Offload Engines

Jim Harris
jimharris@freebsd.org
james.r.harris@intel.com
FreeBSD DevSummit 2012

Agenda

- Motivation for offload framework
- Proposed scope
- Requirements
- Integration with busdma(?)
- Plans
- Next Steps

Motivation for offload framework

- Lack of a generic API for “data movement”
  - Common interface to accelerate operations (using HW or optimized SW) for memcopy, XOR, P+Q, dual/multicast, etc.
  - Common interface for drivers to register as “data movers”

Proposed scope

- Focus on operations not currently supported in opencrypto
- Allow for not only hardware offload, but also arch-specific optimized routines

Requirements for offload framework

- Efficiency
  - Difficult to quantify, but specifically want to avoid:
    - Multiple SG list traversals
    - Using bounce buffers for HW offload
- Transparently use SW routines if HW engines not available
- Chaining
  - DMA from A to B, from C to D, from E to F, then…
  - XOR from B, D, F to G

Requirements for offload framework (cont.)

- Callback mechanism to notify client of completion
- At least rudimentary support for all CPU architectures
  - Some archs may always revert to basic C routines for performing operations
- Non-blocking
  - Don’t put thread to sleep if waiting for resource

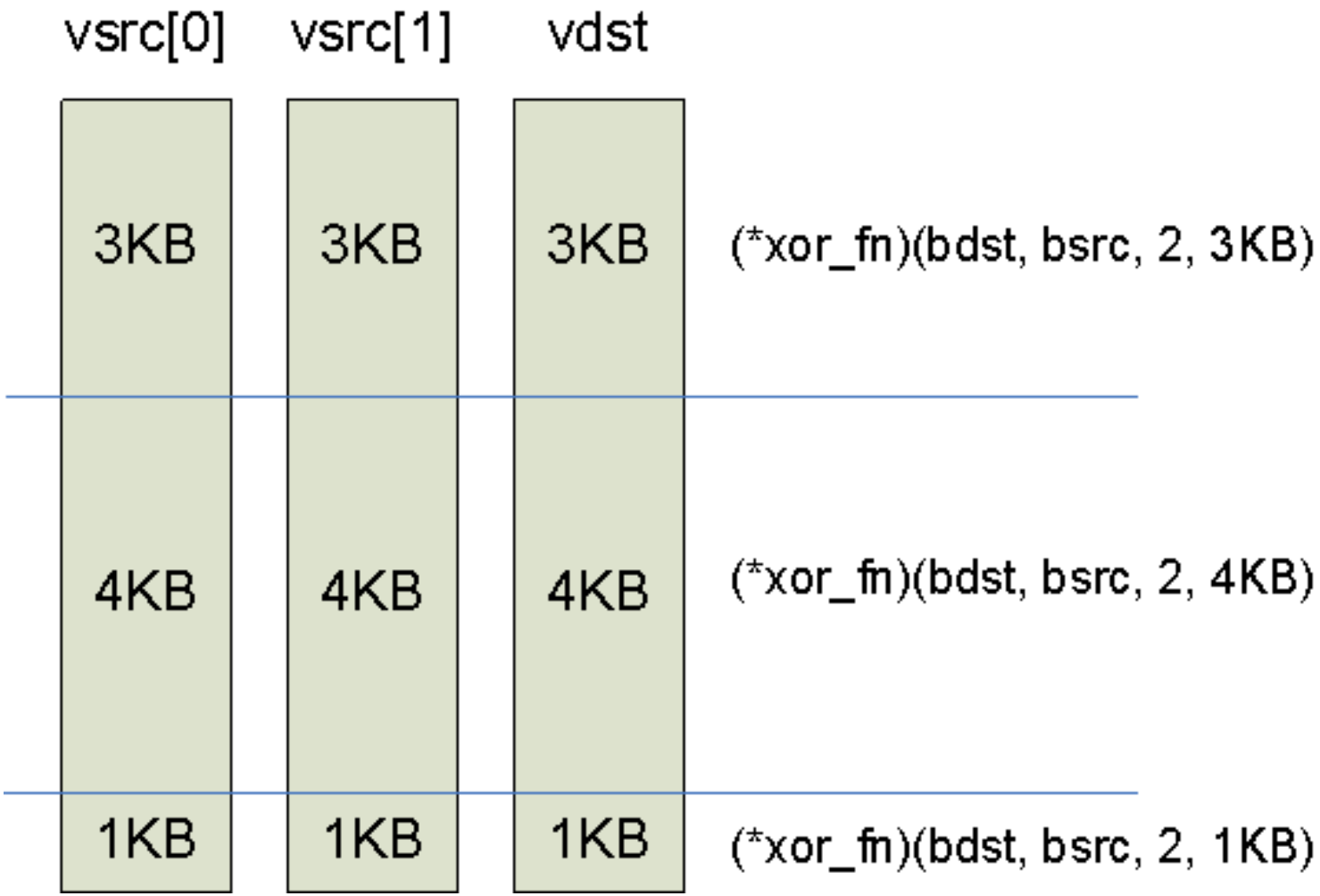
Requirements for offload framework (cont.)

- Allocate offload resources on demand
  - i.e. avoid allocating descriptor memory if there are no clients for the offload framework
- sysctl hooks
  - Force operations <= X bytes to SW
    - (X == 0 -> disable HW offload)

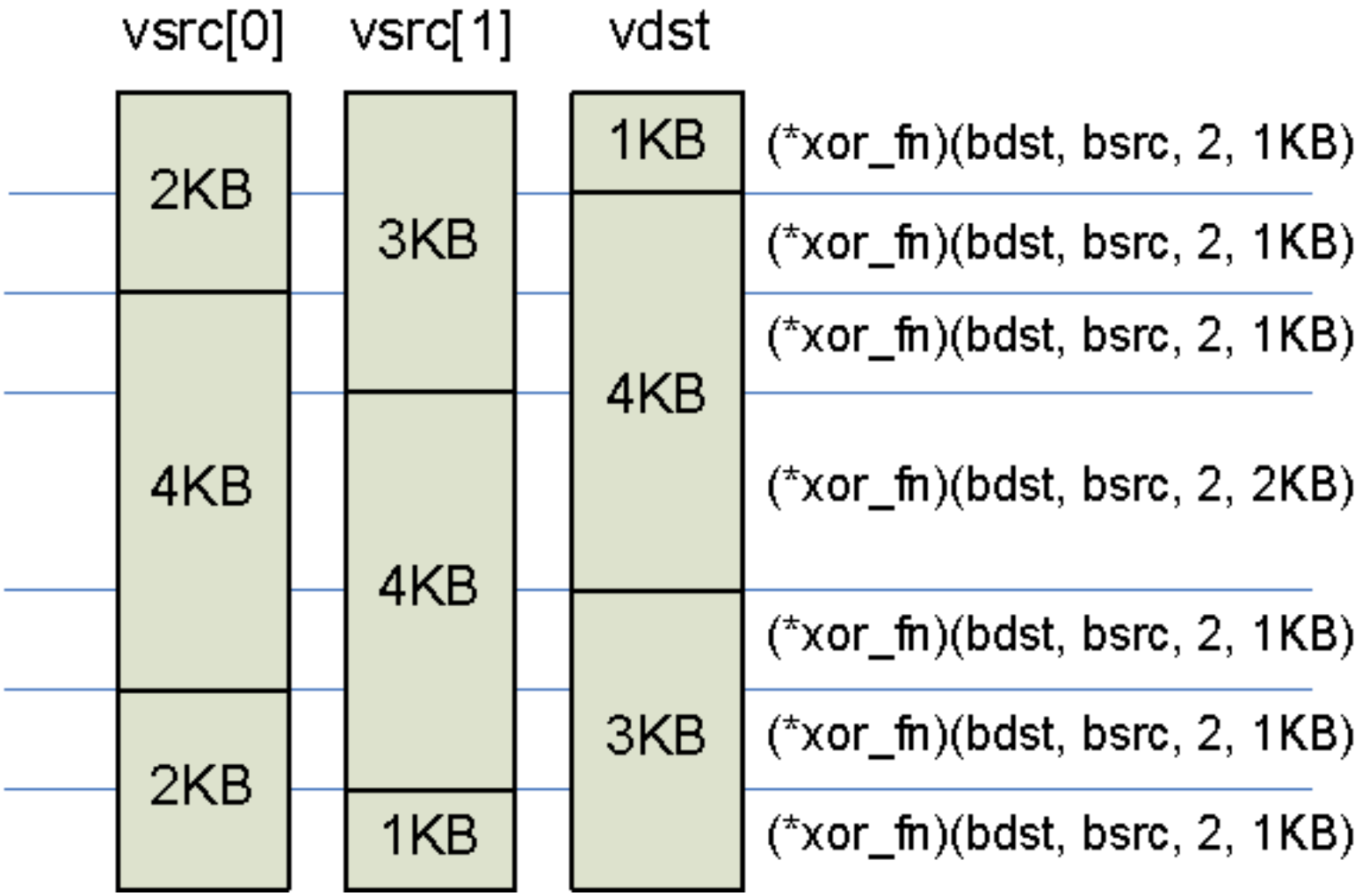
Integration with busdma

- Drivers can use DMA tags to describe addressing restrictions
- Offload framework can use DMA tags to avoid bounce buffering
- Offload framework can walk virtual addresses and perform/initiate operations in line
  - Rather than building SGLs first, then traversing SGLs

Aligned XOR Example
2 sources, 8KB length



Non-aligned XOR Example
2 sources, 8KB length



Plans

- ioat(4) driver under development
  - Also known as “Crystal Beach”
  - Supports DMA, XOR, P+Q, memfill on certain Intel Xeon processors
  - Tracking to EOQ2 for completion
    - Driver only, no offload framework
- Continue working on offload framework in parallel (but not full time)

Next Steps

- Incorporate today's input into a more concrete set of APIs for review
- Prototype framework for x86/ioat
  - Help on plumbing for ZFS RAIDz
- Areas of help needed
  - Support for HW offload for additional architectures
  - Support for additional HW offload engines