# Techniques for Reducing the Connected-Standby Energy Consumption of Mobile Devices

***Jawad Haj-Yahya***[1]

*Yanos Sazeides*[2] *Mohammed Alser*[1] *Efraim Rotem*[3] *Onur Mutlu*[1]

[1] SAFARI [1] ETH zürich [2] University of Cyprus [3] intel

# Executive Summary

- **Motivation:** Mobile devices operate in **connected-standby** mode most of the time. We would like to make this mode more energy efficient.
- **Problem:** In connected-standby mode, mobile devices enter the **Deepest-Runtime-Idle-Power State (DRIPS)**. There are three sources of energy inefficiency in modern DRIPS:
  - Wake-up timer event is toggled in a high-leakage process using a high frequency clock.
  - Several IO signals are always powered on.
  - Processor context is preserved in high-leakage-power SRAMs.
- **Goal:** Reduce power consumption of DRIPS.
- **Mechanism:** **Optimized DRIPS (ODRIPS) based on they key ideas:**
  - Offload wake-up timer to a low leakage chip (e.g., chipset) with significantly slower clock.
  - Offload always-on IO functionality to power-gate all processor IOs.
  - Transfer processor context to a secure memory region inside DRAM.
- **Evaluation:**
  - We implement ODRIPS in **Intel's Skylake** mobile processor.
  - ODRIPS reduces the platform average power consumption in connected-standby mode by **22%**.

# Presentation Outline

**1. Connected-Standby and DRIPS Overview**

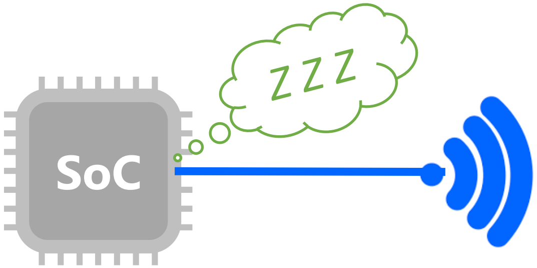
2. The ODRIPS Substrate
   I. Wake-up timer event handling
   II. Offload processor’s always-on IO functionality
   III. Transfers the processor context to DRAM

3. Evaluation

4. Summary

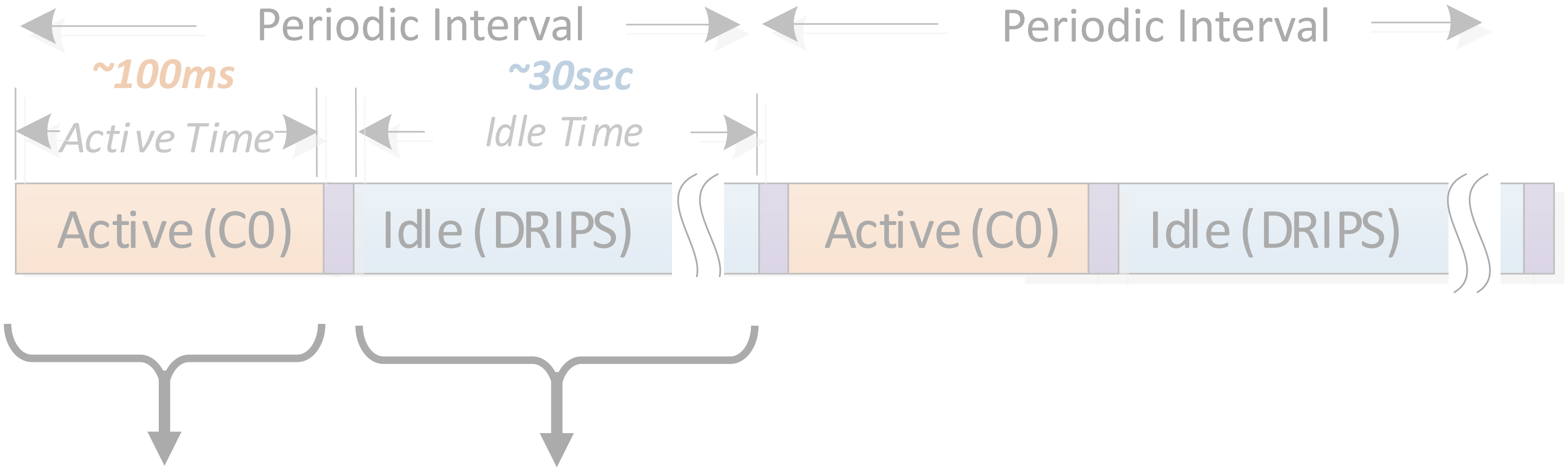
# Connected-Standby Mode (1)

- Mobile devices (phones, tablets, and laptops) are idle the majority (~90%) of the time.
- During idle periods, modern mobile devices
  - Operate at low-power state to increase battery life
  - Remain connected to a communication network for usability (e.g., for email notifications and phone calls).
- This operation mode is called **connected-standby**
  - Microsoft's Modern Standby
  - Apple's Power Nap



- In the connected-standby mode
  - The system spends most of its time in the Deepest-Runtime-Idle-Power-State (**DRIPS**)
  - Display panel is off during connected-standby

# Connected-Standby Mode (2)



Periodic Interval

Periodic Interval

~100ms

Active Time

~30sec

Idle Time

Active (C0)

Idle (DRIPS)

Active (C0)

Idle (DRIPS)

~0.5% of the time (3W)

➔ ~20% of the

~99.5% of the time ( ~60mW )

➔ ~80% of the average power

**~80% of the connected-standby platform average power is consumed in DRIPS**

# DRIPS: Deepest Runtime Idle Power State

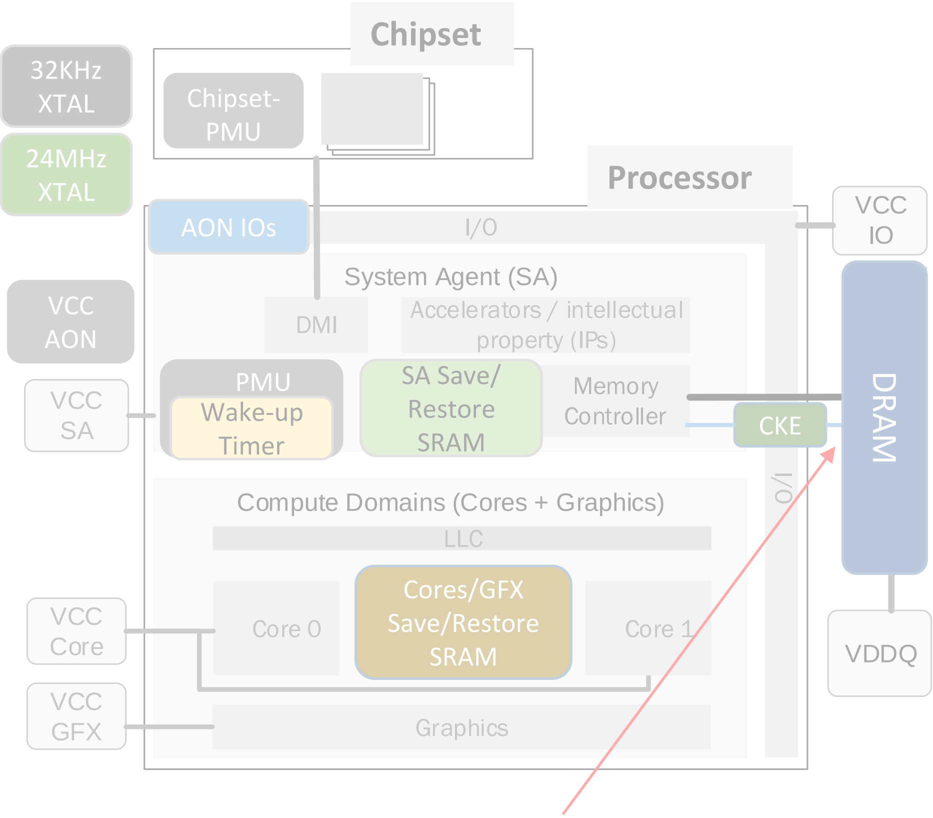
Three major power consumption sources in DRIPS :

**Chipset** includes relatively slow IO (e.g., USB/SATA/PCI) and system power management functions.
The chipset process is optimized for low-leakage and low-frequencies.

The **processor** operates in high frequencies (e.g., 3GHz).
The processor process is optimized for high frequency rather than low-leakage

3) Processor context is preserved in high-leakage power SRAMs (**9%** of the platform average power ).

**Intel Skylake Mobile Architecture**



**15%** of the platform average power

**We target these three inefficiencies**

# Presentation Outline

1. Connected-Standby and DRIPS Overview

2. **The ODRIPS Substrate**
    I. Wake-up timer event handling
    II. Offload processor’s always-on IO functionality
    III. Transfers the processor context to DRAM

3. Evaluation

4. Summary

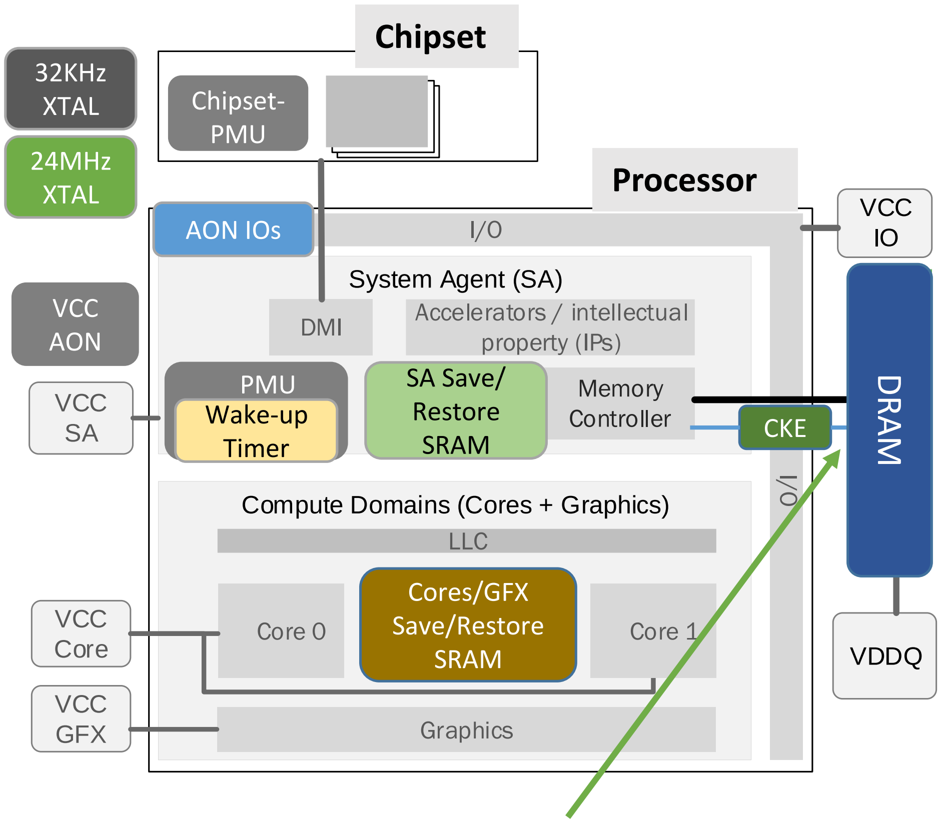
# ODRIPS: Optimized DRIPS

## ODRIPS consists of three key ideas

**Idea 1**. Offload wake-up timer to a low leakage chip (e.g., chipset) and significantly slower clock (**5%** of the platform average power ).

**Idea 2.** Offload always-on IO functionality to chipset and power-gate all processor IOs (**7%** of the platform average power).

**Idea 3.** Transfer processor context to a secure memory region inside DRAM (**9%** of the platform average power).



15% (idea in the paper)

SAFARI

# Presentation Outline

1. Connected-Standby and DRIPS Overview

**2. The ODRIPS Substrate**

  **I. Wake-up timer event handling**

  II. Offload processor's always-on IO functionality

  III. Transfers the processor context to DRAM

3. Evaluation

4. Summary

# Idea 1: Wake-up Timer Handling

**Problem 1:** Wake-up timer event handling consumes 5% of platform power in DRIPS.

**Key idea 1:** Offload wake-up timer to a low-leakage chip (e.g., chipset) and significantly slower clock.

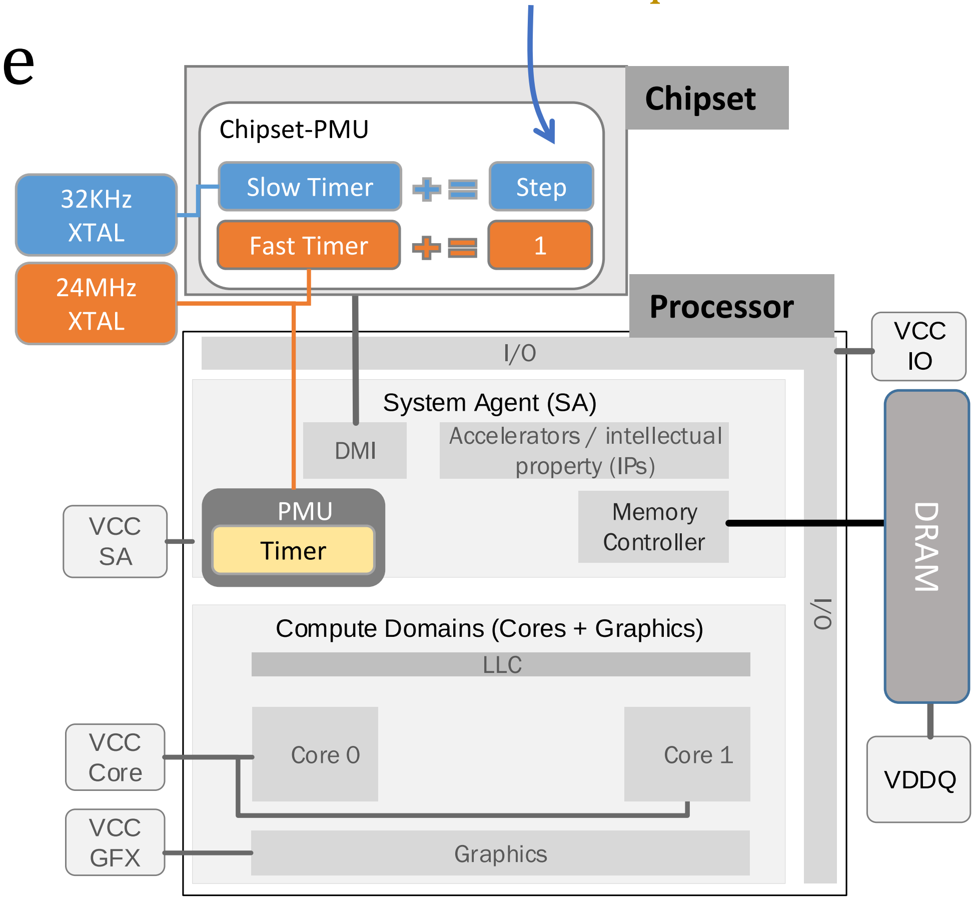
# Idea 1: Wake-up Timer Handling

Baseline Architecture

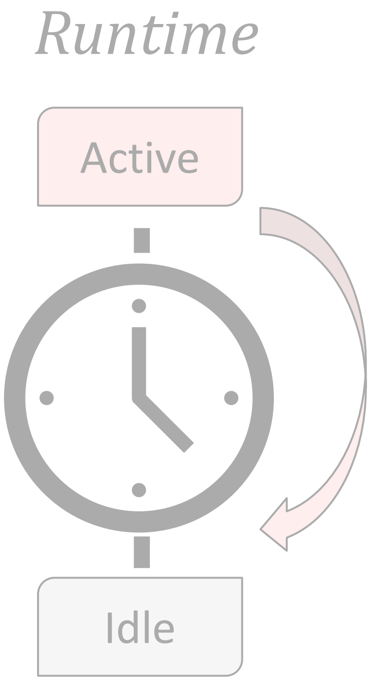
ODRIPS Architecture

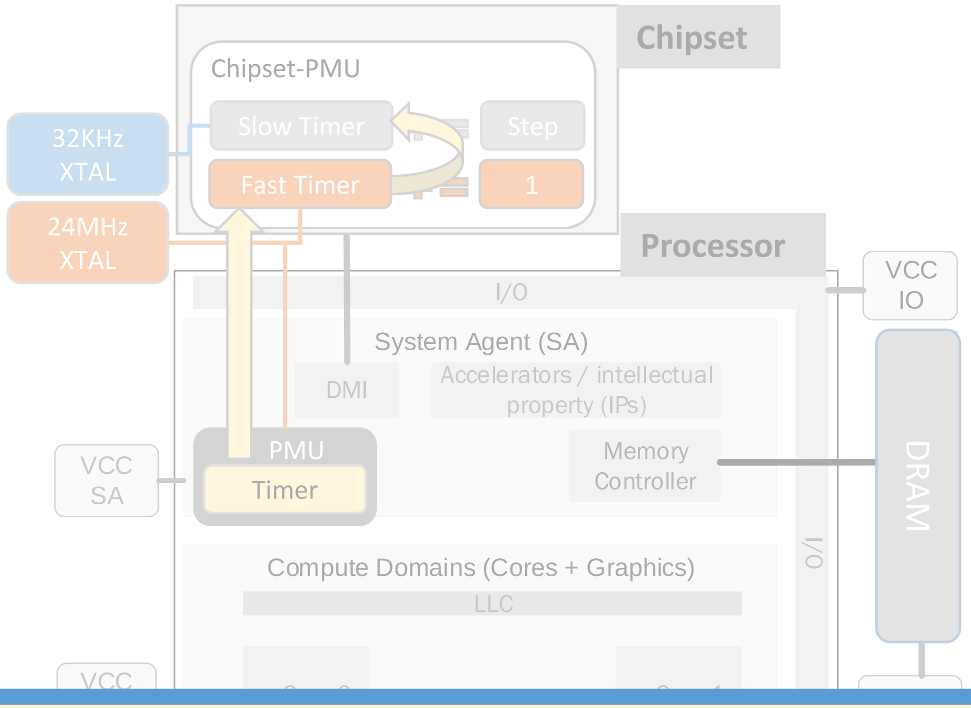
Step = 24MHz/32KHz (in fixed-point)

Calibration is required



# Idea 1: Wake-up Timer Handling





**This idea saves 5% of the DRIPS power**

# Presentation Outline

1. Connected-Standby and DRIPS Overview

**2. The ODRIPS Substrate**

   I. Wake-up timer event handling

   **II. Offload processor's always-on IO functionality**

   III. Transfers the processor context to DRAM

3. Evaluation

4. Summary

# Idea 2: Offload Always-On IOs

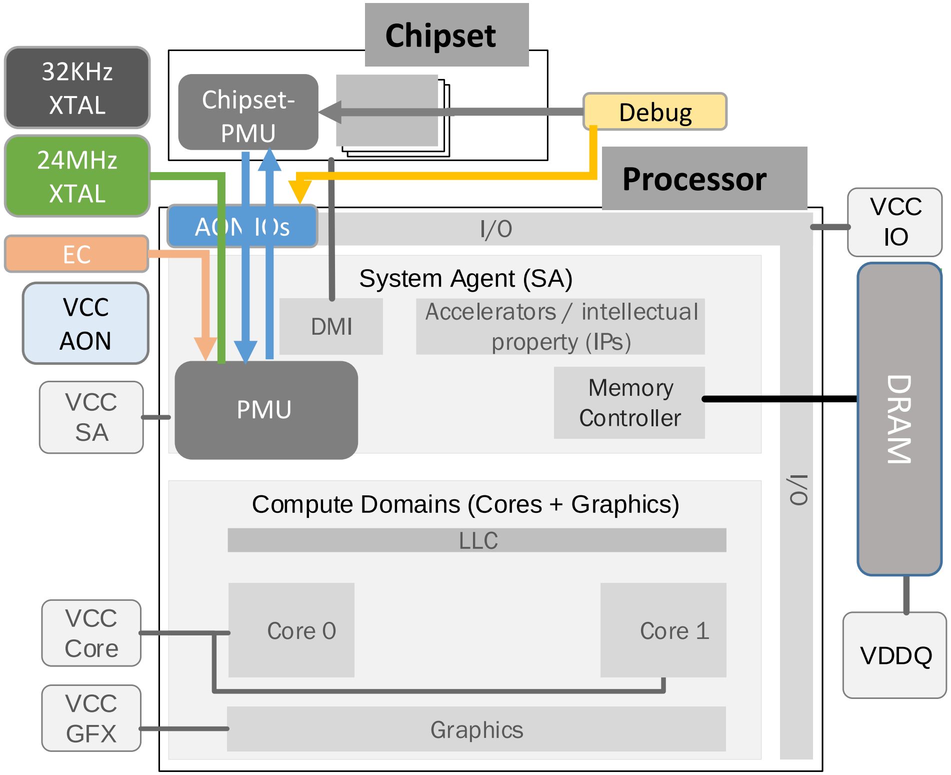
**Problem 2:** Several IO signals are always-on in DRIPS consuming **7%** of platform average power.

**Key idea 2:** Offload the always-on IO signals functionality to chipset and dynamically power-gate these IOs.

# Baseline Always-On IOs

## Always-On IOs:

- 24MHz clock
- Power management links
- Thermal reporting
- Debug (JTAG)



# Idea 2: Offload Always-On IOs

## 24MHz clock

- 24MHz clock is no longer needed after offloading the timer to chipset

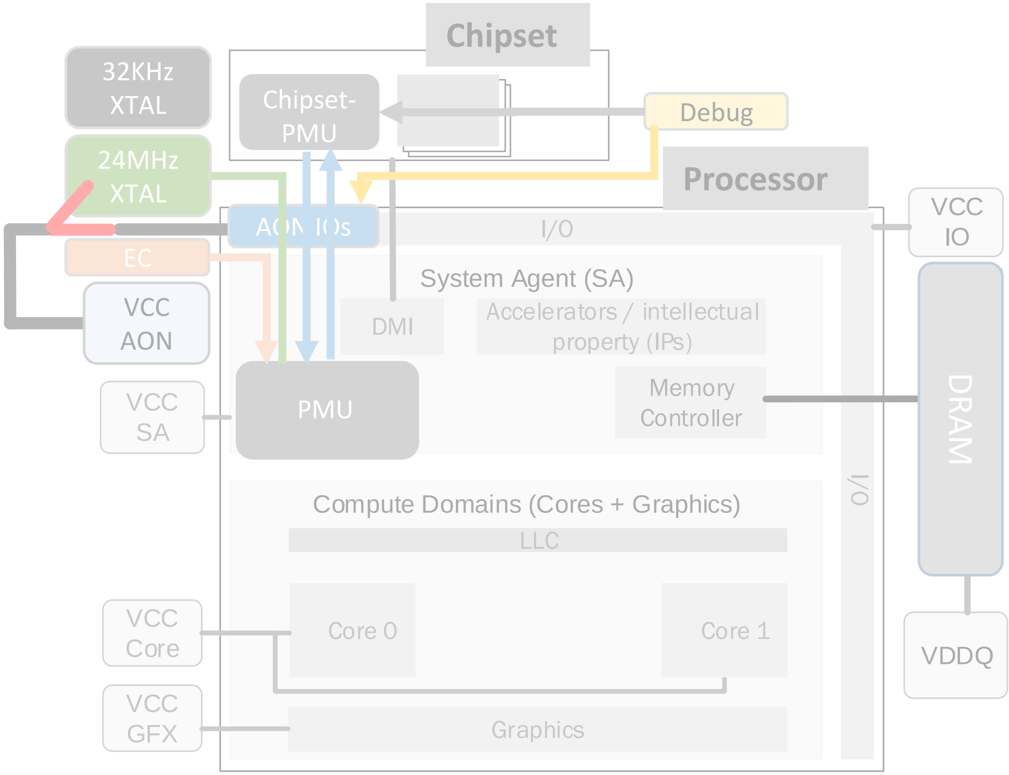
## Power management links

- No need for power management in ODRIPS

## Thermal reporting

- Offload the Embedded Controller (EC) thermal reporting to chipset using General Purpose IO (**GPIO**)

## Debug (JTAG)



**This idea saves 7% of the DRIPS power**

# Presentation Outline

1. Connected-Standby and DRIPS Overview

**2. The ODRIPS Substrate**

   I. Wake-up timer event handling

   II. Offload processor’s always-on IO functionality off-chip

   **III. Transfers the processor context to DRAM**

3. Evaluation

4. Summary

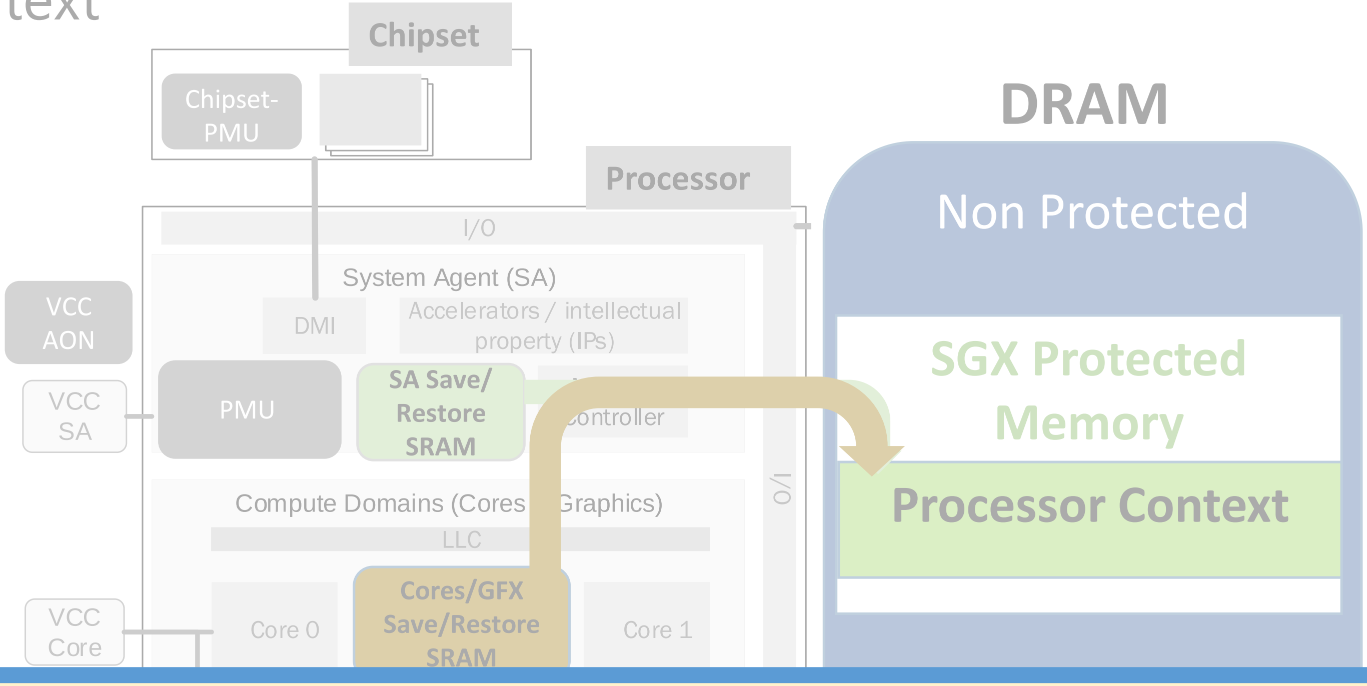
# Idea 3: Transfers Processor Context to DRAM

**Problem 3:** Leakage power consumption of the **Save/Restore SRAMs** that saves the processor context is high and consuming **9%** of the platform average power in DRIPS.

**Key idea 3:** Dynamically transfer the processor context from the SRAMs to **DRAM**.

# Idea 3: Transfers Processor Context to DRAM

- We move the processor context from save/restore SRAMs to **SGX protected Memory**
- A 200KB out of the 128MB of SGX memory is “stolen” to save the processor context



**This idea saves 9% of the DRIPS power**

# Presentation Outline

1. Connected-Standby and DRIPS Overview

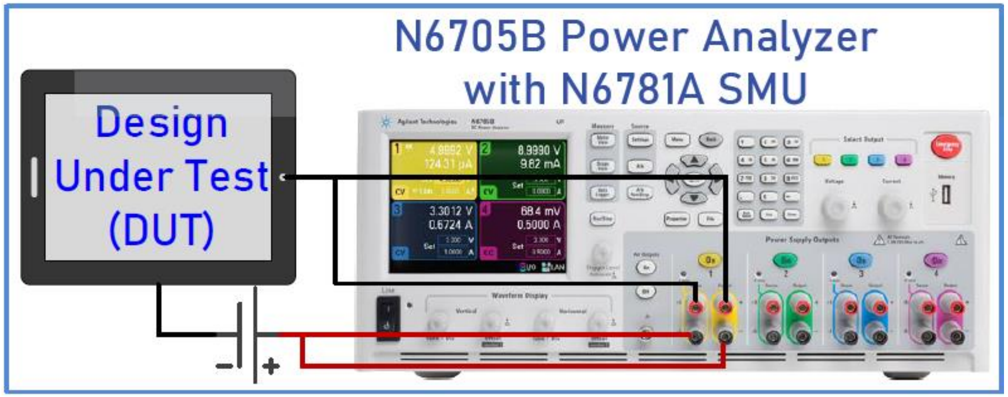
2. The ODRIPS Substrate
   I. Wake-up timer event handling
   II. Offload processor's always-on IO functionality off-chip
   III. Transfers the processor context to DRAM

3. **Evaluation**

4. Summary

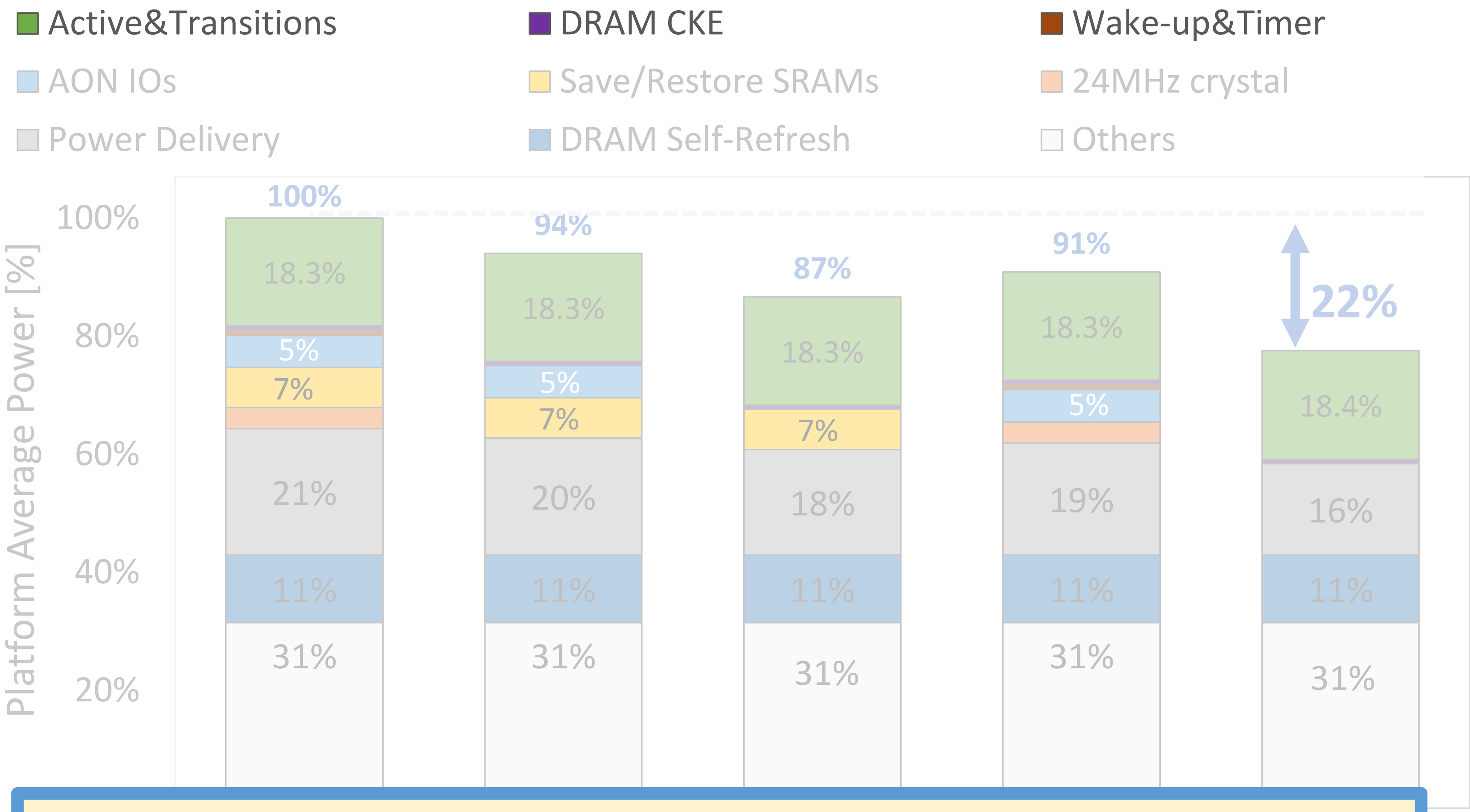
# Methodology

- Intel Skylake for mobile devices includes **ODRIPS**.
- We evaluate ODRIPS using a real Intel Skylake system.
- We used Keysight measurement instruments



- We use an in-house power model for sensitivity studies.

# Results



**ODRIPS reduces the connected-standby platform average power by 22%**

# Other Results in the Paper

- Using Non-Volatile Memories (NVMs) with ODRIPS
    - An idea to use Phase Change Memory (**PCM**) instead of DRAM
        - This idea reduces connected-standby platform average power by additional **15%**.
    - Use embedded MRAM (eMRAM) instead of on-chip SRAMs
- Connected-standby platform average power sensitivity to:
    - Core frequency in Active state
    - DRAM frequency in Active state

# Presentation Outline

1. Connected-Standby and DRIPS Overview

2. The ODRIPS Substrate
   I. Wake-up timer event handling
   II. Offload processor's always-on IO functionality off-chip
   III. Transfers the processor context to DRAM

3. Evaluation

4. **Summary**

# Summary

- **Motivation:** Mobile devices operate in **connected-standby** mode most of the time. We would like to make this mode more energy efficient.

- **Problem:** In connected-standby mode, mobile devices enter the **Deepest-Runtime-Idle-Power State (DRIPS)**. There are three sources of energy inefficiency in modern DRIPS:
  - Wake-up timer event is toggled in a high-leakage process using a high frequency clock.
  - Several IO signals are always powered on.
  - Processor context is preserved in high-leakage-power SRAMs.

- **Goal:** Reduce power consumption of DRIPS.

- **Mechanism:** **Optimized DRIPS (ODRIPS) based on they key ideas:**
  - Offload wake-up timer to a low leakage chip (e.g., chipset) with significantly slower clock.
  - Offload always-on IO functionality to power-gate all processor IOs.
  - Transfer processor context to a secure memory region inside DRAM.

- **Evaluation:**
  - We implement ODRIPS in **Intel's Skylake** mobile processor.
  - ODRIPS reduces the platform average power consumption in connected-standby mode by **22%**.

# Techniques for Reducing the Connected-Standby Energy Consumption of Mobile Devices

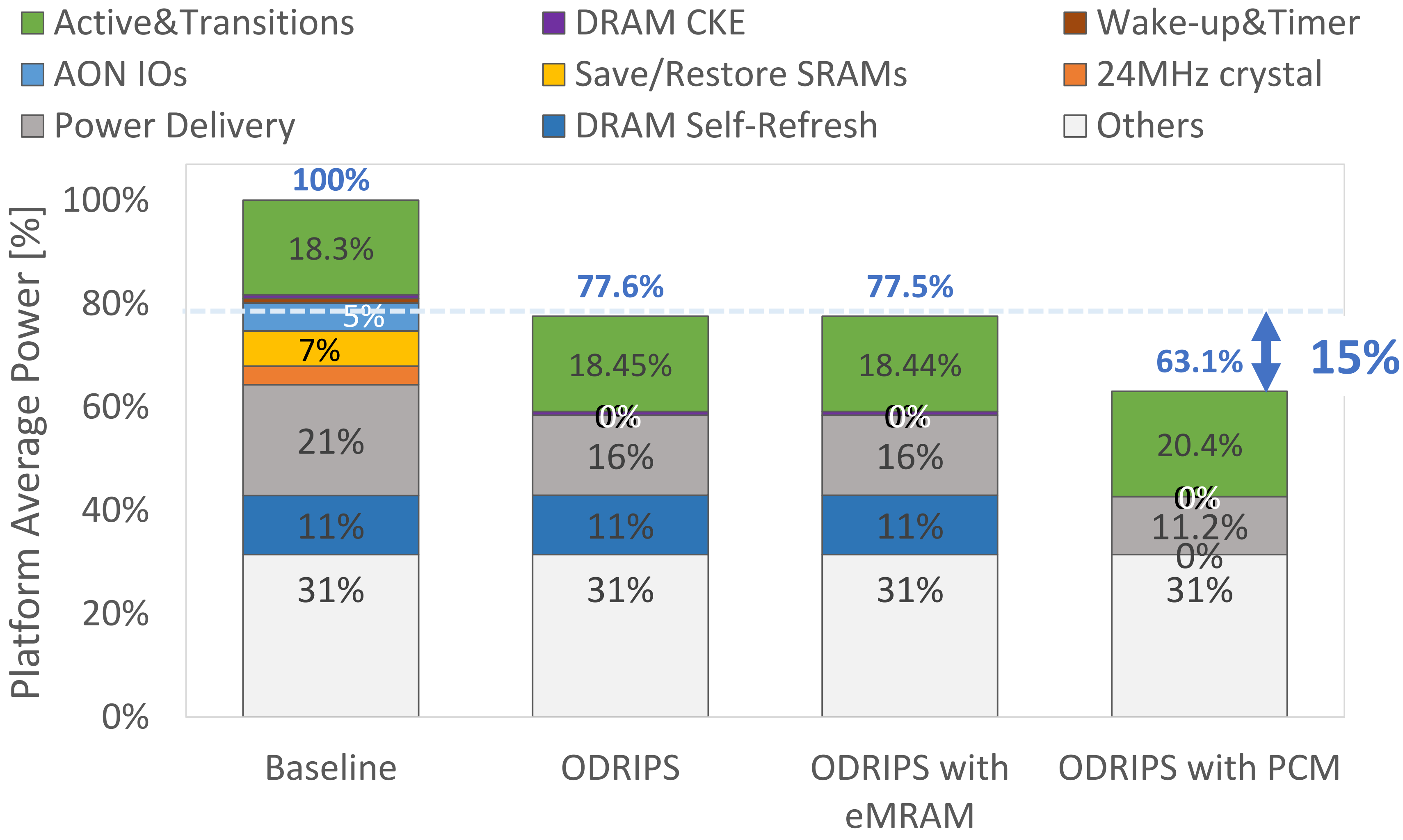
***Jawad Haj-Yahya***[1]

*Yanos Sazeides*[2] *Mohammed Alser*[1] *Efraim Rotem*[3] *Onur Mutlu*[1]

[1] SAFARI [1] ETH zürich [2] University of Cyprus [3] intel

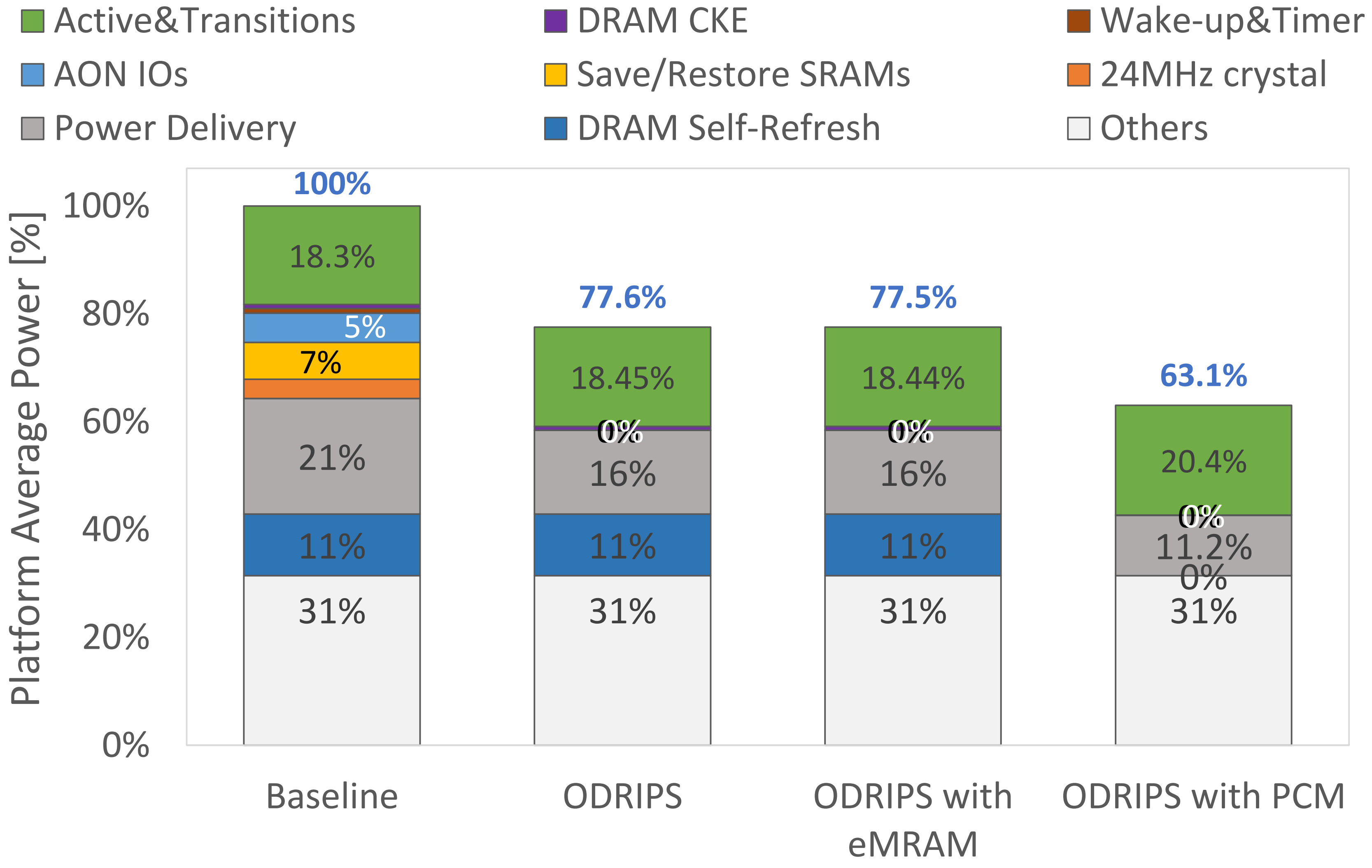
# Backup

# Results – Use NVMe



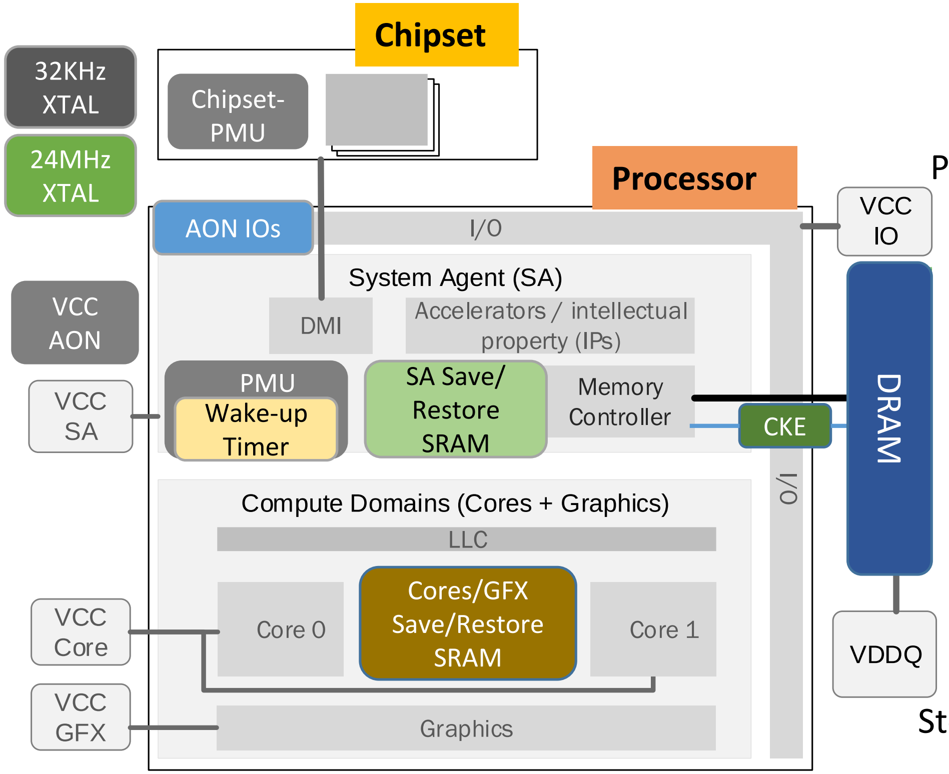
SAFARI

# Results – Core and DRAM Freq.

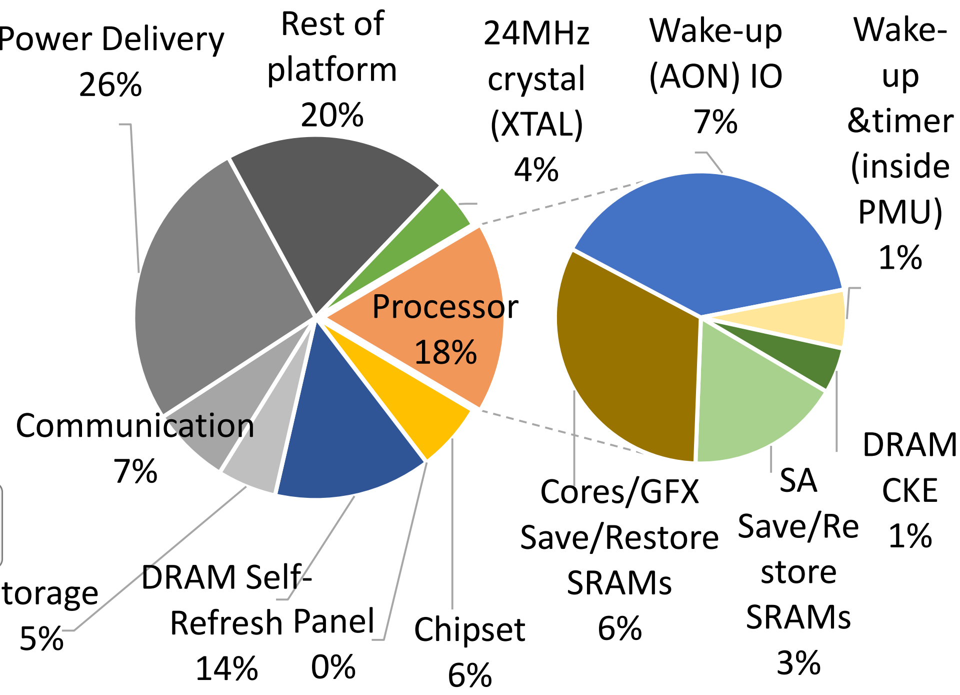


Core Frequencies DRAM Frequencies

# DRIPS: deepest runtime idle power state

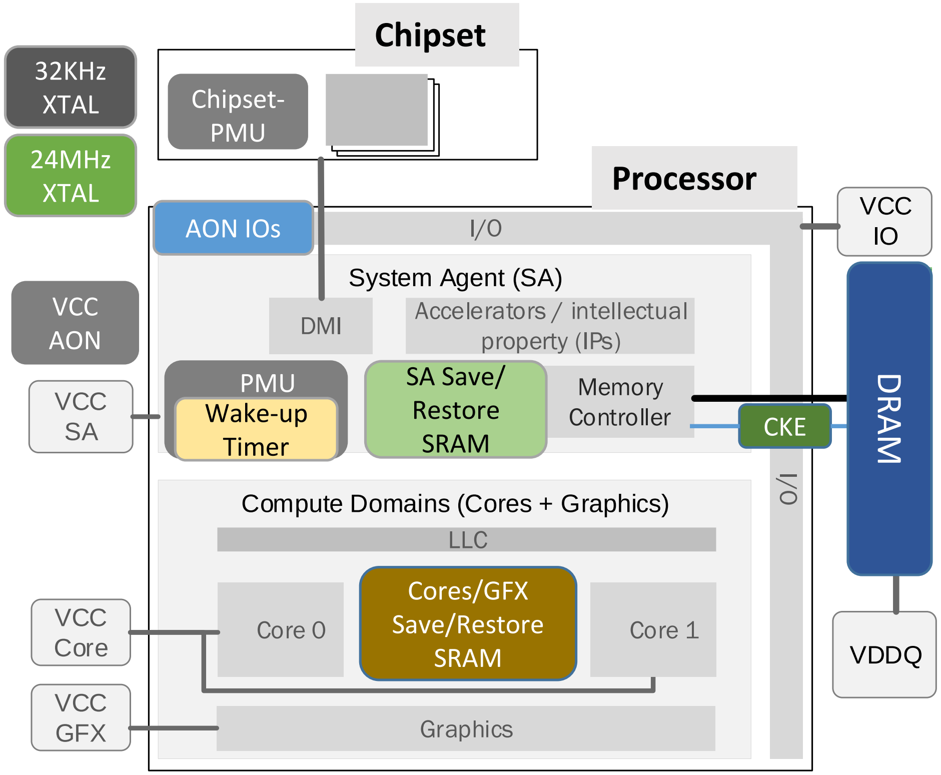


List all components



# ODRIPS: Optimized DRIPS

1) Offload wake-up timer to a low leakage chip (e.g., chipset) and significantly slower clock



2) Offload always-on IOs functionality to other chip and power-gate all processor IOs

3) Transfer processor context to a secure memory region inside DRAM