Transkernel: Bridging Monolithic Kernels to Peripheral Cores

Liwei Guo, Shuang Zhai, Yi Qiao, and Felix Xiaozhu Lin

Purdue ECE

http://xsel.rocks

What is Transkernel?

- A novel OS model to run **unmodified binary of a monolithic kernel**
  - on a **microcontroller-like core ...**
  - of **a heterogeneous SoC**
- Key techniques: dynamic binary translation + kernel service emulation

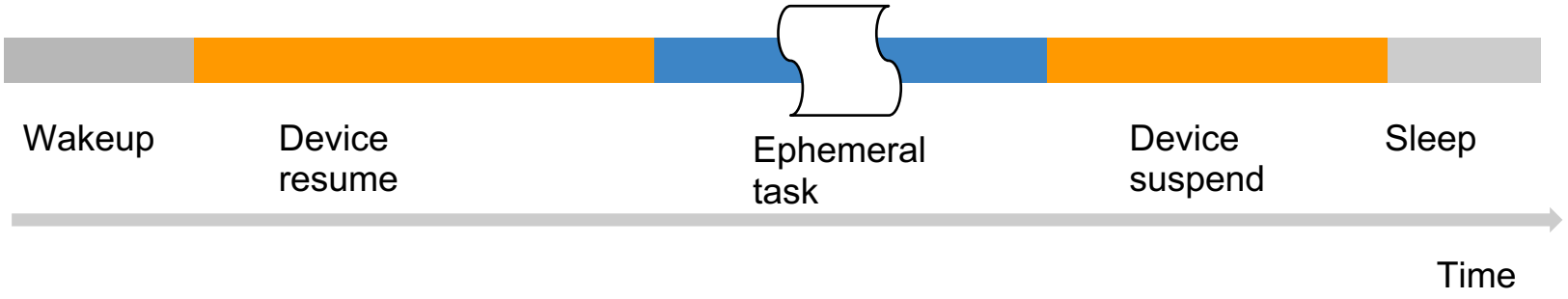
Motivation: Ephemeral tasks in smart things

Prevalent: push notifications, periodic data logging, etc.

- User tasks running on a monolithic kernel (e.g., Linux)

Energy-hungry: ~30% or higher battery drain in smart things [1]

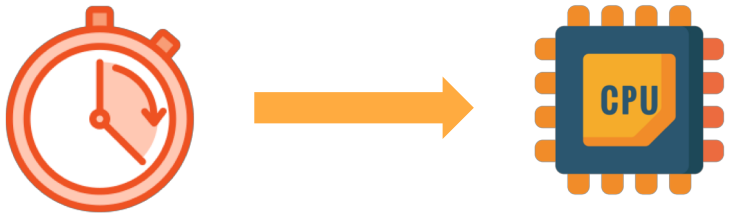
Device suspend/resume is the key bottleneck [2]



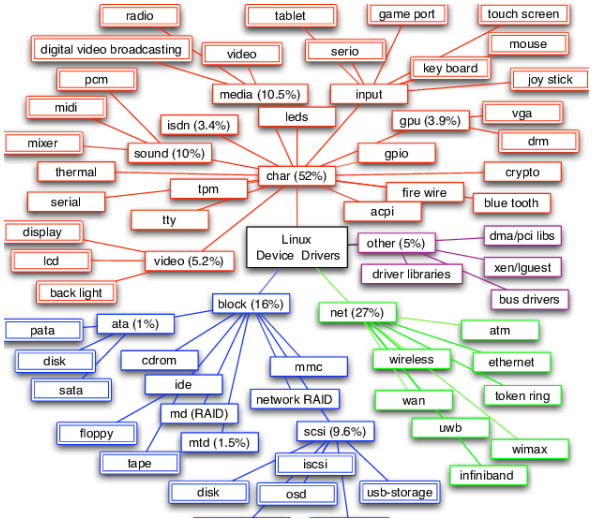
[1] Smartphone background activities in the wild: Origin, energy drain, and optimization, Chen et al., *MobiCom'15*
[2] Decelerating Suspend and Resume, Zhai et al., *Hotmobile'17*

Why is device suspend/resume so inefficient?

Slow power state transitions keep CPU waiting



Difficult to parallelize due to device dependencies

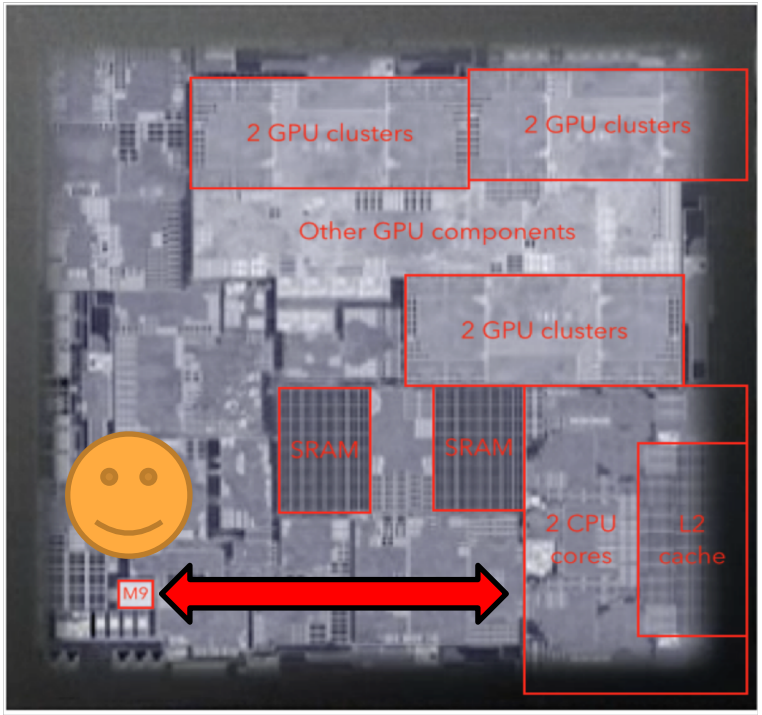


Understanding modern device drivers, Kadav et al., *ASPLOS'12*

Our proposal: suspend/resume on a peripheral core

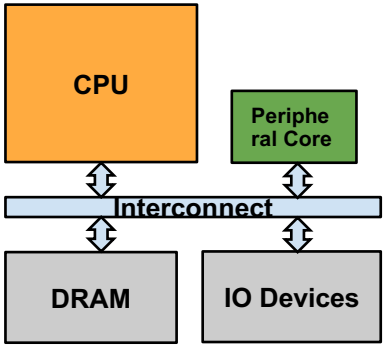
- **Benefit: Lower** idle power and **higher** busy execution efficiency

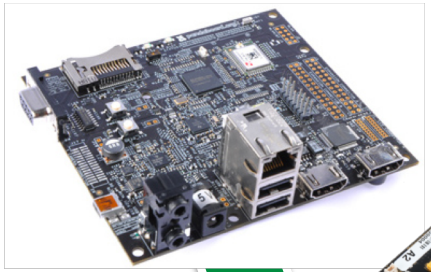
- Asymmetric processors
  - CPU + Peripheral core
- Heterogeneous, yet similar ISAs
  - Same family, different profile
  - e.g., ARMv7a + v7m
- Loose coupling
  - CPU can be turned on/off independently
- Shared platform resources
  - IRQs
  - DRAM



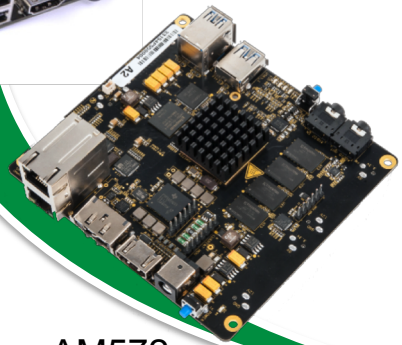
Apple A9 (Chipworks)

Many SoCs fit this hardware model



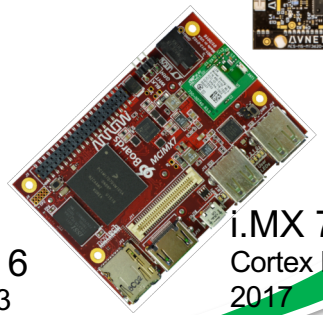


OMAP4460
Cortex M3 + A9
2010

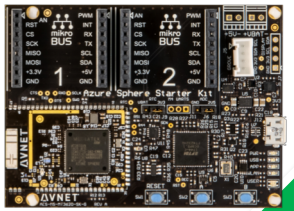


AM572x
Cortex M4 + A15
2014

iPhone 6
Cortex M3
2014



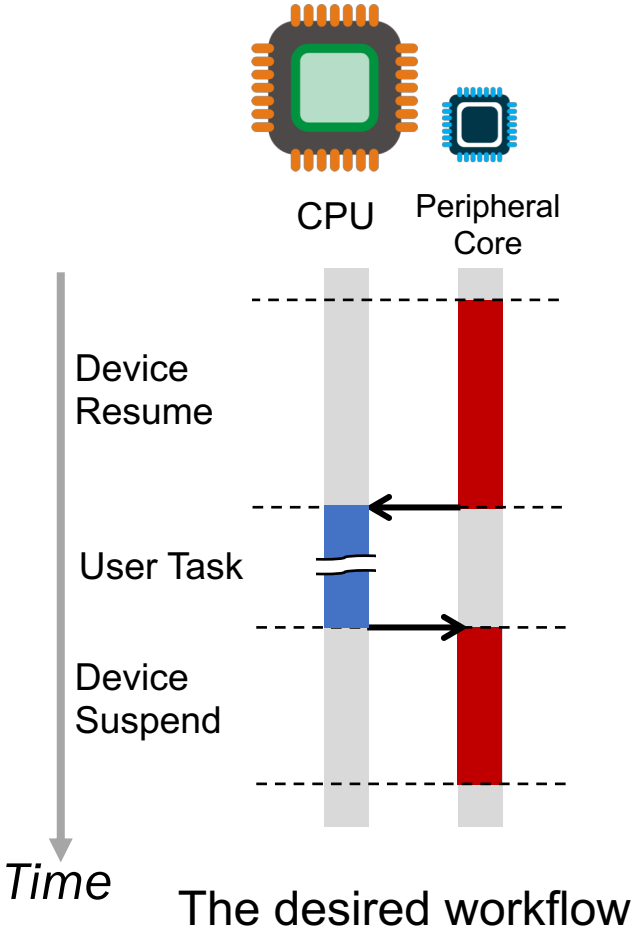
i.MX 7
Cortex M4 + A7
2017



Azure Sphere
Cortex M4 + A7
2019

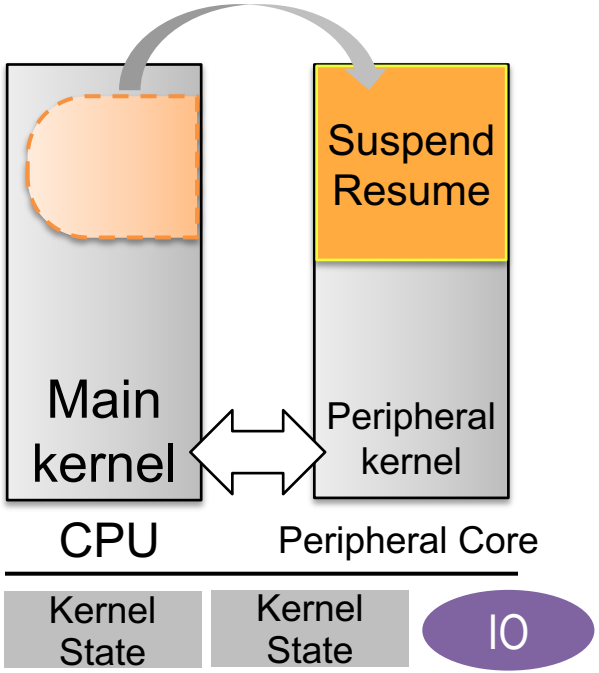
Problem statement

- ➔ On a heterogeneous SoC
- ➔ Given a commodity monolithic kernel (e.g., Linux)
- ➔ How to offload device suspend/resume kernel phase to the peripheral core?



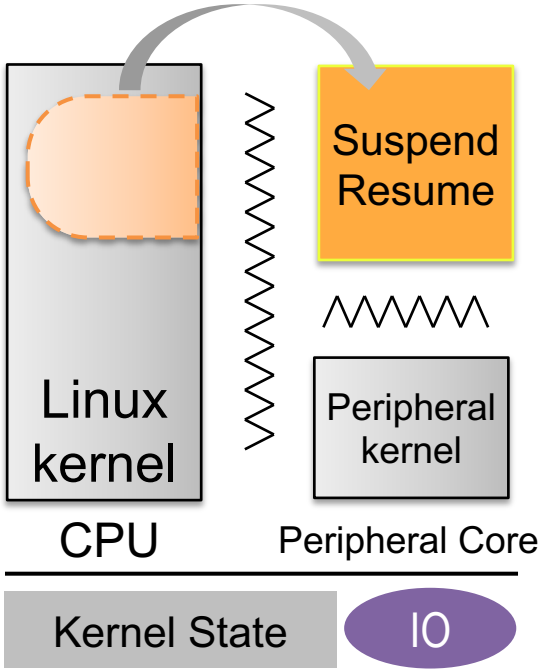
The desired workflow

Design space exploration: multikernel

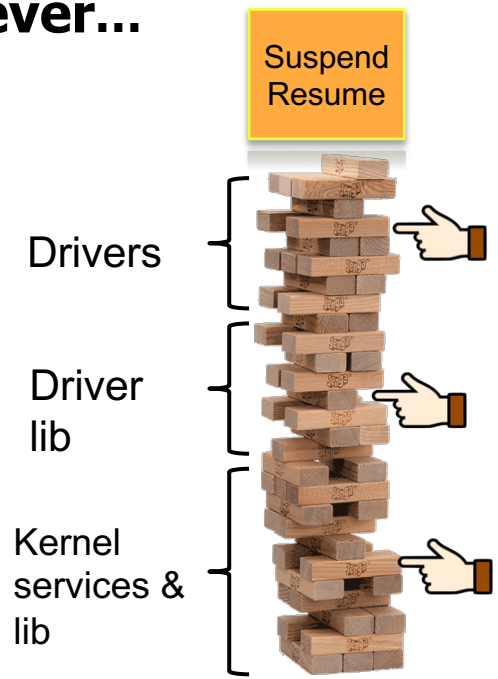


However…

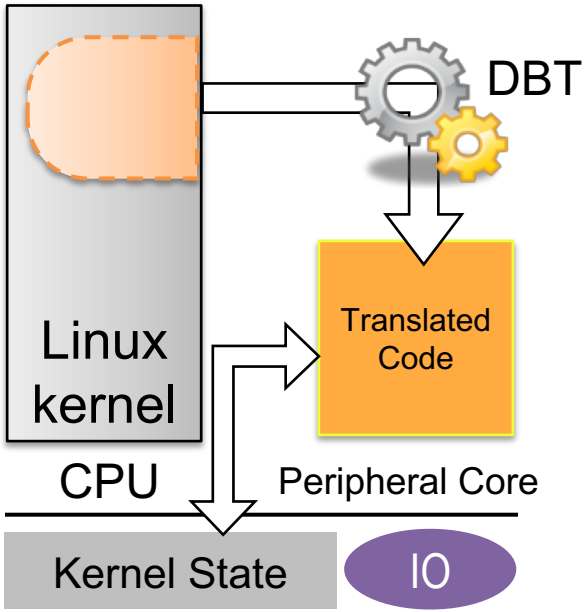
Design space exploration: code transplant



However…



Design space exploration: full virtual execution

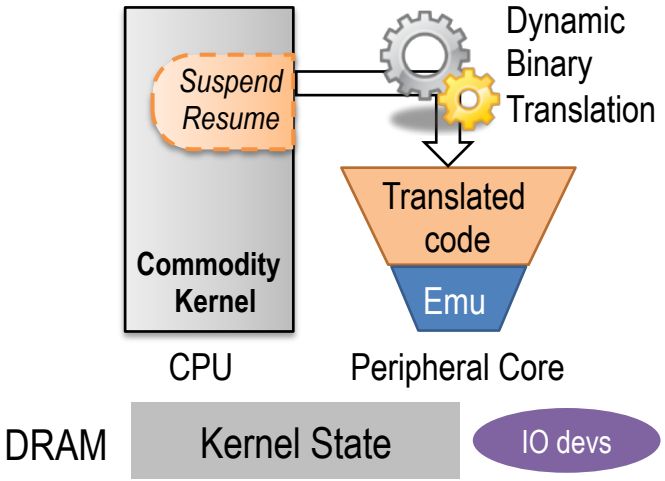


However…

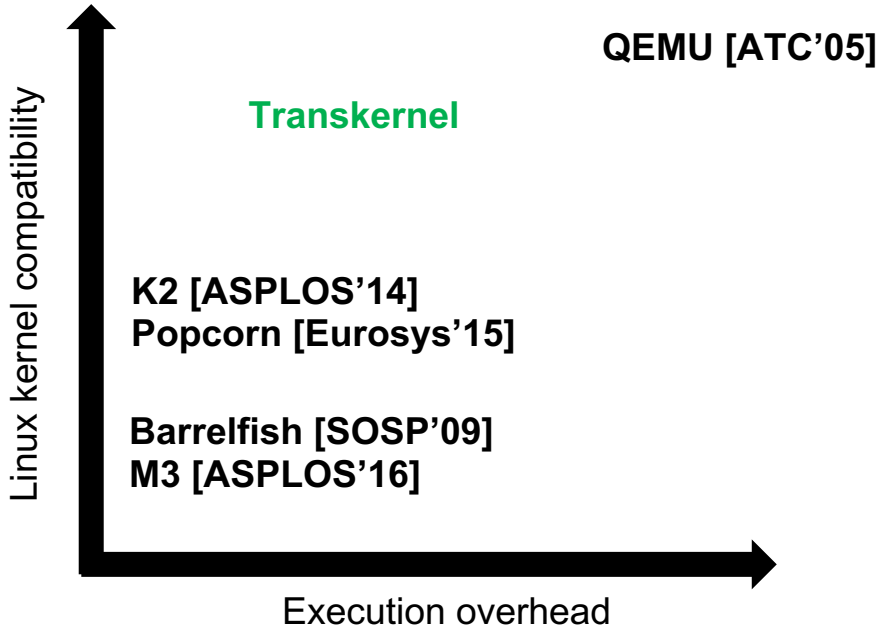
> 25x overhead with current DBT

Our proposal: Transkernel

- Goal: Linux kernel offloading with affordable overhead
- Approach: the peripheral core **dynamically translates the kernel binary**, supported by a small set of **emulated kernel services**

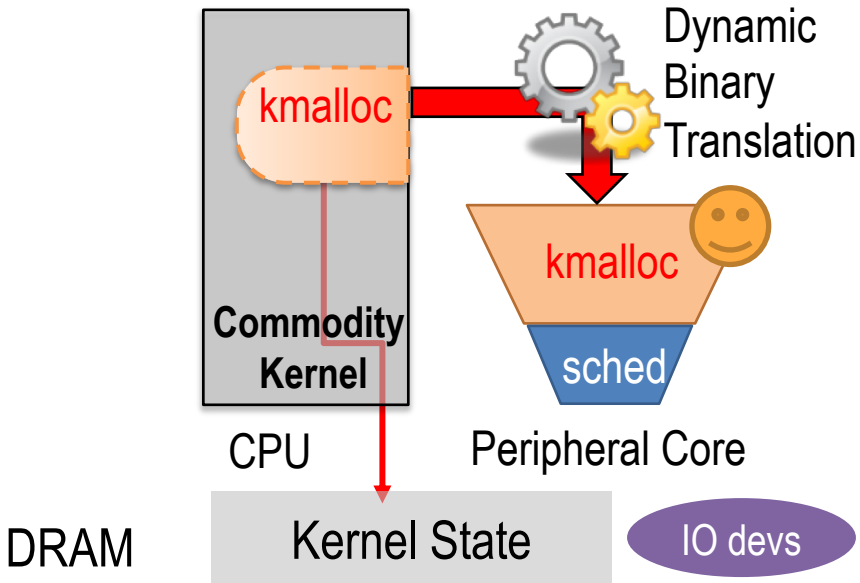


Transkernel in the design space



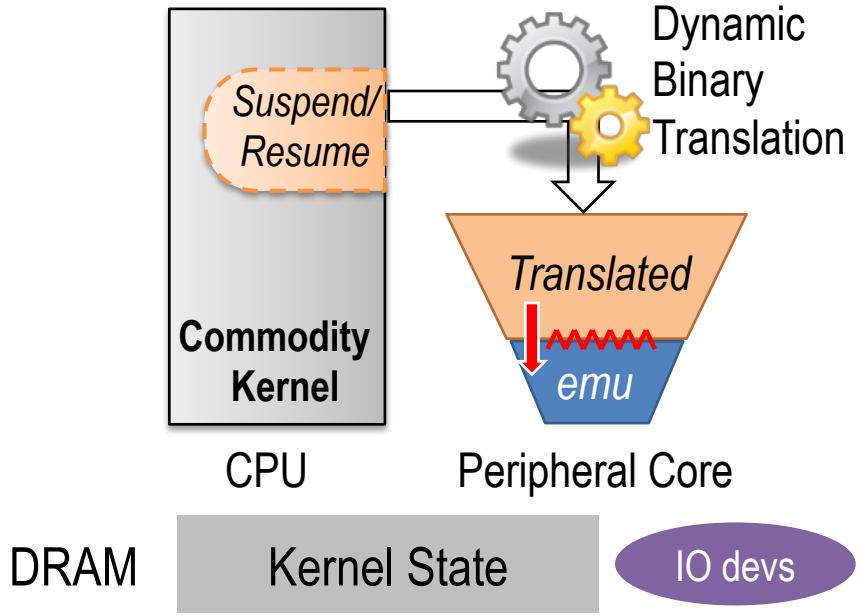
Principle 1: translate stateful code, emulate stateless

- Stateful vs. stateless: whether the states of the kernel are shared across cores
- Translated code: state-sharing made easy
- Emulated services: drop-in replacement



Principle 2: identify narrow trans/emulation interface

- The interface has to be:
  - Narrow
  - Stable
- Maintenance of emulated services made easy



Principle 3: specialize for hot paths

- Hot paths: 99% of executions
  - Encounter no errors
  - All needed resources acquired
- Going off? Fall back to CPU
- Simplify DBT implementation on a peripheral core

Principle 4: exploit ISA similarity

- Between the ISAs of CPU & peripheral core:
  - General purpose registers
  - Control flow registers (SP, LR, PC)
  - Flag semantics (NZCV)

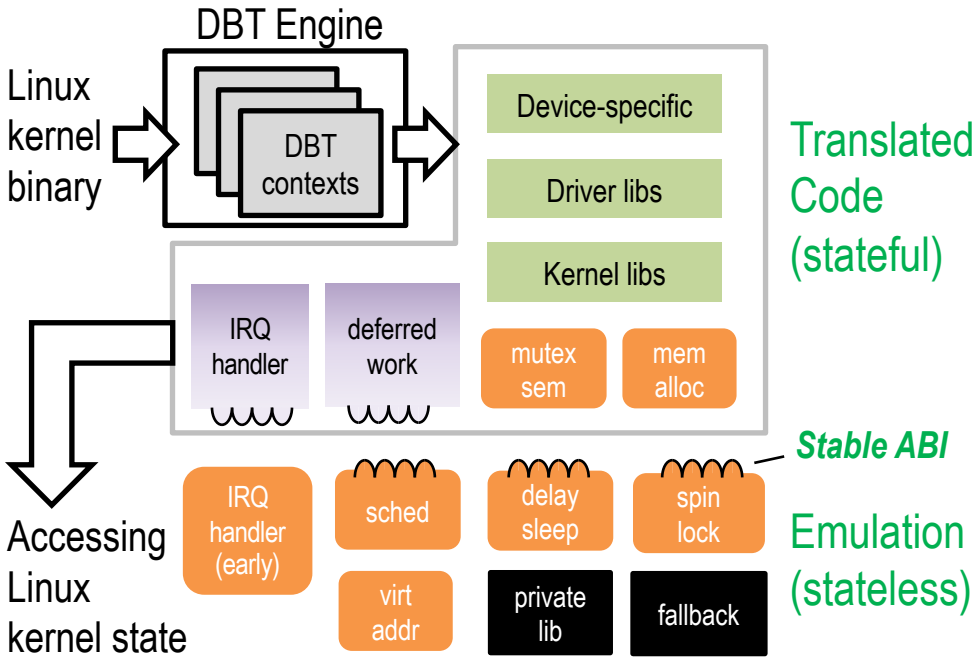
- Reducing the number of emitted instructions in DBT
- Key to low overhead!

ARK: an **AR**m trans**K**ernel

Platform: OMAP4460 (Cortex A9+M3)

ARK instantiates the principles on Linux

- Execute unmodified Linux kernel binary on the peripheral core
- Depend on stable ABIs (only 12 functions + 1 variable)
- Focus on hot paths; may fall back to CPU
- Low-overhead ARM v7a -> v7m DBT



ARK: the cross-ISA DBT engine

- Systemize similar semantics of ARM v7m & v7a from formal specification [1]
- Most instructions have identical semantics (447)
- Others instructions …
  - Side effect
  - Constant constraints
  - Shift modes
- Our DBT engine correctly executes over 200 million instructions!

	v7a insn count	Each translated to # of v7m insns
Identity	447	1
Side effect	52	3-5
Const constraints	22	2-5
Shift modes	10	2
No counterparts	27	2-5
Total (v7a)	558	

[1] Trustworthy Specifications of ARM v8-A and v8-M System Level Architecture, Reid et al., *FMCAD'16*

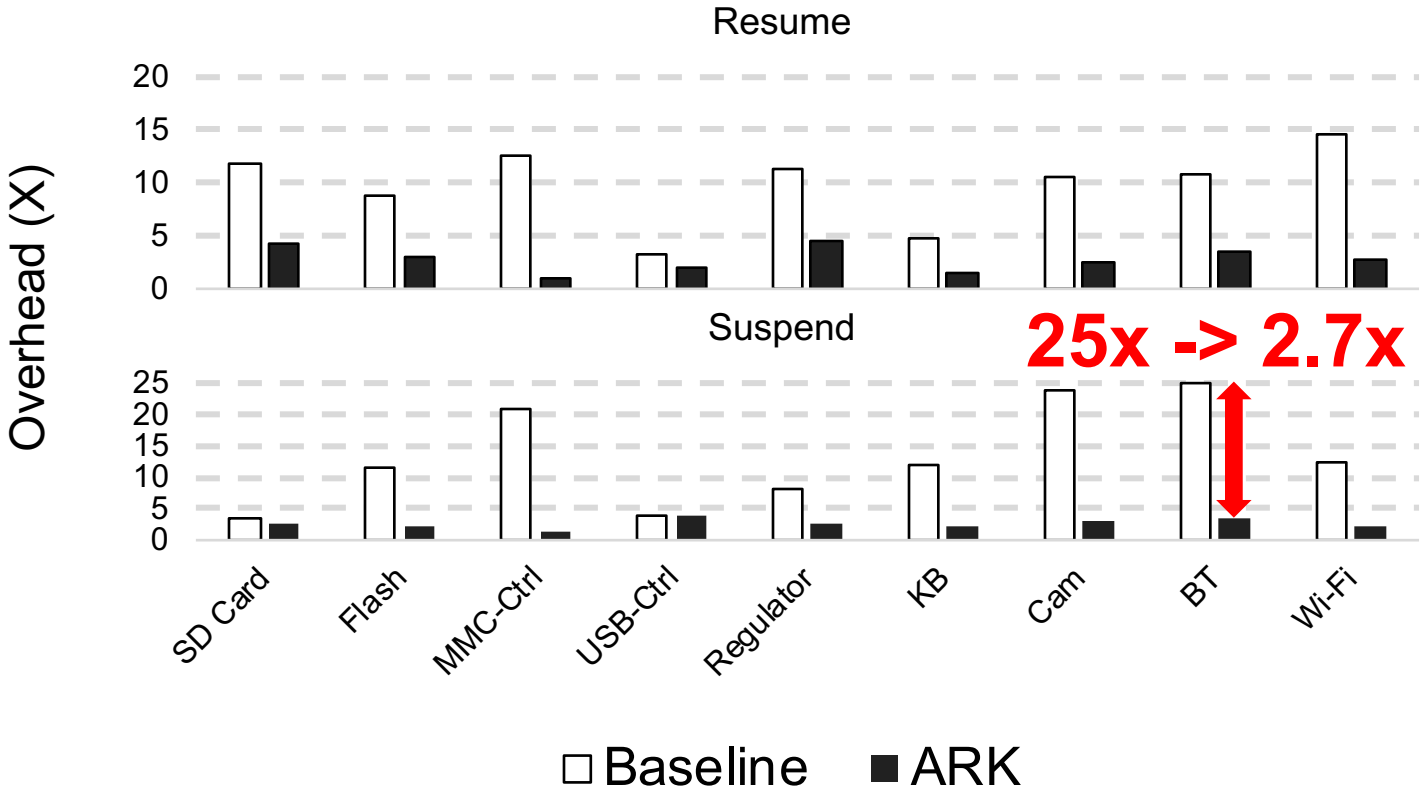
Evaluation

- Does ARK …
  a. Incur low-overhead?
  b. Incur tractable engineering efforts?
  c. Yield efficiency benefit?

- Benchmarks setup
  - Test the whole suspend/resume phase, driven by a userspace test harness

  - Diverse drivers: SD card, Flash drive, MMC controller, USB controller, Regulator, Keyboard, Camera, Bluetooth NIC, Wi-Fi NIC

ARK's DBT achieves low execution overhead



ARK reuses Linux with low efforts

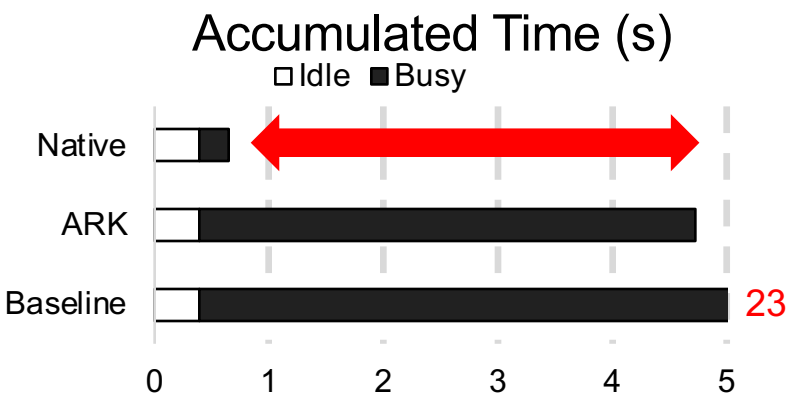
- Good code reuse: 10K vs. 40K and even more
- Good compatibility: multiple versions and configurations of Linux kernel

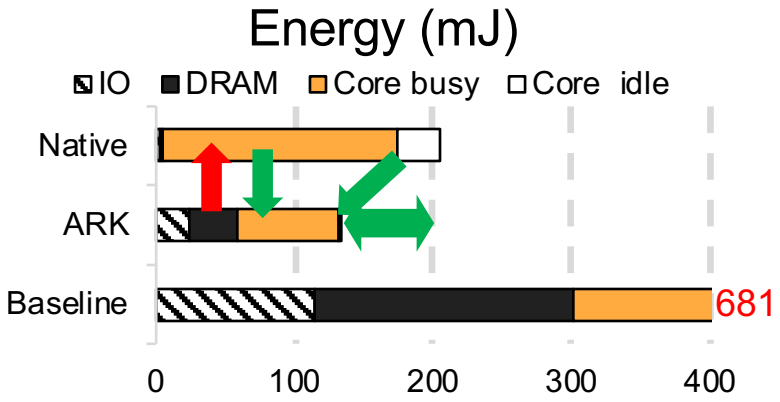
New implementation	
DBT	9K SLoC
Emulation	1K SLoC

Existing code (unchanged)	
Translated	15K SLoC
Substituted w/ emu	25K SLoC

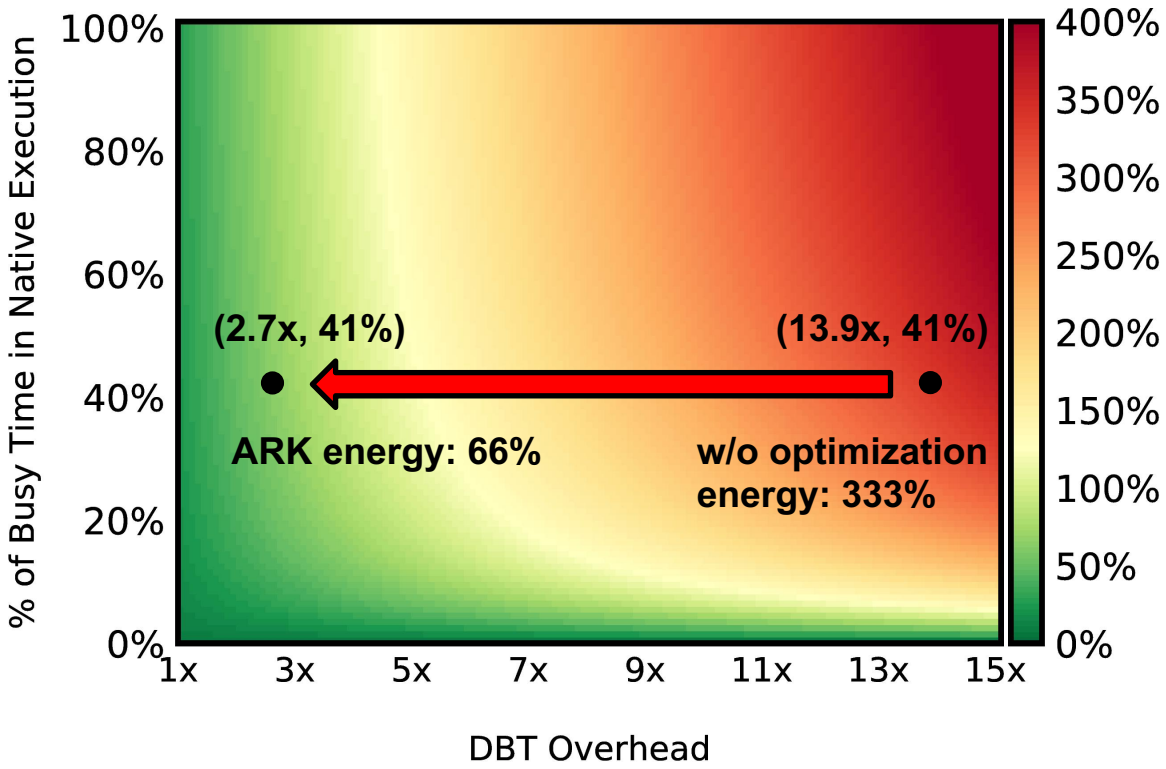
End-to-end execution time & energy

- Time: prolonged execution time
- Energy: 34% energy saved
- Interesting finding: ARK sees higher DRAM energy





What-if analysis



Take-home messages

- Transkernel & its key techniques
  - An appropriate translation/emulation boundary inside a monolithic kernel
  - Exploit ISA similarity
- To OS
  - A new model to span a monolithic kernel over heterogeneous cores
- To DBT
  - Efficiency loss can enable efficiency gain
  - DBT applies to translate a specific path of a complex software stack!
- To Architects
  - A heterogenous SoC friendly to transkernel