LegUp High-Level Synthesis and its Commercialization

Jason Anderson

Workshop on Open-Source Design Automation (OSDA)

March 29, 2019

https://janders.eecg.utoronto.ca

http://legupcomputing.com

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

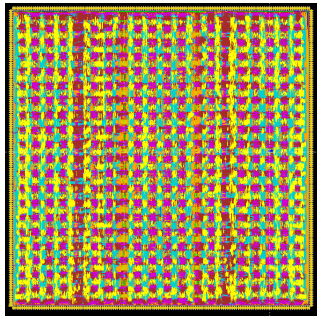
Specifying Computations

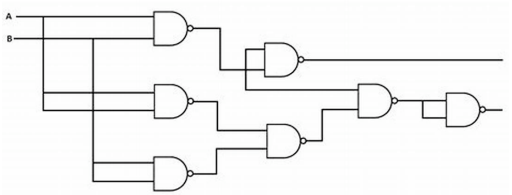
Write Software for a Processor

- Easy (comparatively speaking)
- Flexibility → lower performance

Design Custom Hardware

- High performance, low power
- Need specialized knowledge





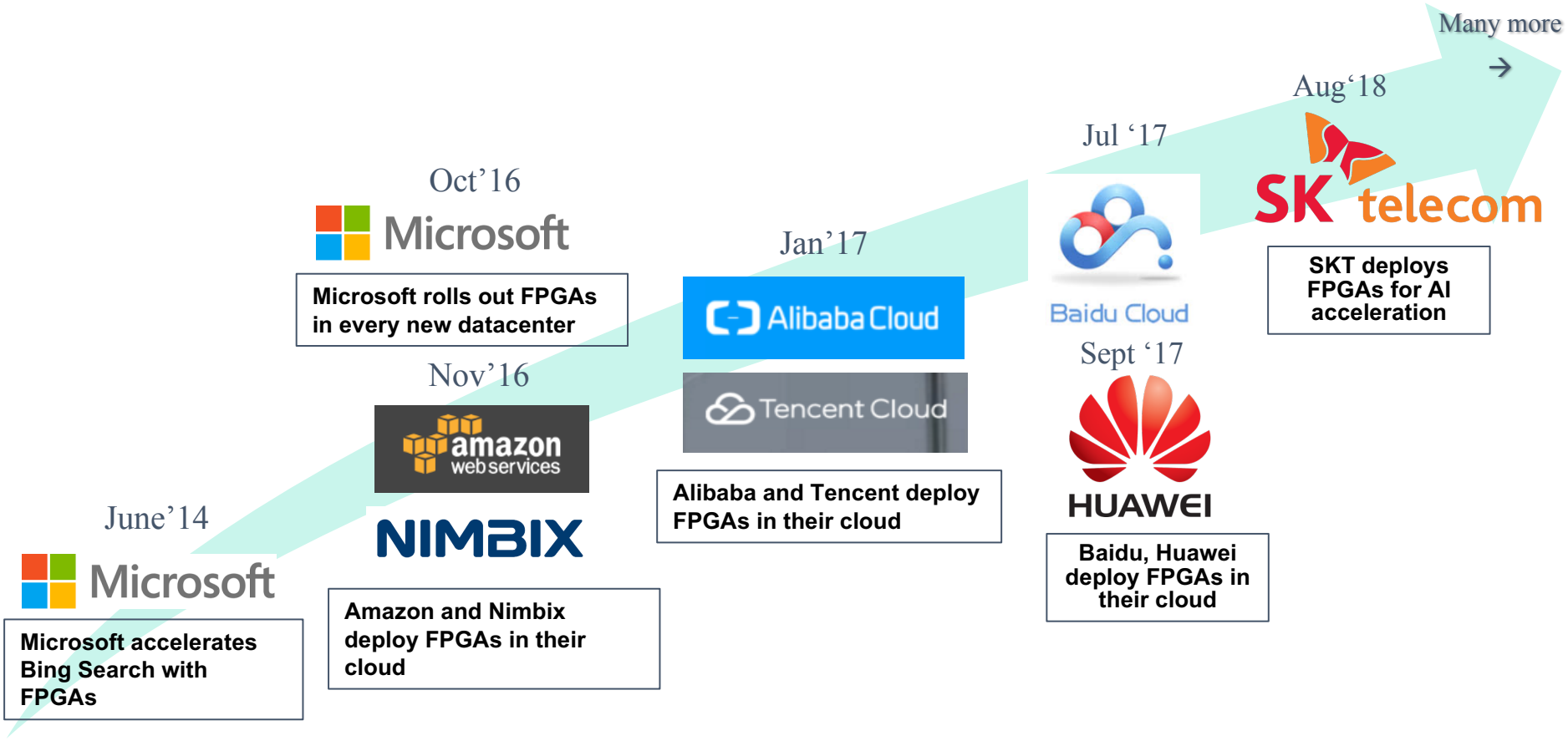
The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

FPGA-Based Acceleration

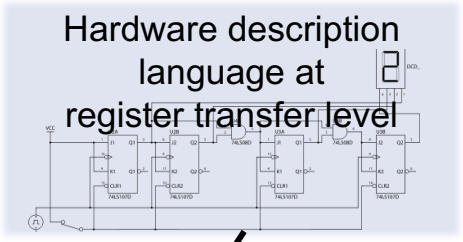
- Implementing computations in hardware can have speed/energy advantages over software:
  - Biophotonic simulations: **4X** speed-up, **67X** more energy efficient [Cassidy, Betz, FCCM'14]
  - Options pricing: **4.6X** faster, **25X** more energy efficient [Tse, Thomas, Luk, TVLSI'12]
  - Deep learning accelerator on Arria 10: **1.4** TOPS, 1020 img/s for ImageNet inference [Aydonat et al., FPGA'17]
  - Microsoft Bing search: **2X** speed-up, **29%** latency reduction [Putnam et al., ISCA'14]

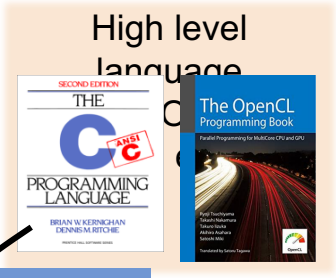
The Era of FPGA Cloud Computing is Here

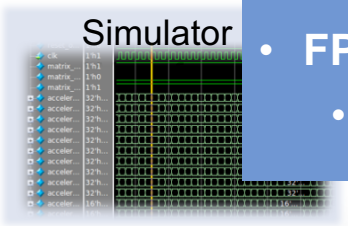
Rapidly emerging FPGA-as-a-Service landscape



Problem: FPGAs Are Difficult to Use







- **FPGA design is difficult even for hardware engineers**
  - **Software engineers simply cannot use FPGAs**

FPGAs

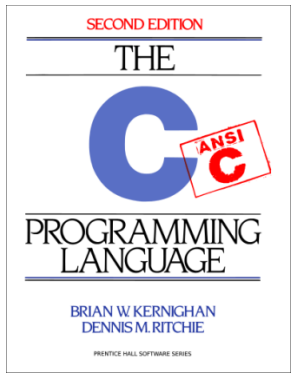
- Requires specialized knowledge to design hardware
- Design time: months ~ year

CPUs / GPUs

- Software is relatively easy
- Design time: weeks ~ months
- 10 software engineers for every hardware engineer

The Edward S. Rogers Sr. Department of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

A Solution



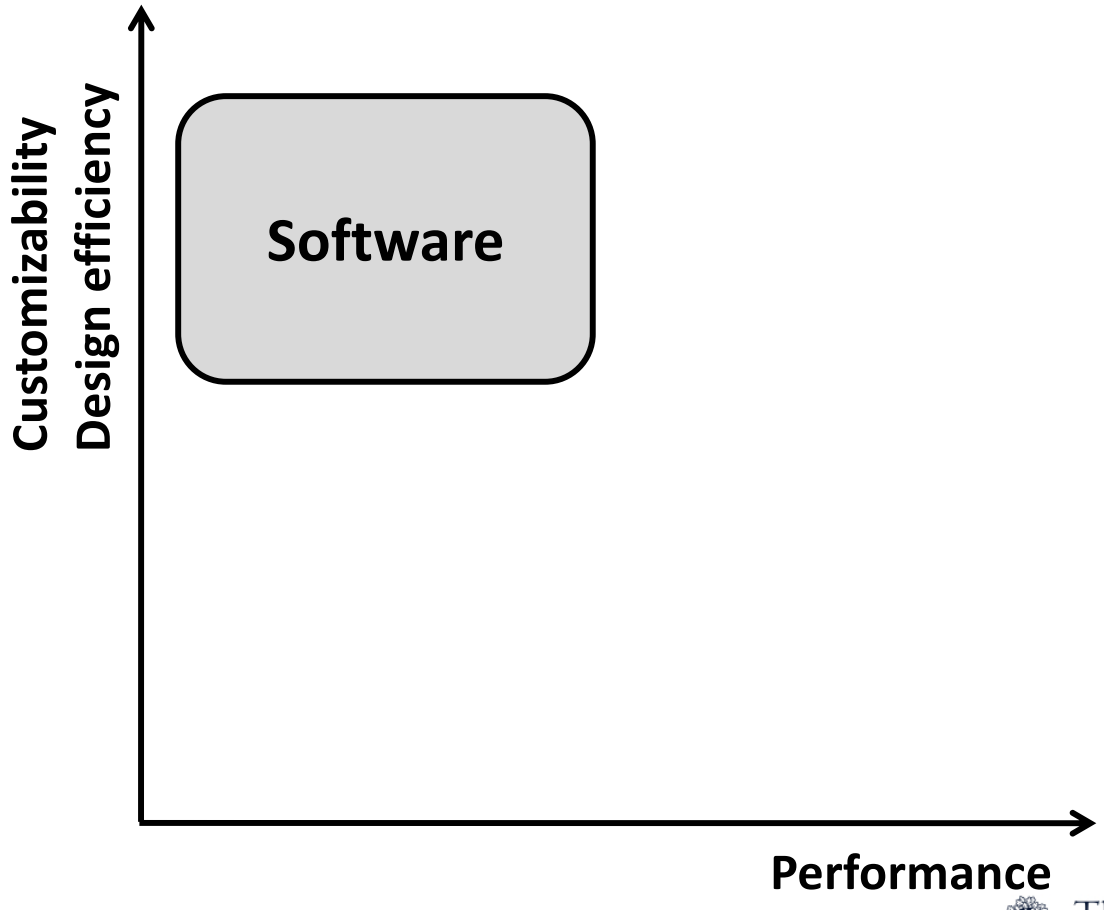
High-level Synthesis

Flexibility/ Ease of Use

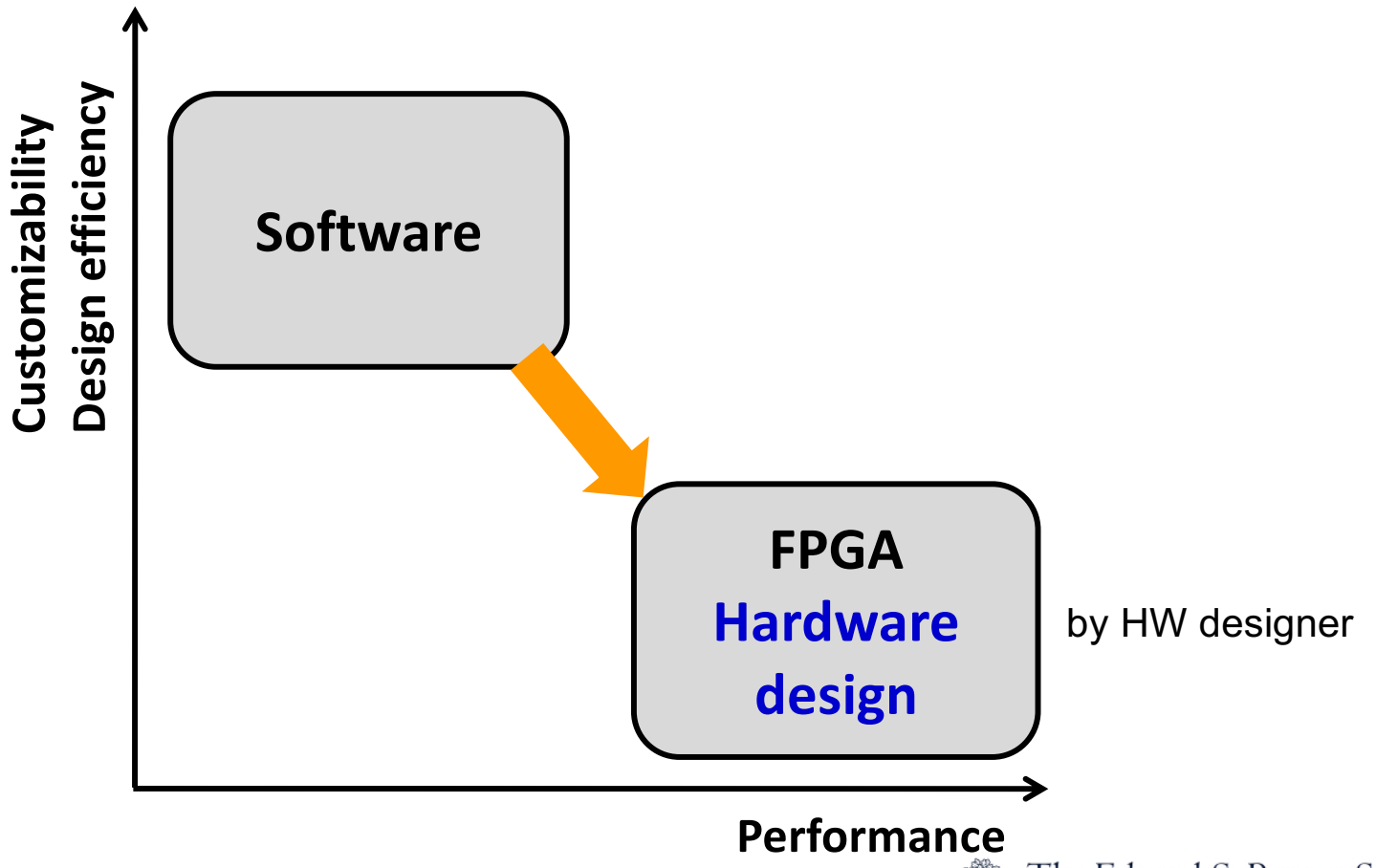
High-performance/ Energy-efficiency

The Edward S. Rogers Sr. Department of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

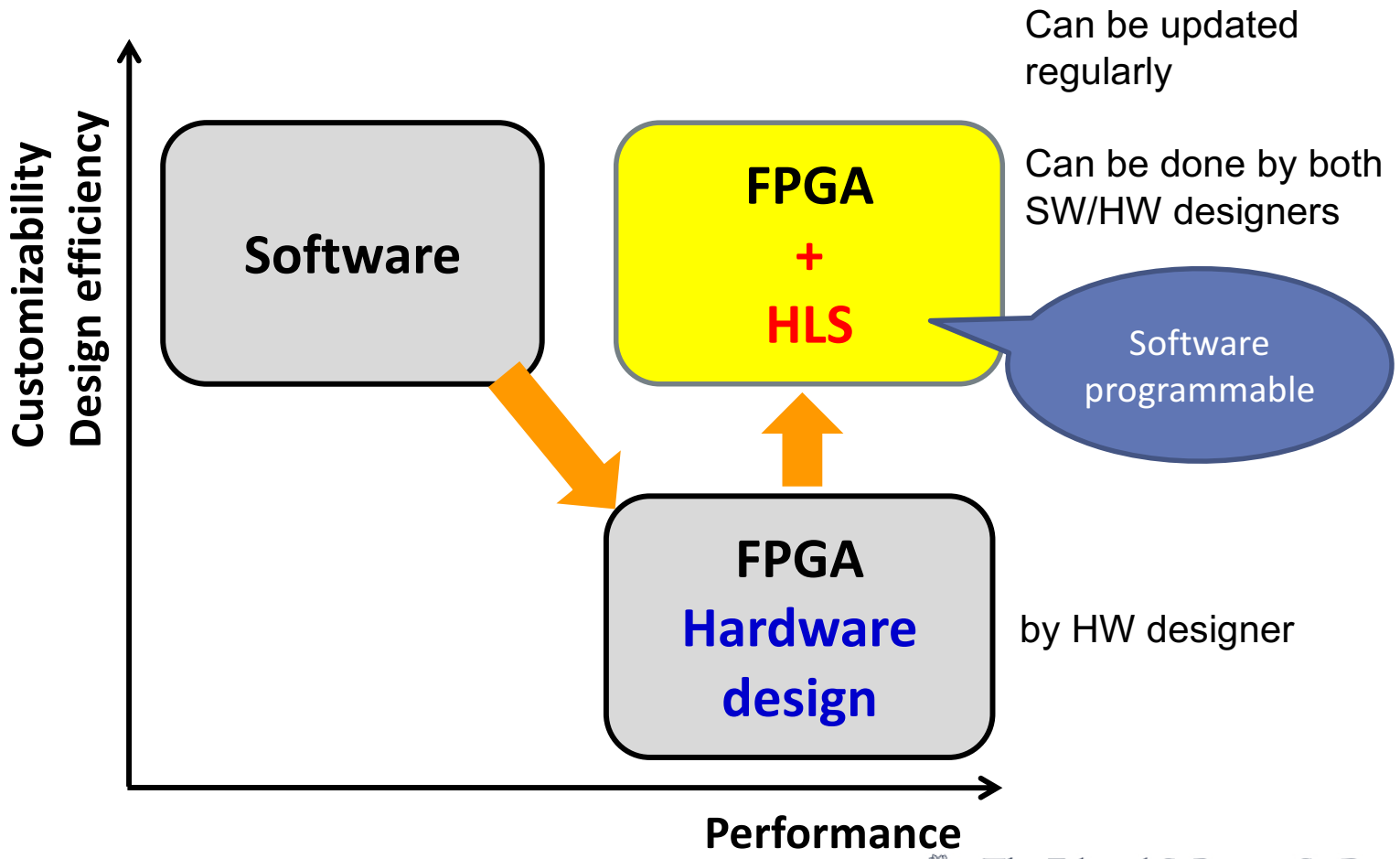
HLS Value Proposition



HLS Value Proposition



HLS Value Proposition



Benefits of HLS

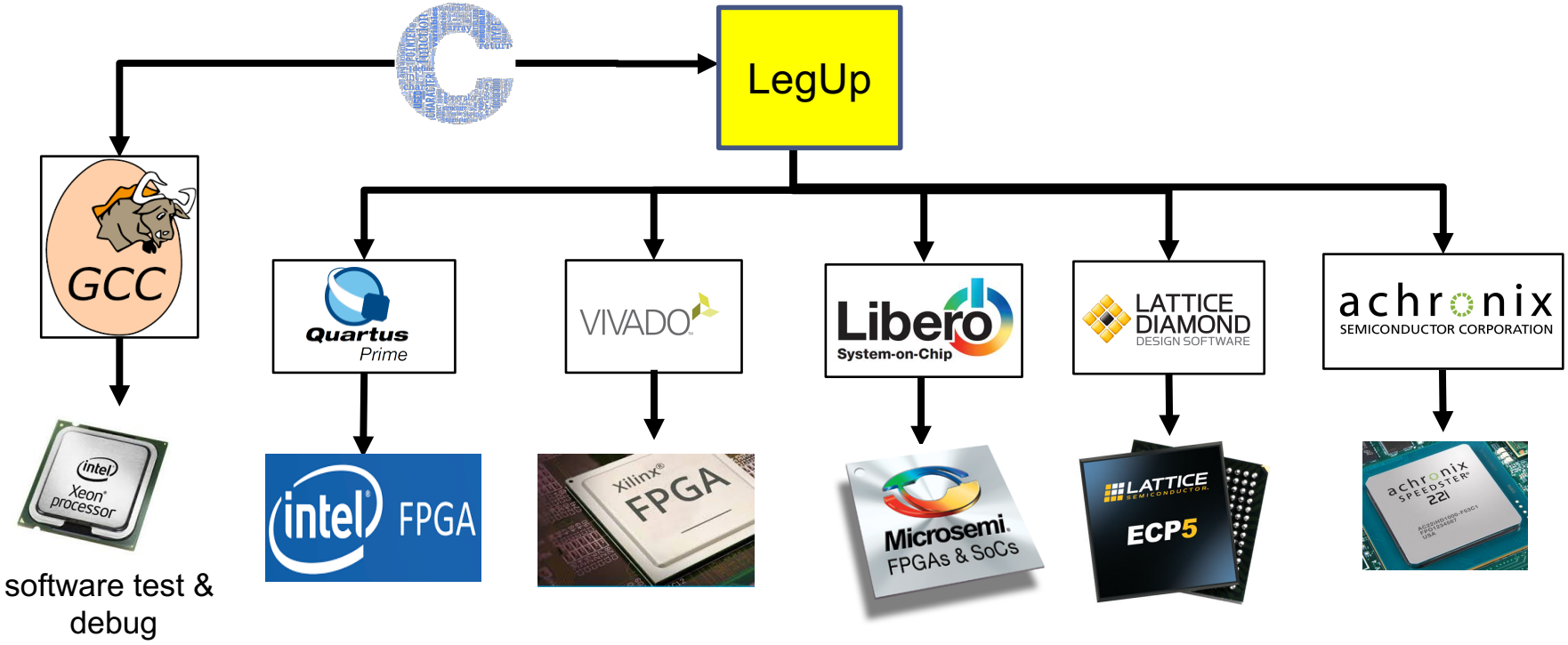
- Time-to-market (lower NRE)
- Easier modifiability/maintainability
  - Design spec is in SW
  - Important for some appls where spec isn't firm or changes frequently, e.g. finance models
- Rapid exploration of HW solution space
- Make FPGA HW accessible to SW engineers
  - Bring the energy and speed benefits of HW to those with SW skills

The Time is Right for HLS

- HLS papers first appeared in the 80's
  - e.g., Yorktown Silicon Compiler (IBM)
- Many "false starts"
  - e.g. Synopsys Behavioral Compiler in 90's
- So… why should it fly now?
  - Hardware size and complexity becoming unmanageable
  - Can't ride wave of processor perf. improvements
    - Must deliver better speed/power through other means
  - Improvements in compiler technology
  - FPGA is the right "IC media" for HLS

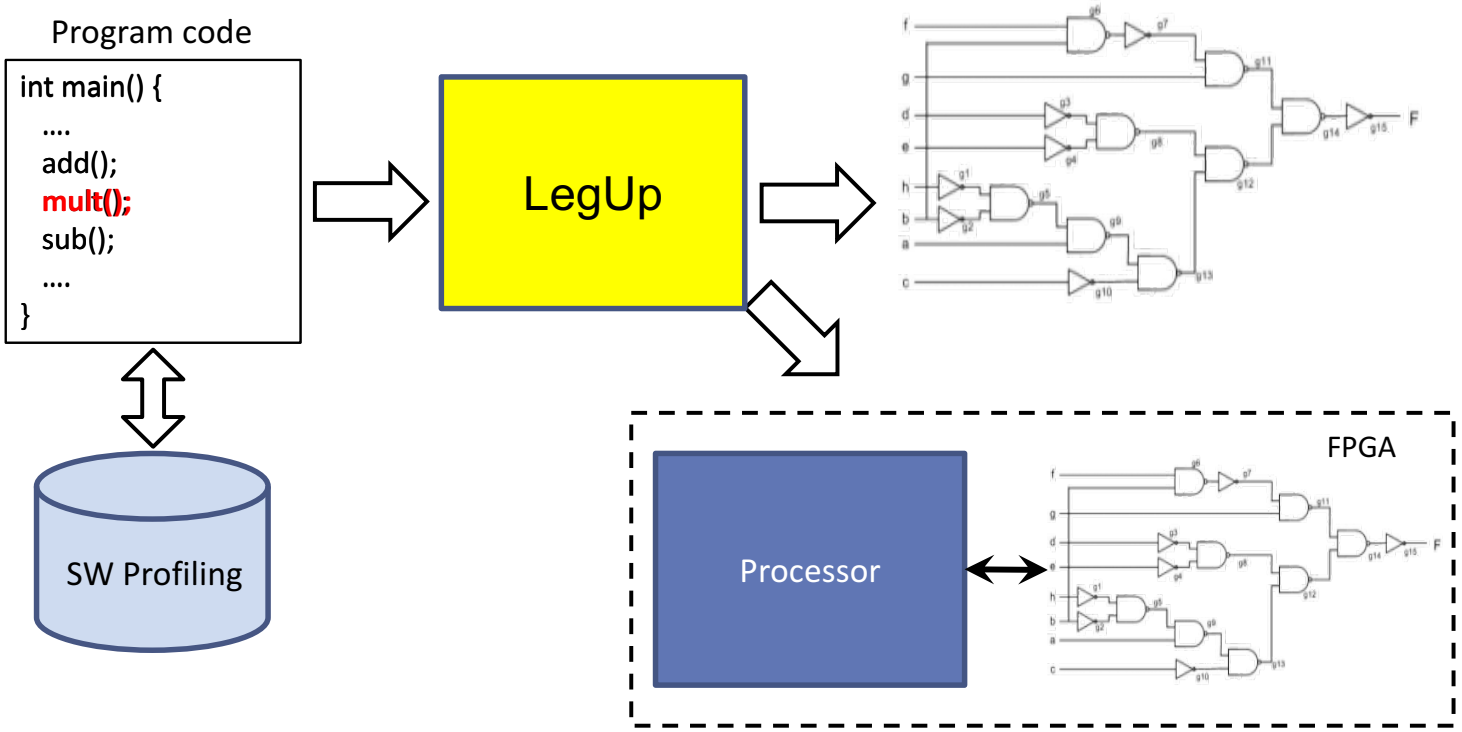
LegUp High-Level Synthesis

- Programming layer that can target any FPGA



The Edward S. Rogers Sr. Department of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

LegUp Overview



LegUp Overview (2)

- Under development since 2009
- 5000+ downloads since first release in 2011
  - Open-source license for non-commercial research purposes
- 20+ conference/journal publications, book chapter, multiple awards; community Award at FPL, BP Award at FPL 2017
- Used LegUp to teach summer courses in HK, Harbin, Europe
- Many grad and undergrad "LegUp alumni"

legup.eecg.toronto.edu

LegUp Overview (3)

- Why?
  - Few open-source HLS projects
  - Addresses key FPGA challenge: too hard to program
    - Xilinx/Altera didn't have HLS
  - Inspired by success of other projects:
    - VPR/VTR: FPGA architecture, packing, placement, routing
    - ABC: logic synthesis
  - Do a "big" project with many students
  - Had industry and government funding for it…

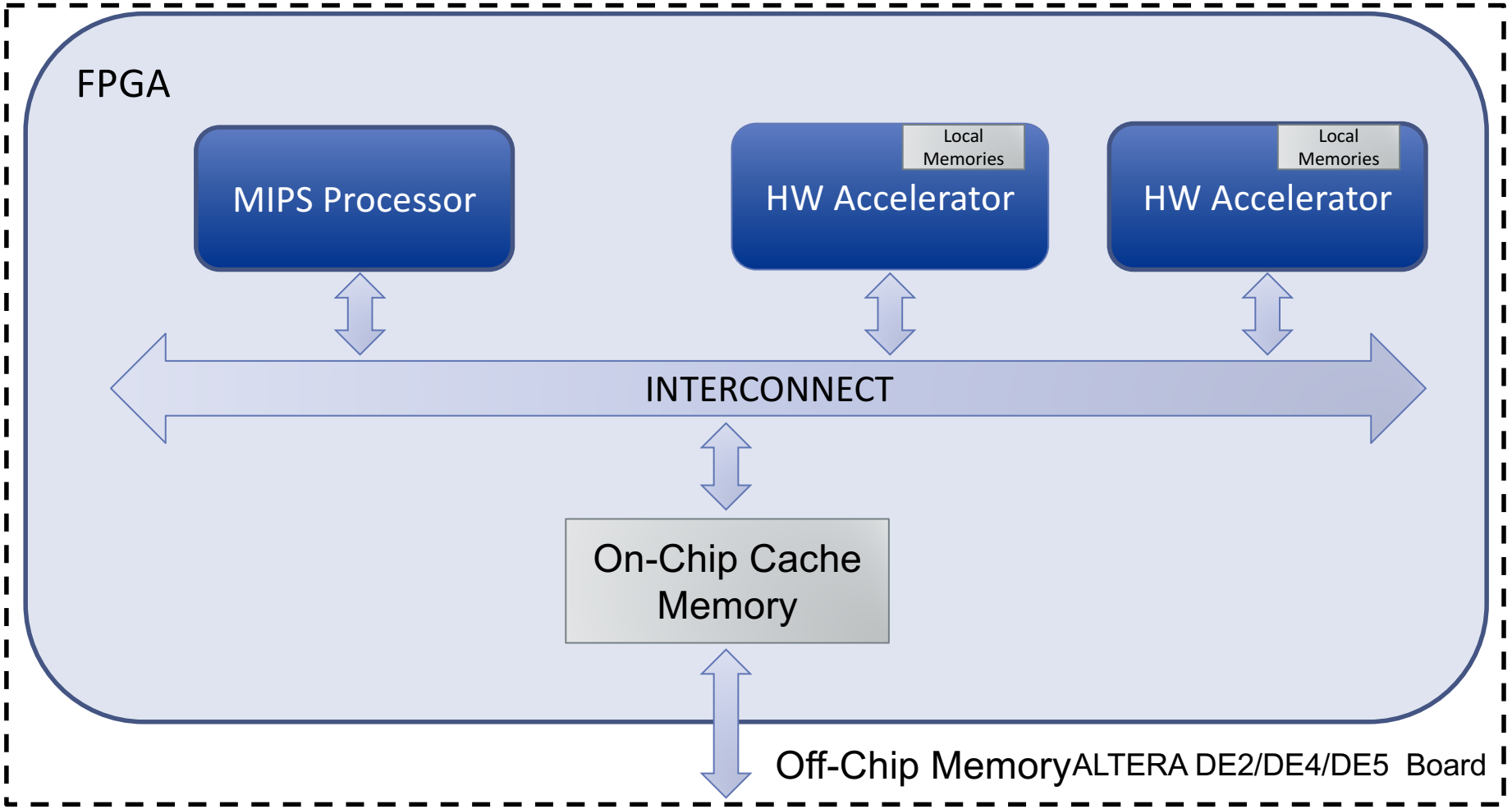
Unique Features and Recent Directions

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

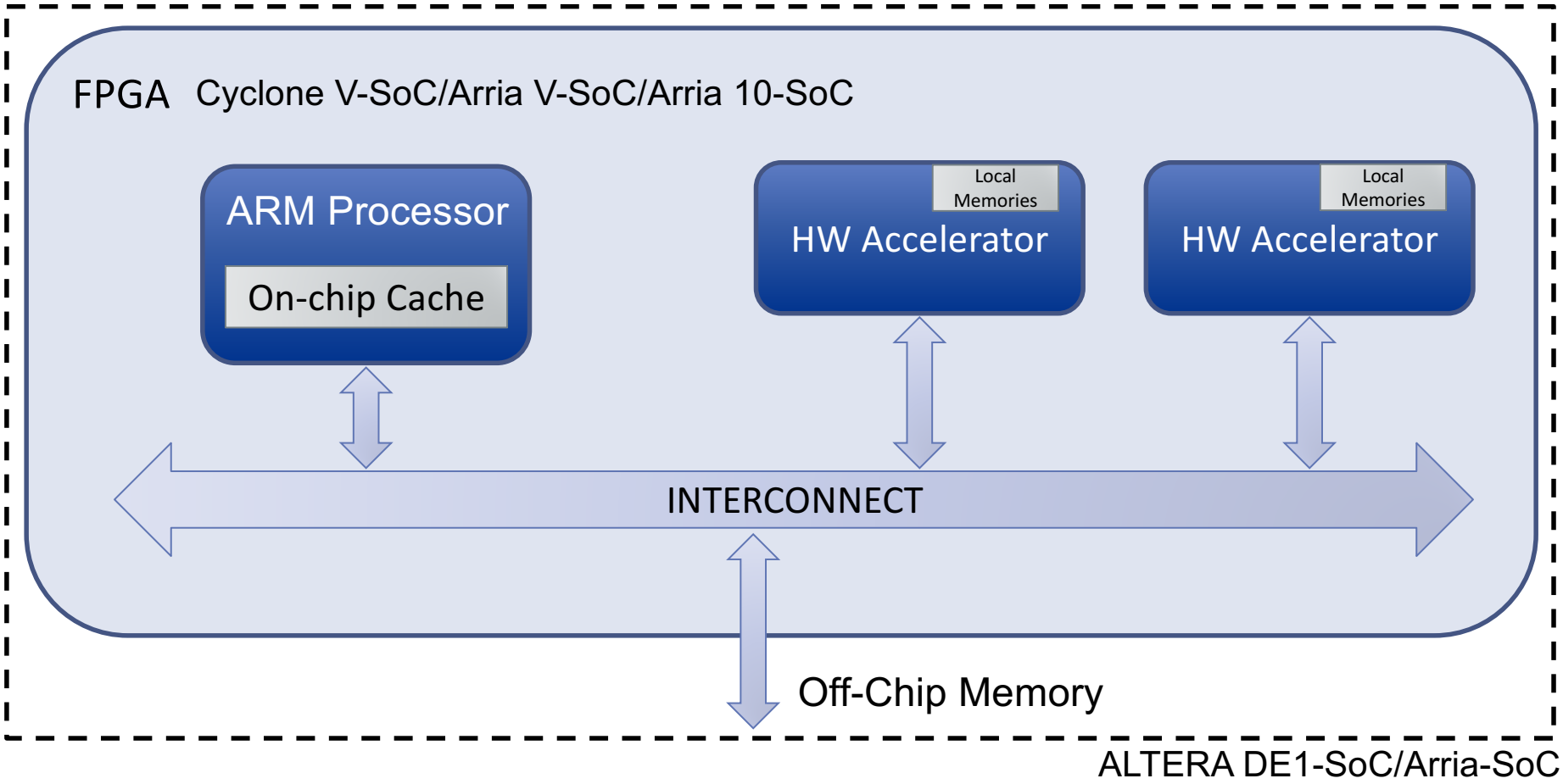
SoC Generation

- With a ***single*** command, LegUp generates a System-on-Chip with embedded processor & hardware accelerators
  1. User designates function(s) for hardware acceleration
  2. LegUp performs software/hardware partitioning
  3. LegUp compiles hardware partition into hardware accelerator
  4. Software partition is compiled for an embedded processor
  5. Complete system is generated with memories and interconnect

System-on-Chip: MIPS Soft Processor

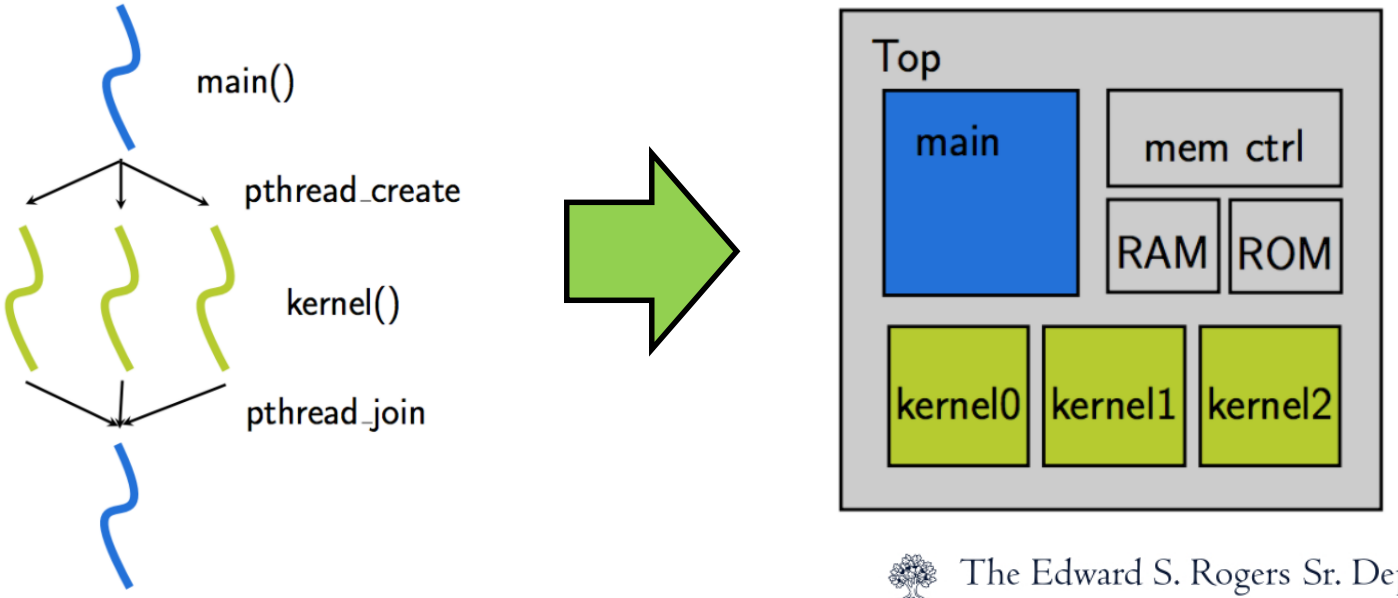


System-on-Chip: ARM Hard Processor



Parallel Software to Parallel Hardware

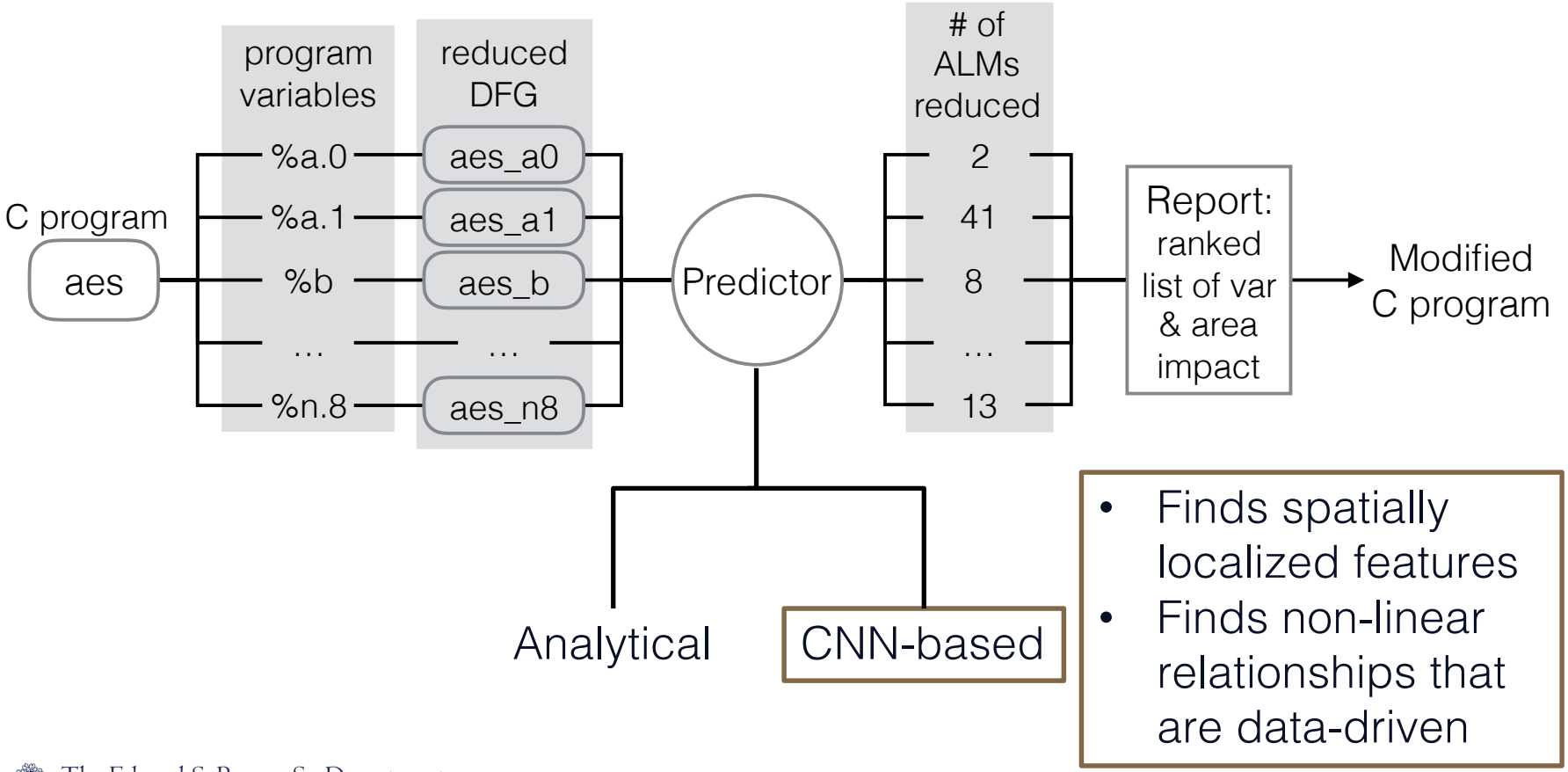
- With hardware, one can exploit spatial parallelism
  - Unfamiliar to software engineers
- LegUp can synthesize *software* parallelism (Pthreads/OpenMP) into *spatial* hardware parallelism
- Each SW thread synthesized into a HW module



The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

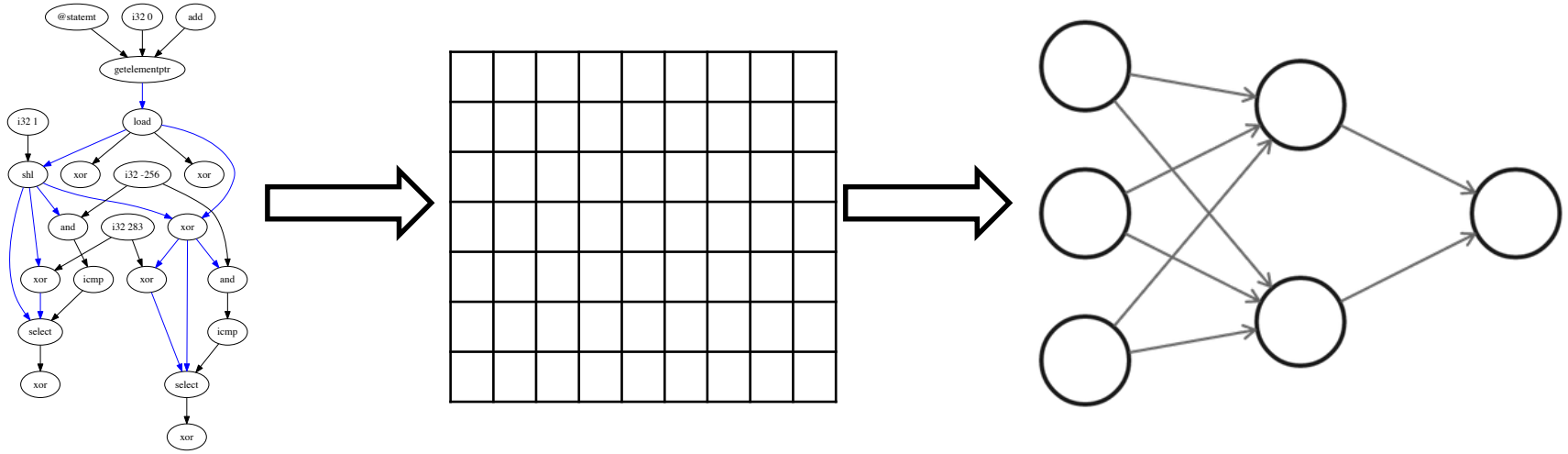
ML-Based Area Reduction Advisor

- Apply ML for prediction and/or decision making in HLS

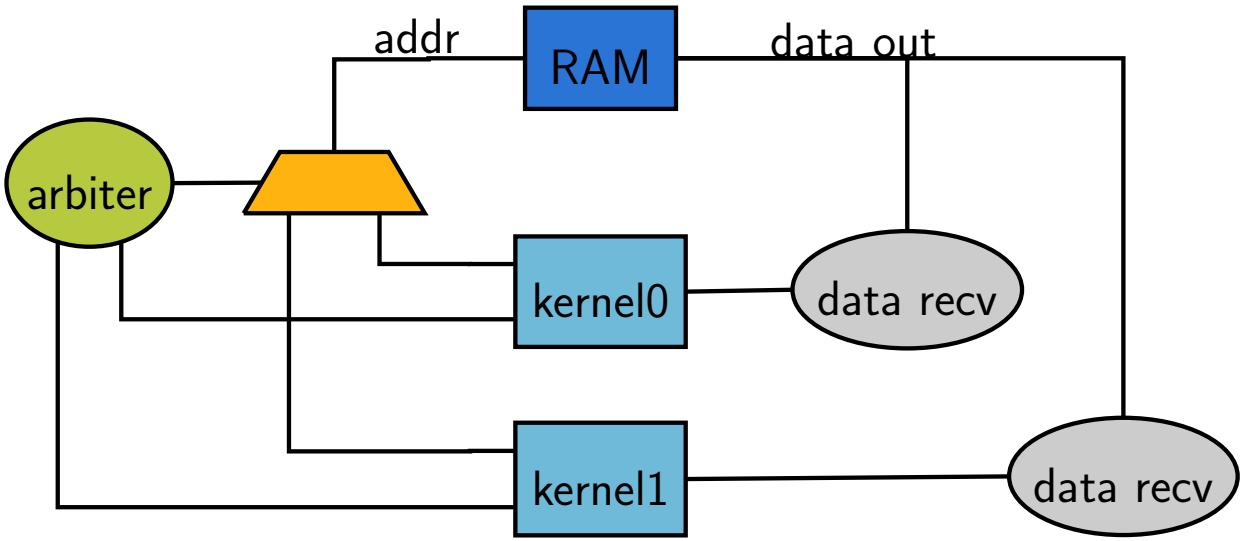


CNN-Based Circuit Area Predictor

Map a program's DFG onto an input image representation for the CNN



Memory Architecture Synthesis



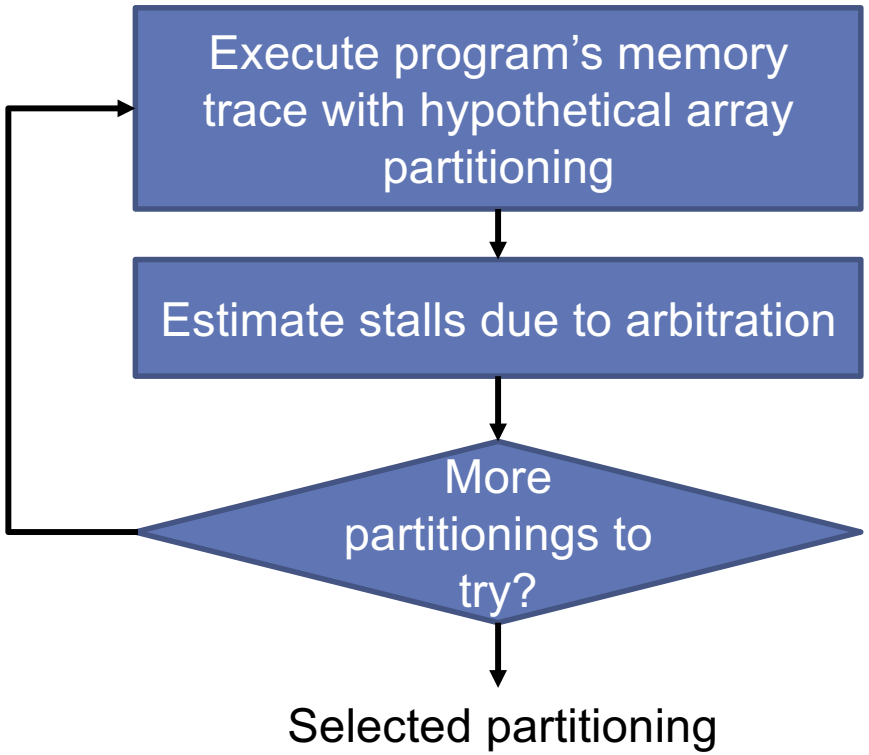
What if kernel0 and kernel1 want to access the RAM in the same cycle?

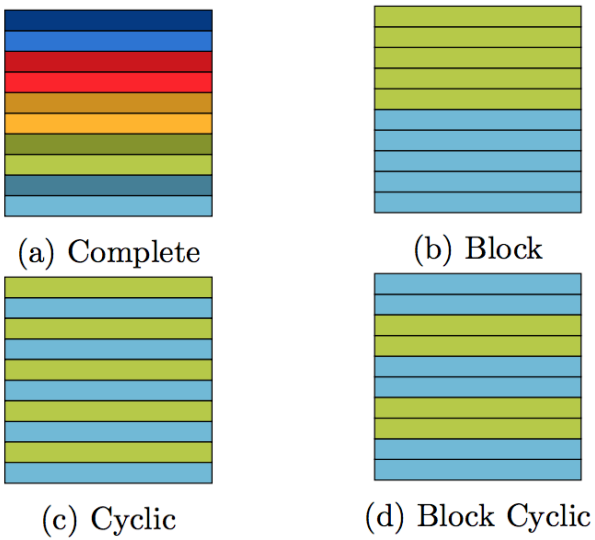
Automatically partition RAM into sub-RAMs
based on kernel access patterns

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Memory Architecture Synthesis (2)

- Profile multi-threaded program behavior
- Partition arrays into sub-arrays (implement in separate RAMs) to provide threads with exclusive access (to extent possible)

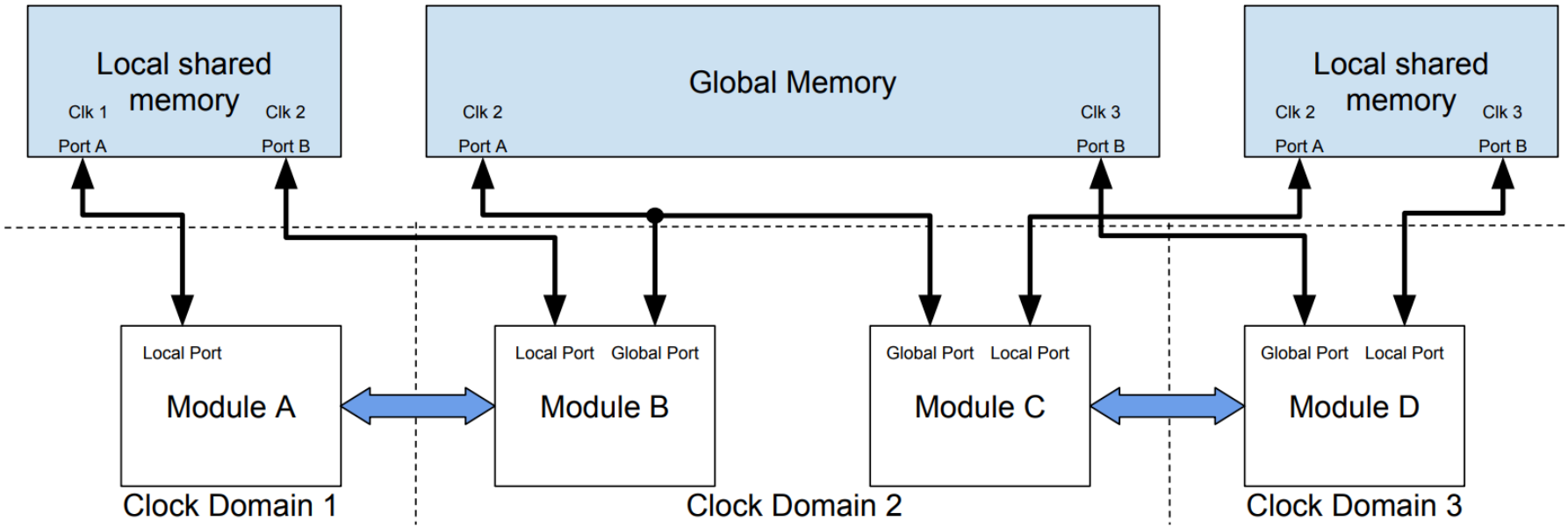




The Edward S. Rogers Sr. Department of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Multi-Clock HLS

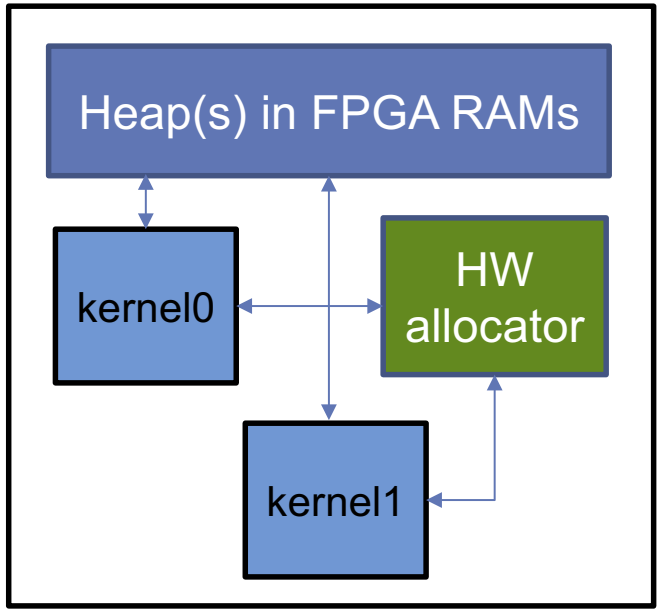
- Partition circuit into modules operating on separate clock domains
- Why? Raise circuit performance by allowing sub-circuits to operate as fast as possible
- Automatically insert clock-domain-crossing circuitry
- Proper handing of memories accessed by modules in different domains



HLS for Dynamic Memory

- HLS tools cannot support synthesis of malloc/free (new/delete), yet these are used heavily in programs
- Researching approaches to realize in hardware

```
void foo(…) {
   …
   p = malloc(…)
   …
   free(q)
   …
}
```

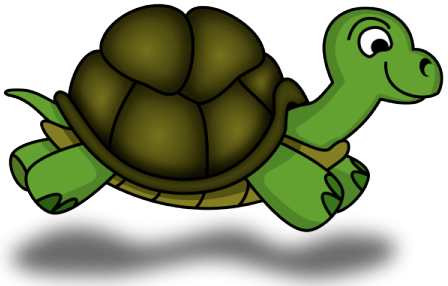


HLS Research Challenges

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

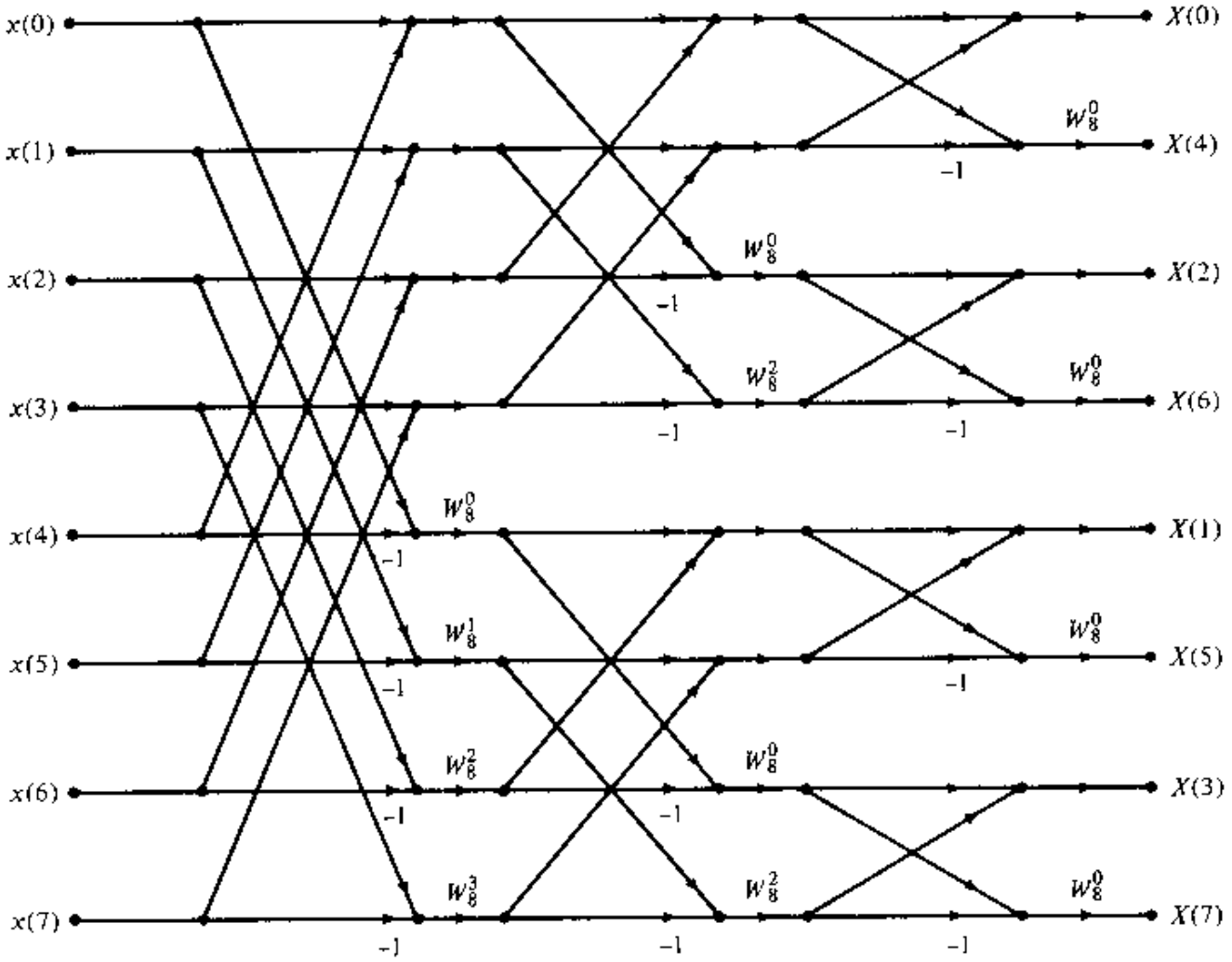
Quality of the Hardware

- HLS-generated circuits may not be as “good” as human-expert-designed circuits



- However, HLS-generated circuits are better (speed+energy efficiency) than SW on a processor in many/most cases

FFT: Hard to Auto-Synthesize



s Sr. Department
uter Engineering
UNIVERSITY OF TORONTO

Syntactic Variance / Constraints

- HLS tool QoR highly sensitive to style of input code + constraints

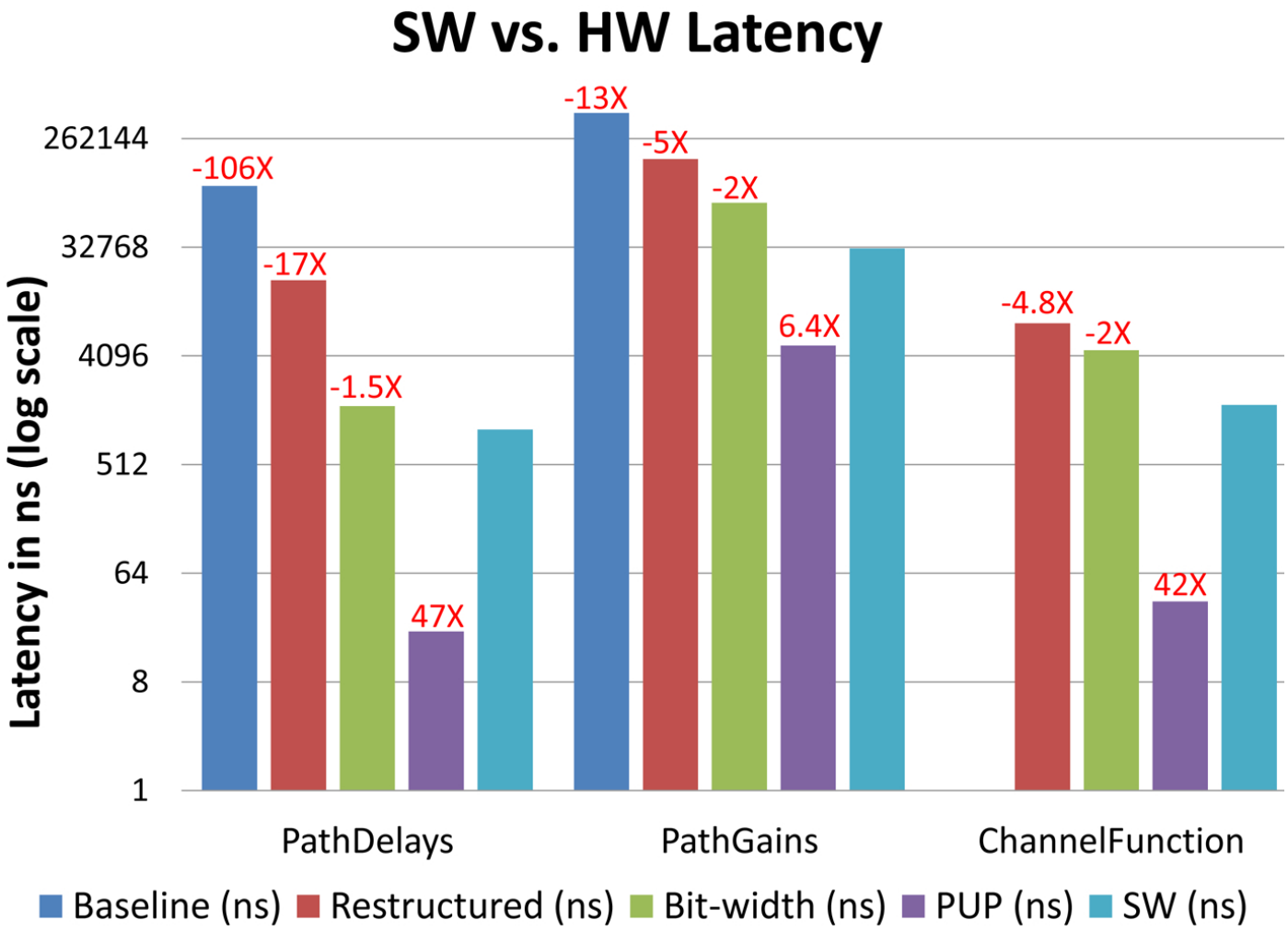
```
for (i = 0; i < 100; i++) {
   if (A[i] & 1)
      sum += A[i];
   else
      sum -= A[i];
}
```

Possibly cannot loop pipeline

```
for (i = 0; i < 100; i++) {
   temp1 = sum + A[i];
   temp2 = sum – A[i];
   sum = (A[i] & 1) ?
            temp1 : temp2;
}
```

Can loop pipeline

Syntactic Variance / Constraints (2)



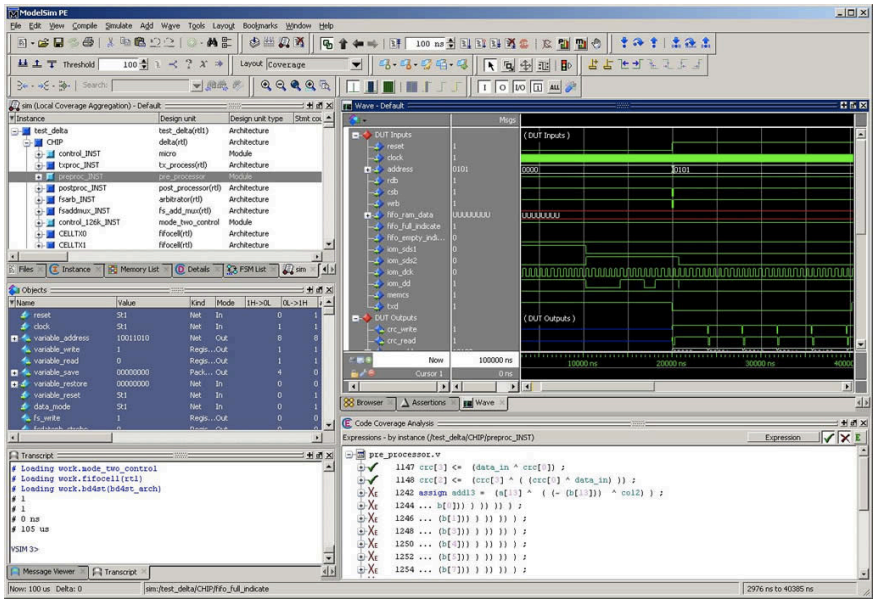
Matai et al., “Designing a Hardware in the Loop Wireless Digital Channel Emulator for Software Defined Radio”, FPT 2012.

Raising Abstraction Further / Beyond C

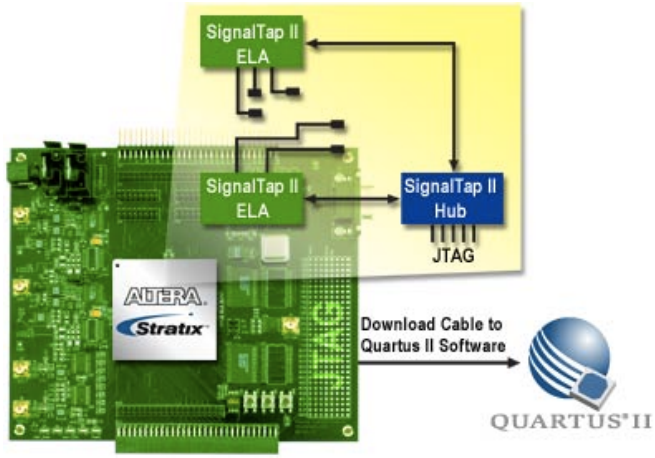
- Learning curve to write HLS-style software+pragmas
- Libraries for specific domains
  - Easy-to-use C/C++ libraries with clean API
  - Underlying implementation of functions is written in “HLS style”
  - Machine learning, compression, computational finance
- Domain-specific languages (DSLs)

Debugging

- Invariably… things go wrong, e.g.:
  - Integration of synthesized HW in system
  - Silicon issues: timing, reliability (SEUs)
- Today's HLS:

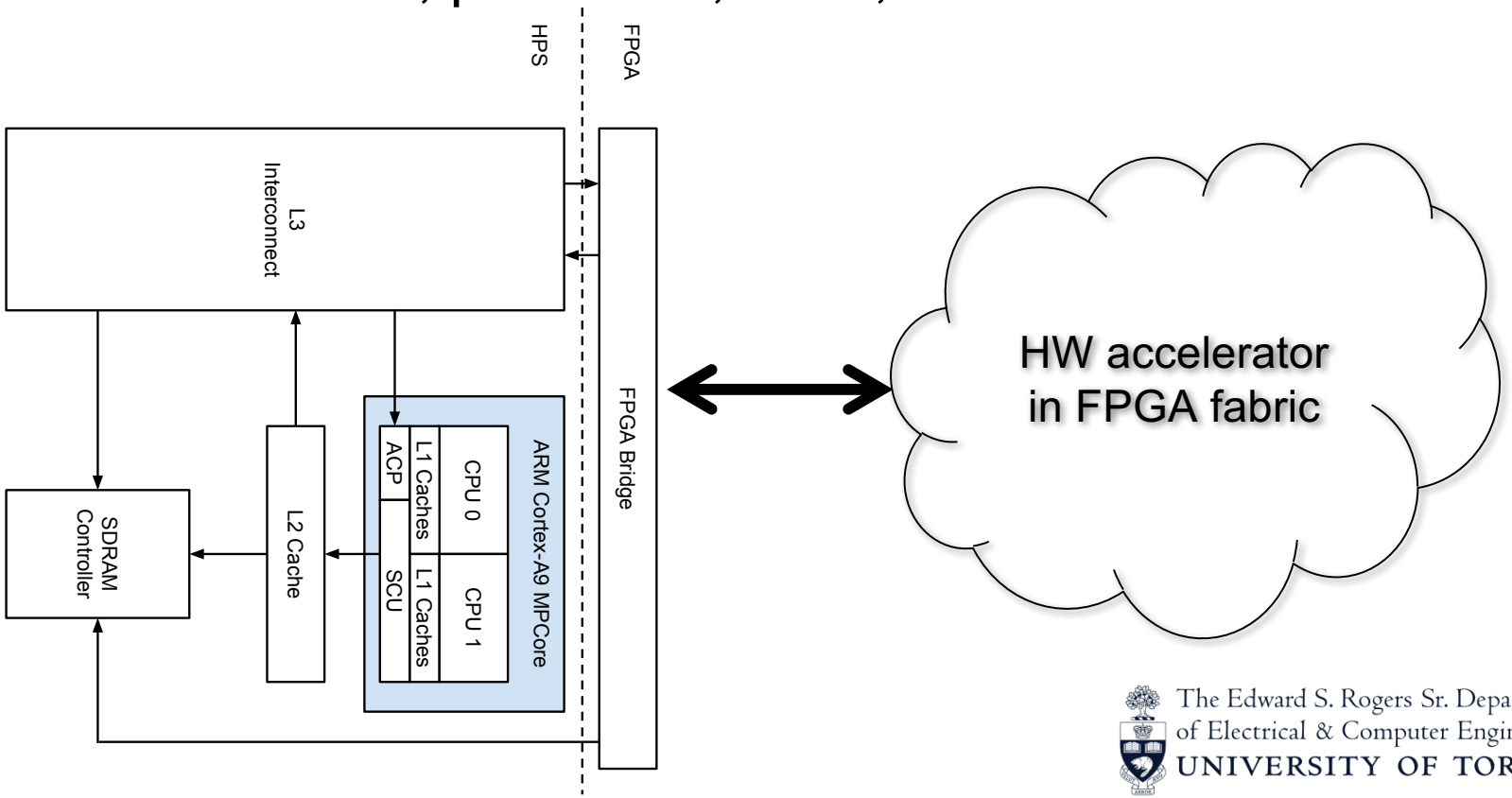


Chip
ScopePro



Debugging Heterogeneous Platforms

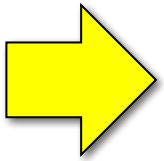
- Debugging just the HLS code is a challenge in itself
- Debugging heterogeneous system with HLS-generated accelerator code, processor, GPU, …

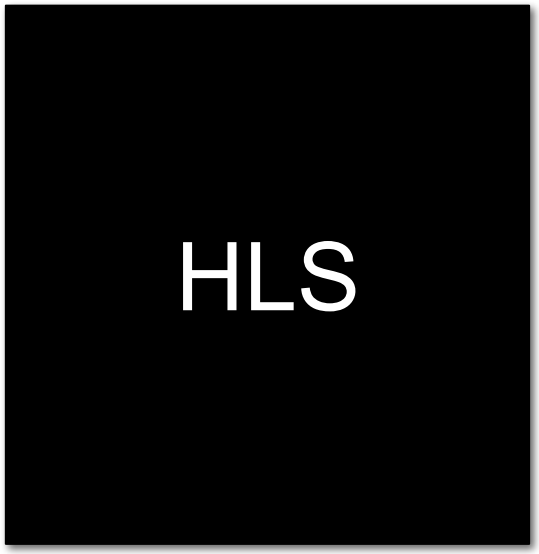


Visualization

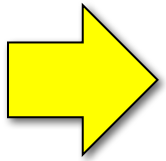
- Today's HLS:

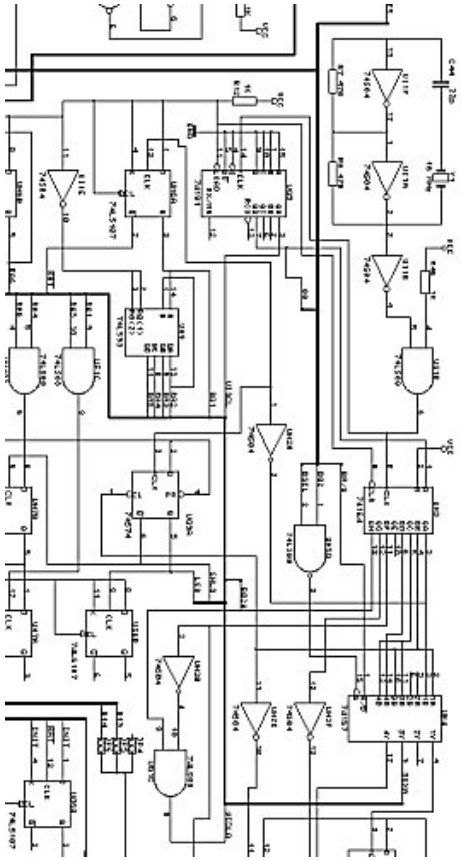
```
for (i = 0; i < 4*Nk; i+
    W[i] = key[i];
for (i = Nk; i < Nb*(Nr+
    {
    temp = Wb[i-1];
    if ((i%Nk) == 0)
        temp = SubByte(R
    Wb[i] = Wb[i - Nk]^t
    }
```





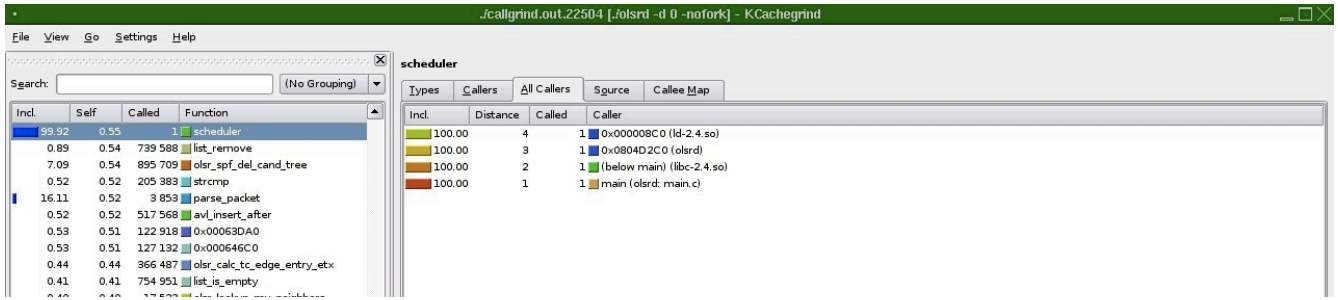
"Black box"



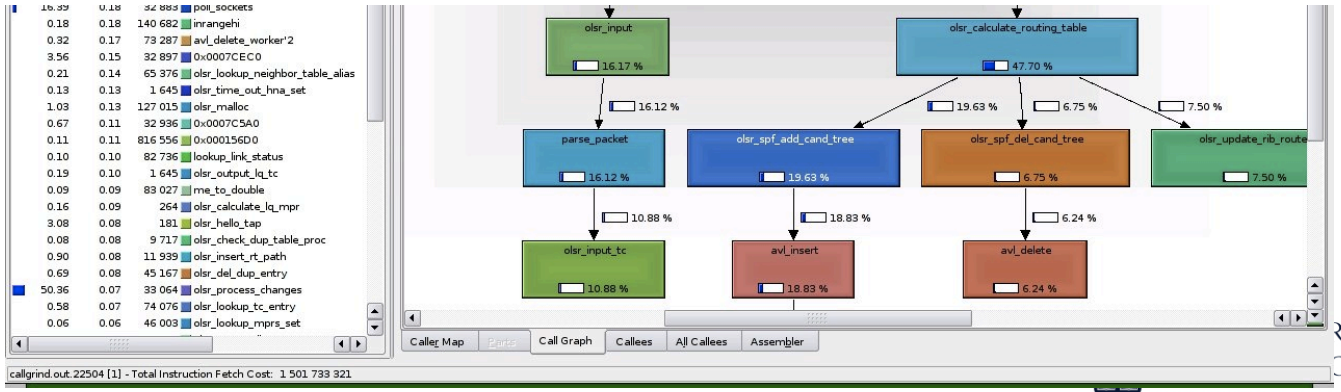


(hundreds/tens) thousands of lines of HDL code

Visualization (2)



"SW-engineer comprehensible"
HW visualization capabilities are needed that guide HW optimization



Commercialization

FOUNDING TEAM

Our research at University of Toronto developed the award-winning LegUp FPGA high-level synthesis design tool

Andrew Canis, Ph.D

CEO

Altera, Sun Labs,
Oracle Labs
10 technical publications

Jongsok Choi, Ph.D

CTO

Intel, Qualcomm, Marvell,
STMicroelectronics
15 technical publications

Ruolong Lian, M.A.Sc

COO

Altera, Google

2 technical
publications

Professor Jason Anderson

Chief Scientific Advisor

University of Toronto
10+ years Xilinx
80+ publications, 28 patents

ENGINEERING TEAM

Zhi Li, M.Eng

Head of Systems Engineering

Intel,
Waratah Capital Advisors

Joined in March 2018

Omar Ragheb, M.Eng

Software Engineer

KACST, Mobiserve

Joined in Feb. 2018

Mehul Gupta

Software Engineering Intern

University of Waterloo

Joined Jan. 2019

40

Company Background

- LegUp Computing was founded in 2015
- Spin-off from the University of Toronto
- Offices in Toronto, Canada
- 6 full-time engineers and growing
- Seed funding from Intel in January 2018

- Revenue:
  - 3+ years ongoing contract with FPGA vendor using our HLS/SoC tools
  - Licensing revenue from embedded engineers using LegUp for low-latency motor control applications

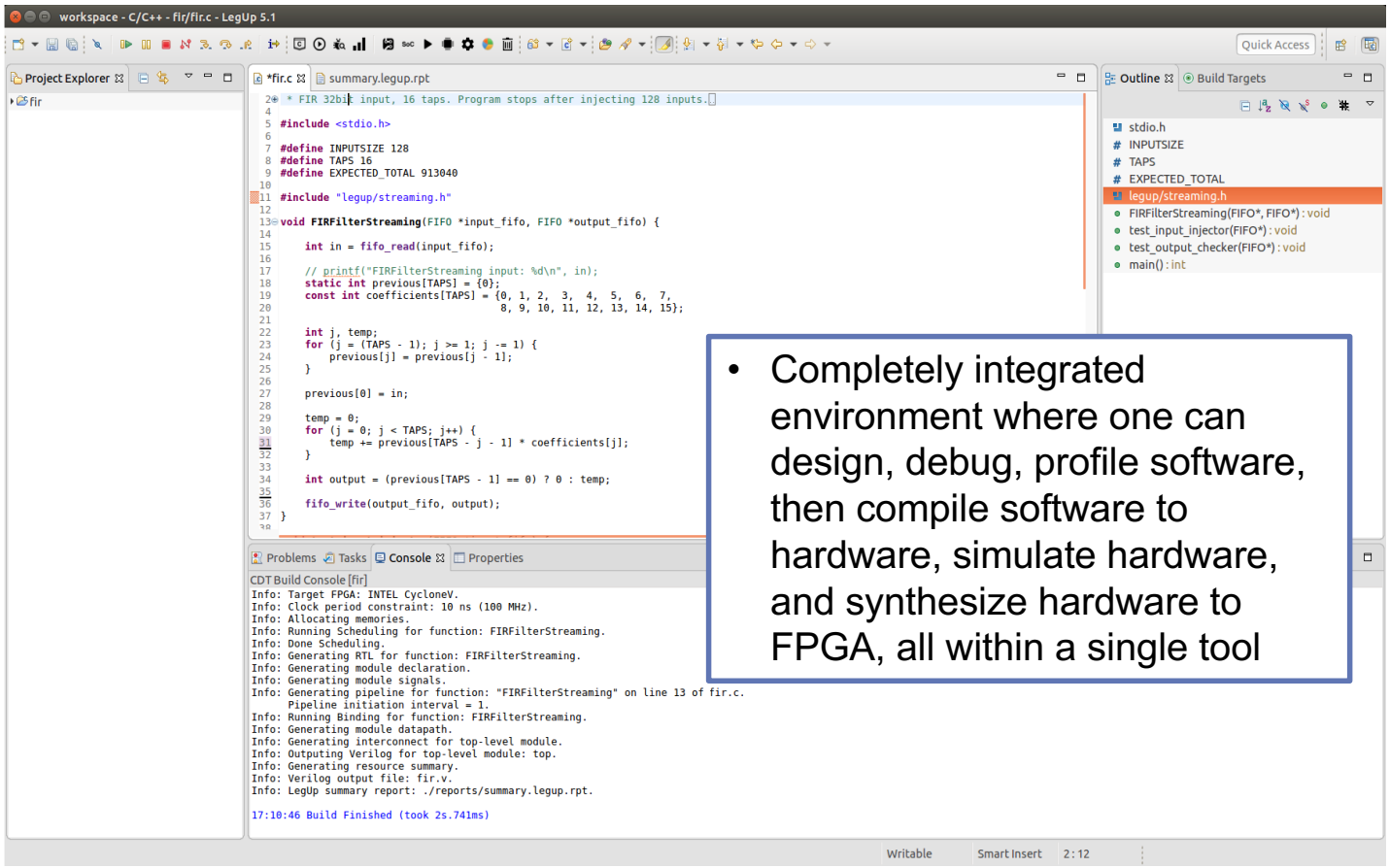
www.legupcomputing.com

LegUp HLS: Commercial Release

- Latest 6.7 release in Mar. 2019
- Downloadable via website
  - 30-day free trial; paid yearly subscription
- Windows & Linux support

- Key Features
  - Multi-threading support
  - Best-in-the-class pipelining
  - Push-button System-on-Chip generation
  - Vendor-agnostic

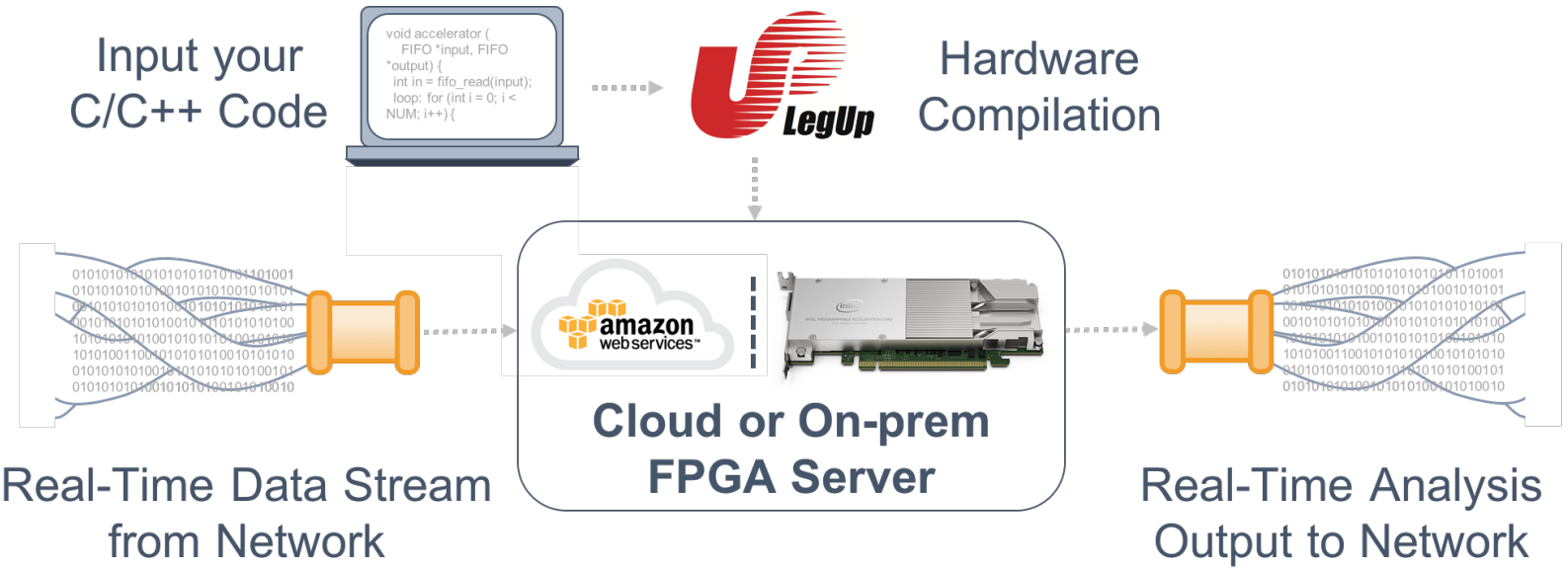
LegUp Graphical IDE: Windows/Linux



- Completely integrated environment where one can design, debug, profile software, then compile software to hardware, simulate hardware, and synthesize hardware to FPGA, all within a single tool

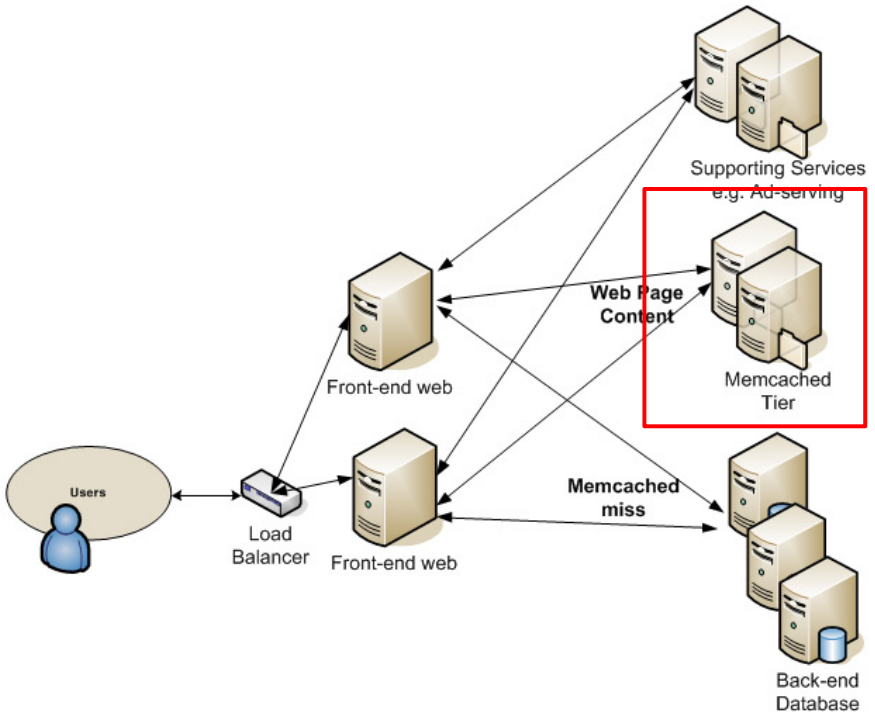
CLOUD PLATFORM

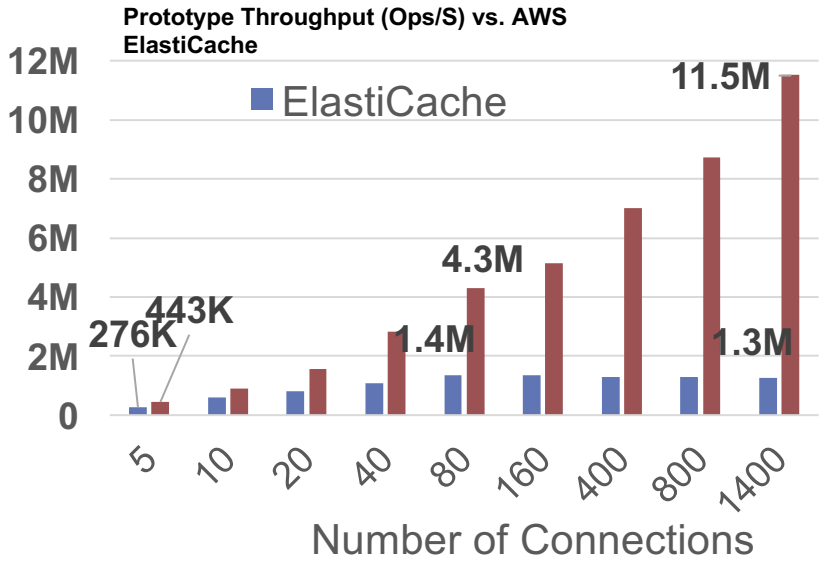
- Network processing engines on cloud FPGAs and Intel's on-premises acceleration cards



Accelerating Memcached on AWS

- Memcached is high-performance, distributed memory object caching system
  - Used by Facebook, Twitter, Reddit, Youtube, etc





Business Models

1. **Software licensing model**
   - Revenue:
     - Yearly licensing fee per seat of the HLS software
     - Support contract for features and bug fixes
   - Customers:
     - Engineers using FPGAs who want higher productivity
     - FPGA vendors who need a HLS tool to stay competitive
     - Software engineers using FPGAs in cloud
2. **Applications running on FPGAs**
   - Cloud FPGA or on-premise FPGA applications
     - Database applications like Memcached
     - Financial trading and risk analysis algorithms
     - Deep learning, image/audio processing, analytics
   - Revenue: $/instance/hour on cloud or licensing bitstream for on-premise

THANK YOU! QUESTIONS?

https://janders.eecg.utoronto.ca/
janders@eecg.toronto.edu

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO