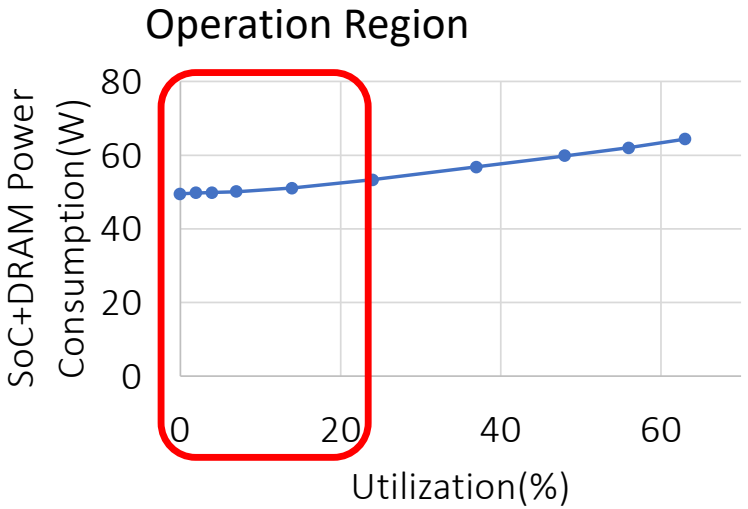
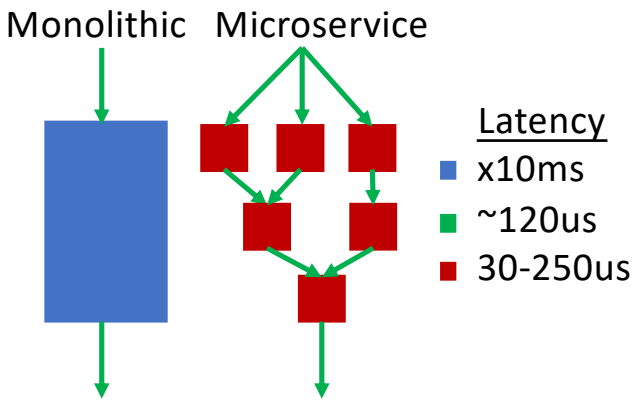
# AgilePkgC

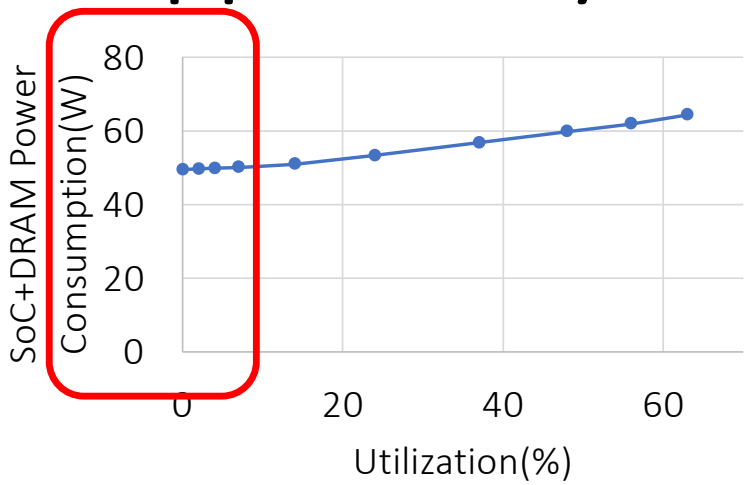
# Datacenters are not Energy Proportional

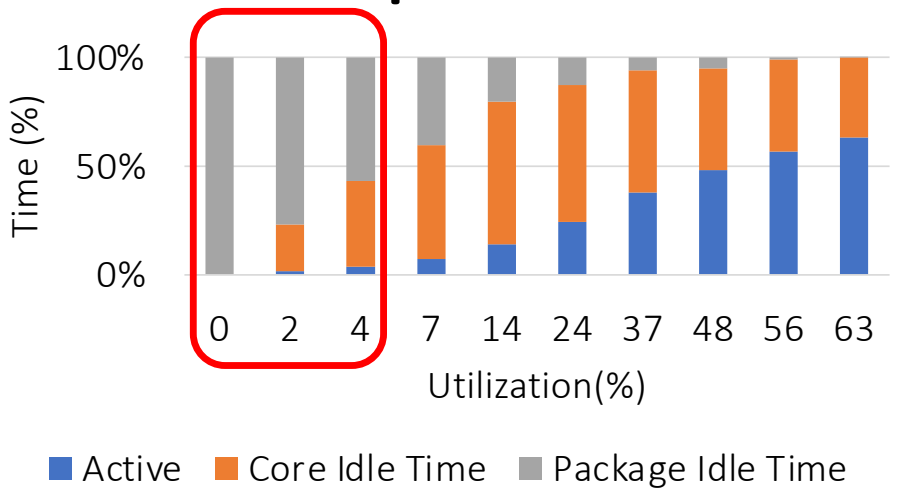
- Datacenters not energy proportional for low utilization
- Cloud Applications microservice software architecture
  - Strict Latency Requirements (30-250us)
    - "Killer Microsecond Problem"
  - Low Average Utilization (5-20%)
  - Increase Idle Opportunity
- Power Saving Techniques
  - High Transition Latency (10 - 133us)



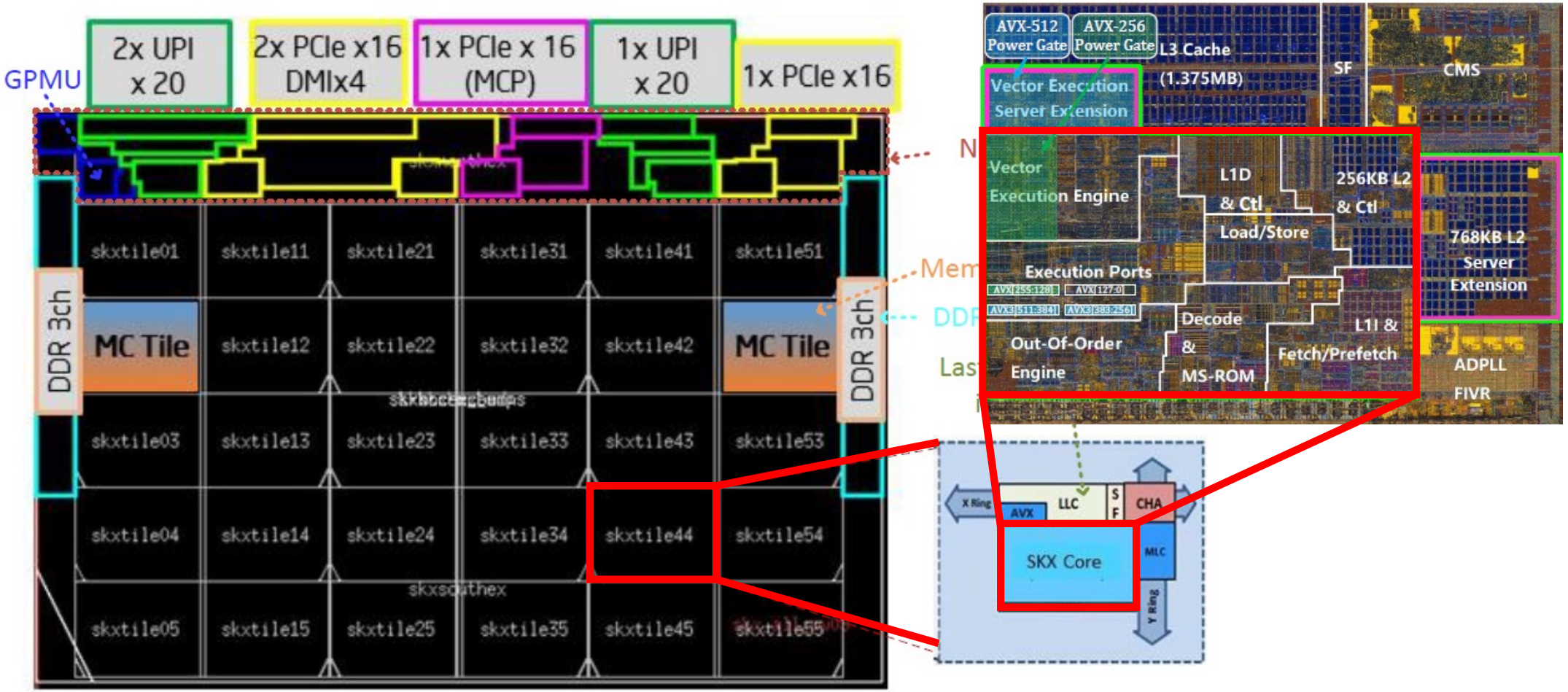
# Idle Opportunity vs Power Consumption



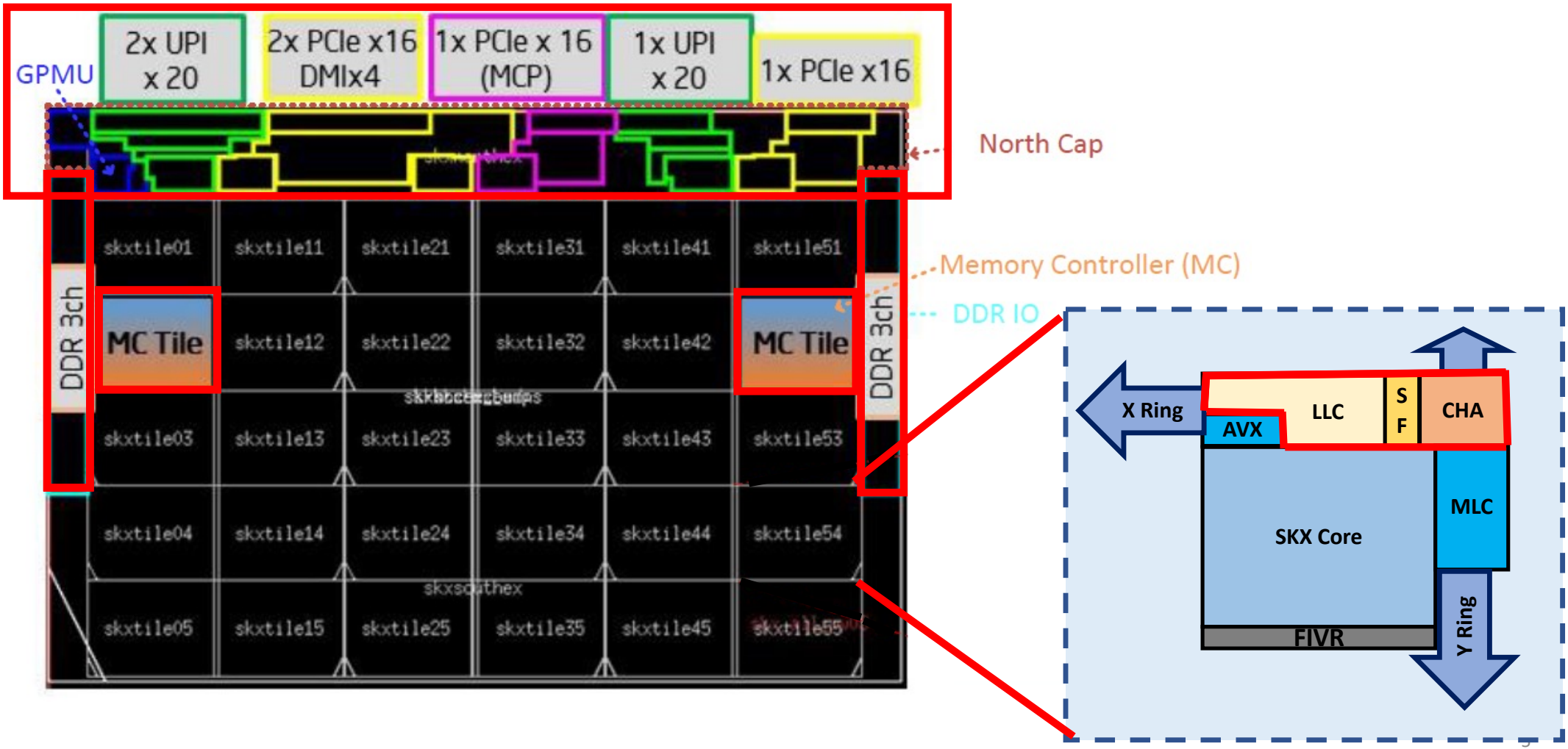




- Power Hungry Resources remain active even when not needed
- Power usage of main subsystem for a server
  - CPU + DRAM ~50% of idle power consumption
  - “Energy Proportionality … cannot be achieved through CPU optimization alone”
- AgileWatts : Energy Efficient Core Idle State Architecture *[HajYahya_MICRO2022]*
- This Work : Energy Efficient Package Idle State Architecture (Uncore + DRAM)

# AgileWatts Area of Focus



# AgilePkgC Area of Focus



# Goal

- To make the Package C-states usable for latency critical applications, we propose AgilePkgC.

- AgilePkgC achieves it's objectives by introducing a new Package C-state called PC1A.
  - Requires cores to enter Core C-state C1
  - It has sub-microsecond transition time
  - Retains most power savings of deep idle state

# AgilePkgC Architecture

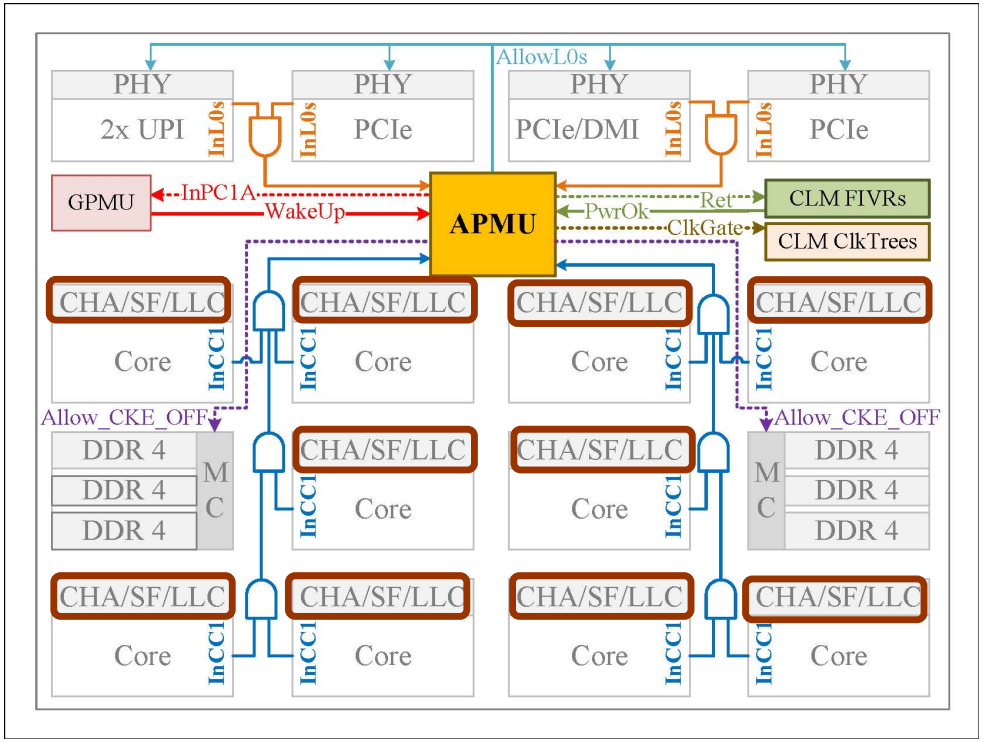
We achieve AgilePkgC goals with three key components:

1. CHA, LLC and Mesh Retention (CLMR)
2. IO Standby Mode (IOSM)
3. Agile Power Management Unit (APMU)

# 1. CHA, LLC and Mesh Retention (I)

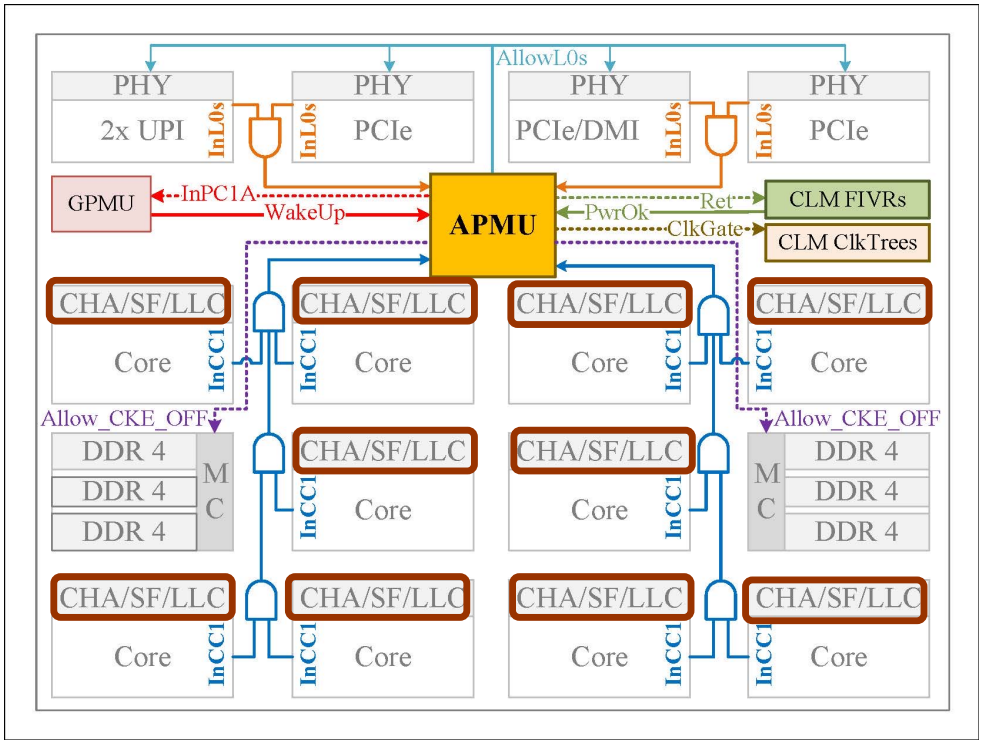
- Skylake Package Architecture:
  - Each skylake tile includes a portion of the LLC, a CHA and a SF
  - Two voltage domains (FIVR) power the LLC, CHA, SF and mesh interconnect

- The GPMU when initiates entry to PC6:
  1. It turns off the Phased Locked Loop (PLL)
  2. It reduces the voltage of the CLM to retention level



# 1. CHA, LLC and Mesh Retention (II)

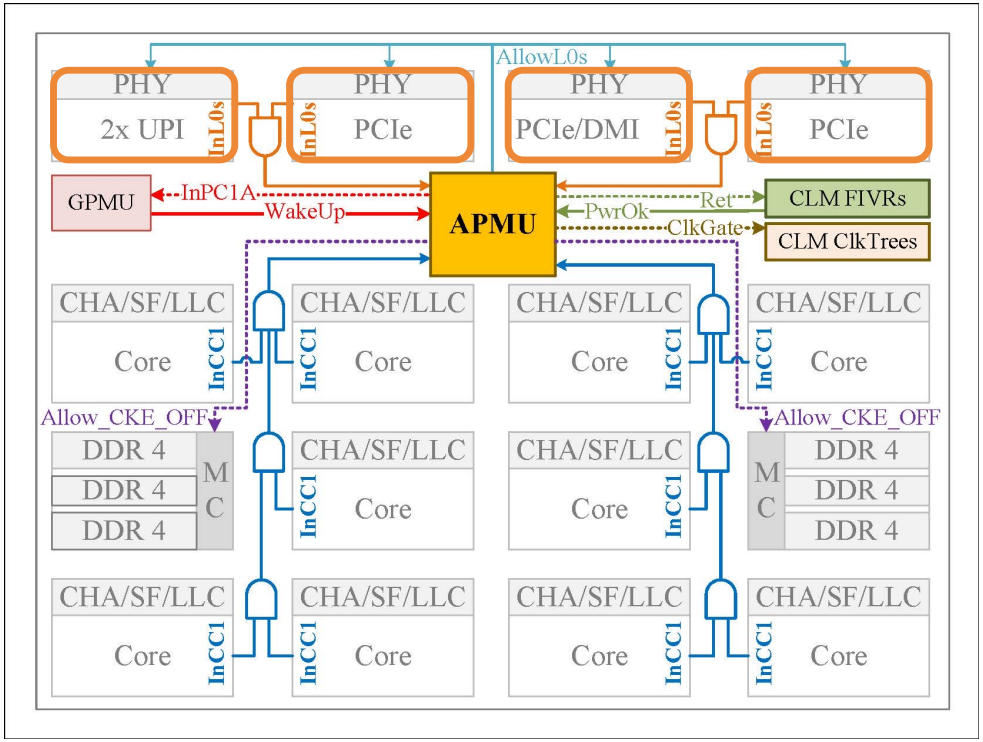
- AgilePkgC keeps the PLL locked to cut the time of re-locking (few microseconds)
- To allow agile communication between the APMU, the Clock Tree and the CLM FIVRs:
  - Add ClkGate signal to allow for fast clock-gating
  - Add Ret and PwrOk signal to allow for agile response of the CLM FIVR



# 2. IO Standby Mode (IOSM) (I)

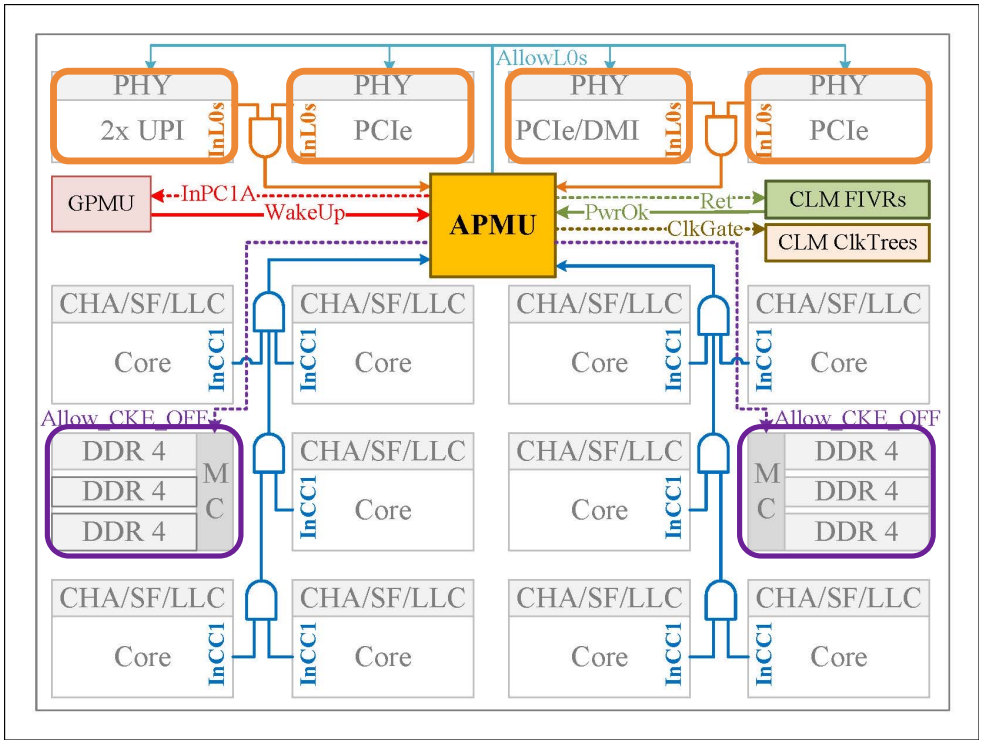
- IOSM leverages shallow power states to enable significant power savings in PC1A (IO, DRAM) with nanosecond-exit latency

- Skylake Package Architecture:
  - The North-Cap (top portion of SoC) contains all the high-speed IO controllers (PCIe, UPI and DMI)
  - Two core tiles on the left and right column are replaced with memory controllers (MC)



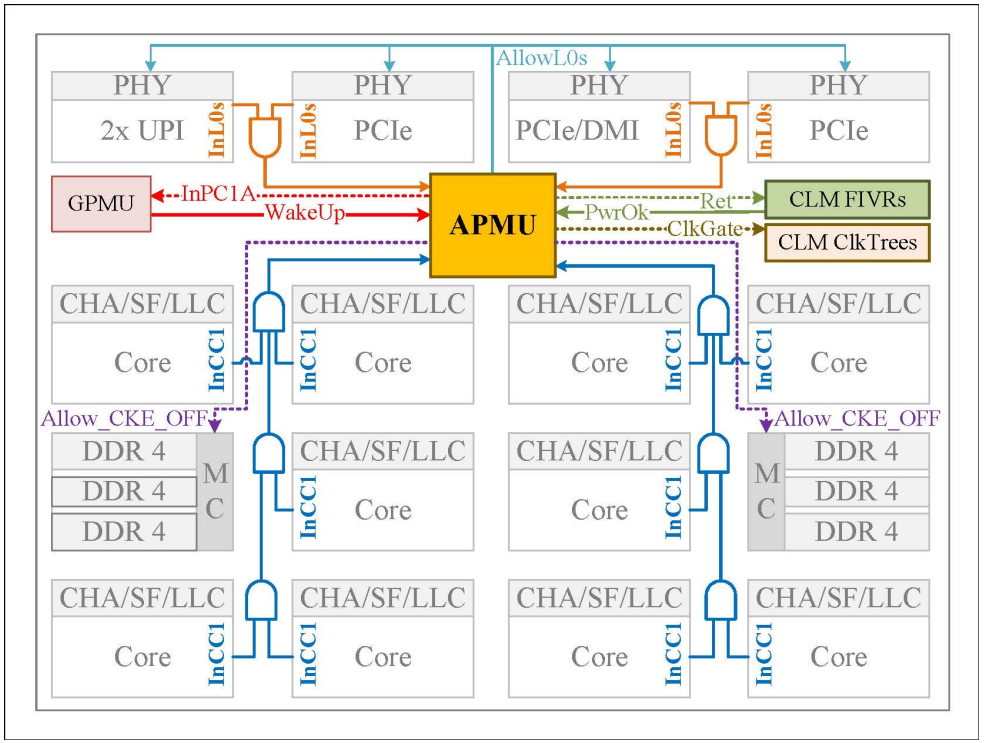
# 2. IO Standby Mode (IOSM) (II)

- AgilePkgC puts PCIe, DMI, UPI and DRAM into shallow power states
- PCIe, DMI and UPI are placed into L0s/L0p once all cores are in CC1
  - Add AllowL0s signal to notify IO controllers once all cores in CC1
  - Add InL0s signal to notify the power management firmware
- DRAM is placed into a CKE off mode
  - Add Allow_CKE_OFF signal to instruct MC to put DRAM into CKE off mode



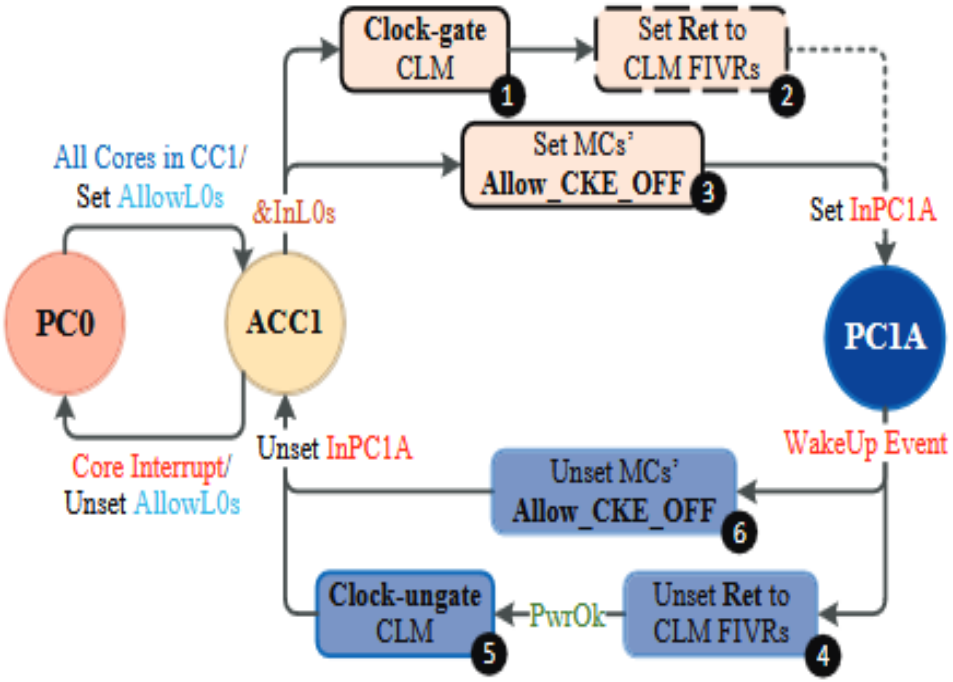
# 3. Agile Power Management Unit (APMU) (I)

- APMU is responsible to orchestrate the transition from PC0 to PC1A
- APMU implements three key components:
  - Finite-state machine
  - Status and event signals (i.e., InCC1)
  - Control Signals (i.e., Allow_CKE_OFF)



# 3. Agile Power Management Unit (APMU)(II)

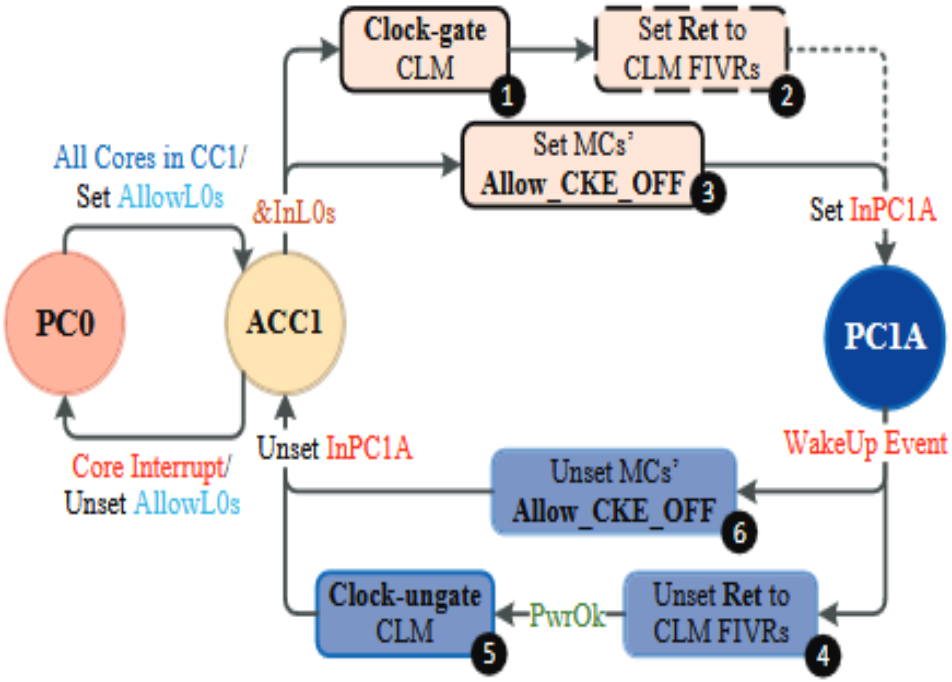
- Entry/Exit Flow:
  - Transition from PC0 to ACC1 once all cores in CC1
  - Set AllowL0s signal
  - Once all links in L0s, concurrently
    - 1) Clock-gate CLM and 2) Initiate a CLM voltage (retention) transition
    - Set Allow_CKE_OFF to allow MC to enter CKE mode



# 3. Agile Power Management Unit (APMU)(III)

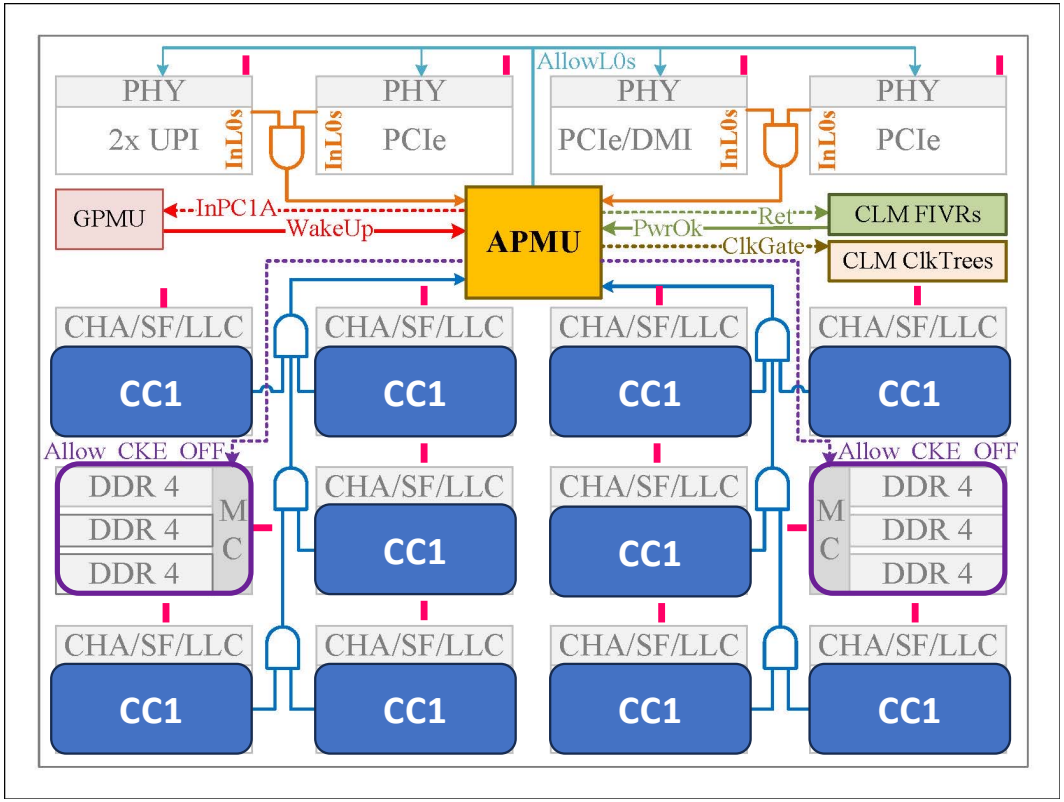
- Sources of wake-up events:
  - IO traffic arrival
  - GPMU (interrupt, timer expiration, thermal event).

- Exit flow is the reverse of the entry with some minor differences depending on the wakeup source.



# PC1A (Deep & Agile) Package C-state

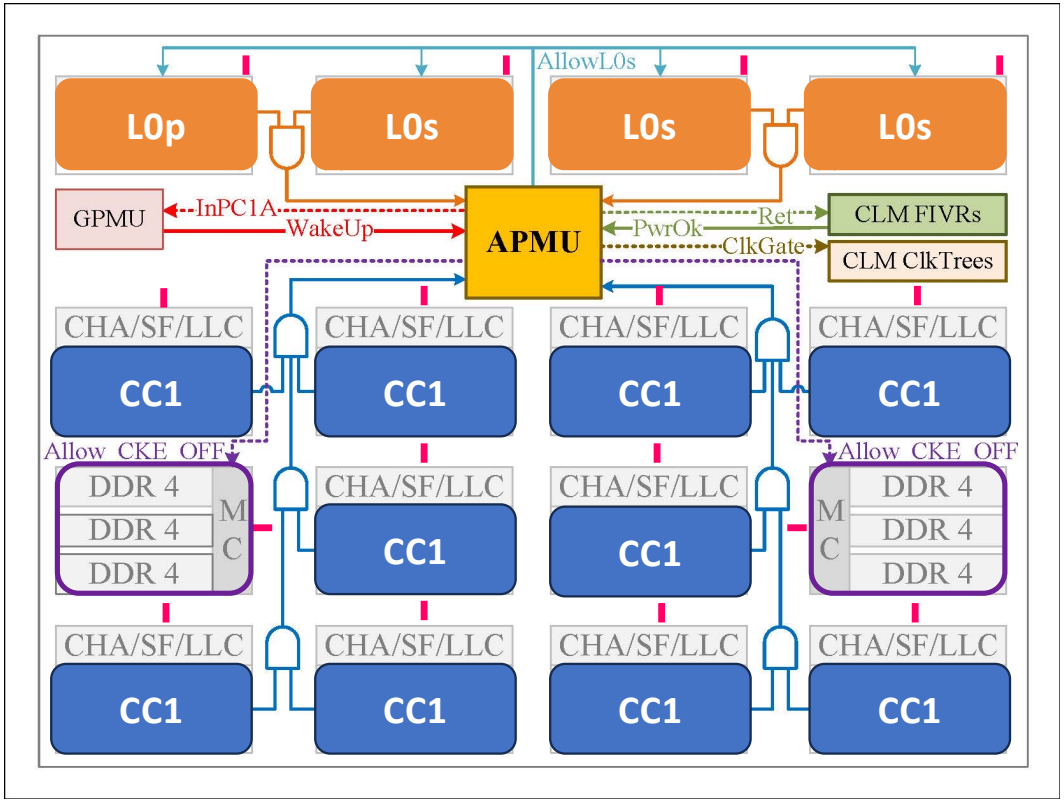
**All cores in CC1**




| PC-state | CC-state | Clock | PLL | Voltage | LLC | IO | DRAM |
|---|---|---|---|---|---|---|---|
| PC1A | C1 | On | On | Nom | Coherent | On | On |

# PC1A (Deep & Agile) Package C-state

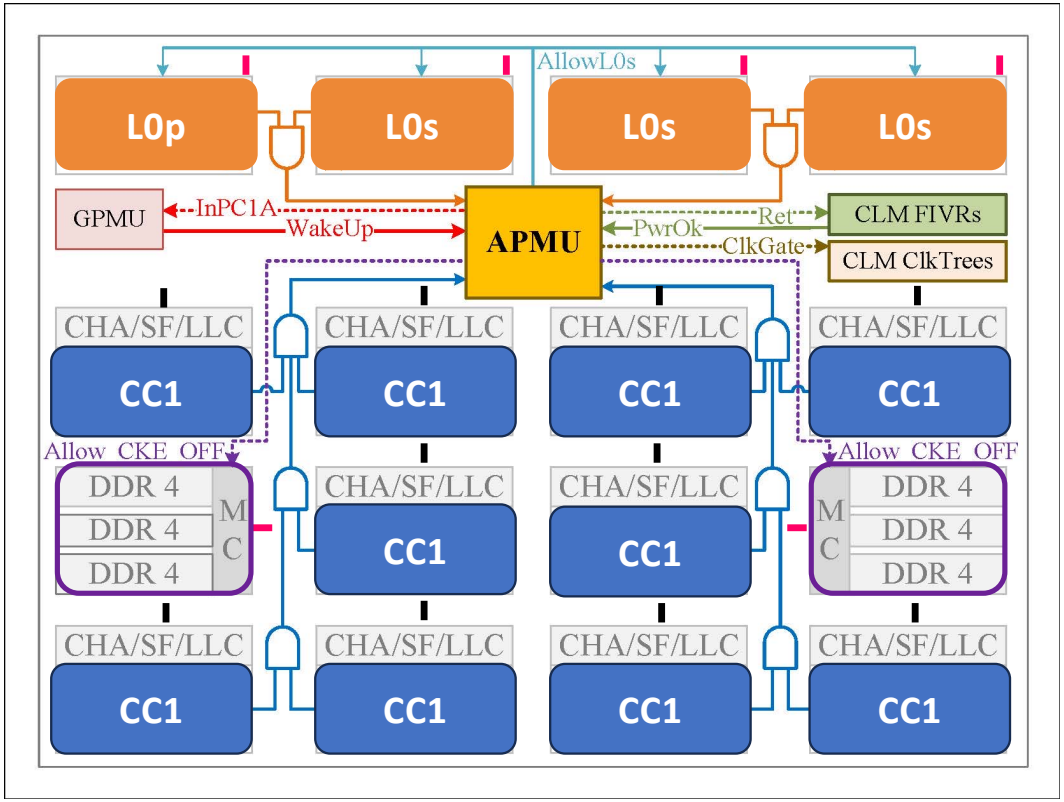
**Set AllowL0s**
**Enter L0s/L0p States**
**Set InL0s**




| PC-state | CC-state | Clock | PLL | Voltage | LLC | IO | DRAM |
|---|---|---|---|---|---|---|---|
| PC1A | C1 | On | On | Nom | Coherent | L0p/L0s | On |

# PC1A (Deep & Agile) Package C-state

**Clock-gate CLM**

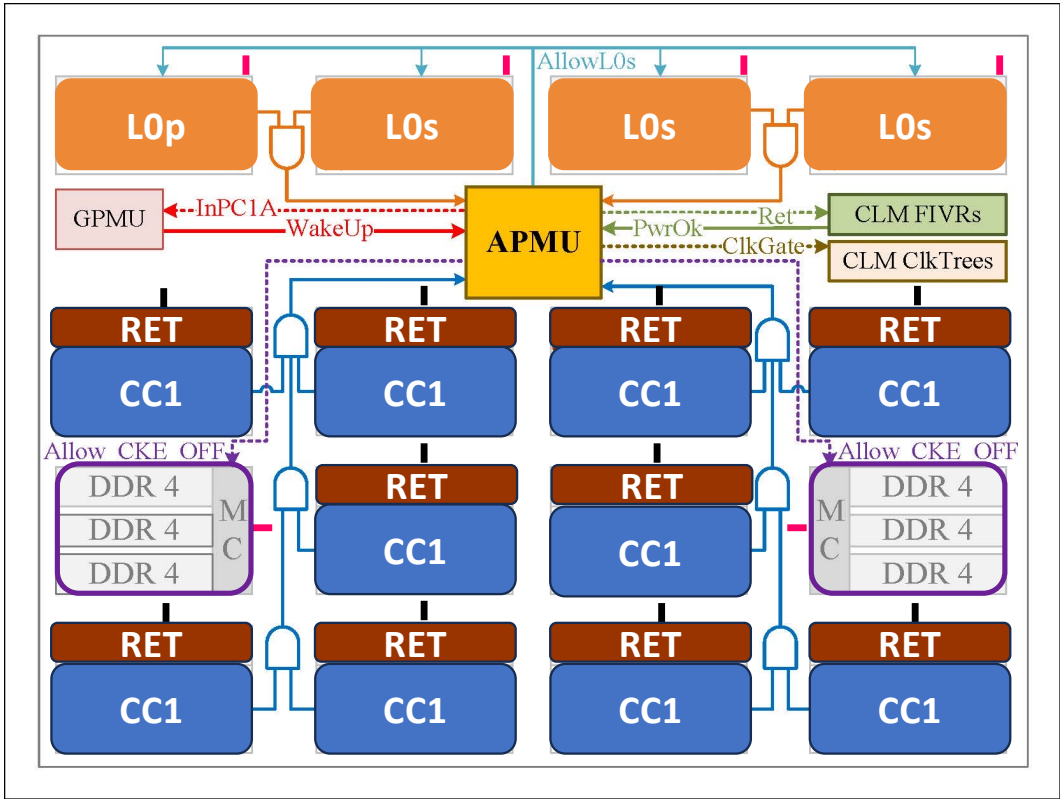



| PC-state | CC-state | Clock | PLL | Voltage | LLC | IO | DRAM |
|---|---|---|---|---|---|---|---|
| PC1A | C1 | Stopped | On | Nom | Coherent | L0p/L0s | On |

# PC1A (Deep & Agile) Package C-state

**Clock-gate CLM**

**Reduce CLM voltage to retention**



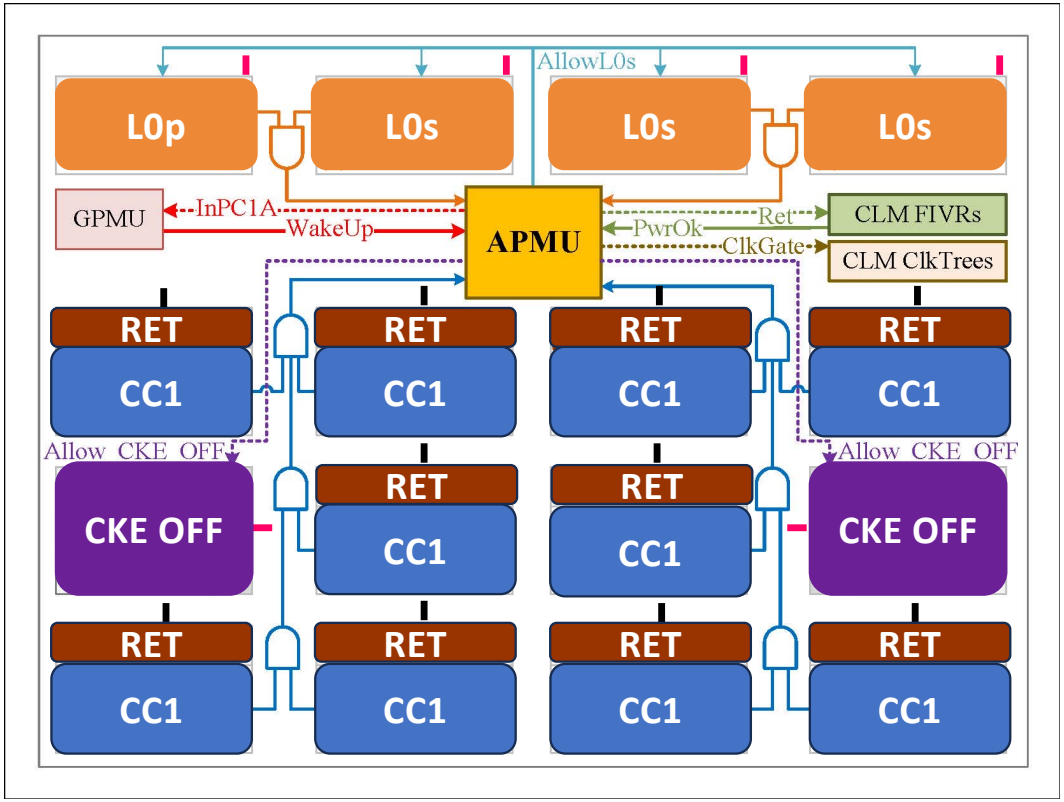

| PC-state | CC-state | Clock | PLL | Voltage | LLC | IO | DRAM |
|---|---|---|---|---|---|---|---|
| PC1A | C1 | Stopped | On | RET | Coherent | L0p/L0s | On |

# PC1A (Deep & Agile) Package C-state

**Clock-gate CLM**

**Reduce CLM voltage to retention**

**DRAM enters CKE OFF mode**




| PC-state | CC-state | Clock | PLL | Voltage | LLC | IO | DRAM |
|---|---|---|---|---|---|---|---|
| PC1A | C1 | Stopped | On | RET | Coherent | L0p/L0s | CKE OFF |

# Experimental Methodology

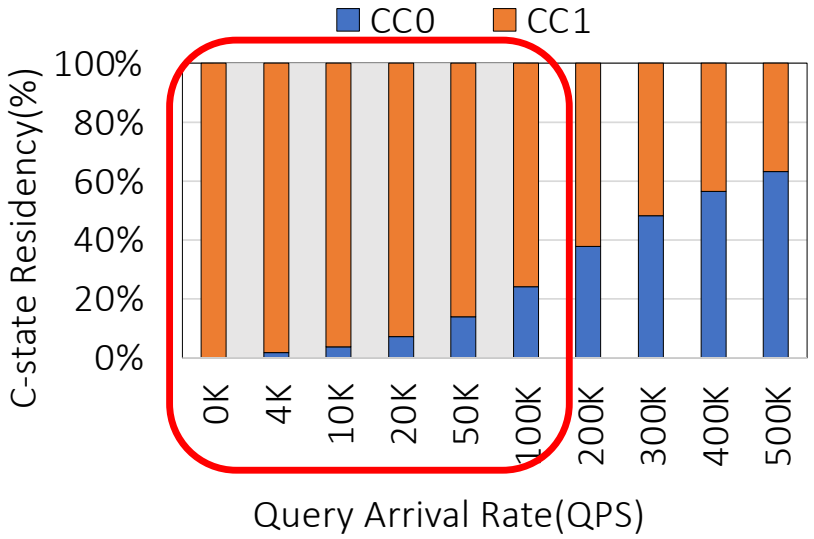
- **Workloads:** We evaluate AgilePkgC using three latency-critical applications: Memcached, Apache Kafka, and MySQL

- **Baseline Configuration:** We consider two baseline configurations $C_{shallow}$ representative of a real modern datacenter that configures servers for maximum performance and $C_{deep}$ where all core and package C-states are enabled

- **Power and Performance Estimations:** to evaluate AgilePkgC, we use a combination of models and real measurements. For the performance estimations we use the PC1A transitions measured in our baseline scenario and the additional transition latency required for PC1A

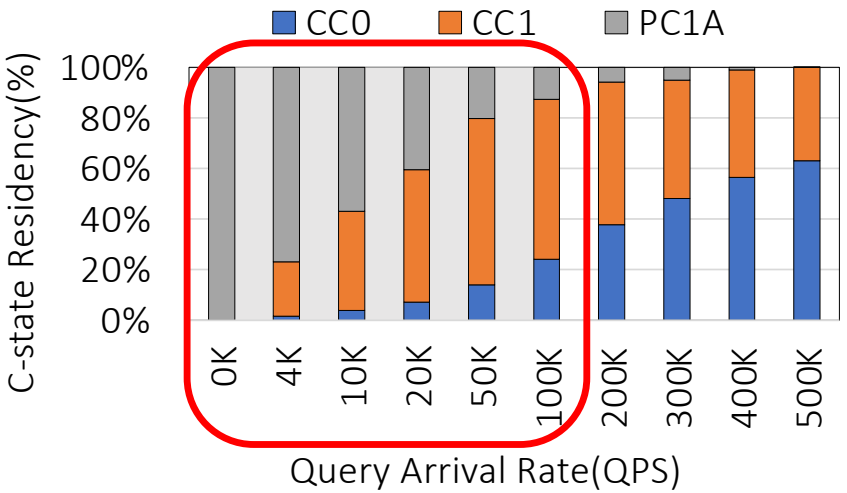
- **Power Event Tracing:** We use SoCwatch to measure the PC1A residency

# PC1A Opportunity

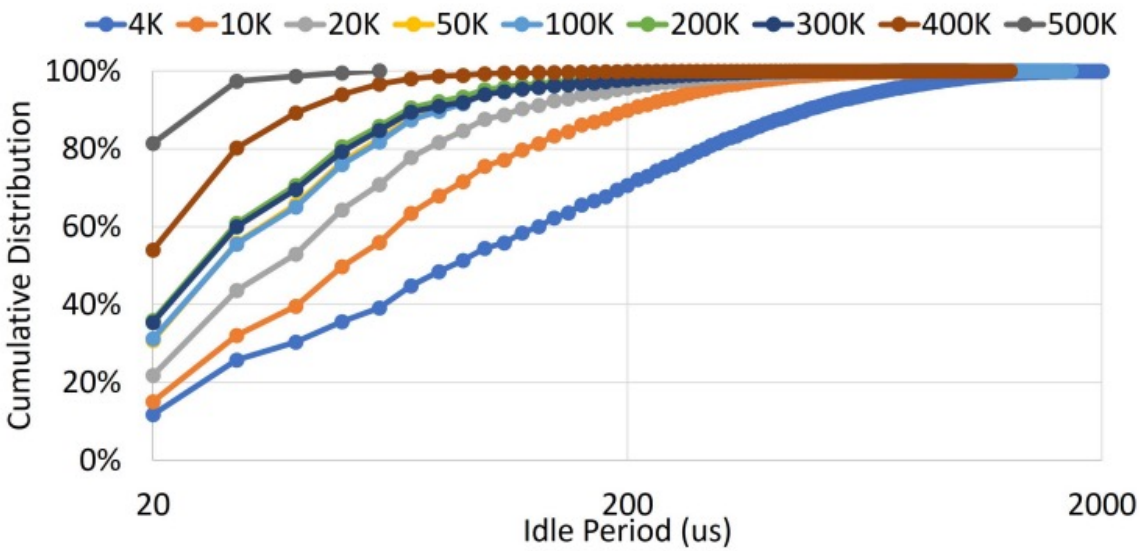



- Average fraction of time each core is in active state (CC0) and shallow core sleep state (CC1)
- For **low load (<= 100 KQPS),** for a large fraction of time **(76% to 98%)** a core is **in** shallow core sleep state (**CC1**)

- **Entering PC1A** requires **all** core to be in shallow core sleep state **C1** (CC1).
- Average fraction of time each core is in active (CC0) shallow core sleep state (CC1) and PC1A respectively
- Although, **PC1A** residency diminishes at high load, the **opportunity** is **significant** at **low load**(>=12 %)

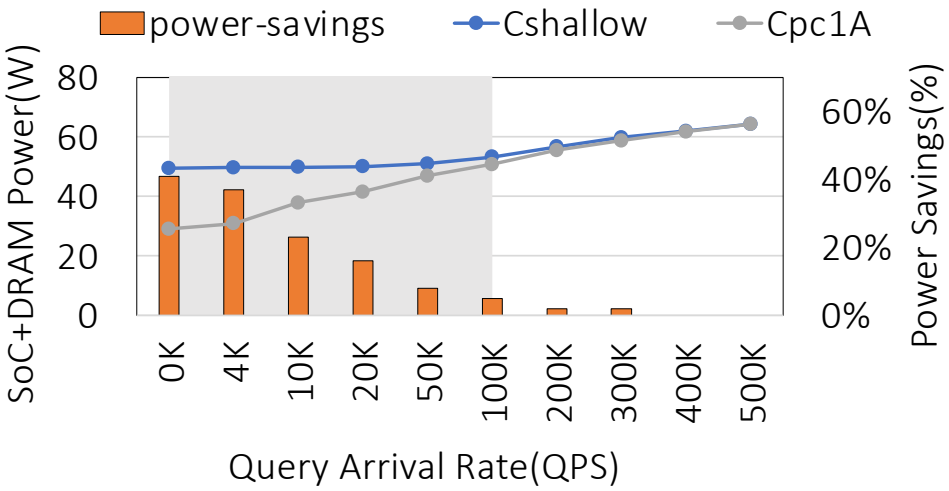


# PC1A Opportunity



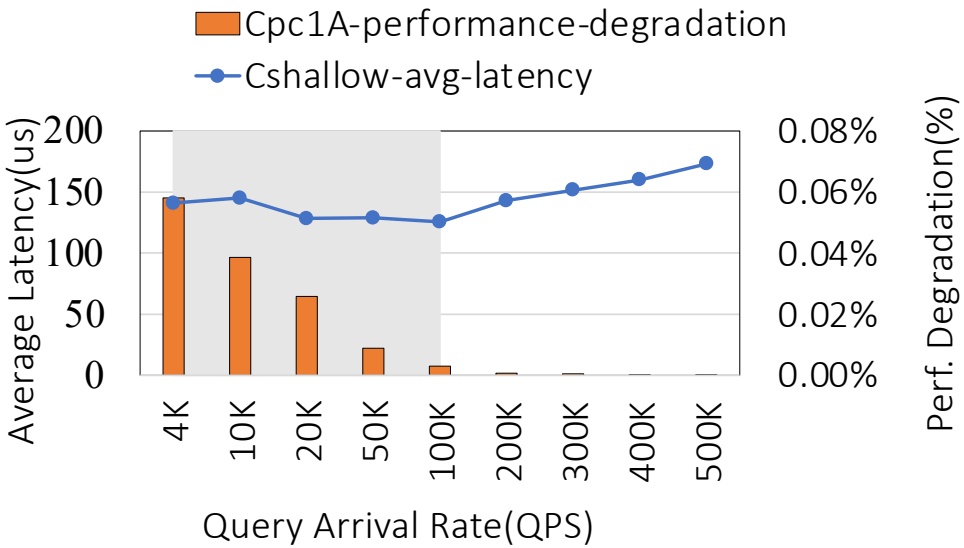

- At low load, 60% of the idle periods have duration between 20us – 200us.
- The PC1A transition latency is <= 200ns.
- The fast transition latency enables to reap most of the power reduction opportunity during short periods, which is infeasible with existing PC6 states(~50us transition latency).

# Evaluation – Power and Performance




- End – to – end latency includes server-side latency plus network latency (~117us).
- Even in the worst case, **PC1A** has a **negligible**(<0.1%) impact on average latency.
- Non-changing **network** latency **dominates** the **overall response time**.

- **PC1A** has lower (or lower) **power consumption** than the baseline system.
- **Power savings more pronounced at low load**, as opportunity to enter PC1A is higher.
- At 4K QPS PC1A has **37%** lower power, while at 50K QPS, it has **8%.**



# Other Details in the Paper

- Implementation and hardware cost
- PC1A latency
- Design alternatives for CLM Retention
- IO activity
- Analysis of additional workloads
- Datacenter cost saving analysis
- End-to-end latency and power consumption sensitivity study

# Summary

- AgilePkgC is a new deep Package C-state that improves the energy proportionality for servers running latency critical applications
- AgilePkgC leverages shallower PCIe, UPI, DMI and DRAM power states
- AgilePkgC reduces the transition latency by 250x and the energy consumption by up to 41% with minimal performance degradation
- AgilePkgC is an effective approach to improving energy consumption of servers running latency-critical applications by enabling the server's package to enter deep energy-saving states during short idle periods (all core idles) with negligible performance impact