FPGA Logic

Nachiket Kapre
nachiket@uwaterloo.ca

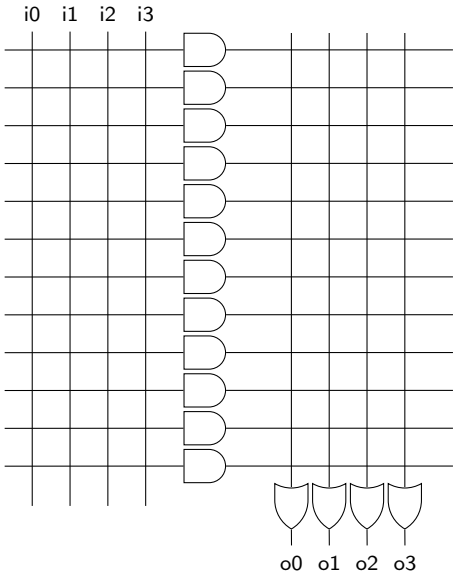
Lecture Outline

- **PLAs and LUTs** – Beginning of programmable logic fabrics
- **LUT Clusters** – Need for packing multiple LUTs together

Properties of Circuits

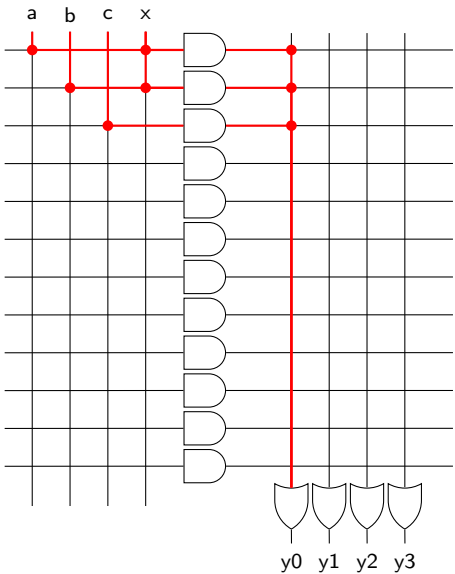
- Expressed as collection of boolean gates connected by wires
  - Arithmetic operations typically not expressed at gate-level
- Flexibility in choices of gate + pattern of wire connections
- Reconvergent fanout
- Pipelining (periodic insertion of registers)

Early Configurable Logic



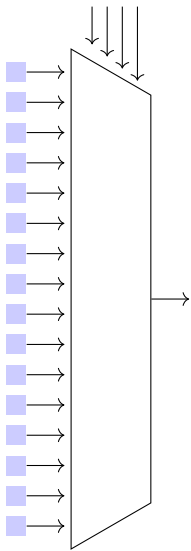
- PLAs (programmable logic arrays) provide some configurability
- Each crosspoint is programmable
  - Visualize a truth table overlayed on top of the AND crosspoints
- The number if `pterms` is often $<<$ 2^{inputs}.
  - A k-LUT, in contrast can support 2^k pterms with no extra cost
- PLAs are cheaper/simpler starting points

Implement poly on a PLA



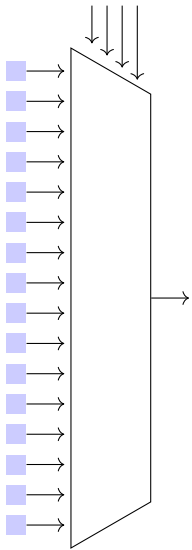
- Logic equation to map:
  $a \cdot x \cdot x + b \cdot x + c = a \cdot x + b \cdot x + c$
- Problem has 4 Inputs, 3 Product Terms, One Output
- Map two expression to the AND plane
  - $and0 = a \cdot x \cdot x = a \cdot x$
  - $and1 = b \cdot x$
  - $and2 = c$
- Map the final result to the OR plane
  - $y0 = and0 + and1 + and2$

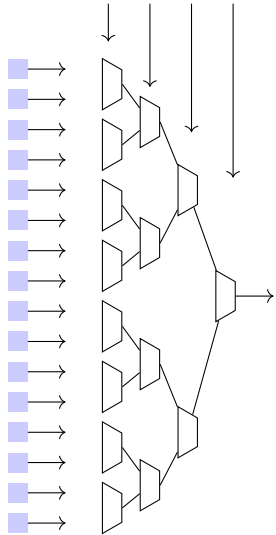
Logic Elements

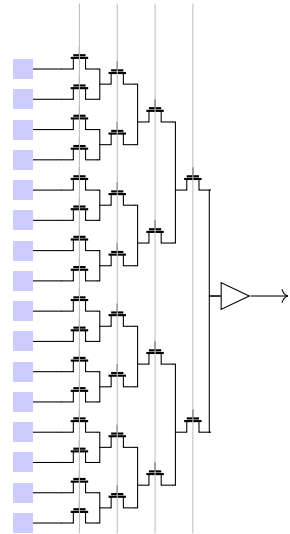


- Boolean table mapped to Configuration cells (SRAMs, or fuses).
- k-input table has 2^k cells.
- Actel FPGAs have cheaper one-time use fuses.
- Tabula failed at using deeper configuration memories (circuit changes per cycle)
- Samsung/Stanford DRAM-based FPGAs (must worry about refresh)

Logic Elements (Pass gates)







select lines are alternately inverted

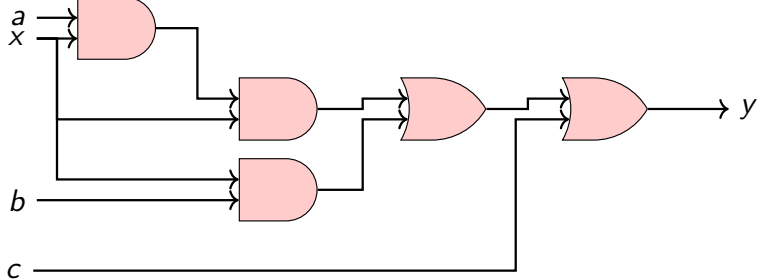
Area/Delay Estimates

- $A_{LUT} = A_{Config} * 2^k + A_{Mux} * (2^k - 1) + A_{FF}$
  $\approx 2^k * (A_{Config} + A_{Mux}) + A_{FF}$
  - 2^k cells are configured as RAMs
  - Need $2^k - 1$ muxes as well
  - An SRAM cell typically needs 6–8 transistors
  - A pass-gate implementation of 2:1 Mux takes 2 transistors
  - A DFF takes 16 transistors
  - Depending on load, larger transistors needed
- $T_{LUT} = k \cdot T_{Mux}$
  - k levels of multiplexer hierarchy

How to choose k?

- If k is too small, a large gate must be split across multiple LUTs.
  - Increases area used by circuit
  - Adds delay due to composition of multiple LUTs
  - Interconnect is expensive, absorb wires where you can
- If k is too large, need to allocate silicon area for 2^k SRAM cells.
  - Logic synthesis may generate gates that are at most 7 or 8-input
  - Most gates may be 2-input (distribution varies with application, logic synthesis optimizations)
  - Underutilization of silicon area due to over-provisioning of LUT capacity

Mapping poly to LUTs for different goals

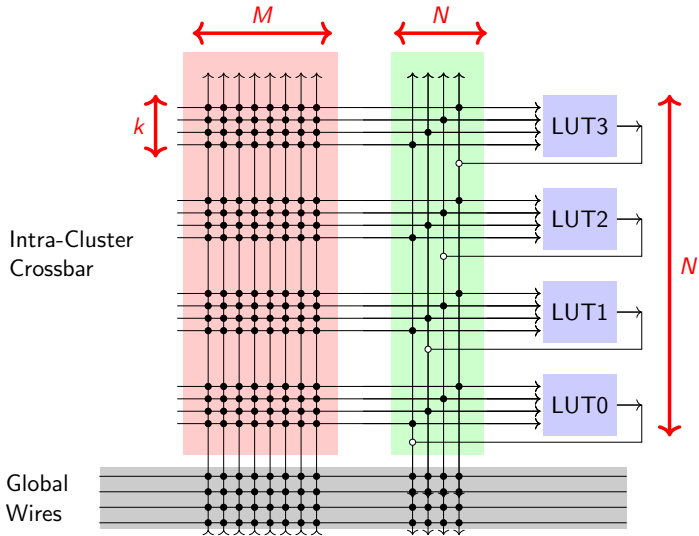


- k=3 $\rightarrow$ two LUTs
  - $A_{k=3} = 2 \cdot A_{3LUT} = 2 \cdot (A_{Config} * 2^3 + A_{Mux} * (2^3 - 1) + A_{FF}) =$ $16 * A_{Config} + 14 * A_{Mux} + 2 * A_{FF}$
  - $T_{k=3} = 2 \cdot T_{3LUT} = 2 \cdot 3 \cdot T_{Mux} = 6 \cdot T_{Mux}$
- k=4 $\rightarrow$ one LUT
  - $A_{k=4} = A_{4LUT} = A_{Config} * 2^4 + A_{Mux} * (2^4 - 1) + A_{FF} = 16 * A_{Config} + 15 * A_{Mux} + A_{FF}$
  - $T_{k=4} = T_{4LUT} = 4 \cdot T_{Mux}$
- $A_{k=3} - A_{k=4} = A_{FF} - A_{Mux}$
- $T_{k=3} - T_{k=4} = 2 \cdot T_{Mux}$

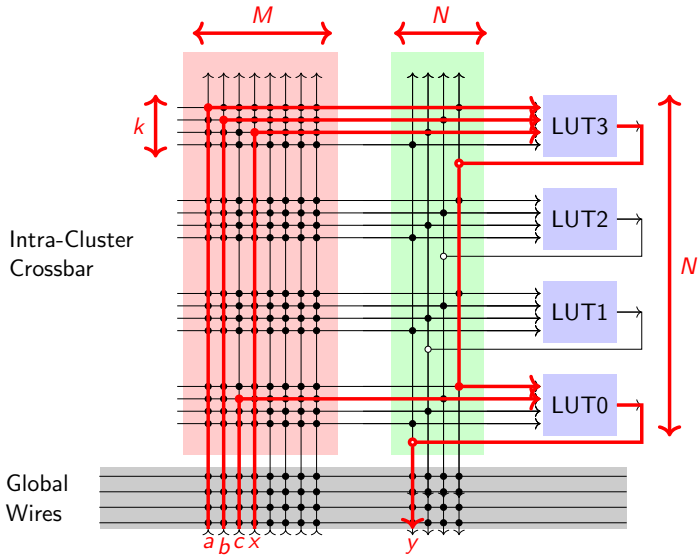
Clustering FPGA LUTs into Logic Blocks

- Exposing k LUTs creates $k+1$ connections between logic and routing.
  - Can we reduce this without sacrificing connectivity?
  - Real-world circuits exhibit locality anyway. Can we exploit this?
- Increasing k costs us FPGA area (2^k SRAM cells).
- Clusters are small collections of N LUTs with separate local interconnect
  - Pay $N \cdot 2^k_{small}$ area cost instead of 2^k_{large}.
  - Assume some inputs are common to the N k_{small} LUTs. Thus we do not need $N \cdot k_{small}$ IOs from cluster.
  - Intra-cluster wires as faster, and can be richer
  - CAD is faster as it can be split into global and local phases

Clustering FPGA LUTs into Logic Blocks



Clustering FPGA LUTs into Logic Blocks



Area Estimates

- $A_{Cluster} = N * A_{kLUT} + A_{Green} + A_{Red}$
- $A_{Green} = N * k * (N - k + 1) * A_{sw}$
  - Output needs to connect to any one input of k-LUT
  - LUT inputs are all equivalent, sufficient to provide connection to single LUT pin
- $A_{Red} = k * N * M * A_{sw}$
  - Internal routing block (Red) is a crossbar.
  - In the worse case, $M = k * N$
  - Typical value $M = \frac{k}{2} * (N + 1)$
- $A_{sw} = A_{config} + A_{Mux}$

Putting it all together $A_{Cluster}$

$$
\begin{aligned}
&A_{Cluster} \\
&= N * A_{kLUT} + A_{Green} + A_{Red} \\
&= N \cdot (2^k \cdot A_{Config} + 2^k \cdot A_{mux} + A_{FF}) + N \cdot k \cdot (N - k + 1) \cdot (A_{Config} + A_{Mux}) \\
&+ k \cdot N \cdot M \cdot (A_{Config} + A_{mux}) \\
&= N \cdot (2^k \cdot A_{Config} + 2^k \cdot A_{mux} + A_{FF}) + N \cdot k \cdot (N - k + 1) \cdot (A_{Config} + A_{Mux}) \\
&+ k \cdot N \cdot \frac{k}{2} \cdot (N + 1) \cdot (A_{Config} + A_{mux}) \\
&= N \cdot (2^k + k \cdot (N - k + 1) + \frac{(N + 1) \cdot k^2}{2}) \cdot A_{Config} \\
&+ N \cdot (2^k + k \cdot (N - k + 1) + \frac{(N + 1) \cdot k^2}{2}) \cdot A_{mux} + N \cdot A_{FF}
\end{aligned}
$$

- Area of the cluster scales as a function of k, N and M
- Substituting terms from previous slides, we get a result that grows as $O(N^2 \cdot k^2)$

External Interconnect (Peek)

- Area of Grey region will be covered in detail in the Interconnect lecture.
- $A_{Gray} = M * W * F_{cin} * A_{sw} + N * W * F_{cout} * A_{sw}$
- External routing block (cbox) is partially connected with F_{cin} and F_{cout} parameters
- W is the global channel width of the FPGA *i.e.* number of wires in the global interconnect channel beside the cluster

How to choose cluster size N?

- If N is too small, we do not offload sufficient wiring costs in the global interconnect.
  - If connectivity between cluster to routing tracks is poor, we will waste LUTs that are unreachable
- If N is too large, the intra-cluster routing area can become prohibitively expensive
- How do we choose N to balance these effects?

Class Wrapup

- Underling FPGA resources can be modeled using a simple analytical approach
- Heuristic/Experimental technique to determine best balance between N and k parameters of the resource model
- Example of engineering tradeoff $\rightarrow$ more compute, or more interconnect?