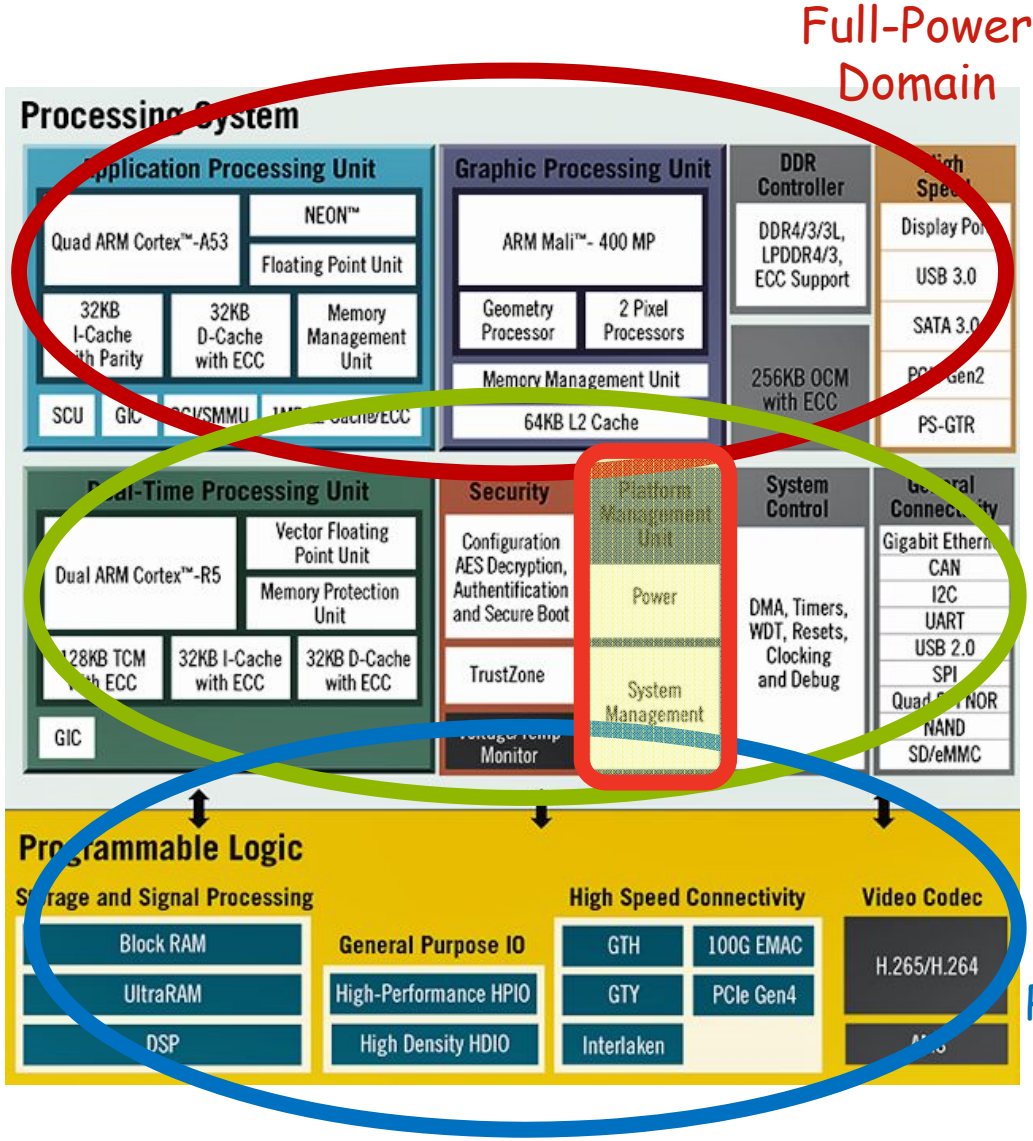
System-Level Power Management Requirements

Ahmad Ansari

Introduction

- **Power and Energy Management is of great interest to Xilinx**
  - Our products include large FPGAs and System-On-Chip solution
  - Products are used in application domains that are power sensitive
- **Devices are becoming larger**
  - Include components such as processors, accelerators, peripherals, memory subsystems, and interconnection networks that typically have well-specified power modes
- **We rely on third-party IP, tools, and software**
- **There is requirement for effective board-level designs**
  - Accommodate management and efficient distribution of power
  - Support power testing and measurement
- **System-level Hardware/Software Integration is becoming more challenging**

Xilinx Zynq MPSOC



- **Two Major Processing Subsystem Clusters**
  - All can potentially run independently of each other
  - No off-the-shelf OS can handle Power Management at the System level
  - Programmable Logic allows for arbitrary additional system components
  - TrustZone security
  - Virtualization
- **Power Management**
  - Converges as a single entity which has holistic system knowledge

Power Management Attributes

- Power Supply Sequencing during Power up
- Power Switch Topology and Sequencing for Power Islands
- Power domain Isolation Strategy
- Clock and Reset Architecture/Sequencing
- Interrupt Processing and Delegation
- Physical Dependency between Modules
- Functional Dependency between Blocks
- Initialization/Configuration of Blocks post power up
- Power/Performance Monitor
- Power Regulator Control
- Software/Control Stack
  - ROM ↔ Firmware ↔ Power Protocol Manager ↔ Hypervisor ↔ OS
  - Communication mechanisms across different software layers

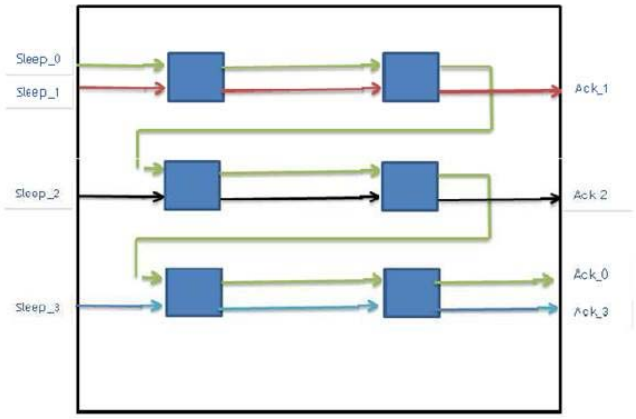
Power Supply Sequencing between Modes

- **To enter or exit a power mode, Power Supplies may require to be turned on or off in a specific sequence**
  - To prevent latch-up in certain IP Macros
  - To eliminate unnecessary power draw during transitions
  - To simplify sequencing of certain functional signals such as resets
  - Other
- **PMU with potential help from External Power Supply Controllers is responsible for proper sequencing of the supplies**

Supply Group	Power Off	Full Power Mode		Low Power Mode		Deep-Sleep Mode	
	On/Off	Sequence	On/Off	Sequence	On/Off	Sequence	On/Off
LP Supply	Off	1	On		On		On
FP_Supply	Off	1	On	2	Off		Off
Aux Supply	Off	2	On		On		On
PLL_Supply	Off	2	On		On	1	Off
DDR_PLL_Supply	Off	2	On	2	Off		Off
DDR_IO_Supply	Off	3	On	1	Off		Off
IO_1	Off	3	On		On	1	Off
IO_2	Off	3	On		On	1	Off
IO_3	Off	3	On		On	1	Off
IO_4	Off	3	On		On		On
Battery Supply	On		On		On		On
High_Speed _IO	Off	2	On	2	Off		Off

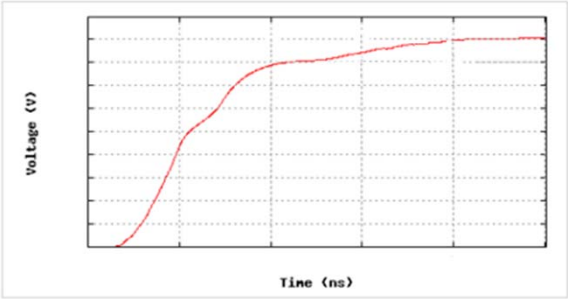
Power Distribution and Switch Topology

- **On-chip or Off-chip power switches**
  - Off-chip switches are used to control power to domains
    - Merging/Separation of the rails may have system-level or Noise/Performance impact
  - On-chip switches control power to islands

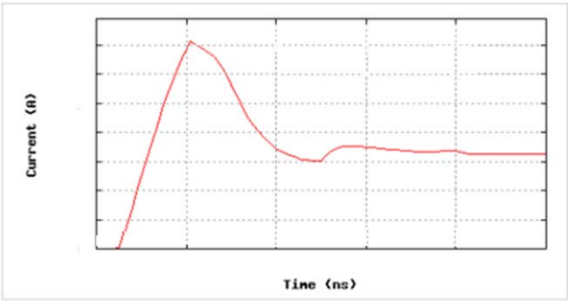


- **Topology of the on-chip switches**
  - Multiple groups of power switches are used to power up an island in order to provide fine control over inrush current, power up latency, and more balanced charging of the island to minimize power waste due to crowbar effects.
  - Nature of header-cells in the library impacts the topology of the switches (Ex: Mother/daughter cells)
  - Sequencing of the chains is done by the PMU which applies proper delays between enabling different chains to optimize inrush current and power-up cost.

Worst ramp up/down switch

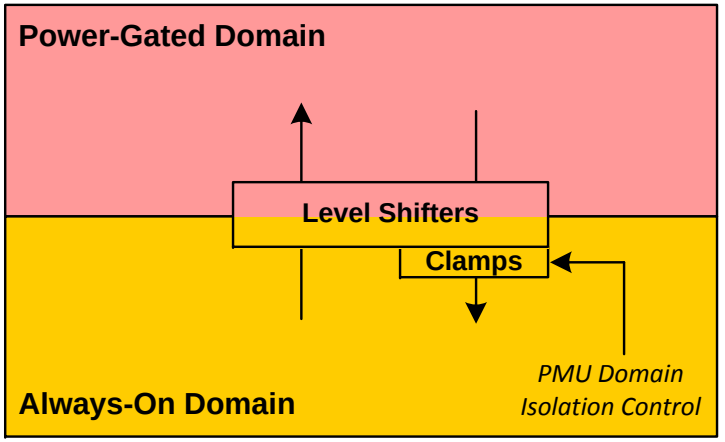


Internal Domain Current Waveform

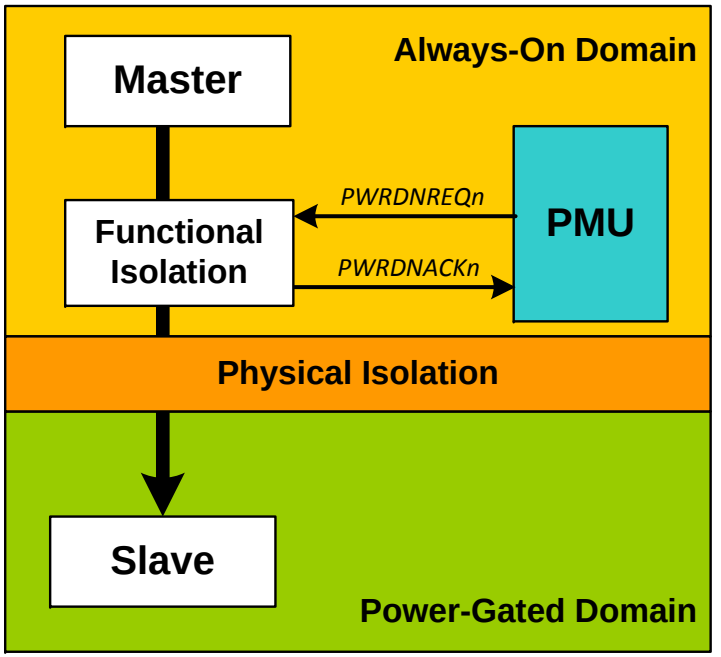


Power Domain Isolation Strategy

- **Physical Isolation during Power down**
  - Prevent both Electrical and Functional Issues during power-down of a domain
  - Include Level-Shifters if domains have separate supplies
  - Clamp outputs of the domain that is going to be powered down



- **Functional Isolation prior to Power down**
  - Consist of Bridges that flush outstanding transactions on the buses/interconnects
    - Upon Request from PMU, reject new transactions to the domain that is going to be powered down and generate errors
    - Generate Acknowledge to the PMU when there are no outstanding transactions

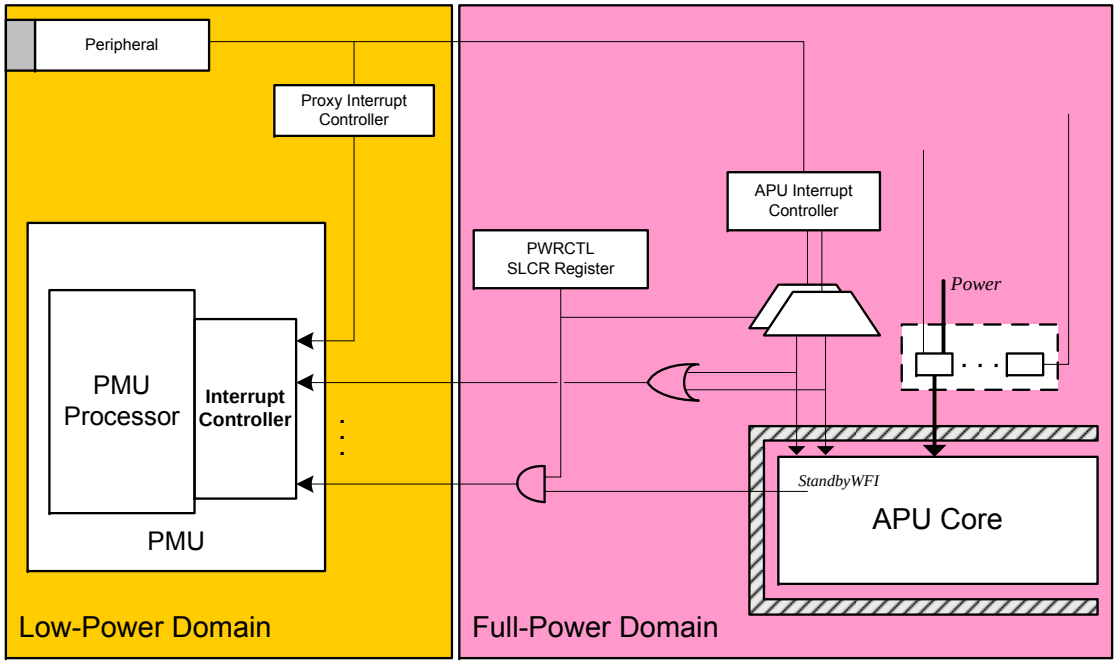


Clock and Reset Requirements

- **Clocks are turned off when blocks go to suspend state**
- **Clocks are slowed down when blocks go to low-power operational states**
- **Reset is asserted before Blocks are powered up and unisolated**
  - Reset is not applied prior to the power down in order not to waste power
- **When a block is powered up, clocks typically need to become available before block is unisolated.**
  - Allows propagation of reset in the block
- **Some Resets are released by the PMU while processing the Wake**
  - Reset to the masters are typically processed by the PMU
  - Reset to the interconnects and Local Clock/Reset modules are released when the domain is powered up
- **Release of the Reset to peripherals is usually done by the higher-level code executing on the processor that owns the peripheral**

Interrupt Delegation during Power down

- **Mode 1 – Processor Core is down, the rest of the subsystem is up**
  - Processor Core directs the interrupts from the main Interrupt controller to the PMU
  - Executes a WFI to initiate a power down request
- **Mode 2 – Processor Core is down, the rest of the subsystem is also down**
  - Prior to power down, Processor Core executes the following:
    - Sets up a Proxy Interrupt Controller in the Always-On domain to direct Wake interrupts to the PMU
    - Programs the Recipe for powering down the entire Full-Power Domain
    - Executes a WFI instruction to request the power down of the entire domain



Physical and Functional Dependency

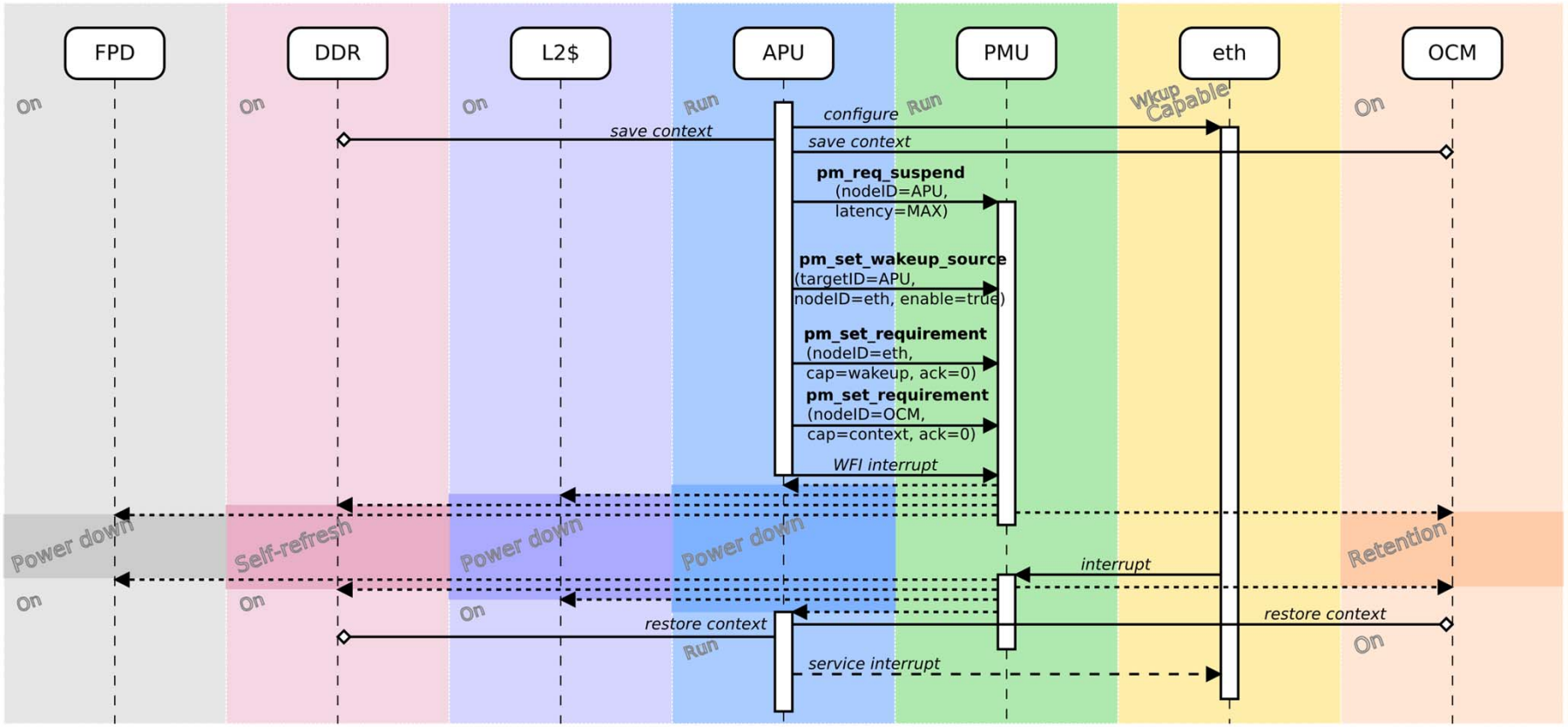
- **Physical Dependency between modules**
  - Two independent blocks may coexist in one island or power domain but belong to separate subsystems. When one subsystem is entering low-power state, the power-down of the specific island is delayed until both subsystems are ready.
    - Example: DDR Controller is in the Full-power domain but can be shared between APU and RPU subsystems. In that case, Full-power domain cannot be powered down even when APU is in the hibernate state unless RPU is also powered down.
- **Functional Dependency between Blocks**
  - A block for its operation may require another block.
    - Example: Boot Code for the processor is located in a RAM that is in the Retention State. Therefore, power up of the processor required the RAM to be available before reset to the processor is released.

Initialization of Blocks Post Power up

- **Blocks need to be initialized after power up before they are used**
  - Clock and Reset Programming
  - Configuration of the Operational Parameters
  - Configuration of the Processors
    - Boot Mode
    - Locked-Step or Split
  - Potential Loading of Boot Code into RAMs that were powered down
  - Configuration of the Interconnects
    - Protection/Security
    - Functional Isolation States
  - IO Configuration
    - Protocol/Voltage
    - Termination, etc.
- **Configuration of the Programmable Logic**

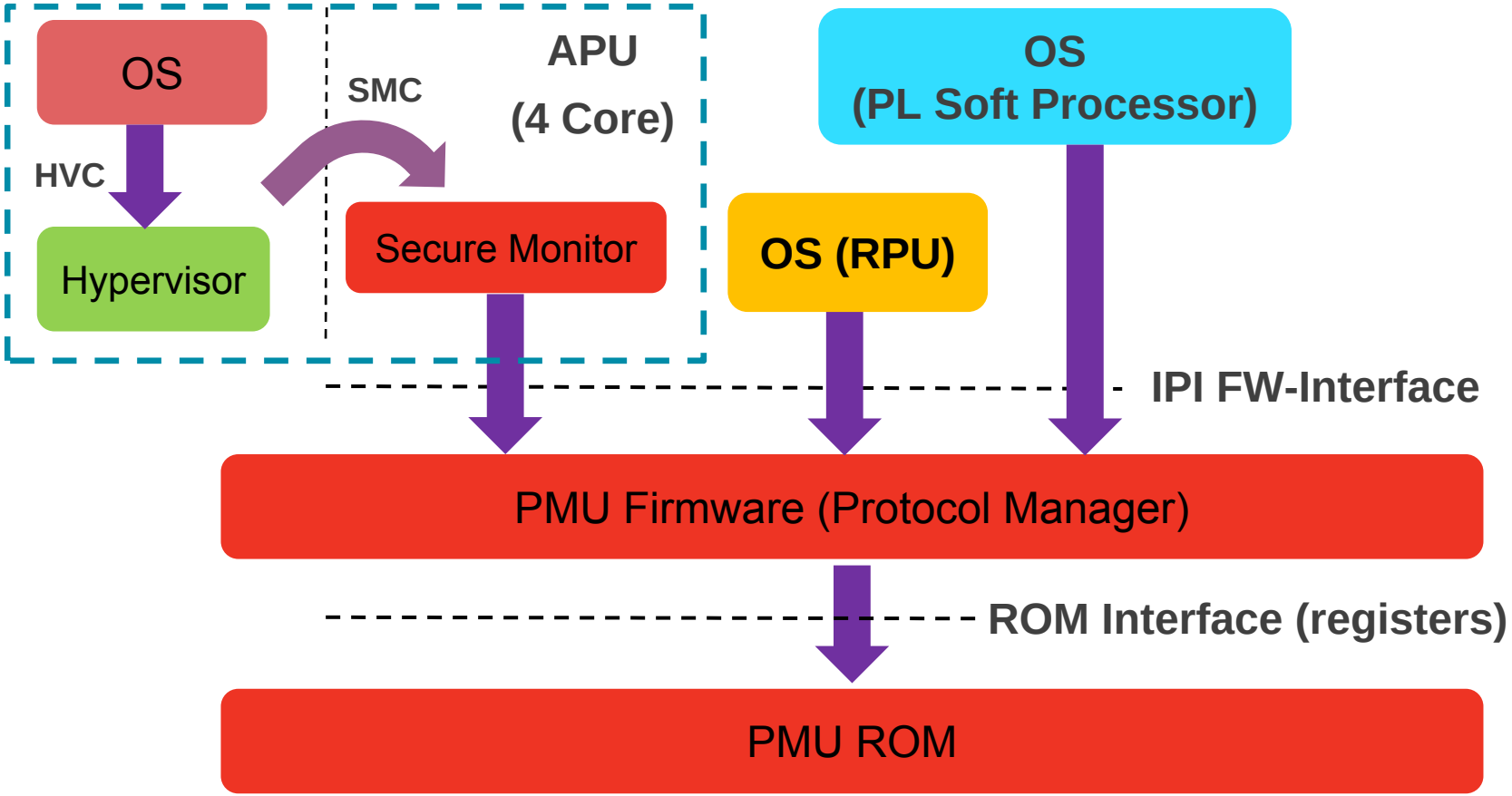
Example: APU Power-down with Wake on LAN

- Ethernet Device is owned by the APU
- Power-down of the Full-power domain and APU is triggered by the APU
- Wake Event is passed through Proxy Interrupt Controller to the PMU which would trigger the Power up of the FPD, including the APU, and release of the reset to both



Software Power Management Control Layers

- ROM ↔ Firmware ↔ Power Protocol Manager ↔ Hypervisor ↔ OS
- Standard Communication protocols at different levels
  - Memory mapped Interfaces to ROM functions
  - Inter-Processor Interrupt Interfaces (IPI) at higher levels



Conclusion

- System-level Power Management involves many tasks
- Some of these tasks such as control of the power switches to islands and physical isolation are defined directly by the power intent description of the design
- Many other tasks manifest themselves at the higher level due to physical/functional dependencies of the blocks, functional integrity, and system-level protection.
- It is critical to create power models that can accommodate both groups and can operate at different abstraction levels. This would allow both hardware and software use a common model providing interoperability among all tools an IP.