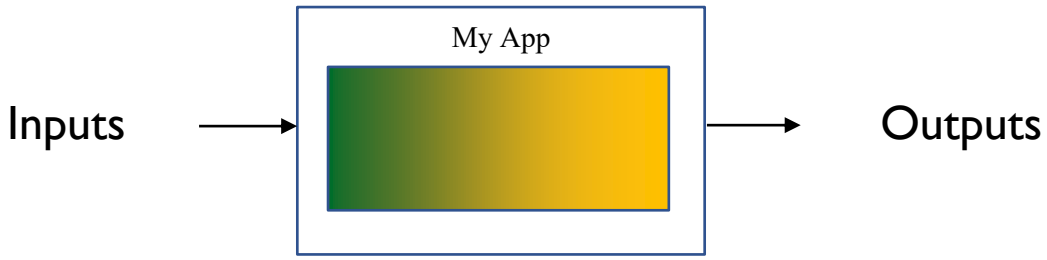
Hardware for Event-Driven Architectures

David Schall,

Andreas Sandberg, Boris Grot

Event-Driven Architectures

Traditional

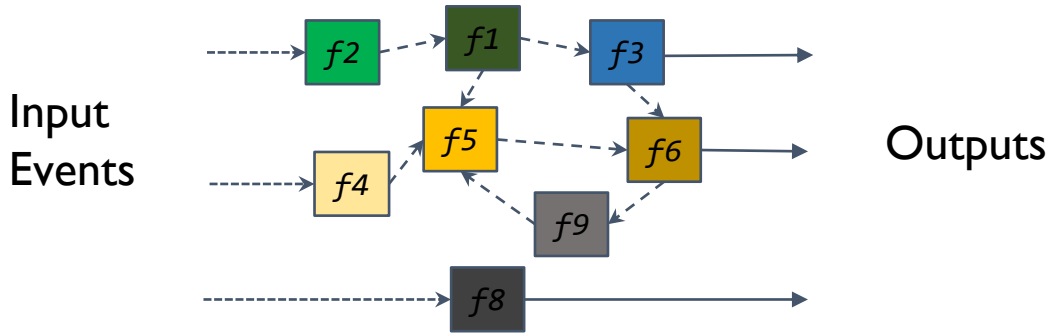


Traditional Architecture:

- Implements functionality as a sequence of instructions and control commands
- Application prompts for input
- Program defines control flow

Good for small applications

Event-Driven



Event-driven Architecture:

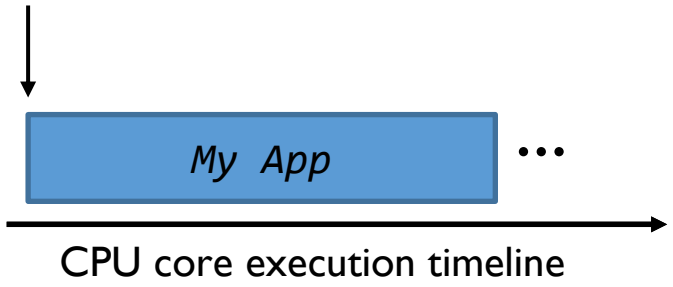
- Breaks functionality into many functional independent event handlers
- Event handlers react and communicate via events
- Events determine control flow

Decoupling improves flexibility and enables large applications

Event-Driven architectures are booming!

Event-Driven Application on a CPU

Traditional Workload



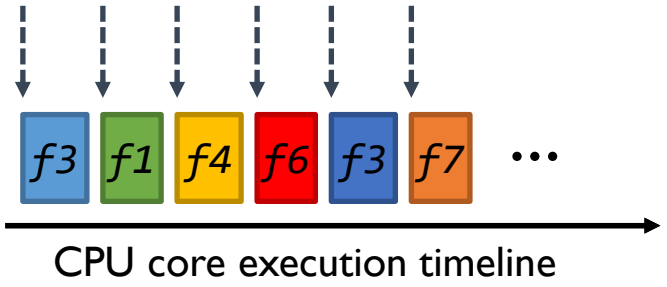
Traditional Applications
- Long and constantly running
- Large piece of code with fixed control flow

CPUs are designed for traditional applications ✓

Microarchitecture warms up
- Caching can exploit temporal, spatial locality
- Prefetcher, branch predictor (BPU) are able to learn control flow

→ CPU performance is high ☺

Event-Driven Workload



Event Handlers
- Short execution times, run only on-demand
- Tiny code fractions in arbitrary order

CPUs are **not** designed for event-driven applications ✗

Microarchitecture remains cold
- Caching has no benefit: Too short, too infrequent execution
- Prefetchers and BPU cannot learn from the random order of invocations

→ CPU performance is low ☹

CPUs are not designed for event-driven applications!

Can We Do Something?

We study serverless functions as a use case of event-driven architectures for better understanding.

Interested in what we found?

Hardware for Event-Driven Architectures

David Schall[1], Andreas Sandberg[2], Boris Grot[1]
[1]University of Edinburgh, [2]Arm Research

Event-Driven Architectures

What is Event-Driven?

Sequential

Event-Driven

Why Event-Driven?

Disadvantages of Sequential Architectures:

- Gets quickly complex for large applications (Spaghetti code)