THE UNIVERSITY OF TEXAS AT DALLAS

Department of Electrical and Computer Engineering

HAMMER : Hardware-aware Runtime Program Execution Acceleration through runtime reconfigurable CGRAs

Qilin Si, and **Benjamin Carrion Schaefer**

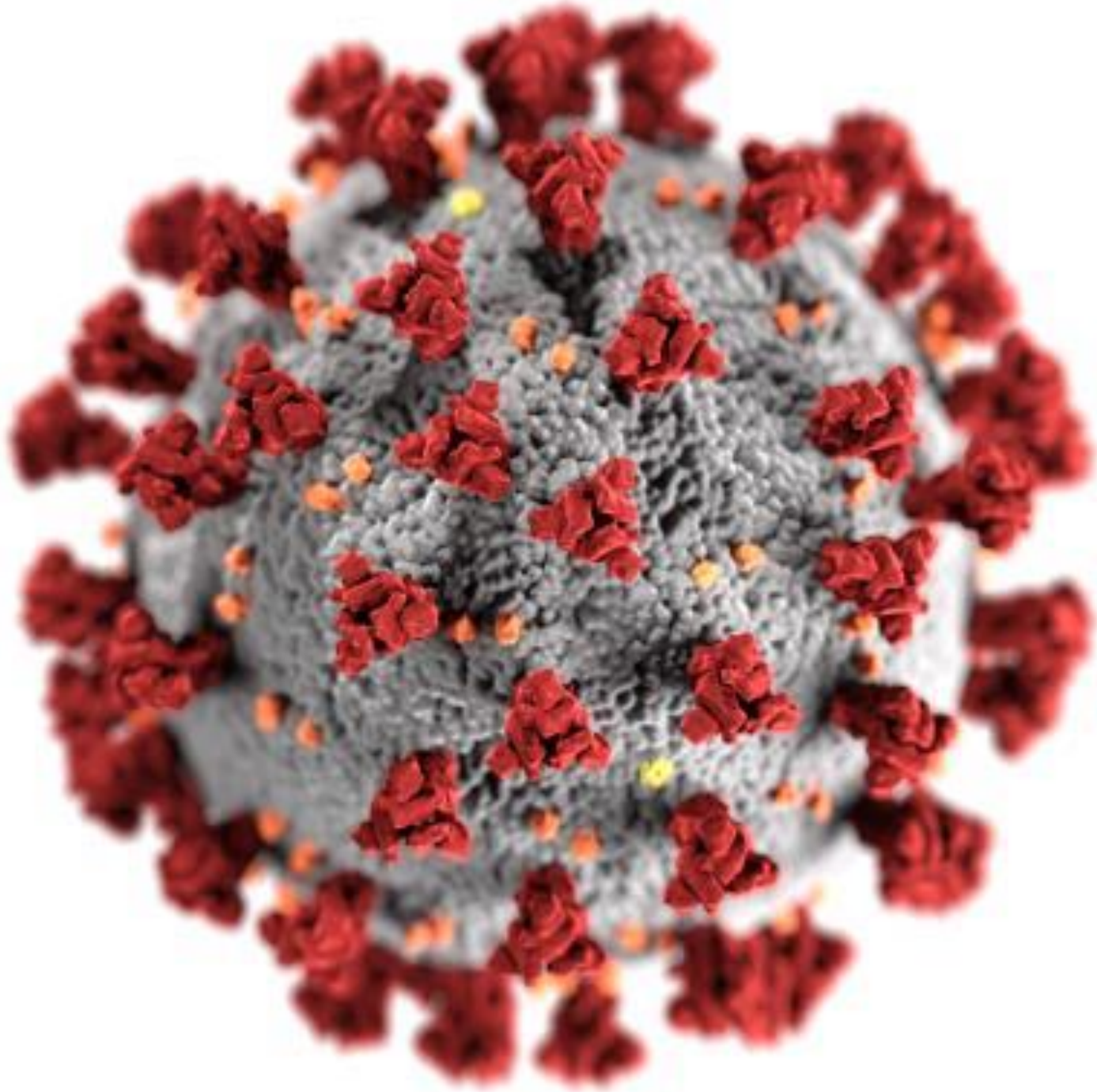
Qilin.si@utallas.edu, **schaferb@utdallas.edu**

ASIA SOUTH PACIFIC DESIGN AUTOMATION CONFERENCE

30th Asia and South Pacific Design Automation Conference

ASP-DAC 2025

January 21, 2025, Tokyo



AUTOS

How Covid led to a $60 billion global chip shortage for the auto industry

PUBLISHED THU, FEB 11 2021•7:17 AM EST

Michael Wayland
@MIKEWAYLAND

SHARE

How a tiny chip wounded the U.S. auto industry

Visit ›

Administration

JUNE 17, 2021

Why the Pandemic Has Disrupted Supply Chains

CEA › WRITTEN MATERIALS › BLOG

By Susan Helper and Evan Soltas

Motivation

- Similar electronic systems have different components. E.g., low-end vs. high-end cars

What

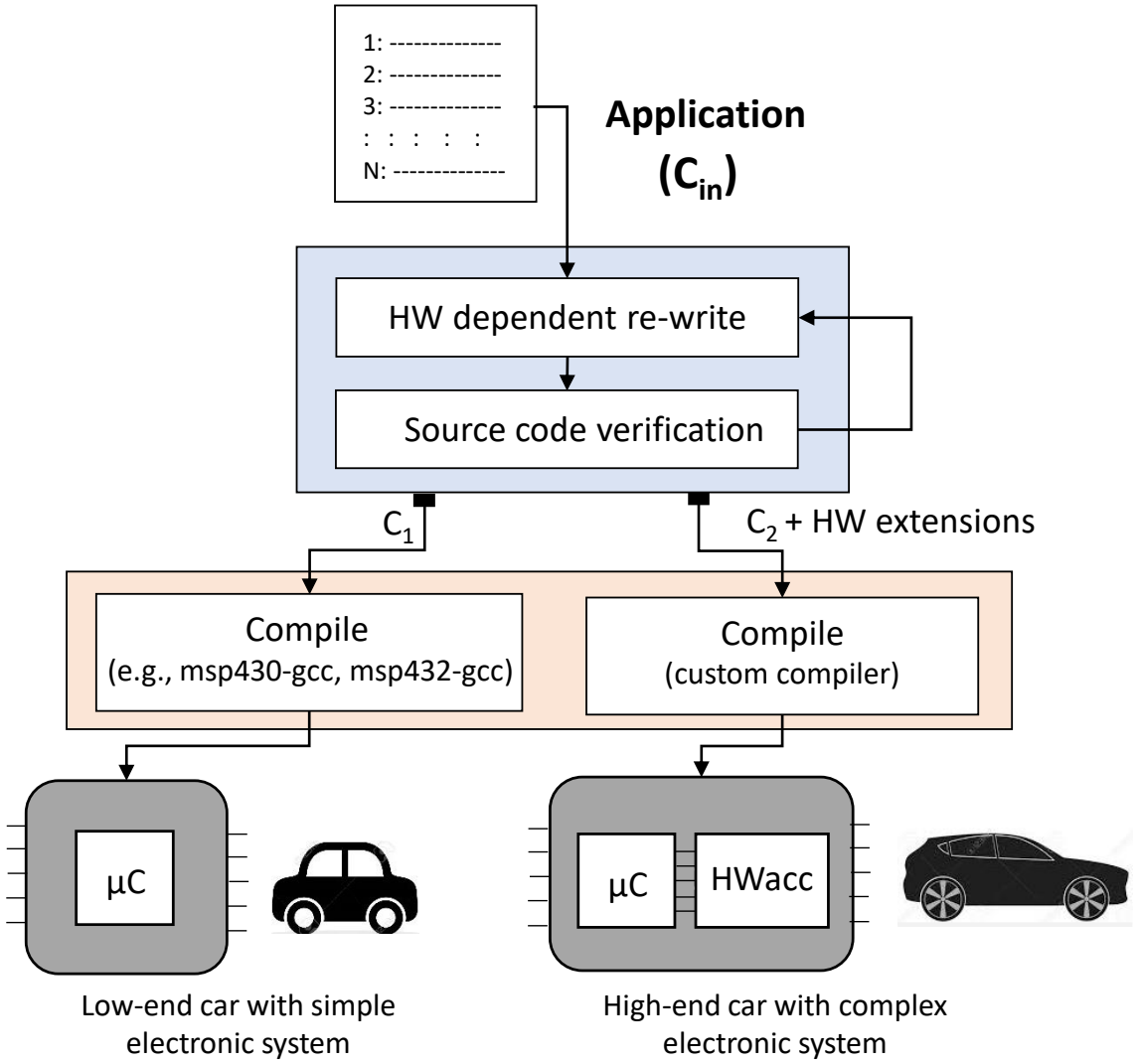
- Runtime hardware-aware architecture and design flow that is able to automatically accelerate compiled programs based on underling hardware

Why

- Reduces development time (design and verification)

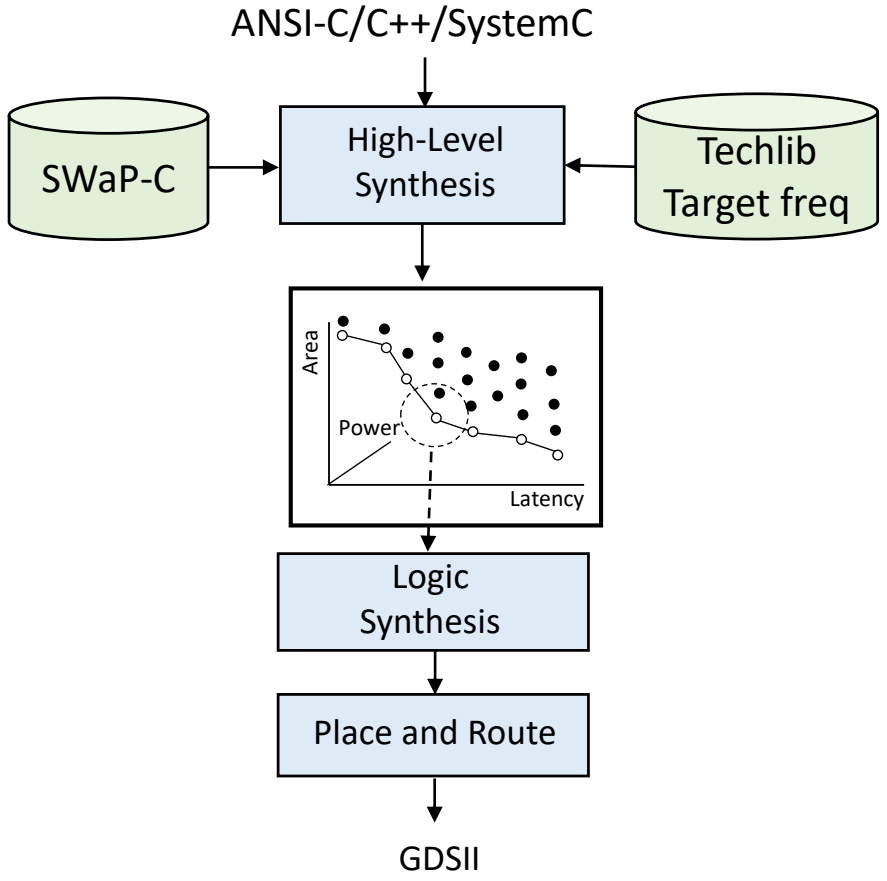
How

- Map accelerators to Coarse-grain Reconfigurable array which his programmed with accelerator based on runtime detection of piece of SW which can be accelerated

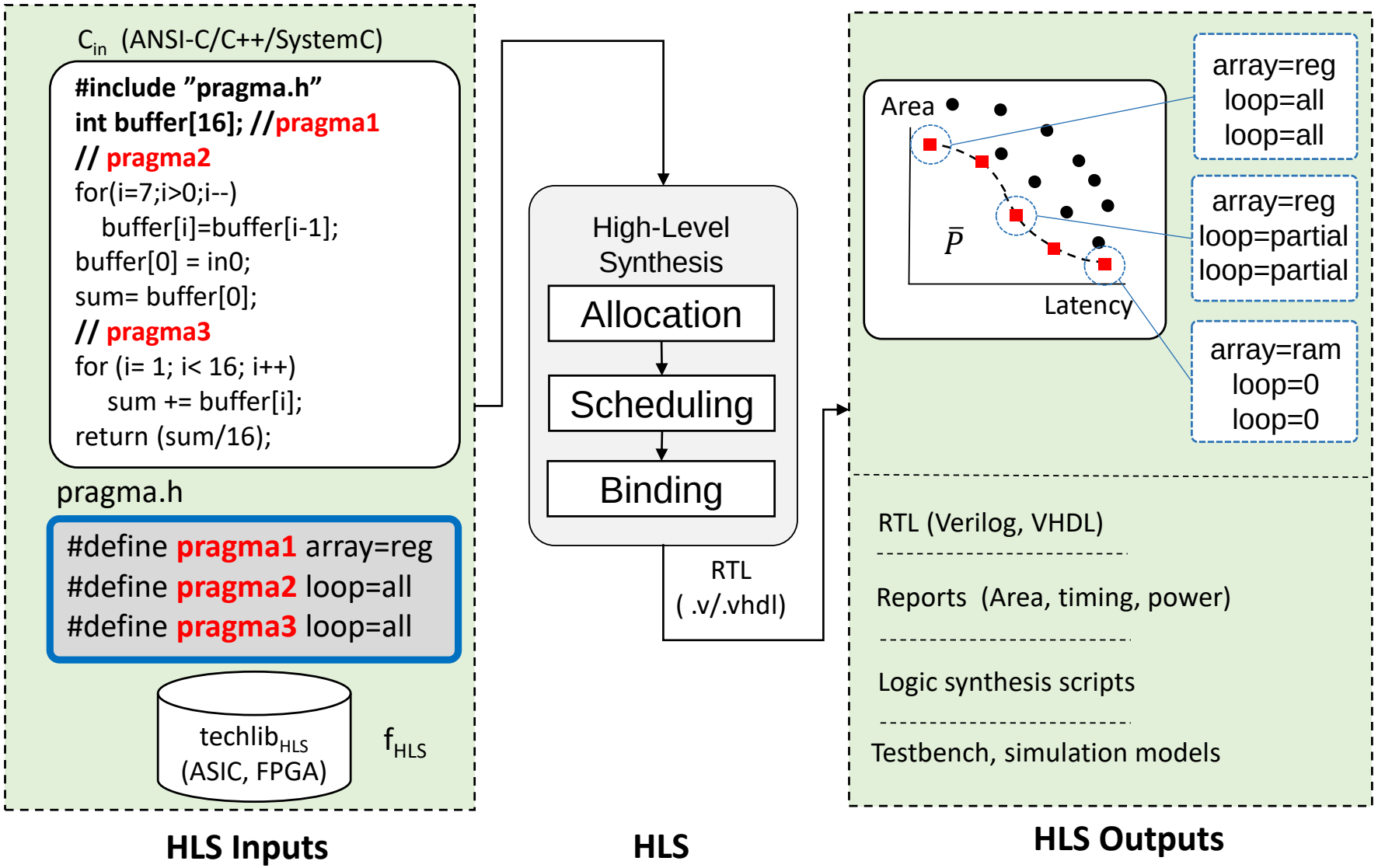


High-Level Synthesis

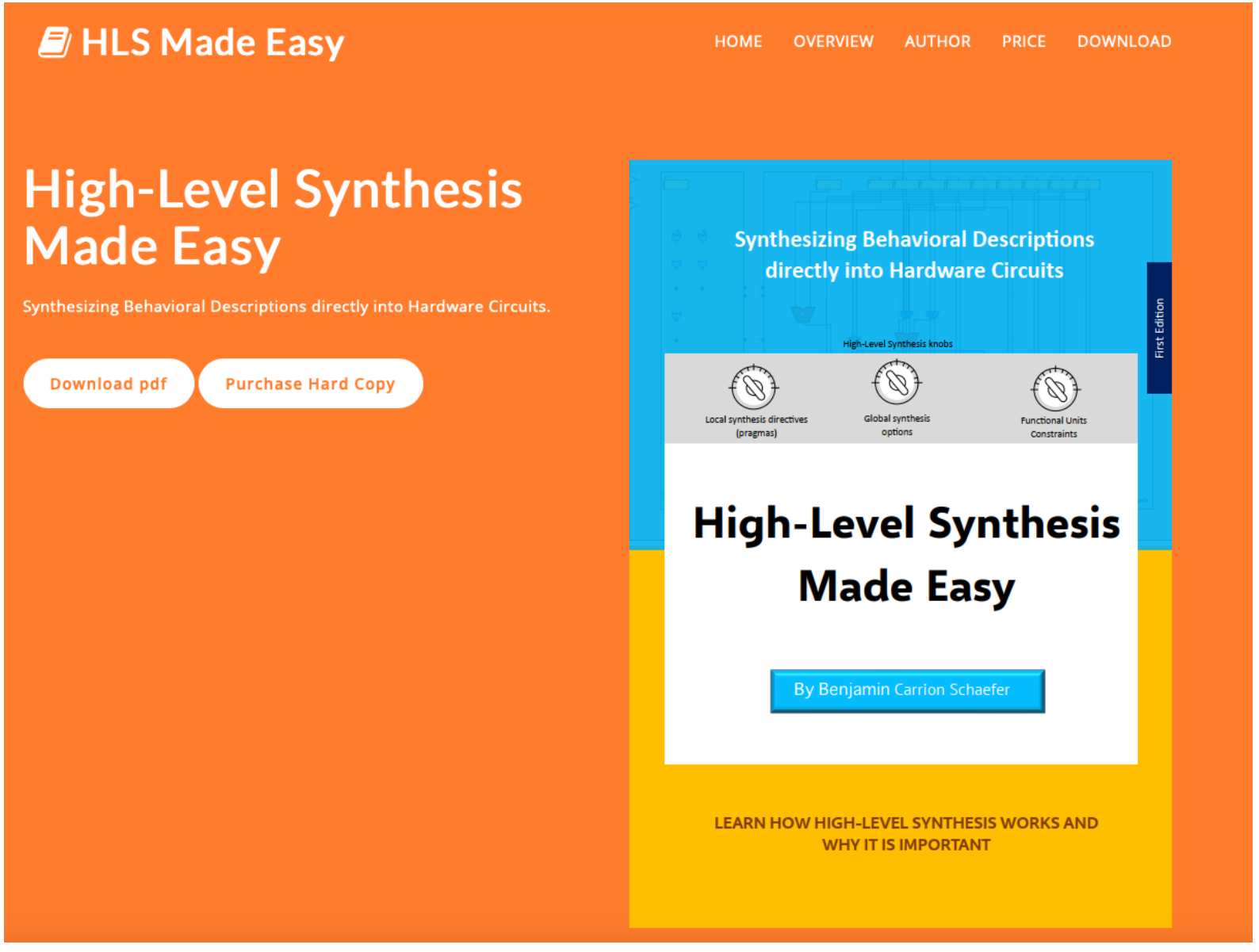
- **High-Level Synthesis**
  - Design circuits using software languages
  - Definition:
    "Automatic conversion of a behavioral, untimed description into efficient hardware that implements that behavior"
  - Benefits
    1. Software programmability and hardware performance
    2. Faster verification
    3. Decouples functionality from implementation → allows to easily re-target any behavioral description to new technologies and newer design constraints (i.e., area, power, performance)



High-Level Synthesis in Practice

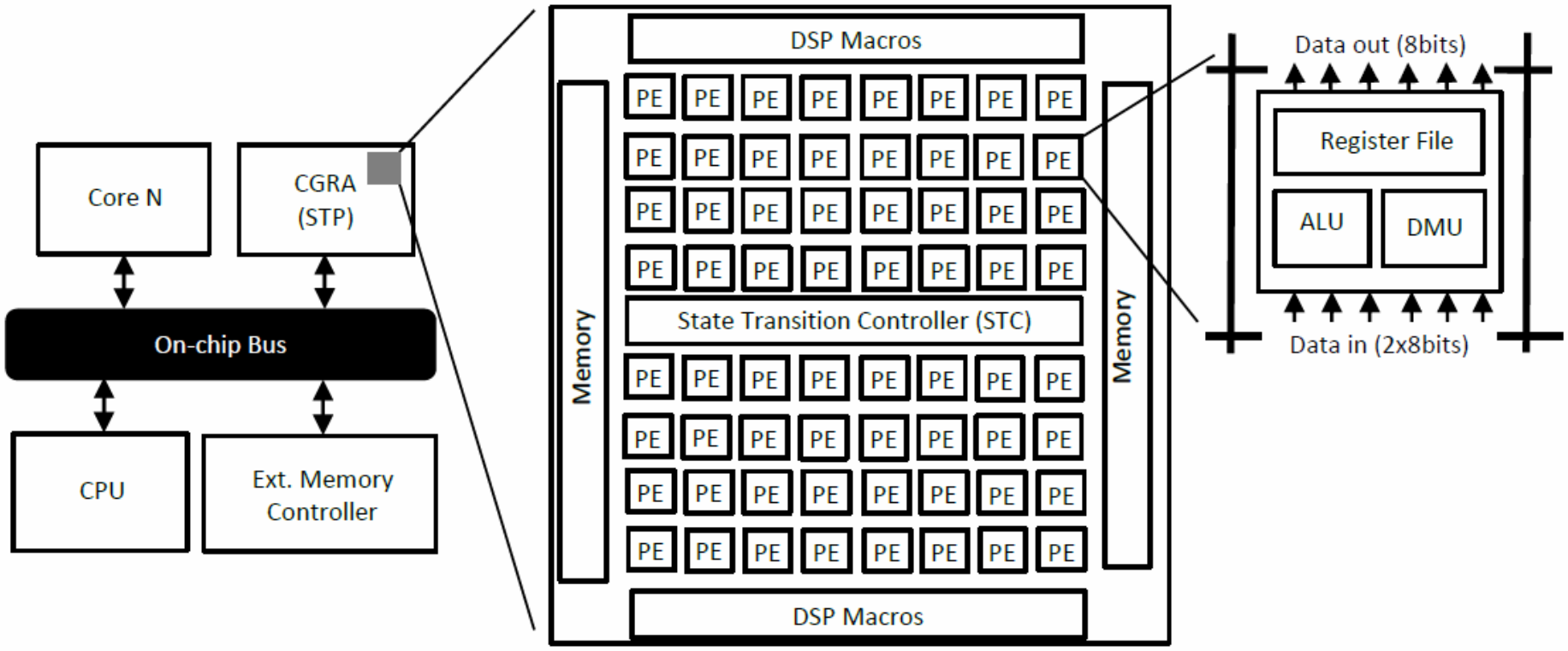


High-Level Synthesis Made Easy - www.hlsbook.com



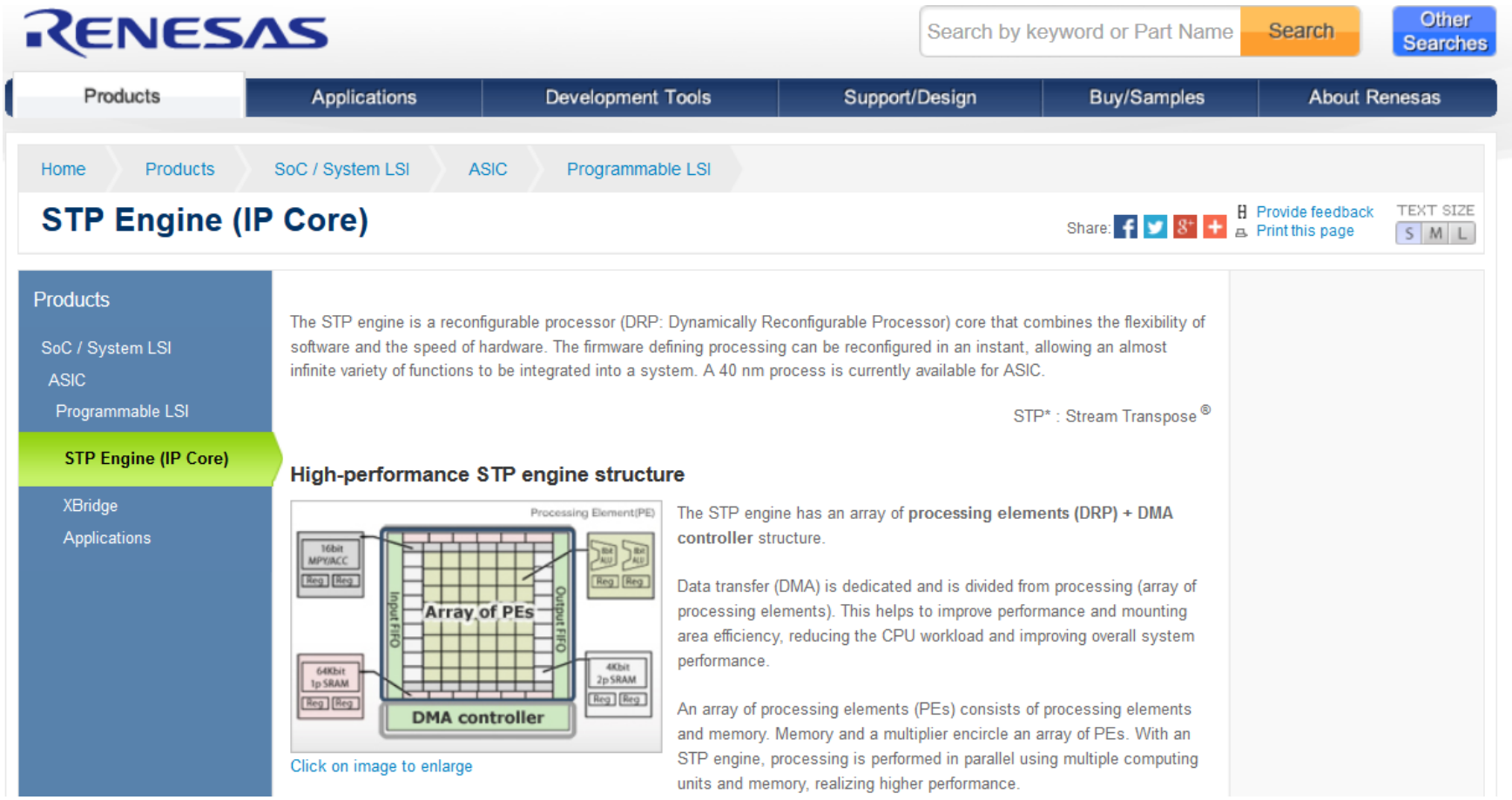
Coarse-grain Runtime Reconfigurable Array

- Coarse Grained Runtime Reconfigurable FPGAs
- Often included as reconfigurable IP in heterogeneous SoCs



Stream Transpose Processor (STP)

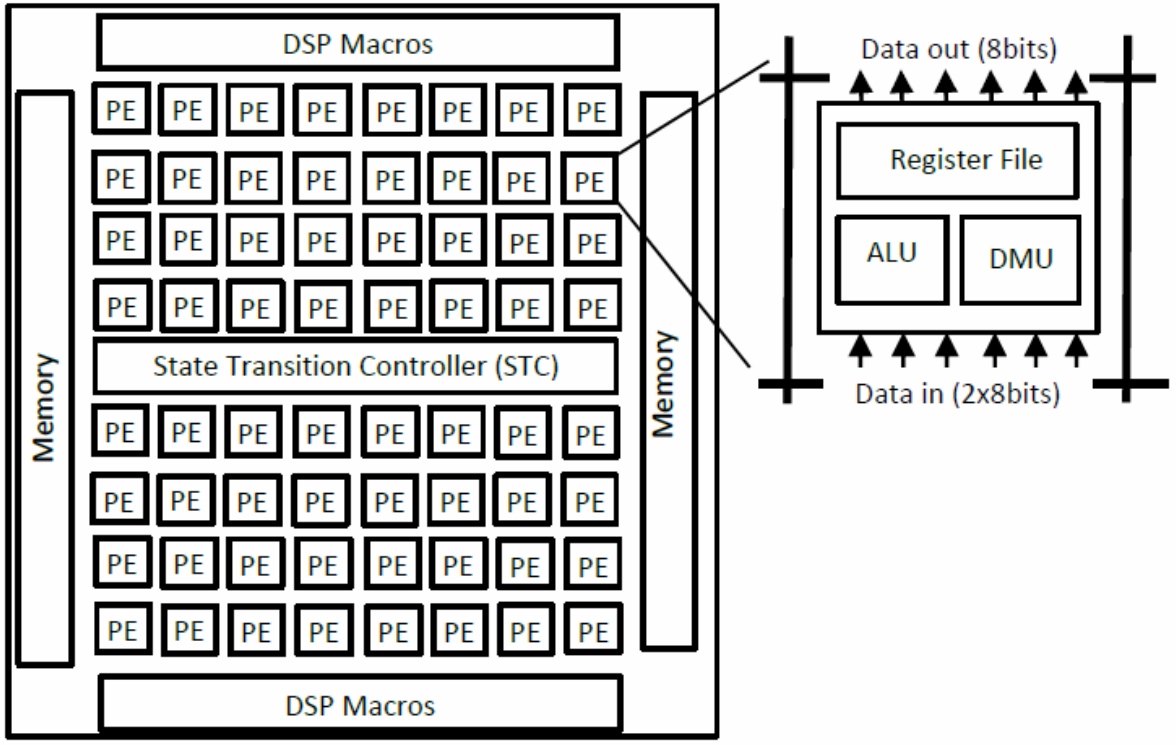
- Runtime Coarse Grain Reconfigurable Architecture (1ns to reconfigure)
- Programmed using High-Level Synthesis (HLS)



http://am.renesas.com/products/soc/asic/programmable/stp/index.jsp

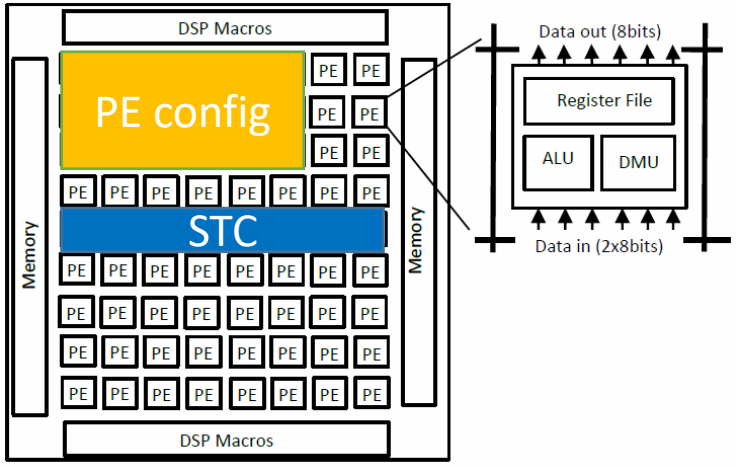
Stream Transpose Processor cont

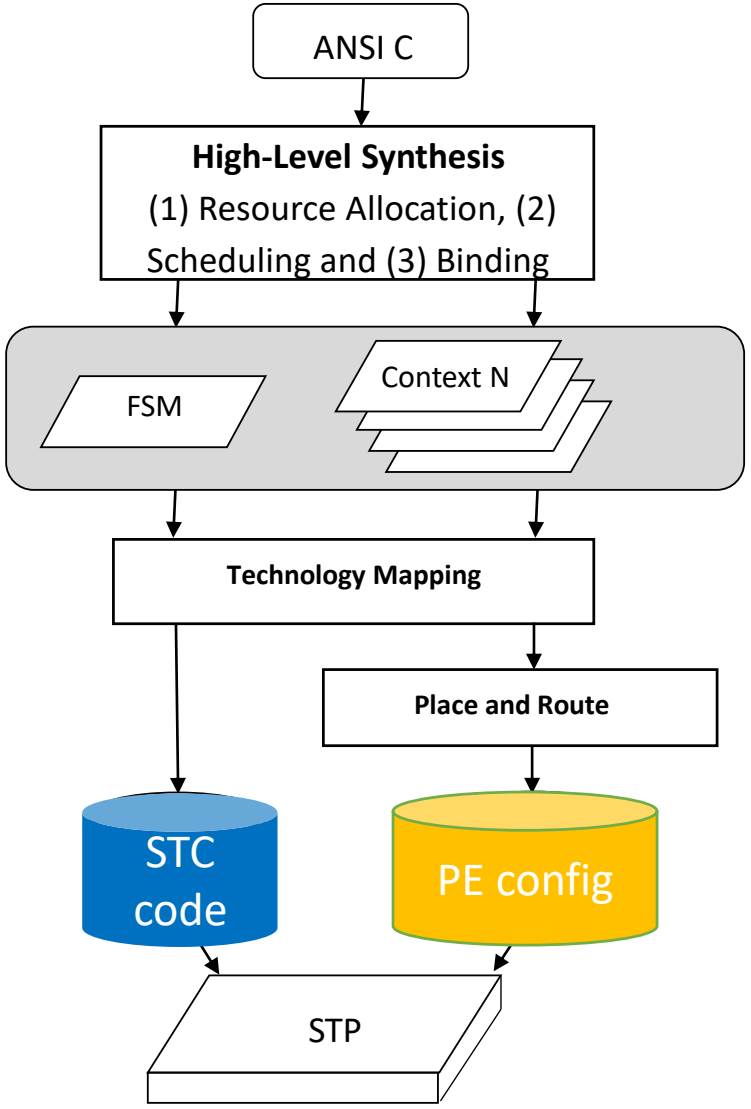
- The main building blocks are called tiles
- Each tile consists of an array of 8x8 PEs. Each PE contains:
  - 8-bit arithmetic logic unit (ALU), an 8-bit data manipulation unit (DMU) for 1-bit logic operations and 8-bit shifting and masking and an 8-bit flip-flop unit (FFU).
- Surrounded by embedded memory and embedded multipliers
- The STP can hold up to 64 contexts in its State Transition Controller located in the middle of the PE array



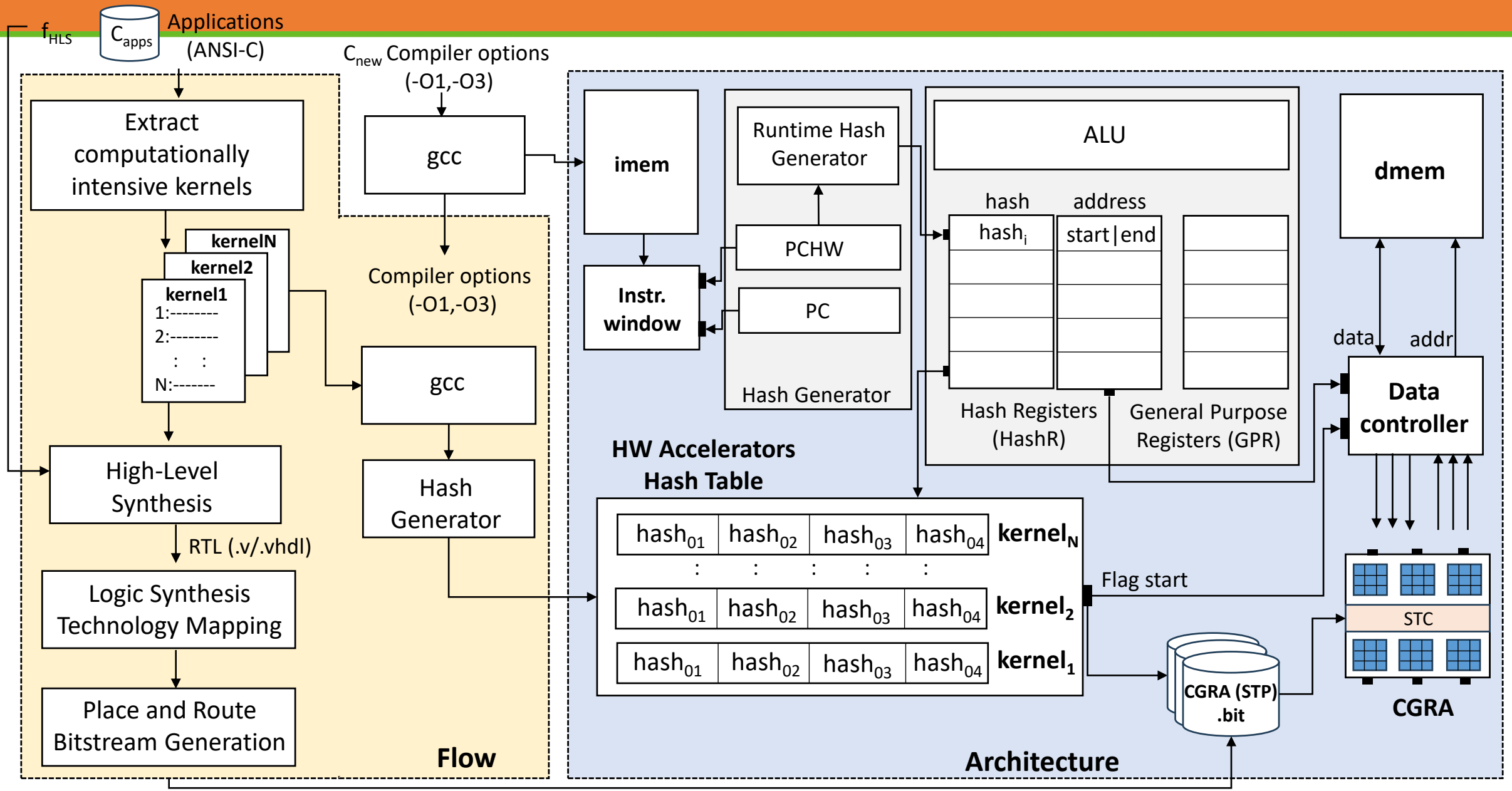
STP Configuration Flow

- Input : Sequential description in C
- Synthesis : High-Level Synthesis
- Output:
  - FSM mapped to State Transition Controller (STC)
  - Data Path mapped to PEs in the form for contexts which the STC uses to reconfigure the reconfigurable fabric every clock cycle

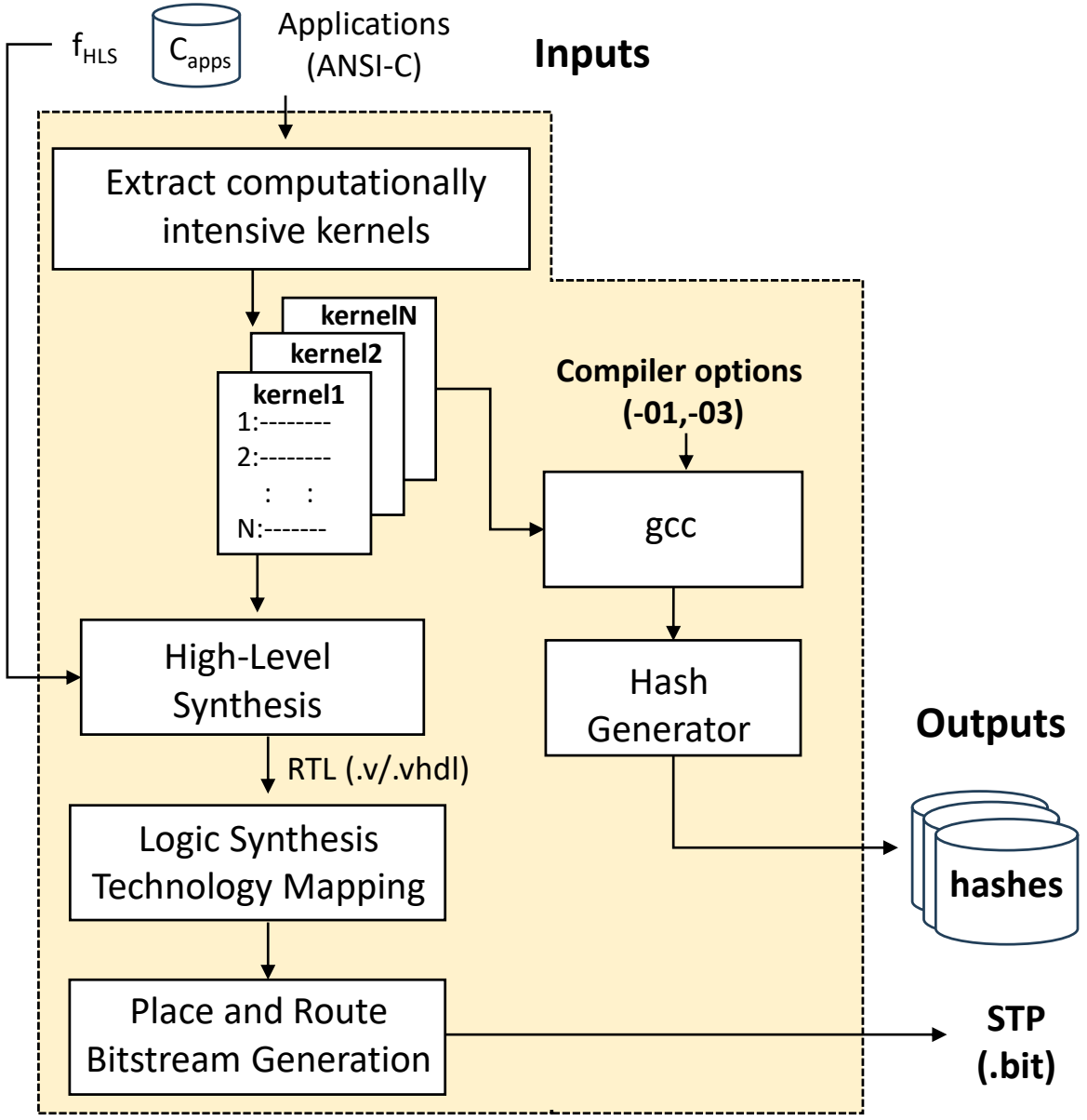




HAMMER Flow and Architecture



HAMMER - Flow



- **Inputs**: Applications in C
- **Outputs**:
  1. Synthesized (HLS) kernels mapped onto CGRA
  2. Hash value that identifies the kernel based on instruction sequence

     Kernels are compiled with different compiler options (-O1, O2, etc..) and their hash values included in database

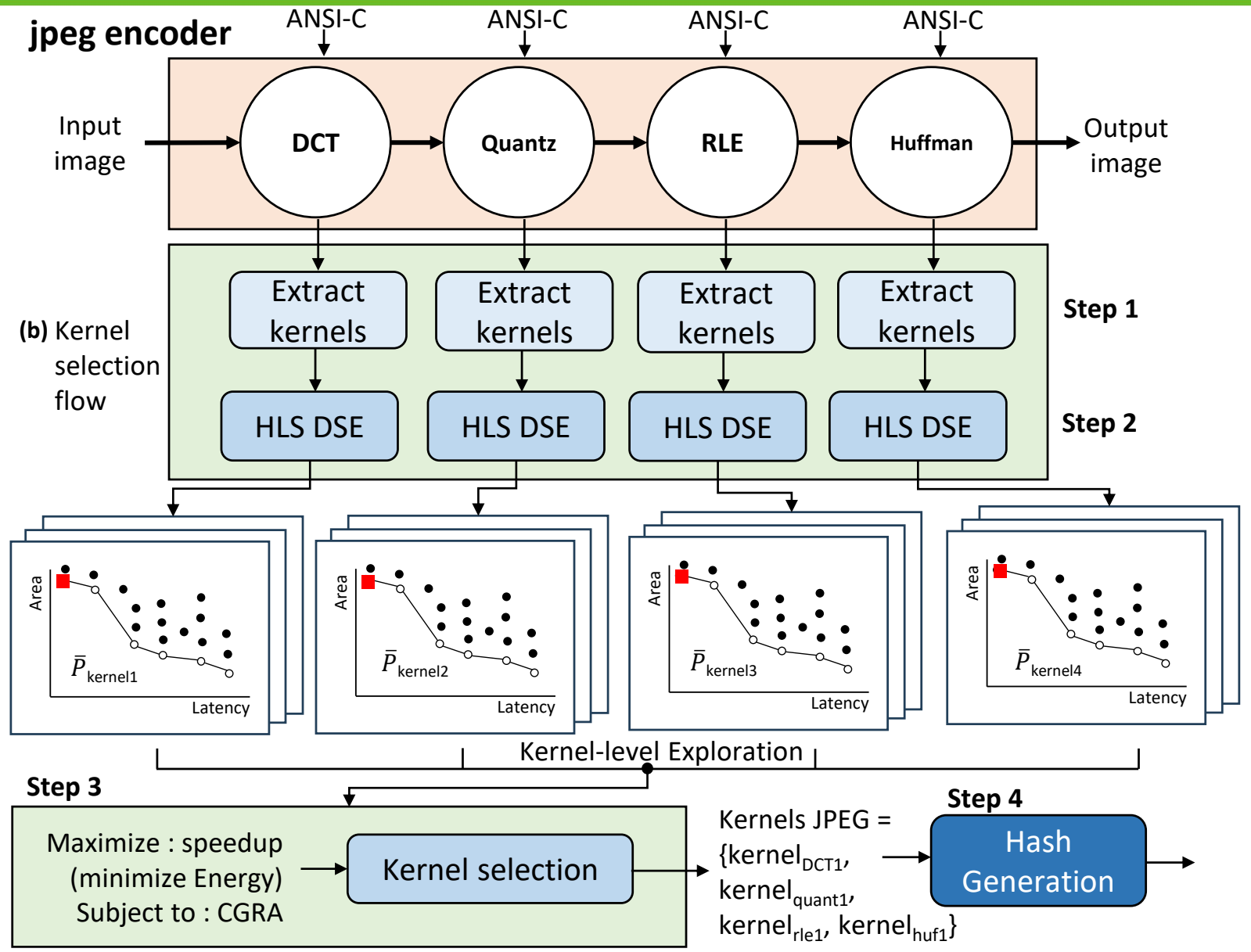
Kernel Pre-Characterization

Step 1 : Extract kernels that can be accelerated

Step 2: Perform a full HLS pragma-based HLS Design Space Exploration (DSE)

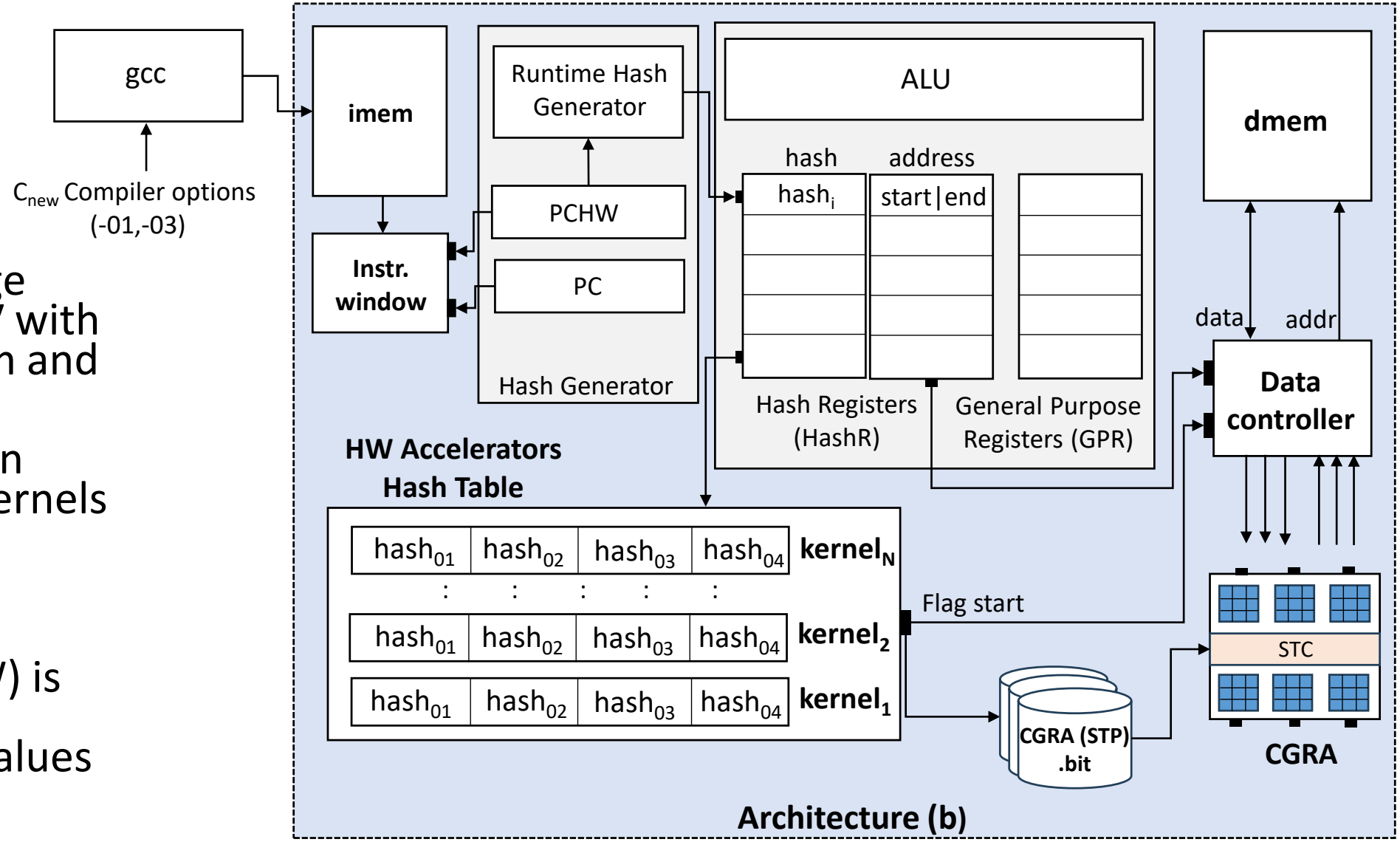
Step 3: Kernel selection

Step 4: Generate Hash for each kernel



HAMMER - Architecture

- Processor 5-stage pipelined RISC-V with imem and dmem and CGRA
- CGRA .bit contain database with kernels that can be accelerated
- Two Program Counters (PCHW) is ahead of PC to compute hash values in advance



Experimental Setup

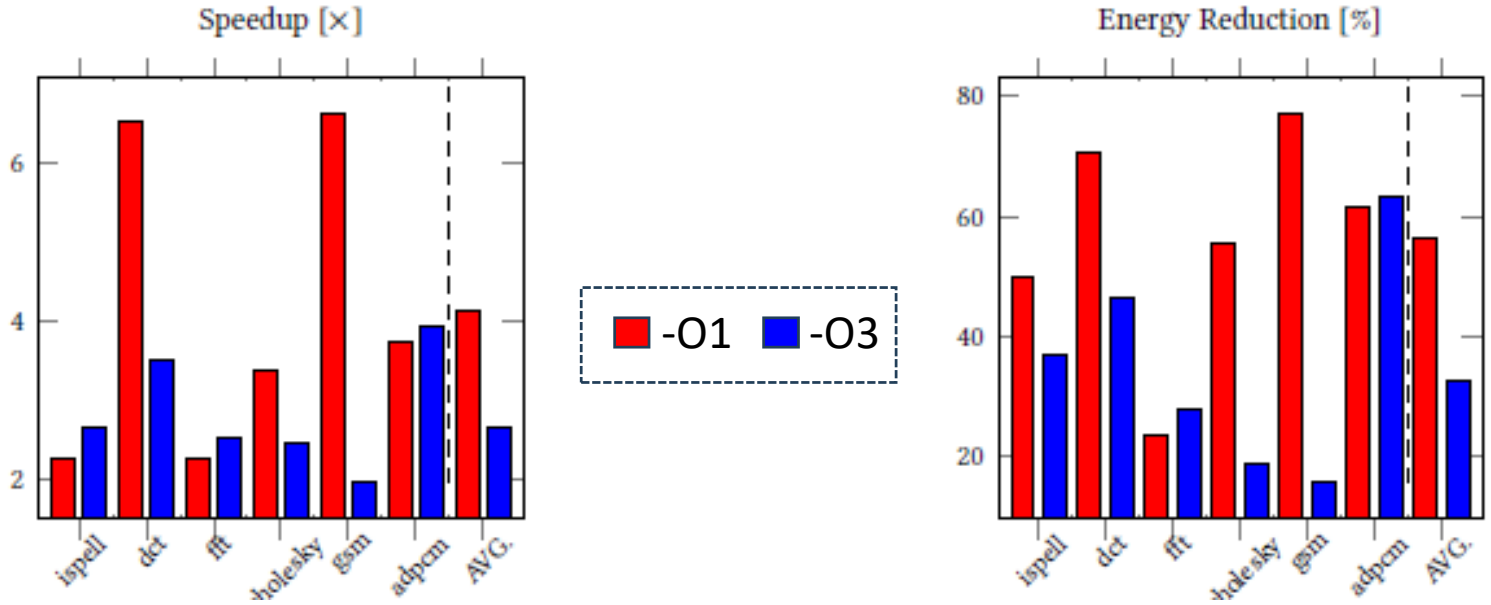
Tools

- HLS Tools : Renesas Electronics Musketeer 1.2
- Logic Synthesis tool: Synopsys Design Compiler v.0-2018.06-SP1
- Target technology: Nangate Opencell 45nm
- Target synthesis frequency: 200 MHz
- Power Simulator: Synopsys PrimePower v.P-2019.03-SP5
- RTL and gate-level simulator: Synopsys VCS 0-2018.06
- Process Compiler: riscv-gcc v. 20191213
- Applications: MiBench and CHstone

Evaluation

Benchmark	Source	#Instr (-O1)	#Instr (-O3)	Kernels
ispell	MiBench	82K	115k	1
dct	Chstone	130K	179k	1
fft	Mibench	141K	142K	2
cholesky	MiBench	144K	110K	2
gsm	MiBench	170K	257k	3
adpcm	CHStone	186K	186l	1

Experimental Results : Speedup and Energy



Observation 1: Our proposed architecture was able to accelerate at least one kernel in all of the benchmarks which lead to different speedup and energy reduction in all of the benchmarks

Observation 2: The average speedup achieved was 4.12 and 2.64 for the -O1 and -O3 cases respectively

Observation 3: On average the energy could be reduced by 56% and 33% for the-O1 and -O3 cases respectively.

Experimental Result –CGRA Fabric Size (JPEG Case Study)

CGRA fabric size

	100%	80%	60%	40%
-O1 Speedup kernels	4.65	3.85	3.25	1.95
-O3 Speedup kernels	4.85	3.75	2.95	1.85

100% = 64 STP tiles each with 64 PEs

Observation 1: The CGRA size does not impact the number of kernels that can be accelerated
→ determined by the memory that holds the CGRA bitstreams

Observation 2: The speedup is highly correlated with the size of the CGRA. Larger fabric allows larger and faster kernel implementations

Conclusions

- Proposed a processor system based on a standard RISC-V augmented with a CGRA that can accelerate sequential code without recompilation
- The system detects at runtime if a portion of the code can be accelerated or not
- High-Level Synthesis is used to pre-characterize the hardware accelerators (HLS DSE)

Thank You