Latte: Locality Aware Transformation for High Level Synthesis

Jason Cong, Peng Wei, Cody Hao Yu, Peipei Zhou

Motivation
- Frequency decreases as design size scales out in HLS accelerators.
- Severe frequency degradation:
  - 30% chip resource -> 200MHz
  - 74% chip resource -> 150MHz
  - 90% chip resource -> 132MHz
  - Extreme case: FFT, 88% area -> 57MHz

Achilles' heel
- Identify four common collective communication and computation pattern:
  - scatter, gather, broadcast and reduce.
  - One-to-all or all-to-one on-chip data movement.
  - Wirelength scales up to critical path delay.
  - Patterns are used in most, if not all, accelerators.

Common Practice Accelerator
- Use manual unrolling.
- Use #pragma loop unrolling.
- Use DRAM interface -> on-chip bram buffers.

Chip Layout Look Like?
- Scattered distribution of local buffers (PEs)
  - HLS optimistically estimate memcpy() function wire delay without considering locality of partitioned local buffers.
  - Scatter, gather, broadcast and reduce are similar to the patterns of scatter and gather.

Latte in HLS
- 260 lines of code (LOC) to manually implement Latte in HLS, 10x more than baseline code.
- PTC in Scatter Code snippet

Latte Automation
- Latte provides semiautomatic framework
  - Users insert simple Latte pragmas into user-written HLS kernel.
  - Best design configuration is picked.

“Caffé” Latte
- Latte boosts frequency
  - Latte-optimized design improves timing of the baseline by 1.5x with only 3.2% LUT overhead
    - 30% chip resource, 200MHz -> 227MHz
    - 74% chip resource, 150MHz -> 189MHz
    - 90% chip resource, 132MHz -> 175MHz
    - Extreme case: FFT, 88%, 57MHz -> 152MHz

Latte is “Light Roast”
- Latte improves freq. with negligible overhead
  - 1.5x with 3.2% LUT, 5.1% FF on average
  - 2.66x with 2.7% LUT, 5.1% FF at max
  - Helps greatly in frequency degradation

How does Latte work?
- PTCs are chained in linear fashion through FIFOs. Each PTC connects to a local set of buffers to constrain wire delay.

Can we do better? Yes!