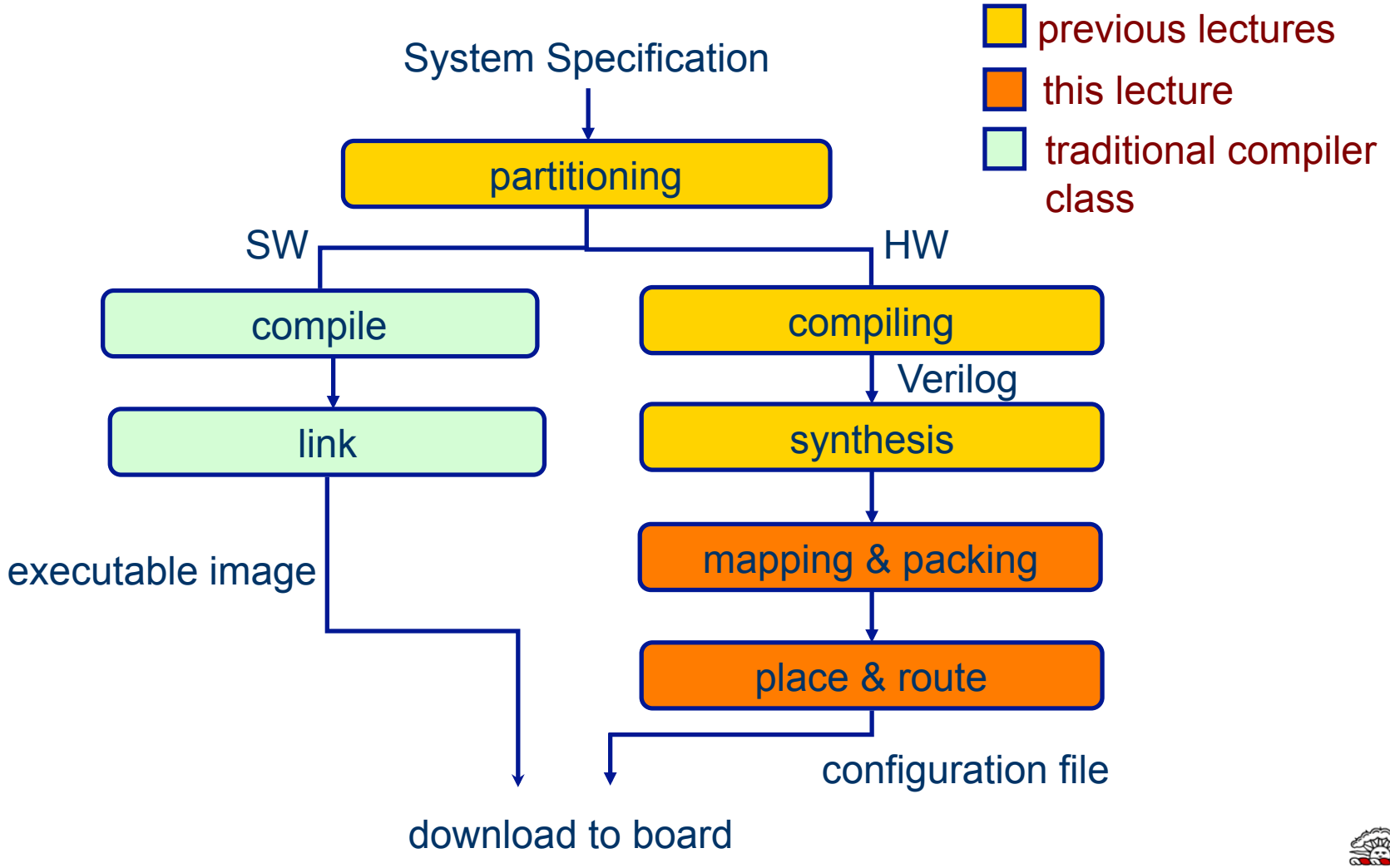
EN2911X: Reconfigurable Computing
Lecture 13: Design Flow: Physical Synthesis (5)

Prof. Sherief Reda
Division of Engineering, Brown University
http://scale.engin.brown.edu
Fall ’09

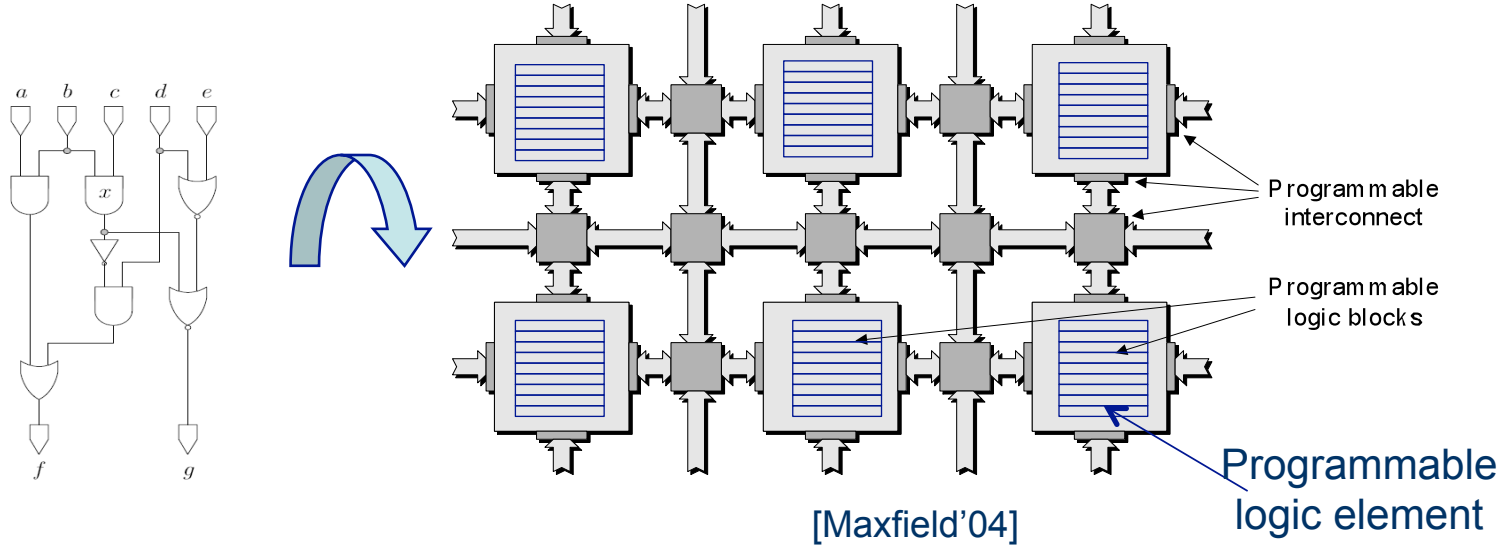
BROWN

Summary of the last few lectures



BROWN

Embedding a digital circuit to FPGA fabric

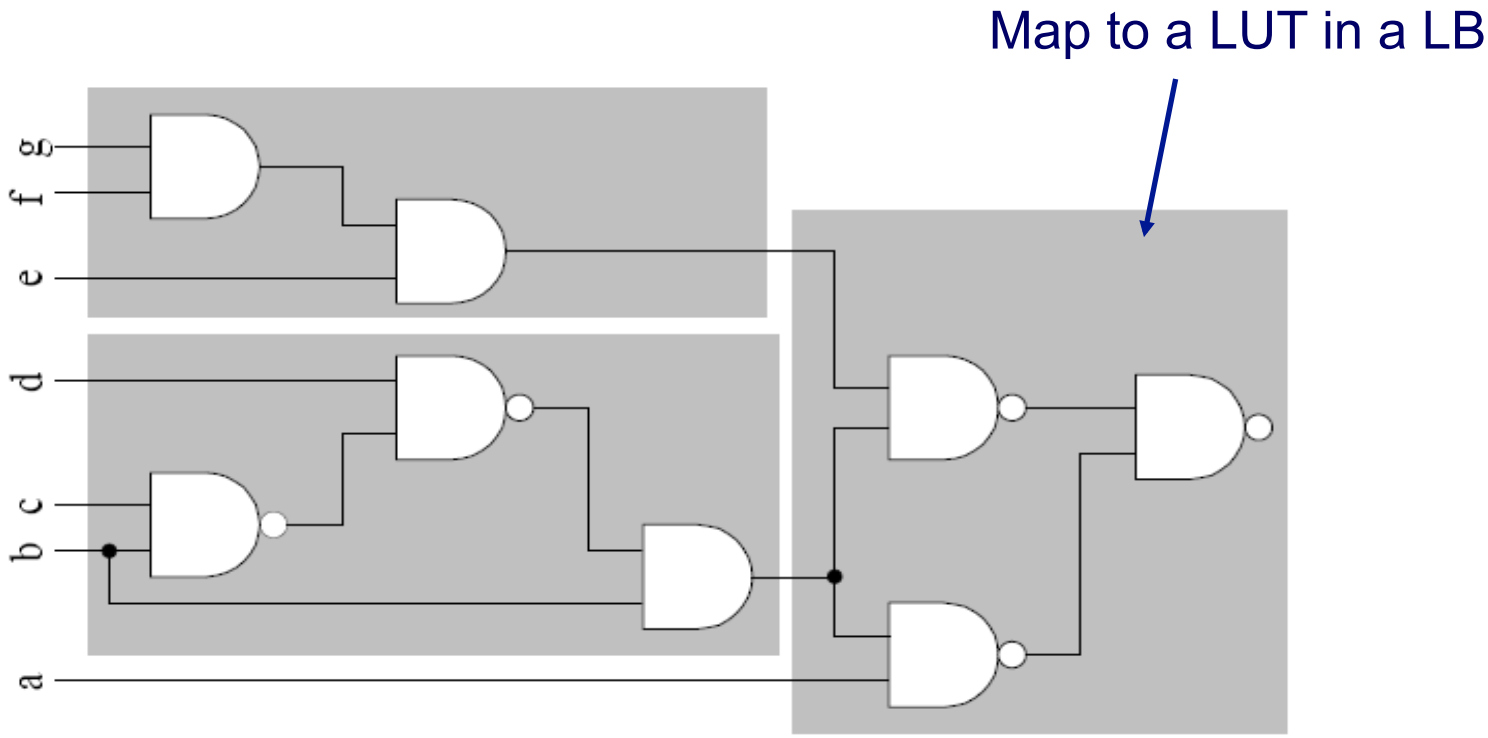


1. Mapping decomposes the circuit into logic sections and flip-flops such that each section fits into a K-LUT LE.
2. Packing groups LEs into clusters so that each cluster fits into a LAB
3. Placement determines the position of each cluster into the LABs of the island style FPGA
4. *Routing* determines the exact routes for the communicating LE/ LABs

What are the objectives/metrics that these algorithms should pursue?

BROWN

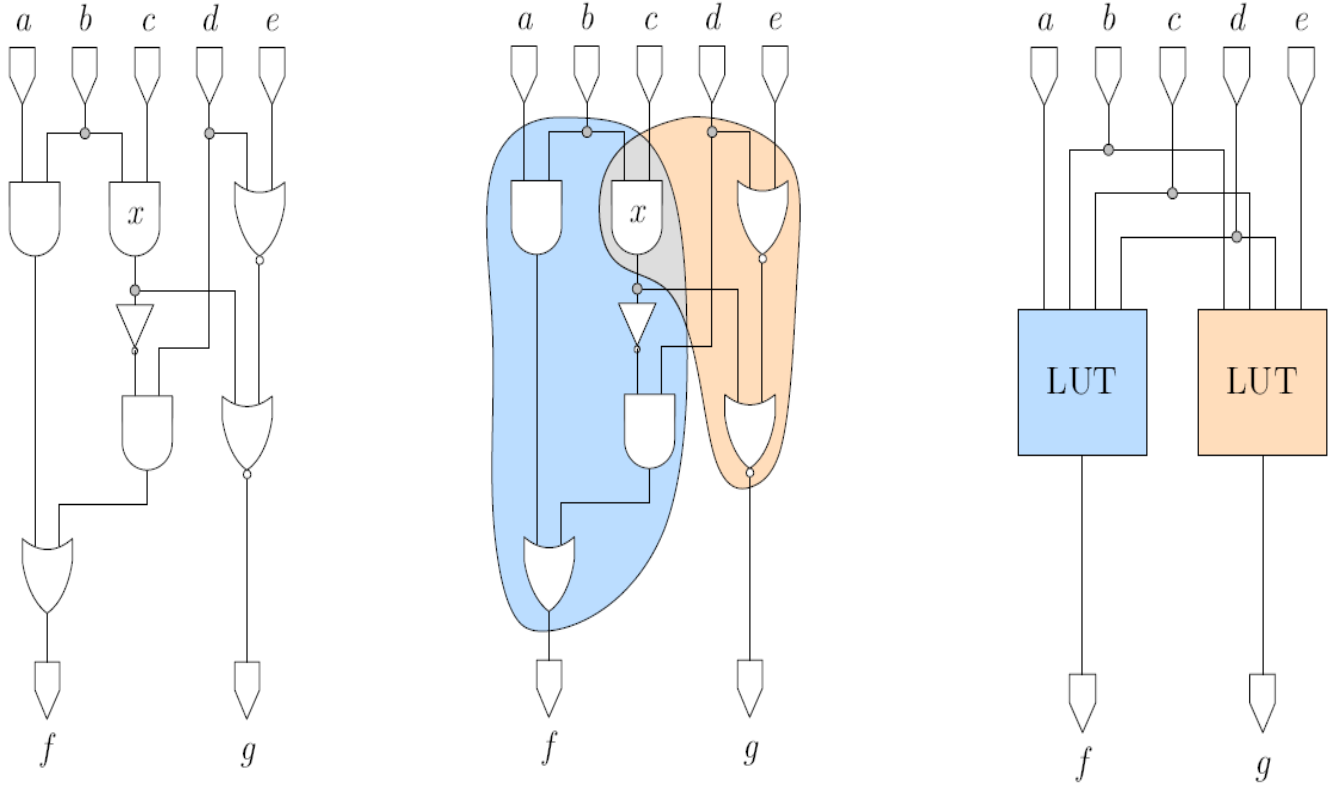
1. Mapping finds a covering for a given circuit using K-LUT



[Figure form Cong FPGA'01]

BROWN

A covering example

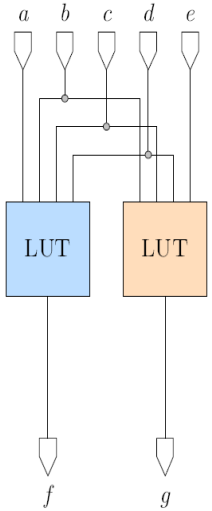


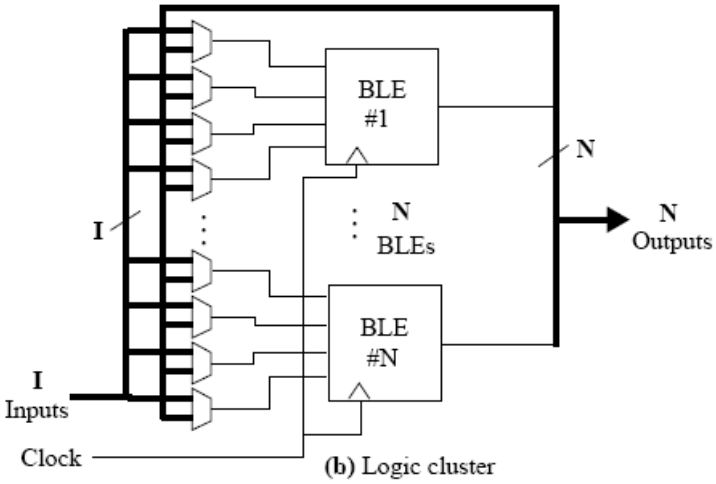
[From Ling et al. DAC'05]

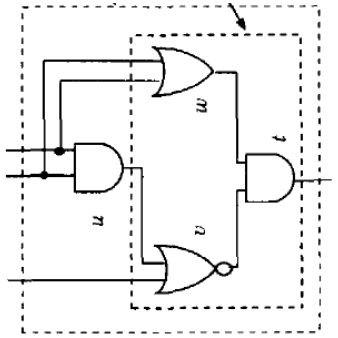
There could be many possible covering? Which one should be picked?

BROWN

2. Packing







How can we decide which LEs should go together in the same logic cluster?

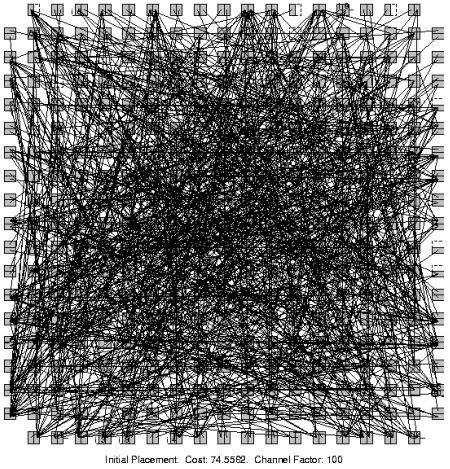
Possible method (VPACK): Construct each cluster sequentially

- Start by choosing seed LE for the cluster
- Then greedily selects the LE which shares the most inputs and outputs with the cluster being constructed
- Repeat the procedure until greedily until the cluster is full or the number of inputs exceed the limit I
- Can addition of a LE to a cluster reduces the number of distinct inputs?

BROWN

3. Placement

- Placement assigns an exact position or LAB for each cluster in the input netlist
- Suppose you start with a random placement, how can you improve it?

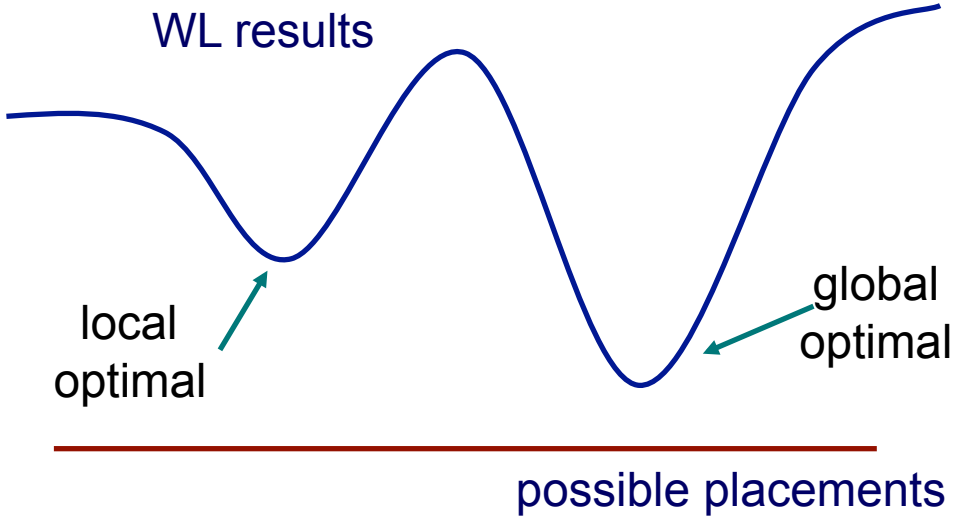


Possible algorithm:

- Pick a pair of cells and swap their locations if this leads to reduction in WL

What's wrong with the previous greedy algorithm?

$\Rightarrow$ It can simply get stuck in a local optimal result

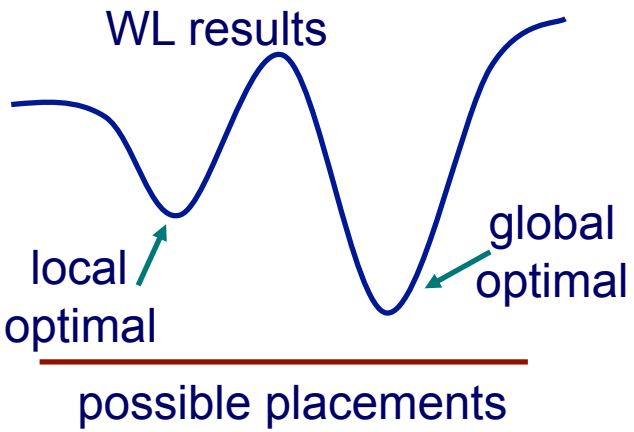


BROWN

Simulated annealing allows us to avoid getting trapped in a local minima

Modified algorithm

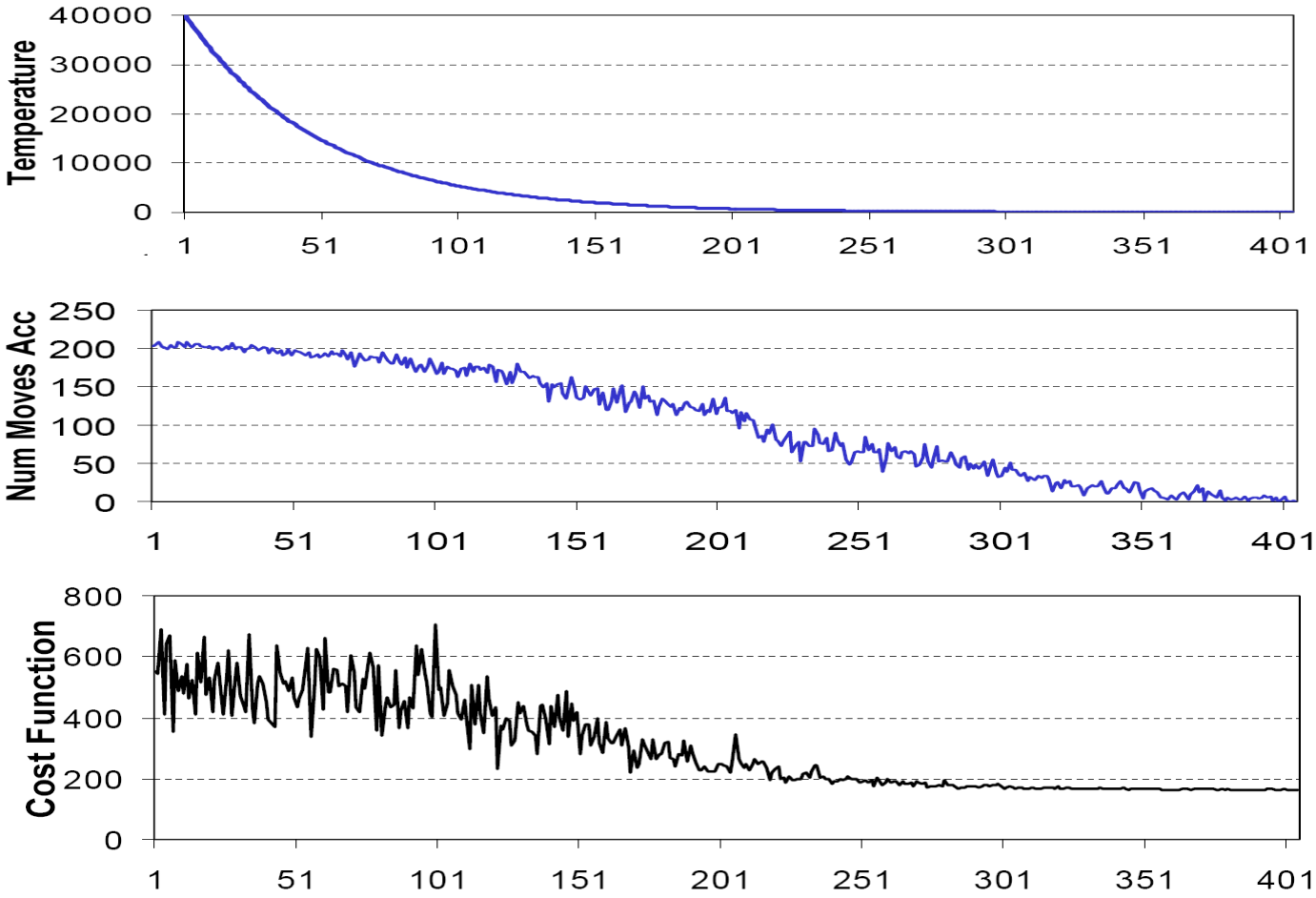
- Generate a random move (say a swap of two cells)
  - calculate the chance in WL (ΔL) due to the move
  - if the move results in reduction ($\Delta L < 0$) then *accept*
  - else reject with probability $1-e^{-\Delta L/T}$



- T (temperature) controls the rejection probability
- Initially, T is high (thus avoiding getting trapped early in a local minima) then the temperature *cools down in a scheduled* manner; at the end, the rejection probability is 1
- With the right "slow-enough" cooling scheduling, simulated annealing is guaranteed to reach the global optimal

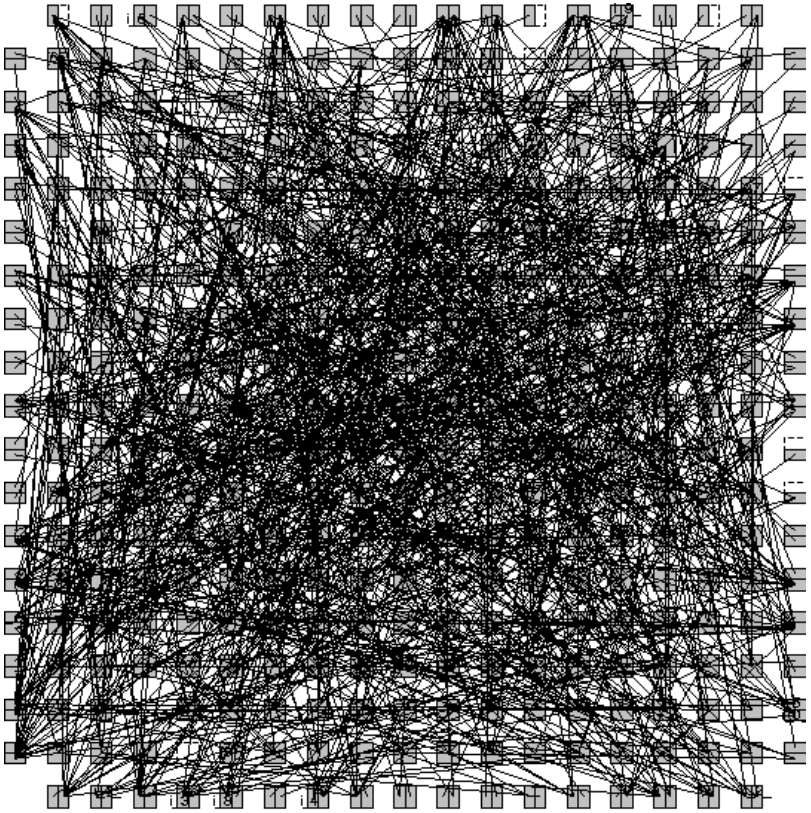
BROWN

How do the cooling scheduling and corresponding cost functions look like?

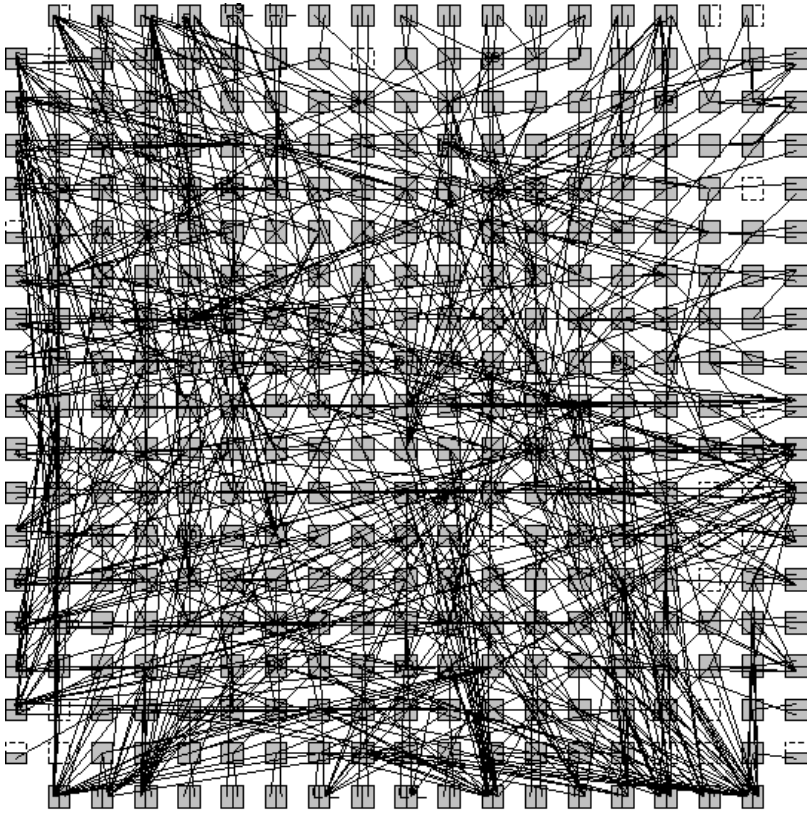


[source: I. Markov]

Placement before & after simulated annealing



Initial Placement. Cost: 74.5562. Channel Factor: 100



Final Placement. Cost: 28.5384. Channel Factor: 100

[using VPR tool]

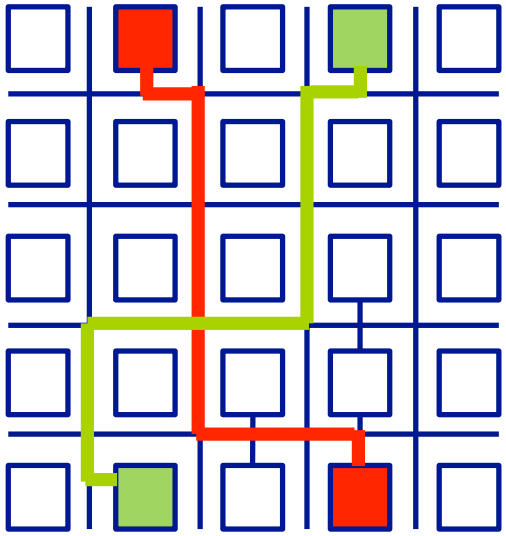
BROWN

4. Routing

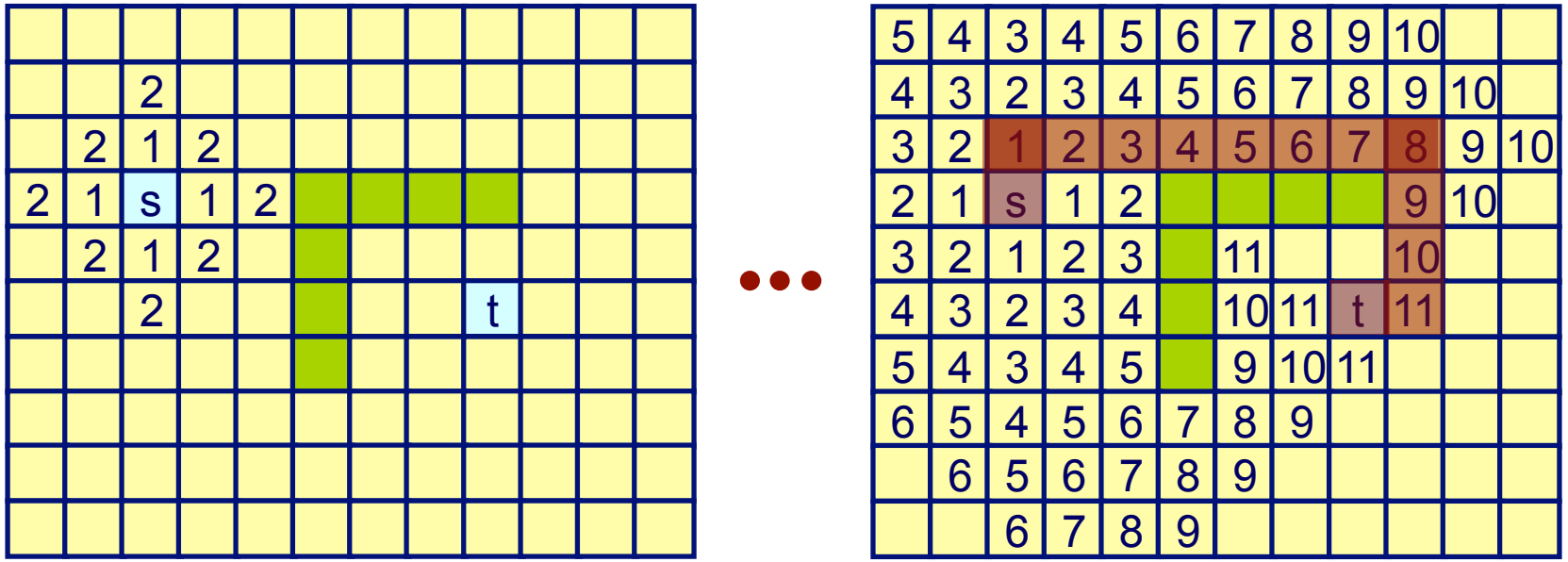
Assign exact routes for each wire in the given circuit in the FPGA fabric such that no two wires overlap

General idea:

- Order the wires according to some criteria
- Sequentially route each wire using shortest path algorithms (after removing the resources consumed from preceding routed wires)



Maze routing



Problem: Find the shortest path for a 2-pin wire from s to t

- grid cell capacity is full
- grid cell still has available tracks

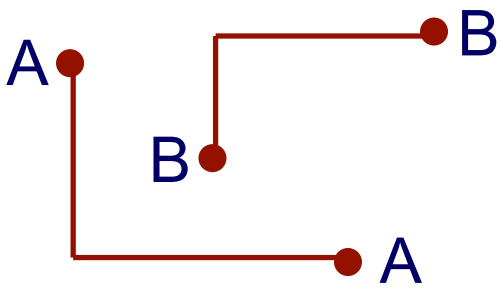
Speed ups are possible using A* search algorithms and other AI search techniques

BROWN

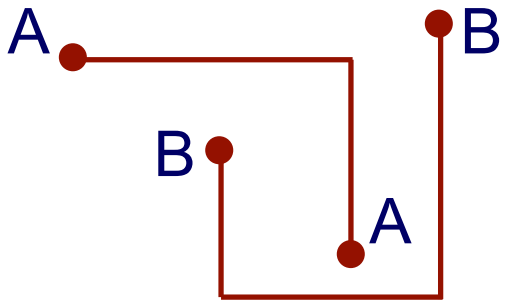
Impact of Net Ordering

- A bad net ordering
  - may unnecessarily increase the total wirelength
  - or even yield the chip unroutable!

- Example: Two nets A and B



B first then A
(Good order)



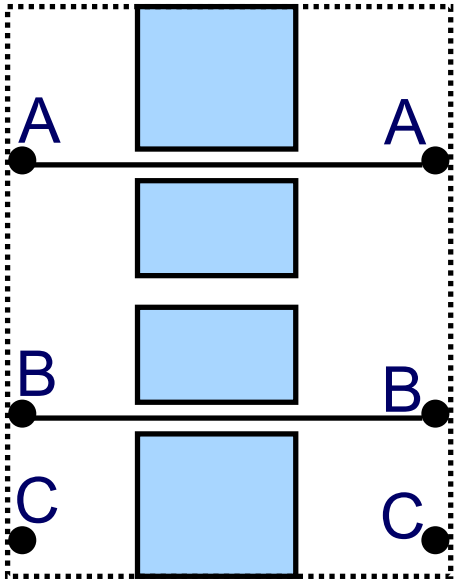
A first then B
(Bad order)

- Length in placement
- Timing criticality

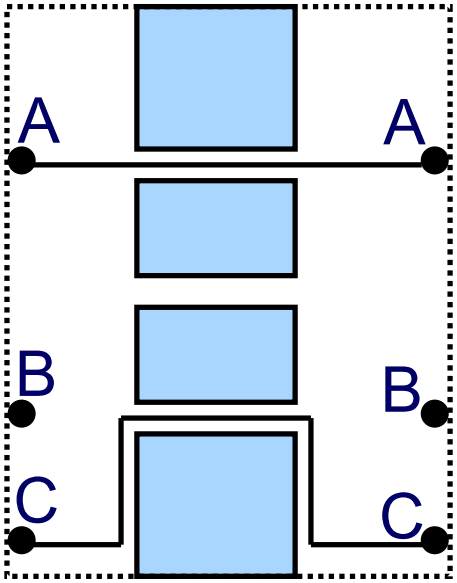
BROWN

When a route for a net can't be found then rip up and re-route

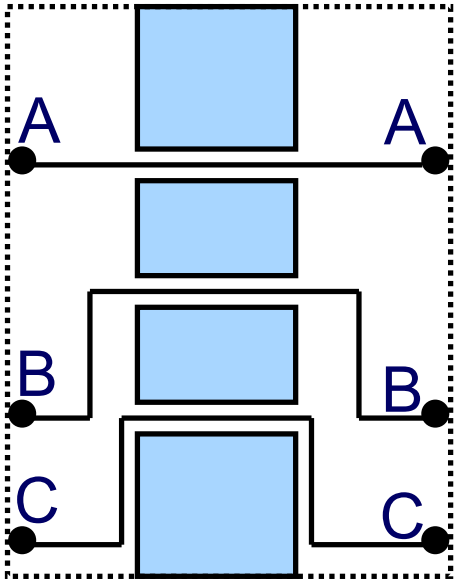
Cannot route C



So rip-up B and route C first.

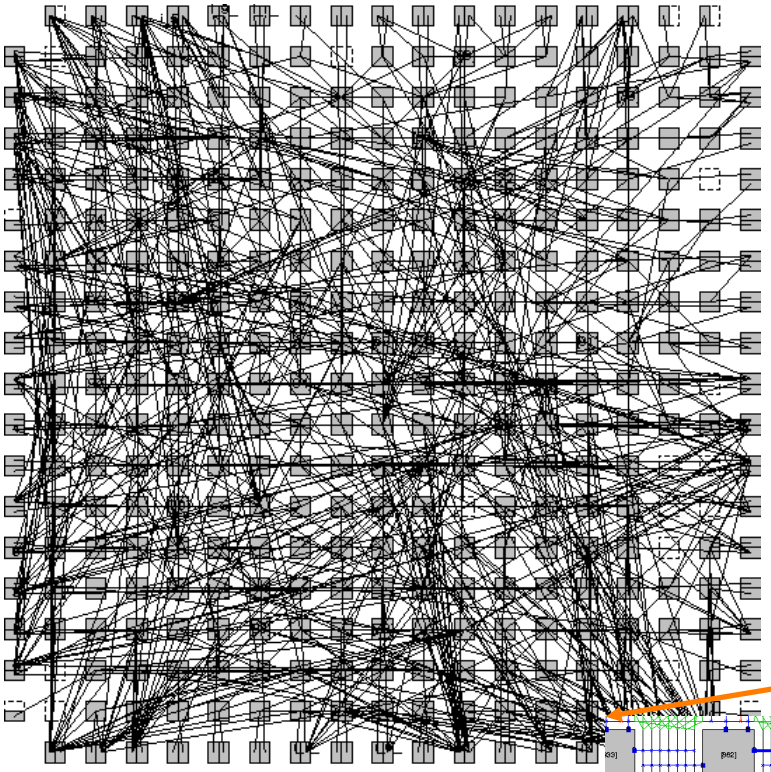


Finally route B.

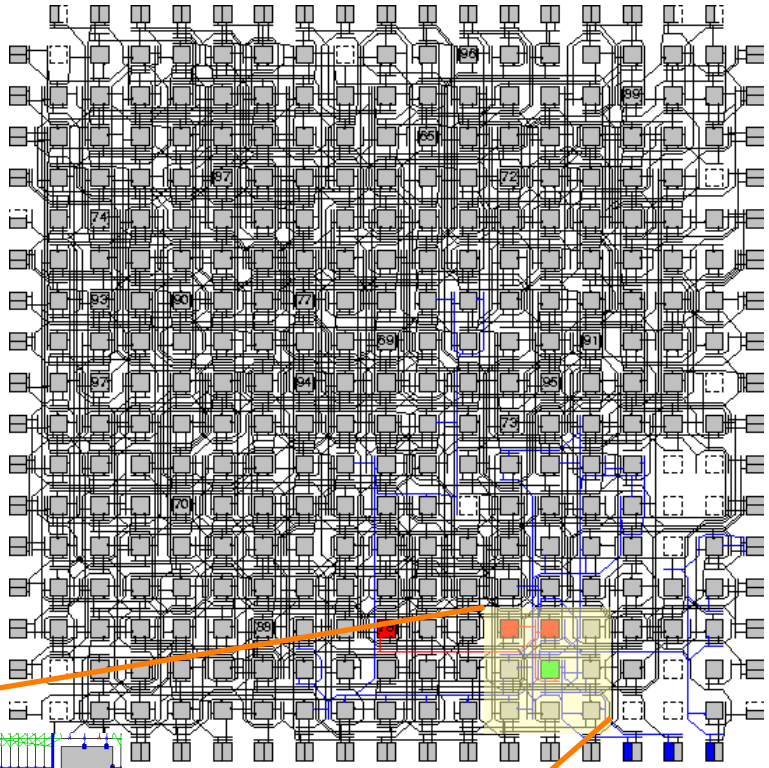


[Example from Prof. D. Pan Lecture]

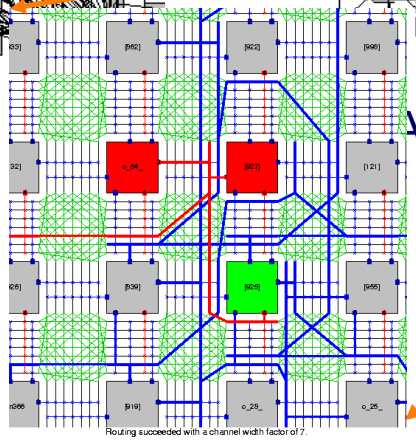
VPR. After routing



After placement



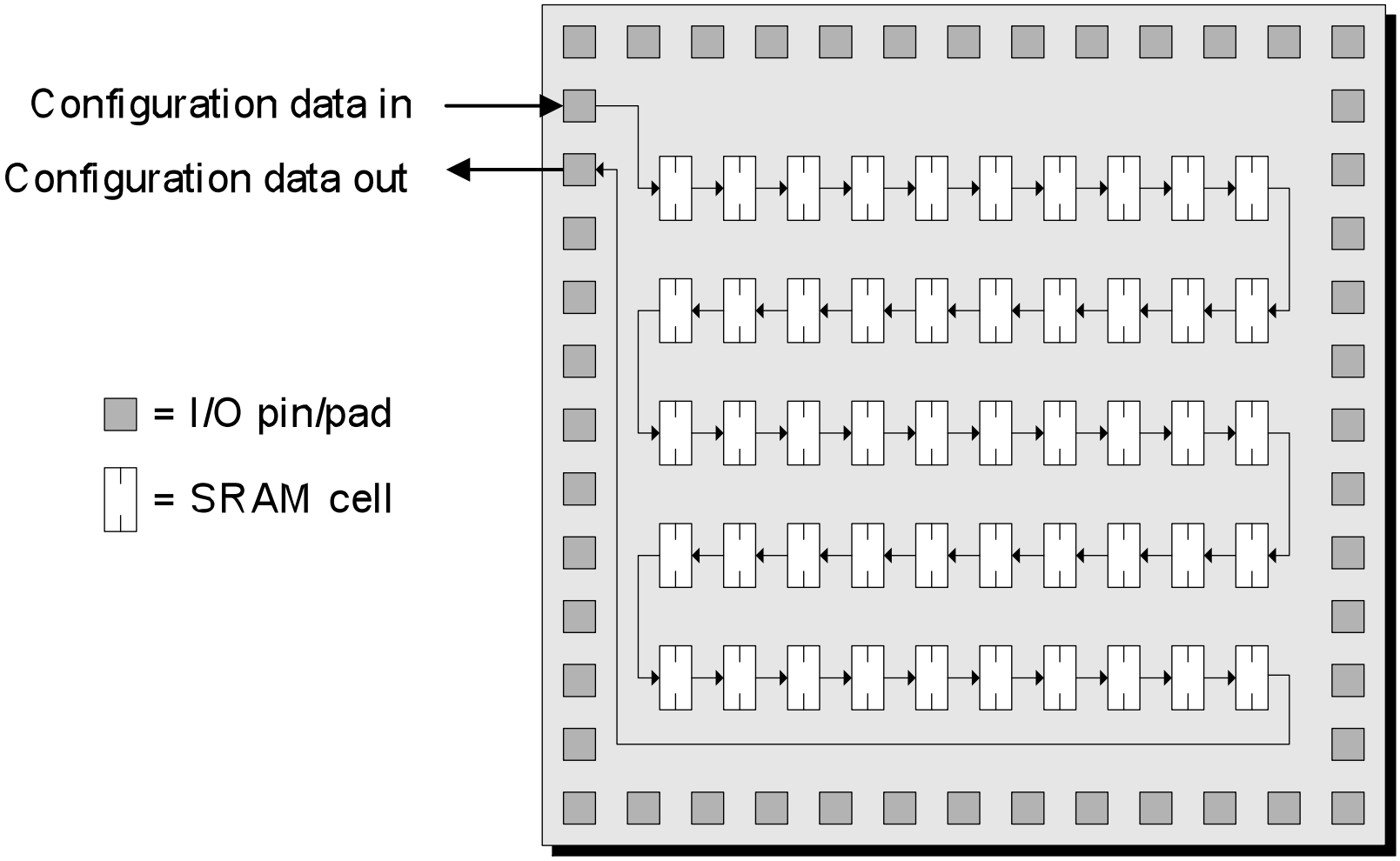
After placement and routing



You probably saw similar layouts from the Quartus II tool

BROWN

Finally programming the FPGA



BROWN