Interrupt Queues

May 13, 2011

John Baldwin
jhb@FreeBSD.org

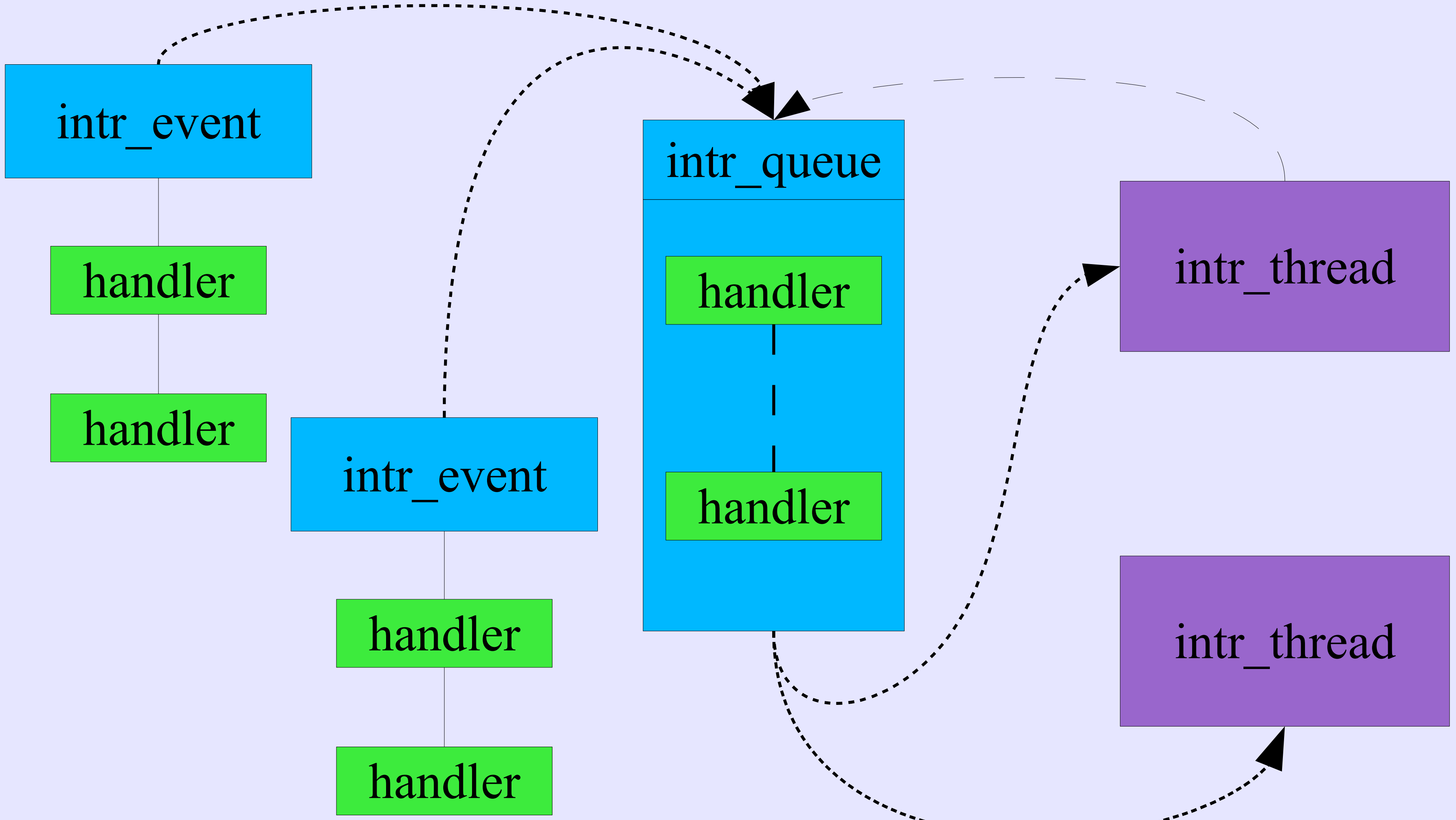
Why Change Interrupt Threads?

- Interrupt filters are broken
  - Design choice to only schedule one thread breaks in practice
- Other features desired
  - Cooperative scheduling among handlers
  - Ability to bind interrupt sources to CPUs and/or threads independently
  - Safely remove filters

Interrupt Queues?

- Interrupt handlers use a state machine
- Handlers are tied to a queue rather than a thread
- Queues are associated with a pool of threads, but threads are only associated with a single queue
- Interrupt threads do not iterate over a single IRQ's handler list, but drain a queue

Overview of Queues



New Features

- Filter can schedule multiple handlers
- Handlers can reschedule themselves
    - Can be used to implement cooperative polling in drivers
- Greater flexibility with IRQs and threads

Current Implementation

- Preserves existing behavior: each IRQ has dedicated queue with one thread
- Cannot create alternative topologies currently
- //depot/user/jhb/intr/... in p4