DPDK

DATA PLANE DEVELOPMENT KIT

Recent Power Management Enhancements in DPDK

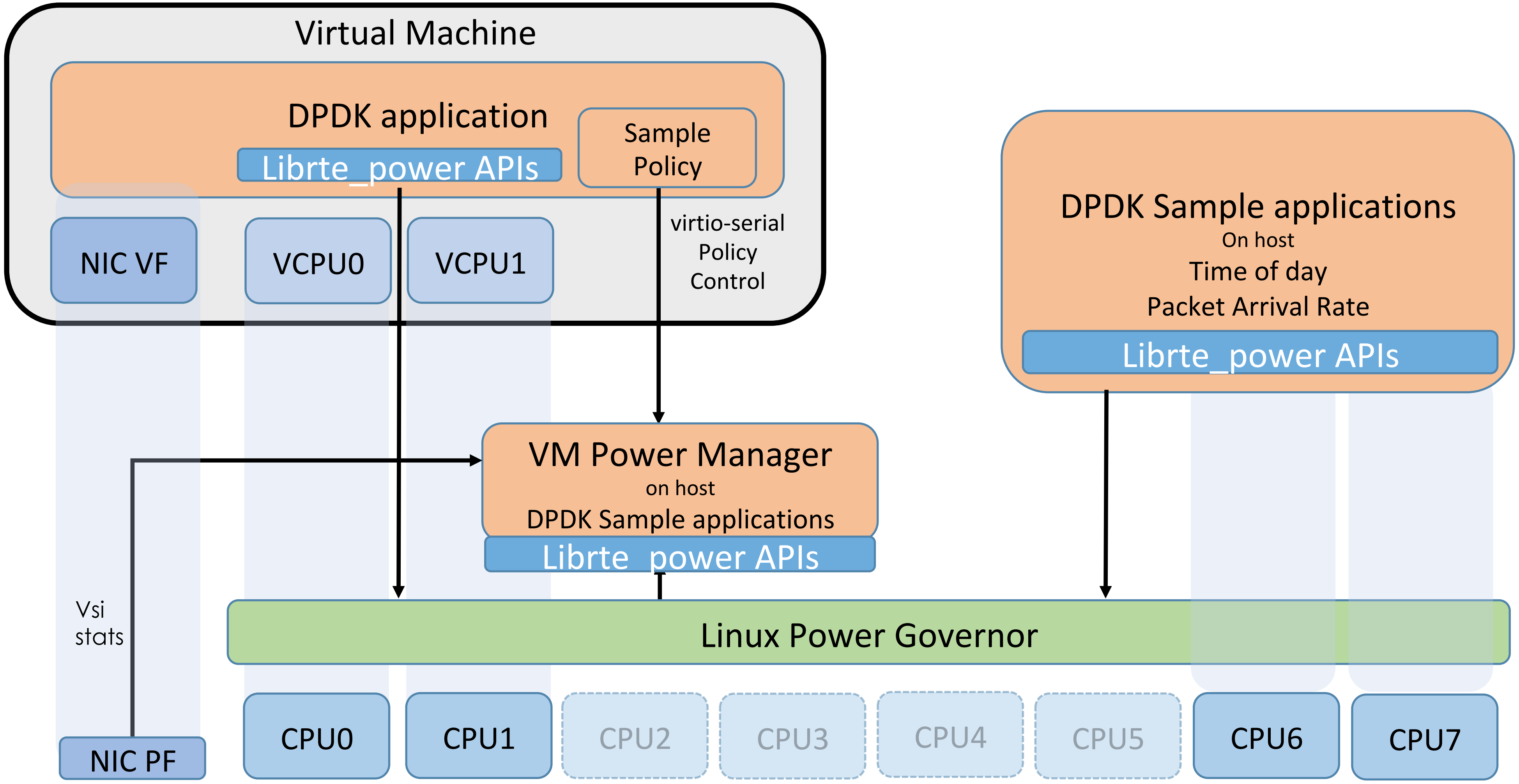
DAVE HUNT, CHRIS MACNAMARA, LIANG MA, RADU NICOLAU

Updates Since Last Time

- Quick reminder…. feature review

- Updates & Discussions to follow
  - How Busy Am I? 100% …. Hmmmm
  - New Methods to Trigger Power State Changes
    - Load conditions in a 100% Polling environment
    - Out of Band Energy Efficiency determination for 100% Polling DPDK PMDs
  - Power Policies for Containers

Existing DPDK Power Capabilities



Many use cases, support for direct control, virtualized architecture

Existing DPDK Power Features

Challenge / Problem	DPDK Solution / Status
L3fwd power using C states	Sample app
Traffic always running, always on cores	Added core Frequency State APIs
Increase performance on key cores when busy or overloaded	Added Turbo Boost APIs in rte_power.h
Virtualized Software Architecture: Long latency of a VM detect, waste of monitoring and changing state, move to policy based control	Inband: New SW Arch for policy control via virtio-serial
Match CPU power to network load (Scale down when not busy, turbo when busy)	Sample app: Time of day
Fast scale up when burst arrives	Sample app: Packet arrival rate (NIC stats)

Librte_power APIs and Sample Apps

New DPDK Features Since Last Time!

Challenge / Problem	DPDK Solution / Status
Pin DPDK threads/lcores to high priority cores	Pinning relevant workloads to Turbo Cores
App Agnostic mechanism to detect when DPDK is 100% polling and no packets or work	Sample code: Branch prediction ratio used as trigger to detect idle -> modify power
Mechanism to determine load (Experimental branch)	Empty polling trend analysis and trigger to modify power (e.g. how busy)
Power Policies for Containers	New FIFO interface to Power Manager that accepts policies via JSON

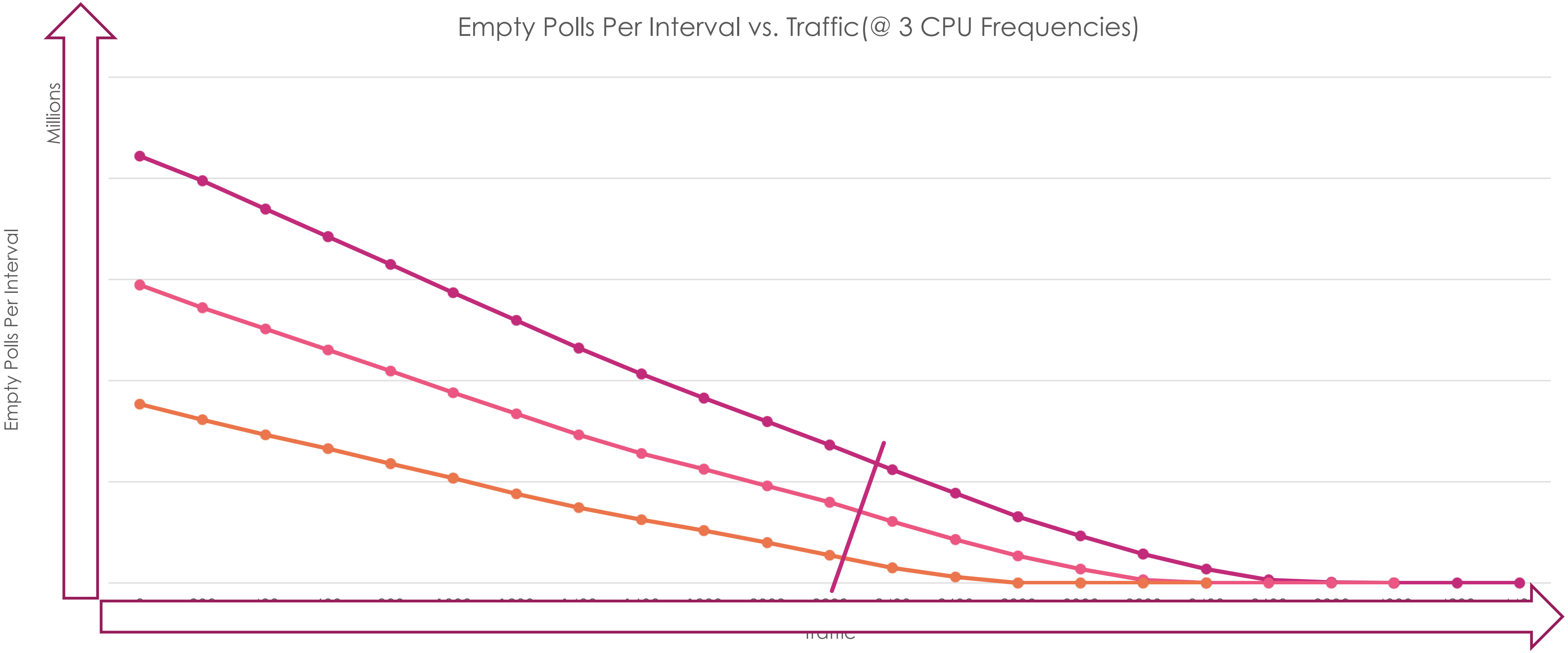
New triggers and capabilities enabling new use cases

DPDK

DATA PLANE DEVELOPMENT KIT

Determining Load

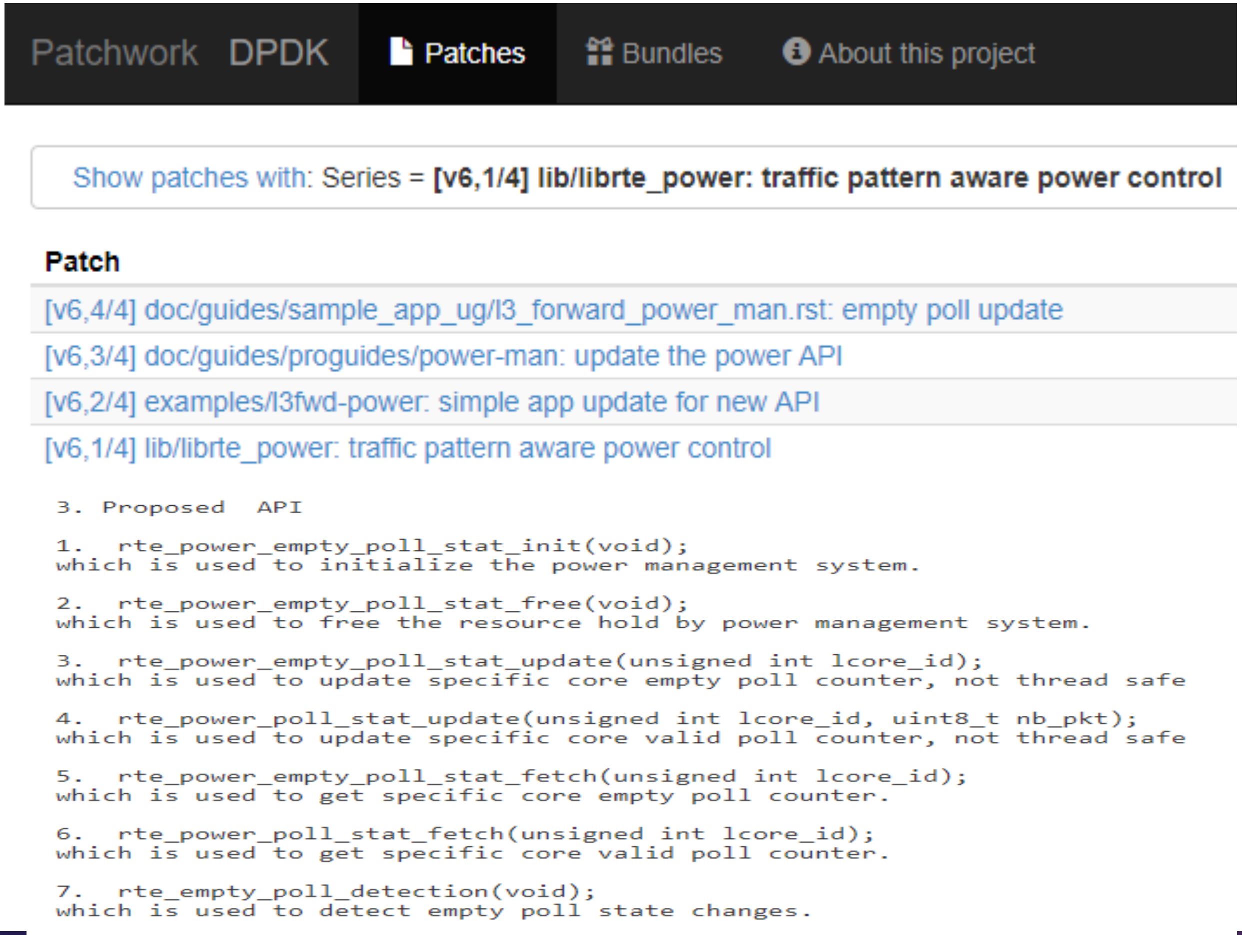
How busy am I … determining load



Using empty polls useful for load detection & action trigger

Pushed Patches To Support This (Traffic Aware)

- Submitted to mailing list
  http://patches.dpdk.org/project/dpdk/list/?series=1143
- API marked as experimental



Show patches with: Series = **[v6,1/4] lib/librte_power: traffic pattern aware power control**

Patch
[v6,4/4] doc/guides/sample_app_ug/l3_forward_power_man.rst: empty poll update
[v6,3/4] doc/guides/proguides/power-man: update the power API
[v6,2/4] examples/l3fwd-power: simple app update for new API
[v6,1/4] lib/librte_power: traffic pattern aware power control

```
3. Proposed  API

1.  rte_power_empty_poll_stat_init(void);
which is used to initialize the power management system.

2.  rte_power_empty_poll_stat_free(void);
which is used to free the resource hold by power management system.

3.  rte_power_empty_poll_stat_update(unsigned int lcore_id);
which is used to update specific core empty poll counter, not thread safe

4.  rte_power_poll_stat_update(unsigned int lcore_id, uint8_t nb_pkt);
which is used to update specific core valid poll counter, not thread safe

5.  rte_power_empty_poll_stat_fetch(unsigned int lcore_id);
which is used to get specific core empty poll counter.

6.  rte_power_poll_stat_fetch(unsigned int lcore_id);
which is used to get specific core valid poll counter.

7.  rte_empty_poll_detection(void);
which is used to detect empty poll state changes.
```

@Init
How many polls can we do
Set thresholds (idle/busy)

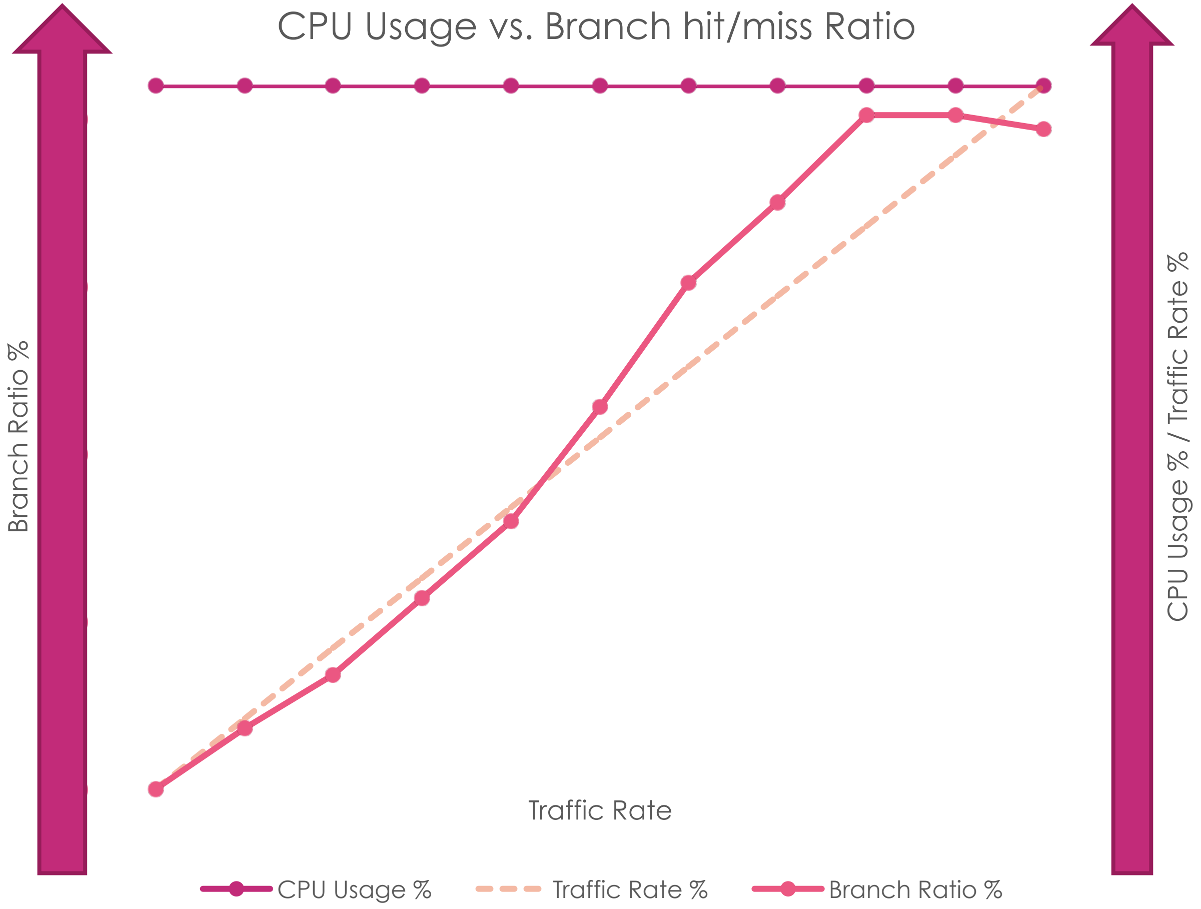
@run
Count empty polls
Check against threshold
Am I busy?

@run
Adjust State, snooze or run faster

Out of Band Energy Efficiency

Poll Loop Work Rate Detection (PMD Load%)

- CPU Load is always 100% for DPDK PMD Poll Loops
- Actual workload may be zero (processing zero packets)
- Use the ratio between Branch Hits and Branch Misses
- Ratio is low when tight code loop (empty polling), and significantly is higher when processing packets (due to larger code path)
- *Almost* linear with traffic rate



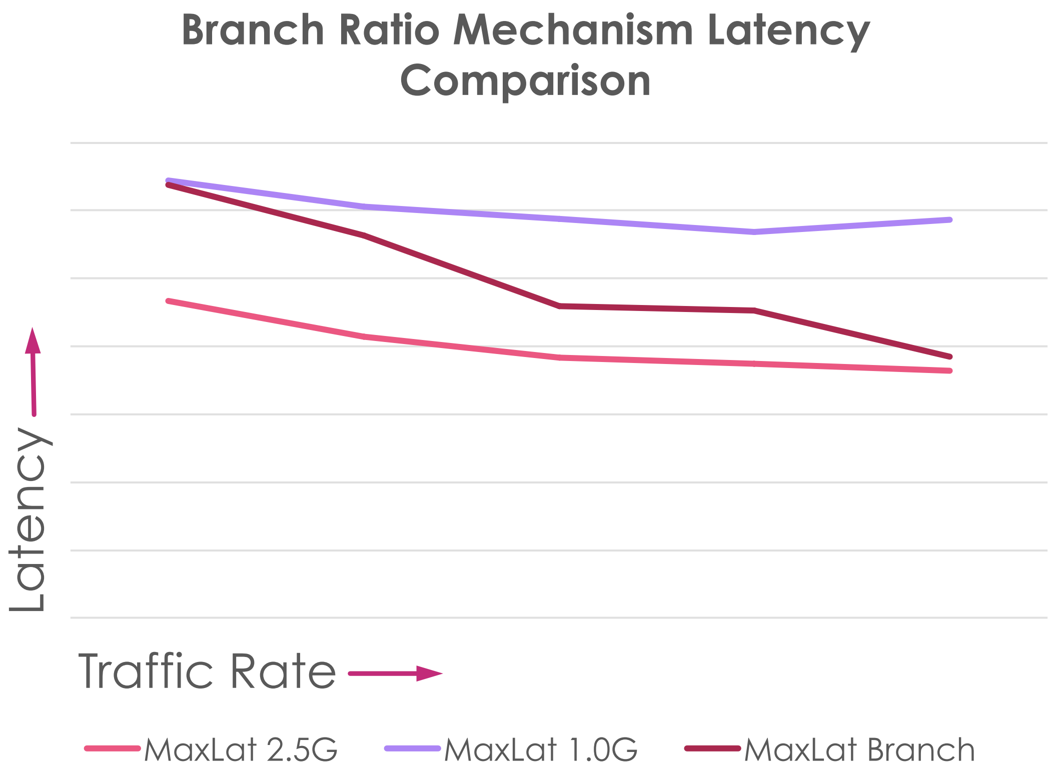
Application Agnostic Idle Detection using Branching

Latency Comparison

- Using out-of-band branch ratio mechanism of power management (Merged in 18.08)

- Three measurements shown
  - 2.5GHz fixed core frequency
  - 1.0GHz fixed core frequency
  - 1.0GHz – 2.5GHz variable base on branch ratop

- Branch Ratio mechanism reading core counters every 100uS



Branch Ratio Latency as expected

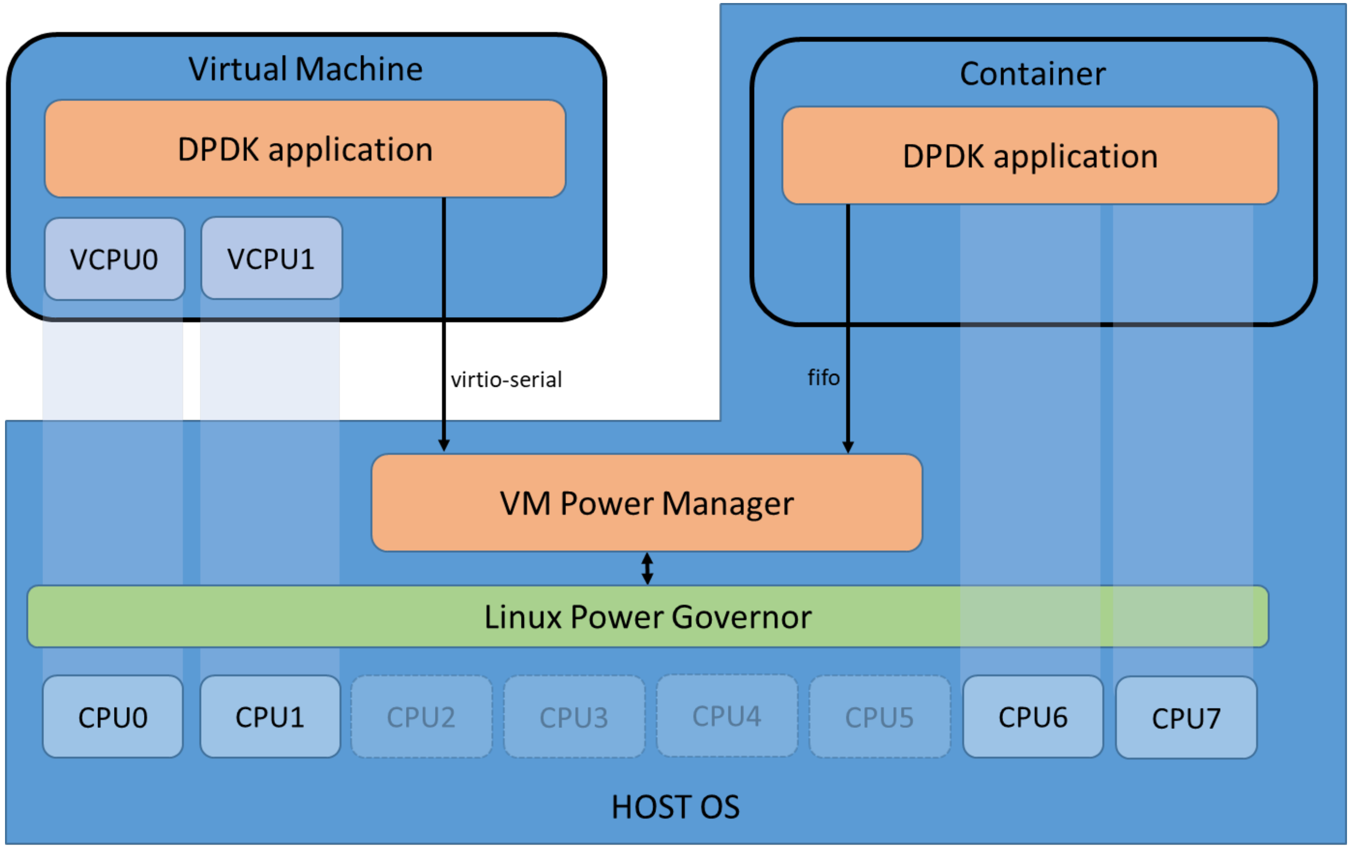
Pushed Patches To Support This (Branch Ratio)

- Applied in DPDK 18.08

Power Policies for Containers

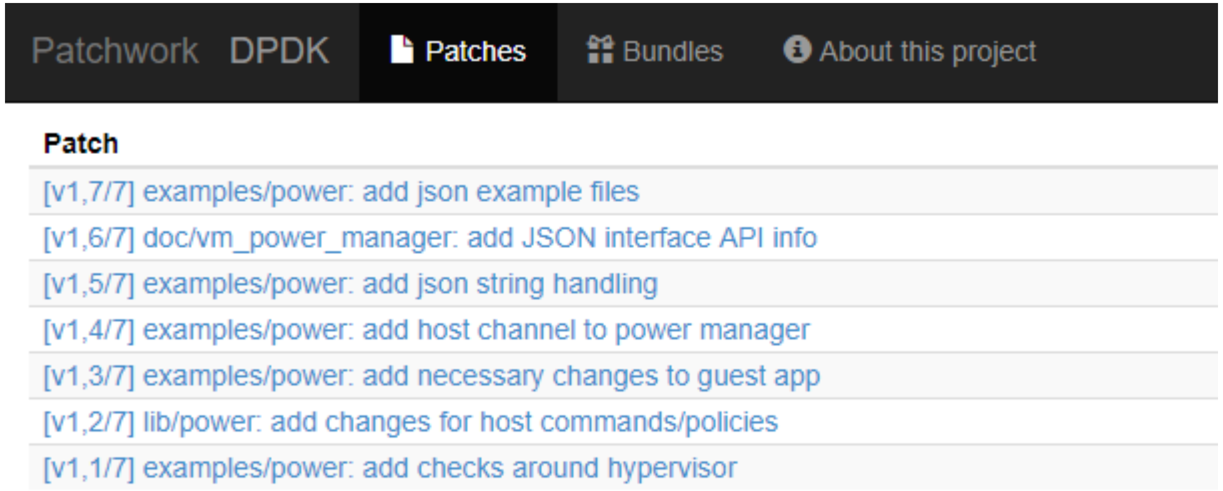
JSON interface via FIFO for Power Manager

- Current mechanism to send policies from VMs to Power Manager via virtop-serial
- New patch-set adds additional interface into Power Manager via file system FIFO
- Handles existing power commands, max, min, up, down, etc.
- Handles power polices, similar to VM virtio-serial channels
- Can be used by any application with access to the FIFO in the Host OS
  - Host Applications
  - Container Applications



Pushed Patches To Support This (Policies for Containers)

- Submitted to mailing list for 18.11
- http://patches.dpdk.org/project/dpdk/list/?series=1109



Patch
[v1,7/7] examples/power: add json example files
[v1,6/7] doc/vm_power_manager: add JSON interface API info
[v1,5/7] examples/power: add json string handling
[v1,4/7] examples/power: add host channel to power manager
[v1,3/7] examples/power: add necessary changes to guest app
[v1,2/7] lib/power: add changes for host commands/policies
[v1,1/7] examples/power: add checks around hypervisor

Thank You

Chris MacNamara (chris.macnamara@intel.com)
Dave Hunt (david.hunt@intel.com)

Liang Ma <liang.j.ma@intel.com>

Acknowledgements

Mike Glynn, John Geary, Stephen Byrne, Tim O’Driscoll, Walt Gilmore