SEE TEXT ONLY

ARMv8

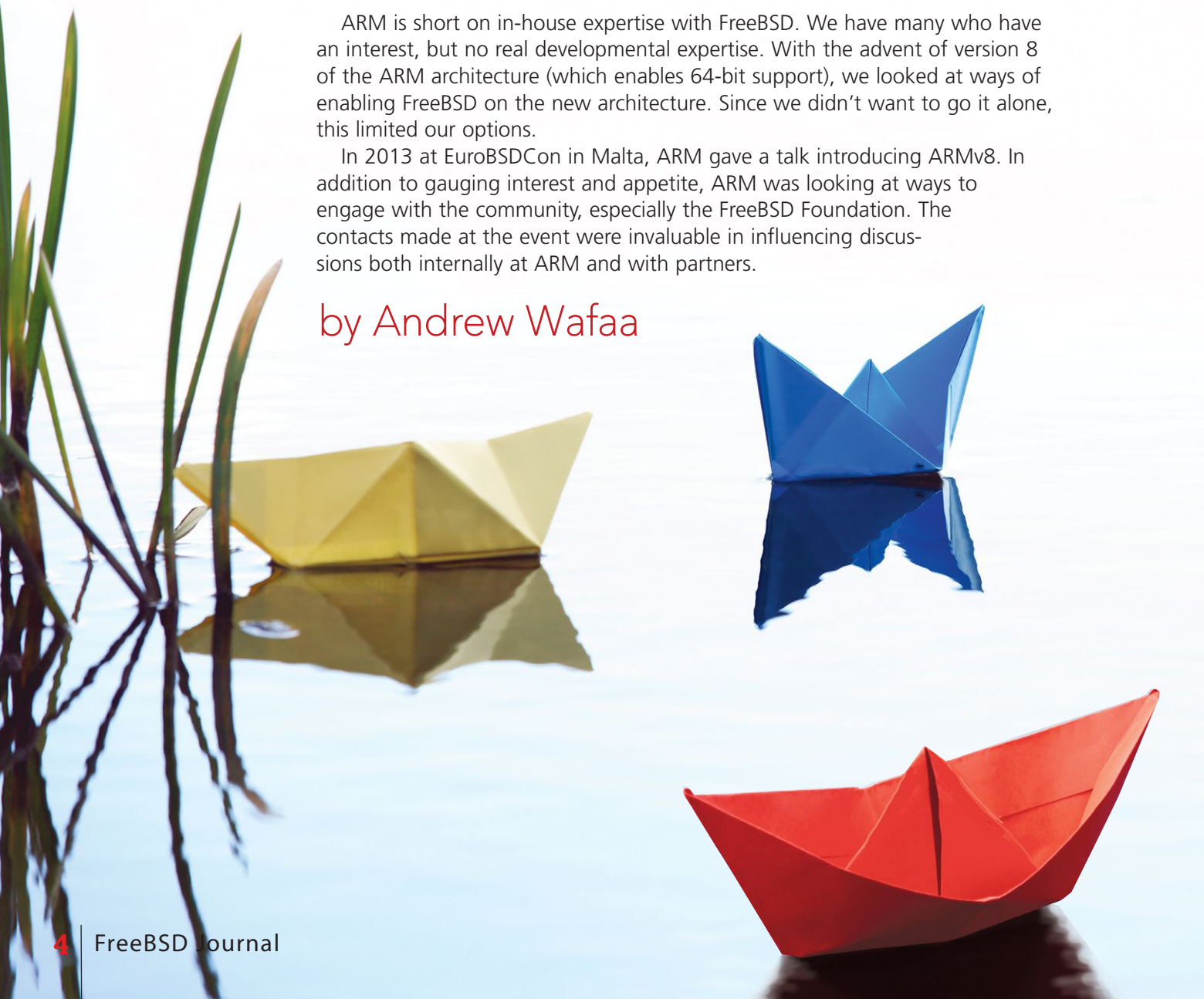
ENABLED AS TIER 1 ARCHITECTURE: A COLLABORATIVE DEVELOPMENT PROJECT

FreeBSD support for the ARM architecture has historically been a grassroots affair, and for a whole host of embedded platforms this works well. With the launch of ARMv8, the development of ARM-based systems has expanded into the enterprise space, both from a server infrastructure and a network infrastructure standpoint. To support the enterprise market, it was clear we needed a more aggressive development model to deliver the most complete and competitive offering of FreeBSD.

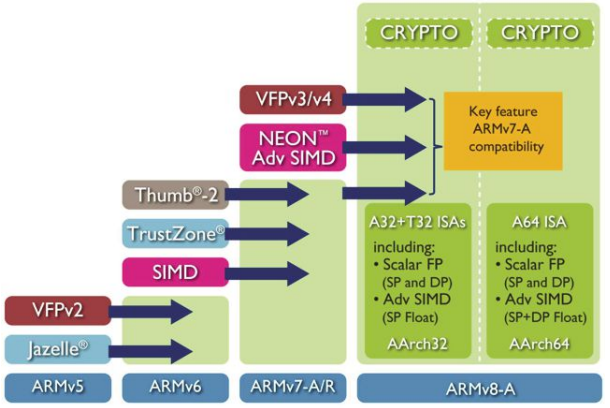
ARM is short on in-house expertise with FreeBSD. We have many who have an interest, but no real developmental expertise. With the advent of version 8 of the ARM architecture (which enables 64-bit support), we looked at ways of enabling FreeBSD on the new architecture. Since we didn't want to go it alone, this limited our options.

In 2013 at EuroBSDCon in Malta, ARM gave a talk introducing ARMv8. In addition to gauging interest and appetite, ARM was looking at ways to engage with the community, especially the FreeBSD Foundation. The contacts made at the event were invaluable in influencing discussions both internally at ARM and with partners.

by Andrew Wafaa



Time ticked on, and in 2014 ARM was confident enough to take the plunge and truly engage with the community in getting ARMv8 supported on FreeBSD as a Tier 1 architecture. ARM decided the best path forward was to sponsor the FreeBSD Foundation and work with them to get the right people, including Cavium Inc., engaged to move forward with enablement. As these discussions were going on, Andrew Turner, a longtime FreeBSD developer and committer, began work on getting 64-bit ARM (known as AArch64 or ARMv8 or ARM64) support into FreeBSD. ARM has been in discussions with Semihalf since 2012 on the topic of 64-bit ARM, Cavium started discussions in 2013 with Semihalf, and it was clear to both ARM and Cavium that Semihalf was a partner that needed to be involved.



Overview of the ARM Architecture

On October 1, 2014, Cavium released a public announcement describing the partnership with ARM and the FreeBSD Foundation. Both ARM and Cavium provided funding to the Foundation, which in turn engaged Andrew Turner and Semihalf. By working with and through the Foundation, both ARM and Cavium wanted to ensure that the effort was an inclusive and collaborative affair.

A TRUE COLLABORATION

As some people may not know all the parties involved, here is a brief overview of the key entities:

ARM Ltd. is an IP design house with a comprehensive product offering, including 32-bit and 64-bit RISC microprocessors, graphics processors, enabling software, cell libraries, embedded memories, high-speed connectivity products, peripherals, and development tools.

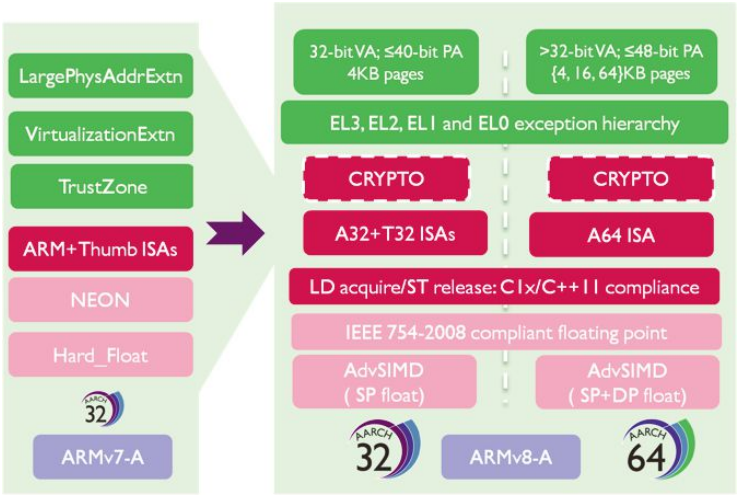
Cavium (NASDAQ: CAVM) is a leading provider of highly integrated semiconductor products that enable intelligent processing in enterprise, data center, cloud, and wired and wireless service provider applications. Cavium offers a broad portfolio of integrated, software-compatible processors ranging in performance from 100 Mbps to 100 Gbps that enable secure, intelligent functionality in enterprise, data-center, broadband/consumer and access and service provider equipment. In 2014 Cavium launched the ThunderX® Workload Optimized SOC focused on server workloads and scale out Data Center deployments.

Semihalf creates software for advanced solutions in the areas of operating systems, virtualization, networking, and storage. They make software tightly coupled with the underlying hardware to achieve maximum system capacity. Their partners and customers are semiconductor industry leaders. The following team at Semihalf was responsible for ThunderX/ARM64 development work: Wojciech Macek (project leader, FreeBSD committer; responsible for ARMv8 base support, PCI, SMP and storage areas), Zbigniew Bodek (FreeBSD committer; ARMv8 base support, GICv3/ITS, VNIC), Dominik Ermel (AHCI, testing), Michał Stanek (PCI, SMP, debugging), and Tomasz Nowicki (KDB, watchpoints), with additional help from Michał Mazur (UEFI bring-up and debug).

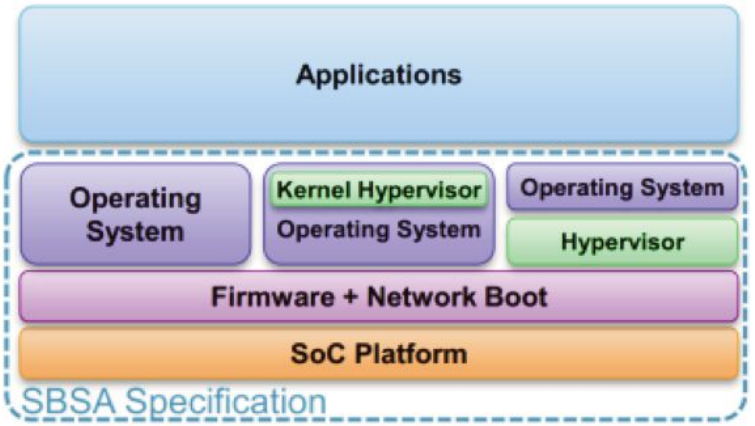
Andrew Turner is a freelance software engineer and FreeBSD committer focusing on running FreeBSD on ARM-based chips. He started the ARM64 FreeBSD port. Andrew was responsible for the overall system architecture, the parts of the system that may be used by many SoCs. Prior to this, Andrew worked as an embedded software engineer for consulting companies and silicon vendors.

Divide and Conquer

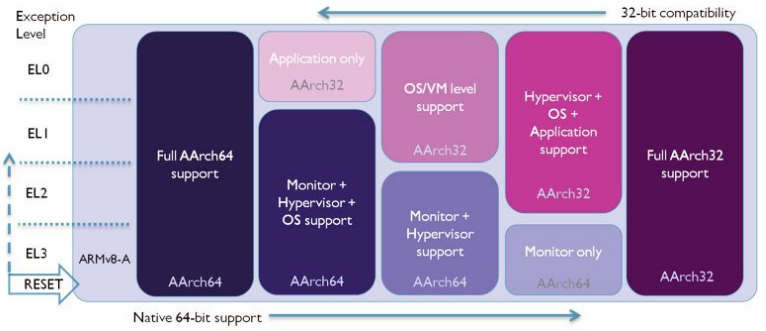
As ARMv8 is a new architecture and not an extension of previous ones, it was agreed to break the required work into three tangible pieces that comprised Kernel, Userland, and Platform. ARM was on hand to provide support to Andrew and Semihalf, and naturally enabled both parties with the required tools—develop-



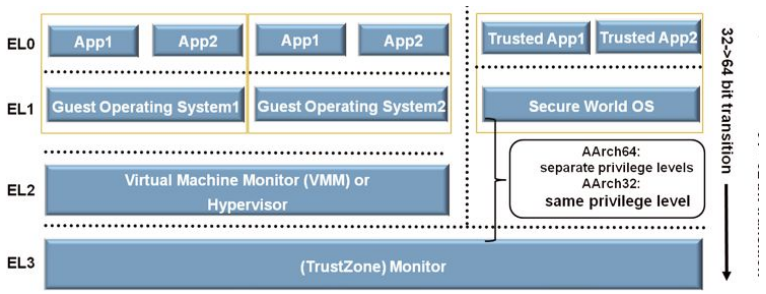
Evolution of the ARM Architecture



Overview of SBSA



ARMv8-A exceptional levels



Hyervisor exceptional levels

ment platforms, debugging, and documentation. As a silicon vendor, Cavium was able to provide Semihalf with access to early development boards and initial development servers to the Sentex colocation facility, enabling more developers access to server hardware.

Vanquishing Runaway Differentiation

This effort was seen as foundational and the basic enabler for the ARM and FreeBSD ecosystems to build upon. As such, ARM wanted to make sure that it was as standards compliant as possible, to help ensure there was no fragmentation (something that ARM was accused of in the past). This meant ensuring support and compliance with the ARM Server Base System Architecture (SBSA), the ARM Server Base Boot Requirements (SBBR), Power State Control Interface (PSCI), and the VM Specification.

The SBSA is a hardware specification for Firmware and OS developers. It enables application developers to target a single OS image for all ARMv8-A systems. OS and Firmware vendors benefit from having a well-defined platform to target, and a single OS image will work across all compliant systems regardless of vendor or micro-architecture.

The SBBR is a follow-on companion to the SBSA. It defines the platform firmware abstractions necessary for OS deployment and boot, hardware configuration and management, and power control of SBSA-compliant systems. It covers items such as UEFI, ACPI, and Trusted Firmware.

PSCI defines a standard interface for power management that can be used by OS vendors for supervisory software working at different levels of privilege. Operating systems, hypervisors, and secure firmware must interoperate when power is being managed; this standard should ease the integration between supervisory software from different vendors working at different privilege levels.

The VM specification is targeted at images for virtualized guests, ensuring that images are portable across hypervisors.

In addition to being compliant with the above specifications, other items included:

- Basic CPU support
  - Initial target of ARMv8, including the Cortex®-A57 and ThunderX microarchitectures
- Environment
  - Toolchain
  - Loader
- Peripherals
  - Timers, GIC (Global Interrupt Controller)
  - Block storage
- ARM locore
  - Assembly macros & definitions
  - Bootstrap
  - Exception handling
  - Cache maintenance and context switching
- VM / pmap
  - TLB maintenance and Page fault routines
  - Conversion of pmap to 64-bit
- 64bit userspace glue
  - Syscalls, signals handling, libraries entrypoint
- SMP
- Dual Socket w/ Cache Coherency

Starting the FreeBSD ARMv8 Port

Work started in earnest just after summer of 2014, and by spring of 2015 it was possible to log into FreeBSD on an AArch64 system. Support was still very basic, and there were some pieces missing. A key component missing was DTrace. Under advice from the Foundation, ARM engaged the University of Cambridge to work on implementing DTrace and Hardware Performance Counter support. This work greatly increased the options for debugging and helped build a more complete FreeBSD story. At BSDCan 2015, Semihalf demonstrated the progress that had been made by showing FreeBSD running on a 48-core Cavium ThunderX platform.

FreeBSD on ARM, prior to the advent of the ARM64 port, relied on u-boot as a key part of the boot flow. While u-boot has the dominant position in this role for the embedded space, UEFI-compliant boot solutions find a more prominent role in the enterprise/networking space. As such, UEFI compliance was chosen as the default boot loader requirement. The FreeBSD loader was ported to be an EFI

application that could be loaded and executed by UEFI-compliant boot loaders.

One of the more challenging areas was in the area of CPU scalability. A lot of blood, sweat, and tears were shed by Semihalf and Andrew Turner in getting Cavium's lowest core count (48) supported, especially when it came to static assumptions in the kernel, which were resolved case by case. Semihalf's Zbigniew Bodek performed a live demo on Cavium's 48-core ThunderX system at the June 2015 FreeBSD developer summit held at the BSDCan conference in Ottawa, Canada.

Together we have made tremendous progress. In the interest of continuing improvement, all parties are still working together to solidify and improve the earlier work. ARM and Cavium continue to work with our partners and the FreeBSD community and sponsor the FreeBSD Foundation.

George Neville-Neil, a member of the FreeBSD Foundation board, said, "The work done to bring ARMv8 to FreeBSD has been a significant undertaking that could not have happened without the support of both ARM and Cavium, who, working with the FreeBSD Foundation, were able to get a brand new architecture up and running in record time. ARMv8 is now a significant architecture within the FreeBSD ecosystem and continues to progress along with the rest of the operating system and its tools."

The Path to FreeBSD 11

We are currently preparing the FreeBSD/arm64 port to be included in the upcoming FreeBSD 11.0 release, addressing the remaining attributes required of a Tier-1 architecture. This includes updates to the documentation, running the FreeBSD test suite and addressing failures, sourcing and installing hardware for the FreeBSD release engineering team and security officer, and validating the install procedures.

The FreeBSD release engineering team produces AArch64 snapshot install media and virtual machine images. Snapshots are announced on the *freebsd-snapshots* mailing list, and are available from ftp://ftp.freebsd.org/pub/FreeBSD/snapshots/VM-IMAGES/11.0-CURRENT/aarch64/Latest/.

For those who would like to test FreeBSD on AArch64, there are a few options. One could use emulation with QEMU. The following steps detail how to go about this:

- Install the QEMU emulator pkg, or build from ports

```
% pkg install qemu-devel
```

- Fetch the snapshot virtual machine image

```
% fetch ftp:// ftp.freebsd.org/pub/FreeBSD/snapshots/VM-IMAGES/11.0-CURRENT/aarch64/Latest/
FreeBSD-11.0-CURRENT-arm64-aarch64.qcow2.xz
```

- Fetch a special version of the UEFI boot firmware

```
% fetch http://people.FreeBSD.org/~gjb/QEMU_EFI.fd
```

- Uncompress the vm image

```
% xz -d FreeBSD-11.0-CURRENT-arm64-aarch64.qcow2.xz
```

- Start the virtual machine

```
% qemu-system-aarch64 -m 4096M -cpu cortex-a57 -M virt -bios QEMU_EFI.fd -serial telnet::
4444,server -nographic -drive if=none,file=FreeBSD-11.0-CURRENT-arm64-aarch64.qcow2,id=hd0 -device
virtio-blk-device,drive=hd0 -device virtio-net-device,netdev=net0 -netdev user,id=net0
```

- Connect to the virtual machine console with telnet

```
% telnet localhost 4444
```

Emulation is fine for some use cases, but nothing beats real hardware. A variety of hardware platforms are becoming available at various price points and capabilities. Inexpensive (under $250 USD) single board systems are available from the 96Boards project, including the HiSilicon-powered HiKey board and the Qualcomm Dragonboard. Cavium, AMD, and Applied Micro provide higher-spec processors and systems.

Given its strong feature set and availability, Cavium's ThunderX platform is the primary hardware reference platform for FreeBSD's ARMv8 port. Instructions for bringing up FreeBSD/arm64 on the ThunderX can be found at https://wiki.freebsd.org/arm64/ThunderX.

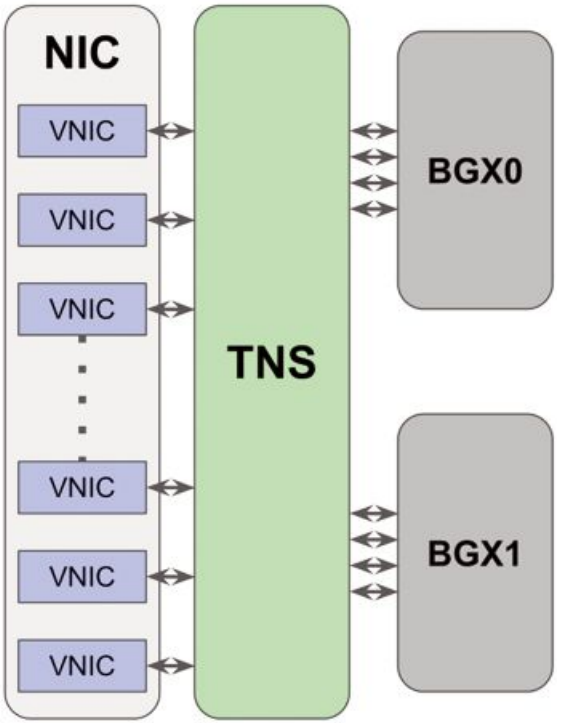
Cavium ThunderX, the Primary Reference Platform

Cavium's ThunderX platform is available in two configurations— a single socket 48-core platform and a dual socket 96-core variant.

ThunderX supports a wide variety of features and peripheral devices and also offers powerful Ethernet capabilities that include 40 Gbps, 20/10 Gbps, and 1 Gbps interfaces. The networking subsystem is partitioned into a few core components:

- IBGX - Common Ethernet Interface
- INIC - Networking Interface Controller
- ITNS - Traffic Network Switch

In the presented order, these implement: MAC layer, Network Interface layer, and hardware switching between the mentioned entities and



High level view of the network subsystem in ThunderX

other on-chip devices. Moreover, NIC is a SR-IOV (Single Root I/O Virtualization) capable device that can incorporate up to 128 Virtual Functions, each providing full network interface features.

In order to support ThunderX networking, FreeBSD introduces drivers for NIC, VNIC, BGX, and MDIO interface (used for the slower connection types such as SGMII for 1Gb Ethernet). The contributed work is available in mainline FreeBSD under sys/dev/vnic/ directory. The core part of the FreeBSD network interface support is located in nicvf_main.c and nicvf_queues.c files that handle

Virtual Function (VF) operations. VFs are able to perform DMA transactions to and from the main memory in order to transfer packet traffic. Each VNIC contains a set of hardware queues (8 send, receive, completion, and 2 Rx buffer) that can be freely used by other VFs if the current one is not enabled. This gives some interesting parallelization capabilities that can be utilized by the OS.

> "A lot of blood, sweat, and tears were shed BY SEMIHALF AND ANDREW TURNER IN GETTING CAVIUM'S LOWEST CORE COUNT (48) SUPPORTED"

The driver was tested with 1, 10, and 40 Gbps connections, and currently the system is capable of saturating 10 Gbps link on Tx path, using 4 CPU cores. The exemplary benchmark that can provide such results is iperf3(1), testing TCP connections using 4 parallel threads. Linerate can be achieved in both copy and zero-copy configurations between userspace and kernel.

This was possible thanks to enabling hardware capabilities such as L3, L4 checksum calculation offloading, TCP Segmentation Offload (TSO), and OS features including Large Receive Offload (LRO). Other Semihalf optimizations seriously improved networking and general I/O throughput. The next step is to further upgrade NIC multicore scalability by enabling additional hardware queues beyond standard sets of 8.

In addition to the ThunderX-based systems, Cavium's OCTEON TX CN80/81XX IOT Gateway/Router and 4-Bay NAS reference designs will be available soon and priced under U.S. $800. These systems are based on the 64-bit ARM-based dual and quad core SoCs that are optimized for low power networking, security, IoT, and control plane applications.

Future Work

There are a number of additional tasks in planning, in order to improve FreeBSD and make the best use of AArch64's capabilities.

1. Improvements to the resource constraints framework, in order to better support OS-level virtualization.

2. Exploration of new scheduling algorithms, taking advantage of FreeBSD's pluggable scheduler. This makes it viable to easily do a completely _new_ scheduler aiming at mobile, for example. The interfaces are also very clean and bereft of all the messy bits that Linux has accumulated over the years.

3. Improved NUMA support in the VM and the scheduler, to improve scalability.

4. Beginning of the implementation of a power management subsystem with more generalized CPU frequency scaling support.

5. CPU count scalability, by profiling bottlenecks using DTrace and hwpmc. This is more from an analysis perspective using some of the "pedigree" tooling that FreeBSD has (such as DTrace for example—SystemTap on Linux is still considered experimental) to figure out how FreeBSD is scaling and where the bottlenecks exist. There is a conversation with David Chisnall and ARM on a similar themed track for Google SoC. Getting analysis is a great first step to then chalk out plans to fix/add features.

6. Power management is still lacking, and ARM has ideas on how best this could be implemented and is discussing these with members of the community. •

Reference Links

Cavium ThunderX Homepage
FreeBSD Foundation Projects
ARM Cortex-A series Homepage
ARMv8 Architecture Reference Manual
Semihalf Homepage

With over 16 years' experience in Open Source, Andrew Wafaa's role at ARM is to ensure that ARM architecture is as well supported within open-source projects as possible. This takes his interactions through the full stack from operating systems to userland applications, and all points in between. A key component is fostering relationships and facilitating getting the job done.