Software Architecture for Next-Generation Multicore Processors
Built on Layerscape Architecture

FTF-SDS-F0253

Rob Oshana | Director, Software R&D, Digital Networking

APR. 2014
Agenda

• Overview of Digital Networking software
• Software development kit (SDK) architecture
• Layerscape software concepts
• Virtualization
• Open source software development model
Digital Networking Software Strategy

Best-in-Class Multicore Software and Debug Solution

Investment in silicon optimized software IP across our Multicore portfolio
- Over 1000 in-house software resources
- Run-time technologies – Application and Platform
- Stand-alone base tools built around standard platforms

In-house resources & IP plus Partners provide open choices for vertical solutions and tools
- Optimized solutions, reference designs and greater application performance
- Peace of mind that software IP will not be locked in
- Freescale Professional Services where needed

Key Software Acquisitions & Investments
1999: Metrowerks
2002: AMC, Lineo
2003: Freescale Professional Services
2005: Seaway Networks
2008: Intoto
2009: MQX Runtime Platform
2010: Processor Expert, Chipwerks
2013: Launch Digital Networking Services

+ Open Ecosystem of Partners
Overview: Software and Solutions for Networking

- Investment in R&D, emerging technologies, acquisitions
- Several production-ready solutions
- Professional services for time to market
- Over 700 software experts in networking domain
Software & Solutions:
Freescale’s Value Proposition
- Comprehensive Solutions for many market segments
- SDK, Tools, Middleware, Acceleration SW for rapid development
- Reference solutions – for time to market
- Production ready software for emerging technologies
- Services and support from experts

Digital Networking Software Solutions

Freescale QorIQ and Layerscape Silicon

- Silicon (QorIQ, LS)
- Enablement SW / Middleware / Acceleration Kits
- Freescale Application Software
- Reference Solution and Open Source Software and Eco-system support

Freescale Application Software
- Enterprise
- Cloud and Data Center
- Wireless & Service Provider

Enablement Software
- SDK
- Middleware
- Acceleration Kits

Tools
- Development Tools

Reference Solutions

Software Support & Professional services
Freescale has the industry’s broadest range of solutions built on ARM® technology for automotive, industrial, consumer and in the future networking applications.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3rd Party</td>
<td>Mentor, Wind River, Greenhills, QNX, IAR, Kiel, Adobe, Bsquare</td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td>Linux, Android, Ubuntu</td>
<td>Founding Member of Linaro, Linux Foundation, Kronos, Continua</td>
</tr>
<tr>
<td></td>
<td>Windows Embedded</td>
<td></td>
</tr>
<tr>
<td></td>
<td>RTOS: MQX, QNX, FreeRTOS…</td>
<td></td>
</tr>
<tr>
<td>Tools</td>
<td>Code Warrior, GNU, 3rd Party</td>
<td>ARM Compiler Technology</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Custom/Segment ARM Tooling</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eclipse: Processor Expert, DDR Tuner, JTAG/COP, etc</td>
</tr>
</tbody>
</table>
Software Development Kit (SDK) Architecture
Next Generation Multi-core Platform: Layerscape

1- Core Agnostic (ARM, Power Arch)
   • ARM V8 Product Roadmap
   • Small / Large footprints

2- Scalable Acceleration Elements
   • Sized to Application Needs
   • Turn key or C-programmable
   • Wire rate I/O switching & TM

3- Ease of Use
   • Real Time Monitoring / Debug
   • SW Management utility
   • I/O virtualization

4- Turn-key Software
   • Fast path modules
   • Linux / BSP
   • Hypervisor: KVM
   • Eclipse-based Tools

Layerscape: Networking 64b Multicore SoC Platform
a) Industry standard Tools & C-programmability
b) Abstracts I/O and Acceleration
c) Turn-key / Production-quality SW
Base Software Development Kit

- Targeted for General SW development (a.k.a. everyone)
  - May include NW Applications
- Everything is upstream
  - No deviations
- Don't invest in middleware
  - Don’t force HW features into middleware framework (QoS, LAG)
  - Don’t add new middleware (e.g. ASF)
  - Optimize within bounds of middleware
- Invest in drivers
  - Ensure F-Lib compliance
  - Ensure middleware compliance
- Evaluate and migrate non-compliances to user-space – e.g. ASF, L2-switch
Layerscape Software Concepts
Isolated interconnect

Layerscape offers **efficient, isolated, high-functionality interconnect** between separate control / management planes, data plane, the services / functions, and physical network. This offloads pure SW-based interconnect.

Augmented virtual switching and network virtual functions

- Shared network access available to applications/containers via virtual network interfaces
- Bump-in-the-wire offloads available as inline accelerated offloads on switch ports (VNIs)
- Fully virtualized access to datapath offloads

* Container may be Linux containers or some other form of VM.
Accelerators in Network Virtual Functions

- New functions may be items such as IDS/IPS that could benefit from accelerators because they are decoupled from IOS functionality and are less monolithic.
- Layerscape provides virtualized and isolated access to accelerators: SEC, PME, DCE.

* Container may be Linux containers or some other form of VM.
Layerscape I/O Virtualization / Resource Management – Layerscape Object Abstraction

Hardware based virtual switching and NICs via LS Objects: SW mediated encapsulations of HW resources for a specific task.
Mgmt Complex-- Objects Formed from HW Resources

L2 Switch Example

Mgmt Complex composes complex “atom” resources into easier objects that abstract underlying HW.

Runtime View– Management Complex and Framework Objects

Mgmt Complex

Initial object Set

Boot time

API object discover
API object create
API object destroy
API object Manage
API object use

API
API
API
API

GPPs

AIOPs

API object use
Standard Linux Includes Virtualization

Virtualization support in standard Linux
- KVM
- Containers
- Access isolation
- Performance isolation (via name spaces, cgroups, etc)
- Direct assignment of devices for performance
  - To containers
  - To KVM guests
  - To standard user space processes
- Isolation via IO-MMU

HW devices can be directly accessed by containers, VMs, and processes— not just the host kernel. It means load/store to device, use DMAs, etc.

Increased performance via host kernel bypass.
Programming Layerscape

- GPPs run OS like Linux and are, programmable in C and many other languages
- Freescale provide the LS objects and many APIs
- AIOP is programmed in C.
- Freescale provides ready-to-use software, but it is open.

• GPPs run OS like Linux and are, programmable in C and many other languages
• Freescale and OS-provides APIs
• Drivers use LS objects

• AIOP has many built-in accelerators such as for table lookups, ordering and atomicity, timers, frame and context presentation, statistics, etc.
• Accelerators accessed via simple synchronous API calls.
• Service layer is like the OS layer, but the AIOP’s scheduler is in hardware. It does not run a conventional OS.
Packet Processor (AIOP) may be used in three ways

1. Can be a resource used to create a Layerscape object.
   - Entirely transparent to GPP software; AIOP “owned by management complex”
   - Example: Advanced NIC
   - AIOP is black box.

2. Vortiqa-Datapath Applications Kit product
   - GPP sees AIOP as an engine running a specific offload image.
   - GPP has control API for the image
   - AIOP and GPP have network interfaces (via LS objects)
   - AIOP is white box. Software is ready-to-use, but customers have access.

3. Program via service layer and using objects
   - AIOP is open
   - Customer provides AIOP image, programming in C above the service Layer.
## Layerscape Packet Processing Modules

<table>
<thead>
<tr>
<th>Package Name</th>
<th>Principal Data Acceleration Kit Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Routing</strong></td>
<td>IP FWD</td>
</tr>
<tr>
<td><strong>SDN/Openflow</strong></td>
<td>Openflow-DP</td>
</tr>
<tr>
<td><strong>Switch Supplement</strong></td>
<td>IP FWD</td>
</tr>
<tr>
<td></td>
<td>BFD</td>
</tr>
<tr>
<td><strong>Wireless access</strong></td>
<td>IP FWD</td>
</tr>
<tr>
<td></td>
<td>CAPWAP</td>
</tr>
</tbody>
</table>
Virtualization
Freescale and ARMv8 Virtualization Technologies

Virtualization technologies, including KVM and Linux Containers (LXC) will be fully supported on Freescale’s Layerscape platforms.

- **KVM**
  - GICv3 next gen interrupt controller. Support for message based interrupts and KVM.
  - Linaro provides PMU support for measuring the performance of guest OSs

- **PCI EXPRESS**
  - Build in Hardware Virtualization

- **VFIO/IOMMU**
  - Support by Linaro for device virtualization in emulated environments

- **CoreLink**
  - Unique Freescale Network Architecture for Packet Offloading
  - SMMU support by ARM providing memory virtualization services in hardware reduces the software interventions needed, minimizing hypervisor overheads and ensuring system performance approaches the optimum.

- **LAYERSCAPE**
  - Freescale's Layerscape platforms support virtualization technologies, including KVM and Linux Containers (LXC).
Freescale – Software Virtualization Technologies

- Our strategy is to enable and offer standard Linux-based virtualization technologies across Power and ARM based SoCs – with superior I/O capabilities
Device Passthrough in KVM

- Device Passthrough-- assigns a physical I/O device to a virtual machine
- Freescale did substantial proof of concept work around device passthrough in 2010-2011
- In 2011-2012 a new framework called vfio was proposed by Red Hat for doing device passthrough
- vfio-pci was accepted into upstream kernel in 3.6
- Freescale is currently working on a vfio solution for doing this for platform/memory-mapped devices
- USB
  - QEMU currently supports passthrough of USB devices to VMs
- Note: currently all interrupts are mediated by the kernel
Comparison of Processor Virtualization Capabilities

ARM, Power, x68 architectures all support similar mechanisms to support virtualization.

<table>
<thead>
<tr>
<th>Capabilities</th>
<th>ARM</th>
<th>Power</th>
<th>x86</th>
</tr>
</thead>
<tbody>
<tr>
<td>3rd privilege level</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Extended Address space</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Hardware guest physical address translation (2-stage)</td>
<td>Yes</td>
<td>(LRAT)</td>
<td>(EPT/NPT)</td>
</tr>
<tr>
<td>Direct guest interrupt management</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes (x2APIC)</td>
</tr>
<tr>
<td>IOMMU</td>
<td>Yes (SMMU)</td>
<td>Yes (PAMU)</td>
<td>Yes (VT-d)</td>
</tr>
</tbody>
</table>
Freescale 64 bit ARM migration

No ‘migration’ needed. Freescale QorIQ SDK based on platform independent opensource ecosystem. Software based on QorIQ SDK will implicitly support PowerPC 32/64 bit and ARM AArch32/64 bit platforms with a minimum of changes.

Freescale is a Linaro member

Linaro is the place where engineers from the world's leading technology companies define the future of Linux on ARM. The company is a not-for-profit engineering organization with over 200 engineers working on consolidating and optimizing open source software for the ARM architecture, including the GCC toolchain, the Linux kernel, ARM power management, graphics and multimedia interfaces.

Linaro is making toolchains and software images for AArch64 (the 64-bit execution state of ARMv8) available to interested developers.

Freescale is a Yocto Project member

The Yocto Project is an open source collaboration project that provides templates, tools and methods to help you create custom Linux-based systems for embedded products regardless of the hardware architecture. It was founded in 2010 as a collaboration among many hardware manufacturers, open-source operating systems vendors, and electronics companies to bring some order to the chaos of embedded Linux development.

Linaro and the Yocto Project are working closely to create full Linux distributions based on ARMv8.
Freescale 64 bit ARM migration

Migration is greatly simplified by leveraging all our ecosystem partners.

Continually upstreaming and targeting SDK to latest upstream releases. Always capturing latest Linux features/support.

Yocto targets for ppc32, ppc64, ARMv7, ARMv8 Freescale PPC toolchains, Linaro ARM toolchains

ARM 32 and 64 bit toolchains, ARMv8 Linux support, ARM technology support including SMMU, ARM-A53, ARM-A57, Device Trees, Timers, KVM (IOMMU, PMU, VFIO, Migration, GICv3), Kexec, PCIe, klib, kgdb, hugetblfs, endianness.

U-boot bootloader for powerpc and ARM. Freescale added ARMv8 support to U-boot.

PPC32 and 64 bit toolchains created by FSL, e6500, e5500, e500 32/64 bit SoCs, Soc device support and KVM all developed by FSL

Single release of multiple platform support for ARM and PPC targets, with FSL, Linux, Linaro, and Yocto content.
SDK Upstreaming Model

As content is upstreamed, different types of content are included:
- **IP Content (ex. ASF)**
- **IP Content (ex. DPAA)**
- **IP Content (ex. Private Optimizations)**
- **NPI Content (ex. LS2100 BSP)**
- **Additional Features**

QorIQ Features and NPIs:
- **Feature Content**
- **Configuration Settings**
- **Repository / Branch / Patches**

Upstream Kernel: + = QorIQ Kernel

**Feature Merge**
**Feature Integration**
**Performance Engineering**
SDK Development Flow

1. **Upstream Repository**
   - Continuously Upstreaming
   - IP Patches

2. **IP Repository**
   - As content upstreamed, removed from IP repository

3. **IP Development**
   - NSSG
   - Simulation and Virtual Machines
   - Unit and Regression Testing

4. **Yocto Project**
   - Linaro
   - Yocto Layers and Recipes
   - meta
   - meta-yocto
   - meta-yocto-qorIQ

5. **Systems Engineering**
   - Integration Test
   - System Test
   - Performance and Systems Integration
   - 5500 system tests
   - 100 boards

6. **External Repository**

7. **Feature Delivery (GERRIT)**
Freescale Linux Software Development Kit

Freescale Linux SDK is a complete Linux development environment (Linux distribution)

- Based on industry standard Yocto/Poky.
- Embedded-style (cross-compilation but native tools also provided)
- Source code provided

Linux SDK main contents:
- GNU tools
- Package system
- Build System
- Kernel source
- Bootloader source
- Package sources
- Hypervisor package sources
- Freescale Network SW packages

Generates

- Bootloader image
- Kernel image
- Customizable file system
- Hypervisor images (optional)
- Freescale optimized package images (optional)

Everything needed to boot and run Linux