Google Project ARA
Power Management Challenges
About the Power Management of a Modular Platform

Patrick Titiano,
System Power Management Expert,
BayLibre co-founder.
www.baylibre.com
Introduction

- Smartphones only available in one-size-fits-all configurations.
- Google Project Ara redefining that by creating a Linux-based platform where consumers may assemble the device they like from just the modular components they need.
- Providing such flexibility comes with many power management challenges
  - Modular components are separate from the "main" phone, hotpluggable.
  - Platform power consumption cannot be pre-characterized/optimized
- New ways must be designed to incorporate management of their power into the existing Android/Linux infrastructure.
  - "How do we do runtime power management of a module?"
  - "How do we ensure there's enough power to supply to a module added dynamically?"
  - "How do we protect the platform from malfunctioning modules?"
  - "How we balance modules communication bus power / performances?"
- ...
Menu of the Day

- ProjectARA basics
- High-Level ProjectARA Power Management Architecture
- ProjectARA Power Management Challenges & Solutions
  - Module Runtime Power Management
  - Unipro Link Power Management
  - Module Idle Power Management
  - Module Power Allocation
  - Module Over-consumption Protection
  - Module Thermal Management
- Q & A
Scope

- **Includes**
  - How to optimize the power consumption of the Ara subsystem, including
    - The Ara modules,
    - The Supervisory Controller (SVC),
    - The UniPro network (Switch, Bridges).

- **Excludes**
  - Application Processor (SoC) power management
    - Identical to regular Android smartphones.
  - Battery Management
ProjectAR A Basics

Technologies, terminology, ...
What is Project ARA?

A **modular device platform** that:
- Acts as a phone, but supports add-on modules
- Detects module insertion and controls module removal
- Manages power to modules independently
- Provides a UniPro network, balancing power and performances
- Uses a manifest to describe module capabilities
- Leverages online sources to support modules if required
Unipro

- High-speed interface technology for interconnecting integrated circuits in mobile electronics
  - Bidirectional multi-lane links, up to 5 Gbps per lane
- Low-power (links can run at lower data rates to reduce power consumption)
- Designed for low pin count, small silicon area, data reliability and robustness
- Used in Project Ara to interconnect modules
Greybus

- Designed for ProjectAra to abstract communication with Modules
- Defines messages sent over connections between modules
- Supports “operations” that pair a request and response message
- Protocols define sets of operations a connection supports
- Protocols implement classes of device behavior
- Modules advertise classes they support in a "manifest"

- https://github.com/projectara/greybus-spec
- https://github.com/projectara/greybus
Endoskeleton

- Rigid substrate, but as lightweight as possible
- Physical guides hold modules in place
- Electrically controlled mechanism prevents removal
- Slots available in several module sizes (1x1, 1x2, 2x2)
- Each slot has an electrical “interface block”

Figure 2.1 - Ara Endo (Medium Variant)
Supervisory Controller (SVC)

- Part of the EndoSkeleton
- Module maintenance
  - Insertion/Removal,
  - Power control,
  - Power monitoring
- Unipro Network Management
  - Switch Control
  - Connection Establishment
Modules

- Different sizes: 1x1, 1x2, 2x2
- Implements 1 or more feature(s)
  - Camera, Speaker, Sensor(s), ...
  - 'Bundle' as per Greybus terminology
- May have 1 or 2 interface block(s)
  - 1x1, 1x2 modules: 1 interface block
  - 2x2 modules: 2 interface blocks
  - 'Interface' as per Greybus terminology
- Includes a Unipro Bridge chip
- Runs NuttX RTOS
High-Level Project ARA
Power Management Architecture
Ara Power Management Architecture
Module Runtime Power Management (1)

- Dynamically power ON/OFF/Suspend a given module, depending on its usage.
  - ON state: module required to execute the active use-case(s),
  - OFF state: module not required to execute any active use-case,
  - SUSPEND state: module not required to execute the current use-case(s) but
    - Module’s state shall be maintained,
    - Power OFF transition latency not matching performance requirements.
- Always driven by use-case (power transitions happen at use-case boundaries only),
- Always initiated by the Application Processor, never by the module.
- Modules Greybus Drivers integrate the standard Linux RuntimePM callbacks
- Leverages dynamic driver loading capability of Linux Kernel
Module Runtime Power Management (2)

- Available:
  - Linux RuntimePM Framework
  - Linux Dynamic Driver Loading

- To Be Done:
  - Greybus RuntimePM Operations
  - Greybus Interface & Bundle Power States
  - Greybus Interface & Bundle Power State Transitions & Dependencies
  - Greybus Interface & Bundle Driver RuntimePM callbacks
  - SVC support for Greybus RuntimePM Operations
  - Module FW support for Greybus RuntimePM Operations
Module Power Allocation (1)

- To protect the Ara platform from brownout
- Make sure that at any time, platform has enough power available to:
  - Supply a new module,
  - Supply a new processing on a module,
  - Eject a module
- Policy aggregating power allocation requests from Greybus drivers
  - Running on AP
  - Constantly monitoring the battery level
    - Power budget decreasing with battery level
- SVC monitors allocated power budget not exceeded by modules.
  - HW-shutdown of faulty modules
Module Power Allocation (2)

- **Existing:**
  - Nothing

- **To be done:**
  - Power Allocation Framework
  - Greybus Module Power Allocation Operations
  - SVC Power Monitoring/Limitation Support
  - Module Power Consumption Characterization
  - Module Power Consumption Profiling
  - Policy (incl. optimization)
Module Over Consumption Protection (1)

- To protect platform from module(s) that may draw more power than allowed
- Shut down module power source ASAP.
  - HW-driven, no SW involved.
- Leverage always-on power monitoring devices
- Leveraged by Module Power Allocation FWK
  - Dynamically adjusting module max power limit
Module Over Consumption Protection (2)

- Existing:
  - Nothing

- To be done:
  - SVC FW Power Monitoring support
    - Nuttx driver for Power Monitoring Chips
    - Power Monitoring FWK
  - Greybus Power Monitoring operations
  - Integration with Module Power Allocation FWK
Module Idle Power Management (1)

- To minimize the power consumption of a module while it is not doing any processing, but required by an active use-case.
  - Transitions a module (interface(s) / bundle(s)) into a low-power state between two processing tasks.
  - From RuntimePM perspective, module remains in ON power state.
- May include clock gating, clock and voltage scaling, power switching
  - Depending on module architecture and capabilities.
- Managed by module FW, without awareness of the Application Processor (AP) kernel (Greybus driver).
Module Idle Power Management (2)

- **Available:**
  - Module RTOS (NuttX) Power Management Framework

- **To be done:**
  - Module Idle States
  - NuttX Power Management Framework callbacks
  - Module Idle Power Consumption Optimization
SVC Idle Power Management

- Similar to Module Idle Power Management
- Reduce the SVC power consumption while it is not processing any greybus operation or event.
  - SVC Idle 99% of the time
- May include clock gating, clock and voltage scaling, power switching.
- SVC still able to detect any new events/requests, even in a lowest power state.
- Driven locally by the SVC itself, without awareness of the Application Processor (AP) kernel.
Unipro Link Power Management (1)

- Dynamic scaling of UniPro link “power mode”, depending on
  - Active system use-case performance requirements,
  - Module data movement profiles
    - Latency-bound, bandwidth-bound, jitter tolerance, ...
    - E.g. Speaker module more sensitive to latency than bandwidth jitter
- Includes heuristic(s), aggregating Unipro performance requests issued by module drivers and other system policies
  - E.g. also used for Thermal Management
- Managed by AP kernel and SVC FW only, modules not involved
- Shall not degrade user experience.
Unipro Link Power Management (2)

- Existing:
  - Nothing

- To be done:
  - Unipro Link Power Management FWK
  - Greybus Link Power Management operation
  - Greybus Bundle Driver integration
  - SVC Link Power Management support
  - Module data movement profiling
  - Heuristics (incl. optimization)
To dynamically reduce the power consumption of the UniPro Switch depending on UniPro network configuration,
- Gating unused UniPro ports,
- Hibernating the device when the Ara platform is suspended,
- Powering down the device when the phone powers down.

May include clock gating, clock and voltage scaling, unipro hibernation.

Driven by the SVC solely, without awareness of the Application Processor (AP) kernel.
Unipro Switch Power Management (2)

- Existing:
  - Nothing

- To be done:
  - SVC FW Support for Switch Power Management
Thermal Management (1)

- Mounted on the back of the phone, modules shall not warm too much and harm user

- Model frame and modules as thermal zones
  - Modules may include thermal sensor

- Model Unipro Links and modules as cooling devices
  - Dynamic scaling of Unipro Links Power Mode (speed)
  - Dynamic power switching of modules

- Modules may include local thermal management
  - FW driven, without kernel awareness

- Leverages the Linux Kernel standard Thermal Management FWK
Thermal Management (2)

- Existing:
  - Linux Thermal Management Framework

- To be done:
  - Greybus Thermal Management operations
  - Module FW Thermal Management support
  - Integration with Linux Thermal Management FWK
  - Integration with Unipro Link Power Management FWK
  - Thermal Policy optimization
Thank you!