Fastpath networking applications on manycore SoCs



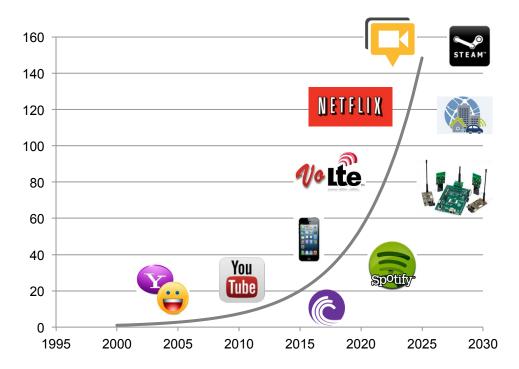
Ola Liljedahl, Oct 29 2013

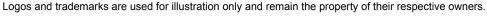


What Is Happening

Super-linear growth in:

- Number of users
- Number of connected devices
- Number of over the top (OTT) applications
- Number of protocols (RFC's)
- Bandwidth usage
- Power consumption of network infrastructure





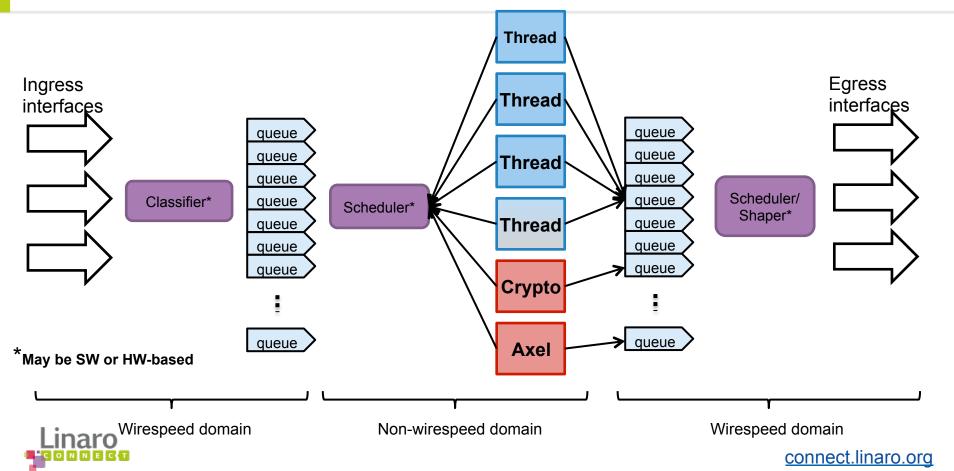


So the Requirements Change

- Need aggregate capacity of multiple processors
- Need hardware accelerations for PPA-efficient execution of repetitive work
- Need flexibility of software and programmable hardware
- Need familiar programming environments with robust tools
- Need portability between different implementations
- Need high abstraction level
- Need efficient support for virtualisation

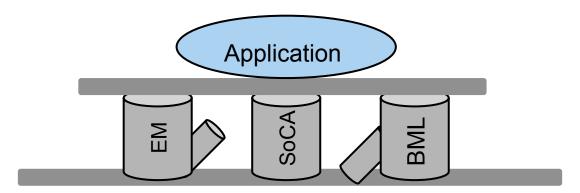


Example Logical View of Packet Processing Flow



The Pillars of OpenDataPlane

- Event Machine
 - Event based IO object handling paradigm and application interface
- SoC Abstraction
 - Portable API's for access to HW/SoC resources
- "Bare Metal" Linux
 - Minimal overhead and deterministic execution in Linux user space



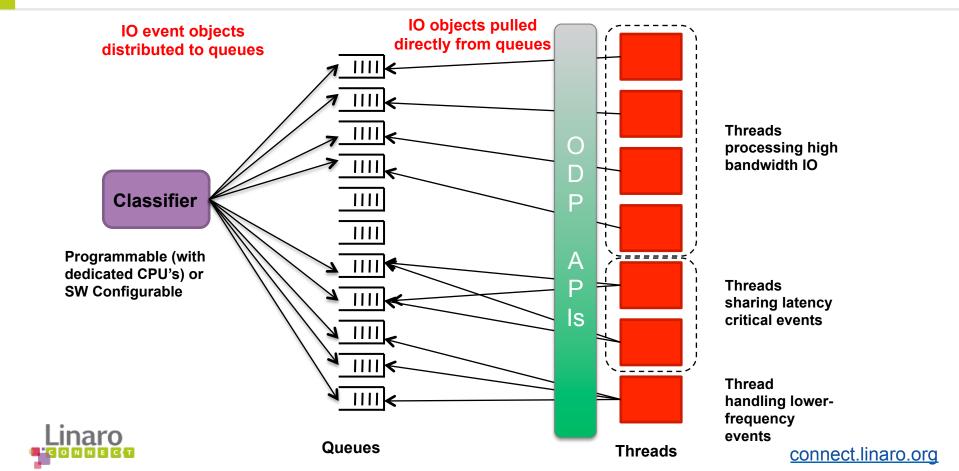


Application Interface

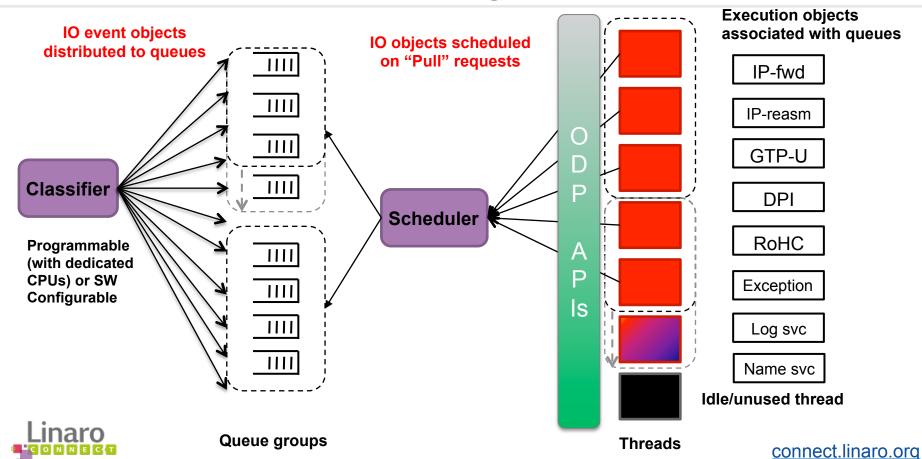
- ODP adopts Event Machine, originally developed by Nokia Solu-Networks
 - Open-ended architecture with reusable concepts
 - C-based API's and implementations
- Event based programming model abstraction for handling IO
 - Logical queue based interface
 - Supports packet flows, physical and virtual network interfaces, accelerators, SW endpoints
 - Events represent different types of data that should be processed: packets, timers, baseband data, HW notifications, SW messages...
- Supports scheduler based IO model
 - Enables scheduling of IO events using different algorithms and knowledge of work in progress
 - Support for both HW and SW based schedulers
- Supports different IO load balancing approaches
 - Enables implicit synchronisation and mutual exclusion between threads
- Proven, already half a dozen HW-specific implementations



APIs for IO Load Balancing – "Push" model



APIs for IO Load Balancing – "Pull" model



SoC Abstraction

- Portable API's for accelerator access, timer management & delivery...
- Per-operation access functions + queues & events for completion notification
- Example: crypto (as proposed)

```
int odp_crypto_operation(
    struct odp_crypto_session *sesssion,
    odp_packet_buf_desc *pkt_buf_desc, // Data to process
    const struct odp_data_range *cipher_data,
    const struct odp_data_range *hash_data,
    unsigned hash_result_offset,
    caddr_t override_iv_ptr,
    unsigned num_notif, const em_notif_t *notif_tbl // Events and queue-id to use for notification when operation is complete
```

- Supports both synchronous and asynchronous operation
- Supports both processor/ISA and loosely coupled accelerator implementations
- Might define an (optional) synchronous blocking function as well as this is a common usage
- Compression, DPI, timer management potential candidates





"Bare Metal" Linux

- Need to dedicate some cores for dataplane applications
- No TLB misses (huge pages for code and data)
- No context switches (one worker thread per HW-thread/core)
- No thread migration (core affinity and isolation)
- ✗ No ticks (NOHZ/dynticks)
- No interrupts
- No kernel background tasks
- Direct HW access from user space
- Can still
 - Launch, supervise and kill applications, dump core
 - Debug and profile applications
 - Perform system calls from the application





More about Linaro: http://www.linaro.org/about/

More about Linaro engineering: http://www.linaro.org/engineering/

How to join: http://www.linaro.org/about/how-to-join

Linaro members: www.linaro.org/members