FreeBSD on Cavium ThunderX System on a Chip

Zbigniew Bodek
zbb@semihalf.com, zbb@freebsd.org
Wojciech Macek
wma@semihalf.com, wma@freebsd.org

SEMIHALF
EMBEDDED SYSTEMS

Presentation outline

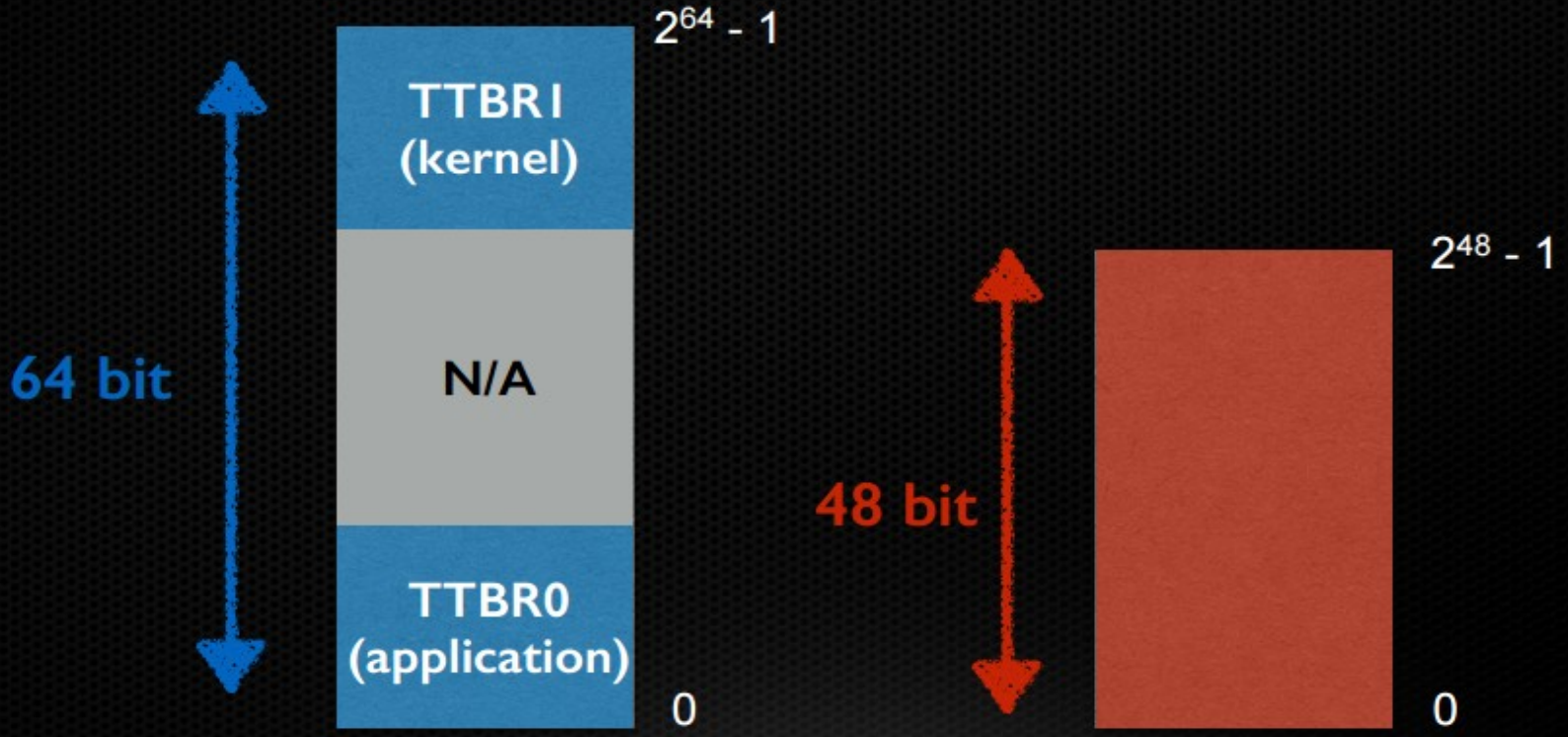
- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - PCIe
  - VNIC
- Current state & future work
- Performance measurements
- Q&A

Presentation outline

- Hardware platform
  - **ARMv8 as a 64-bit successor**
  - Cavium ThunderX
- Porting
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - PCIe
  - VNIC
- Current state & future work
- Performance measurements
- Q&A

ARMv8 as a 64-bit successor

- What is a gain in moving to 64-bit ARMs?
  - Much larger virtual and physical address space



ARMv8 as a 64-bit successor

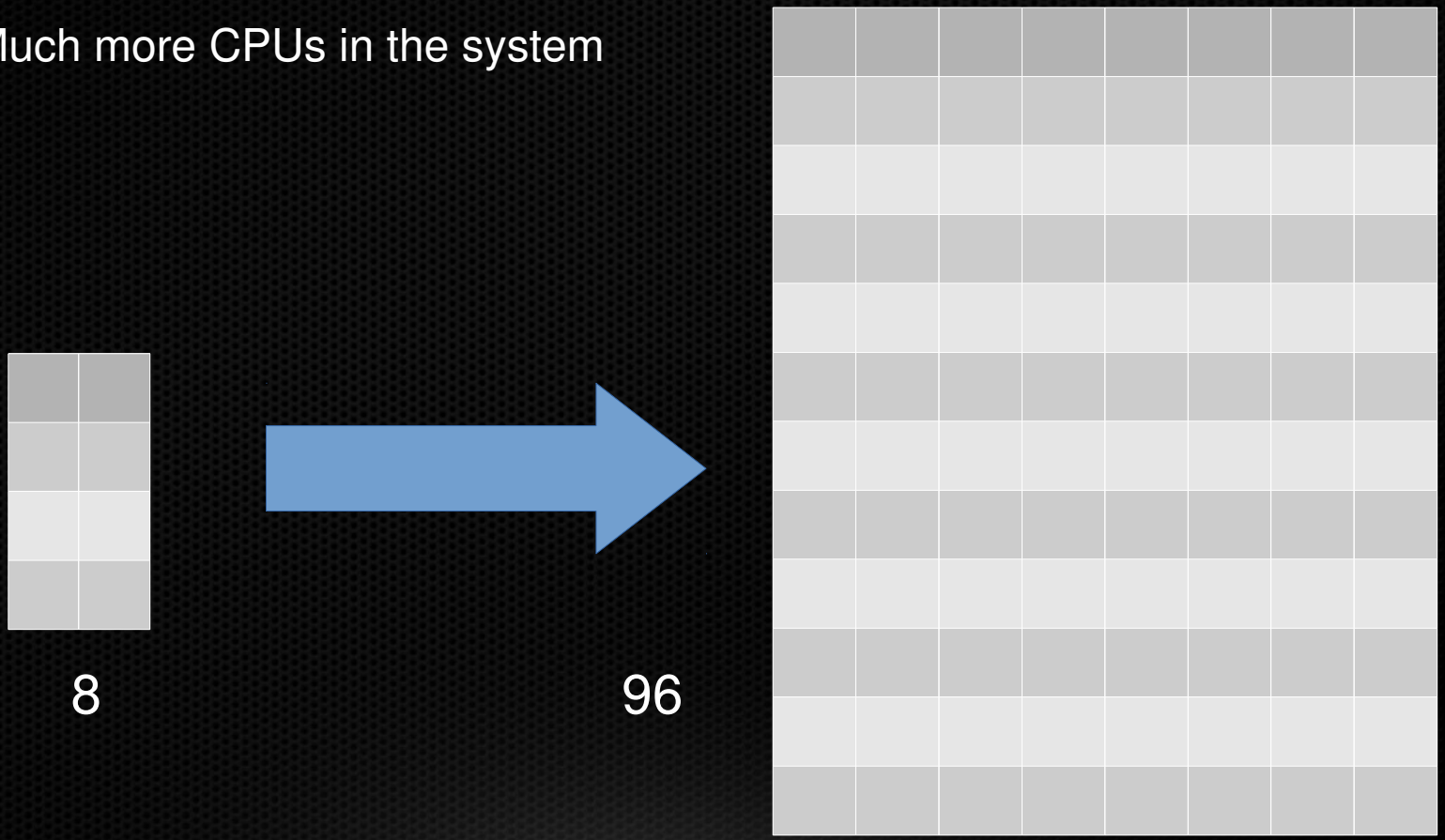
- What is a gain in moving to 64-bit ARMs?
  - Twice as much data in each register

SP	PC

x0	x1	x2	x3	x4	x5	x6	x7
x8	x9	x10	x11	x12	x13	x14	x15
x16	x17	x18	x19	x20	x21	x22	x23
x24	x25	x26	x27	x28	x29	x30 (LR)	xzr

ARMv8 as a 64-bit successor

- What is a gain in moving to 64-bit ARMs?
  - Much more CPUs in the system



ARMv8 as a 64-bit successor

- What is a gain in moving to 64-bit ARMs?
  - New instruction set (aarch64)
  - Clear design (Exception Levels)
  - Cryptographic instructions (AES, SHA-1, SHA-256)
  - Mandatory VFP and NEON
  - Backward compatibility with 32-bit application in 32-bit mode
  - New, standardized components (PSCI, SMMU)

Presentation outline

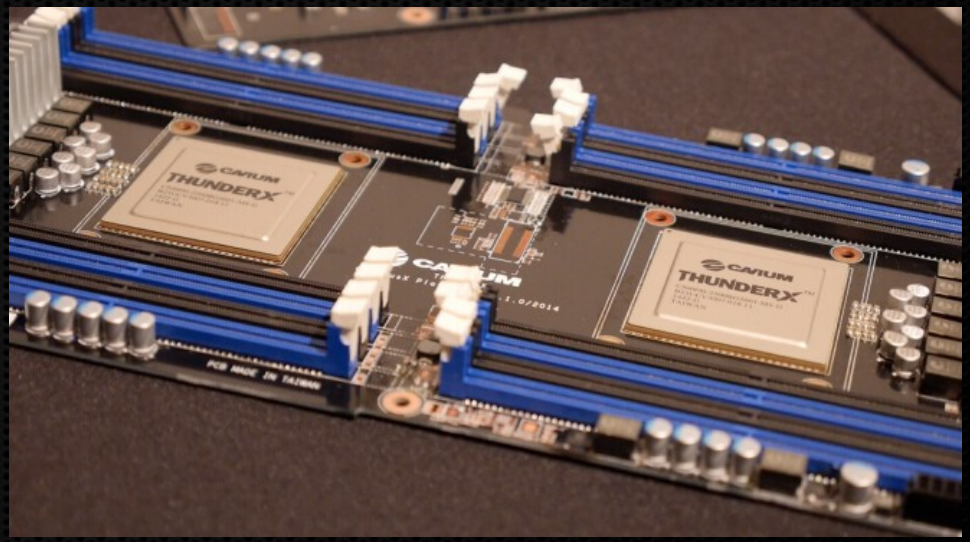
- Hardware platform
  - ARMv8 as a 64-bit successor
  - **Cavium ThunderX**
- Porting to ThunderX
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - PCIe
  - VNIC
- Current state & future work
- Performance measurements
- Q&A

Cavium ThunderX

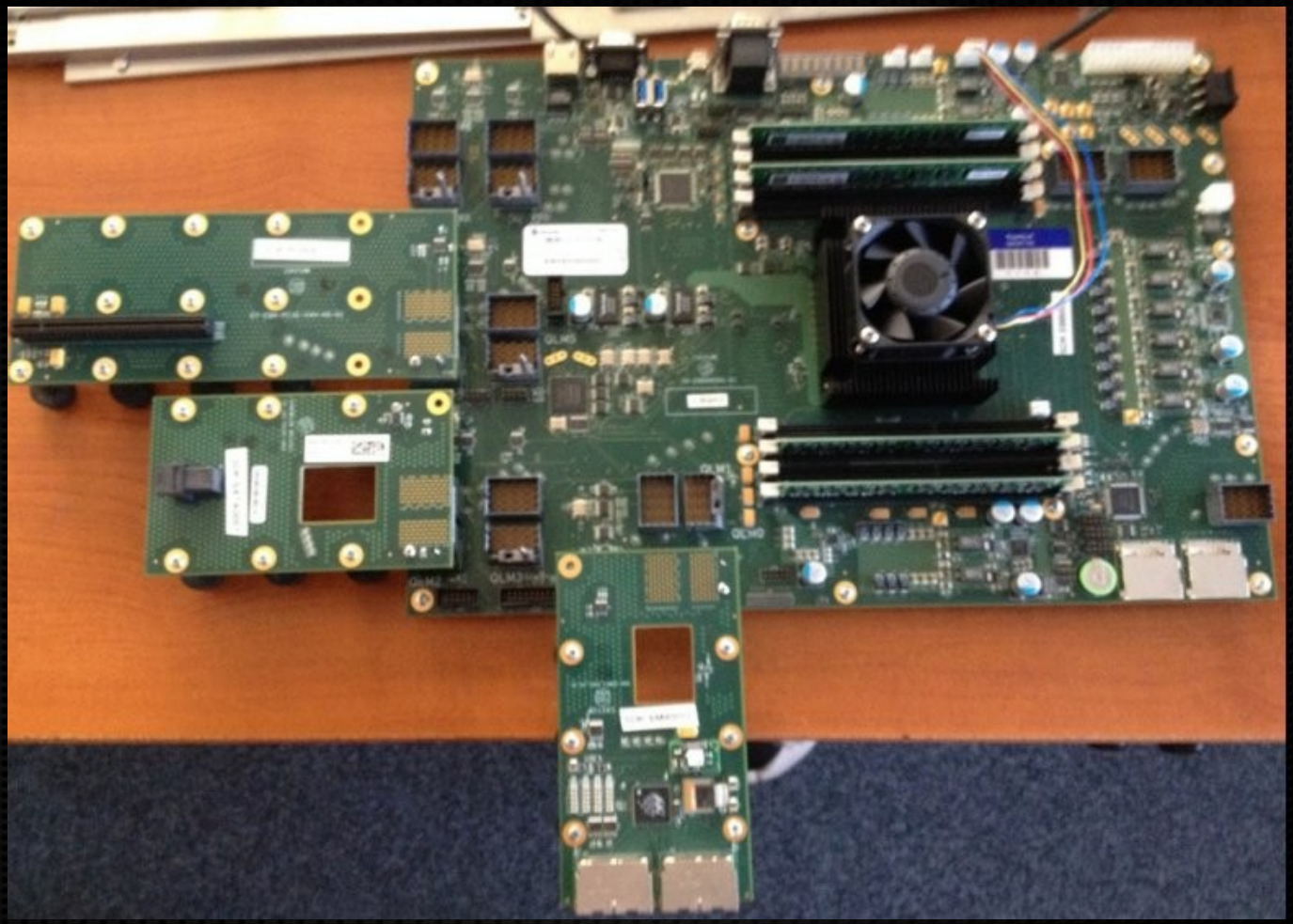
- 48 ThunderX ARMv8 CPUs (per chip)
- Fully coherent in respect to L1, L2 and DMA
- Virtualization

- PCIe3.0-centric
- GIC + ITS interrupt handling
- Up to 1TB of RAM

- 1/10/20/40 GbE
- SATA 3.0
- USB 3.0



Cavium ThunderX



ThunderX EVB with 48-core CPU (early prototype)

Cavium ThunderX

ThunderX dual-socket CRB with 96-core CPU (server chassis form-factor)

Presentation outline

- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting to ThunderX
  - **Initial FreeBSD state**
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - PCIe
  - VNIC
- Current state & future work
- Performance measurements
- Q&A

Initial FreeBSD state

- What was claimed to be working
  - Build tools (toolchain, etc.)
  - Load and boot the kernel
  - Start code (prepare C environment)
  - Low-level initialization
  - VM support
  - Exception handling
  - Userland (libs, syscalls, context switching and so on...)
- What was known to be missing
  - Interrupt controller, GICv3+ITS
  - ThunderX specific drivers

Initial FreeBSD state

- What was actually working
  - Build tools (toolchain, etc.)
  - Load and boot the kernel [BUGS]
  - Start code (prepare C environment) [BUGS]
  - Low-level initialization [BUGS]
  - VM support [BUGS]
  - Exception handling [BUGS]
  - Userland (libs, syscalls, context switching and so on...) [BUGS]

- What was known to be missing
  - Interrupt controller, GICv3+ITS
  - ThunderX specific drivers

Presentation outline

- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting to ThunderX
  - Initial FreeBSD state
  - **Bugs, bugs and... bugs**
- On-chip peripherals support
  - GICv3 + ITS
  - PCIe
  - VNIC
- Current state & future work
- Performance measurements
- Q&A

Bugs, bugs, and... bugs

- The source of bugs
  - Switch from a simulator to a real hardware
    - Cache, pipeline influence
    - Realistic timings
    - Race conditions
  - Platform
    - Hardware imperfections (erratas)
    - Many-core impact
  - Human errors
    - Off-by-one errors
    - Overflows
  - Other
    - Missing bits / genuine bugs

Bugs, bugs, and... bugs

- Example 1

```
/* Restore elr and spsr */
ldp x0, x1, [sp, #16] /* load pair from */
msr elr_el1, x0
msr spsr_el1, x1
(…)
eret /* return from exception to elr_el1 with PSR
        from spsr_el1 */
```

IRQ here overwrites SPSR

Bugs, bugs, and... bugs

- Example 1

```
msr daifset, #2 /* Lock interrupts */
/* Restore elr and spsr */
ldp x0, x1, [sp, #16] /* load pair from */
msr elr_el1, x0
msr spsr_el1, x1
(…)
eret /* return from exception to elr_el1 with PSR
        from spsr_el1 */
```

Bugs, bugs, and... bugs

- Example 2

```
https://community.arm.com/groups/processo
rs/blog/2016/03/11/semihalf-arm-blog-2-de
ad-board-and-stack-growth
```

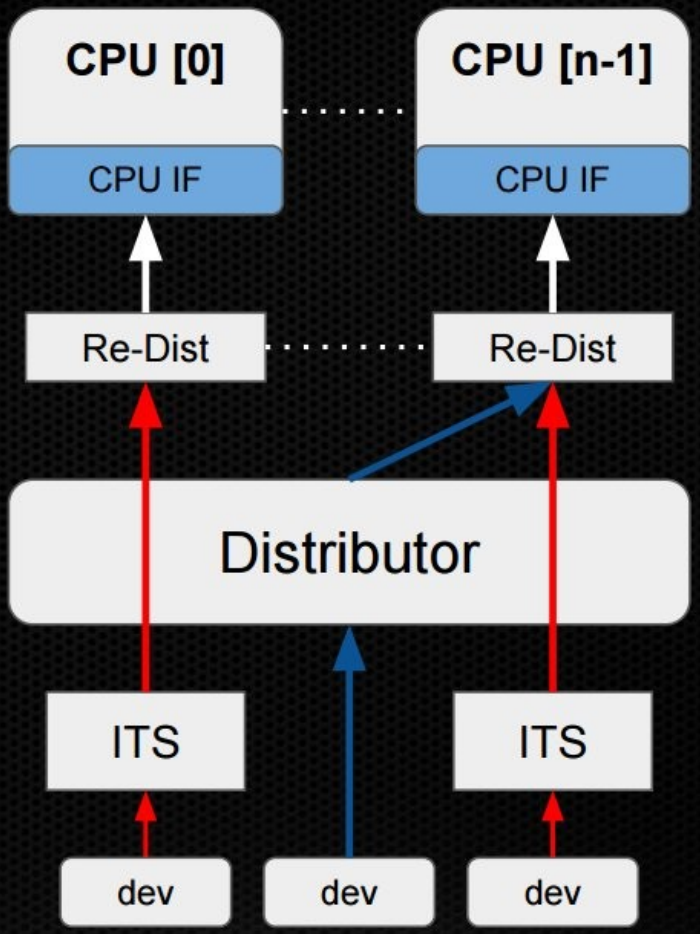
Presentation outline

- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting to ThunderX
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - **GICv3 + ITS**
  - SMP
  - PCIe
  - VNIC
- Current state & future work
- Performance measurements
- Q&A

GICv3 + Interrupt Translation Service

- Improvements in GICv3
  - Three components:
    - Distributor
    - Re-Distributor
    - CPU Interfaces
  - Affinity-based routing (Aff3:Aff2:Aff1:Aff0, domain:socket:cluster:core)
  - MSI/MSI-x support with ITS assist
  - Support for a huge number of interrupts (2^32 - 1)
  - Auto-configuration capabilities

GICv3 + Interrupt Translation Service



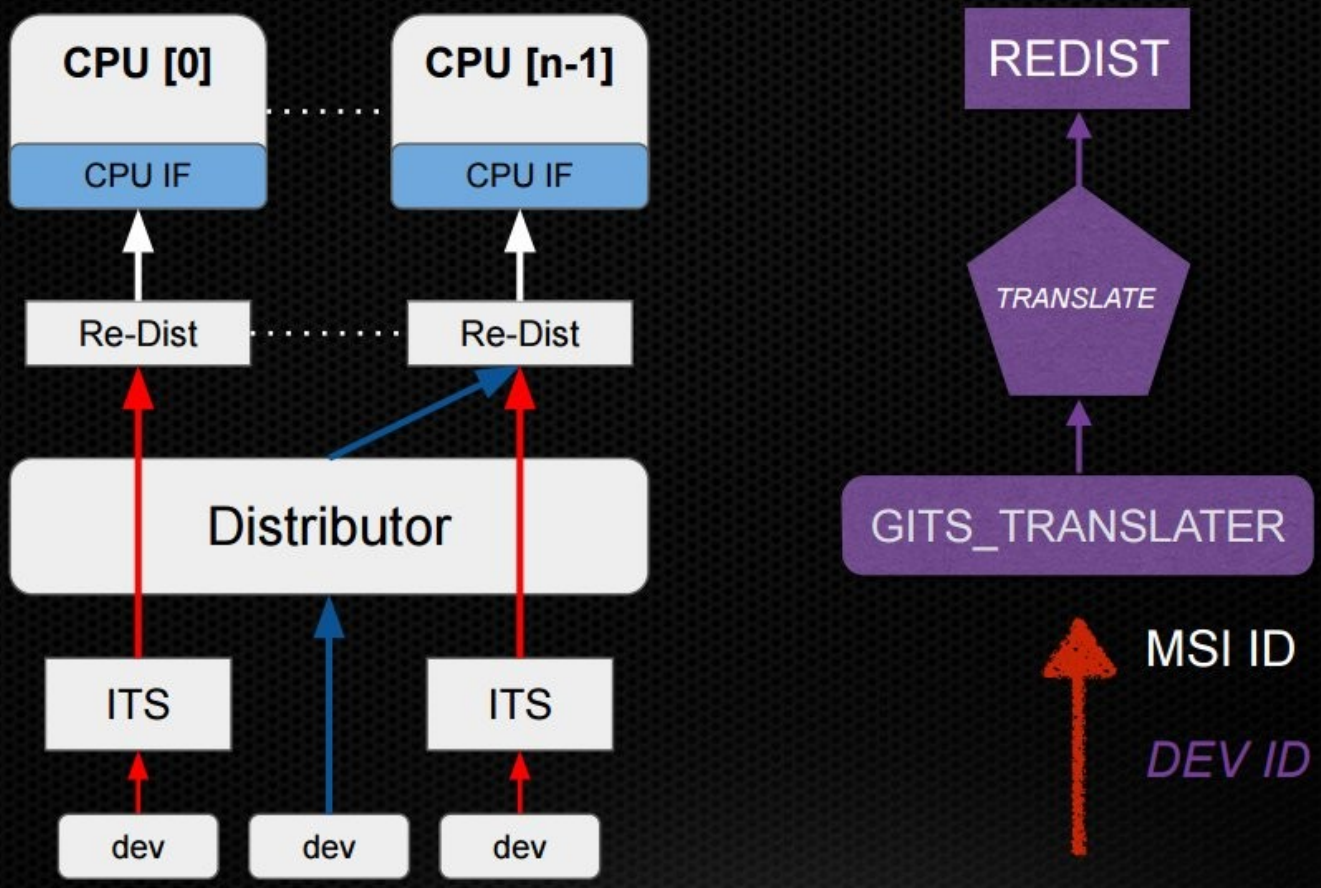
GICv3 + ITS architecture

GICv3 + Interrupt Translation Service

- Responsibilities of GICv3 (Re-Distributor, Distributor):
  - Manage PPIs and SGIs through Re-Distributors
  - Handle legacy interrupts from devices through Distributor
  - Send and receive IPIs based on CPU affinity address

GICv3 + Interrupt Translation Service

- Interrupt flow chart



GICv3 + Interrupt Translation Service

- Responsibilities of Interrupts Translation Service:
  - Add new interrupt class to GIC, LPI – Locality-specific Peripheral Interrupts
  - Provide a way for devices to trigger message-signaled interrupts
  - Translate MSI DEV_ID&MSI_ID to an appropriate IRQ number
  - Pass translated interrupts directly to Re-Distributor based on set up affinity
- Advantages of new approach:
  - Standardized way to handle MSI/MSI-x
  - Great flexibility in interrupt management
  - Drastically increase in total MSI/MSI-x number

GICv3 + Interrupt Translation Service

- Setting up MSI-x (LPI) interrupt for a PCIe device

1. Map interrupt collection to a Re-Distributor
2. Allocate and map Interrupt Translation Table for a PCIe device)
3. Reserve a range of LPIs (these numbers must be unique in the system)
4. Map DEV_ID and MSI vector (i.e. memory address where the write issued by device triggers interrupt) to an appropriate collection and LPI

- Cons:
  - Complex logic, difficult to program
  - Slow and complicated configuration of LPI interrupts
- Pros:
  - Easy to use when programmed
  - Does not need to occupy SPIs

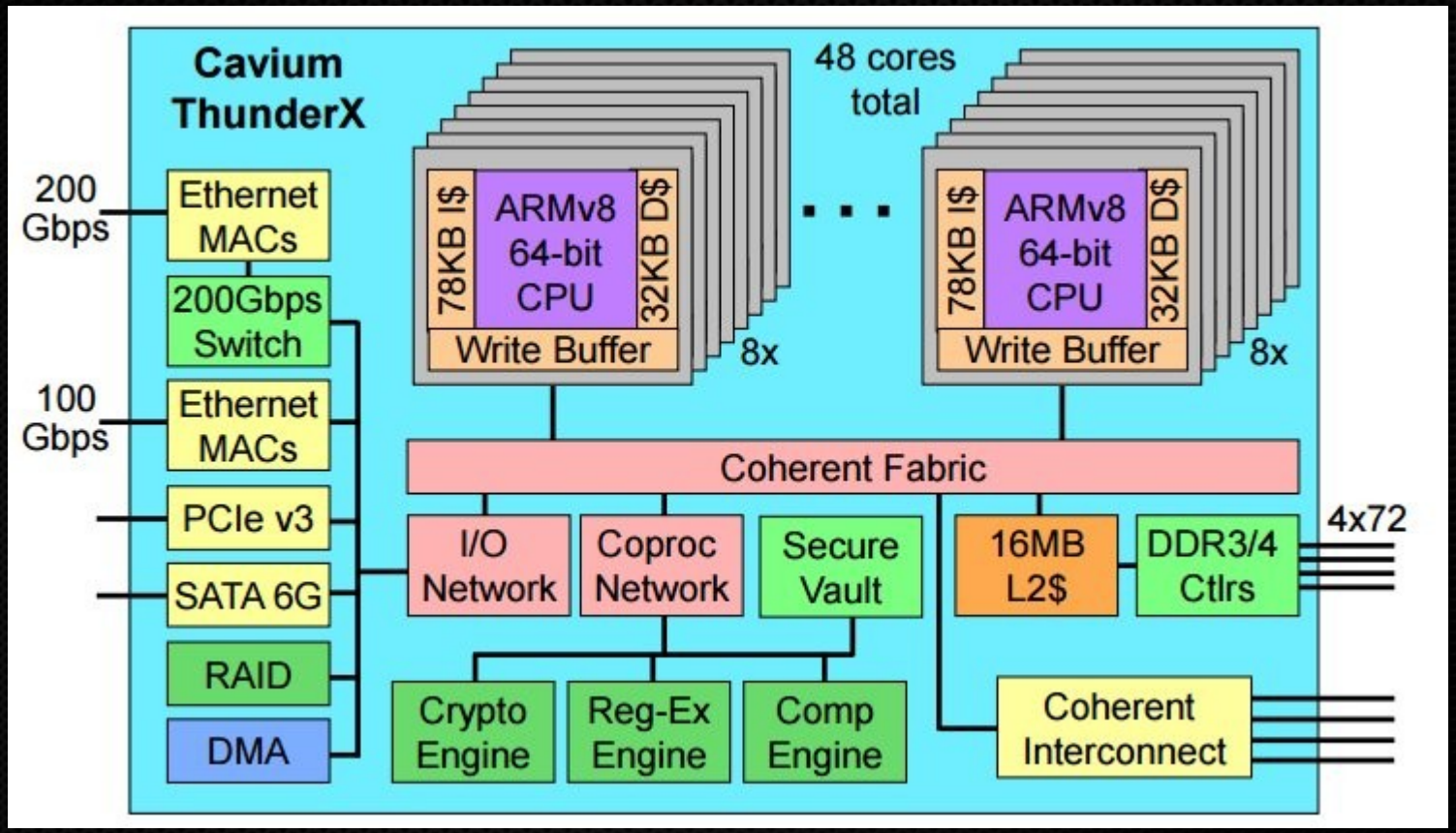
Presentation outline

- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting to ThunderX
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - **PCIe**
  - VNIC
- Current state & future work
- Performance measurements
- Q&A

PCI express

- ThunderX PCI-based architecture
  - All on-chip devices are connected through internal PCIe bus
  - More than 200 PCIe endpoints on internal bus
  - ECAM configuration space (i.e. generic)
- Flexibility of design
  - No DTS or ACPI is required, enumeration done using PCI standard
  - All components can be reused in next chip revisions
  - Message Signaled interrupts

PCI express


Cavium ThunderX
48 cores total
200 Gbps
Ethernet MACs
200Gbps Switch
100 Gbps
Ethernet MACs
PCIe v3
SATA 6G
RAID
DMA
78KB I$
ARMv8 64-bit CPU
32KB D$
Write Buffer
8x
78KB I$
ARMv8 64-bit CPU
32KB D$
Write Buffer
8x
Coherent Fabric
I/O Network
Coproc Network
Secure Vault
16MB L2$
DDR3/4 Ctlrs
4x72
Crypto Engine
Reg-Ex Engine
Comp Engine
Coherent Interconnect

Presentation outline

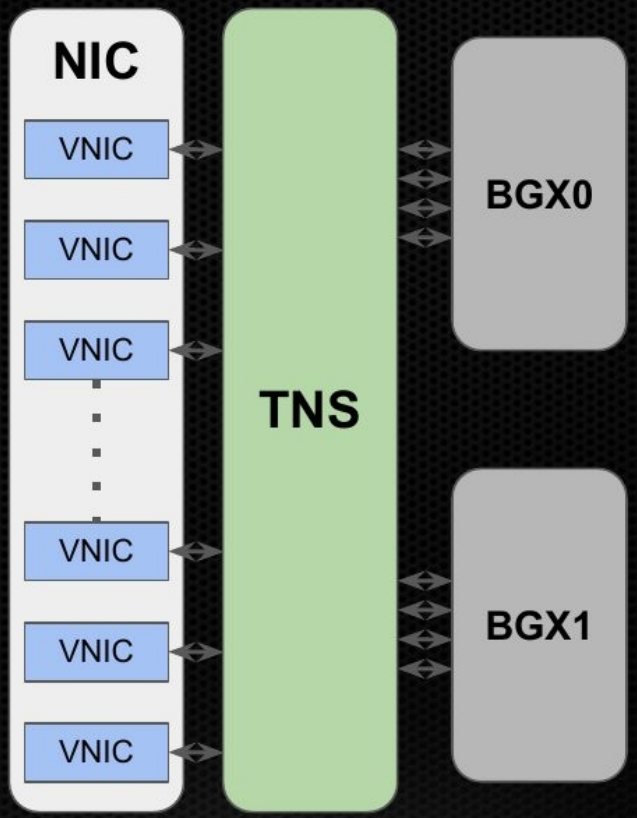
- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting to ThunderX
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - PCIe
  - **VNIC**
- Current state & future work
- Performance measurements
- Q&A

VNIC

- Virtualized NIC features:
  - 1/10/20/40 Gbps Ethernet
  - Partitioning:
    - Programmable MAC layer (BGX)
    - Network Interface Controller (NIC)
    - Traffic Network Switch (TNS), unused

VNIC

- Virtualized NIC internals:



128x VNIC

2x BGX
4x LMAC each

Max 10Gbps per LMAC

NIC partitioning on ThunderX

VNIC

- BGX functions:
  - Ordinary PCIe device on internal PCIe bus
  - Two instances in a chip offering 4xLMACs each
  - LMAC can be connected to the arbitrary VNIC
  - Polling PHYs for link status

VNIC

- NIC, Physical Function:
  - Does not offer network capabilities! - Resource Manager only.
  - Can create up to 128 Virtual Functions (VNICs) using SR-IOV
  - Provides interface to PHYs via BGX.
  - Can communicate with its Virtual Functions using Mailboxes

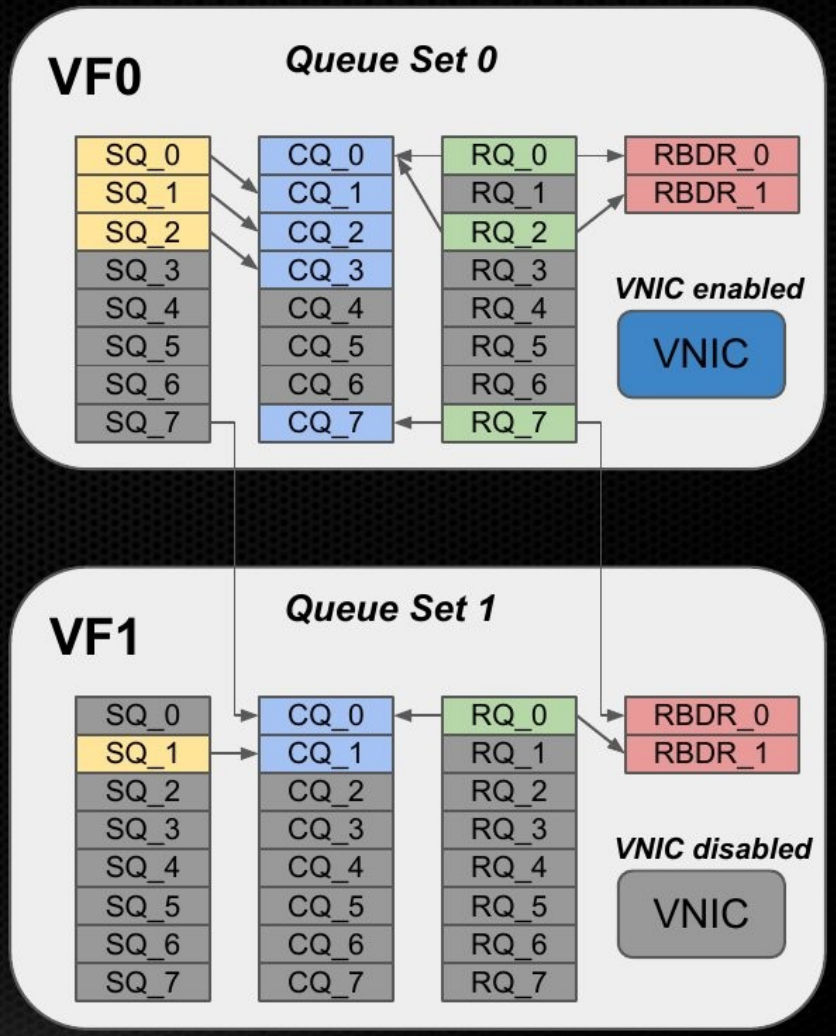
VNIC

- NIC, Virtual Function:
  - Can be a fully functional network adapter, or:
  - Contains set of queues (Q-Set)
    - 8x RQ – Receive Queue
    - 8x SQ – Send Queue
    - 8x CQ – Completion Queue
    - 2x RBDR – Receive Buffer Ring

  - Can be only a container to add more Q-Sets to another VF – networking functions are disabled for this VF then

VNIC

- Example:
  - VF0 offers networking functionality
  - VF1 used for extending resources for VF0
  - TX:
    - SQ used to enqueue packets
    - CQ signals when packet is sent
  - RX:
    - RQ writes to CQ when packet is received
    - Driver can read descriptor and know the source RQ number
  - Many-to-one
    - Both CQ and RBDR can store events from more than one SQ/RQ



Presentation outline

- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting to ThunderX
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - SMP
  - PCIe
  - VNIC
- **Current state & future work**
- Performance measurements
- Q&A

Current state and future work

- Done:
  - Single socket ThunderX is working on HEAD
  - All IO interfaces are supported with good performance
  - System is considered stable
- TODO:
  - Multicore scalability (generic FreeBSD issue)
  - Crypto support
  - Secondary ITS (required for 2S)
  - Dual socket support

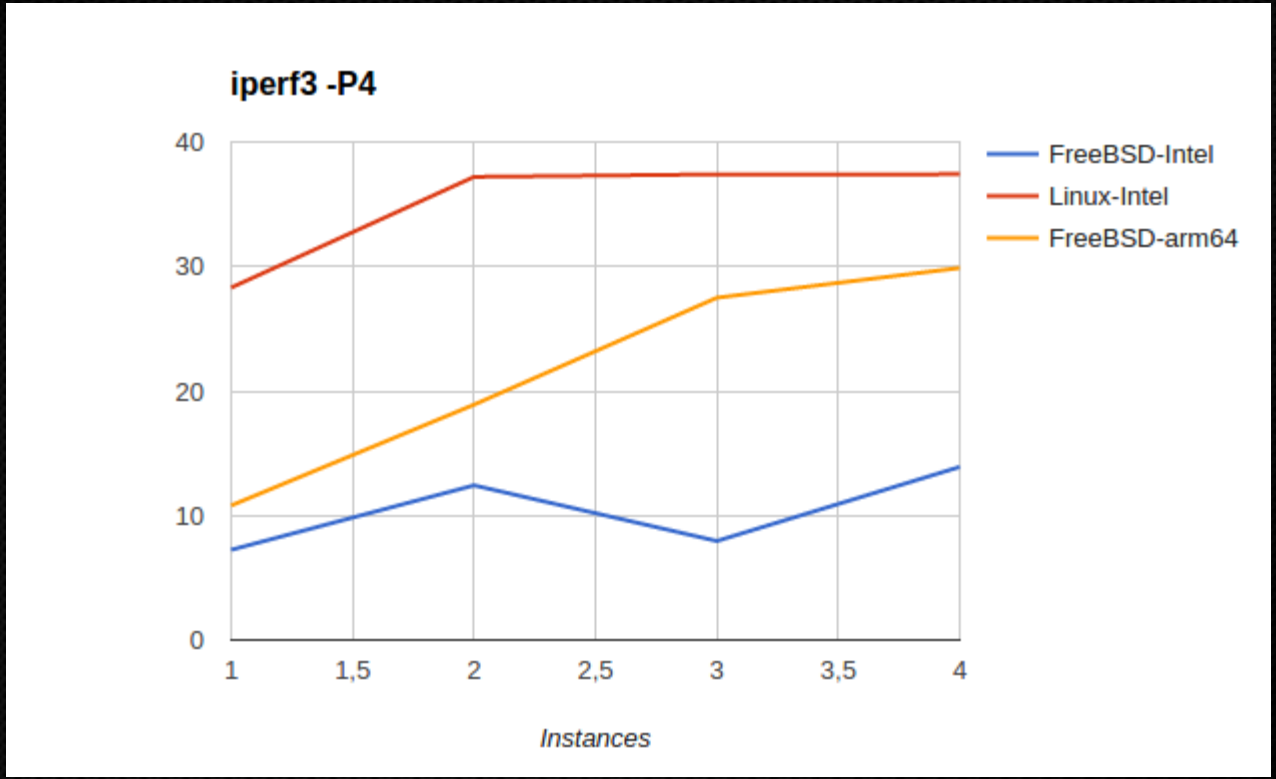
Presentation outline

- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting to ThunderX
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - SMP
  - PCIe
  - VNIC
- Current state & future work
- **Performance measurements**
- Q&A

Performance measurements

- Test1 - iperf3:
  - Using iperf3 and 40Gb Direct-attached-copper link
  - Tested Intel vs ThunderX, FreeBSD vs Linux
  - Using 4 instances of iperf3, with 1 to 4 TCP streams per instance
- Setup 1:
  - ThunderX @ 1.8GHz, 48CPU, FreeBSD HEAD, 40Gb VNIC
- Setup 2:
  - Intel Xeon E5-2603v3 @ 1.6GHz, 6CPU, Mellanox ConnectX-3 40Gb, FreeBSD HEAD
- Setup 3:
  - Intel Xeon E5-2603v3 @ 1.6Ghz, 6CPU, Mellanox ConnectX-3 40Gb, Ubuntu Linux, kernel 4.2

Performance measurements

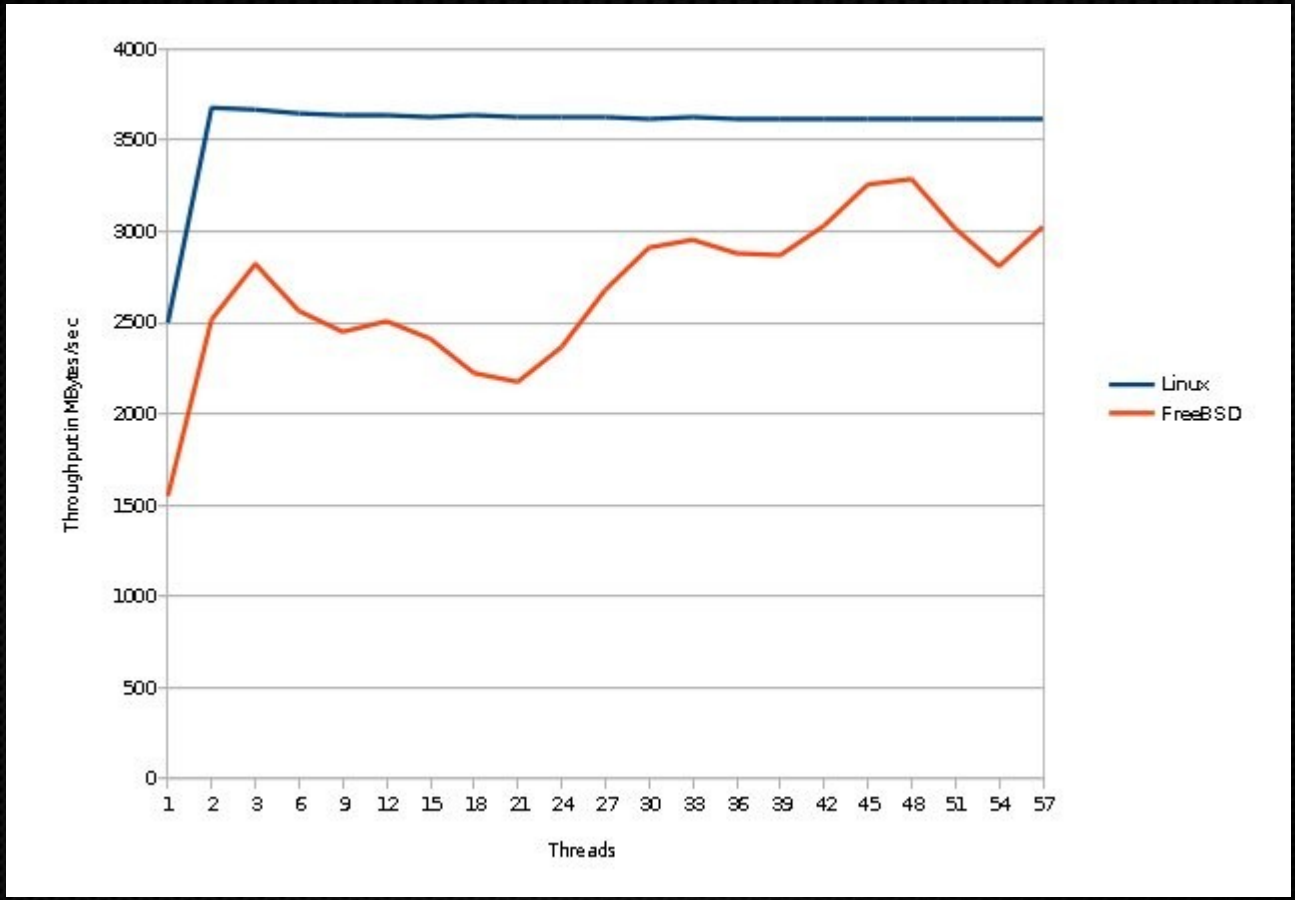


Performance measurements

- Test2 - Nginx:
  - Using Nginx to serve memory-stored files using HTTP
  - Throughput is measured versus number of Nginx threads

- Setup 1:
  - ThunderX @ 1.8GHz, 48CPU, FreeBSD HEAD, 40Gb VNIC
- Setup 2:
  - ThunderX @ 1.8GHz, 48CPU, Ubuntu, kernel 4.2, 40Gb VNIC

Performance measurements

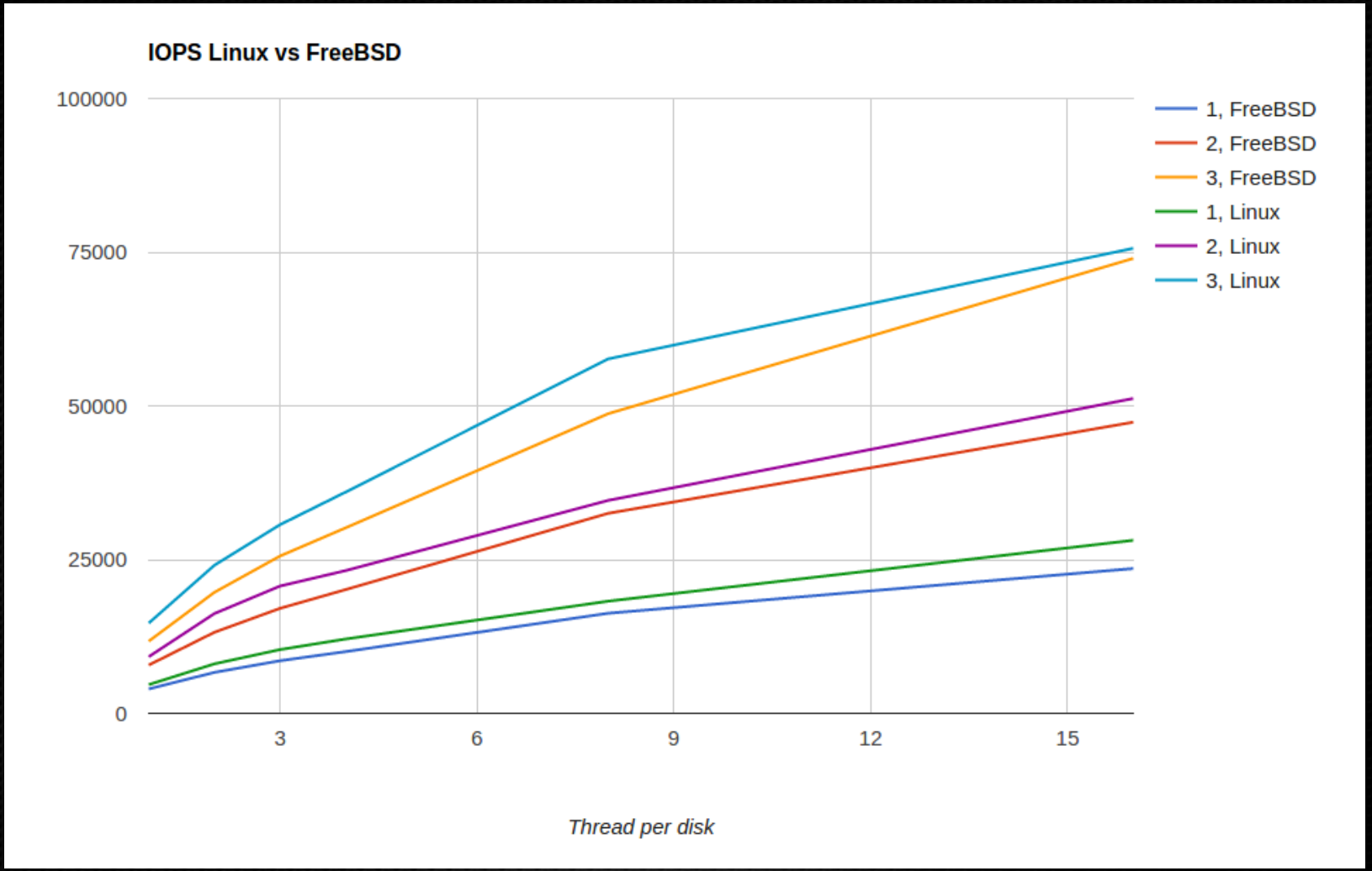


Performance measurements

- Test3 - AIO:
  - Using fio to benchmark block device performance
  - Tested with 3 SSDs, each capable of 50kIOPS@4k

- Setup 1:
  - ThunderX @ 1.8GHz, 48CPU, FreeBSD HEAD
- Setup 2:
  - ThunderX @ 1.8GHz, 48CPU, Ubuntu, kernel 4.2

Performance measurements



Presentation outline

- Hardware platform
  - ARMv8 as a 64-bit successor
  - Cavium ThunderX
- Porting to ThunderX
  - Initial FreeBSD state
  - Bugs, bugs and... bugs
- On-chip peripherals support
  - GICv3 + ITS
  - SMP
  - PCIe
  - VNIC
- Current state & future work
- Performance measurements
- **Q&A**

References

- https://wiki.freebsd.org/arm64
- https://community.arm.com/groups/processors/blog/2015/11/03/semihalfs-arm64-blog-1-the-freebsd-on-the-96-cpu-armv8-soc
- https://community.arm.com/groups/processors/blog/2016/03/11/semihalf-arm-blog-2-dead-board-and-stack-growth
- https://www.youtube.com/watch?v=1q5aDEt18mw

Acknowledgements

- Special thanks go to:
  - *Dominik Ermel, Michał Mazur, Tomasz Nowicki, Michał Stanek (all Semihalf)*
  - *Ed Maste (The FreeBSD Foundation), Andrew Wafaa (ARM), Larry Wikelius (Cavium) for the successful cooperation.*
  - *Andrew Turner (FreeBSD Project) for insightful reviews and advice.*
  - *Rafał Jaworowski (Semihalf) for organizing and managing this project.*

This project was sponsored by ARM, Cavium, The FreeBSD Foundation and Semihalf.

Q&A

Any questions?