VPR Assessment of a Novel Partitioning Algorithm

David Munro

Supervisor: Oliver Diessel

Overall Goal

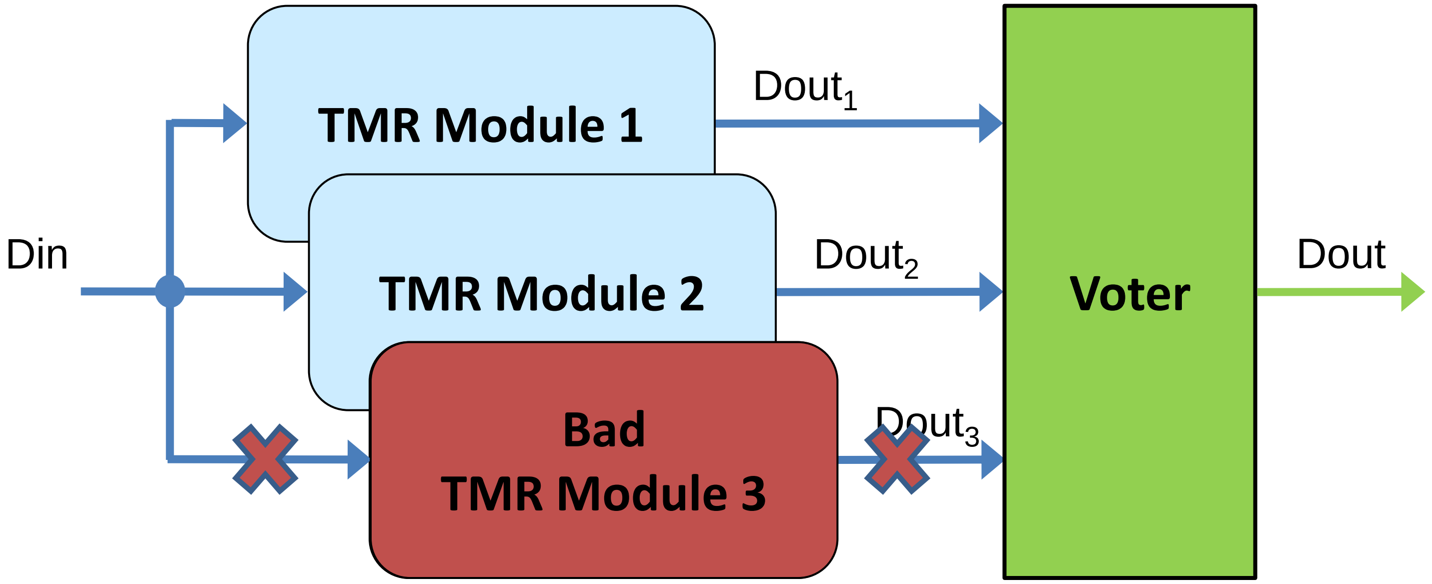
- To implement the partitioning algorithm and assess the results using Versatile Place and Route (VPR) and MCNC benchmarks.
- VPR used as it’s open source.

Overview

- What are we partitioning and why?
- FPGA's are useful for space based applications due to low cost, wide availability, etc. [1].
- Downsides include increased susceptibility to radiation induced errors[1]. For Virtex 4 in geosynchronous orbit, predicted mean time between errors is only 1.4s[2].
- Use Triple Modular Redundancy (TMR) to detect errors.

Triple Modular Redundancy

- Make three copies of a circuit, and feed the outputs to a voter.
- Once an error is detected we can fix it by e.g. selectively reconfiguring the incorrect module.
- Clock frequency, pipeline length and circuit area all affect recovery time.
- Need to detect, reconfigure and resynchronise within error rate.



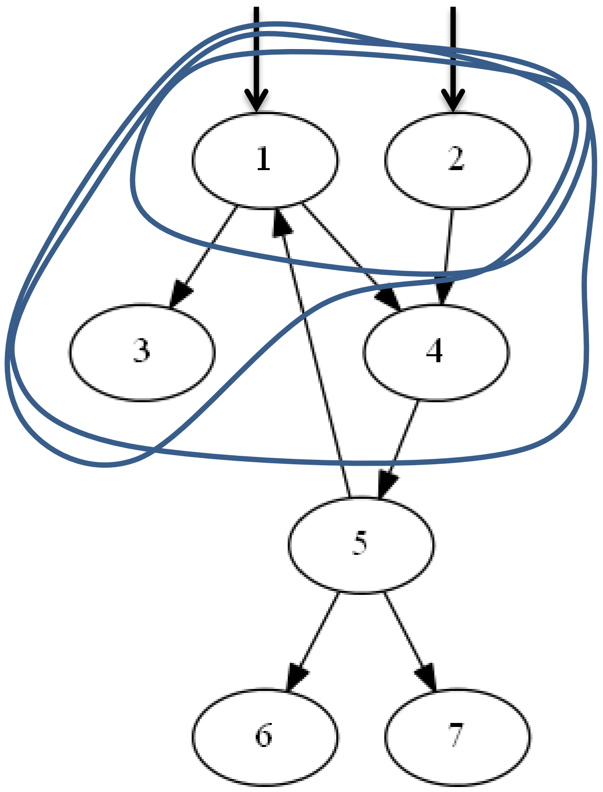
Source Cetin & Diessel (2012)

Partitioning in action

- Start with inputs.
- Add nodes in a breadth first manner.
- Continue until area, frequency or critical path exceeds threshold.

Max recovery time	Estimated recovery time	Area	Critical Path	Frequency
1.00E-08	2.00E-08	40	2	50MHz

- TMR-ify node-set.
- Repeat process until all nodes are TMR'd.



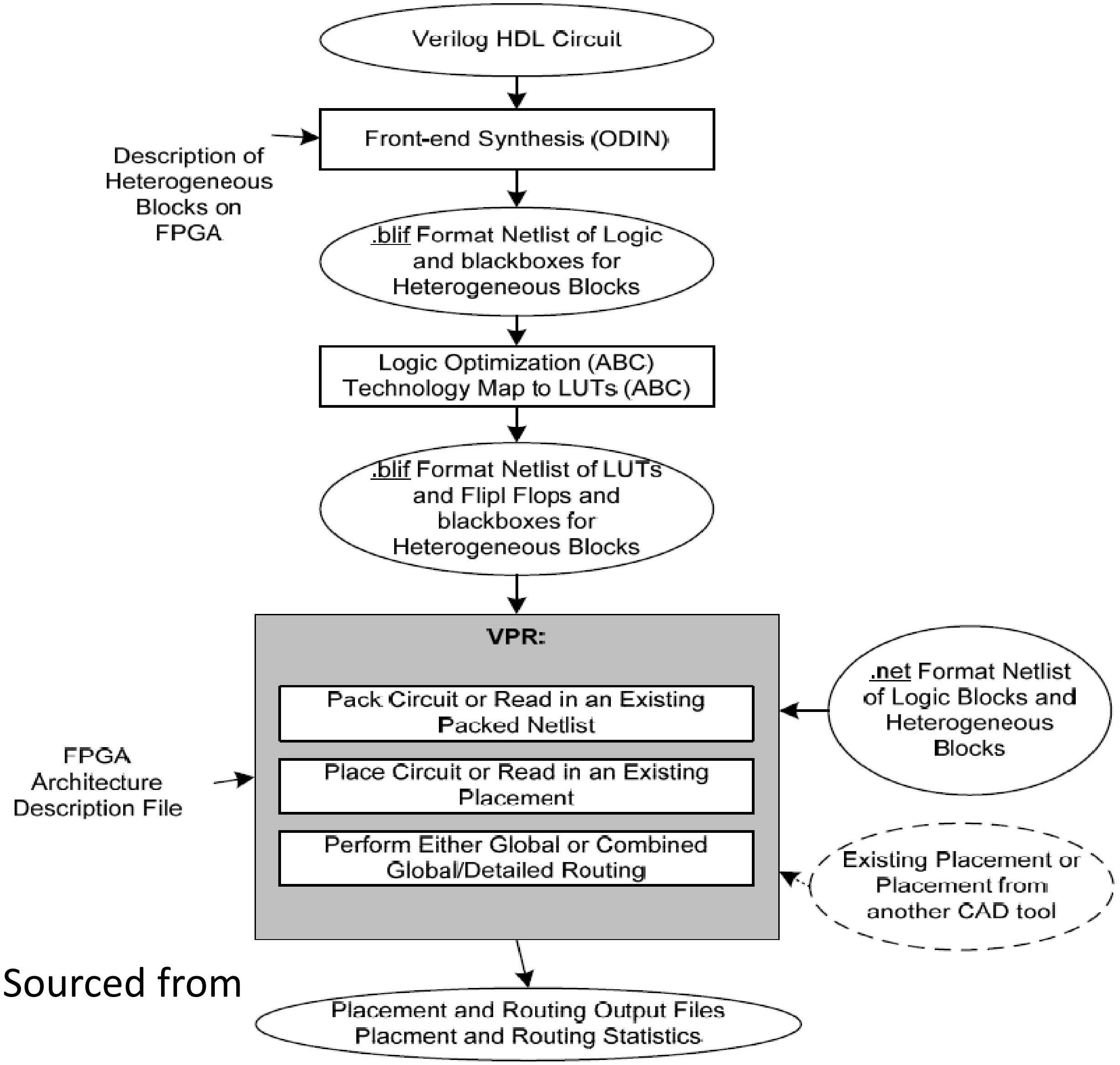
Partitioning wavefront

Partitioning Considerations

- To do this, need a way of efficiently calculating area, frequency and pipeline length for a set of nodes.
- Pipeline length is trivial, the other two not so much.
- No way to tell until design is routed, which takes too long, therefore we need some way of estimating.
- Also, to effectively traverse, need circuit as a graph. VHDL/Verilog too high level, needs to happen post-synthesis, somewhere in CAD flow.

Versatile Place and Route (VPR)

- Open source.
- Open file formats.
- Want to partition as late as possible.
- Adding elements means routing and placing again => partition just before or after packing.
- Before packing is easier, but might be less effective.


Verilog HDL Circuit
Front-end Synthesis (ODIN)
Description of Heterogeneous Blocks on FPGA
.blif Format Netlist of Logic and blackboxes for Heterogeneous Blocks
Logic Optimization (ABC) Technology Map to LUTs (ABC)
.blif Format Netlist of LUTs and Flipl Flops and blackboxes for Heterogeneous Blocks
VPR:
Pack Circuit or Read in an Existing Packed Netlist
Place Circuit or Read in an Existing Placement
Perform Either Global or Combined Global/Detailed Routing
FPGA Architecture Description File
.net Format Netlist of Logic Blocks and Heterogeneous Blocks
Existing Placement or Placement from another CAD tool
Placement and Routing Output Files Placment and Routing Statistics

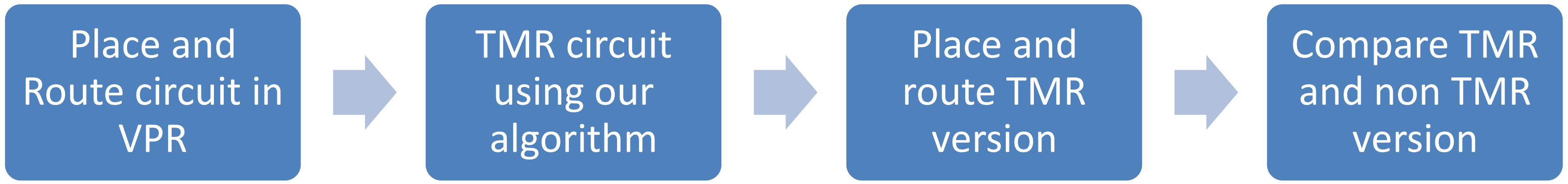
Picture of CAD design flow. Sourced from VPR manual

Input File

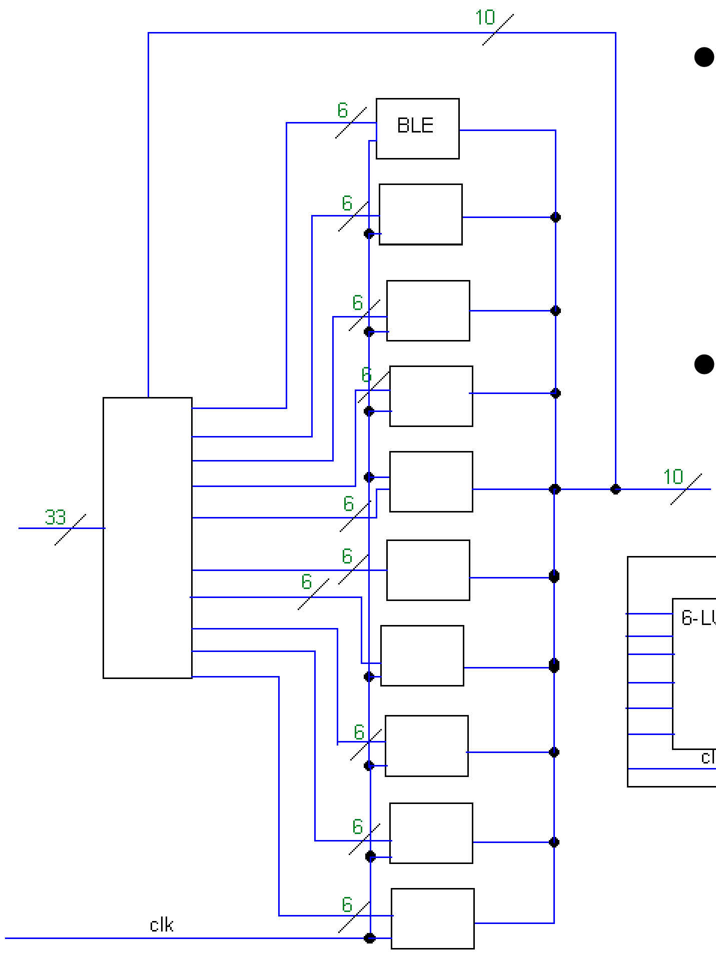
- Read blif file into in-memory graph.
- Partition graph.
- Insert voter logic into blif file as appropriate.
- File format is text input of inputs, outputs and logic elements.
- Only logic elements supported or Look Up Tables (LUT) and latches.

Current Progress

- TMR arbitrary user specified subcircuit preserving surrounding circuit.
- Just need to automate subcircuit selection, instead of user defined.
- To do that need way of estimating area/timing from set of nodes.

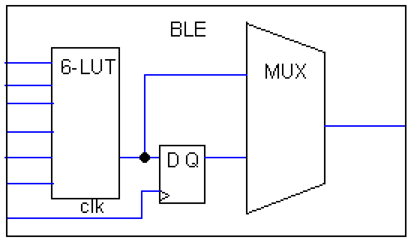

Place and Route circuit in VPR
TMR circuit using our algorithm
Place and route TMR version
Compare TMR and non TMR version

Architecture



CLB and BLE layout

- Use imaginary standard architecture, somewhat similar to Virtex 5.
- FPGA are and routing channels grow to accommodate design.



- Customisable, so can make more accurate to target a specific platform.

Benchmarking

- Two things we’re looking at.
- 1. Comparing circuits with and without TMR, as per thesis title.
- 2. Trying to find a relationship, or suitable guesses, for our partitioner.
- For now we are calculating minimum channels and area to look at general trends. Later we look at the case where we are close to channel/area capacity.
- Circuits used are MCNC (Microelectronics Centre of North Carolina) benchmarks, provided by the VTR (Verilog To Routing) Project.

Results

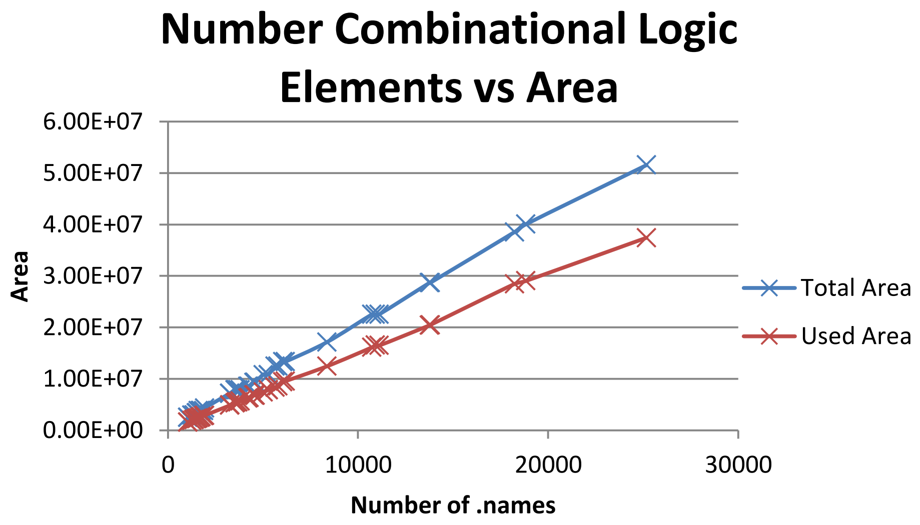
- Quite close to initial guess.
- Area requirement slightly greater than tripled.
- Mean increase in critical path time is 6.9E-10s. Max is 1.87E-09s increase to 8.28E-9s (156MHz->121MHz max frequency)

Name	.latch	.names	FPGA Area	Estimated Critical Path Latency	Channel Width	Av. Wire Segments	Used Logic Block Area	Critical Path
pdc	0	13765	44	7.46E-09	72	7.74884	2.04E+07	8.28E-09
misex3	0	4205	24	5.12E-09	50	5.86212	6.24E+06	5.27E-09
s38417	4389	18232	51	6.39E-09	50	3.96346	2.84E+07	6.80E-09
pdc TMR	0	4575	25	5.47E-09	64	7.8308	6.78E+06	6.41E-09
misex3 TMR	0	1397	14	3.73E-09	42	5.74857	2.07E+06	4.29E-09
s38417 TMR	1463	6042	30	5.91E-09	42	4.02152	9.42E+06	6.32E-09

Results – Estimating Area

- We also care about estimating area and latency from a DFG.

- Clear relationship between nodes and area.

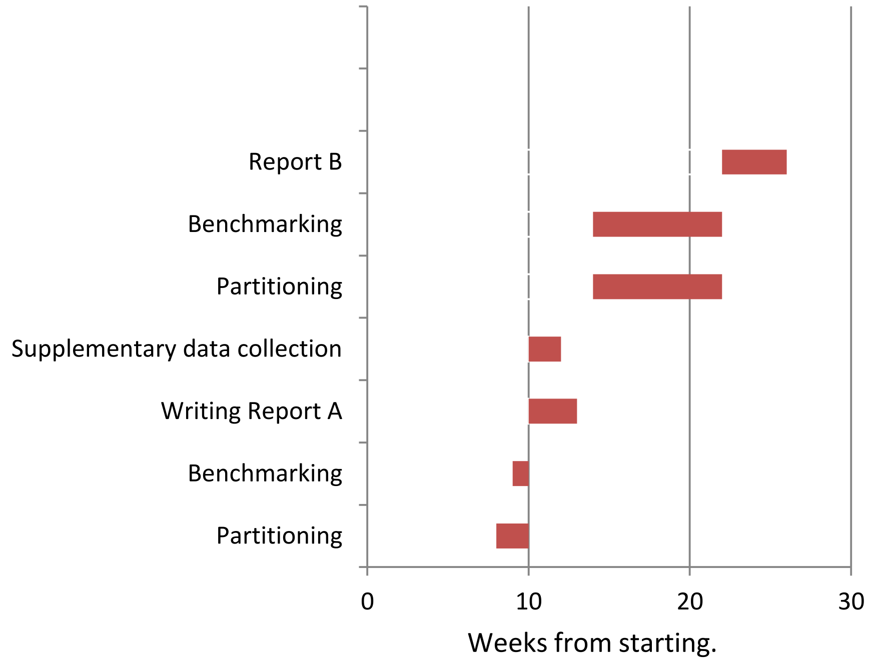


Results – Estimating Time

- Much harder to estimate time before placement.
- Options:
- Guessing. Guided by number of blocks or pre-TMR time, may not be entirely inaccurate.
- Partial placement. Placer uses simulated annealing, so run very rough placement to get estimate [3].
- Partition after placement. Better knowledge, much harder to do.

What Next?

- Using very basic and arbitrary estimation functions for area and timing, implement partitioning.
- Improve estimates.



Timeline, not including holidays

References

[1] E. Cetin and O. Diessel, ”Guaranteed Fault Recovery Time for FPGA-based TMR Circuits Employing Partial Reconfiguration”. In 2012 DAC Workshop 2nd International Workshop on Computing in Heterogeneous, Autonomous ‘N’ Goal-oriented Environments (CHA’N’GE), 2012.

[2] P. J. Pingree, “Advancing NASA’s on-board processing capabilities with reconfigurable fpga technologies,” in Aerospace Technologies Advancements, T. T. Arif, Ed. In Tech, Jan. 2010, ch. 5, pp. 69–86.

[3] Luu Jason, et al. “VPR User’s Manual”, Jan. 2012.

[4] V. Betz, J. Rose & A. Marquardt. “Architecture and CAD for deep-submicron FPGAs,” 1st Edition, March 1999 Springer.