Stanisław Kardach

DPDK → RISC-V

Who am I?

Stanisław Kardach

Software engineer at Semihalf

- Working on dataplane software for > 5 years.
- Experience with Armv8, little bit of MIPS III.
- ODP - platform implementation.
- DPDK
  - Amazon ENA PMD development.
  - In collaboration with StarFive: the RISC-V port.

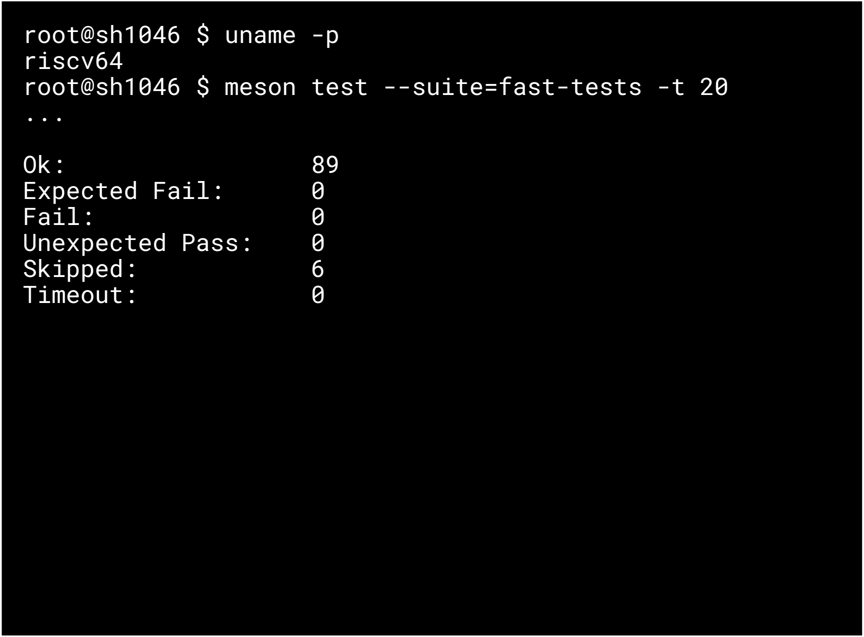
Agenda

- State of the port
- test-pmd or didn’t happen!
- Demo
- How did the port go?
- Next steps
- Q&A

State of the port

What works?

- Based on 21.05-rc1
- Native + cross builds with GCC.
- meson suite fast-tests: 89/95 pass.
- All no-iommu UIO drivers: igb_uio, uio_pci_generic, vfio-pci.
- ixgbe PMD + testpmd
  - Intel x520-DA2



```
root@sh1046 $ uname -p
riscv64
root@sh1046 $ meson test --suite=fast-tests -t 20
...

Ok:                   89
Expected Fail:        0
Fail:                 0
Unexpected Pass:      0
Skipped:              6
Timeout:              0
```

What is missing?

- i40e PMD driver requires vector operations.
- Relocation support issues in compilers:
  - Clang build not supported.
  - `rte_ring` tests cause debug GCC build to fail (workaround patches).
- `rte_crypto` and `rte_ipsec` libraries validation (with openssl).
- FreeBSD not supported yet.

Known limitations

- 128b atomic operations in RV64GC ISA -> no lock-free `rte_stack`.
- No prefetch in RV64GC ISA -> `rte_prefetch` functions are NOPs.
- No RISC-V cpufreq driver -> no `rte_power` support.

test-pmd or didn’t happen!

Test setup

DUT (riscv64):

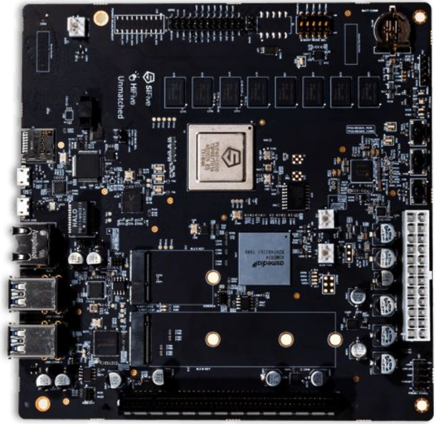
- CPU: 4 x U74 @ 1.2GHz + 16GB DDR4
- NIC: Intel x520-DA2

Packet generator (x86):

- CPU: Intel Core i5-4460 + 16GB DDR3
- NIC: Intel x520-DA2

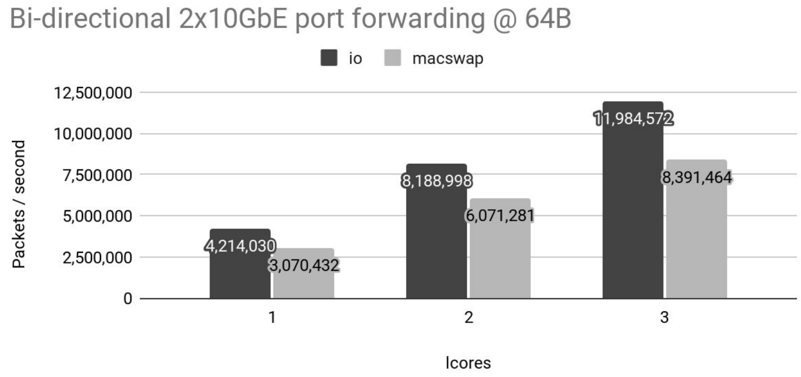
Test parameters:

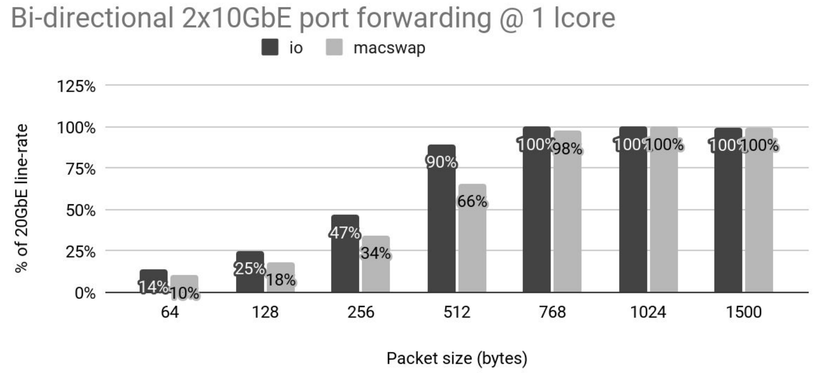
- Burst: 32
- Port topology: paired



Test setup

- 2-port forwarding packet-rate:
  - io: 4 - 4.2 Mpps
  - macswap: 2.8 - 3 Mpps
- 2-port forwarding latency (1 core / port):
  - io: 21us - 1100us
  - macswap: 24us - 1440us
- 20G line-rate / core with ~768B packets





Demo

How did the port go?

Challenges: Platform

- Extra Kernel patches to enable generic PCI resource MMAP.
- RISC-V relaxation issues:
  - GCC 10.2.0 generates `jal` for goto/for loops even in large, inline code.
  - Workaround: un-inline ring unit-test wrappers in functional tests.
  - Clang 11 doesn’t support relaxation at all (and `crt1.o` has _ALIGN).
- RISC-V CPU detection:
  - Linux has SBI calls for MVENDORID, MARCHID, MIMPID…
  - ...just doesn’t put them into /proc/cpuinfo.
- TIME frequency detection:
  - Now via /proc/device-tree/cpus/timebase-frequency: Linux specific.
  - What about when UEFI/ACPI comes?

Challenges: DPDK

- Fixing DPDK vector-ops assumptions.
  - `xmm_t` struct is a platform specific struct but used in generic code.
  - Missing scalar version of LPM.
  - Missing vector stubs for l3fwd.
- Time counting for `rte_rdtsc` (seen same story with Armv8 port):
  - TIME: not-gated, stable frequency, low-resolution (i.e. 1MHz).
  - CYCLE: gated, variable frequency, high-resolution (i.e. 1.2GHz).
- Extra DPDK patches for unit tests:
  - Add lock-free support detection to `rte_stack`.
  - Fix race-conditions in `rte_distributor` tests.

Performance evaluation?

- What about a precise `rte_rdtsc`?
  - CYCLE is gated so any `pause` screws up measurements.
  - TIME is low resolution so less useful.
  - TIME, CYCLE and INSTRET reads are not mandatory in S/U-mode.
- Linux perf driver - Basic now, SBI-based (FW) soon.
  - `perf stat` - works with `cycles` and `instructions`.
  - `perf record` - doesn't work - event filtering and overflow interrupt missing.
    - Sscofpmf extension in fast-track.
    - CYCLE and INSTRET are **not** part of that extension.
- Linux uprobes + eBPF.
  - Kernel eBPF JIT and uprobes support is there (≥5.12).
  - Potential for targeted sampling, but … no RISC-V support in bcc yet (just a PoC).

Next steps

Next steps

- Optimization!
- Add support for new platforms - needs CPU detection in Linux.
  - V extension to enable missing features.
- `rte_crypto` and `rte_ipsec` libraries validation (with openssl).
- Upstreaming (at some point).
- Long run: ISA adaptation for high-performance applications.
  - Explicit memory prefetching hints.
  - Overflow + filtering for `cycle` and `instret` (beyond Sscofpmf).

Thank you

Q & A