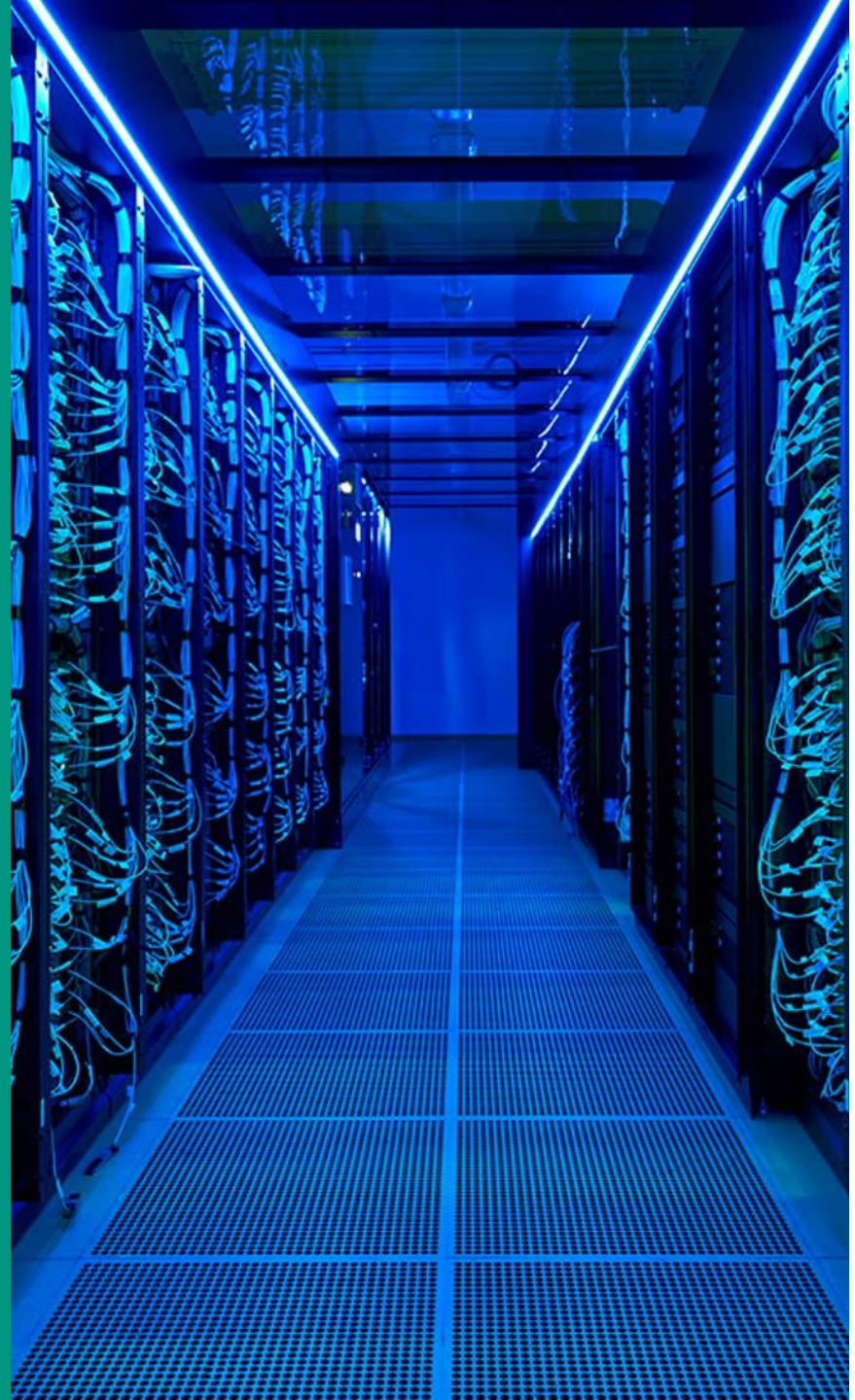
An Introduction to CPU Power Management and System Suspend



Yussuf Khalil
18 May 2026

Electrical Fundamentals

$$P = P_{switching} + P_{short-circuit} + P_{leakage}$$

Electrical Fundamentals

$$P = P_{switching} + P_{short-circuit} + P_{leakage}$$

Charging/discharging transistor capacities

Electrical Fundamentals

$$P = P_{switching} + P_{short-circuit} + P_{leakage}$$

Charging/discharging transistor capacities

Direct connection V ↔ GND during switching

Electrical Fundamentals

$$P = P_{switching} + P_{short-circuit} + P_{leakage}$$

- $P_{switching}$: Charging/discharging transistor capacities
- $P_{short-circuit}$: Direct connection V ↔ GND during switching
- $P_{leakage}$: Ideal world: $R = \infty$ Reality: $R < \infty$

Electrical Fundamentals

$$P = P_{switching} + P_{short-circuit} + P_{leakage}$$

- $P_{switching}$: Charging/discharging transistor capacities
- $P_{short-circuit}$: Direct connection V ↔ GND during switching
- $P_{leakage}$: Ideal world: $R = \infty$ Reality: $R < \infty$

$$P_{switching} = \frac{1}{2} C V^2 f$$

Electrical Fundamentals

$$P = P_{switching} + P_{short-circuit} + P_{leakage}$$

- $P_{switching}$: Charging/discharging transistor capacities
- $P_{short-circuit}$: Direct connection V ↔ GND during switching
- $P_{leakage}$: Ideal world: $R = \infty$ Reality: $R < \infty$

$$P_{switching} = \frac{1}{2} C V^2 f$$

- V^2: Voltage is squared!

Electrical Fundamentals

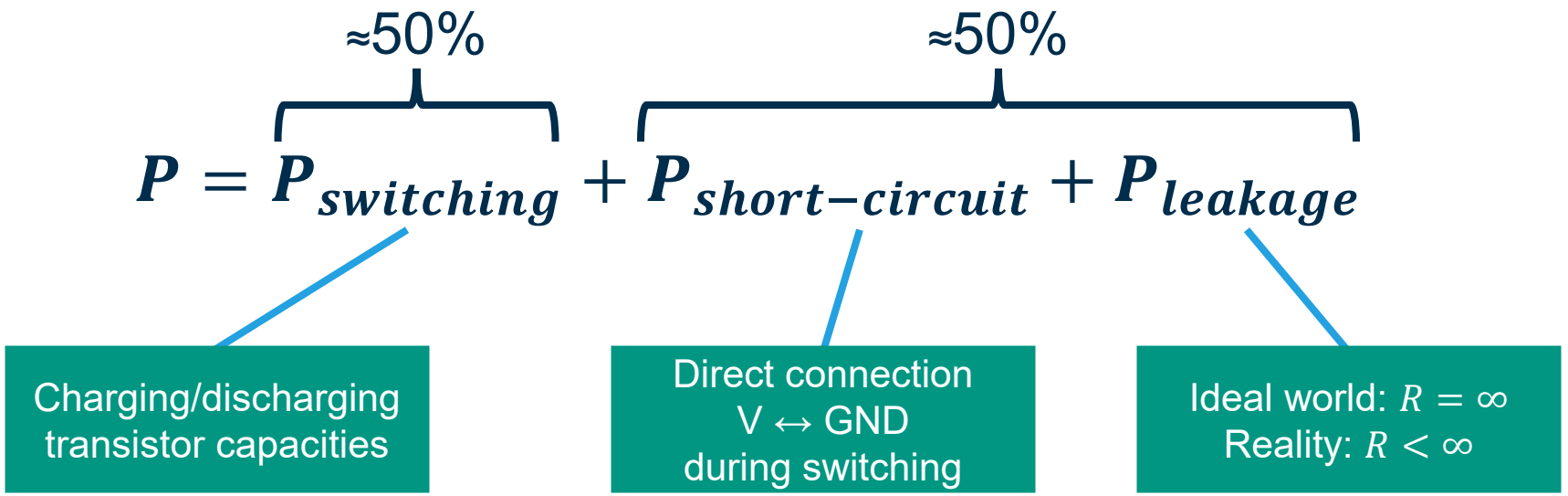
$$P = P_{switching} + P_{short-circuit} + P_{leakage}$$

- $P_{switching}$: Charging/discharging transistor capacities
- $P_{short-circuit}$: Direct connection V ↔ GND during switching
- $P_{leakage}$: Ideal world: $R = \infty$ Reality: $R < \infty$

$$P_{switching} = \frac{1}{2} C V^2 f$$

- V^2: Voltage is squared!
- f: Clock frequency is a function of the voltage!

Electrical Fundamentals



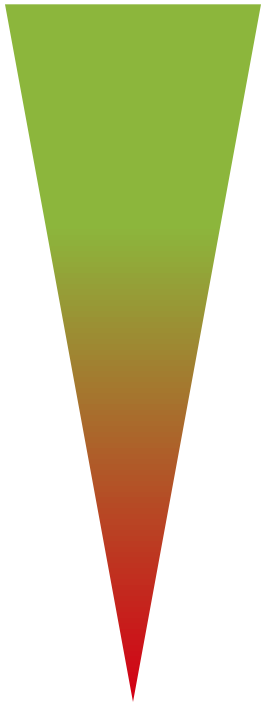
$$P_{switching} = \frac{1}{2} C V^2 f$$

Voltage is squared!

Clock frequency is a function of the voltage!

How To Save Energy

Energy savings

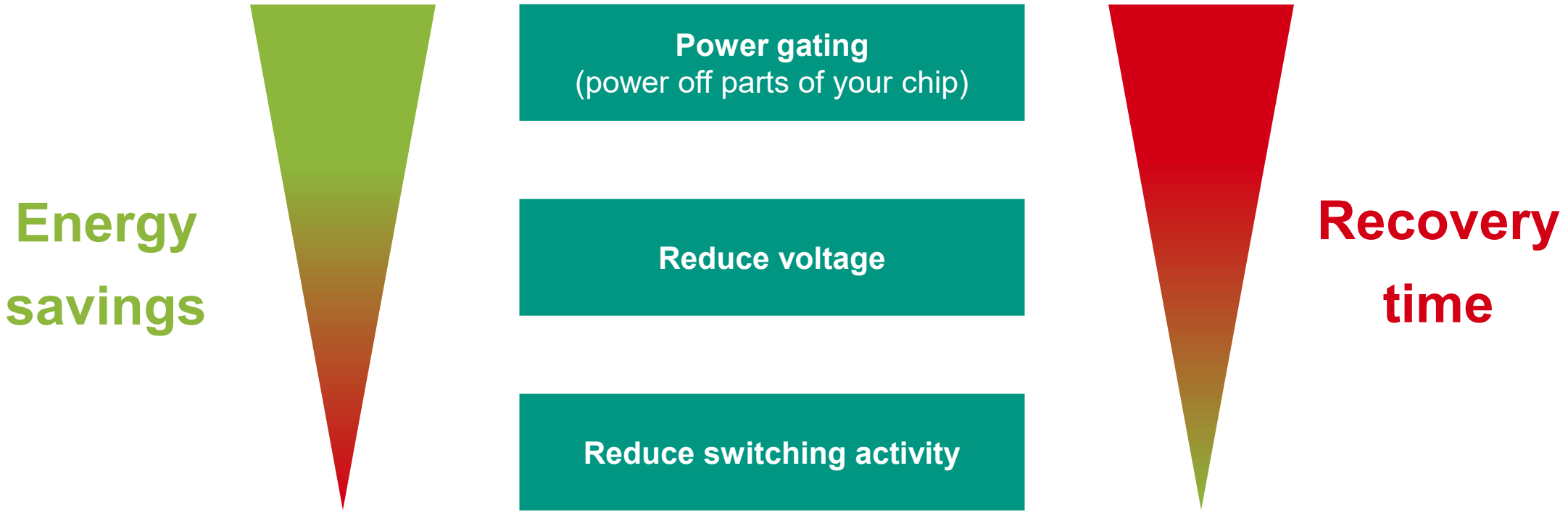


Power gating
(power off parts of your chip)

Reduce voltage

Reduce switching activity

How To Save Energy



Advanced Configuration and Power Interface (ACPI)

- Governing standard for system and device power management features
  - However, many vendor-specific extensions/deviations

Advanced Configuration and Power Interface (ACPI) Specification

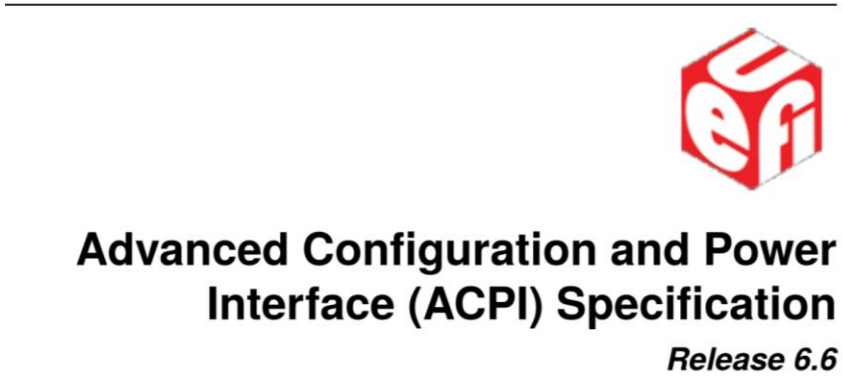
Release 6.6

UEFI Forum, Inc.

May 13, 2025

Advanced Configuration and Power Interface (ACPI)

- Governing standard for system and device power management features
  - However, many vendor-specific extensions/deviations
- *ACPI tables* communicate information from firmware (BIOS) to OS



Advanced Configuration and Power Interface (ACPI)

- Governing standard for system and device power management features
  - However, many vendor-specific extensions/deviations
- *ACPI tables* communicate information from firmware (BIOS) to OS
- *ACPI Machine Language (AML)* allows firmware to specify code that the OS executes on behalf of the firmware at runtime

Advanced Configuration and Power Interface (ACPI) Specification

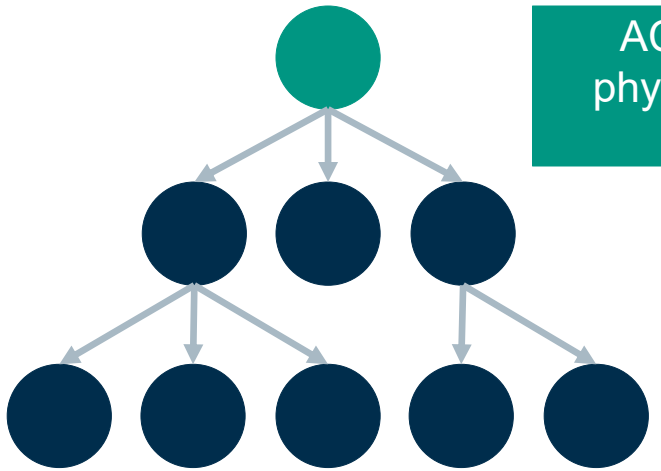
Release 6.6

UEFI Forum, Inc.

May 13, 2025

Advanced Configuration and Power Interface (ACPI)

- Governing standard for system and device power management features
  - However, many vendor-specific extensions/deviations
- *ACPI tables* communicate information from firmware (BIOS) to OS
- *ACPI Machine Language (AML)* allows firmware to specify code that the OS executes on behalf of the firmware at runtime



ACPI represents virtual and physical devices as nodes in a tree

Advanced Configuration and Power Interface (ACPI) Specification

Release 6.6

UEFI Forum, Inc.

May 13, 2025

Advanced Configuration and Power Interface (ACPI)

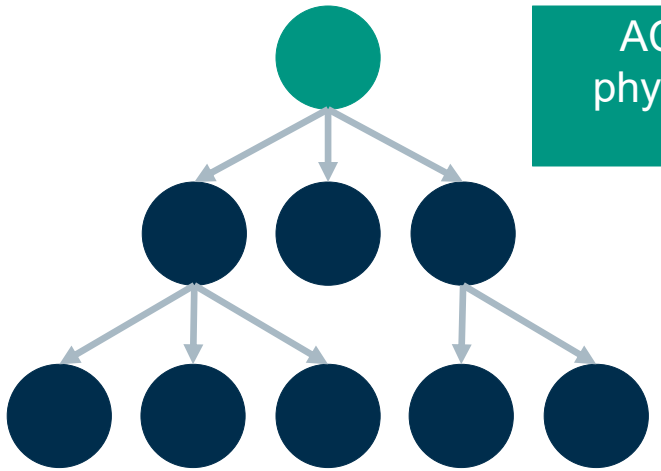
- Governing standard for system and device power management features
  - However, many vendor-specific extensions/deviations
- *ACPI tables* communicate information from firmware (BIOS) to OS
- *ACPI Machine Language (AML)* allows firmware to specify code that the OS executes on behalf of the firmware at runtime

Advanced Configuration and Power Interface (ACPI) Specification

Release 6.6

UEFI Forum, Inc.

May 13, 2025



ACPI represents virtual and physical devices as nodes in a tree

Theory
Independent nodes can suspend/resume in parallel

Advanced Configuration and Power Interface (ACPI)

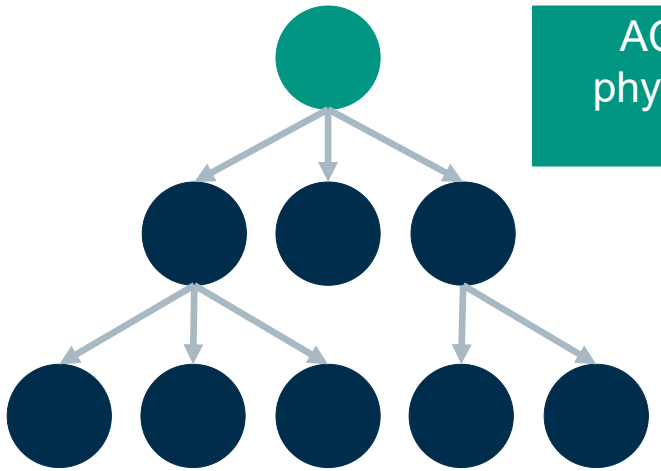
- Governing standard for system and device power management features
  - However, many vendor-specific extensions/deviations
- *ACPI tables* communicate information from firmware (BIOS) to OS
- *ACPI Machine Language (AML)* allows firmware to specify code that the OS executes on behalf of the firmware at runtime

Advanced Configuration and Power Interface (ACPI) Specification

Release 6.6

UEFI Forum, Inc.

May 13, 2025



ACPI represents virtual and physical devices as nodes in a tree

Theory
Independent nodes can suspend/resume in parallel

Reality Check
Linux serializes all operations

ACPI CPU Power Management States

ACPI CPU Power Management States

P-States
How fast is my CPU running?

ACPI CPU Power Management States

P-States
How fast is my CPU running?

C-States
Which parts of my CPU core are powered?

ACPI CPU Power Management States

P-States
How fast is my CPU running?

C-States
Which parts of my CPU core are powered?

PC-States
Which parts of my CPU package are powered?

Disclaimer:
The next slides will mostly use Intel terminology.
Things work generally similar on AMD or ARM.

P-States

P-states = Discrete clock frequency steps

P-States

P-states = Discrete clock frequency steps

- Traditionally: P-states managed by the OS (Intel SpeedStep)

P-States

P-states = Discrete clock frequency steps

- Traditionally: P-states managed by the OS (Intel SpeedStep)
- Today: P-states managed by the CPU itself (Intel SpeedShift / HWP / ACPI CPPC)
  - OS only gives hints and constraints
  - Quicker reaction to thermal/electrical events
  - P-state decision every ~500 µs (Intel Skylake)

P-States

P-states = Discrete clock frequency steps

- Traditionally: P-states managed by the OS (Intel SpeedStep)
- Today: P-states managed by the CPU itself (Intel SpeedShift / HWP / ACPI CPPC)
  - OS only gives hints and constraints
  - Quicker reaction to thermal/electrical events
  - P-state decision every ~500 µs (Intel Skylake)

It's really about the *voltage*, not so much the frequency

P-State Transitions

Decreasing frequency

P-State Transitions

Decreasing frequency

Increasing frequency

P-State Transitions

Decreasing frequency

Increasing frequency

Increasing the frequency takes much longer than decreasing!

C-States

C-states disable portions of a CPU core

C0	Core in full operation
C1	Clock signal gated (= no switching activity)
C1E	Lowest P-state
C2 … C9	Power off caches, TLBs, other portions of core
C10	Core fully powered off

C-States

C-states disable portions of a CPU core

C0	Core in full operation
C1	Clock signal gated (= no switching activity)
C1E	Lowest P-state
C2 … C9	Power off caches, TLBs, other portions of core
C10	Core fully powered off

Precise C-state definitions depend on CPU model

C-States

C-states disable portions of a CPU core

C0	Core in full operation
C1	Clock signal gated (= no switching activity)
C1E	Lowest P-state
C2 … C9	Power off caches, TLBs, other portions of core
C10	Core fully powered off

Precise C-state definitions depend on CPU model

- C-states triggered by OS via `MWAIT` instruction
  - `MWAIT` tells CPU core to halt execution until
    - a specified memory address is written to, or
    - a timeout was reached
  - OS defines target C-state for `MWAIT`

C-States

C-states disable portions of a CPU core

C0	Core in full operation
C1	Clock signal gated (= no switching activity)
C1E	Lowest P-state
C2 … C9	Power off caches, TLBs, other portions of core
C10	Core fully powered off

Precise C-state definitions depend on CPU model

- C-states triggered by OS via `MWAIT` instruction
  - `MWAIT` tells CPU core to halt execution until
    - a specified memory address is written to, or
    - a timeout was reached
  - OS defines target C-state for `MWAIT`
- Recent CPUs: `UMWAIT` as a userspace instruction for short waiting periods

C-State Transitions

Transitioning back to C0 comes at a cost

C-State Transitions

Transitioning back to C0 comes at a cost

Energy consumed while powering up core again before ready for execution

C-State Transitions

Transitioning back to C0 comes at a cost

Energy consumed while powering up core again before ready for execution

Caches cold, may need to fetch data again from memory

C-State Transitions

Transitioning back to C0 comes at a cost

Energy consumed while powering up core again before ready for execution

Caches cold, may need to fetch data again from memory

Too aggressive C-state use can increase energy consumption

PC-States

PC-states disable portions of the entire CPU package

PC0	CPU in full operation
PC2	All cores in low C-states
PC3	All caches powered off, memory clock gated
PC4 … PC9	Disable voltage regulators, interconnects, etc.
PC10	CPU nearly entirely powered off (only architectural state kept)

PC-States

PC-states disable portions of the entire CPU package

PC0	CPU in full operation
PC2	All cores in low C-states
PC3	All caches powered off, memory clock gated
PC4 … PC9	Disable voltage regulators, interconnects, etc.
PC10	CPU nearly entirely powered off (only architectural state kept)

Precise PC-state definitions depend on CPU model

PC-States

PC-states disable portions of the entire CPU package

PC0	CPU in full operation
PC2	All cores in low C-states
PC3	All caches powered off, memory clock gated
PC4 … PC9	Disable voltage regulators, interconnects, etc.
PC10	CPU nearly entirely powered off (only architectural state kept)

Precise PC-state definitions depend on CPU model

- Again, PC-states requested by OS
- Deepest PC-states tied to suspend modes (more on that later)

PC-States

PC-states disable portions of the entire CPU package

PC0	CPU in full operation
PC2	All cores in low C-states
PC3	All caches powered off, memory clock gated
PC4 … PC9	Disable voltage regulators, interconnects, etc.
PC10	CPU nearly entirely powered off (only architectural state kept)

Precise PC-state definitions depend on CPU model

- Again, PC-states requested by OS
- Deepest PC-states tied to suspend modes (more on that later)

CPUs with integrated GPUs have RC-states that influence PC-states

Reality Check

My laptop’s CPU automatically powers down during idle periods

Reality Check

My laptop's CPU automatically powers down during idle periods

Server CPUs only support shallow C- and PC-states

Reality Check

My laptop's CPU automatically powers down during idle periods

Server CPUs only support shallow C- and PC-states

Cloud operators care deeply about (tail) latencies

Deep C-states drastically increase tail latencies

Reality Check

My laptop's CPU automatically powers down during idle periods

Server CPUs only support shallow C- and PC-states

Cloud operators care deeply about (tail) latencies

Deep C-states drastically increase tail latencies

Hyperscalers disable C-states entirely or limit to C1

Reality Check

My laptop's CPU automatically powers down during idle periods

Server CPUs only support shallow C- and PC-states

Cloud operators care deeply about (tail) latencies

Deep C-states drastically increase tail latencies

Hyperscalers disable C-states entirely or limit to C1

No incentive for chip vendors to support deeper C-states

Race-to-Halt

$$E = \int_t P$$

Race-to-Halt

$$E = \int_t P$$

Lower P-states not always beneficial
Lower power but longer time to complete task
⇒ May consume *more* total energy
⇒ Less time spent in deep C-states

Race-to-Halt

$$E = \int_t P$$

Lower P-states not always beneficial
Lower power but longer time to complete task
⇒ May consume *more* total energy
⇒ Less time spent in deep C-states

DRAM power management interplays with P-states
Too much time between memory requests
⇒ Delayed DRAM gear-down

Race-to-Halt

$$E = \int_t P$$

Lower P-states not always beneficial
Lower power but longer time to complete task
⇒ May consume *more* total energy
⇒ Less time spent in deep C-states

DRAM power management interplays with P-states
Too much time between memory requests
⇒ Delayed DRAM gear-down

Race-to-halt
Run faster to be done sooner to save energy

Race-to-Halt

$$E = \int_t P$$

Lower P-states not always beneficial
Lower power but longer time to complete task
⇒ May consume *more* total energy
⇒ Less time spent in deep C-states

DRAM power management interplays with P-states
Too much time between memory requests
⇒ Delayed DRAM gear-down

Race-to-halt
Run faster to be done sooner to save energy

Does not help with heavily memory-bound tasks due to increased pipeline stalls

Dennard Scaling

Moore's Law
Integration density doubles every two years

Dennard Scaling

Moore's Law
Integration density doubles every two years

is enabled by

Dennard's Law
Energy density (J/cm^2) stays constant with scaling

Dennard Scaling

Moore's Law
Integration density doubles every two years

is enabled by

Dennard's Law
Energy density (J/cm^2) stays constant with scaling

Reality Check
Dennard scaling does not hold with modern transistor scales

Dennard Scaling

Moore's Law
Integration density doubles every two years

is enabled by

Dennard's Law
Energy density (J/cm^2) stays constant with scaling

Reality Check
Dennard scaling does not hold with modern transistor scales

What if…

Energy density > thermal conductance of heatsink?

Dennard Scaling

Moore's Law
Integration density doubles every two years

is enabled by

Dennard's Law
Energy density (J/cm^2) stays constant with scaling

Reality Check
Dennard scaling does not hold with modern transistor scales

What if…

Energy density > thermal conductance of heatsink?

Thermal runaway!

Turbo Boost / RAPL

Turbo Boost
Operate some CPU cores at higher frequency than highest P-state

Turbo Boost / RAPL

Turbo Boost
Operate some CPU cores at higher frequency than highest P-state

governed by

Running Average Power Limit (RAPL)
- Measure energy over sliding time window
- Apply configured power limits
  - Baseline limit for long-term consumption
  - ≥1 power limits for short-term spikes

Turbo Boost / RAPL

Turbo Boost
Operate some CPU cores at higher frequency than highest P-state

governed by

Running Average Power Limit (RAPL)
- Measure energy over sliding time window
- Apply configured power limits
  - Baseline limit for long-term consumption
  - ≥1 power limits for short-term spikes

Dim silicon
Operate most of silicon below capability most of time to decrease energy density

Turbo Boost / RAPL

Turbo Boost
Operate some CPU cores at higher frequency than highest P-state

governed by

Running Average Power Limit (RAPL)
- Measure energy over sliding time window
- Apply configured power limits
  - Baseline limit for long-term consumption
  - ≥1 power limits for short-term spikes

Dim silicon
Operate most of silicon below capability most of time to decrease energy density

Race-to-halt strategy
Reduce total energy through short boosts

Ski Rental Problem

Ski Rental Problem

You want to go skiing

Ski Rental Problem

You want to go skiing

You do not own skis

Ski Rental Problem

You want to go skiing

You do not own skis

Purchasing skis costs €300,
renting skis costs €50.

Ski Rental Problem

You want to go skiing

You do not own skis

Purchasing skis costs €300,
renting skis costs €50.

Maybe you want to go skiing again in the future. Maybe not.

Ski Rental Problem

You want to go skiing

You do not own skis

Purchasing skis costs €300,
renting skis costs €50.

Maybe you want to go skiing again in the future. Maybe not.

Decide on each trip: **Do you buy or rent skis?**

Ski Rental Problem

You want to go skiing

You do not own skis

Purchasing skis costs €300,
renting skis costs €50.

Maybe you want to go skiing again in the future. Maybe not.

Decide on each trip: **Do you buy or rent skis?**

Solution
Buy skis on your sixth trip

Ski Rental Problem

You want to go skiing

You do not own skis

Purchasing skis costs €300,
renting skis costs €50.

Maybe you want to go skiing again in the future. Maybe not.

Decide on each trip: **Do you buy or rent skis?**

Solution
Buy skis on your sixth trip

$\leq$ **5 trips**	$\geq$ **6 trips**
Paid optimal cost	Paid €550 (€300 optimal)

Ski Rental Problem

You want to go skiing

You do not own skis

Purchasing skis costs €300,
renting skis costs €50.

Maybe you want to go skiing again in the future. Maybe not.

Decide on each trip: **Do you buy or rent skis?**

Solution
Buy skis on your sixth trip

$\leq$ 5 trips
Paid optimal cost

$\geq$ 6 trips
Paid €550 (€300 optimal)

Algorithm never more than 83.3% worse than optimal solution,
no better algorithm possible under given assumptions

Yes, Theoretical Computer Science Can be Useful

Yes, Theoretical Computer Science Can be Useful

Transitioning to lower power state incurs additional energy cost for resuming

Yes, Theoretical Computer Science Can be Useful

Transitioning to lower power state incurs additional energy cost for resuming

Calculate break-even time
energy cost of staying in high state
vs.
time needed in lower state to actually reduce energy

Yes, Theoretical Computer Science Can be Useful

Transitioning to lower power state incurs additional energy cost for resuming

Calculate break-even time
energy cost of staying in high state
vs.
time needed in lower state to actually reduce energy

Apply Ski Rental Problem
Switch to lower power state when
elapsed idle time = break-even time

Yes, Theoretical Computer Science Can be Useful

Transitioning to lower power state incurs additional energy cost for resuming

Calculate break-even time
energy cost of staying in high state
vs.
time needed in lower state to actually reduce energy

Apply Ski Rental Problem
Switch to lower power state when
elapsed idle time = break-even time

Can we do better in practice?

Predictive policies, learning policies, stochastic modeling, workload scheduling…

Practical Tips for Idle Consumption

```
$ sudo turbostat
[…]
Core    CPU     C1%     C1E%    C6%     CPU%c1  CPU%c6  Pkg%pc2 Pkg%pc6 PkgWatt
-       -       0.00    0.02    99.92   0.05    99.82   97.88   0.00    38.37
0       16      0.00    0.15    99.79   0.22
1       17      0.00    0.00    99.88   0.02
2       18      0.00    0.00    99.94   0.04
3       19      0.00    0.00    99.96   0.11
4       20      0.00    0.08    99.87   0.10
5       21      0.00    0.02    99.93   0.03
6       22      0.00    0.00    99.98   0.09
7       23      0.00    0.00    99.96   0.01
[…]
```

Practical Tips for Idle Consumption

```
$ sudo turbostat
[…]
Core    CPU     C1%     C1E%    C6%     CPU%c1  CPU%c6  Pkg%pc2 Pkg%pc6 PkgWatt
-       -       0.00    0.02    99.92   0.05    99.82   97.88   0.00    38.37
0       16      0.00    0.15    99.79   0.22
1       17      0.00    0.00    99.88   0.02
2       18      0.00    0.00    99.94   0.04
3       19      0.00    0.00    99.96   0.11
4       20      0.00    0.08    99.87   0.10
5       21      0.00    0.02    99.93   0.03
6       22      0.00    0.00    99.98   0.09
7       23      0.00    0.00    99.96   0.01
[…]
```

Verify C-state residency

Verify PC-state residency

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check firmware configuration

- C-states disabled/limited?
- PC-states disabled/limited?
- SpeedStep (EIST) disabled?
- HWP/SpeedShift/CPPC disabled?
- Firmware update available?

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check firmware configuration

- C-states disabled/limited?
- PC-states disabled/limited?
- SpeedStep (EIST) disabled?
- HWP/SpeedShift/CPPC disabled?
- Firmware update available?

Check OS configuration

- Kernel from distribution or custom?
- Up-to-date kernel?
- `intel_pstate` or `intel_idle` flags on command line?

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check firmware configuration

- C-states disabled/limited?
- PC-states disabled/limited?
- SpeedStep (EIST) disabled?
- HWP/SpeedShift/CPPC disabled?
- Firmware update available?

Check OS configuration

- Kernel from distribution or custom?
- Up-to-date kernel?
- `intel_pstate` or `intel_idle` flags on command line?

Try playing with tunables in `powertop` tool

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check firmware configuration

- C-states disabled/limited?
- PC-states disabled/limited?
- SpeedStep (EIST) disabled?
- HWP/SpeedShift/CPPC disabled?
- Firmware update available?

Check OS configuration

- Kernel from distribution or custom?
- Up-to-date kernel?
- `intel_pstate` or `intel_idle` flags on command line?

Try playing with tunables in `powertop` tool

Check other `turbostat` columns

- NMIs and SMIs should be 0
  - non-maskable and system management interrupts
- IRQs shouldn't be too high

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check periphery hardware power management

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check periphery hardware power management

```
$ sudo lspci -vvv
[…]
LnkCap: Port #0, Speed 5GT/s, Width x1, ASPM L0s L1, Exit Latency L0s <512ns, L1 <32us
                        ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp+
                LnkCtl: ASPM Disabled; RCB 64 bytes, LnkDisable- CommClk+
                        ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
[…]
```

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check periphery hardware power management

Active State Power Management (ASPM) supported

```
$ sudo lspci -vvv
[…]
LnkCap: Port #0, Speed 5GT/s, Width x1, ASPM L0s L1, Exit Latency L0s <512ns, L1 <32us
                        ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp+
                LnkCtl: ASPM Disabled; RCB 64 bytes, LnkDisable- CommClk+
                        ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
[…]
```

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check periphery hardware power management

Active State Power Management (ASPM) supported

```
$ sudo lspci -vvv
[…]
LnkCap: Port #0, Speed 5GT/s, Width x1, ASPM L0s L1, Exit Latency L0s <512ns, L1 <32us
                    ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp+
                LnkCtl: ASPM Disabled; RCB 64 bytes, LnkDisable- CommClk+
                    ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
[…]
```

But disabled!
⇒ Check firmware configuration

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check periphery hardware power management

Active State Power Management (ASPM) supported

```
$ sudo lspci -vvv
[…]
LnkCap: Port #0, Speed 5GT/s, Width x1, ASPM L0s L1, Exit Latency L0s <512ns, L1 <32us
                ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp+
        LnkCtl: ASPM Disabled; RCB 64 bytes, LnkDisable- CommClk+
                ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
[…]
```

But disabled!
⇒ Check firmware configuration

If firmware setting seem fine:
Can try enabling ASPM via `setpci` tool or `pcie_aspm=force` on kernel command line

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check early turbostat output

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check early turbostat output

```
$ sudo turbostat
[…]
cpu0: MSR_MISC_PWR_MGMT: 0x00403040 (ENable-EIST_Coordination DISable-EPB DISable-OOB)
cpu0: MSR_PM_ENABLE: 0x00000001 (HWP)
cpu0: MSR_HWP_CAPABILITIES: 0x05081422 (high 34 guar 20 eff 8 low 5)
cpu0: MSR_HWP_REQUEST: 0x20002208 (min 8 max 34 des 0 epp 0x20 window 0x0 pkg 0x0)
cpu0: MSR_HWP_REQUEST_PKG: 0x8000ff00 (min 0 max 255 des 0 epp 0x80 window 0x0)
cpu0: MSR_HWP_STATUS: 0x00000004 (No-Guaranteed_Perf_Change, Excursion_Min)
cpu0: EPB: 6 (balanced)
cpu0: MSR_IA32_POWER_CTL: 0x1200a40059 (C1E auto-promotion: DISabled)
C-state Pre-wake: ENabled
cpu0: MSR_PKG_CST_CONFIG_CONTROL: 0x00000403 (UNlocked, pkg-cstate-limit=3 (pc6r))
[…]
```

Practical Tips for Idle Consumption

What if low PC-states are never reached?

Check early turbostat output

```
$ sudo turbostat
[…]
cpu0: MSR_MISC_PWR_MGMT: 0x00403040 (ENable-EIST_Coordination DISable-EPB DISable-OOB)
cpu0: MSR_PM_ENABLE: 0x00000001 (HWP)
cpu0: MSR_HWP_CAPABILITIES: 0x05081422 (high 34 guar 20 eff 8 low 5)
cpu0: MSR_HWP_REQUEST: 0x20002208 (min 8 max 34 des 0 epp 0x20 window 0x0 pkg 0x0)
cpu0: MSR_HWP_REQUEST_PKG: 0x8000ff00 (min 0 max 255 des 0 epp 0x80 window 0x0)
cpu0: MSR_HWP_STATUS: 0x00000004 (No-Guaranteed_Perf_Change, Excursion_Min)
cpu0: EPB: 6 (balanced)
cpu0: MSR_IA32_POWER_CTL: 0x1200a40059 (C1E auto-promotion: DISabled)
C-state Pre-wake: ENabled
cpu0: MSR_PKG_CST_CONFIG_CONTROL: 0x00000403 (UNlocked, pkg-cstate-limit=3 (pc6r))
[…]
```

This is hard to read, but may contain the answer.

Practical Tips for Reliable Measurements

Practical Tips for Reliable Measurements

Disable Turbo Boost

- Creates energy and performance fluctuations

Practical Tips for Reliable Measurements

Disable Turbo Boost

- Creates energy and performance fluctuations

Disable short-term RAPL power limits

- Again, energy fluctuations

Practical Tips for Reliable Measurements

Disable Turbo Boost

- Creates energy and performance fluctuations

Disable short-term RAPL power limits

- Again, energy fluctuations

Disable SpeedShift/HWP/CPPC

- **MUST** be disabled via firmware settings
  - OS can not disable HWP when already enabled by firmware
  - Boot kernel with `intel_pstate=no_hwp`
- Set fixed P-state for all cores for predictable energy under load/idle
  - `sudo cpupower frequency-set -g performance`
  - `sudo cpupower frequency-set -g powersave --max …`

Permission Control for RAPL Interface

Permission Control for RAPL Interface

```
$ ls -la /sys/class/powercap/intel-rapl/intel-rapl:0
[…]
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_power_limit_uw
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_time_window_us
-r-------- 1 root root 4096 May 17 19:05 energy_uj
[…]
```

Permission Control for RAPL Interface

```
$ ls -la /sys/class/powercap/intel-rapl/intel-rapl:0
[…]
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_power_limit_uw
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_time_window_us
-r-------- 1 root root 4096 May 17 19:05 energy_uj
[…]
$ cat /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
cat: '/sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj': Permission denied
```

Permission Control for RAPL Interface

```
$ ls -la /sys/class/powercap/intel-rapl/intel-rapl:0
[…]
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_power_limit_uw
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_time_window_us
-r-------- 1 root root 4096 May 17 19:05 energy_uj
[…]
$ cat /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
cat: '/sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj': Permission denied
$ sudo chmod o+r /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
```

Permission Control for RAPL Interface

```
$ ls -la /sys/class/powercap/intel-rapl/intel-rapl:0
[…]
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_power_limit_uw
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_time_window_us
-r-------- 1 root root 4096 May 17 19:05 energy_uj
[…]
$ cat /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
cat: '/sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj': Permission denied
$ sudo chmod o+r /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
$ cat /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
5199740289
```

Permission Control for RAPL Interface

```
$ ls -la /sys/class/powercap/intel-rapl/intel-rapl:0
[…]
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_power_limit_uw
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_time_window_us
-r-------- 1 root root 4096 May 17 19:05 energy_uj
[…]
$ cat /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
cat: '/sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj': Permission denied
$ sudo chmod o+r /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
$ cat /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
5199740289
```

sysfs is a file system like any other, but transient

Permission Control for RAPL Interface

```
$ ls -la /sys/class/powercap/intel-rapl/intel-rapl:0
[…]
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_power_limit_uw
-rw-r--r-- 1 root root 4096 May 17 19:05 constraint_0_time_window_us
-r-------- 1 root root 4096 May 17 19:05 energy_uj
[…]
$ cat /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
cat: '/sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj': Permission denied
$ sudo chmod o+r /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
$ cat /sys/class/powercap/intel-rapl/intel-rapl:0/energy_uj
5199740289
```

`sysfs` is a file system like any other, but transient

Write a `systemd` unit to preserve permissions across boots
ChatGPT can do that for you ☺

System Suspend

I do not need my laptop to run all day long

System Suspend

I do not need my laptop to run all day long

I want to continue where I was when I come back

System Suspend

I do not need my laptop to run all day long

I want to continue where I was when I come back

But I don’t want to waste electricity

System Suspend

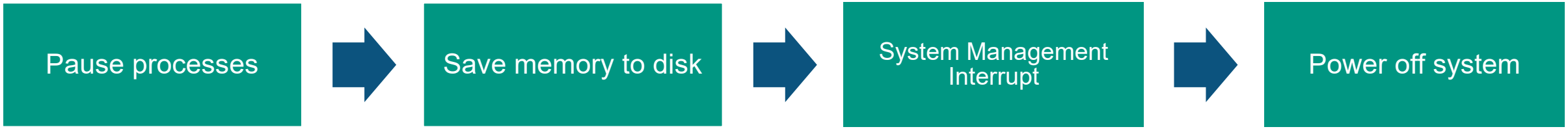
ACPI S4 (suspend-to-disk)

Save memory to disk and power off all hardware
⇒ zero power while suspended, slow resume

ACPI S4 (suspend-to-disk)

Save memory to disk and power off all hardware
⇒ zero power while suspended, slow resume

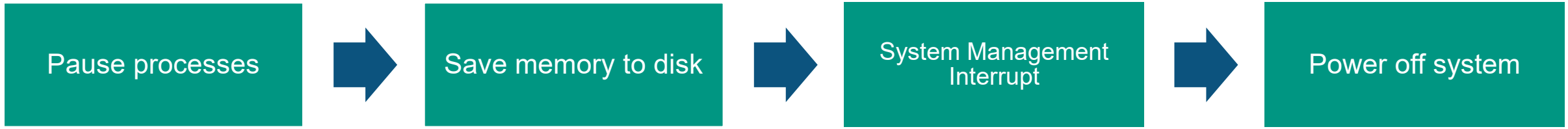
Suspend Flow



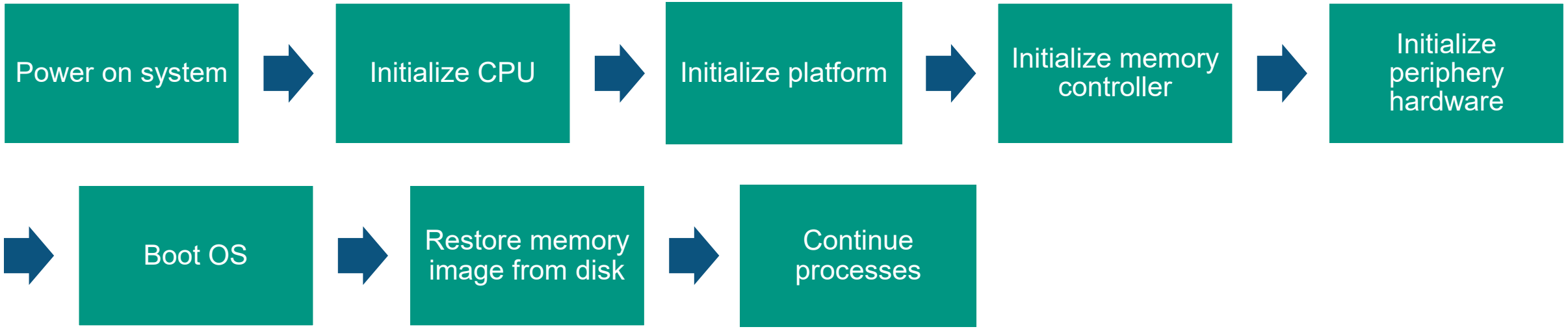
ACPI S4 (suspend-to-disk)

Save memory to disk and power off all hardware
⇒ zero power while suspended, slow resume

Suspend Flow



Resume Flow



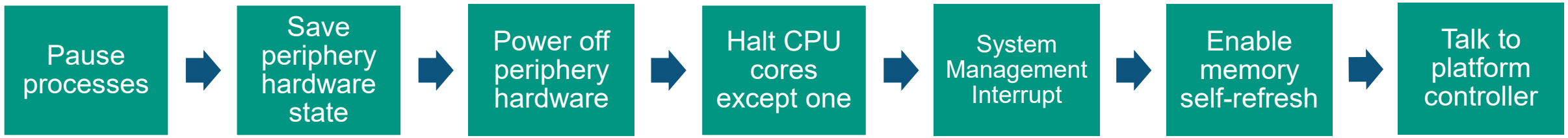
ACPI S3 (suspend-to-memory)

Power off all hardware except system memory
⇒ quick resume times, only memory consumes power

ACPI S3 (suspend-to-memory)

Power off all hardware except system memory
⇒ quick resume times, only memory consumes power

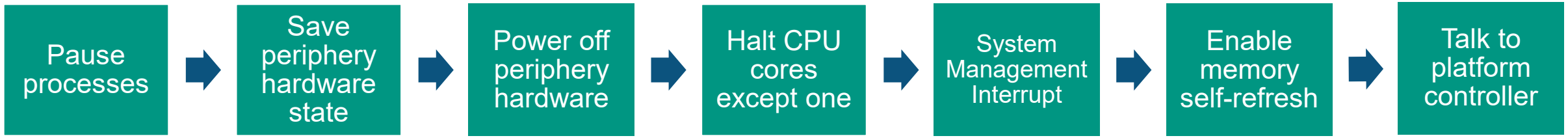
Suspend Flow



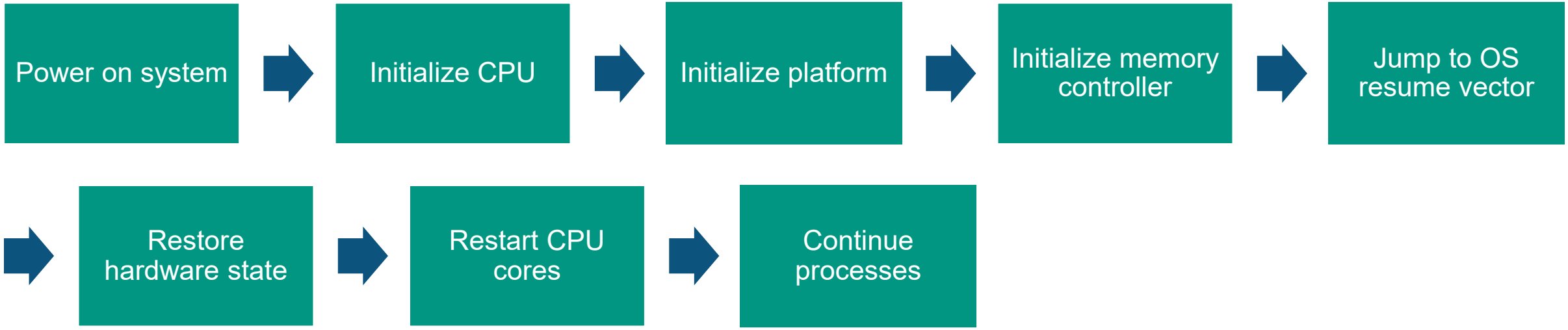
ACPI S3 (suspend-to-memory)

Power off all hardware except system memory
⇒ quick resume times, only memory consumes power

Suspend Flow



Resume Flow



ACPI S3: Design Implications

Power Supply Units (PSUs) inefficient at low load

ACPI S3: Design Implications

Power Supply Units (PSUs) inefficient at low load

- Need separate (smaller) power rail to increase power conversion efficiency in suspended state (also required for legal compliance)
  - Increases PSU production cost
  - Mainboard needs electrical switch
    - Increases mainboard production cost
  - Controller IC needed to manage memory power supply while CPU is off
    - Increases mainboard production cost

ACPI S3: Design Implications

Power Supply Units (PSUs) inefficient at low load

- Need separate (smaller) power rail to increase power conversion efficiency in suspended state (also required for legal compliance)
  - Increases PSU production cost
  - Mainboard needs electrical switch
    - Increases mainboard production cost
  - Controller IC needed to manage memory power supply while CPU is off
    - Increases mainboard production cost

System firmware (BIOS) needs to be involved

ACPI S3: Design Implications

Power Supply Units (PSUs) inefficient at low load

- Need separate (smaller) power rail to increase power conversion efficiency in suspended state (also required for legal compliance)
  - Increases PSU production cost
  - Mainboard needs electrical switch
    - Increases mainboard production cost
  - Controller IC needed to manage memory power supply while CPU is off
    - Increases mainboard production cost

System firmware (BIOS) needs to be involved

- Firmware rarely a prime example of good engineering
  - Bugs may ship and never get fixed
  - Often incredibly slow
- Resume implies partial firmware re-initialization
  - May cause several seconds delay
- S3 firmware implementations rarely achieved 100% reliability
  - Bugs common even after S3 has existed for decades

Reality Check

Reality Check

Workstation under your desk

Uli Cooler-Best Store (aliexpress.com)

Reality Check

Workstation under your desk

Uli Cooler-Best Store (aliexpress.com)

- PSU and mainboard follow ATX specification
- ATX mandates separate 12V standby power rail
- Single 230V AC input
- ≤1 kW per machine
- Suspend & resume every day
- Boot-up speed (firmware performance) matters somewhat

Reality Check

Workstation under your desk

Uli Cooler-Best Store (aliexpress.com)

- PSU and mainboard follow ATX specification
- ATX mandates separate 12V standby power rail
- Single 230V AC input
- ≤1 kW per machine
- Suspend & resume every day
- Boot-up speed (firmware performance) matters somewhat

19” server in your server room

yakkaroo.de

Reality Check

Workstation under your desk

Uli Cooler-Best Store (aliexpress.com)

- PSU and mainboard follow ATX specification
- ATX mandates separate 12V standby power rail
- Single 230V AC input
- ≤1 kW per machine
- Suspend & resume every day
- Boot-up speed (firmware performance) matters somewhat

19” server in your server room

yakkaroo.de

- No widespread governing specification
- Two 230V AC inputs per machine
- Few kW per machine
- Never suspend
- Never reboot
- Firmware designed for reliability, availability, serviceability (RAS)
  - Boot speed barely matters
- No strong incentive to optimize for low load efficiency

Reality Check

Workstation under your desk

Uli Cooler-Best Store (aliexpress.com)

- PSU and mainboard follow ATX specification
- ATX mandates separate 12V standby power rail
- Single 230V AC input
- ≤1 kW per machine
- Suspend & resume every day
- Boot-up speed (firmware performance) matters somewhat

19” server in your server room

yakkaroo.de

- No widespread governing specification
- Two 230V AC inputs per machine
- Few kW per machine
- Never suspend
- Never reboot
- Firmware designed for reliability, availability, serviceability (RAS)
  - Boot speed barely matters
- No strong incentive to optimize for low load efficiency

What hyperscalers are building

Reality Check

Workstation under your desk

Uli Cooler-Best Store (aliexpress.com)

- PSU and mainboard follow ATX specification
- ATX mandates separate 12V standby power rail
- Single 230V AC input
- ≤1 kW per machine
- Suspend & resume every day
- Boot-up speed (firmware performance) matters somewhat

19” server in your server room

yakkaroo.de

- No widespread governing specification
- Two 230V AC inputs per machine
- Few kW per machine
- Never suspend
- Never reboot
- Firmware designed for reliability, availability, serviceability (RAS)
  - Boot speed barely matters
- No strong incentive to optimize for low load efficiency

What hyperscalers are building

- Follows OCP specifications (e.g., Diablo 400)
- 480V AC input
- 400V DC single busbar per rack
- 1 MW per rack
- ABSOLUTELY never suspend
  - Spread workloads across machines to minimize latencies
- Machines never idle

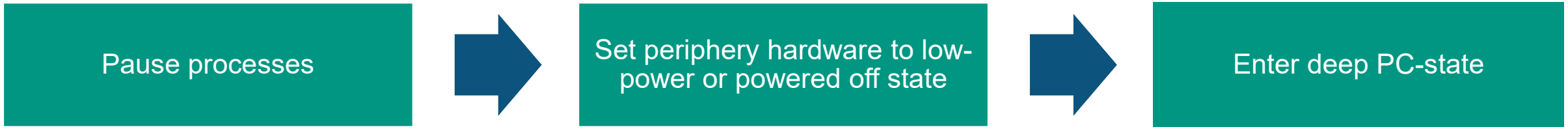
Suspend-to-idle / S0ix / Modern Standby

Leave CPU powered in deep PC-state
⇒ slightly more power than S3, faster resume

Suspend-to-idle / S0ix / Modern Standby

Leave CPU powered in deep PC-state
⇒ slightly more power than S3, faster resume

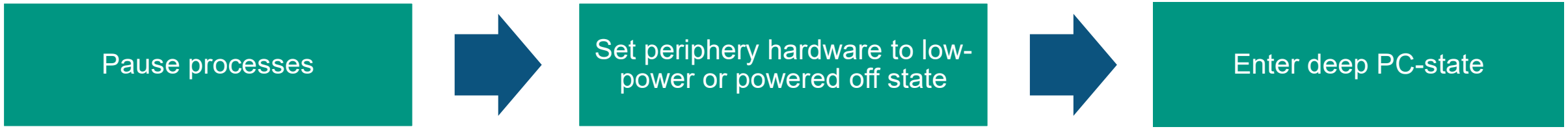
Suspend Flow



Suspend-to-idle / S0ix / Modern Standby

Leave CPU powered in deep PC-state
⇒ slightly more power than S3, faster resume

Suspend Flow



Resume Flow

Suspend-to-idle: Design Implications

OS and driver bugs much more relevant

Suspend-to-idle: Design Implications

OS and driver bugs much more relevant

- Bugs in OS or hardware drivers can prevent CPU from reaching deep PC-state
  - May cause serious energy consumption while suspended
- But easier to fix than firmware?
- Linux typically slower than Windows due to serialized operations

Suspend-to-idle: Design Implications

OS and driver bugs much more relevant

- Bugs in OS or hardware drivers can prevent CPU from reaching deep PC-state
  - May cause serious energy consumption while suspended
- But easier to fix than firmware?
- Linux typically slower than Windows due to serialized operations

Firmware less relevant

Suspend-to-idle: Design Implications

OS and driver bugs much more relevant

- Bugs in OS or hardware drivers can prevent CPU from reaching deep PC-state
  - May cause serious energy consumption while suspended
- But easier to fix than firmware?
- Linux typically slower than Windows due to serialized operations

Firmware less relevant

- Dramatically shortens resume process
- Must configure platform hardware correctly during boot and provide proper ACPI tables
  - But not involved during actual suspend/resume process

Suspend-to-idle: Design Implications

No deep PC-states in server CPUs

Suspend-to-idle: Design Implications

No deep PC-states in server CPUs

- Fundamentally impossible to reach true low power state with suspend-to-idle

Suspend-to-idle: Design Implications

No deep PC-states in server CPUs

- Fundamentally impossible to reach true low power state with suspend-to-idle

Can receive notifications (*Connected Standby*)

Suspend-to-idle: Design Implications

No deep PC-states in server CPUs

- Fundamentally impossible to reach true low power state with suspend-to-idle

Can receive notifications (*Connected Standby*)

- OS can decide to leave network hardware enabled
  - Configure to wake up system on incoming network traffic on specific connections
- Less important on laptops/workstations, more relevant for tablets/phones
- Not possible with traditional ACPI suspend modes

Pro Tip

Want to experiment with ACPI S3 on server-grade CPUs?

Pro Tip

Want to experiment with ACPI S3 on server-grade CPUs?

Go for professional-grade workstation platforms!

Pro Tip

Want to experiment with ACPI S3 on server-grade CPUs?

Go for professional-grade workstation platforms!

Workstation CPUs typically derived from server CPUs

No deep PC-states, but suspend required for workstations

Pro Tip

Want to experiment with ACPI S3 on server-grade CPUs?

Go for professional-grade workstation platforms!

Workstation CPUs typically derived from server CPUs

No deep PC-states, but suspend required for workstations

ACPI S3 still supported!

Pro Tip

Want to experiment with ACPI S3 on server-grade CPUs?

Go for professional-grade workstation platforms!

Workstation CPUs typically derived from server CPUs

No deep PC-states, but suspend required for workstations

ACPI S3 still supported!

Example
Intel Granite Rapids-Workstation (*GNR-WS*)
derived from
Intel Granite Rapids (*GNR*, current server generation)

Pro Tip

Want to experiment with ACPI S3 on server-grade CPUs?

Go for professional-grade workstation platforms!

Workstation CPUs typically derived from server CPUs

No deep PC-states, but suspend required for workstations

ACPI S3 still supported!

Example
Intel Granite Rapids-Workstation (*GNR-WS*)
derived from
Intel Granite Rapids (*GNR*, current server generation)

Not every mainboard is guaranteed to support S3!

Our Research on CXL-Based Hybrid SSDs

Our Research on CXL-Based Hybrid SSDs

Traditional SSD with NVMe interface

Our Research on CXL-Based Hybrid SSDs

Traditional SSD with NVMe interface

- Block-based access
  - Always copy from/to system memory
- Asynchronous communication
  - Pause processes while waiting for completion

Our Research on CXL-Based Hybrid SSDs

Traditional SSD with NVMe interface

- Block-based access
  - Always copy from/to system memory
- Asynchronous communication
  - Pause processes while waiting for completion

CXL-based hybrid SSD

Our Research on CXL-Based Hybrid SSDs

Traditional SSD with NVMe interface

- Block-based access
  - Always copy from/to system memory
- Asynchronous communication
  - Pause processes while waiting for completion

CXL-based hybrid SSD

- Compute Express Link is a novel standard for memory semantics in periphery hardware
  - Byte-granular addressing
  - Cacheable
  - Synchronous load/store access
- A hybrid SSD combines
  - CXL-based memory semantics
  - Traditional block semantics

Our Research on CXL-Based Hybrid SSDs

Traditional SSD with NVMe interface

- Block-based access
  - Always copy from/to system memory
- Asynchronous communication
  - Pause processes while waiting for completion

CXL-based hybrid SSD

- Compute Express Link is a novel standard for memory semantics in periphery hardware
  - Byte-granular addressing
  - Cacheable
  - Synchronous load/store access
- A hybrid SSD combines
  - CXL-based memory semantics
  - Traditional block semantics

Our group is working on a custom hardware prototype

Our Research on CXL-Based Hybrid SSDs

Traditional SSD with NVMe interface

- Block-based access
  - Always copy from/to system memory
- Asynchronous communication
  - Pause processes while waiting for completion

CXL-based hybrid SSD

- Compute Express Link is a novel standard for memory semantics in periphery hardware
  - Byte-granular addressing
  - Cacheable
  - Synchronous load/store access
- A hybrid SSD combines
  - CXL-based memory semantics
  - Traditional block semantics

Our group is working on a custom hardware prototype

Can improve application performance/latencies

Our Research on CXL-Based Hybrid SSDs

Traditional SSD with NVMe interface

- Block-based access
  - Always copy from/to system memory
- Asynchronous communication
  - Pause processes while waiting for completion

CXL-based hybrid SSD

- Compute Express Link is a novel standard for memory semantics in periphery hardware
  - Byte-granular addressing
  - Cacheable
  - Synchronous load/store access
- A hybrid SSD combines
  - CXL-based memory semantics
  - Traditional block semantics

Our group is working on a custom hardware prototype

Can improve application performance/latencies

Can reduce CPU cycles required for I/O
⇒ less active time, more time in deep C-states

Our Research on System Suspend

Our Research on System Suspend

Can use hybrid SSD just like system memory

Our Research on System Suspend

Can use hybrid SSD just like system memory

Idea
Suspend-to-disk, but resume directly from SSD

Our Research on System Suspend

Can use hybrid SSD just like system memory

Idea
Suspend-to-disk, but resume directly from SSD

Eliminates need to copy system state into memory first

Allows to skip large sections of boot process

Combine with smart prefetching algorithm to reduce runtime performance hit

Our Research on System Suspend

Can use hybrid SSD just like system memory

Idea
Suspend-to-disk, but resume directly from SSD

Eliminates need to copy system state into memory first

Allows to skip large sections of boot process

Combine with smart prefetching algorithm to reduce runtime performance hit

Ongoing implementation with custom firmware

Our Research on System Suspend

Can use hybrid SSD just like system memory

Idea
Suspend-to-disk, but resume directly from SSD

Eliminates need to copy system state into memory first

Allows to skip large sections of boot process

Combine with smart prefetching algorithm to reduce runtime performance hit

Ongoing implementation with custom firmware

We aim for zero-power suspend state with S3-like resume latencies

Our Research on System Suspend

Can use hybrid SSD just like system memory

Idea
Suspend-to-disk, but resume directly from SSD

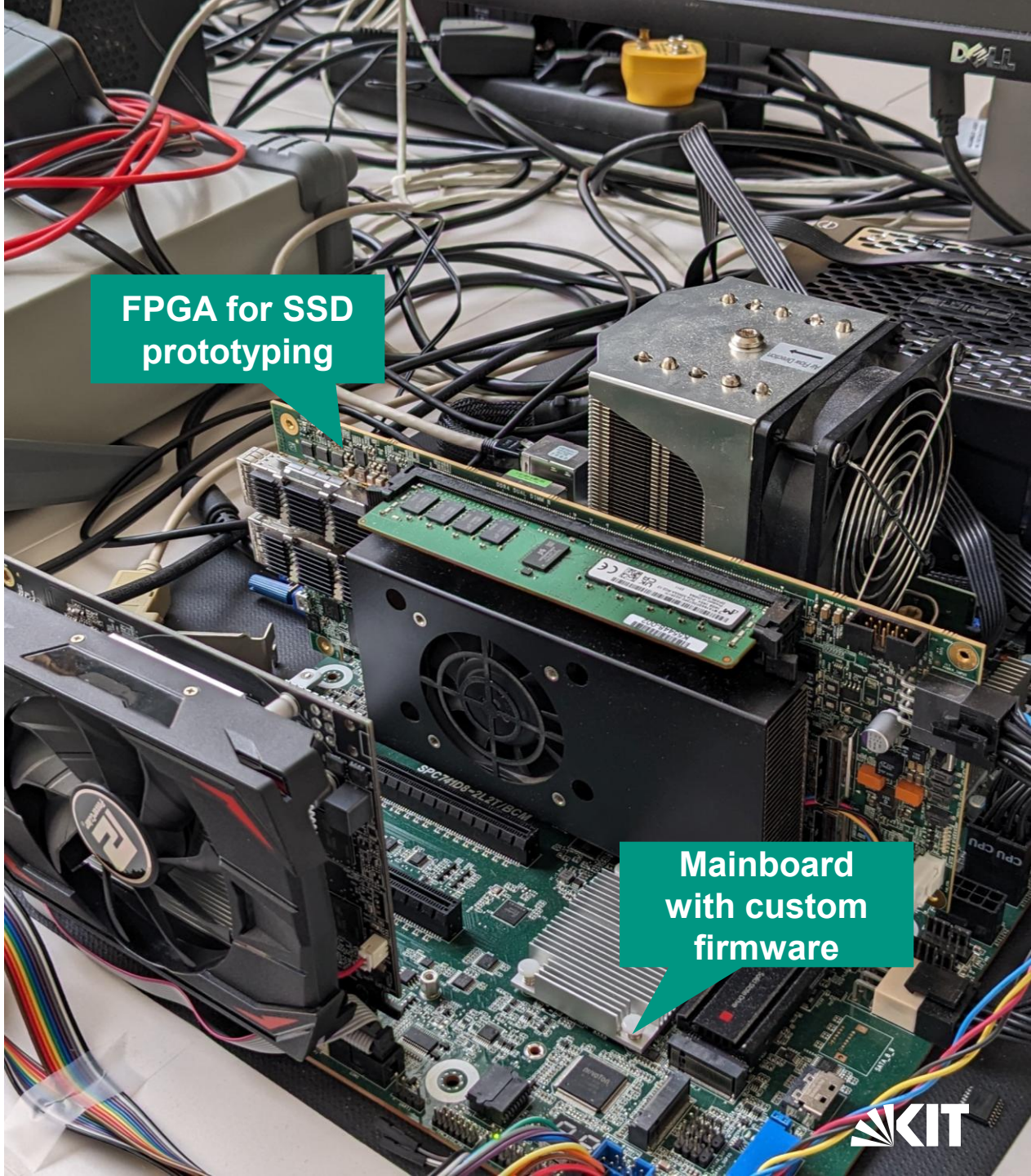
Eliminates need to copy system state into memory first

Allows to skip large sections of boot process

Combine with smart prefetching algorithm to reduce runtime performance hit

Ongoing implementation with custom firmware

We aim for zero-power suspend state with S3-like resume latencies



Reality Check: SSDs vs HDDs

Reality Check: SSDs vs HDDs

	Energy consumption over 5 years	Energy CO2e	Embodied CO2e	Total CO2e
HDD	183.9 kWh	79.6 kg	20 kg	99.6 kg
SSD	56.9 kWh	24.6 kg	160 kg	184 kg

Tannu et al. 2023

Reality Check: SSDs vs HDDs

	Energy consumption over 5 years	Energy CO2e	Embodied CO2e	Total CO2e
HDD	183.9 kWh	79.6 kg	20 kg	99.6 kg
SSD	56.9 kWh	24.6 kg	160 kg	184 kg

Tannu et al. 2023

SSDs are known for their energy efficiency

Reality Check: SSDs vs HDDs

	Energy consumption over 5 years	Energy CO2e	Embodied CO2e	Total CO2e
HDD	183.9 kWh	79.6 kg	20 kg	99.6 kg
SSD	56.9 kWh	24.6 kg	160 kg	184 kg

Tannu et al. 2023

SSDs are known for their energy efficiency

But manufacturing much more dirty

Reality Check: SSDs vs HDDs

	Energy consumption over 5 years	Energy CO2e	Embodied CO2e	Total CO2e
HDD	183.9 kWh	79.6 kg	20 kg	99.6 kg
SSD	56.9 kWh	24.6 kg	160 kg	184 kg

Tannu et al. 2023

SSDs are known for their energy efficiency

But manufacturing much more dirty

Want to reduce your carbon footprint? Buy less stuff!

Reading List (in no particular order)

Efraim et al. (2012): *Energy Aware Race to Halt: A Down to EARtH Approach for Platform Energy Management*

Lee et al. (2021): *GreenDIMM: OS-assisted DRAM Power Management for DRAM with a Sub-array Granularity Power-Down State*

Weissel et al. (2002): *Process Cruise Control: Event-Driven Clock Scaling for Dynamic Power Management*

Tannu et al. (2023): *The Dirty Secret of SSDs: Embodied Carbon*

Harris et al. (2022): *When Poll is More Energy Efficient than Interrupt*

Haj-Yahya et al. (2020): *FlexWatts: A Power- and Workload-Aware Hybrid Power Delivery Network for Energy-Efficient Microprocessors*

Zhou et al. (2025): *Sleeping with One Eye Open: Fast, Sustainable Storage with Sandman*

Meng et al. (2021): *Proactive Energy-Aware Adaptive Video Streaming on Mobile Devices*

Taylor (2012): *Is Dark Silicon Useful? Harnessing the Four Horsemen of the Coming Dark Silicon Apocalypse*

Maji et al. (2025): *Data Centers Carbon Emissions at Crossroads: An Empirical Study*

Anand et al. (1998): *High-level power analysis and optimization*

Khalil et al. (2025): *Transparent DAX Mappings: Towards Automatic Kernel Bypass with CXL-Based Hybrid SSDs*

Conclusion

- System power management is all about trade-offs
- P-states, C-states, PC-states govern your CPU's power consumption
- Suspend states have evolved (ACPI S3 -> suspend-to-idle)
  - Not to everyone's advantage
- Server power management is particularly hard
  - Latency requirements dominate everything
  - Interactive systems more graceful

Conclusion

- System power management is all about trade-offs
- P-states, C-states, PC-states govern your CPU's power consumption
- Suspend states have evolved (ACPI S3 -> suspend-to-idle)
  - Not to everyone's advantage
- Server power management is particularly hard
  - Latency requirements dominate everything
  - Interactive systems more graceful

Do not forget about embodied carbon!