Synthesizable FPGA Fabrics Targetable by the VTR CAD Tool

Jin Hee Kim and Jason Anderson

FPL 2015
London, UK
September 3, 2015

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Motivation for Synthesizable FPGA

- Trend towards ASIC design flow
- Design cost saving compared to custom layout
  - Manual layout is time costly
  - Number of design rule increases as process decreases
- Completes VTR (Verilog-to-Routing) flow
  - Flow currently “ends” after routing
  - Architectures from VTR can be realized in silicon
- SoC integration
  - Easily integrated
  - Allows programmability on SoCs
- Democratizes access to FPGA fabrics
  - Dominated by big companies
  - Opens up FPGAs to other companies

Motivation for Synthesizable FPGA

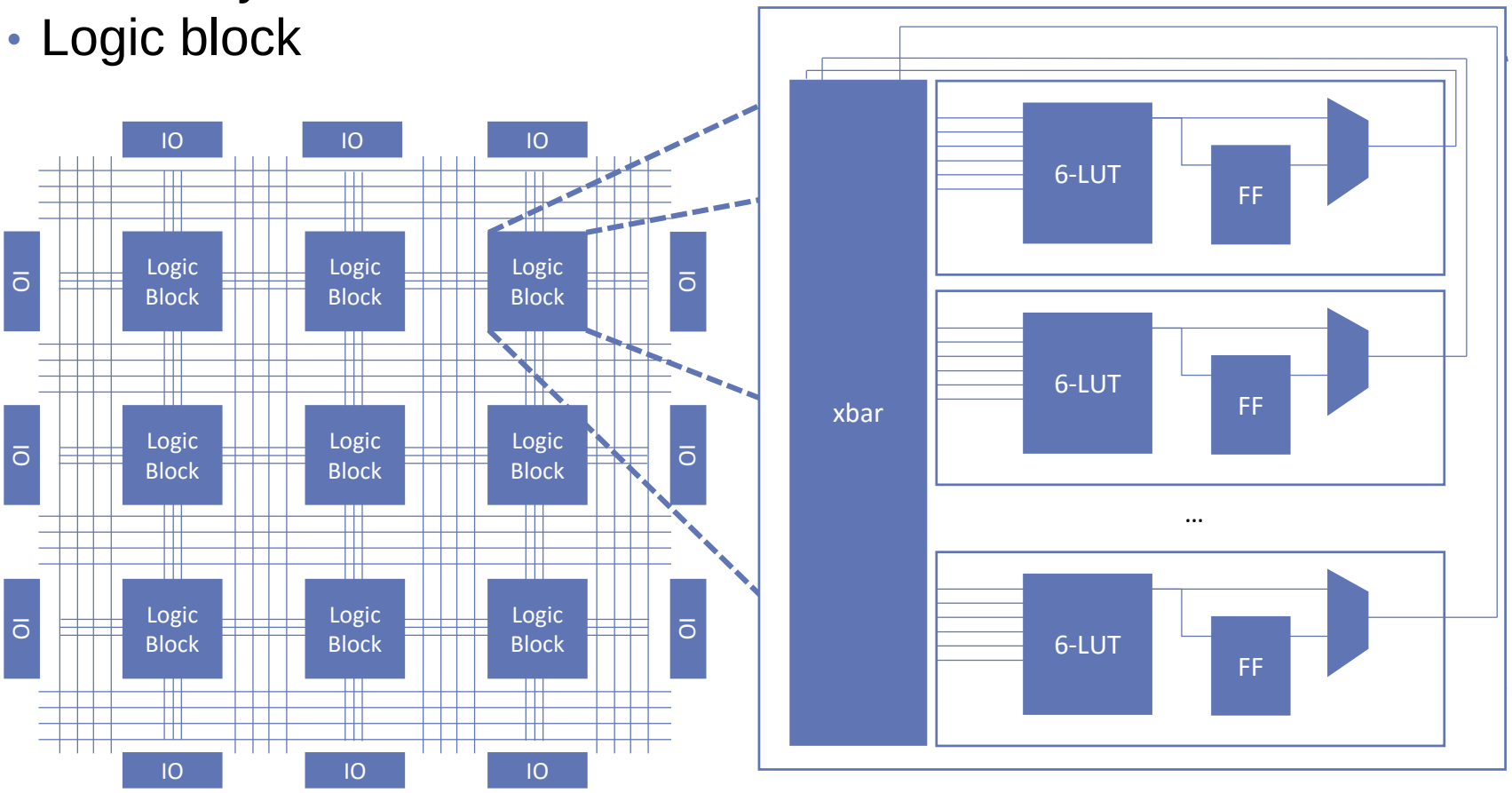
- Trend towards ASIC design flow
- Design cost saving compared to custom layout
  - Manual layout is time costly
  - Number of design rule increases as process decreases
- Completes VTR (Verilog-to-Routing) flow
  - Flow currently “ends” after routing
  - Architectures from VTR can be realized in silicon
- SoC integration
  - Easily integrated
  - Allows programmability on SoCs
- Democratizes access to FPGA fabrics
  - Dominated by big companies
  - Opens up FPGAs to other companies

Motivation for Synthesizable FPGA

- Trend towards ASIC design flow
- Design cost saving compared to custom layout
  - Manual layout is time costly
  - Number of design rule increases as process decreases
- Completes VTR (Verilog-to-Routing) flow
  - Flow currently “ends” after routing
  - Architectures from VTR can be realized in silicon
- SoC integration
  - Easily integrated
  - Allows programmability on SoCs
- Democratizes access to FPGA fabrics
  - Dominated by big companies
  - Opens up FPGAs to other companies

FPGA Architecture

- Island-style FPGA
- Logic block

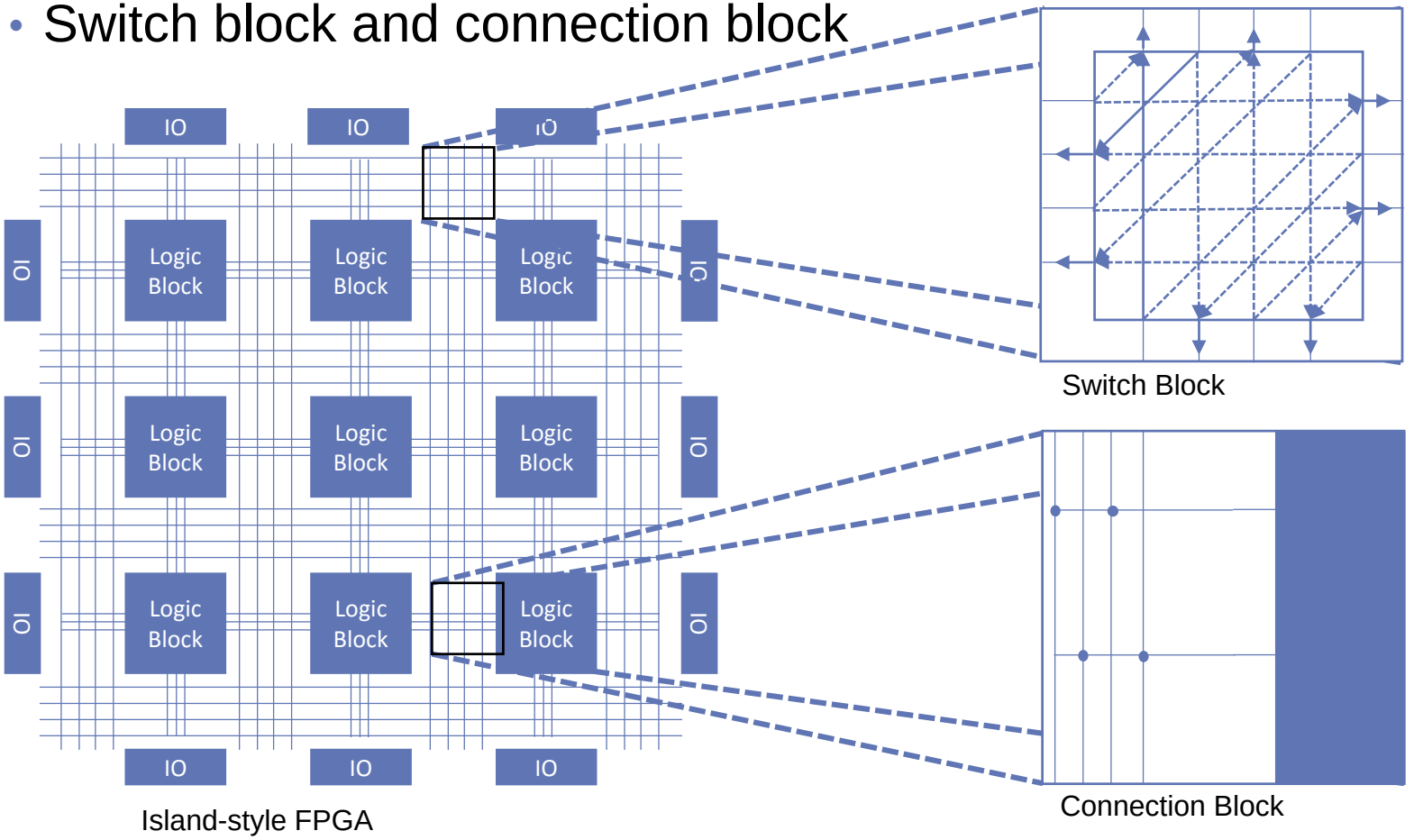


Island-style FPGA

Logic Block

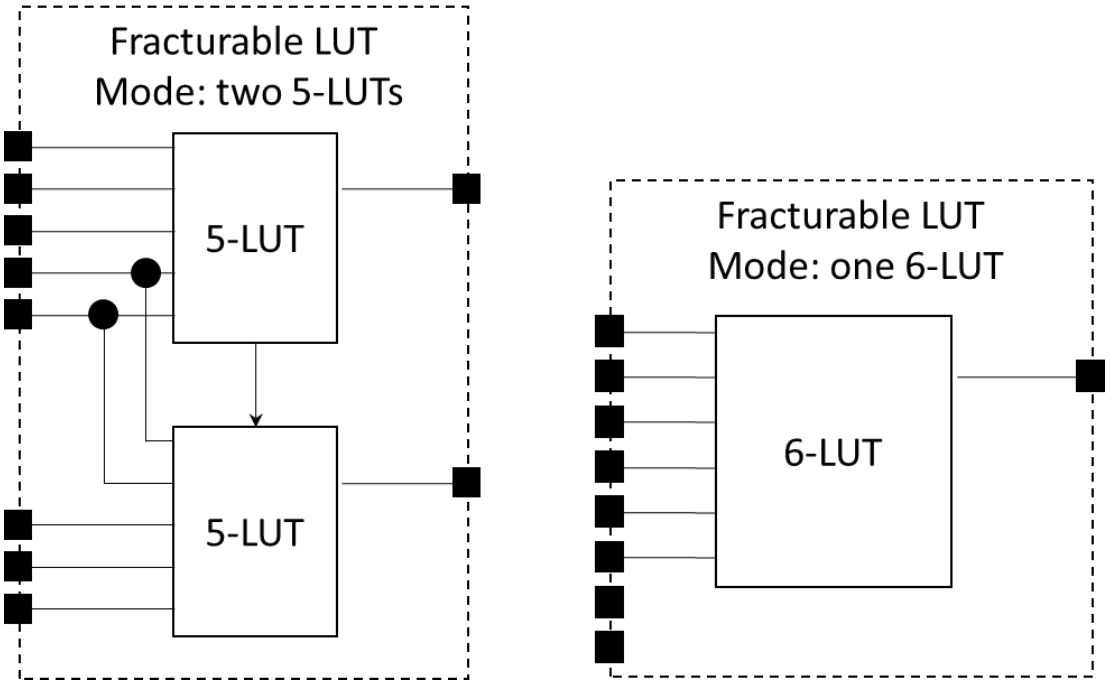
FPGA Architecture

- Horizontal and vertical routing tracks
- Switch block and connection block



Extensive Architecture (EArch)[1]

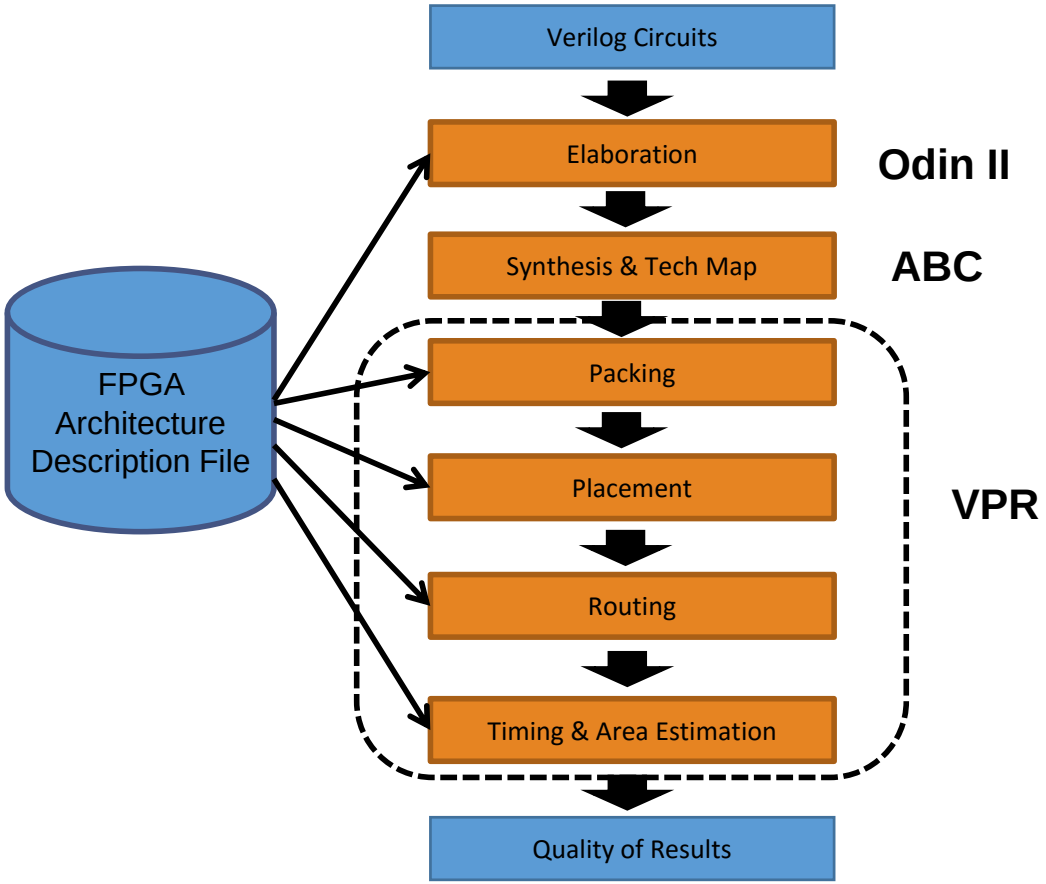
- Fracturable LUT
- 8 inputs (2 shared)



[1] Luu, Jason. *Architecture-Aware Packing and CAD Infrastructure for Field-Programmable Gate Arrays*. Diss. University of Toronto, 2014.

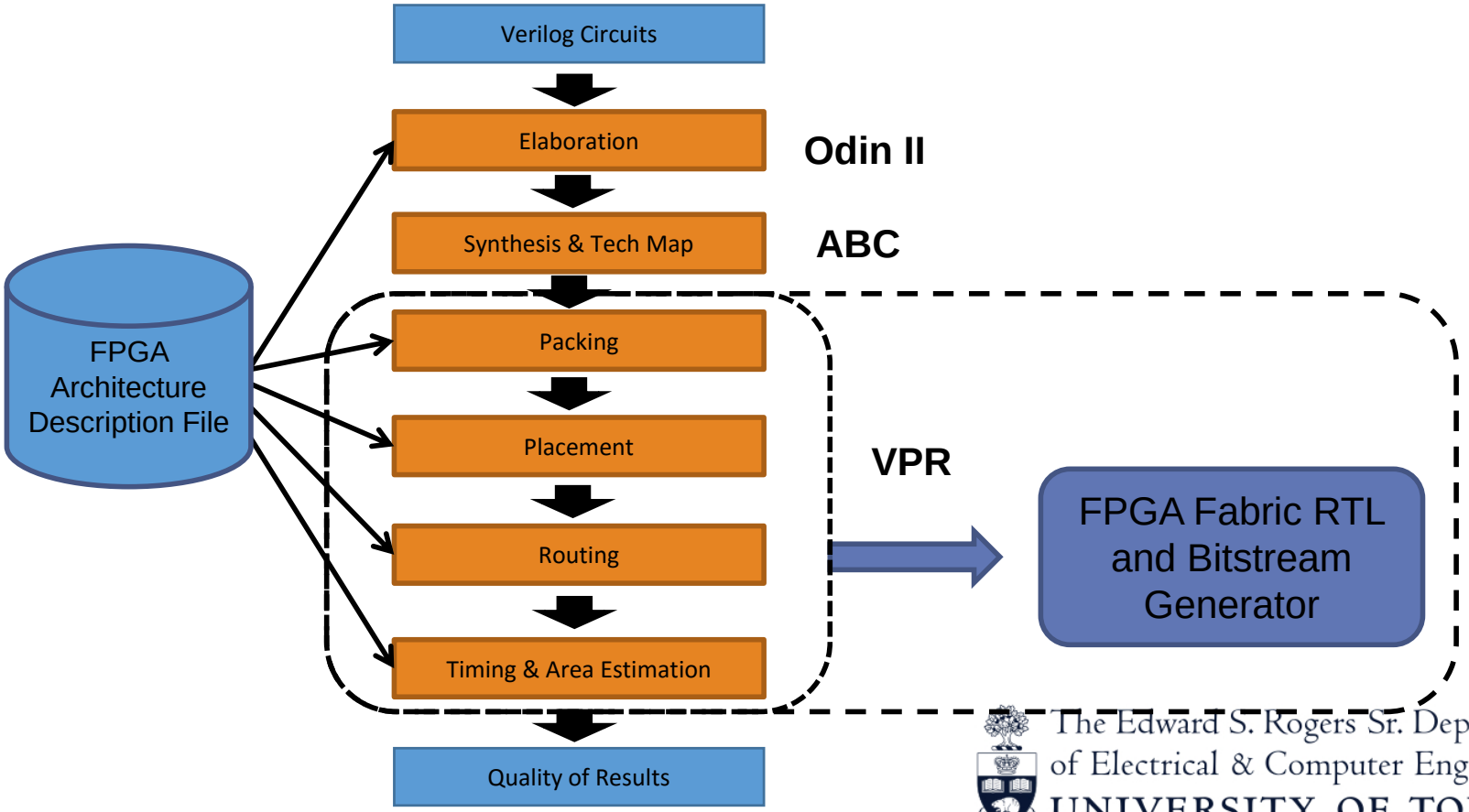
VTR Flow

- Takes Verilog circuits and places and route on an FPGA
- Currently VTR ends after routing



VTR Flow

- C code to extend VTR to generate RTL (Verilog) and bitstream

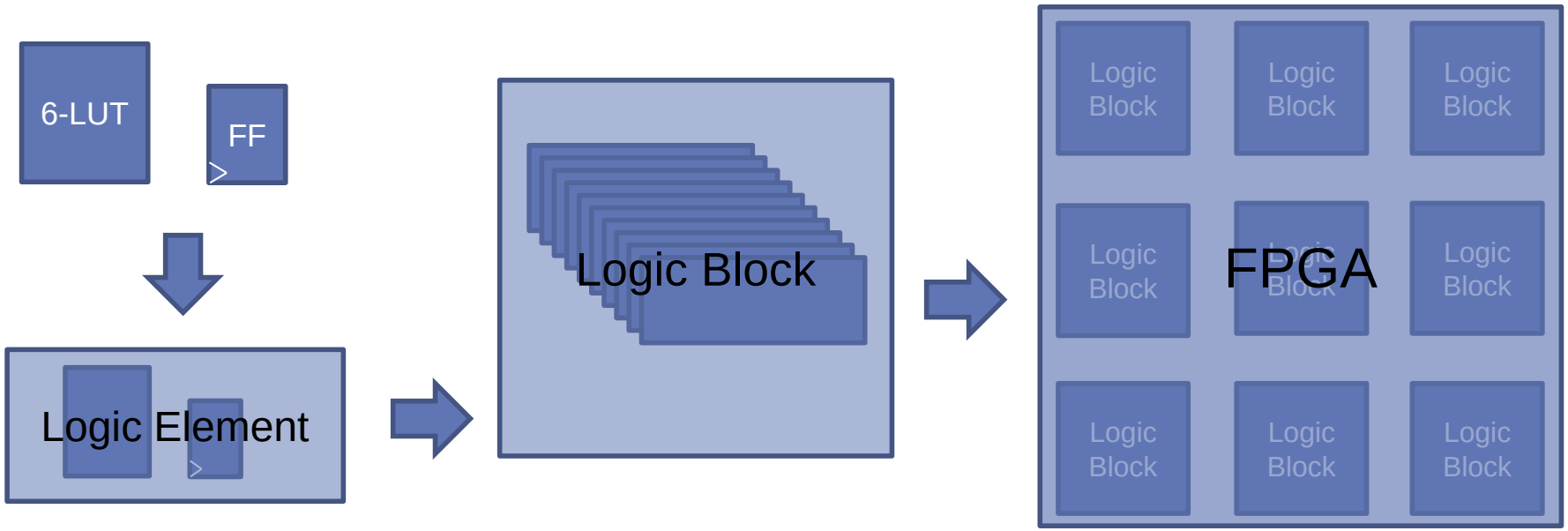


FPGA Fabric RTL Generation

- VTR represents the complete architecture in memory
- Walk the in-memory architecture model to generate Verilog for:
  - Logic blocks
  - Intra-logic block routing
  - Inter-logic block routing

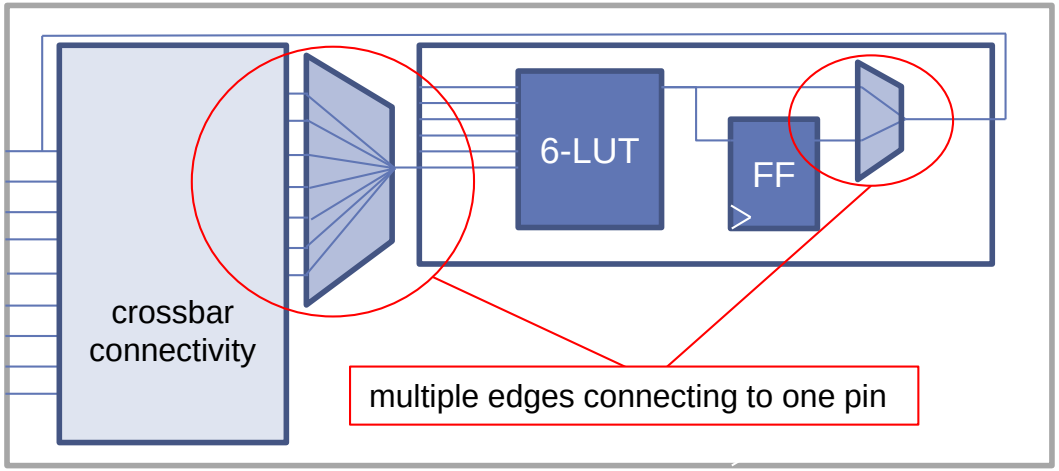
Generating Verilog for Logic

- Logic blocks:
  - Declare and instantiate modules from primitives (IO, LUT, FF)

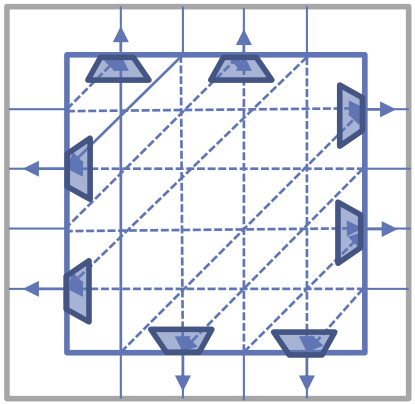


Generating Verilog for Routing

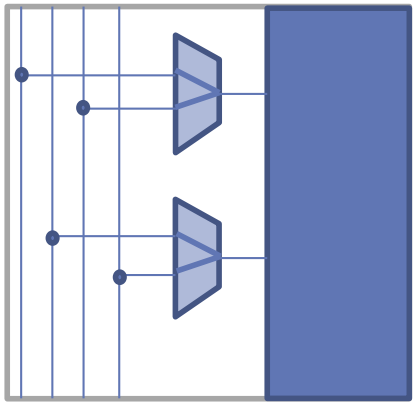
- Intra-logic block routing:



- Inter-logic block routing:



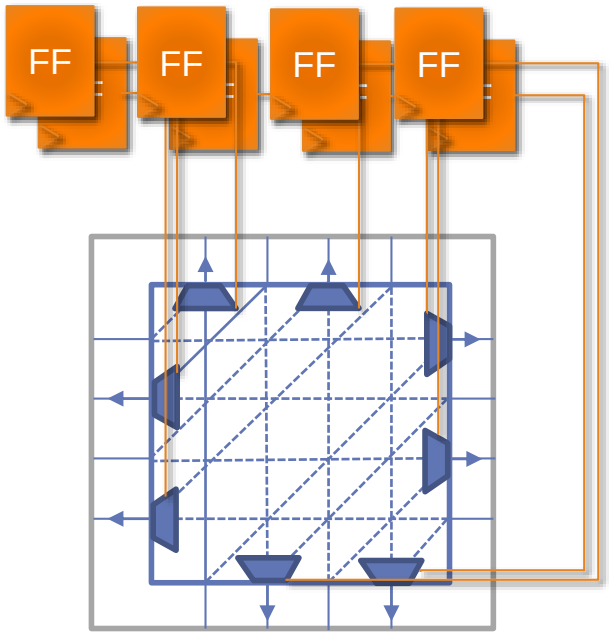
Switch Block



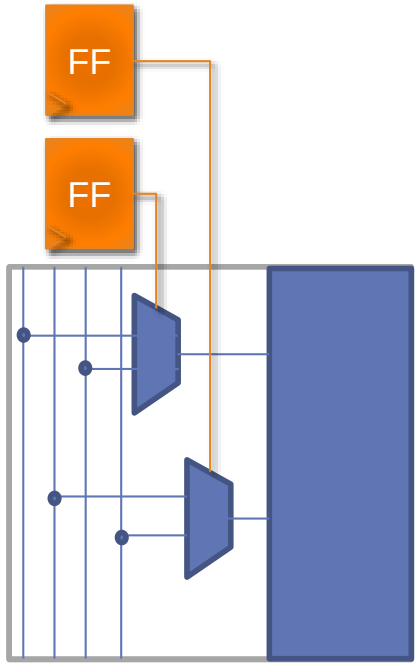
Connection Block

Generating Verilog for Configuration

- Configuration cells (FF) are attached to all MUXes and LUTs



Switch Block



Connection Block

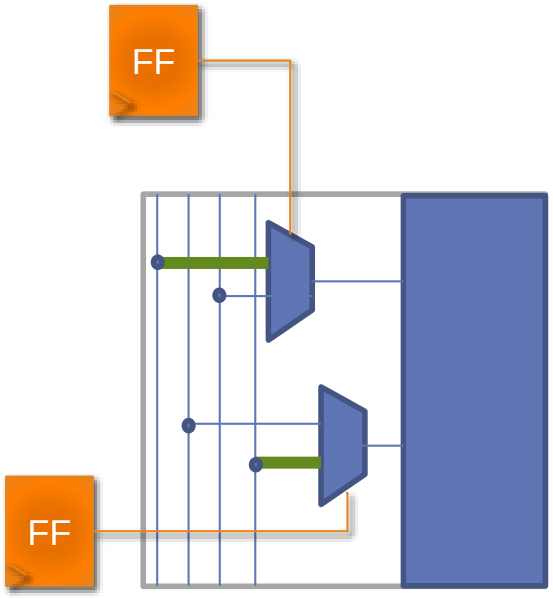
Generating Verilog for Configuration

- Configuration cells are all connected like a shift register
- Number of configuration bits are in ~750 000 for 20x20 FPGA

Configuration Cell Chain

Bitstream Generation

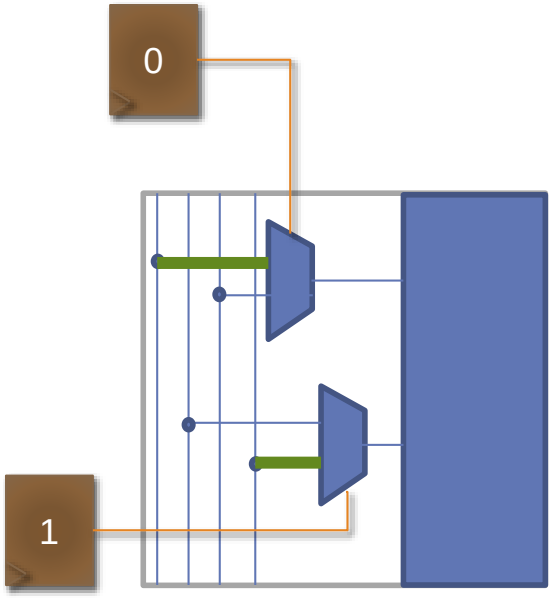
- VTR stores placed and routed circuit in memory
- Follows the same order as RTL generation
  - Configure LUTs and MUXes as necessary



Connection Block

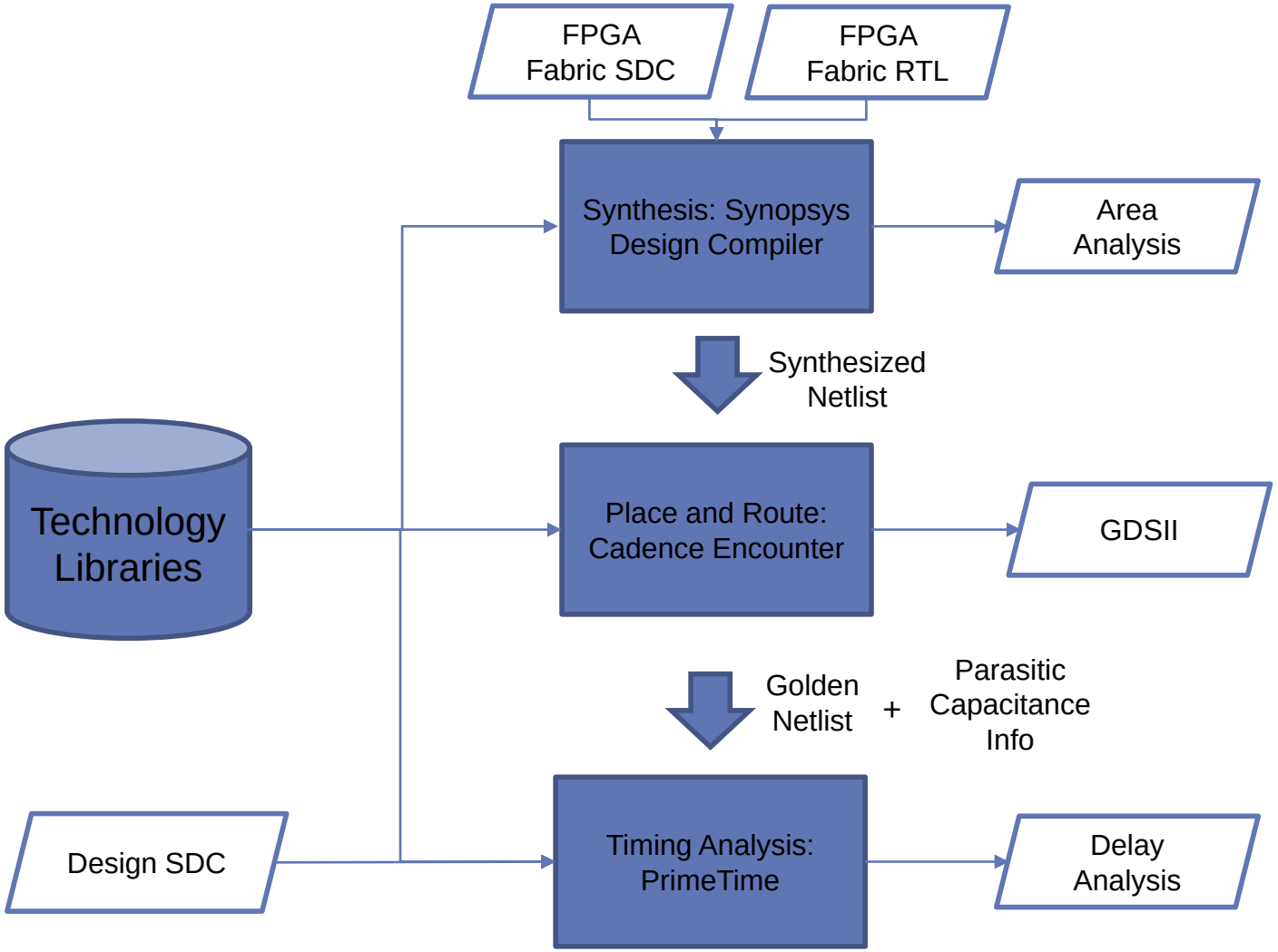
Bitstream Generation

- VTR stores placed and routed circuit in memory
- Follows the same order as RTL generation
  - Configure LUTs and MUXes as necessary



Connection Block

ASIC Design Flow



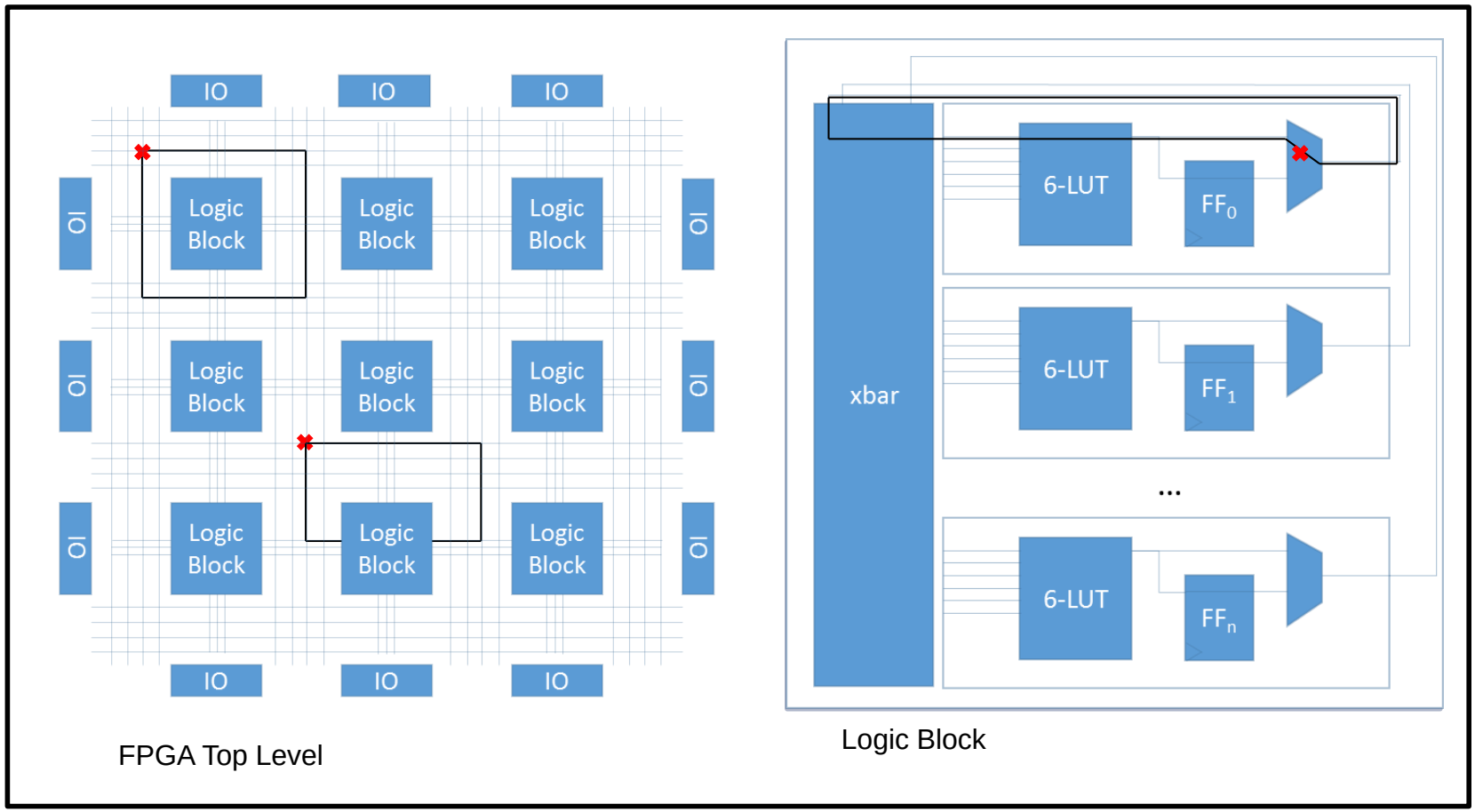
Synthesis: Constraints

- Area
  - Top-level is not constrained
  - For all other modules, constrain area to be minimum
    - Set max area to 0
- Timing
  - Set clock frequency and input-to-output delays
  - Cannot apply constraints with combinational loops
    - Designs configured in FPGAs generally do not have loops
    - Timing analysis tools does not know how to handle loops
    - Need to break loops by disabling timing arcs
- Balanced
  - Optimize for both area and timing

Synthesis: Constraints

- Area
  - Top-level is not constrained
  - For all other modules, constrain area to be minimum
    - Set max area to 0
- Timing
  - Set clock frequency and input-to-output delays
  - Cannot apply constraints with combinational loops
    - Designs configured in FPGAs generally do not have loops
    - Timing analysis tools does not know how to handle loops
    - Need to break loops by disabling timing arcs
- Balanced
  - Optimize for both area and timing

Synthesis: Constraints



Breaking Combinational Loops

Synthesis: Constraints

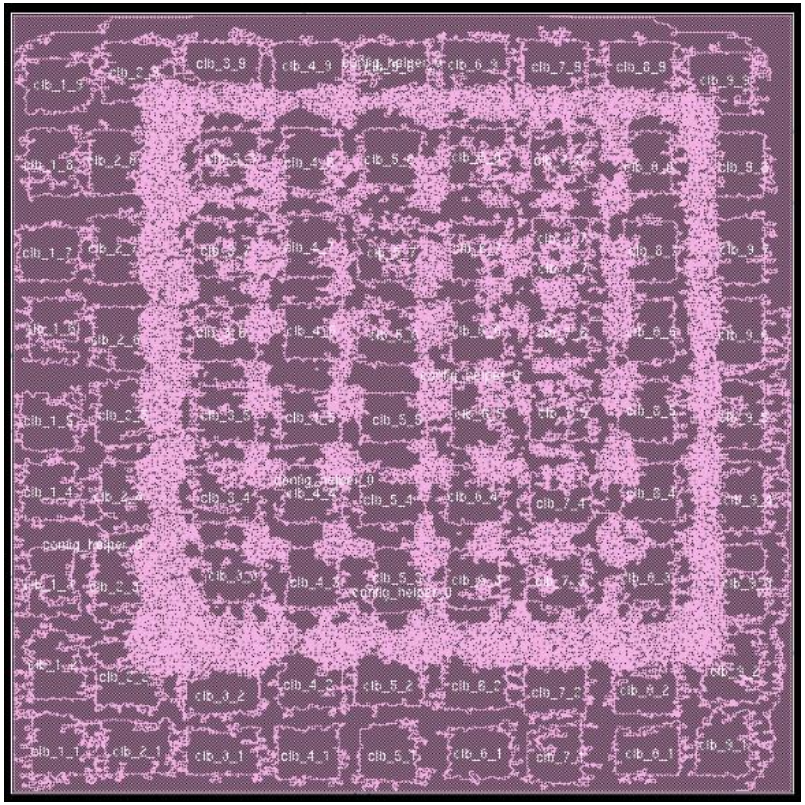
- Area
  - Top-level is not constrained
  - For all other modules, constrain area to be minimum
    - Set max area to 0
- Timing
  - Set clock frequency and input-to-output delays
  - Cannot apply constraints with combinational loops
    - Designs configured in FPGAs generally do not have loops
    - Timing analysis tools does not know how to handle loops
    - Need to break loops by disabling timing arcs
- Balanced
  - Optimize for both area and timing

Place and Route

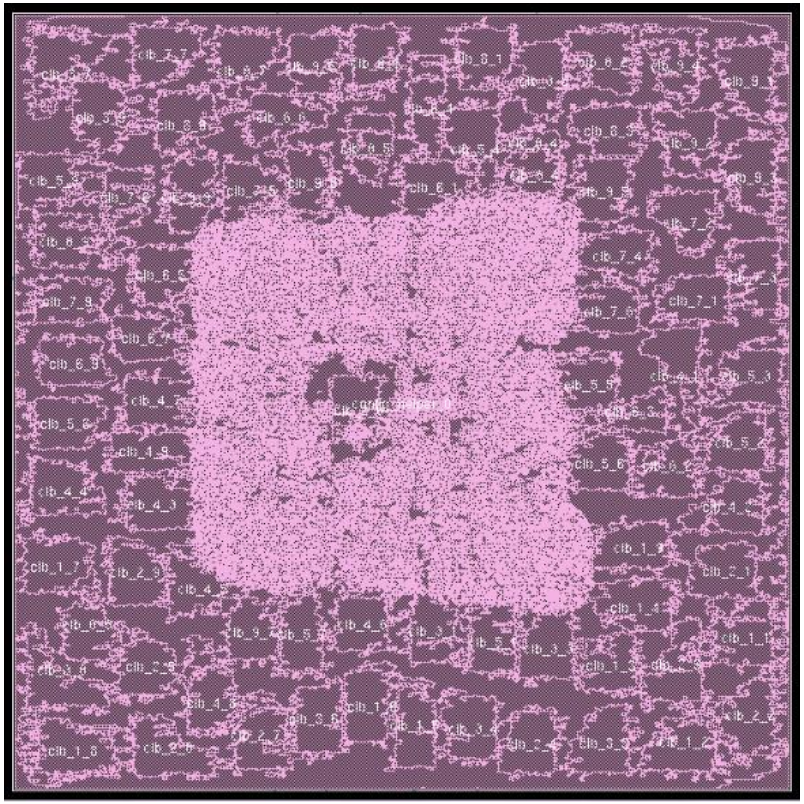
- Placed and routed as a flat design
- VTR has certain notion of where each logic block is located but Encounter tool does not
- Floorplanning:
  - Logic blocks, IOs, and routing MUXes are floorplanned at their respective x and y location from VTR
  - 85% utilization[2]
- Parasitic capacitances obtained

[2] Kuon, Ian, and Jonathan Rose. "Measuring the gap between FPGAs and ASICs." *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 26.2 (2007): 203-215.

Floorplanning Impact

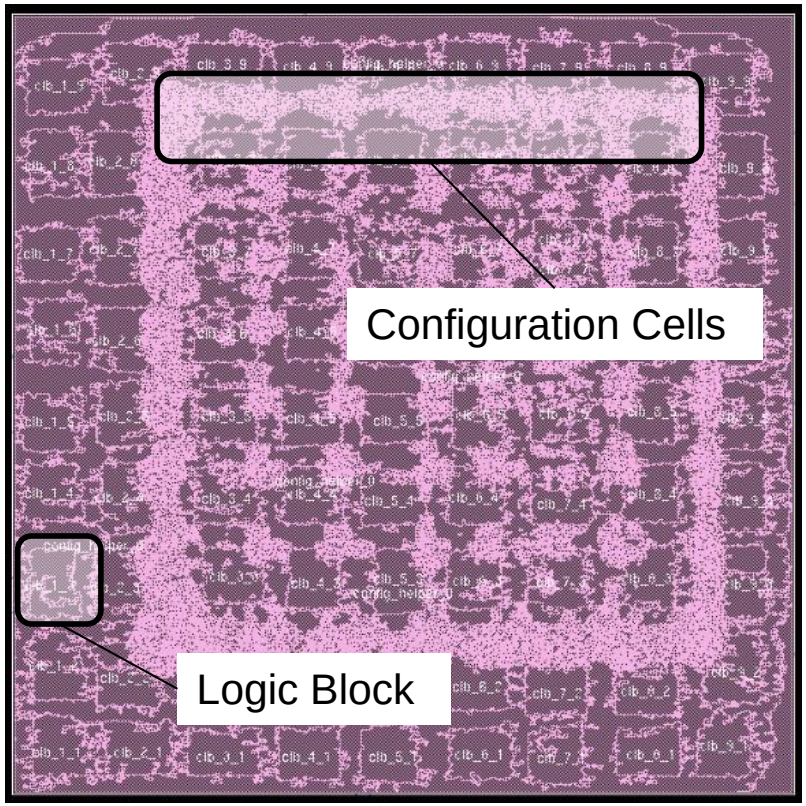


FPGA with Floorplanning

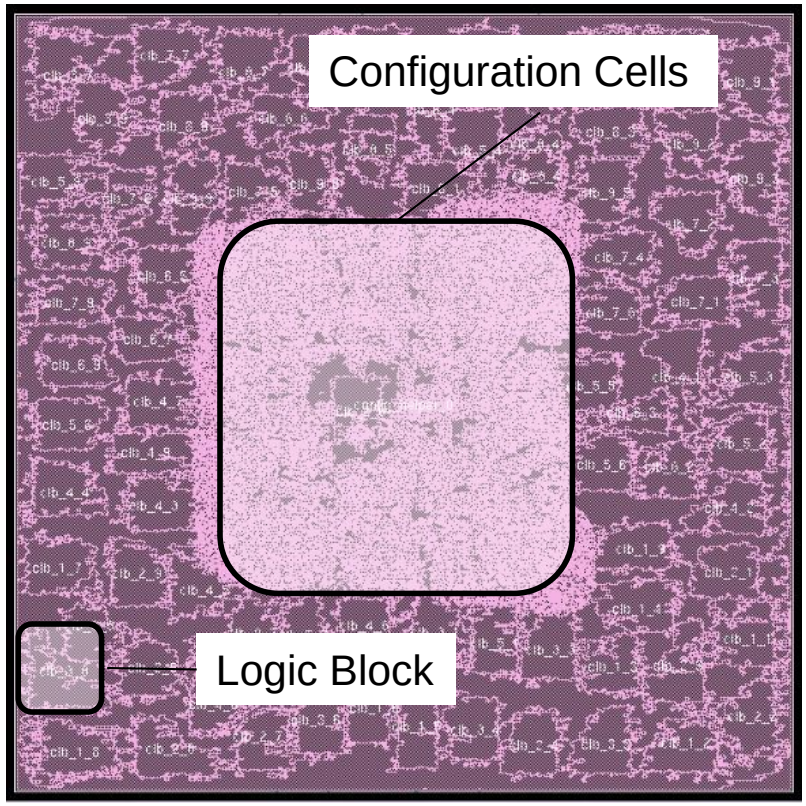


FPGA without Floorplanning

Floorplanning Impact

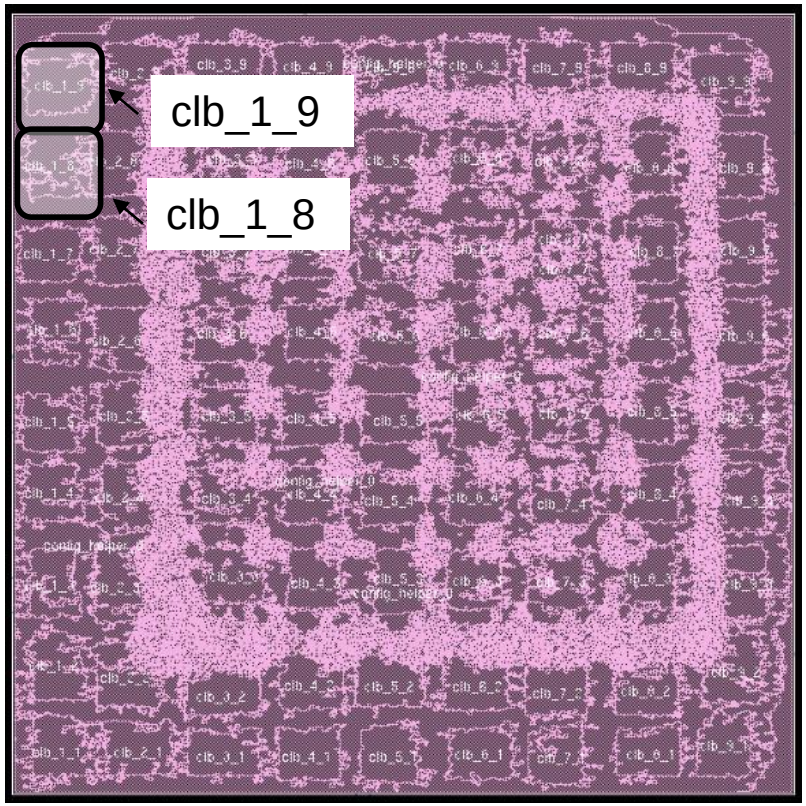


FPGA with Floorplanning

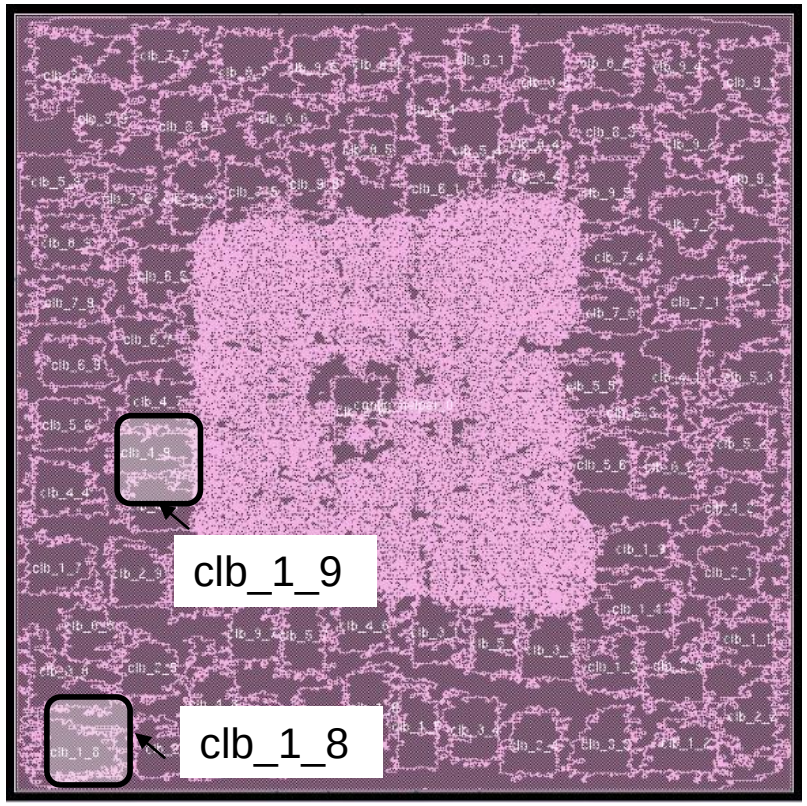


FPGA without Floorplanning

Floorplanning Impact

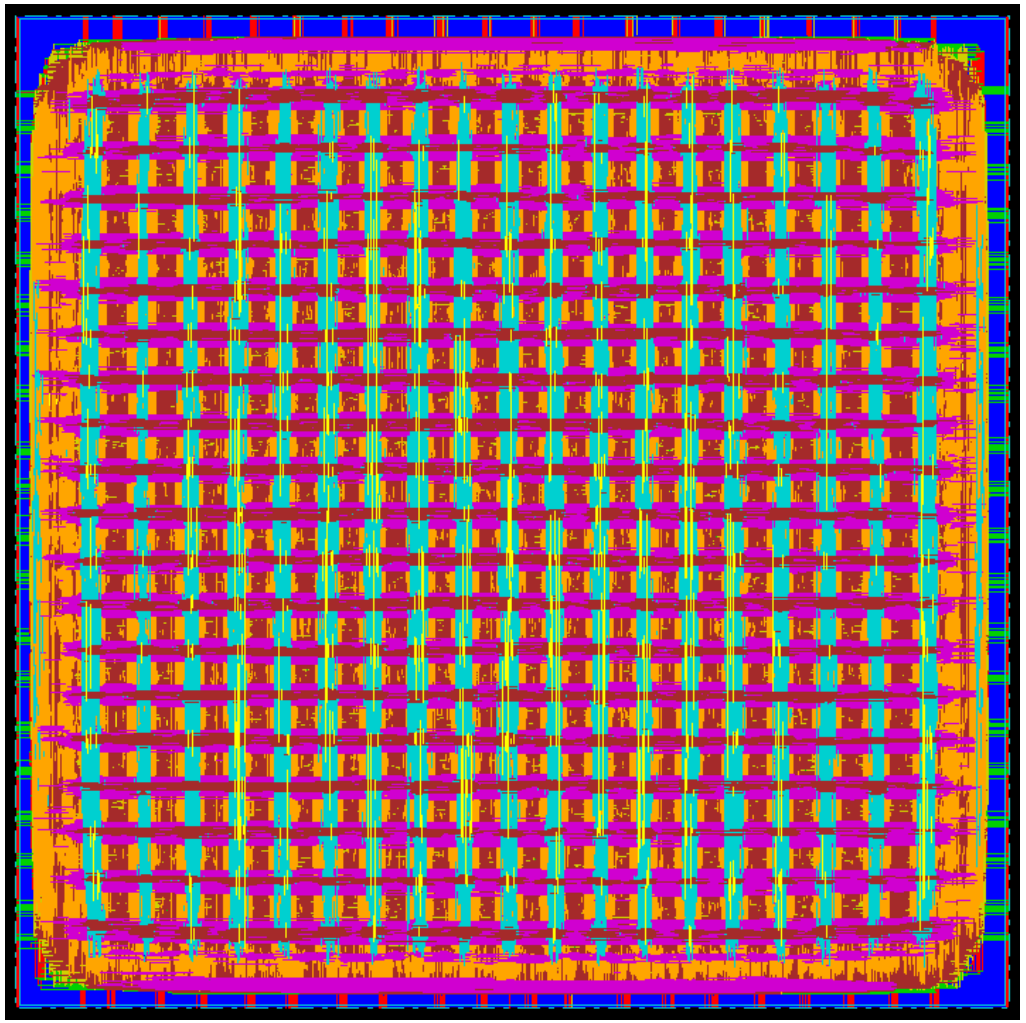


FPGA with Floorplanning



FPGA without Floorplanning

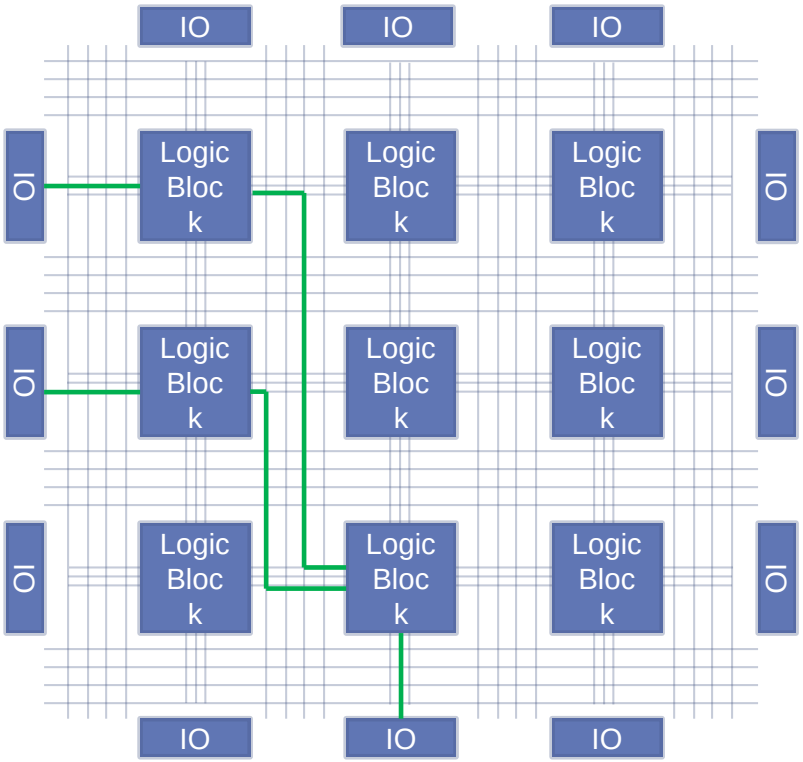
FPGA Layout

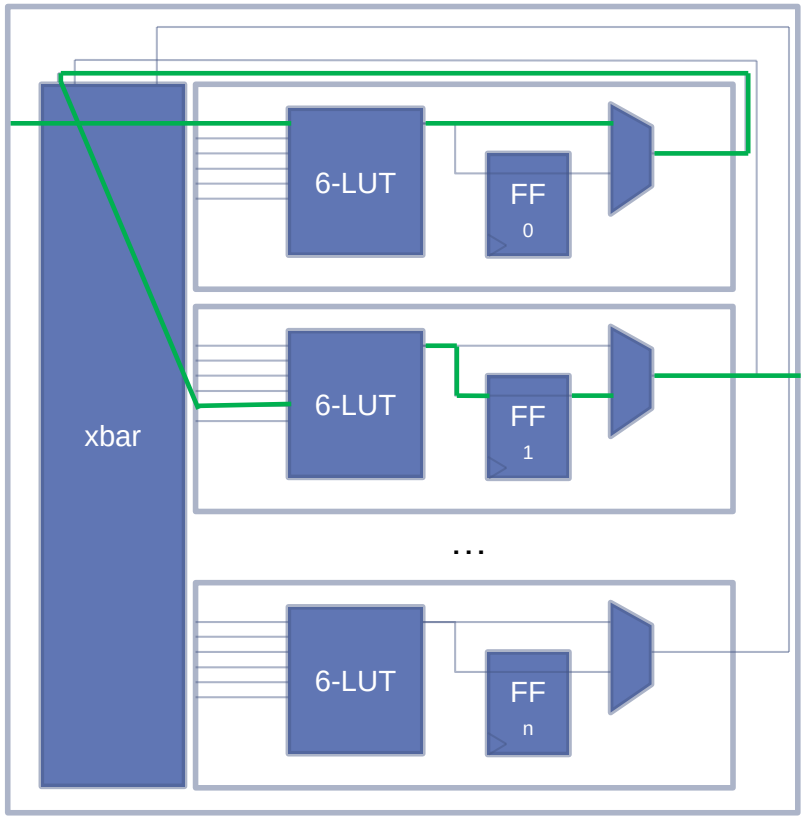


PrimeTime

- Timing analysis for design on FPGA
  - Find critical path and delay
- When programmed, should not have loops
- Unused blocks should not be included
- Constraint file produced from VTR to imitate a programmed FPGA
  - Disable timing arc
  - Different than synthesis

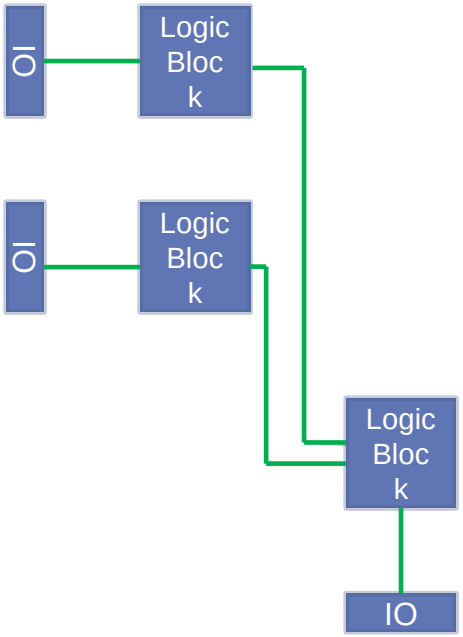
PrimeTime: Programming Design

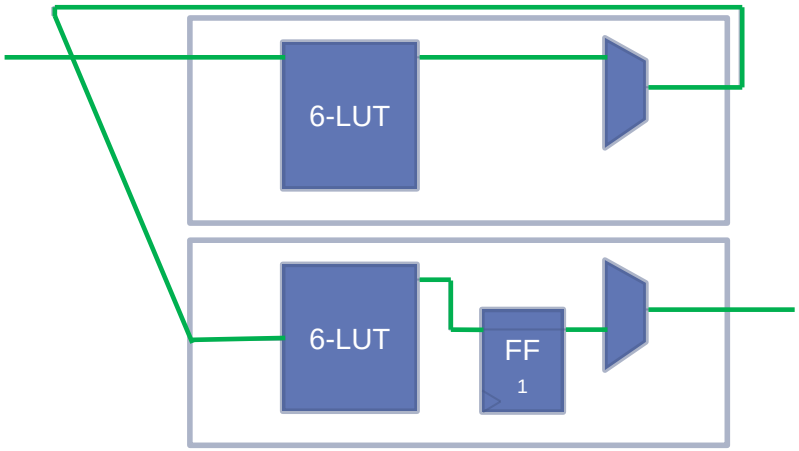




- Green line represents programmed path

PrimeTime: Programming Design

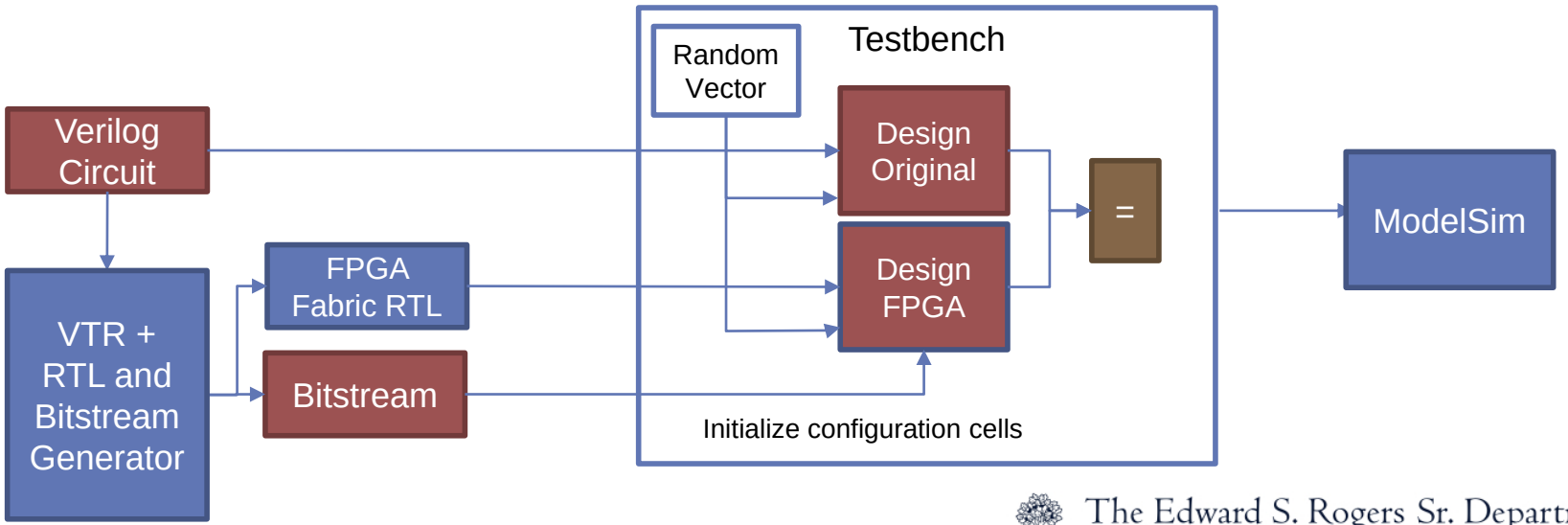




- The critical path now will be determined from the used paths

Verification

- Pre-synthesis
- Post-synthesis
- Post-place and route



Experimental Setup

- FPGA architecture
  - 20 x 20 logic blocks
  - 300 routing channel width[3]
  - 87% length 4 wires and 13% length 16 wires[3]
  - EArch (no carry chain and no hard blocks)[1]
- FPGA fabric type
  - Area-optimized
  - Timing-optimized
  - Balanced
- TSMC 65nm library

[3] Murray, Kevin E., et al. "Titan: Enabling large and complex benchmarks in academic CAD." *Field Programmable Logic and Applications (FPL), 2013 23rd International Conference on*. IEEE, 2013.

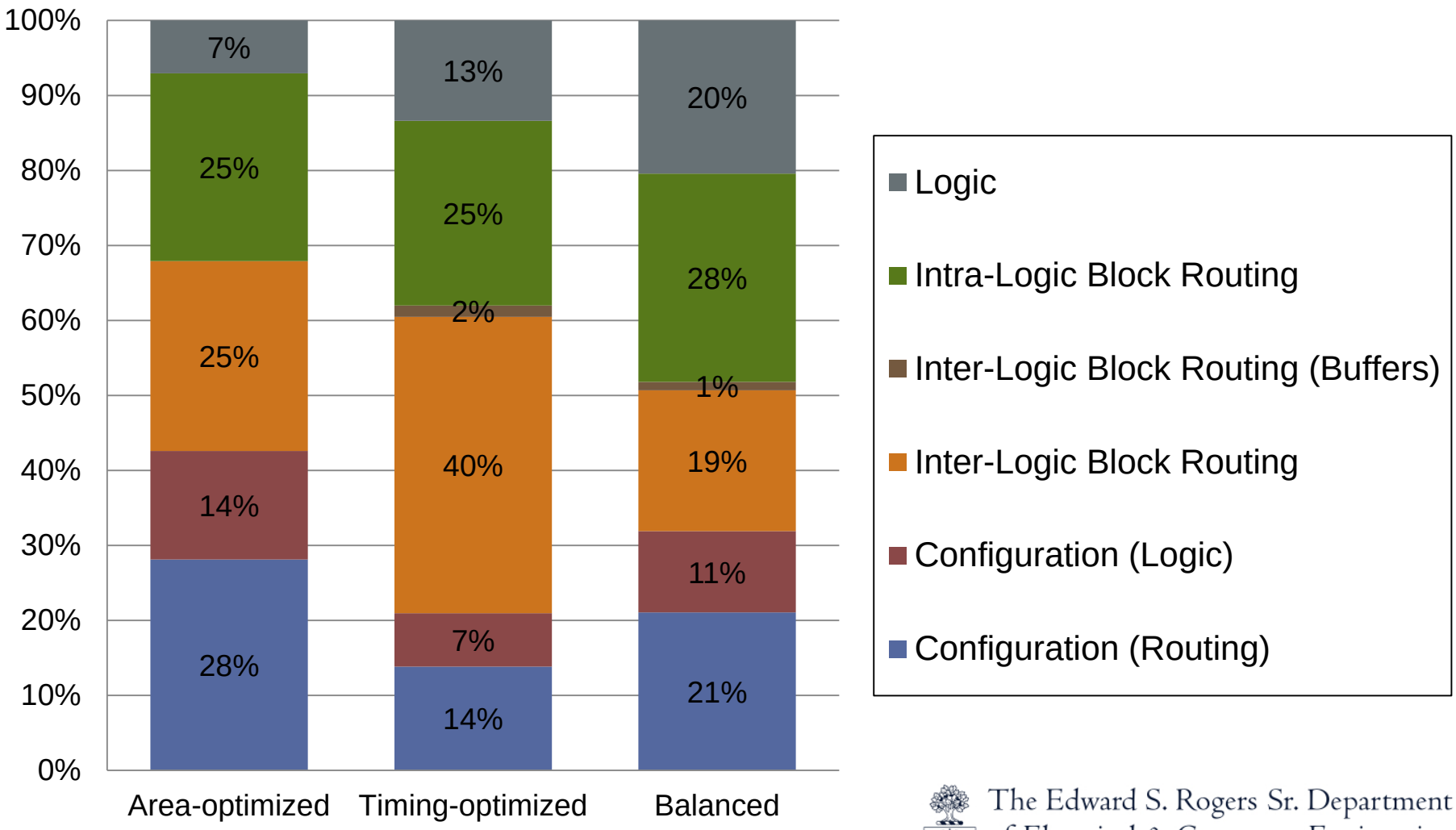
Area Comparison

- Similar architecture to Altera's Stratix III
  - 65nm process
  - 10 fracturable LUTs
  - 50% depopulated crossbar
  - No carry-chain
- Stratix III LAB tile area[4]: 0.0221mm^2
- Can achieve relatively close area with area-optimized

FPGA Fabric	# of Std. Cells	Total Area (mm^2)	Tile Area (mm^2)	Tile Area Vs. Stratix III
Area-optimized	3,577,520	12.65	0.0316	1.5x
Timing-optimized	7,521,616	25.72	0.0643	2.9x
Balanced	5,298,588	16.89	0.0422	1.9x

[4] Wong, Henry, Vaughn Betz, and Jonathan Rose.
"Comparing FPGA vs. custom CMOS and the impact on processor microarchitecture."
Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays. ACM, 2011.

Area Breakdown



The Edward S. Rogers Sr. Department of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Delay Comparison

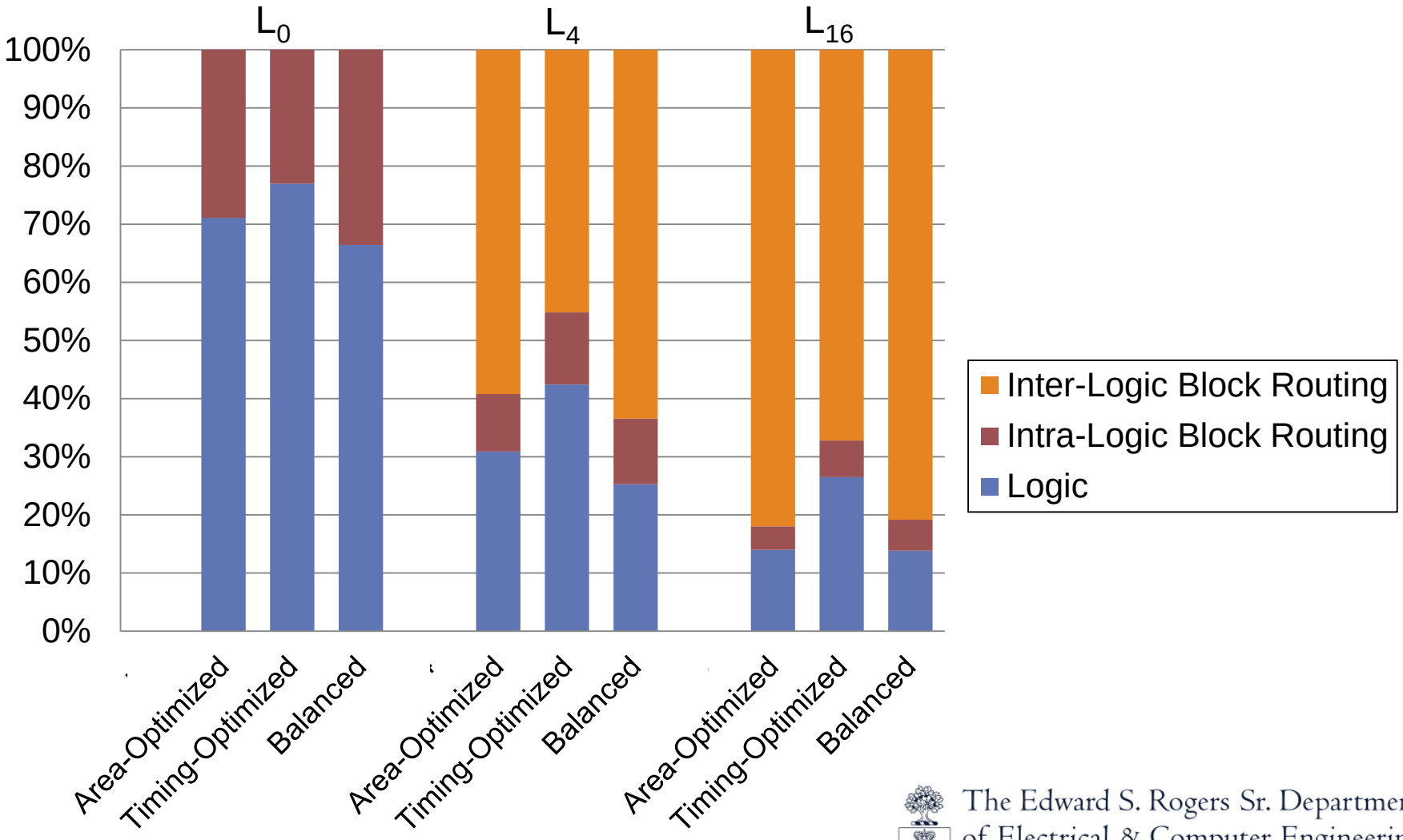
- Experiment 1: Architecture delay (benchmark agnostic)
  - L_0: FF ➔ xbar ➔ LUT ➔ FF
  - L_4: FF ➔ length 4 wire ➔ xbar ➔ LUT ➔ FF
  - L_{16}: FF ➔ length 16 wire ➔ xbar ➔ LUT ➔ FF
- Experiment 2: Benchmark delay
  - Design tool and architecture dependent

Delay Comparison: Architecture

- Average delay of paths in 6 different areas of FPGA
- 2.3x to 3.5x slower as the wire length grows

FPGA Fabric	L_0 (ns)	L_4 (ns)	L_{16} (ns)
Area-optimized	3.71	7.38	17.31
Timing-optimized	1.79	2.90	4.92
Balanced	1.34	3.73	7.32
Stratix III	0.73	1.03	1.54

Delay Breakdown: Architecture



The Edward S. Rogers Sr. Department of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

Delay Comparison: Design

Benchmark	Area-optimized (ns)	Timing-optimized (ns)	Balanced (ns)	Stratix III (ns)
alu4	67.8	22.29	32.47	5.293
apex4	74.62	23.06	36.08	5.271
des	68.65	21.58	31.26	6.696
ex1010	103.59	31.36	48.09	7.248
ex5p	68.68	21.86	31.54	5.455
misex3	74.78	22.59	34.32	5.281
pdc	111.02	33.72	51.66	7.213
seq	69.79	22.66	32.66	5.742
spla	112.73	34.92	51.3	6.654
diffeq	69.42	23.28	29.74	4.391
dsip	35.94	11.71	17.75	5.918
elliptic	103.42	32.45	44.02	6.909
frisc	118.72	38.06	52.37	7.865
tseng	60.43	20.05	25.92	4.519
addshift16	37.01	13.43	16.61	4.31
fsm	5.75	1.71	2.85	1.113
Geo. Mean	63.13	20.1	28.97	5.25

Combinational circuits

Sequential circuits

Conclusion

- VTR extended to generate synthesizable FPGA fabric RTL and bitstream
- ASIC design flow used to synthesize 3 FPGA fabrics
  - Area-optimized
  - Timing-optimized
  - Balanced
- Area-optimized fabric is 1.5x bigger than Stratix III
- Designs on timing-optimized fabric is 3.8x slower

Questions?