Enabling Legacy Applications on Heterogeneous Platforms

Michela Becchi, Hari Cadambi and Srimat Chakradhar
NEC Laboratories America, Princeton, NJ, USA.

NEC Laboratories America
Relentless passion for innovation

Context

Legacy applications that are retargeted to heterogeneous (CPU + multiple accelerators) systems using library interposing

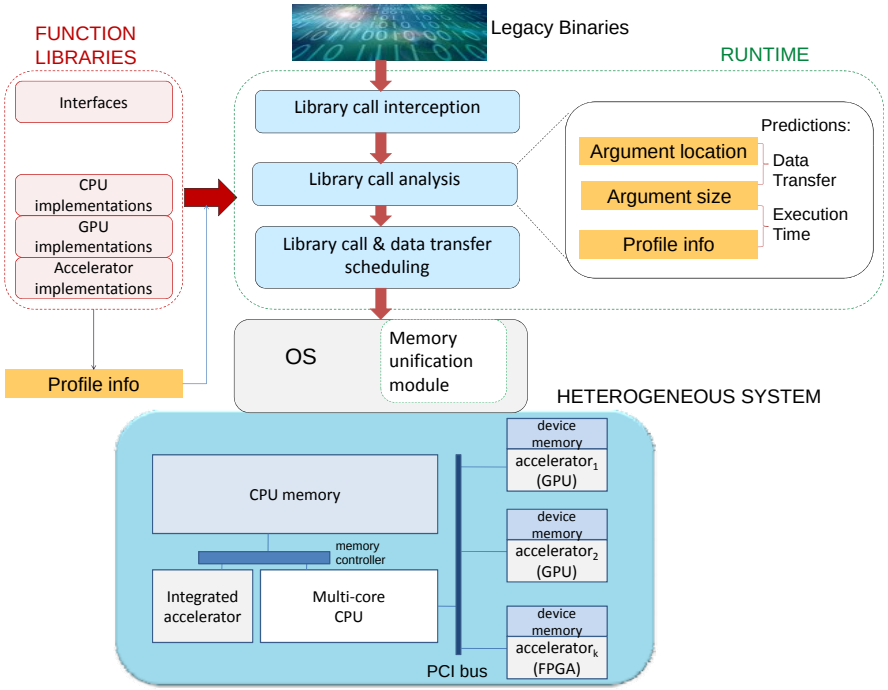
Goals

- Schedule kernels to multiple accelerators considering both computation and data
- Improve performance by reducing data movement.

Strategy

- Profile and estimate each kernel's performance on each accelerator.
- **Lazy data updates: l**eave data where it was last modified.
- Schedule next computation taking into consideration (i) estimated performance of the accelerator and (ii) cost to move data to the accelerator.

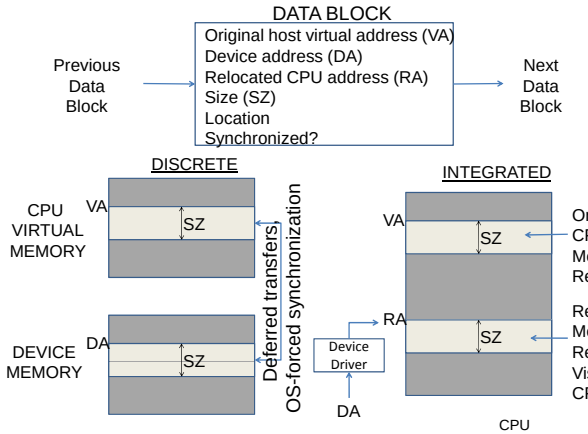
System



EXAMPLE

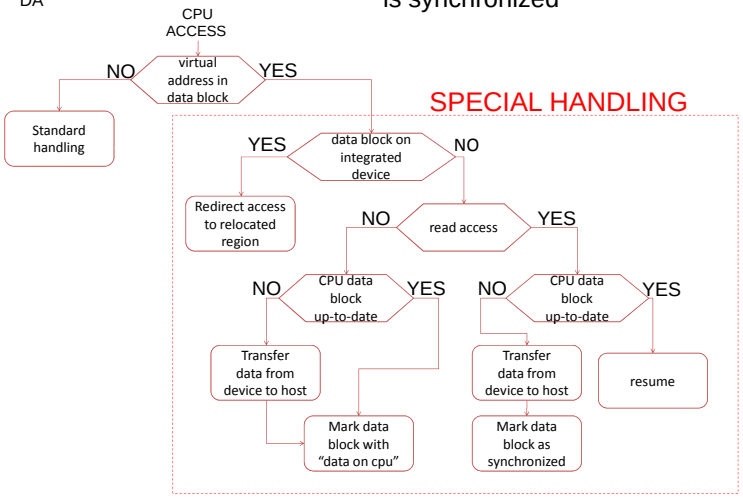
matmul(in A, in B, out C, sizes…)	Model says GPU performance is better. So transfer A, B, C and schedule on GPU. Leave C on GPU.
matmul(in C, in D, out E, sizes…)	Model says CPU performance is better, but C too big to transfer back, so schedule on GPU. Leave C, E on GPU.
func(C, E)	CPU accesses C, E which are stale since latest copies are on GPU. Must force synchronization now for coherence.

Enforcing coherence using OS



- Associate each process with a set of data blocks
- Similar to Linux associating each process with a set of memory regions
- Each data block stores accelerator mapped addresses, sizes of the memory regions, location information and whether data is synchronized

- Trigger page faults when CPU accesses an address that is also mapped to accelerator
- Modified page fault handler: forces synchronization if required by transferring data
- If no sync required, does nothing



Results