Congestion-Driven Re-Clustering for Low-cost FPGAs

Darius Chiu
Guy Lemieux, Steven Wilton

FPT 2009
December 10, 2009

University of British Columbia
Department of Electrical and Computer Engineering
Vancouver, BC, Canada
Outline

• Motivation and background
• Algorithm
• Results
• Conclusion
• Future Work
Problem: Unroutable Circuit

CAD flow fails to route a circuit design
- Available routing < circuit requirements

Congestion Map:
- Spatial map of CLB location
- Colors indicate routing resource usage
- Red ➔ high interconnect use
- Blue ➔ low interconnect use
Problem: Unroutable Circuit

- Only localized area is actually unroutable
- Routing congestion happens locally
- CAD flow needs to avoid local congestion
Motivation

• Meet local channel-width constraint
  – # routing tracks fixed at manufacture

• This work most suitable for low-cost devices:
  – Low-cost devices $\rightarrow$ narrow channel width
  – High-cost devices $\rightarrow$ wide channel width
Contributions

• Simultaneous area and runtime improvement over previous work

• **Congestion-Driven Reclustering**
  – Selecting congested regions
  – Interconnect-demand model to predict routability

• Findings
  – Up to 5.5x runtime improvement
  – Up to 20% area improvement
Reclustering

Netlist → Cluster + Place → Routing

Re-route

Re-place + re-cluster
(Whitespace Insertion)

Unroutable!
Background: Un/DoPack

Marvin Tom [ICCAD2006]
New Un/DoPack (CMR)

Cluster → Place → Route

Re-cluster

Whitespace Insertion

Region selection

Target CW Met?

YES

NO

Un/DoPack

VPR
New Un/DoPack

Cluster → Place → Route

Region selection

Re-cluster → Identify

Target CW Met?

NO

Place → Route

YES

Un/DoPack
Region Selection

Find congested regions
  – Use post-routing congestion information

Old approach
  – Center region on most congested CLB

New approach
  – Move region, find most congested region
Region Selection

Force directed shift:
- Congestion values generate direction to move region
Region Selection

Search
- Find region with highest average congestion
Region Selection

Mark all regions

Sort and depopulate
  – Sort regions by average congestion
  – CLBs depopulated only once
Whitespace Insertion

• How much whitespace to add?

  – Utilize interconnect usage information post-routing
  – Estimate new region channel-width after depopulation
Interconnect-Demand Model

\[ W_{\text{abs.min}} = \frac{\lambda R}{2} = \rho \lambda \]

\[
W = W_{\text{abs.min}} + \frac{1}{\beta} \left( \frac{W_{\text{abs.min}}}{F_s} \right) \left( \frac{W_{\text{abs.min}}}{F_{\text{Cin}}} \right)^{\alpha_{\text{in}}} \left( \frac{W_{\text{abs.min}}}{F_{\text{Cout}}} \right)^{\alpha_{\text{out}}}
+ \frac{\lambda(L-1)}{4} \left( 1 + \frac{1}{F_{\text{Cin}}} \right)
\]

\[ W = \rho \lambda + k_0 (\rho \lambda)^{k_1} + k_2 \]

W. Fang and J. Rose. “Modeling routing demand for early-stage FPGA architecture development”
Interconnect-Demand Model

\[ W = \rho \lambda + k_0 (\rho \lambda)^{k_1} + k_2 \]

Apply model to region:

1. Use congestion map & architectural parameters, determine \( \rho \)
2. Using \( \rho \) and desired \( W \), determine \( \lambda \)
3. Re-cluster region until target \( \lambda \) met
Regional Interconnect

Internal interconnect
Depopulating spreads interconnect demand

External interconnect
Depopulating does not affect external interconnect

• Use bounding-box based congestion estimation method to determine how much internal vs external (D. Yeager, D. Chiu, G, Lemieux. “Congestion estimation and localization in FPGAs”)

• region interconnect-demand = internal demand + external demand
Congestion-Model Multiregion Un/DoPack (CMR)

- Multiple region selection
- Budget based on congestion-model
- Re-cluster each region to meet target $\lambda$
Outline

• Motivation and background
• Algorithm
• Results
• Conclusion
• Future Work
Benchmark Circuits

• Metacircuits designed to emulate large SOC circuits
  ~ 40000 LUTs
  – Hierarchical synthetic circuits created using GNL
  – Same average Rent parameter, different standard deviations for individual regions
Runtime / Area Tradeoff

Previous Approaches

• Faster runtime ➔ Worse area
• Better area ➔ Worse runtime
Runtime / Area Tradeoff

Previous Approaches

• Faster runtime ➔ Worse area
• Better area ➔ Worse runtime
Runtime / Area Tradeoff

Previous Approaches

- Faster runtime ➔ Worse area

- Better area ➔ Worse runtime
Runtime + Area Improvements
Typical Results
(Stdev004 Circuit)
Congestion Driven Placement

• When used with congestion-driven placer
   ➔ improves area further

• Aside: this particular placer is very slow
Conclusions

• Use congestion information to perform better re-clustering
  – Up to 5.5x runtime speedup versus baseline
  – Up to 20% area savings versus baseline

• Improve Runtime/Area tradeoff of Un/DoPack Flow
  – Simultaneous area and runtime savings
Future Work

• Consider effect of neighboring regions after re-clustering

• Other congestion-driven tools
  – Faster congestion-driven placement
  – Congestion-driven clustering, routing

• Accurate congestion estimator
  – Eliminate routing step for identifying congested regions / calculate whitespace insertion
Acknowledgements

• Funding from Altera, NSERC
Questions?
Budgeted Multiregion Un/DoPack (BMR)

- Identify regions
- Expand regions
  (1 row & 1 column of region)
- Limit New CLBs
  (1 row & 1 column of FPGA)
Total Regional Interconnect

Use congestion estimation method to determine % of each

Maintain flexibility to use when no routing information is available