Pin Assignment Optimization for Multi-2.5D FPGA-based Systems

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Results
- Conclusions

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Results
- Conclusions

Introduction

- Multi-FPGA systems are widely used for
  - Logic emulation
  - Rapid prototyping of large designs
  - Reconfigurable custom computing platforms

Introduction

- A multi-FPGA system connected through
  - Direct hardwired connections
  - Programmable interconnection network
    - Consist of one or more FPICs

- The available pin counts of the FPGAs limit the utilization of FPGA logic resources in a mulit-FPGA system.
  - Solution : time division-multiplexing (TDM)

Introduction

- 2.5D FPGA

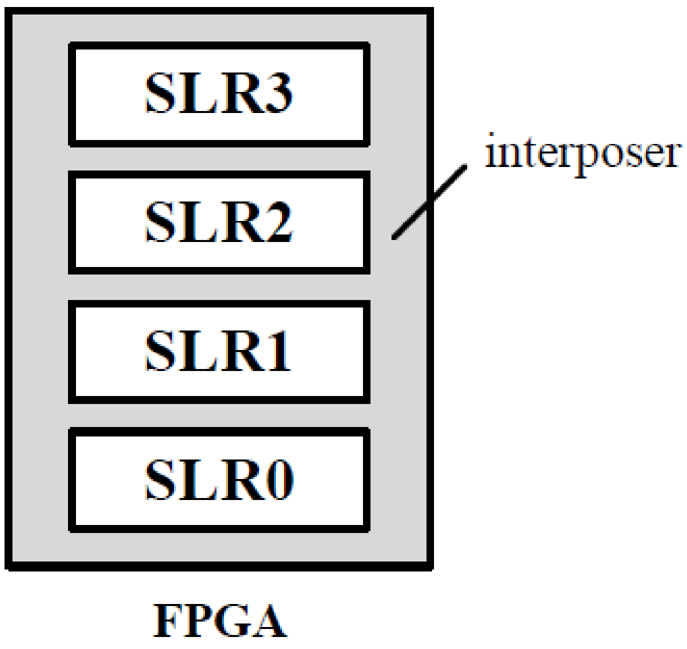
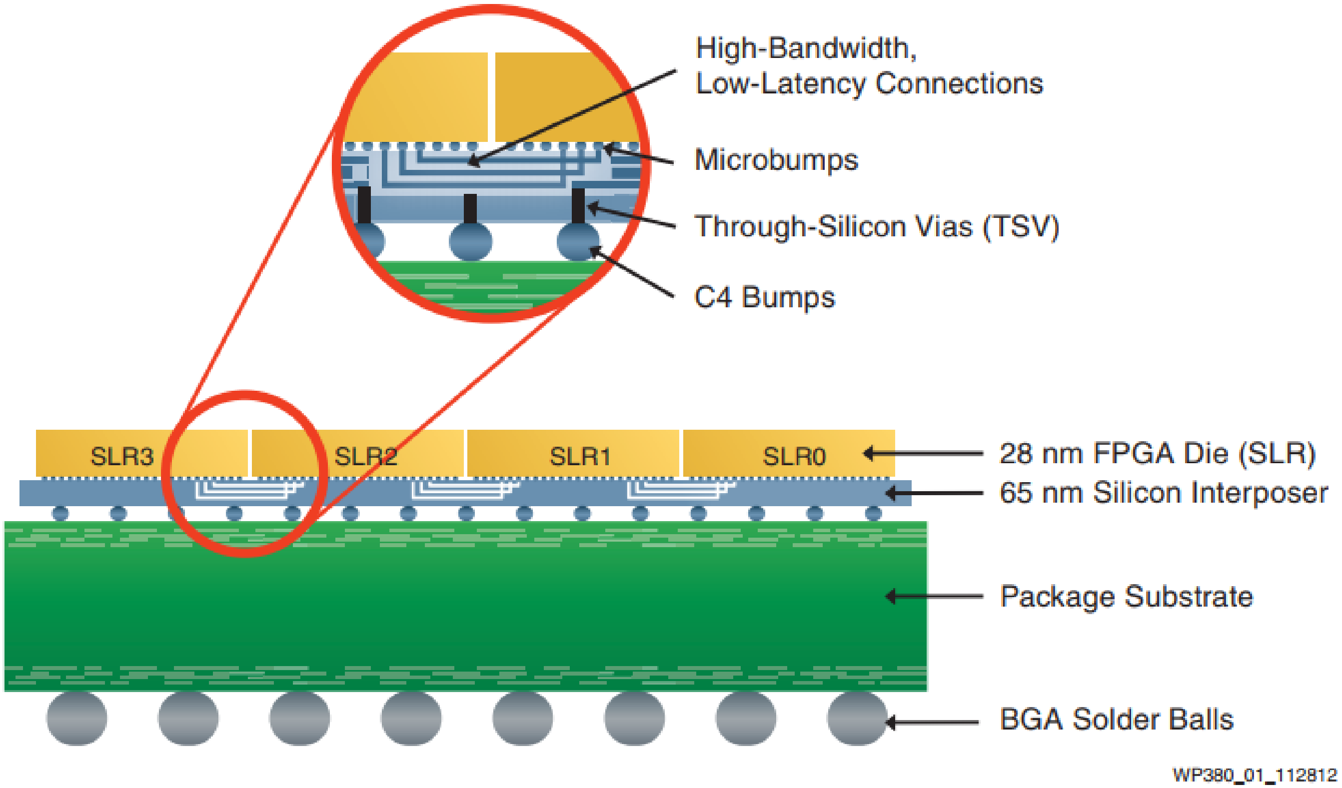


Figure 1: Structure of a commercial 2.5D FPGA.

Introduction

➢ 2.5D FPGA



Introduction

- Interconnect resources between dies
  - the amount is less than that within a die
  - there is increased delay to cross the interposer

Introduction

- [6] : reducing the SLR crossings
  - routability
  - circuit speed

- [6] focused on reducing the total SLR crossings by SLR partitioning in a stand alone 2.5D FPGA.

Introduction

- The I/O signal counts per FPGA is enormous.

- I/O pin assignment can have a big impact on the overall SLR crossings

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Results
- Conclusions

Preliminaries

- multi-FPGA system
  - multiplexed hardwired inter-FPGA connections
  - 2.5D FPGAs

Preliminaries

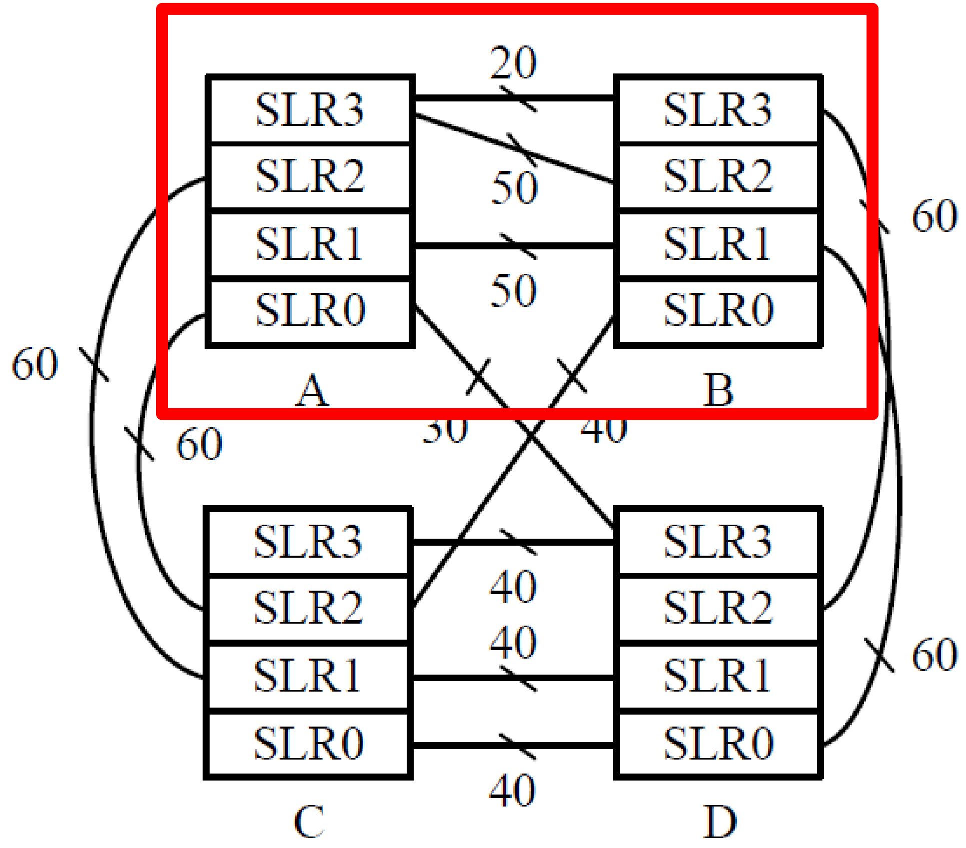


Figure 2: A multi-2.5D FPGA system.

Preliminaries

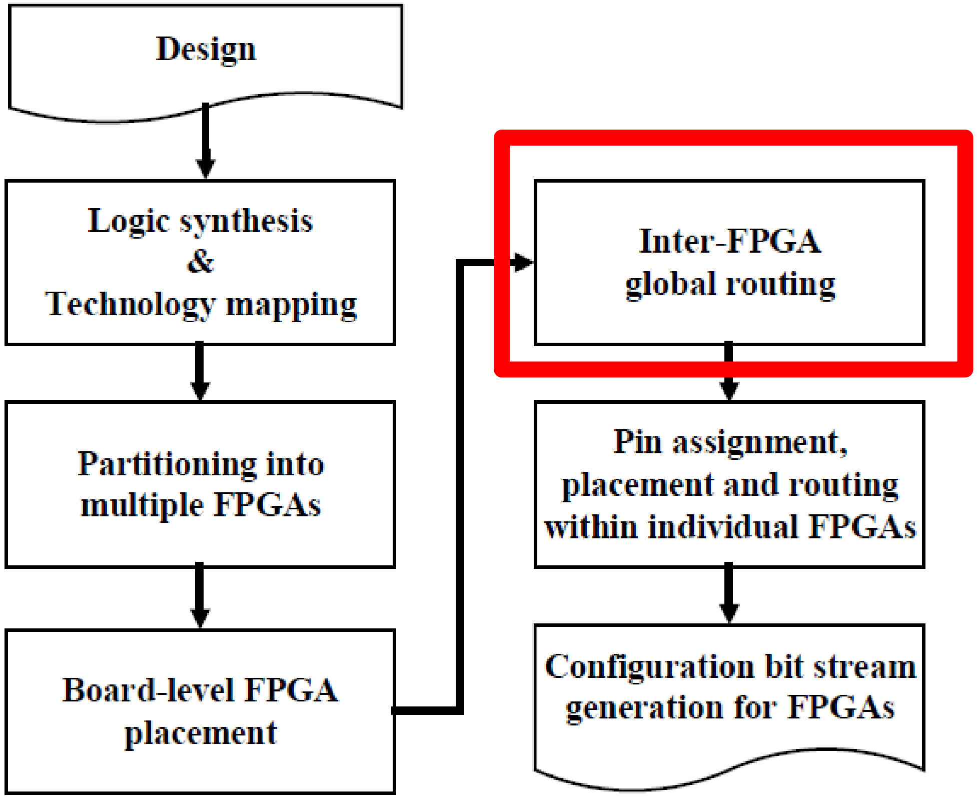
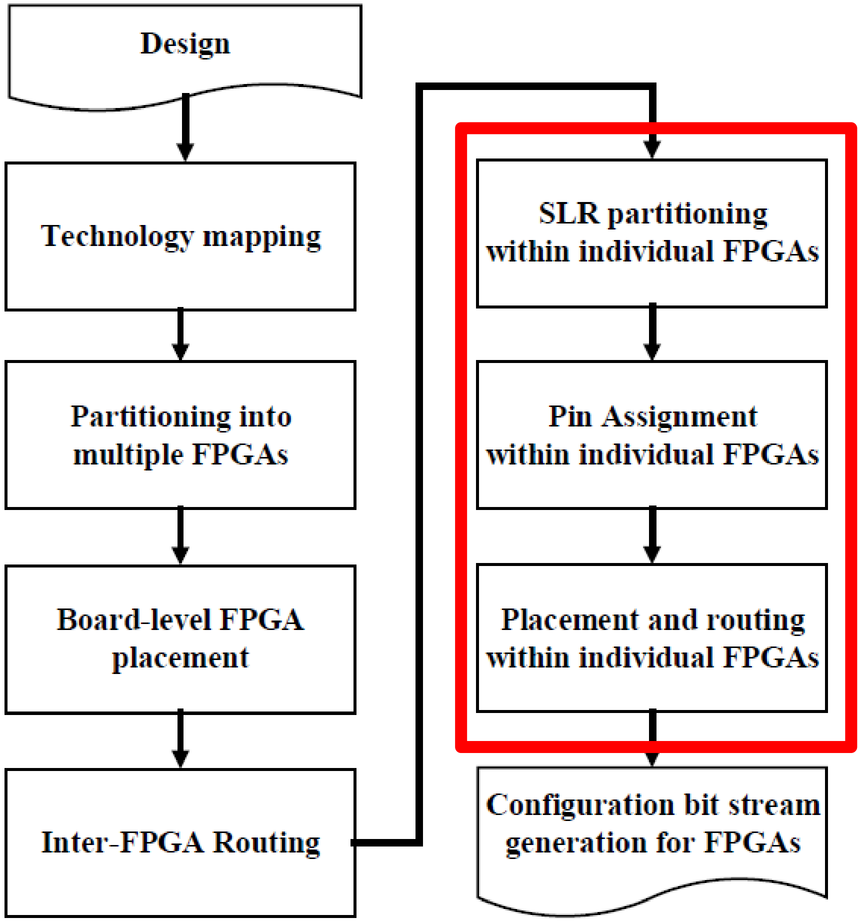


Figure 3: A typical compilation flow for a multi-FPGA system.

Preliminaries



Preliminaries

- SLR partitioning and pin assignment results of a FPGA are propagated to the FPGA P&R tool

SLR-Aware Pin Assignment

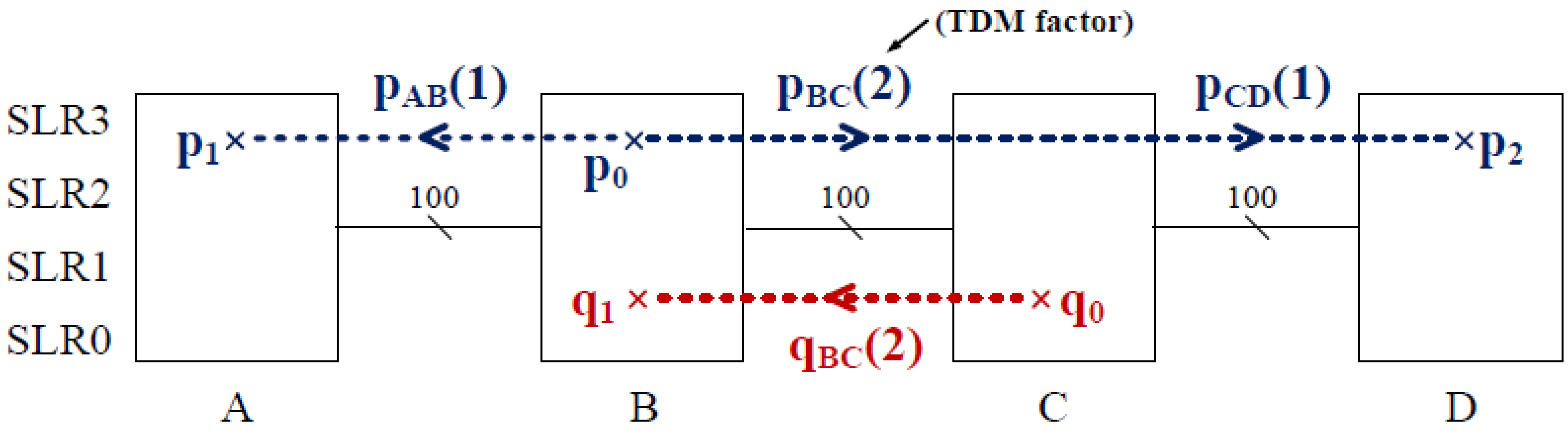


Figure 4: A global routing result. Net p induces three inter-FPGA 2-pin subnets p_{AB}, p_{BC}, and p_{CD}. Net q induces one inter-FPGA 2-pin subnet q_{BC}.

SLR-Aware Pin Assignment

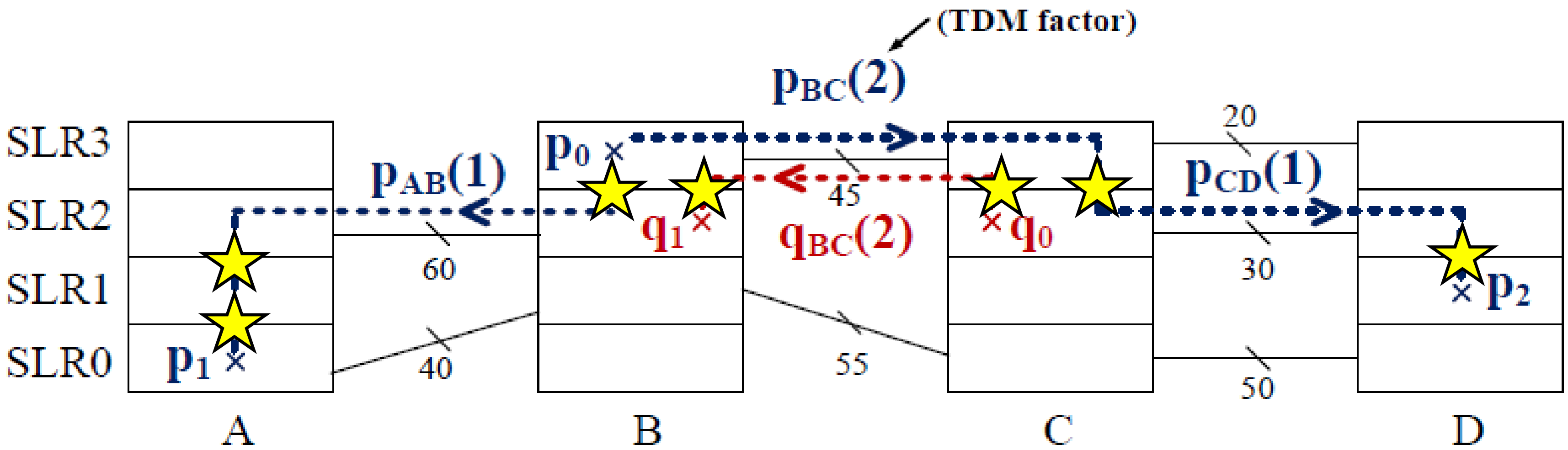


Figure 5: A feasible pin assignment with 7 SLR crossings.

SLR-Aware Pin Assignment

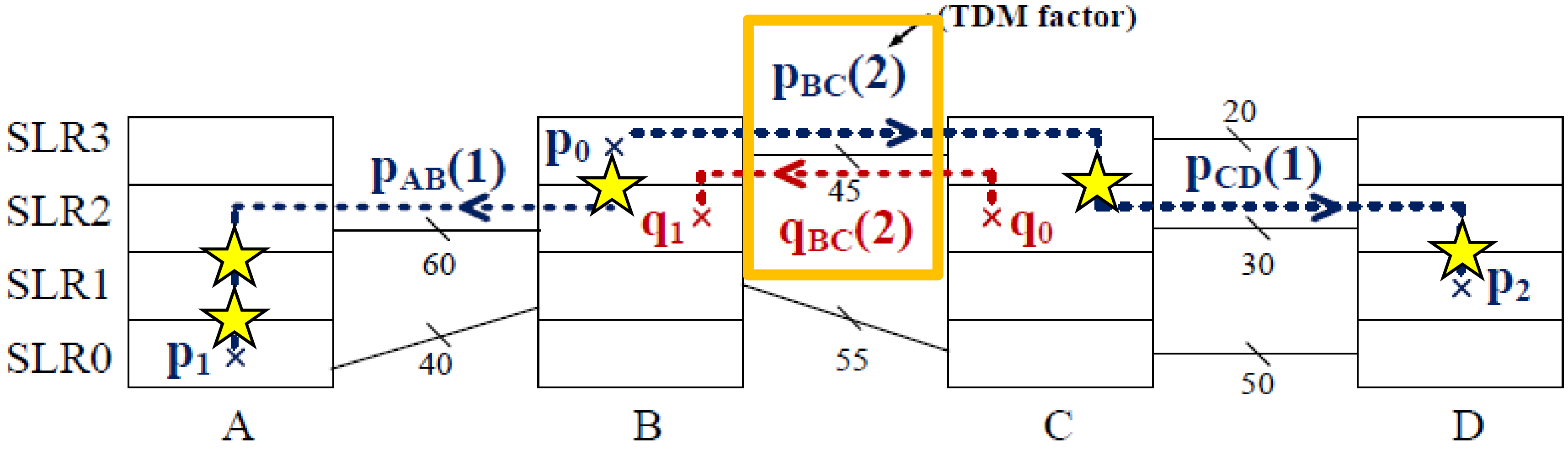


Figure 5: A feasible pin assignment with 7 SLR crossings.

SLR-Aware Pin Assignment

- Given :
  - Set of 2-pin inter-FPGA subnets
    - TDM factors
    - Directions
    - which originate from the same net

- Goal :
  - Assign all these 2-pin subnets to physical wires and pins to minimize the total SLR crossings subject to the constraints on the TDM factors and directions.

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Results
- Conclusions

Algorithm

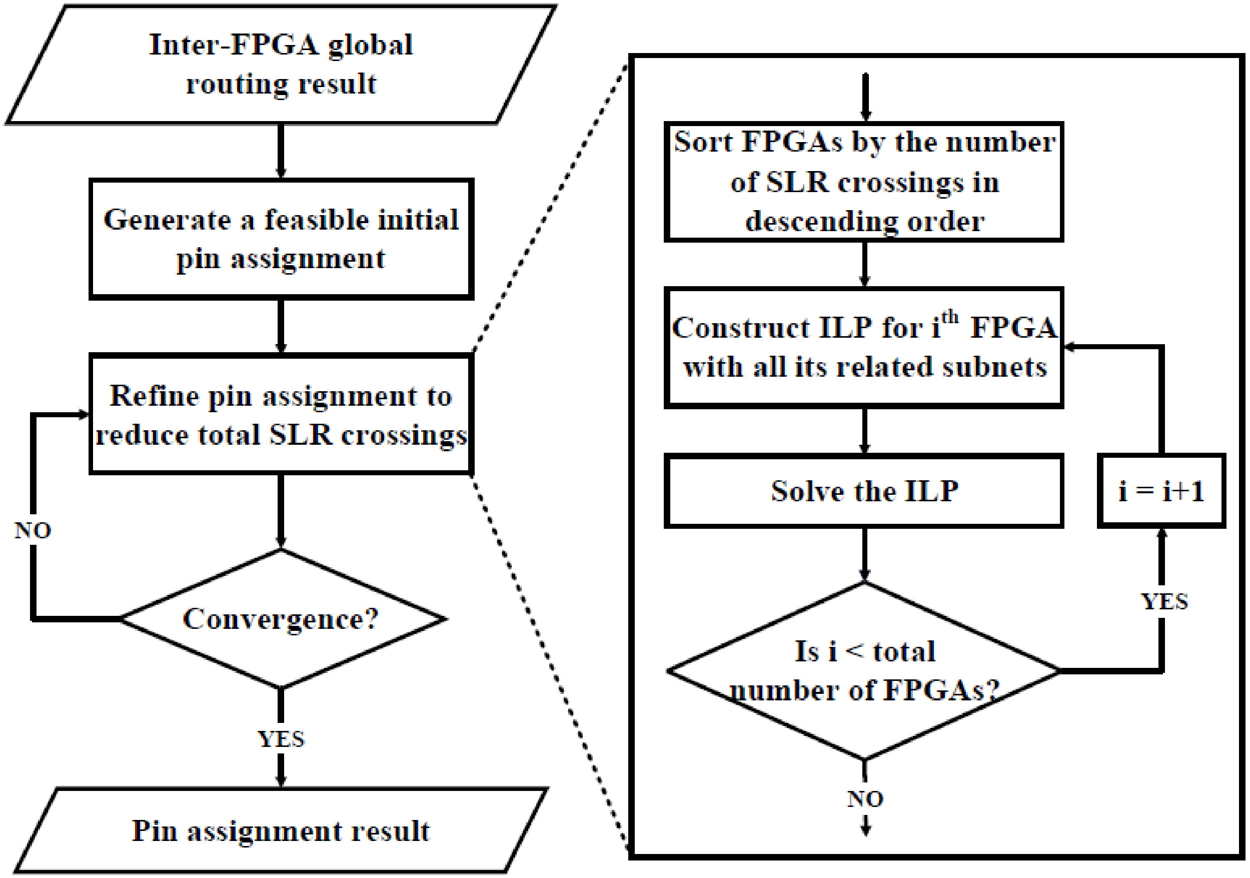
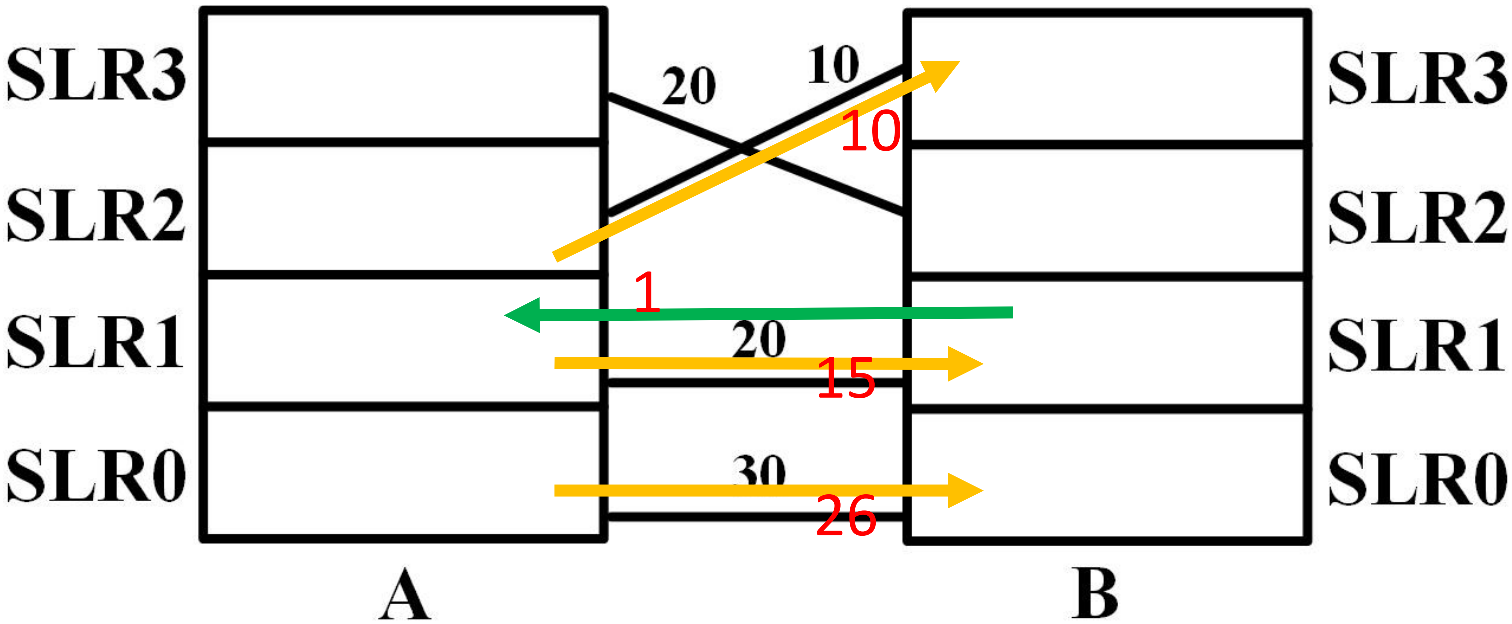


Figure 3.1: Overview of our pin assignment optimization algorithm.

Initial Feasible Pin Assignment Generation



= 5

TDM = 10

Required physical wires = $\lceil 5/10 \rceil = 1$

= 101

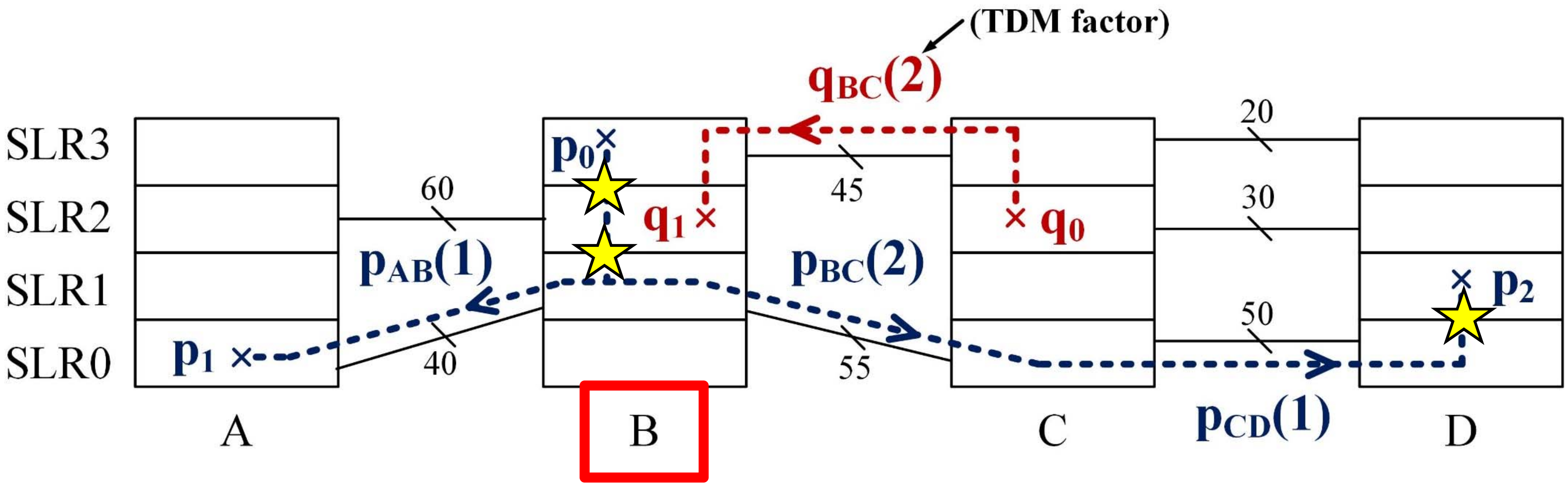
TDM = 2

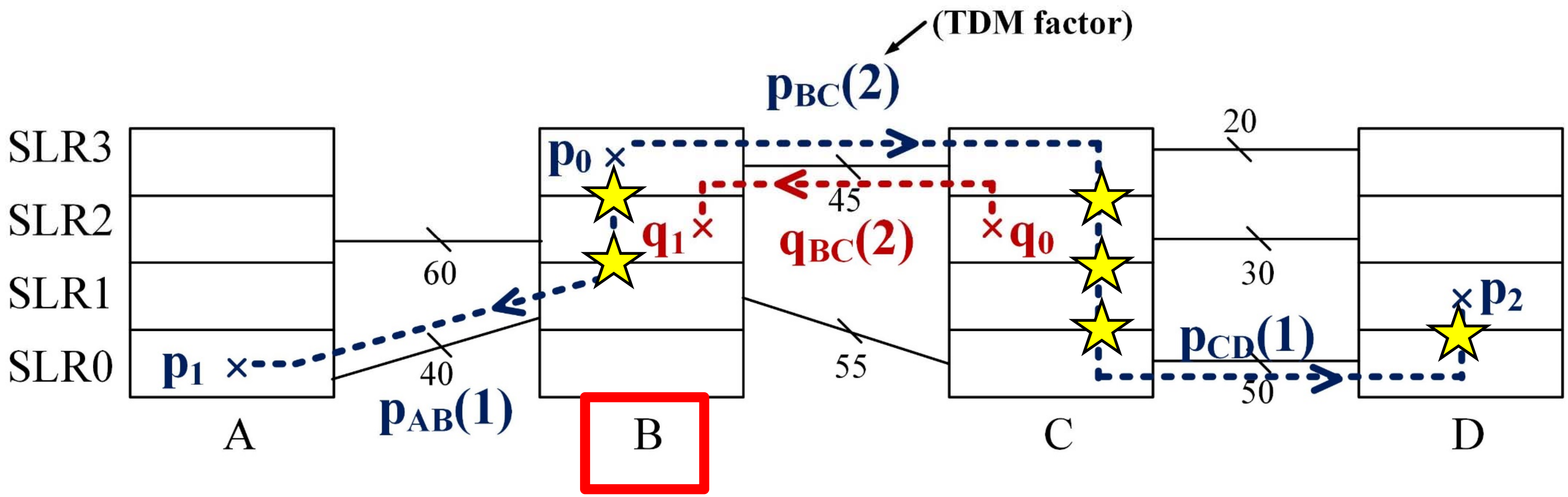
Required physical wires = $\lceil 101/2 \rceil = 51$

Pin Assignment Refinement

$$minimize$$

$$\sum_{\alpha_u} \sum_{m \in M_{\alpha_u}} COST_m * y_m$$


(TDM factor)
qBC(2)
SLR3
SLR2
SLR1
SLR0
p0
q1
q0
p1
p2
pAB(1)
pBC(2)
pCD(1)
60
40
45
55
20
30
50
A
B
C
D


(TDM factor)
pBC(2)
SLR3
SLR2
SLR1
SLR0
p0
q1
q0
qBC(2)
p1
p2
pAB(1)
pCD(1)
60
40
45
55
20
30
50
A
B
C
D

Pin Assignment Refinement

$$\sum_{\forall t, t \in TYPES_{uv}} n^t_{G_{u_i v_j}} = |G_{u_i v_j}|, \; \forall neighbor\ v\ of\ u,\ \forall i, j \tag{1}$$

$$\sum_{\forall \alpha_{uv} : TYPE_{\alpha_{uv}} = t} x_{\alpha_{uv} G_{u_i v_j}} \leq TDM_t * n^t_{G_{u_i v_j}}, \quad \forall neighbor\ v\ of\ u,\ \forall i, j,\ \forall t \in TYPES_{uv} \tag{2}$$

$$\sum_{G_{u_i v_j}} x_{\alpha_{uv} G_{u_i v_j}} = 1, \; \forall neighbor\ v\ of\ u,\ \forall \alpha_{uv} \tag{3}$$

$$n^t_{G_{u_i v_j}} \in \mathbb{N}, \; \forall neighbor\ v\ of\ u,\ \forall i, j,\ \forall t,\ t \in TYPES_{uv} \tag{4}$$

Pin Assignment Refinement

$$\sum_{m \in M_{\alpha_u}} y_m = 1,\ \forall \alpha_u \tag{5}$$

$$x_{\alpha_{uv} G_{u_i v_j}} = \sum_{m \in M_{\alpha_u} s.t. m[\alpha_{uv}] = G_{u_i v_j}} y_m,\ \forall \alpha_{uv},\ \forall i, j \tag{6}$$

$$y_m = 0\ or\ 1,\ \forall m \tag{7}$$

Outline

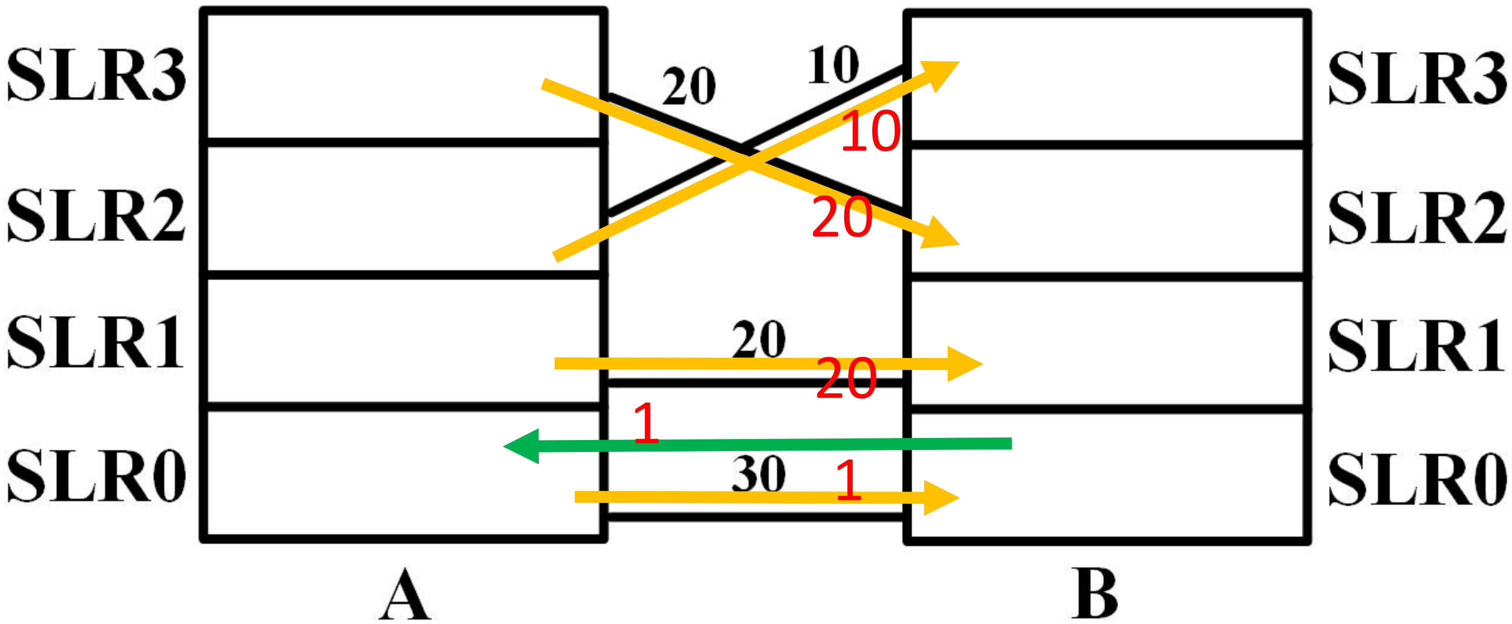
- Introduction
- Preliminaries
- Algorithm
- Experimental Results
- Conclusions

Experimental Results

Test cases Characteristics.

Test cases	#FPGAs	#nets	#2-pin subnets
1	10	10,000	37,851
2	10	20,000	75,208
3	10	30,000	113,636
4	20	20,000	71,506
5	20	30,000	106,320
6	20	50,000	177,790

Heuristic



= 5

TDM = 10

Required physical wires = $\lceil 5/10 \rceil = 1$

= 1001

TDM = 20

Required physical wires = $\lceil 1001/20 \rceil = 51$

Experimental Results

Comparing our proposed algorithm against two other approaches.

Test case	#SLR crossings			$\frac{A-C}{A}$	$\frac{B-C}{B}$
	Best of 1000(A)	Heuristic(B)	Ours(C)		
1	65,612	59,720	46,934	28.44%	21.41%
2	129,309	113,930	81,021	37.32%	28.89%
3	198,570	178,757	140,465	29.29%	21.42%
4	123,565	113,685	70,508	42.94%	37.98%
5	184,030	171,157	101,475	44.88%	40.71%
6	307,620	290,695	168,859	45.11%	41.91%

Experimental Results

Execution statistics of our algorithm.

Test case	#SLR crossings			#rounds	Runtime(sec)
	Initial	Final	Reduction (%)		
1	67,889	46,934	30.87	4	43.34
2	134,407	81,021	39.72	5	110.99
3	204,376	140,465	31.27	4	107.37
4	126,795	70,508	44.39	4	103.26
5	186,807	101,475	45.68	5	182.68
6	310,660	168,859	45.65	5	278.82

Experimental Results

Comparing 10 trials of our algorithm
using different initial feasible pin assignments on each test case.

Test case	#SLR crossings			
	Initial		Final	
	Range	Variation	Range	Variation
1	67,262-70,030	3.953%	46,931-46,974	0.092%
2	132,839-136,201	2.468%	81,040-81,120	0.099%
3	204,990-209,537	2.170%	140,400-140,527	0.090%
4	125,122-126,833	1.349%	70,463-70,532	0.098%
5	185,555-188,480	1.552%	101,439-101,532	0.092%
6	308,957-314,968	1.908%	168,937-169,035	0.058%

Outline

- Introduction
- Preliminaries
- Algorithm
- Experimental Results
- Conclusions

Conclusions

- We introduced the SLR-aware pin assignment problem for modern multi-FPGA system utilizing high capacity 2.5D FPGAs.
- We proposed an iterative improvement algorithm based on integer linear programming to minimize the total number of SLR crossings in all FPGAs.
- Experimental results showed that the amount of SLR crossings can be significantly reduced by over 30% on average compared to two other approaches.

Thank you.

Q & A