - Recompile heterogeneous binary with modified linker-script
- Minimal alignment offset amount used to make a compact binary

Runtime Migration



- Migrate at **function boundaries** – avoid stack re-writing or implementing common ABI between architectures through logical separation of call stack
- When entering migratable function:
  1. Move function arguments from host arch stack frame into aligned global memory
  2. Pop host arch stack frame & migrate execution through Popcorn system call
  3. Resume execution at aligned function address on destination arch
  4. Set up destination arch registers & stack frame
  5. Restore arguments from global memory & execute function on new arch – Popcorn migrates application data on-demand through DSM
- When finished executing function on destination, migrate execution back through similar process & continue application execution on home arch
- Developers mark migratable functions w/ pragmas, application refactored using libtooling (clang/LLVM)

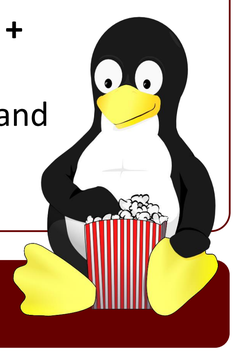
Performance & Power



- Applications experience different execution times on different architectures
- Seek to develop policies for migrating parts or whole applications to achieve high performance, energy efficiency, performance/watt, etc.

Conclusions

- Using proposed toolchain and Popcorn Linux, applications can **migrate between heterogeneous-ISA architectures**
- Developers mark migration points, **toolchain + Popcorn handles mechanics of migration**
- **Future work** – investigate migration policies and lifting the limitation of migrating only at function boundaries



www.popcornlinux.org