

RippleFPGA: A Routability-Driven Placement for Large-Scale Heterogeneous FPGAs

Chak-Wa Pui, Gengjie Chen, Wing-Kai Chow, Ka-Chun Lam, Jian Kuang,
Peishan Tu, Hang Zhang, Evangeline F. Y. Young, Bei Yu
CSE Department, Chinese University of Hong Kong, Hong Kong

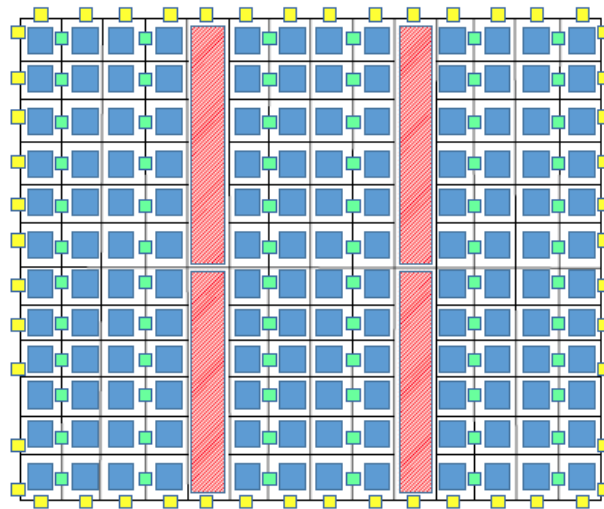
Speaker: Jordan, Chak-Wa Pui

Outline

- Background
- Problem Formulation
- Algorithms
- Experimental Results

Introduction

- As the scale of FPGA grows rapidly, routability becomes a major problem in FPGA placement
- The complex architecture of heterogeneous FPGAs yields more sophisticated placement techniques



Architecture sample of heterogeneous FPGA

Previous Works

- Three major categories
 - Simulated annealing based, e.g. VPR
 - Partitioning-based, e.g. [1]
 - Analytical approach, e.g. [2][3]
- Limitations of previous works
 - Very few of recent works considering routability
 - Previous works mainly consider routability in packing
 - Most of previous works on heterogeneous FPGAs pack logic elements into CLB and seldom change them after packing

[1] Timing-driven partitioning-based placement for island style FPGAs. TCAD2006

[2] Analytical placement for heterogeneous FPGAs. FPL2012

[3] An efficient and effective analytical placer for FPGAs. DAC2013.

Contributions

- A framework for heterogeneous FPGA flat placement
- Several methods are proposed to reduce routing congestion
 - Partitioning
 - Multi-stage congestion-driven global placement
 - Alignment-aware detailed placement

Problem Formulation

- Routability-driven FPGA placement
 - Given the netlist and architecture of an FPGA
 - Minimize: routed wirelength measured by VIVADO
 - Subject to: each logic element has no overlap, no violation to the architecture specific legalization rules

Overview of Our Algorithm

- Partitioning
- Packing
- Routability-Driven Global Placement
- Legalization
- Routability-Driven Detailed Placement

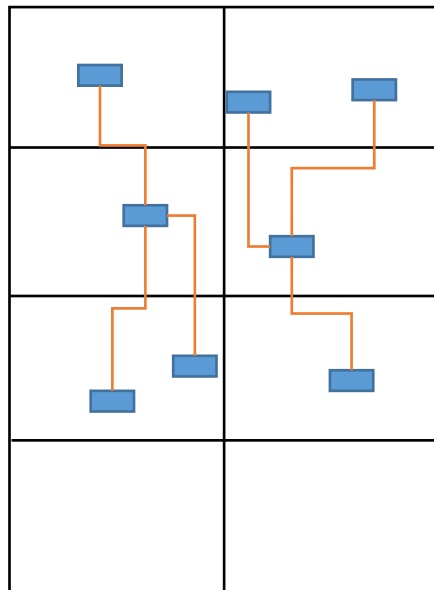
Overview of Our Algorithm

- Partitioning
- Packing
- Routability-Driven Global Placement
- Legalization
- Routability-Driven Detailed Placement

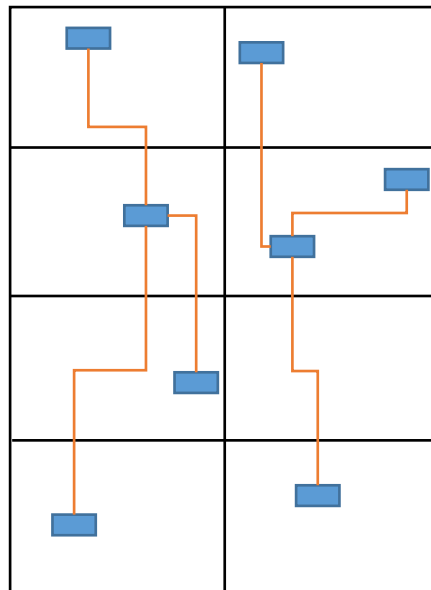
Partitioning

- Motivation

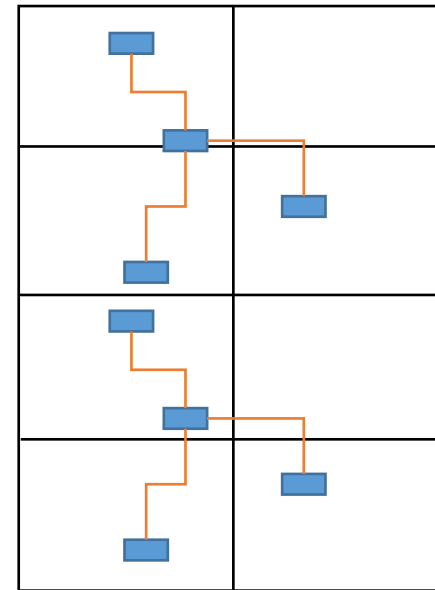
- Unbalance between width and height of the chip
- Cannot be resolved by spreading but by reallocating



(a)origin



(b)spreading

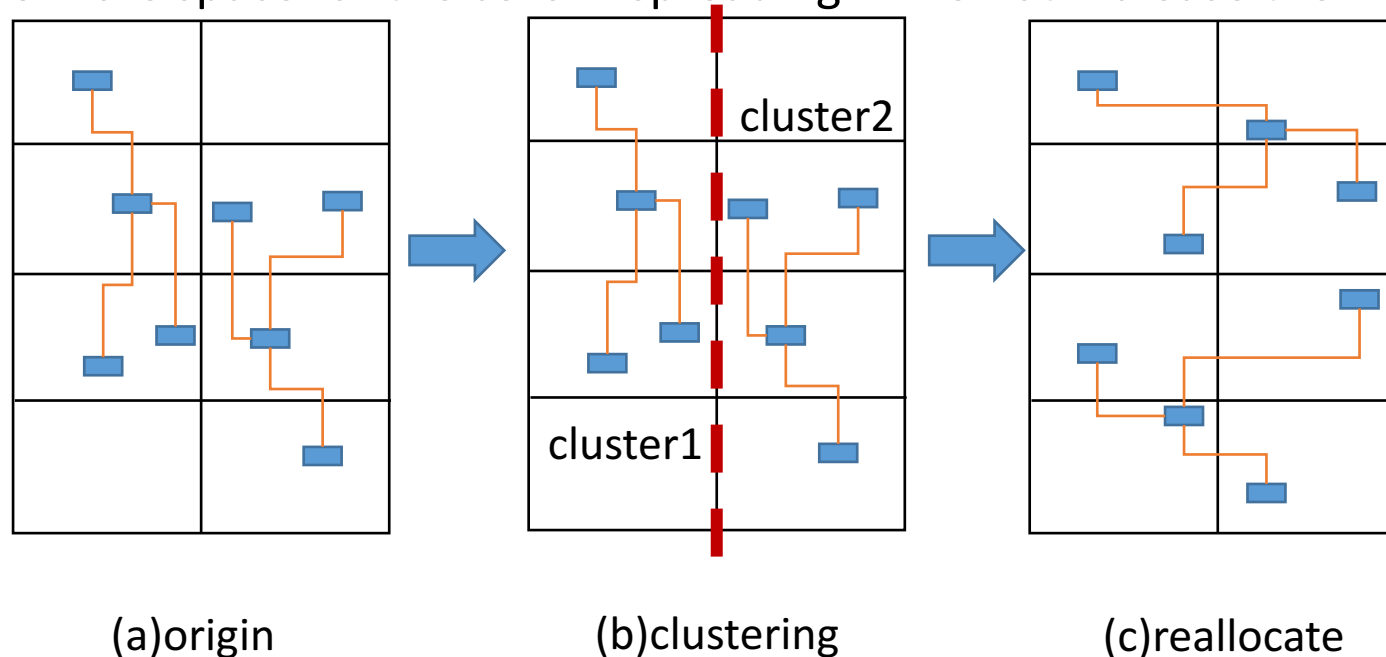


(c)reallocation

Comparison of different methods in solving congestion

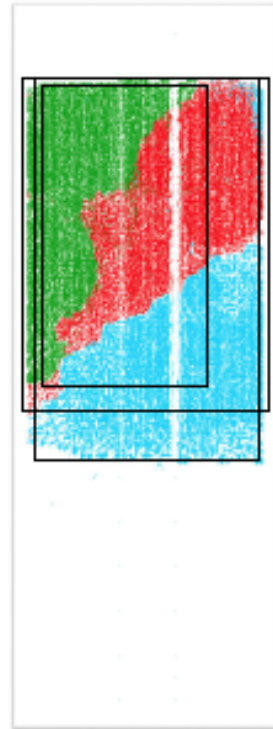
Partitioning

- Solution:
 - Partition the circuit into sub-circuits using recursive bi-partitioning
 - Cluster size less than 25% of #cells, cut size less than 5% of #net
 - Reallocate the cells across the chip as sparse as we can
 - Maintain relative order of clusters and cells inside the same cluster
 - Give more space for the cells in spreading while not increase the HPWL too much

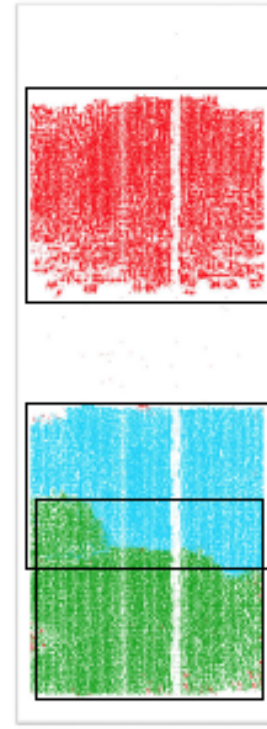


Partitioning

- Effect on real test case



(a)w/o partitioning



(b)with partitioning

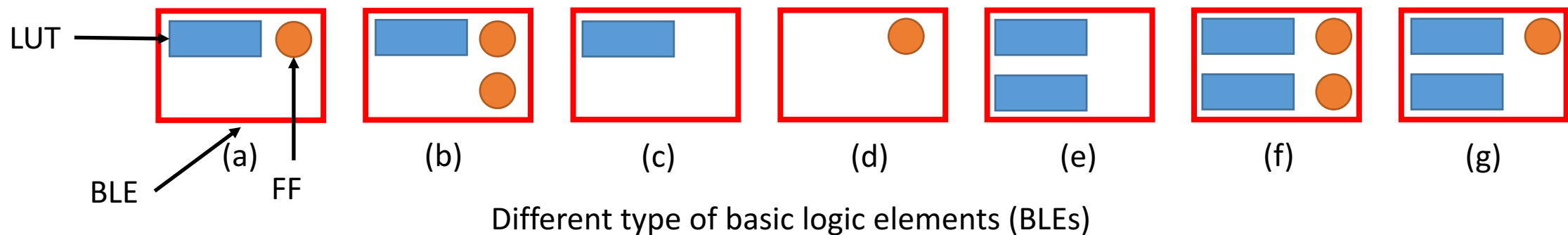
Comparison of spreading result w/o and with partitioning

Overview of Our Algorithm

- Partitioning
- **Packing**
- Routability-Driven Global Placement
- Legalization
- Routability-Driven Detailed Placement

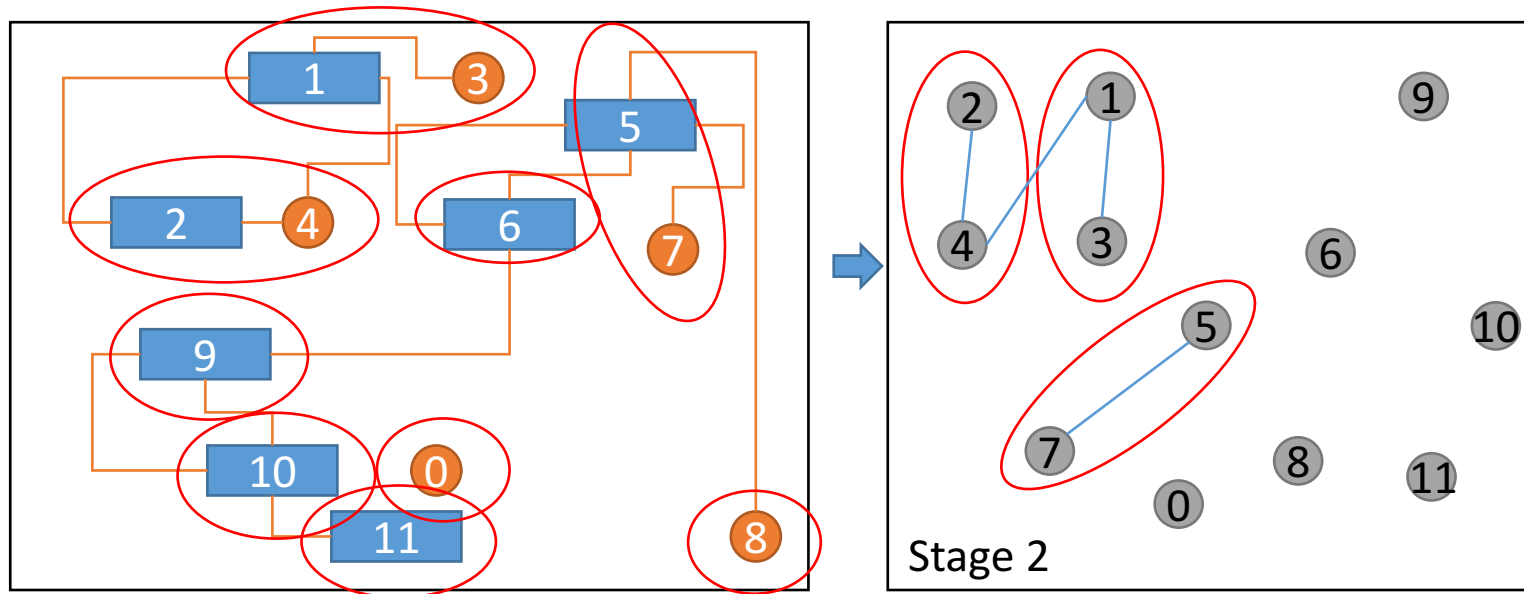
Packing

1. Short global placement
2. Forming basic logic elements(BLEs) that consist of only one LUT and at least one FFs
3. Let the remaining LUTs and FFs be BLEs of itself only
4. Merging two BLEs into one if their LUTs have many connections



Packing

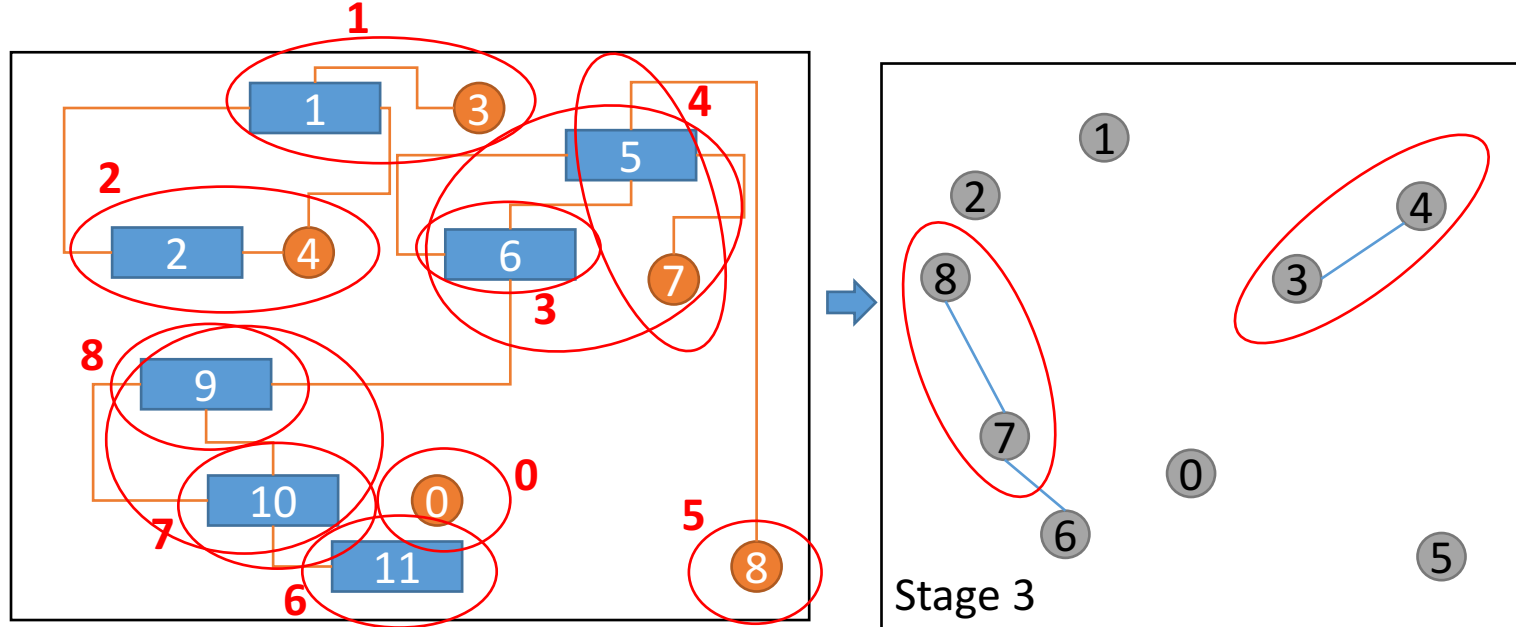
- Use maximum weight matching in stage2, weight proportional to distance, only connected LUTs and FFs have edges
- In stage3, let the remaining LUTs and FFs be BLEs of itself only



How we do packing in stage2,3

Packing

- Use maximum weight matching in stage4, weight proportional to distance and connections between the LUTs of the vertices



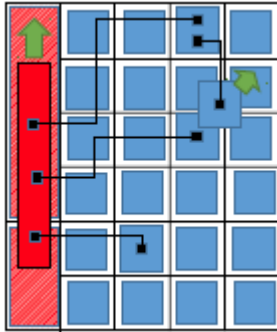
How we do packing in stage4

Overview of Our Algorithm

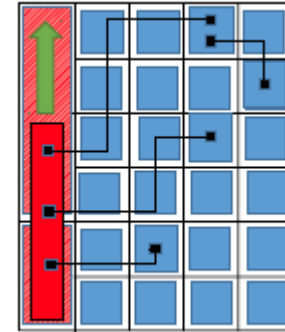
- Partitioning
- Packing
- **Routability-Driven Global Placement**
- Legalization
- Routability-Driven Detailed Placement

Global Placement

- Quadratic placement based on Ripple
- Three-stage Optimization
 - First two stages, optimize wirelength, fix DSP/RAM to their legal position after stage 1
 - Legalizing DSP/RAM disturbs the global placement result a lot



Same displacement, difference in HPWL



Large displacement

- The third stage optimize routability using inflation (Accumulative)

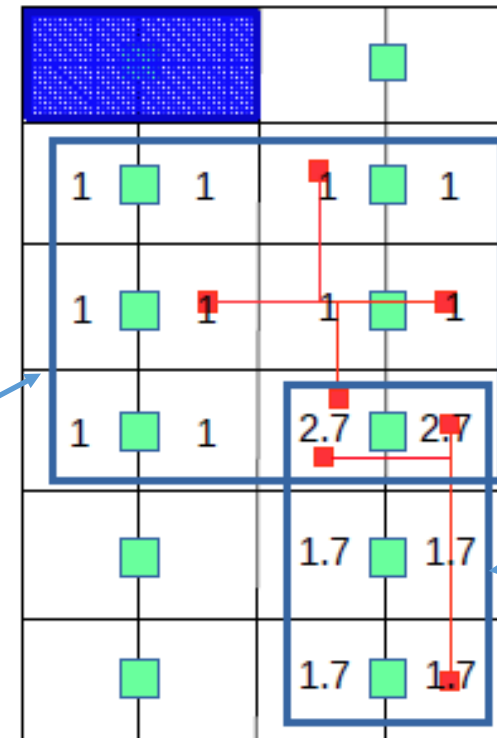
Global Placement

- Routing congestion estimation
 - Probabilistic model
 - Consider both bounding box and HPWL

$$\text{Cong}_{s_i} = \sum_{m \in N_i} \frac{W_m \cdot \text{HPWL}_m}{\#G\text{-Cells covered by net } m}$$

$$W_m \cdot \text{HPWL}_m = 6$$

$$\#G\text{-Cell} = 6$$

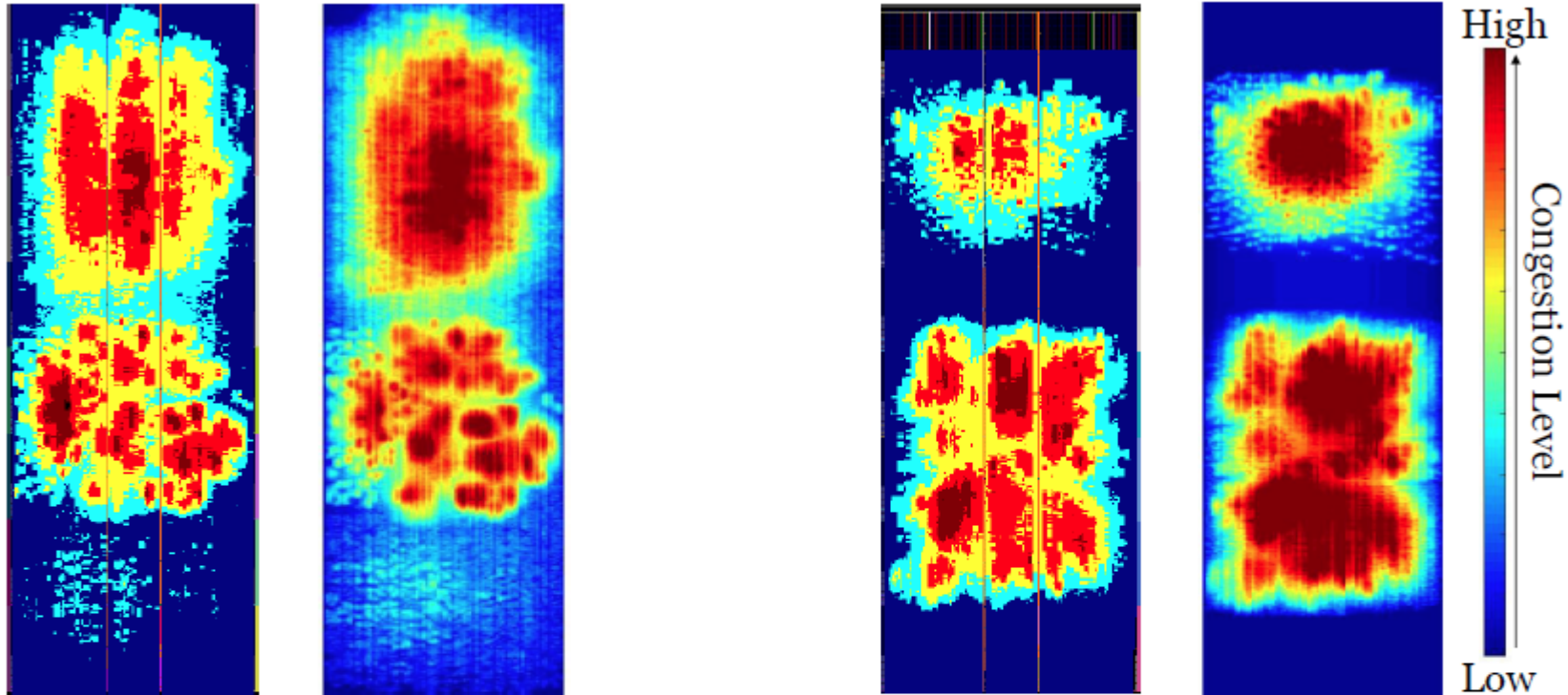


$$W_m \cdot \text{HPWL}_m = 5$$

$$\#G\text{-Cell} = 3$$

G-Cell
 Switch Box
 Pin
 BB

Routing congestion estimation



(a)VIVADO

(b)Ours

(c)VIVADO

(d)Ours

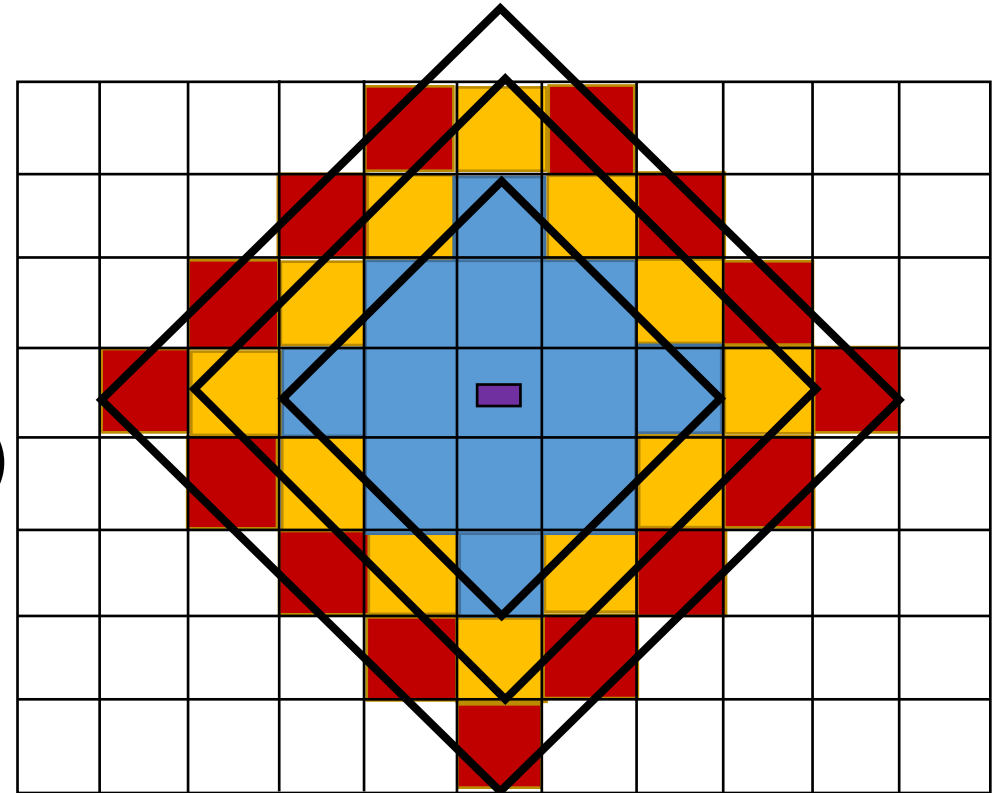
Comparison of the routing congestion estimation obtained by VIVADO and us

Overview of Our Algorithm

- Partitioning
- Packing
- Routability-Driven Global Placement
- **Legalization**
- Routability-Driven Detailed Placement

Legalization

- Greedy window-based cell by cell legalization process
 - Start from a small window
 - Sites inside a window are considered to have same displacement
 - Sort the sites by an objective function (HPWL)
 - If cannot be legalized, slowly increase the window size until it's legalized
 - Keep BLEs intact unless cannot be legalized

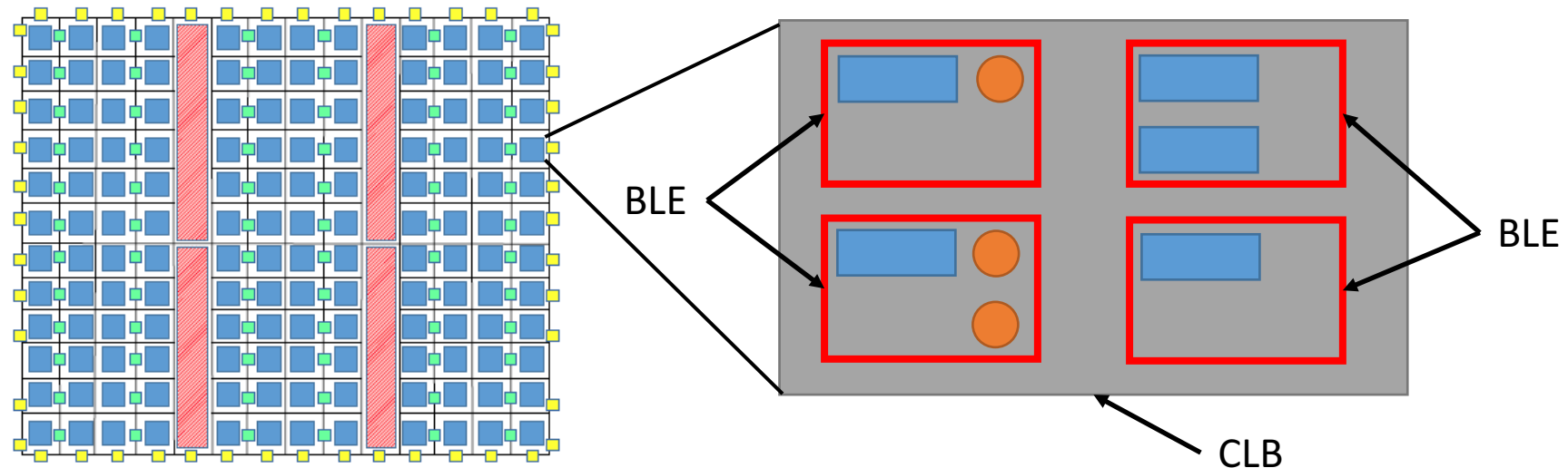


Overview of Our Algorithm

- Partitioning
- Packing
- Routability-Driven Global Placement
- Legalization
- **Routability-Driven Detailed Placement**

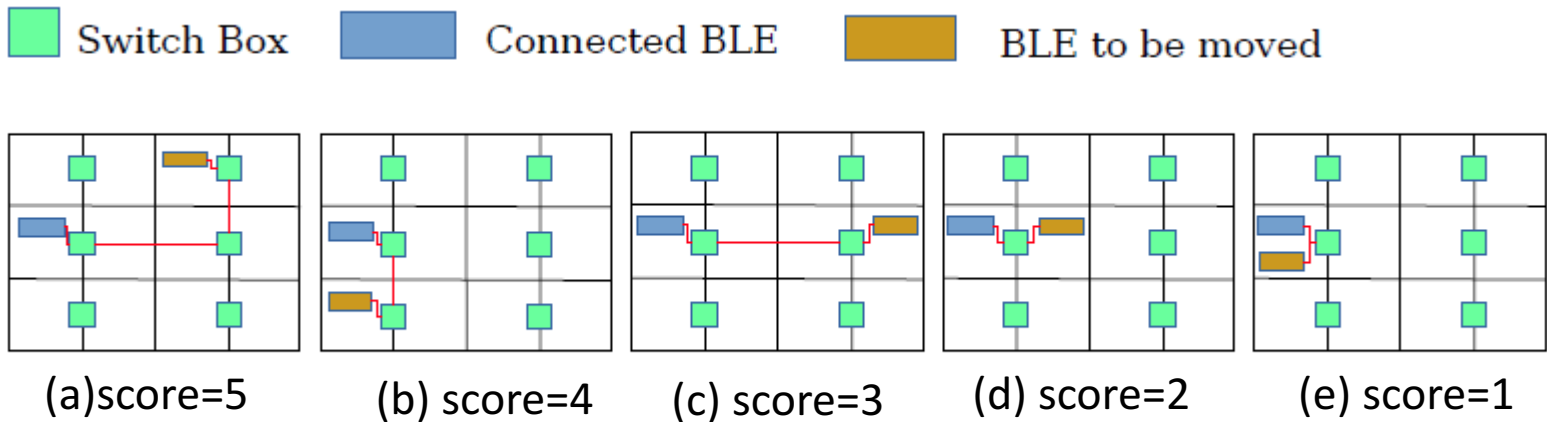
Detailed Placement

- Move to optimal region to optimize HPWL
 - In both **BLE** level and **CLB** level
 - In **CLB** level, if the site is occupied, swap the cells if the HPWL does not increase
 - In **BLE** level, the BLE can only be moved to a slice if it won't violate legalization rules



Detailed Placement

- If already in optimal region, move to site to optimize alignment(BLE level).
 - Compute the score of each site by assuming the cell is moved to there and get the alignment score by considering all related nets
 - Sort the candidate sites by their alignment score, try to move the cell to a site with smaller score



Comparison of alignment of different placement

Experimental Result

Design	Ours			1st Place			2nd Place			3rd Place		
	WL	Time(s)	Normalized WL	WL	Time(s)	Normalized WL	WL	Time(s)	Normalized WL	WL	Time(s)	Normalized WL
FPGA-1	362563	74	1	-	-	-	379932	118	1.048	581975	97	1.605
FPGA-2	677563	167	1	677877	435	1.000	679878	208	1.003	1046859	191	1.545
FPGA-3	3617466	1037	1	3223042	1527	0.891	3660659	1159	1.012	5029157	862	1.390
FPGA-4	6037293	621	1	5628519	1257	0.932	6497023	1449	1.076	7247233	889	1.200
FPGA-5	10455204	1012	1	10264769	1266	0.982	-	-	-	-	-	-
FPGA-6	6960037	2772	1	6330179	2920	0.910	7008525	4166	1.007	6822707	8613	0.980
FPGA-7	10248020	2170	1	10236827	2703	0.999	10415871	4572	1.016	10973376	9196	1.071
FPGA-8	8874454	1426	1	8384338	2645	0.945	8986361	2942	1.013	12299898	2741	1.386
FPGA-9	12954350	2683	1	-	-	-	13908997	5833	1.074	-	-	-
FPGA-10	8564363	5555	1	-	-	-	-	-	-	-	-	-
FPGA-11	11226088	3636	1	11091383	3227	0.988	11713479	7331	1.043	-	-	-
FPGA-12	8928528	9748	1	9021768	4539	1.010	-	-	-	-	-	-

Comparison of wirelength and runtime of our placer and the top3 winners

Experimental Result

Design	Raw		With congestion-driven GP		With partitioning		With Both	
	WL	Normalized WL	WL	Normalized WL	WL	Normalized WL	WL	Normalized WL
FPGA-1	362563	1.000	364143	1.004	378029	1.043	377883	1.042255
FPGA-2	681418	1.000	677563	0.994	696417	1.022	689360	1.011655
FPGA-3	4027586	1.000	3999517	0.993	3645846	0.905	3617466	0.898172
FPGA-4	6037293	1.000	6087199	1.008	6265158	1.038	6357766	1.053082
FPGA-5	-	-	-	-	-	-	10455204	-
FPGA-6	7801736	1.000	7723476	0.990	7016684	0.899	6960037	0.892114
FPGA-7	10248020	1.000	10615672	1.036	10338763	1.009	10580828	1.032475
FPGA-8	9171179	1.000	9392039	1.024	8874454	0.968	9013564	0.982814
FPGA-9	12954350	1.000	13437554	-	-	-	13834692	1.067957
FPGA-10	-	-	10372369	-	8782789	-	8564363	-
FPGA-11	-	-	-	-	11226088	-	11688504	-
FPGA-12	-	-	10286583	-	-	-	8928528	-

Comparison of wirelength of different method used in our placer

Thanks