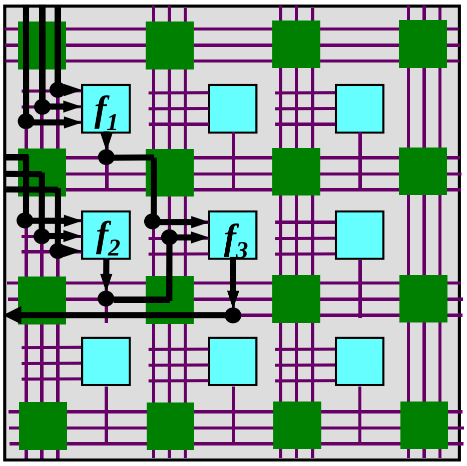
FPGA Routing

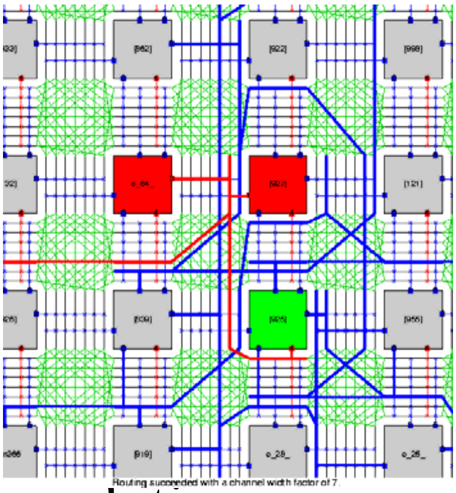
Outline

- Constraints and objectives
- Routing resource graph
- Global routing and detailed routing
- VPR router
- Timing-driven routing



Introduction

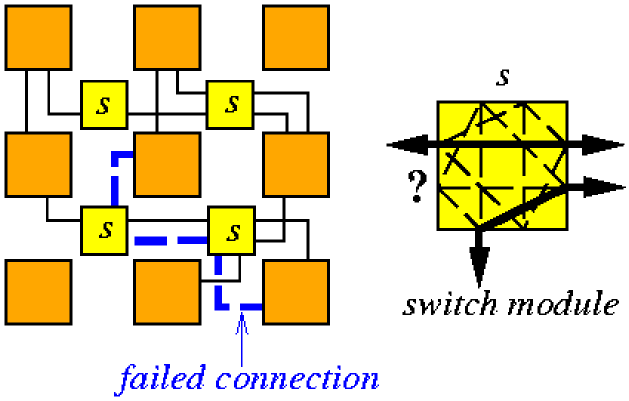
- FPGA routing must make use of prefabricated routing resources.



- #1 objective: 100% routing completion.
- Can be performed in two phases (*global routing*, *detailed routing*) or combined.

Routing in FPGA

- Must consider *switch-module architectural constraints*.

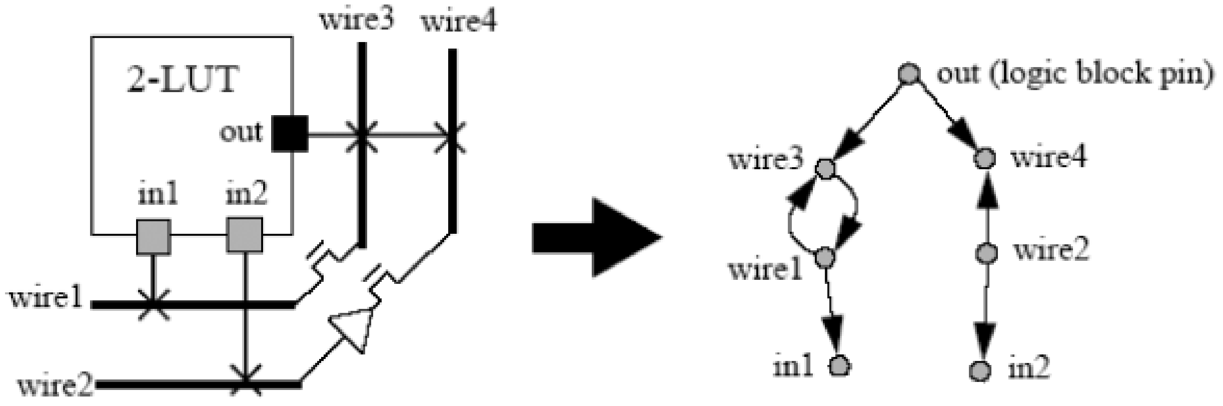


Each channel has a capacity of 2 tracks.

- For *performance-driven routing*,
  - □ **Minimize # of switches passed.**
  - □ Minimize the maximum wire length.
  - □ Minimize the maximum path length.

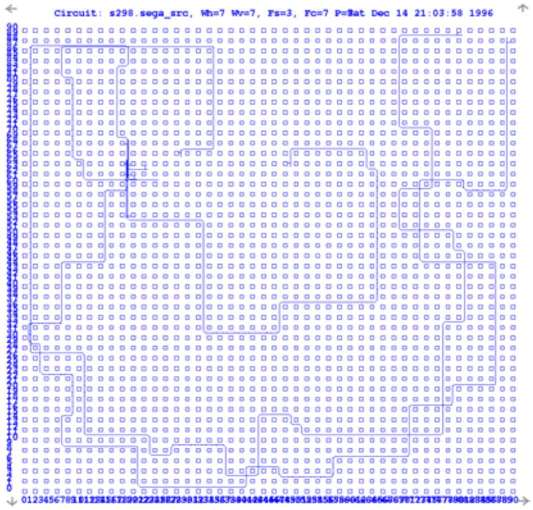
Routing-Resource Graph

- A *graph model* for routing
  - □ Wire/pin → node
  - □ Unidirectional switch → a directed edge
  - □ Bidirectional switch → two directed edges



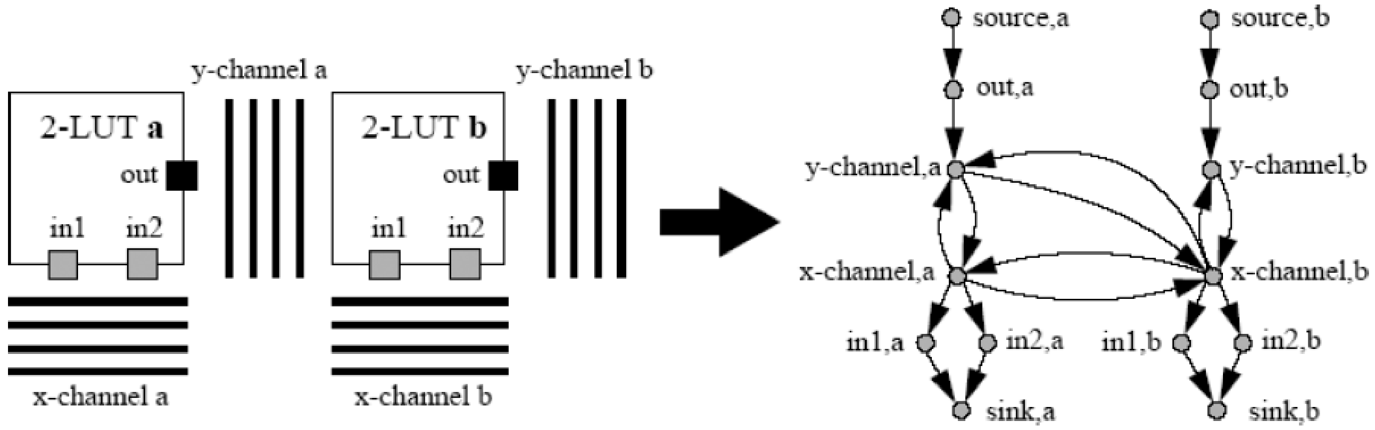
Global Routing

- Find a *routing tree* for each net.
- Select a set of channels, but not specific routing tracks.
- Subject to *channel capacity*.


Circuit: s298.sega_src, Wh=7 Wv=7, Fs=3, Fc=7 P=Sat Dec 14 21:03:58 1996

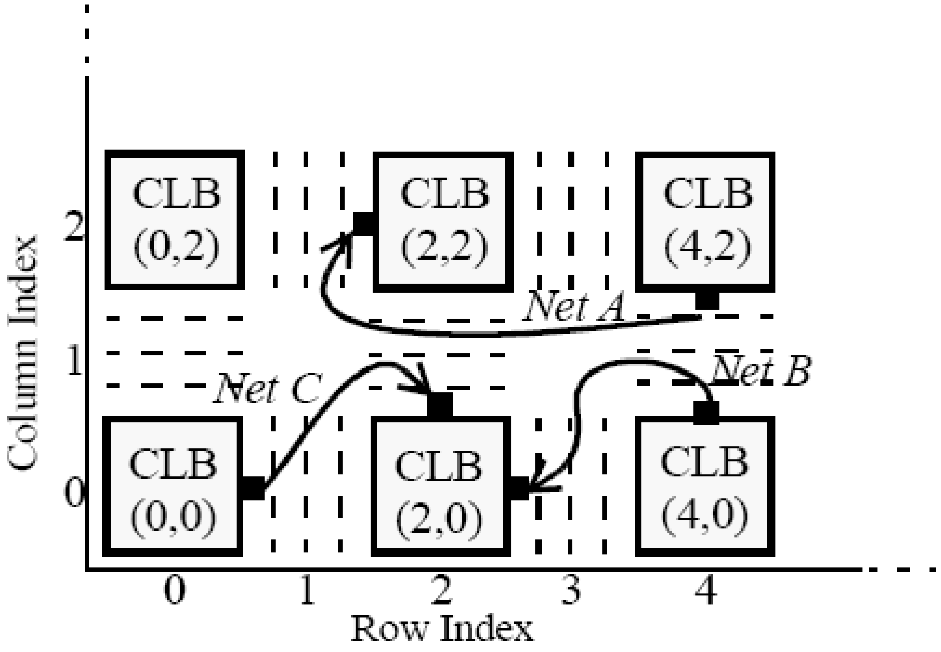
Coarse Routing Resource Graph

- A graph model for global routing
  - □ Associate a capacity to a node
  - □ Capacity of a node = corresponding channel capacity



Detailed Routing

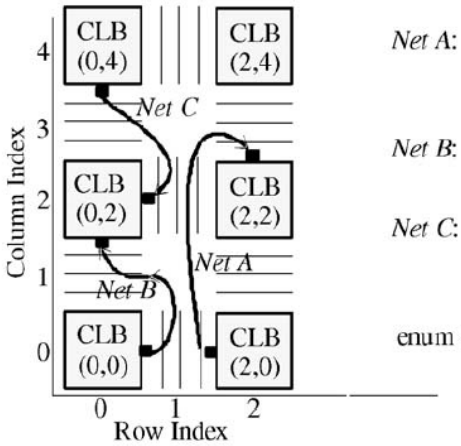
- Find a track assignment for each net under its given global routing configuration.



Detailed Routing via SAT

- Formulate a routing instance as a *Boolean satisfiability problem* in conjunctive normal form.
- Consider all nets simultaneously, and can prove unroutability.
- Connectivity constraints: each net must be connected.
- Exclusivity constraints: each track must be used by ≤ 1 net.
- Rely on efficient SAT-solver (SAT is NP-hard).

Detailed Routing via SAT



Net A: [pin 1 of CLB(2,0), C-block(1,0), S-block(1,1), C-block(1,2), S-block(1,3), C-block(2,3), pin 2 of CLB(2,2)]

Net B: [pin 3 of CLB(0,0), C-block(1,0), S-block(1,1), C-block(0,1), pin 0 of CLB(0,2)]

Net C: [pin 0 of CLB(0,4), C-block(0,3), S-block(1,3), C-block(1,2), pin 3 of CLB(0,3)]

enum {0, 1, 2} *AV*, *AH*, // Net A track variables
BV, *BH*, // Net B track variables
CH, *CV*; // Net C track variables

$$
\begin{aligned}
Conn(A) = {} & [(AV \equiv 0) \vee (AV \equiv 1) \vee (AV \equiv 2)] \wedge && \text{// Vertical channel 1} \\
& [AV = AH] \wedge && \text{// S-block(1,3)} \\
& [(AH \equiv 0) \vee (AH \equiv 1) \vee (AH \equiv 2)] && \text{// Horizontal channel 3}
\end{aligned}
$$

$$
\begin{aligned}
Excl(V1) = {} & (AV \neq BV) \wedge \\
& (AV \neq CV)
\end{aligned}
$$

Detailed Routing via SAT

- Pros and cons?

pro:
1. guarantee to find a feasible routing solution if the problem is feasible
   or to prove the problem is infeasible
2. easy to implement
3. leverage the advance in SAT solver

con:
1. just return one feasible routing solution but no consideration of the quality of the solution
2. can be slow
3. can be computationally expensive for detailed routing on a large network

- Attractive for routing highly congested network where finding feasible solution is hard but network size is not too large e.g. FPGA clock network, soft IP, inter-die interface of 2.5D FPGA

VPR Router

- Combined global and detailed routing.
- Routing algorithm based on PathFinder.
- Use maze expansion to construct routing tree from a signal’s driver to its loads in the routing graph.
- Congestion component is used to gradually resolve congestion by encouraging nets to take *detours around congested resources*.

Routing approaches
1. sequential routing
route each net one after another.
2. concurrent routing of all nets
3. iterative negotiation-based routing

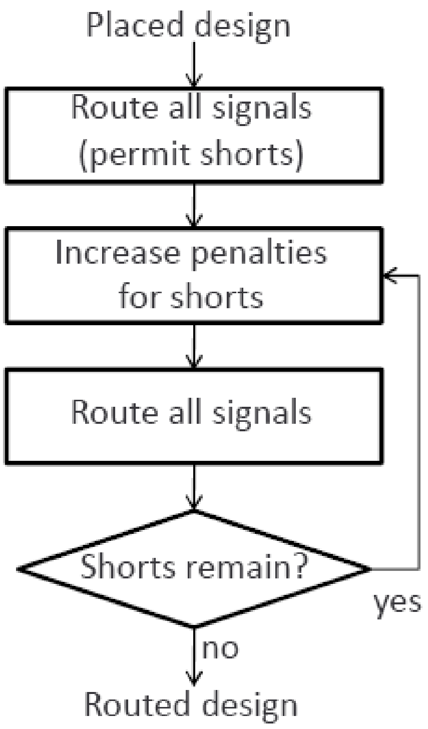
Details of VPR Router

sequential routing quality may be worse
因為沒辦法知道未來的線想要繞得怎樣
所以可能現在繞的佔據掉未來的好解

concurrent routing is too difficult

- VPR router is based on PathFinder which is an *iterative negotiation-based* routing approach.
- In each iteration, route all nets independently w/ minimum cost (nets may share same routing resource)
- Costs of over-congested routing resources will be increased.
- Nets that can use lower congestion alternatives are forced to do so in subsequent iterations.

Negotiated Congestion Routing Flow



Placed design

Route all signals (permit shorts)

Increase penalties for shorts

Route all signals

Shorts remain?

yes

no

Routed design

Details of VPR Router

- Cost of using a resource is based on its *current & historical congestion*.
- Cost of resource *n* is

$$c_n = (b_n + h_n) * p_n$$

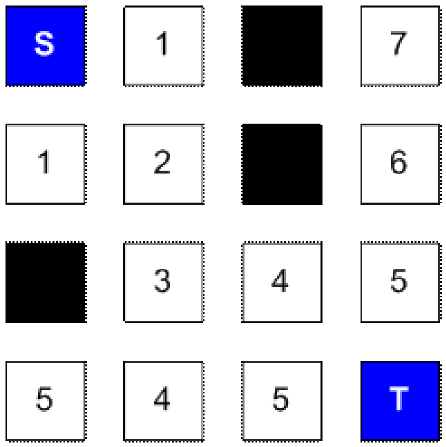
where b_n is the base cost for delay of using *n*,

p_n is #nets presently using *n*,

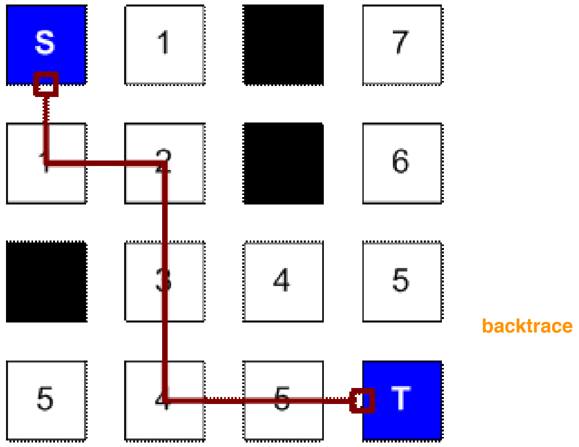
h_n is historical congestion of *n* (at *i*-th iteration, $h_n^i = h_n^{i-1} + 1$ if *n* has an overflow, or h_n^{i-1} otherwise)

Routing Two-Terminal Nets in VPR

- Use maze expansion to route two-terminal net:



Labelling (breadth-first traversal)



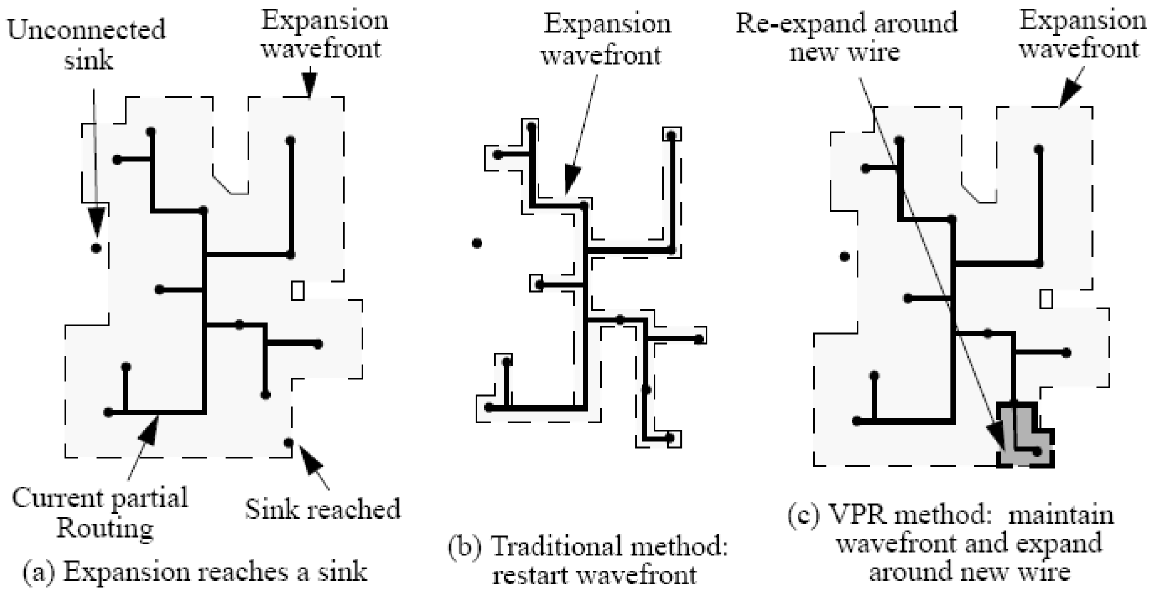
Connection after labelling

Routing Multi-Terminal Nets in VPR

- Use an *incremental* technique for faster multi-terminal net routing.

for a multi-pin net, want to compute a routing tree to connect all the pins

1. expand from source until the nearest target is reached
2. Then expand from the current partial routes until next nearest target is reached
3. repeat 2 until all targets are connected



(a) Expansion reaches a sink

(b) Traditional method: restart wavefront

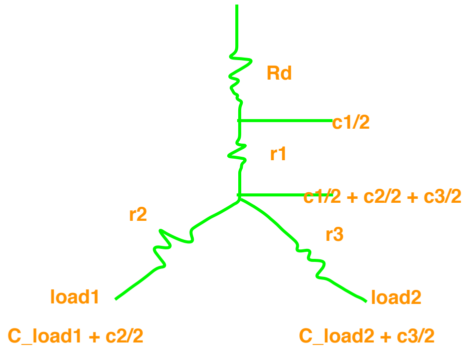
(c) VPR method: maintain wavefront and expand around new wire

Timing-Driven VPR Router

- Take delay into account
- Use *Elmore delay* model
- Initially, route all nets in minimum delay independently.
- Subsequently, use a weighted sum of *congestion* and *delay* as cost.

Elmore delay of a path = $\sum_{i \in path}$(resistance of element i * downstream capacitance of element i)

Rd*(c1+c2+c3+C_load1+C_load2)
+ r1*(c1/2 + c2 + c3 + C_load1 + C_load2)
+ r2*(c2/2 + C_load1)



Timing-Driven VPR Router

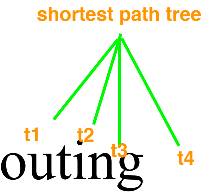
- At the end of each routing iteration, *timing analysis* is performed.
- From timing analysis, the timing *criticality* $Crit_{ij}$ of every connection *(i,j)* is computed as follows:

$$Crit_{ij} = 1 - Slack_{ij} / \mathrm{CriticalPathDelay}$$

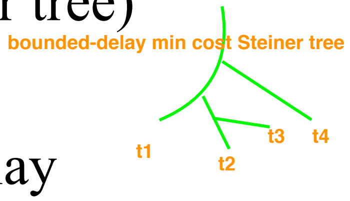
- Cost of using resource *n* for routing net *i* to its sink *j*:

$$Cost_n(i,j) = Crit_{i,j} * delay\_cost(n) + [1 - Crit_{i,j}] * cong\_cost_n$$

Timing-Driven Routing

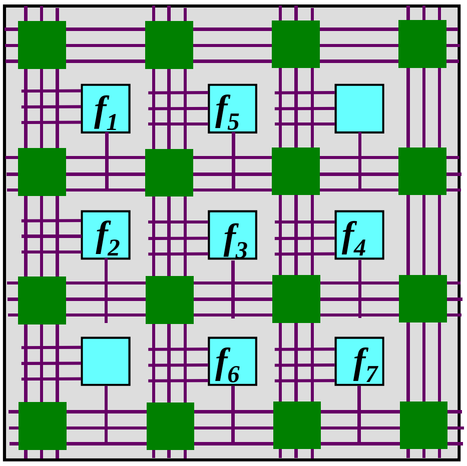


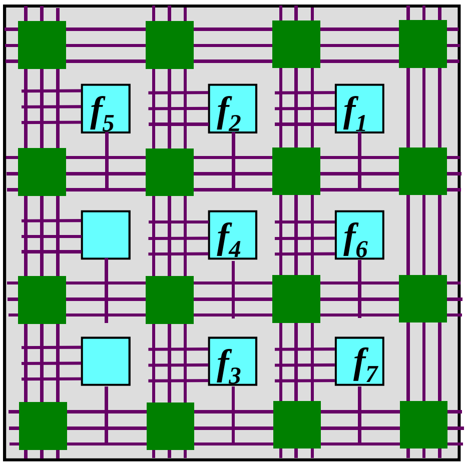
- Common techniques:
  - □ Route timing *critical nets first* for sequential routing
  - □ Routing tree *topology optimization* (e.g. shortest-path tree, bounded-delay minimum-cost Steiner tree)
  - □ *Delay penalty* (e.g. VPR)
  - □ *Static slack distribution* given the path delay constraints
  - □ *Dynamic net weighting* (e.g. VPR)



Placement Evaluation

- One often needs to evaluate a placement e.g. SA will generate many placement solutions.
- Interested to *estimate routability and/or performance* quickly.





References

- "A Comparative Study of Two Boolean Formulations of FPGA Detailed Routing", *IEEE Trans. on Computers, June 2004.*
- "Boolean Satisfiability-Based Routing and Its Application to Xilinx UltraScale Clock Network", in *FPGA'16*.
- "SAT Based Place-And-Route for High-Speed Designs on 2.5D FPGAs", in *FPT'18*.
- "PathFinder: A Negotiation-Based Performance-Driven Router for FPGAs", in *FPGA'*95.
- "VPR: A New Packing, Placement and Routing Tool for FPGA Research," in *FPL'97*
- "A Fast Routability-Driven Router for FPGAs", in *FPGA'98*