Background

Target Challenge: Data Supply is the Fundamental Bottleneck in Accelerator-Rich Computing Systems
- Hardware accelerators make data supply bottlenecks dominate runtime
- Key bottlenecks lie in supplying specialized accelerators with data
- Different accelerators and applications have different data supply needs
- Accelerators lack general-purpose latency-tolerance mechanisms
- Accelerator-rich computing requires big increases in memory bandwidth
- Targets machine learning and complex graph applications

Approach

DECADES is a Vertically-Integrated Software/Hardware approach that combines Language and Compiler support to map complex graph and Machine Learning applications to a novel, heterogeneous, accelerator-rich manycore architectures.

DECADES Key Innovations:
- Intelligent Storage tiles orchestrate on-chip data movement between accelerators and accelerators, accelerators and core, and core to core
- Best-of-breed pluggable accelerator socket and High-Level Synthesis flow ease accelerator integration (ESP and ESP4ML)
- Rich compiler (DEC++) and language infrastructure automatically slices applications and maps graph applications onto accelerators and cores
- DECADES architecture contains both near memory and in-memory computation to reduce energy of data movement (ComputeDRAM)
- Strong commitment to open source release of software and hardware

ESP4ML Innovations
1) Accelerator Chaining
   - Avoid memory roundtrips
   - Fine-grained accelerators
   - synchronization

```
ESP accelerators replace software kernels 2 and 4
| int *buffer = esp_alloc(size); for (...) {
    kernel 1(buffer, ...);
    cfg_k2 has the accelerator configuration parameters
    esp_run(buffer, cfg_k2);
    esp_run(buffer, cfg_k4);
    esp_free(buffer); }
```

2) Accelerator invocation API
   A 3-functions API for accelerator invocation from user applications
   - Automatically generated
   - Linux device drivers
   - No data copies needed at accelerator-invocation time

Results and Impact

GraphAttack!: HW/SW co-design to hide long latencies of indirect Neighbor Memory Accesses (NMAs) that bottleneck graph applications
- DEC++ Producer/Consumer program slicing where Producer issues NMAs and Consumer performs computation with their data
- Intelligent Storage Tile asynchronously performs NMAs
- Producer issues memory request; data provided to Consumer

ESP4ML Case Study
Two multi-accelerator SoC prototypes on FPGA with multiple accelerators
- Night-vision
- Image classifier
- Denoiser
- Autoencoder

Energy efficiency: 100x gain vs. Jetson and i7
Performance: 4.5x gain with p2p & parallelization
Memory accesses: 3x decrease due to p2p