

Hardware accelerating Linux network functions

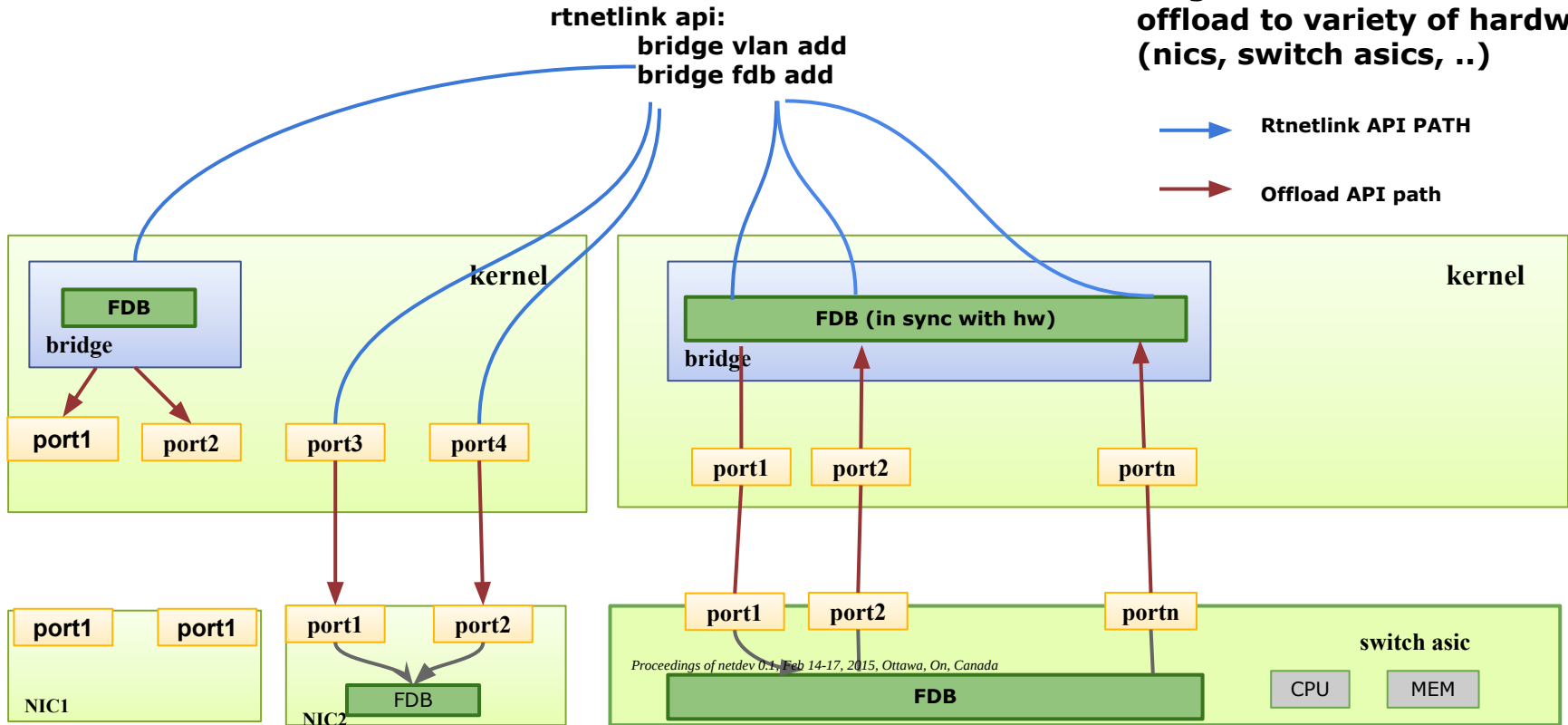
Roopa Prabhu, Wilson Kok

Agenda

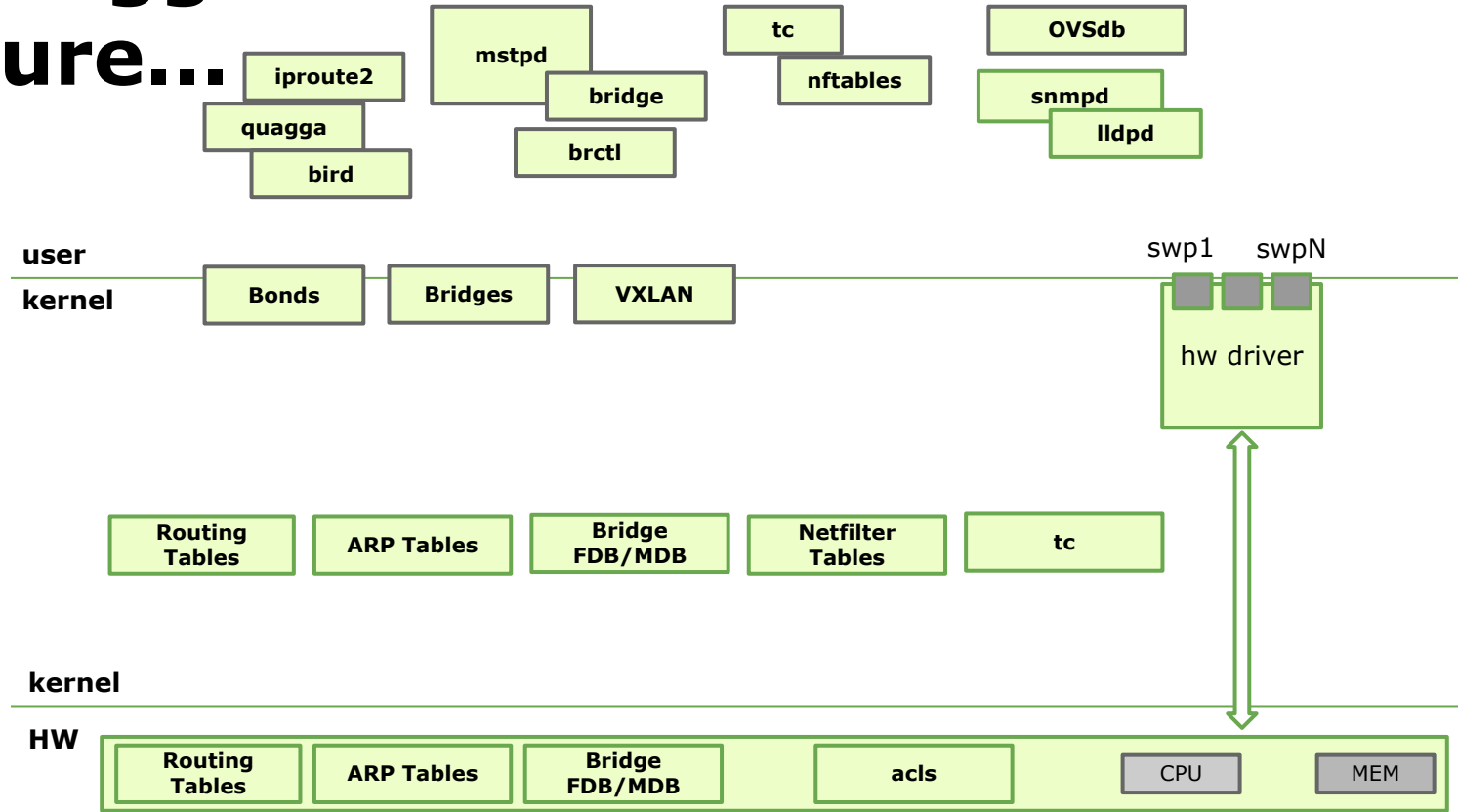
- Recap: offload models, offload drivers
- Introduction to switch asic hardware
- L2 offload to switch ASIC
 - Mac Learning, ageing
 - stp handling
 - igmp snooping
 - vxlan
- L3 offload to switch ASIC

Offload models ...

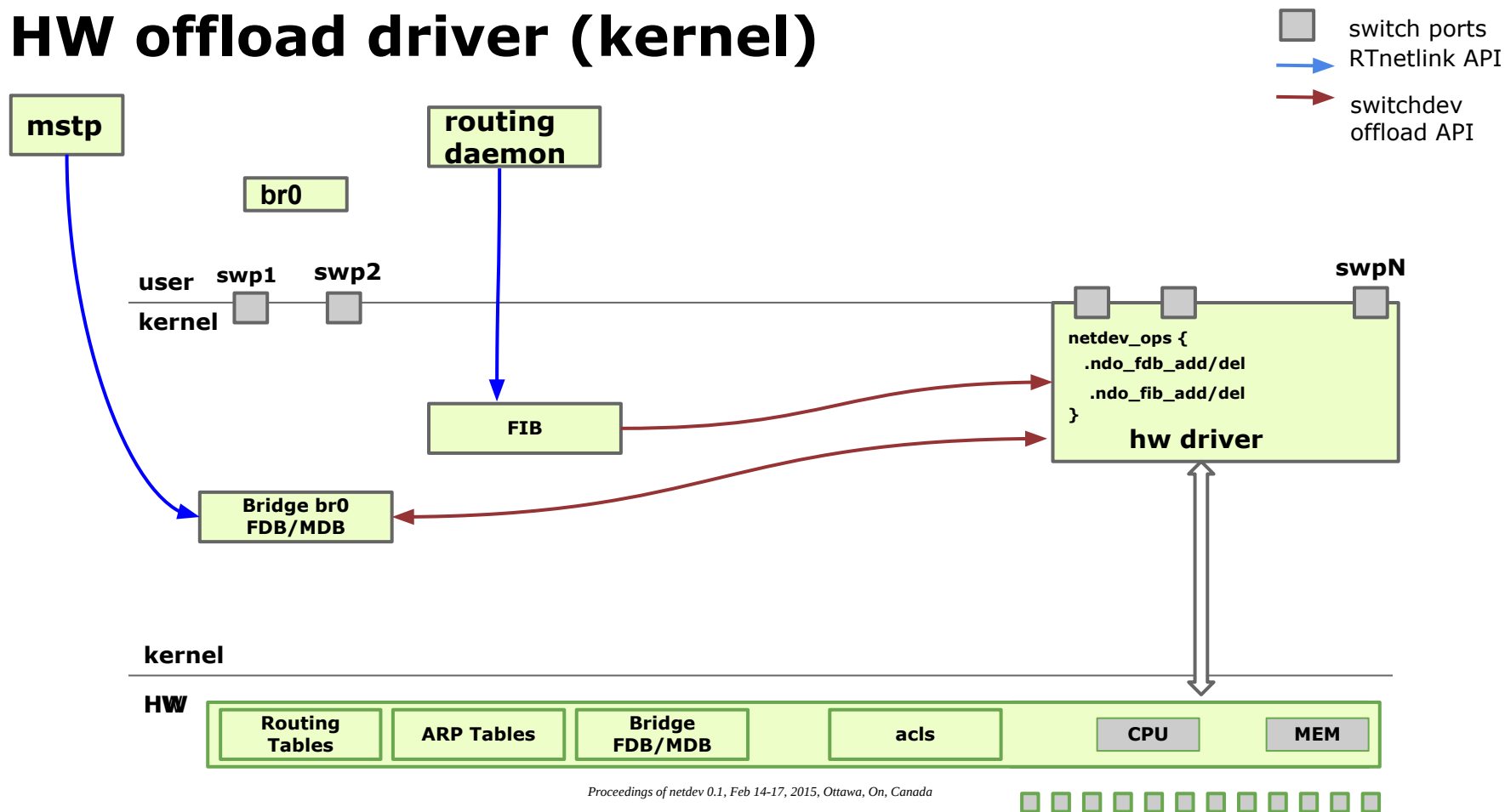
- Single consistent netlink based UAPI
- Single kernel offload API to offload to variety of hardware (nics, switch asics, ..)



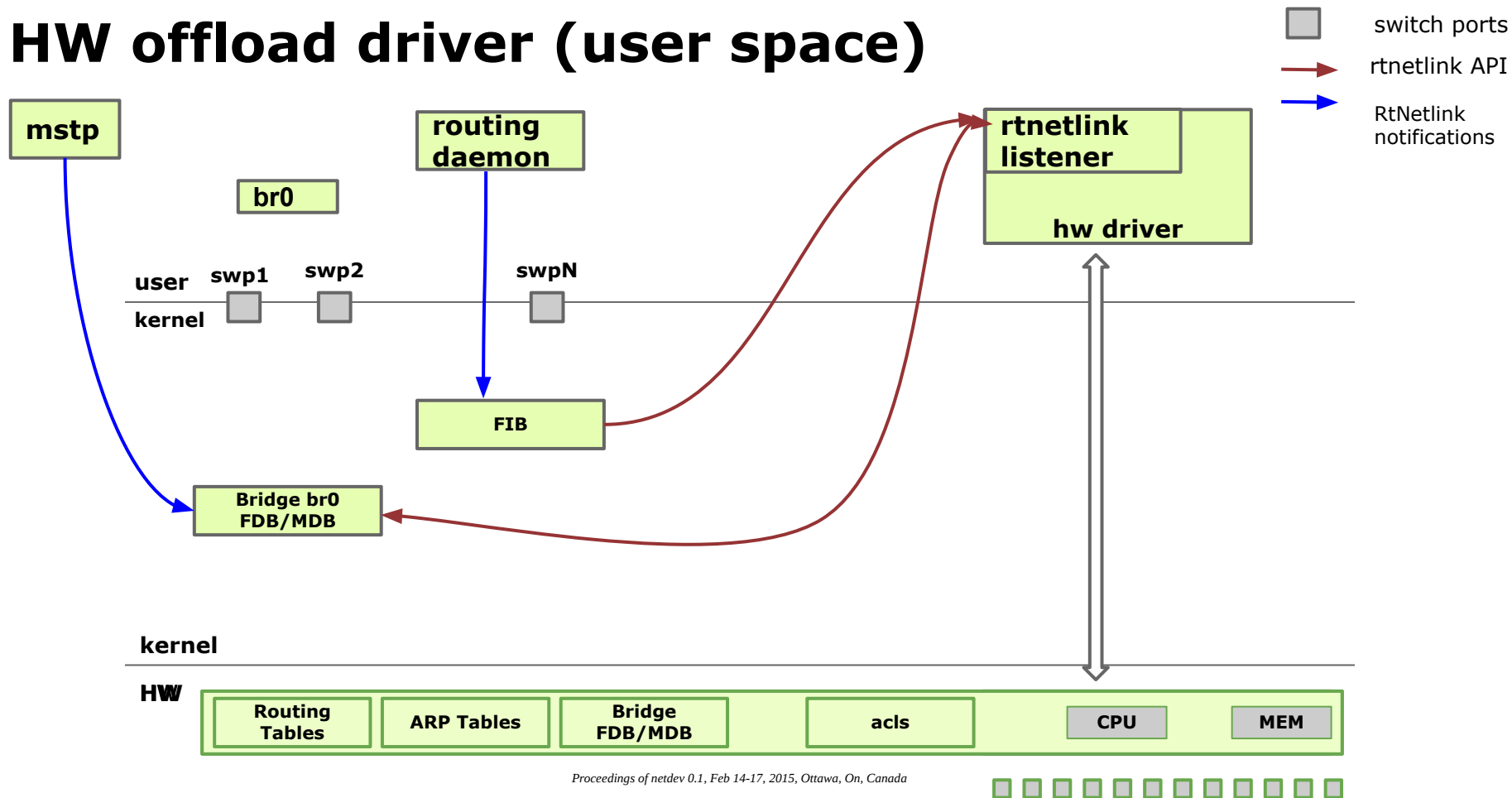
The bigger picture...



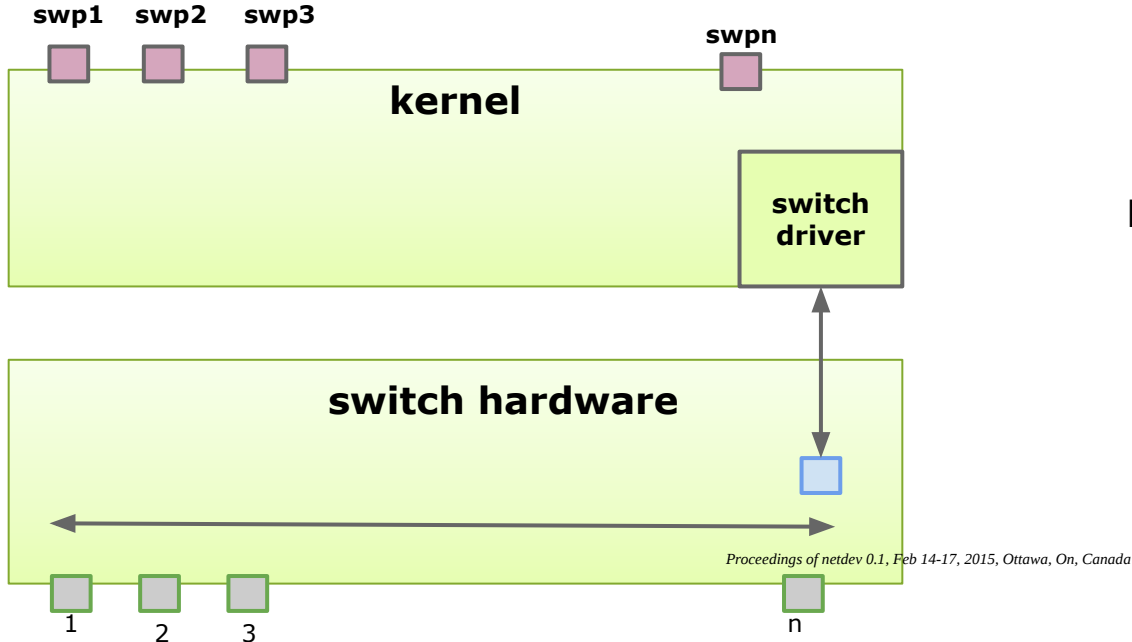
HW offload driver (kernel)



HW offload driver (user space)

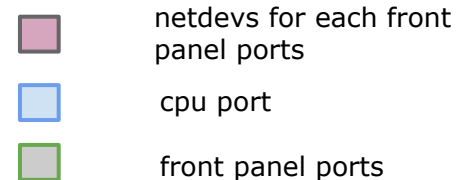


switch hardware



switch driver:

- **Creates netdevs for front panel ports**
- **Port netdevs only see traffic forwarded to the CPU port**
- **Sets hardware offload flag `NETIF_F_HW_SWITCH_OFFLOAD` on netdevs**



ip link show switch ports

```
# ip link show
```

```
1: lo: <LOOPBACK> mtu 16436 qdisc noqueue state  
DOWN mode DEFAULT
```

```
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:  
00:00
```

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP>  
mtu 1500 qdisc mq state UP mode DEFAULT qlen 1000
```

```
link/ether 00:e0:ec:27:4e:b6 brd ff:ff:ff:ff:ff:ff
```

```
3: swp1: <BROADCAST,MULTICAST,UP,LOWER_UP>  
mtu 1500 qdisc pfifo_fast state UP mode DEFAULT  
qlen 500
```

```
link/ether 44:38:39:00:27:ac brd ff:ff:ff:ff:ff:ff
```

```
4: swp2: <BROADCAST,MULTICAST> mtu 9000 qdisc  
pfifo_fast state DOWN mode DEFAULT qlen 500
```

```
link/ether 00:e0:ec:27:4e:b8 brd ff:ff:ff:ff:ff:ff
```

```
55: swp53: <BROADCAST,MULTICAST> mtu 1500  
qdisc noop state DOWN mode DEFAULT qlen 500
```

```
link/ether 00:e0:ec:27:4e:f7 brd ff:ff:ff:ff:ff:ff
```

```
56: swp54s0: <BROADCAST,MULTICAST> mtu 1500  
qdisc noop state DOWN mode DEFAULT qlen 500
```

```
link/ether 00:e0:ec:27:4e:fb brd ff:ff:ff:ff:ff:ff
```

```
57: swp54s1: <BROADCAST,MULTICAST> mtu 1500  
qdisc noop state DOWN mode DEFAULT qlen 500
```

```
link/ether 00:e0:ec:27:4e:fc brd ff:ff:ff:ff:ff:ff
```

```
58: swp54s2: <BROADCAST,MULTICAST> mtu 1500  
qdisc noop state DOWN mode DEFAULT qlen 500
```

```
link/ether 00:e0:ec:27:4e:fd brd ff:ff:ff:ff:ff:ff
```

```
59: swp54s3: <BROADCAST,MULTICAST> mtu 1500  
qdisc noop state DOWN mode DEFAULT qlen 500
```

```
link/ether 00:e0:ec:27:4e:fe brd ff:ff:ff:ff:ff:ff
```

```
[snip]
```



switch ports



management port

ethtool on switch port

\$ethtool swp1

Settings for swp1:

Supported ports: [FIBRE]

**Supported link modes: 1000baseT/Full
10000baseT/Full**

**Supported pause frame use: Symmetric
Receive-only**

Supports auto-negotiation: Yes

Advertised link modes: 1000baseT/Full

Advertised pause frame use: No

Advertised auto-negotiation: No

Speed: 10000Mb/s

Duplex: Full

Port: FIBRE

PHYAD: 0

Transceiver: external

Auto-negotiation: off

Current message level: 0x00000000

(0)

Link detected: yes



Creating a hardware accelerated Linux bridge device

```
# ip link add br0 type bridge
```

```
# ip link set dev swp1 master br0
```

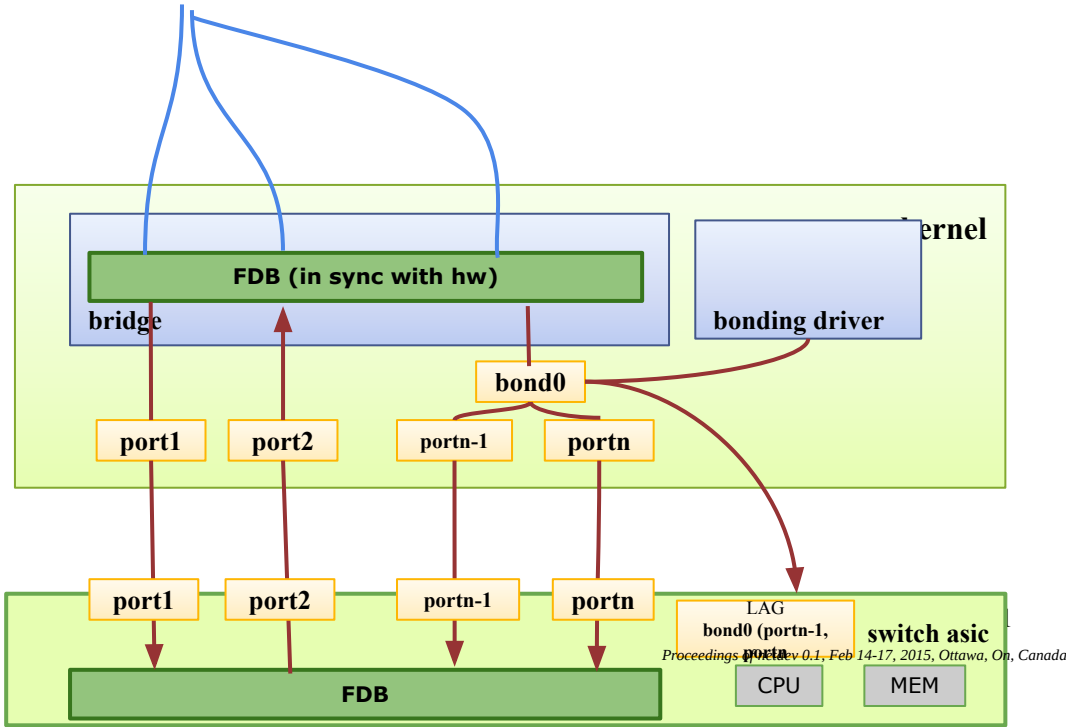
```
# ip link set dev swp2 master br0
```

```
# bridge vlan add vid 10-20 dev swp1
```

```
# bridge vlan add vid 20-30 dev swp2
```

Bonds as bridge ports

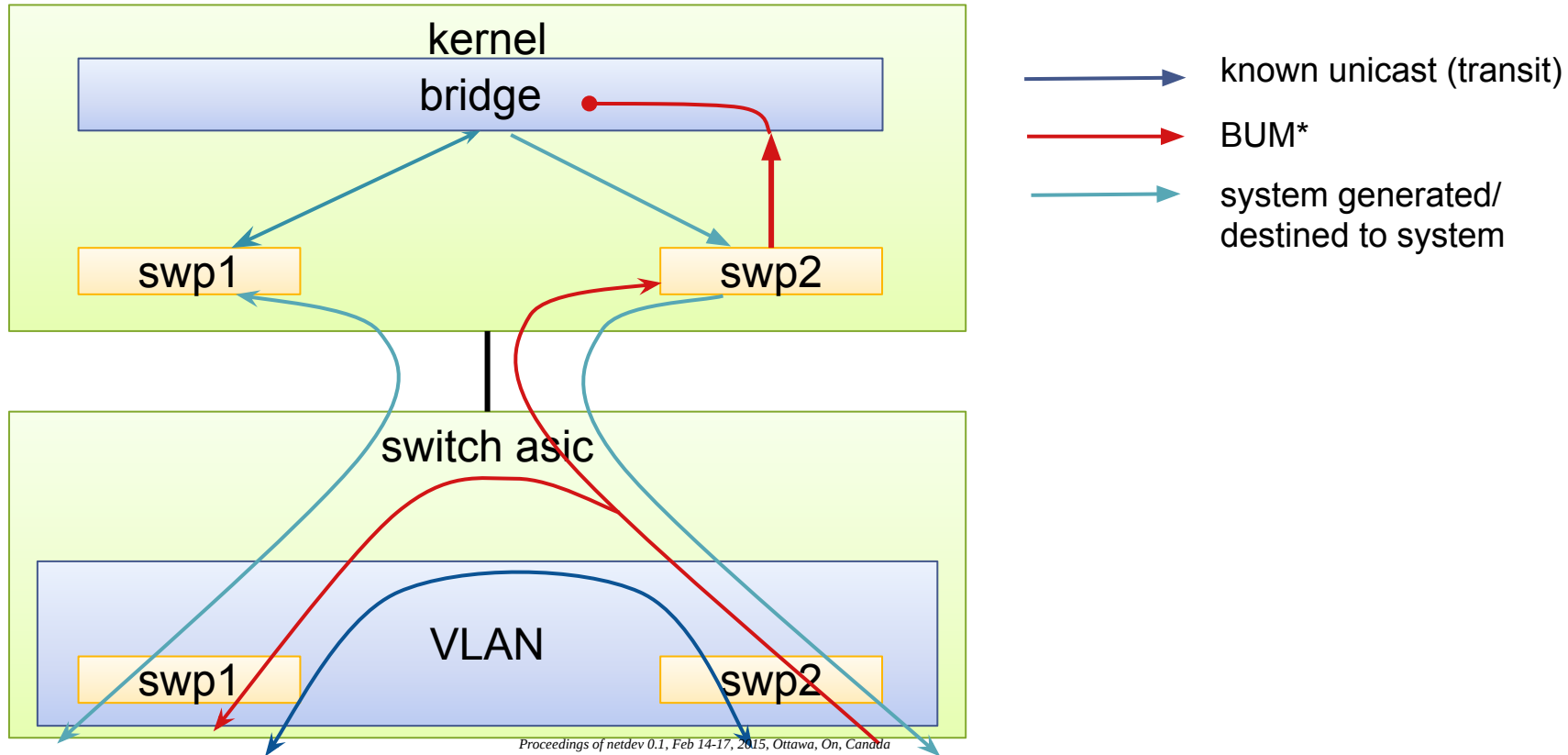
rtnetlink api:
bridge vlan add
bridge fdb add



- **switch ASICS support Link aggregation**
- **bonding driver LAG config is offloaded to the switch ASIC**
- **fdb and vlan offloads go through the bonding driver**

→ rtnetlink API
→ switchdev offload API

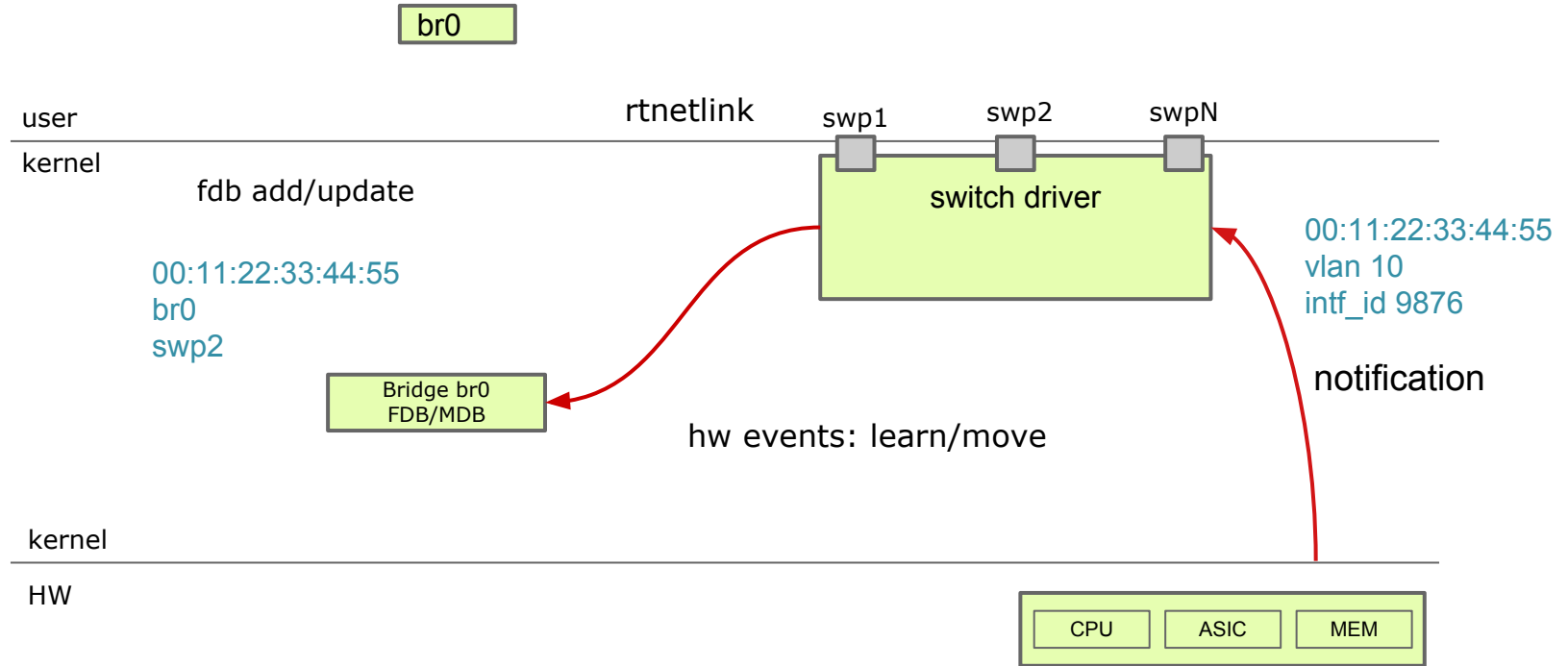
Bridging hardware offload: packet path



Bridging hardware offload: packet path

- Known unicast traffic not destined to system is forwarded only in hardware
- BUM traffic is forwarded in hardware plus a copy MAY be sent to kernel
- BUM traffic in kernel should not be forwarded again (duplicate copies from hardware and software)

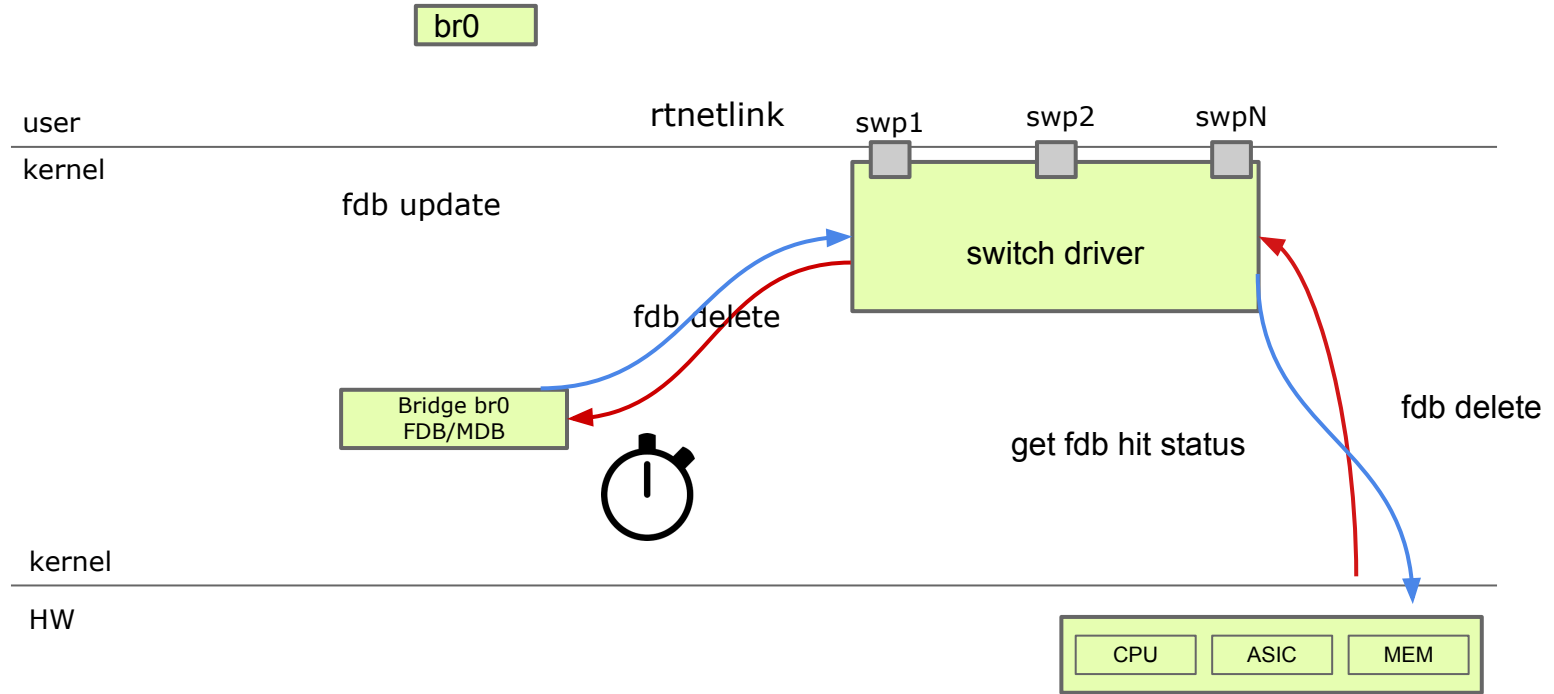
Bridging hardware offload: fdb learn



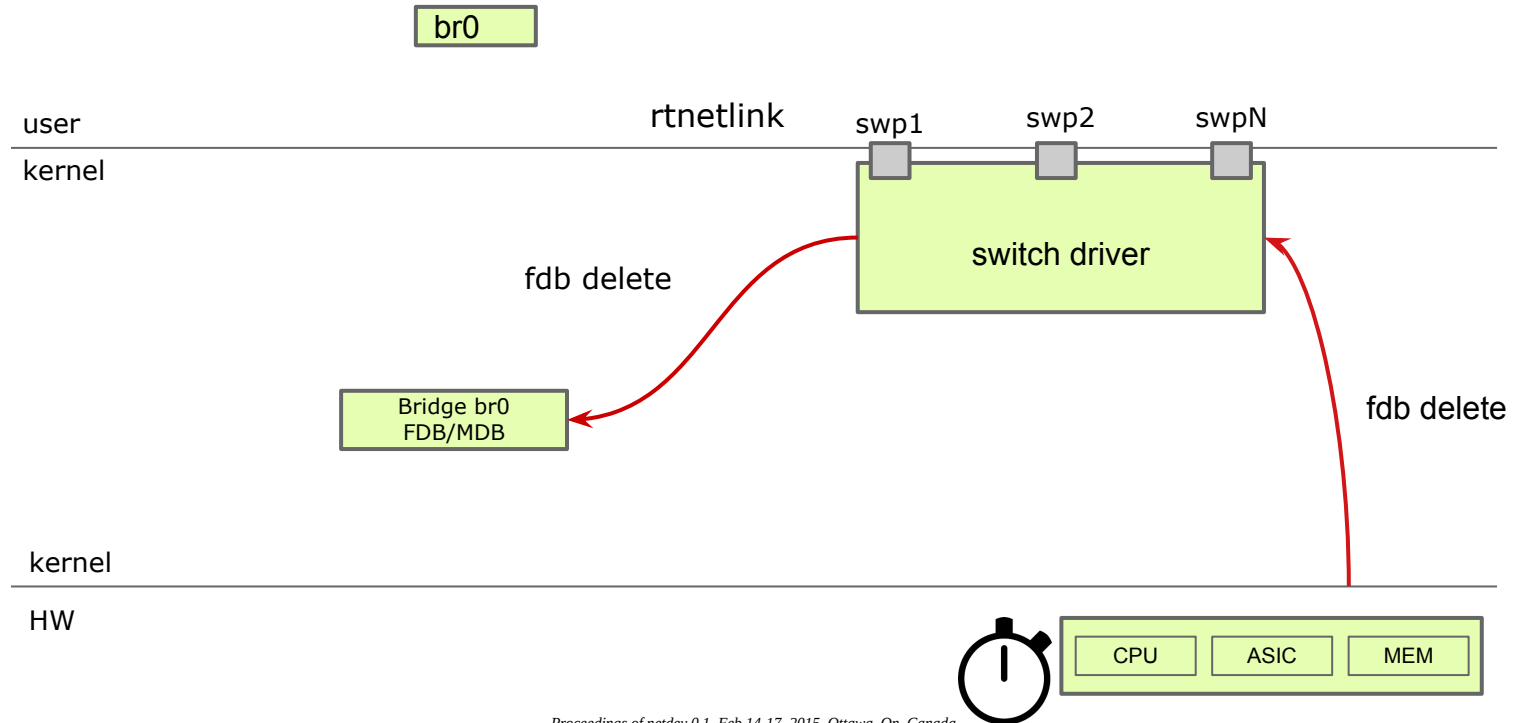
Bridging hardware offload: learning in HW

- Turn off learning in bridge driver
- switch driver listens to learn notifications from hardware
- converts hardware interface id and vlan to kernel ifindex of bridge port (and vlan) and bridge
- sends netlink fdb update to kernel (userspace driver) or calls bridge driver learn sync switchdev API (kernel driver)

Bridging hardware offload: kernel ageing



Bridging hardware offload: hardware ageing



Bridging hardware offload: ageing

Bridge driver very seldom sees packets with hardware offload. FDB age is not up to date.

Hardware ageing

- bridge driver should not do ageing if hardware is doing it
- fdb show will need to get age from hardware during 'show', or need periodic age update from switch driver

Kernel ageing

- definitely need periodic age update from switch driver

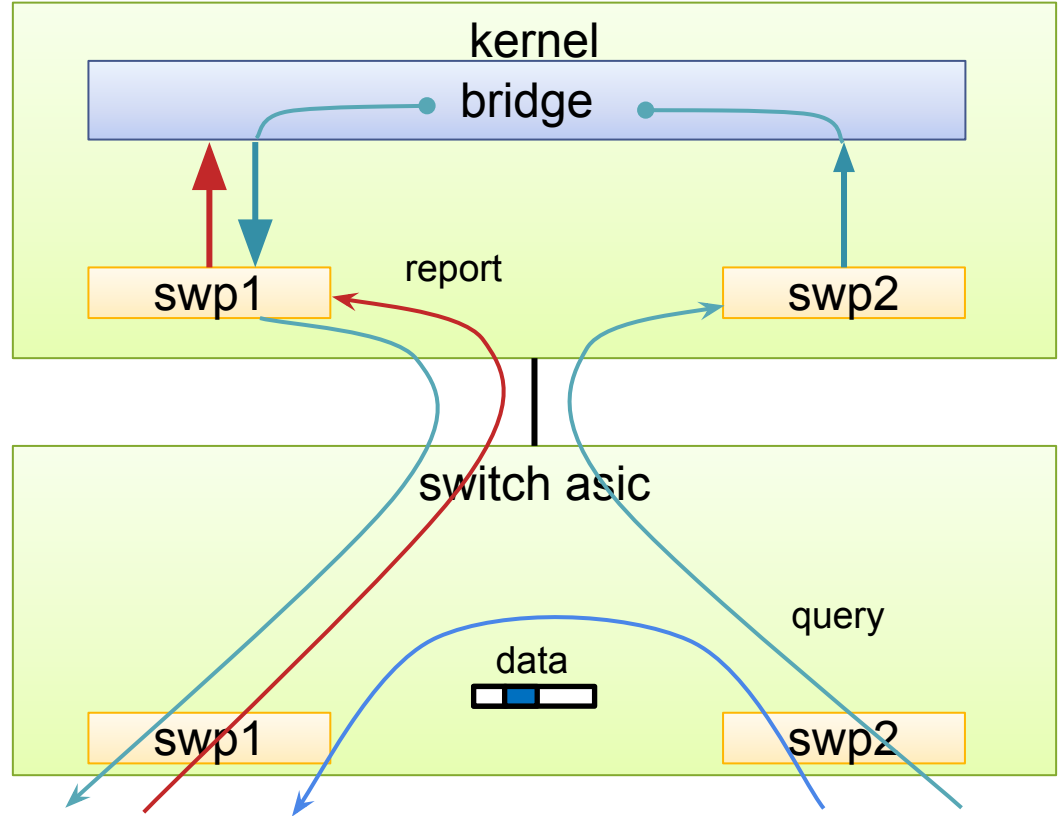
STP offload

STP

- bridge driver maintains STP states (either kernel STP or userspace STP)
- bridge driver communicates STP states to switch driver using switchdev offload API
- OR a switch driver in userspace can listen to STP state notifications to update HW state

IGMP snooping offload

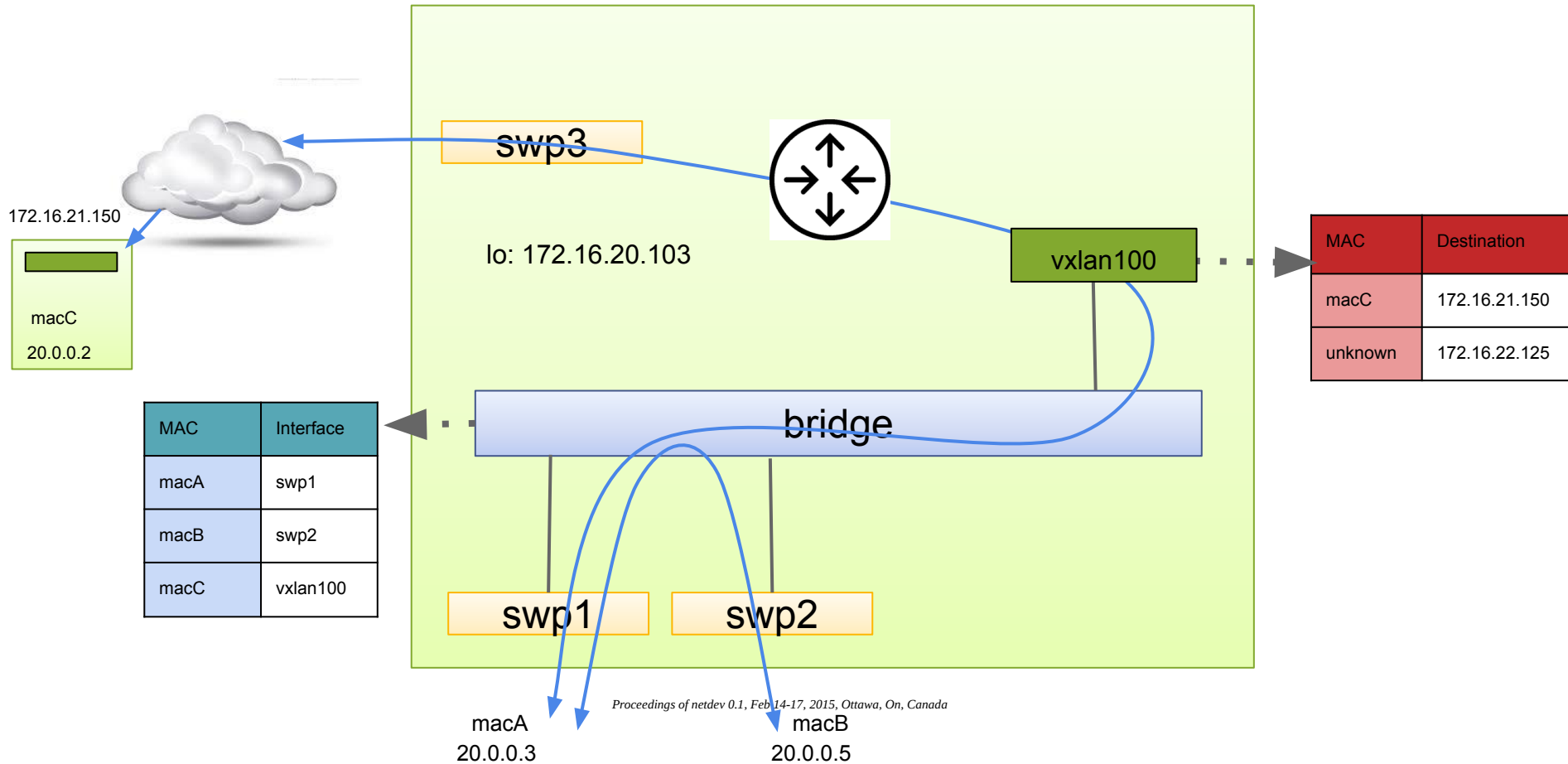
dev bridge port swp1 grp 224.1.2.3 temp
router ports on bridge: swp2



IGMP snooping offload

- switch driver configures hardware to send IGMP reports and queries to software
- bridge driver maintains IGMP group membership
- in some cases the reports or queries need to be re-forwarded in the kernel

VXLAN offload - hardware vtep



VXLAN offload - hardware vtep

Model

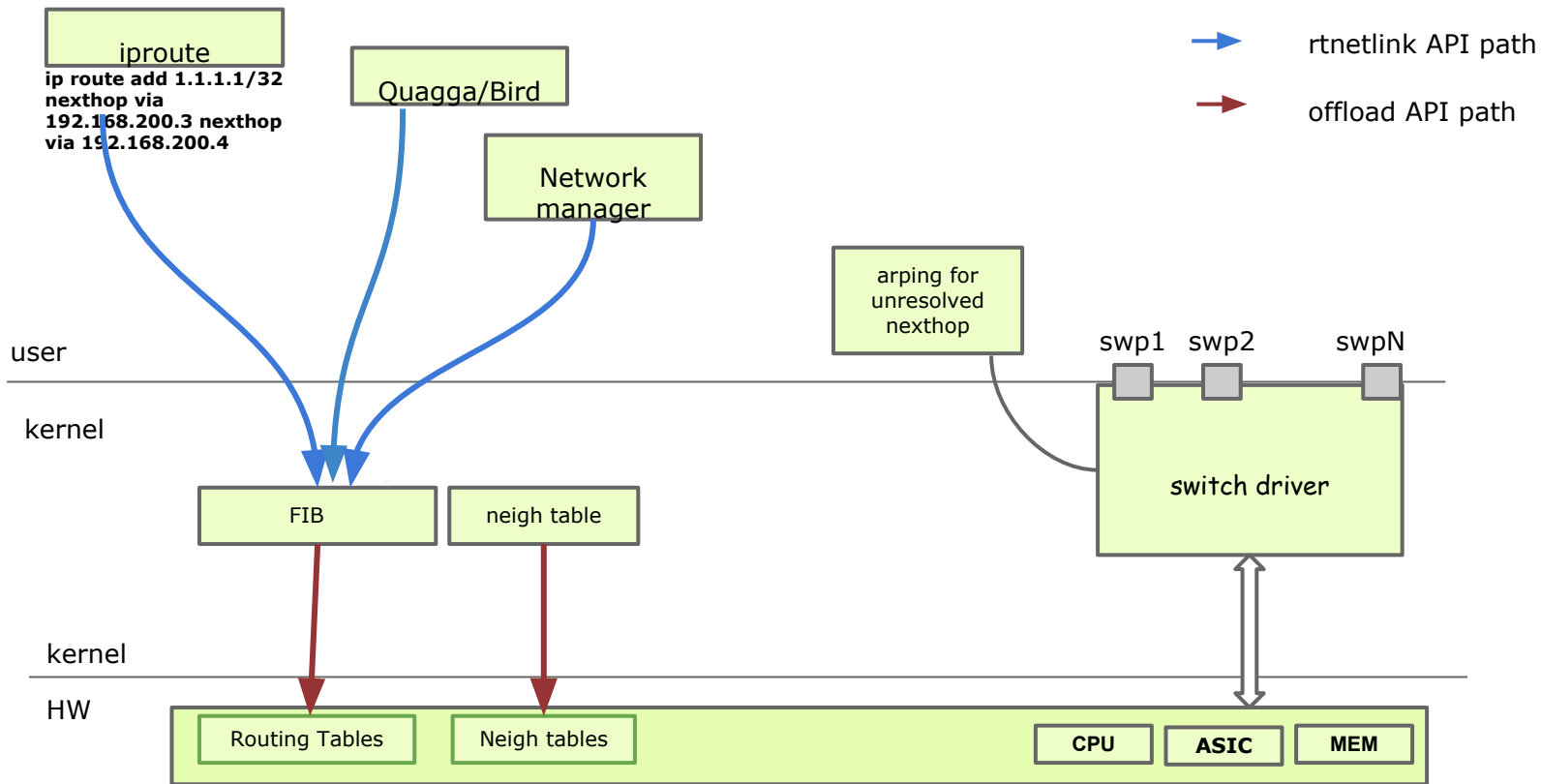
- VXLAN link as bridge port
 - bridging between local ports
 - VXLAN tunneling for remote MACs
- BUM traffic handling
 - multicast
 - using off-system replicator
 - could have a list of redundant replicators, need to choose ONE out of the list of remote dests (per flow or per vni etc.)
 - self replication
 - vtep sends to a list of remote vteps, need to choose ALL of the list of remote dests

VXLAN offload - ovsdb integration

Agent to translate ovsdb schema objects to kernel constructs.

OVSDDB	Linux kernel
logical switch	vxlan link + bridge
physical switch tunnel_ip	vxlan link local ip
logical port binding	bridge member port, vlan
unicast remote mac + physical locator	bridge fdb (mac, vlan, dst <remote ip>)
mcast remote mac "unknown" + physical locator list	vxlan link default dest
unicast local mac + physical locator	bridge fdb (mac, vlan, local dev)

I3 offloads



I3 hardware offload

- Routes via routing daemons go to the kernel
- Unresolved next hops, point to CPU in HW
- switch driver tries to resolve them by probes (arping)
- Refresh neigh entries for pkts routed through hardware (hit bit provided by hardware)