Latte: Locality Aware Transformation for High Level Synthesis

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

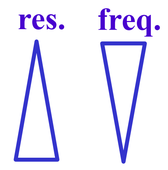
Peipei Zhou @ UCLA

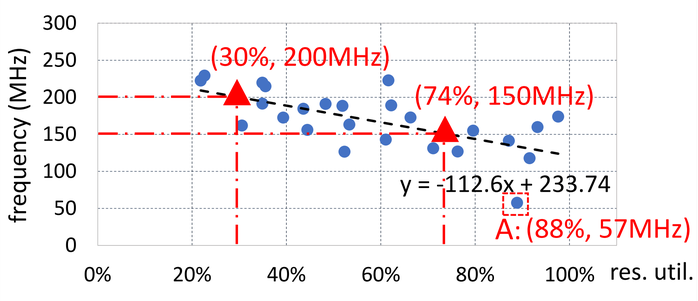
ICCM18Latte

Motivation

◆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 -> critical path delay
- Patterns are used in most, if not all, accelerators

Benchmark	Domain	Scatter	Gather	Broadcast	Reduce
AES	Encryption	✓⋆	✓⋆	✓	
FFT	Signal	✓	✓⋆		
GEMM	Algebra	✓	✓⋆	✓⋆	
KMP	String	✓		✓	✓⋆
NW	Bioinfo.	✓	✓		
SPMV	Algebra	✓	✓⋆	✓	
STENCIL	Image	✓	✓⋆	✓	
VITERBI	DP	✓	✓		

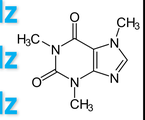
Checkmark ✓ represents the design has the pattern.
A star ⋆ represents that a critical path lies in the pattern.
For broadcast, GEMM uses bc_in_compute while others use bc_by_copy.

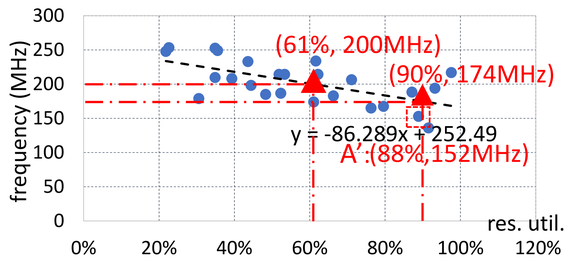
Ouch! Painful -> Latte

"Caffè" Latte

◆Latte boosts frequency

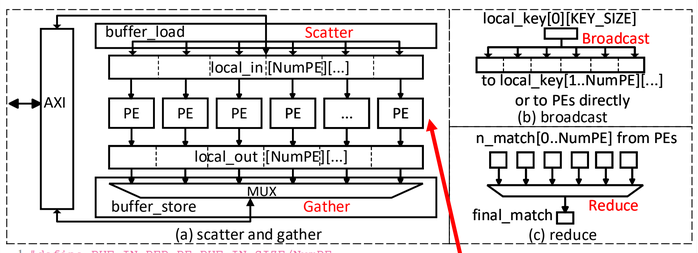
- Latte-optimized design improves timing of the baseline by 1.50x with only 3.2% LUT overhead
  - 30% chip resource, 200MHz -> 227MHz
  - 74% chip resource, 150MHz -> 189MHz
  - 90% chip resource, 132MHz -> 175MHz
  - Extreme case: FFT, 88%, 57MHz -> 152MHz





How does Latte work?

Common Practice Accelerator




```
1 #define BUF_IN_PER_PE BUF_IN_SIZE/NumPE
2 #define BUF_OUT_PER_PE BUF_OUT_SIZE/NumPE
3 void app(int data_size,
4 int *global_in, int *global_out) {
5  // local buffer
6  int local_in_A[NumPE][BUF_IN_PER_PE];
7  int local_in_B[NumPE][BUF_IN_PER_PE];
8  int local_out_A[NumPE][BUF_OUT_PER_PE],
9  int local_out_B[NumPE][BUF_OUT_PER_PE];
10 for (int i = 0; i < data_size/BUF_IN_SIZE+1; i++) {
11   // double buffer
12   if (i % 2 == 0) {
13     buffer_load(local_in_A, global_in+i*BUF_IN_SIZE);
14     buffer_compute(local_in_B, local_out_B);
15     buffer_store(global_out+i*BUF_OUT_SIZE, local_out_A);
16   }
17   else {
18     buffer_load(local_in_B, global_in+i*BUF_IN_SIZE);
19     buffer_compute(local_in_A, local_out_A);
20     buffer_store(global_out+i*BUF_OUT_SIZE, local_out_B);
21 } }
22 void buffer_compute(int** local_in, int** local_out) {
23  for (int i=0; i<NumPE; i++) {
24  #pragma HLS unroll
25    // kernel replication
26    PE_kernel(local_in[i], local_out[i]);}
27 }
```

NumPE kernels

memory partitioning

loop unrolling

```
1 void buffer_load(int local_in[NumPE][], int* global_in) {
2  // memcpy(local_in, global_in, BUF_IN_SIZE);// burst read
3  for(int i = 0; i < NumPE; i++)
4    for(int j = 0; j < BUF_IN_PER_PE; j++) { // for each PE
5    #pragma HLS pipeline II = 1
6    local_in[i][j] = global_in[i*BUF_IN_PER_PE + j];}
7 }
8 void buffer_store(int* global_out, int local_out[NumPE][]) {
9  memcpy(global_out, local_out, BUF_OUT_SIZE);
10 // for loop (similar to buffer_load, not shown)
11 }
```

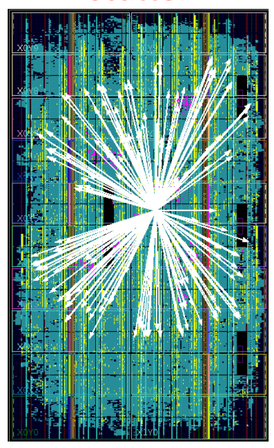
memcpy():
DRAM interface <-> on-chip bram buffers

Chip Layout Look Like?

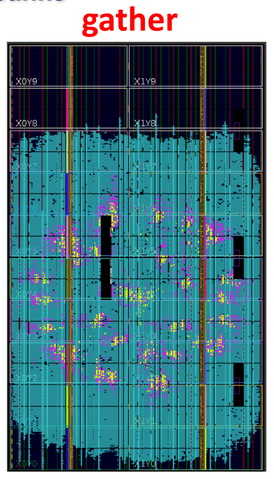
◆Scattered distribution of local buffers (PEs)

- HLS optimistically estimate memcpy() function wire delay without considering the locality of partitioned local buffer banks

scatter



gather



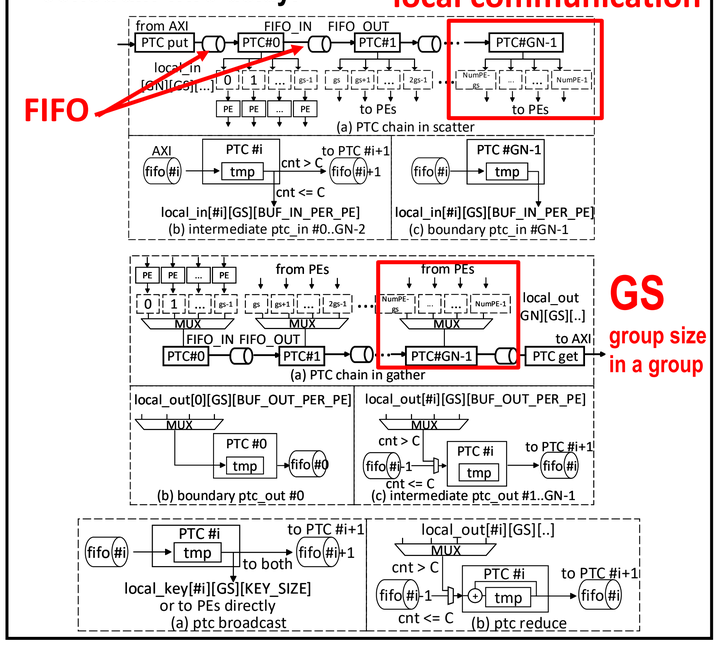
Broadcast and reduce are similar to the patterns of scatter and gather;

Pipelined Transfer Controller (PTC) in Latte

PTCs are chained in linear fashion through FIFOs. Each PTC connects to a local set of buffers to constrain wire delay.

local communication

FIFO



GS
group size in a group

Latte in HLS

260 lines of code (LOC) to manually implement Latte in HLS, 10x more than baseline code

PTC in Scatter Code snippet

```
1 #include <hls_stream.h>
2 int local_in[GN][GS][BUF_IN_PER_PE]; // redef.
3 void PTC_load(
4   int local_in[GN][GS][], int* global_in) {
5 #pragma HLS dataflow
6   hls::stream<int> fifo[GN];// FIFOs, in Fig. 7a
7   ptc_put(global_in, fifo[0]);
8   for(int i = 0; i < GN-2; i++){
9     ptc_in(fifo[i], fifo[i+1], local_in[i], GN-1-i);}
10  ptc_in(fifo[GN-1], local_in[GN-1]);
11 }
12 void ptc_put(int* global_in, stream<int> &fifo){
13   for (int i=0; i<NumPE; i++)
14     for(int j = 0; j < BUF_IN_PER_PE; j++){
15 #pragma HLS pipeline
16       fifo << global_buf[i*BUF_IN_PER_PE+j];}
17 }
18 void ptc_in( // #0..GN-2 ptc_in, in Fig. 7b
19 stream<int>&fifo_in, stream<int> &fifo_out,
20 int local_set[GS][BUF_IN_PER_PE], int todo){
21   int i, j, k; int tmp;
22   for(i= 0; i < GS; i++){ // to local first
23     for(j = 0; j < BUF_IN_PER_PE; j++){
24       tmp = fifo_in.read();
25       local_set[i][j] = tmp;
26   } }
27   for(k=0; k < todo; k++) // to next ptc
28   for(i= 0; i < GS; i++){
29     for(j = 0; j < BUF_IN_PER_PE; j++){
30       tmp = fifo_in.read();
31       fifo_out.write(tmp);
32   } }
33 }
```

Can we do better? Yes!

Latte Automation

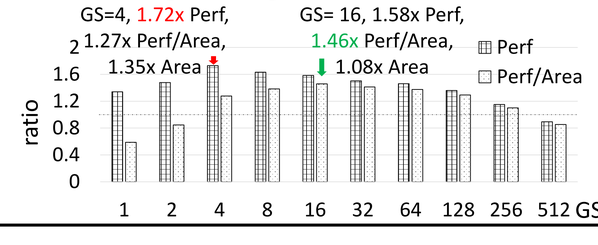
◆Latte provides semiautomatic framework

- Users insert simple Latte pragmas into user-written HLS kernel
- 260 -> 10 LOC, pragma indicates 1. buffers 2. pattern to be optimized

pattern — variable that needs optimization

```
1 #pragma latte scatter var="local_in_B"
2 int local_in_B[NumPE][BUF_IN_PER_PE];
```

- Different PTC group sizes (#GS) are launched
- Best design configuration is picked

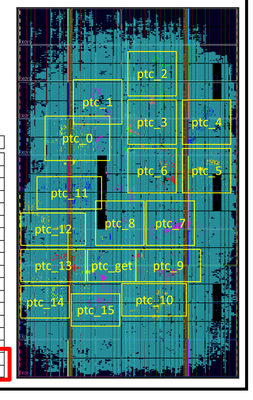


Latte is "Light Roast"

◆Latte improves freq. with negligible overhead

- 1.50x 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

linear layout of PTCs in gather pattern in FFT with 64 PEs and 16 PTCs



Baseline vs Latte

Bench.	type	N / GS	LUT	FF	DSP	BRAM	Freq.
AES	ori.	320 /	50.4%	17.3%	0.1%	76.3%	127
	latte	7 32	1.017	1.009	1	1	165, 1.30x
FFT	ori.	64 /	50.5%	23.2%	88.9%	78.5%	57
	latte	7 4	1.027	1.056	1	1	152, 2.66x
GEMM	ori.	512 /	37.8%	29.6%	71.1%	69.7%	131
	latte	7 16	1.044	0.962	1	1	207, 1.58x
KMP	ori.	96 /	5.0%	3.0%	0.2%	52.3%	126
	latte	7 24	1.045	1.174	1	1	195, 1.54x
NW	ori.	160 /	65.1%	50.7%	0.0%	78.2%	174
	latte	7 80	0.995	0.997	1	1	177, 1.02x
SPMV	ori.	48 /	19.1%	11.9%	18.9%	93.2%	160
	latte	7 6	1.029	1.037	1	1	192, 1.20x
STENCIL	ori.	64 /	12.9%	10.9%	48.1%	87.1%	141
	latte	7 16	1.094	1.139	1	1	188, 1.33x
VITERBI	ori.	192 /	72.0%	25.7%	10.8%	39.3%	155
	latte	7 12	1.008	1.031	1	1	168, 1.08x
Average	ori.	/	NA	NA	NA	NA	120
	latte	/	1.032	1.051	1	1	181, 1.50x