

ECE 412: Microcomputer Laboratory

Lecture 3: Introduction to FPGAs

Objectives

- Understand the basics of how FPGAs work
 - So that the CAD tools make sense to you

Review Questions

- This semester, as quiz preparation, I hope to begin each lecture with one or two questions based on the last lecture's material.

*The answers will be included, and both posted on the web in the **LECTURES** portion, as usual. We will spend a few minutes discussing the answers, and any other issues or items not clear from the earlier lectures; feel free to ask anything!*

Programmable Logic Device Advantages

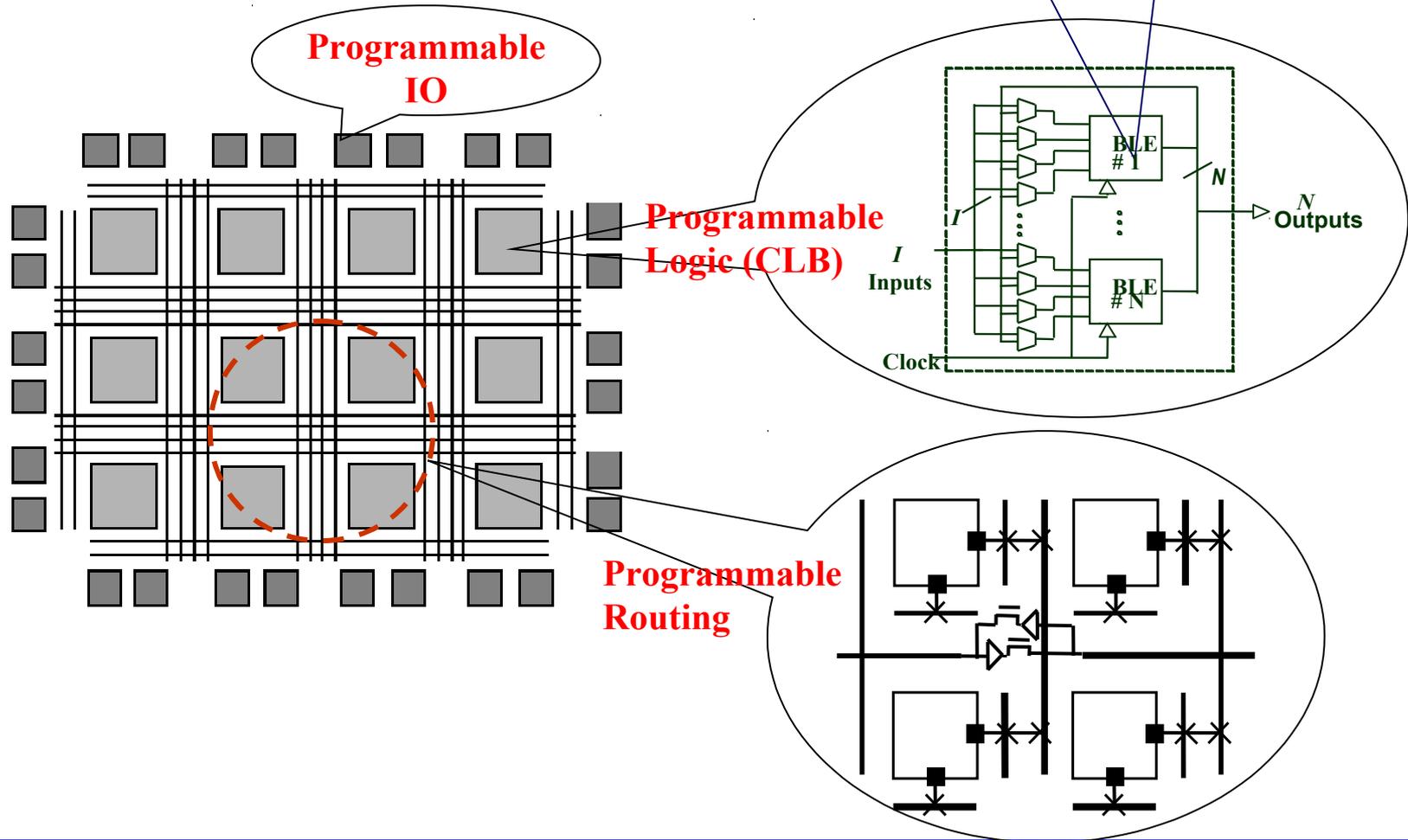


- Short TAT (total turnaround time)
- No or very low NRE (non-recurring expenses)
- Field-reprogrammable
- Platform-based design

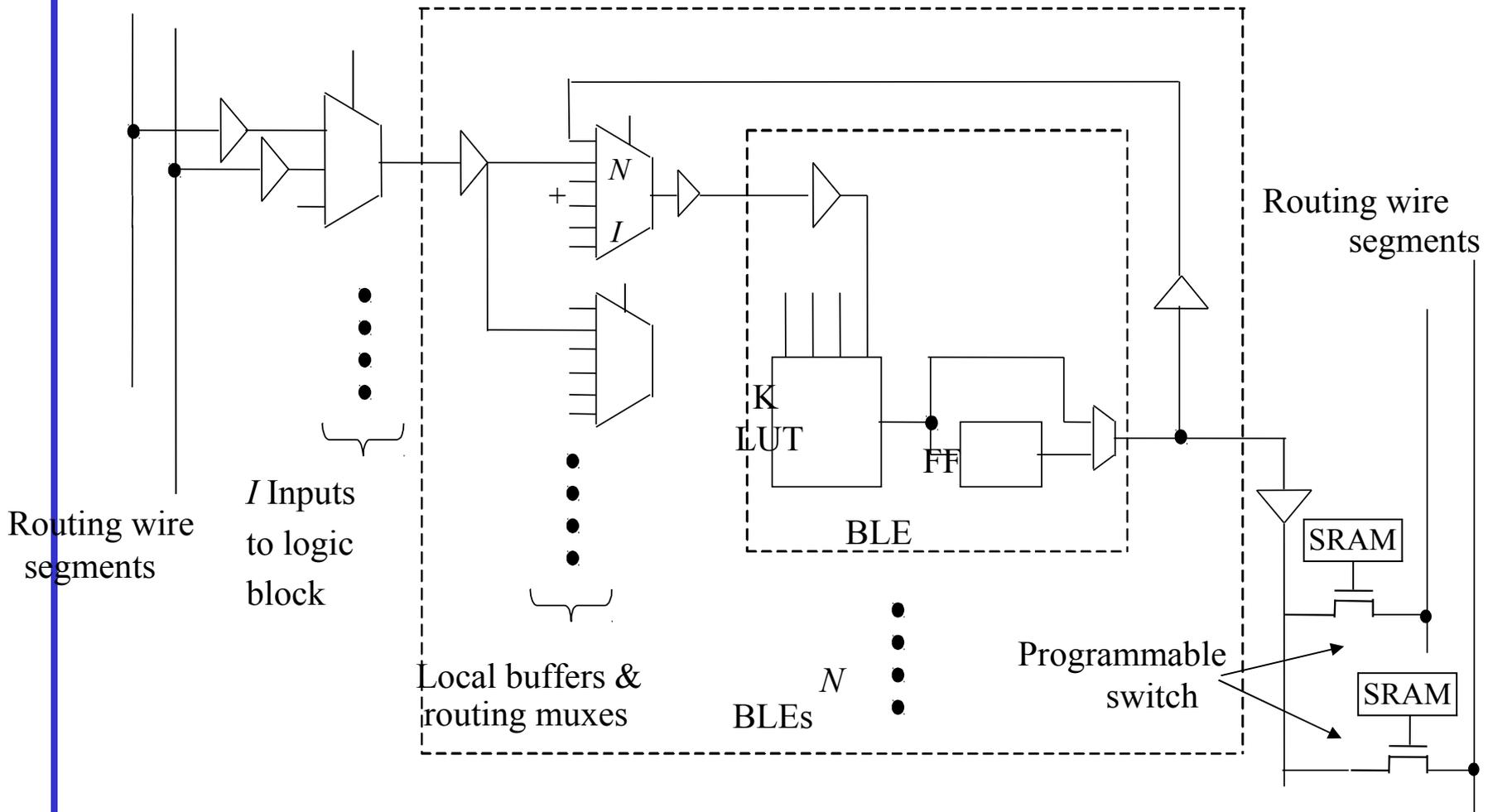
Today - Two Major Types of Programmable Logic

- CPLD (complex programmable logic device)
 - coarse-grained two-level AND-OR programmable logic arrays (PLAs)
 - fast and more predictable delay
 - simpler interconnect structures
- FPGA (field programmable gate array)
 - fine-grained logic cells
 - high logic density
 - good design flexibility

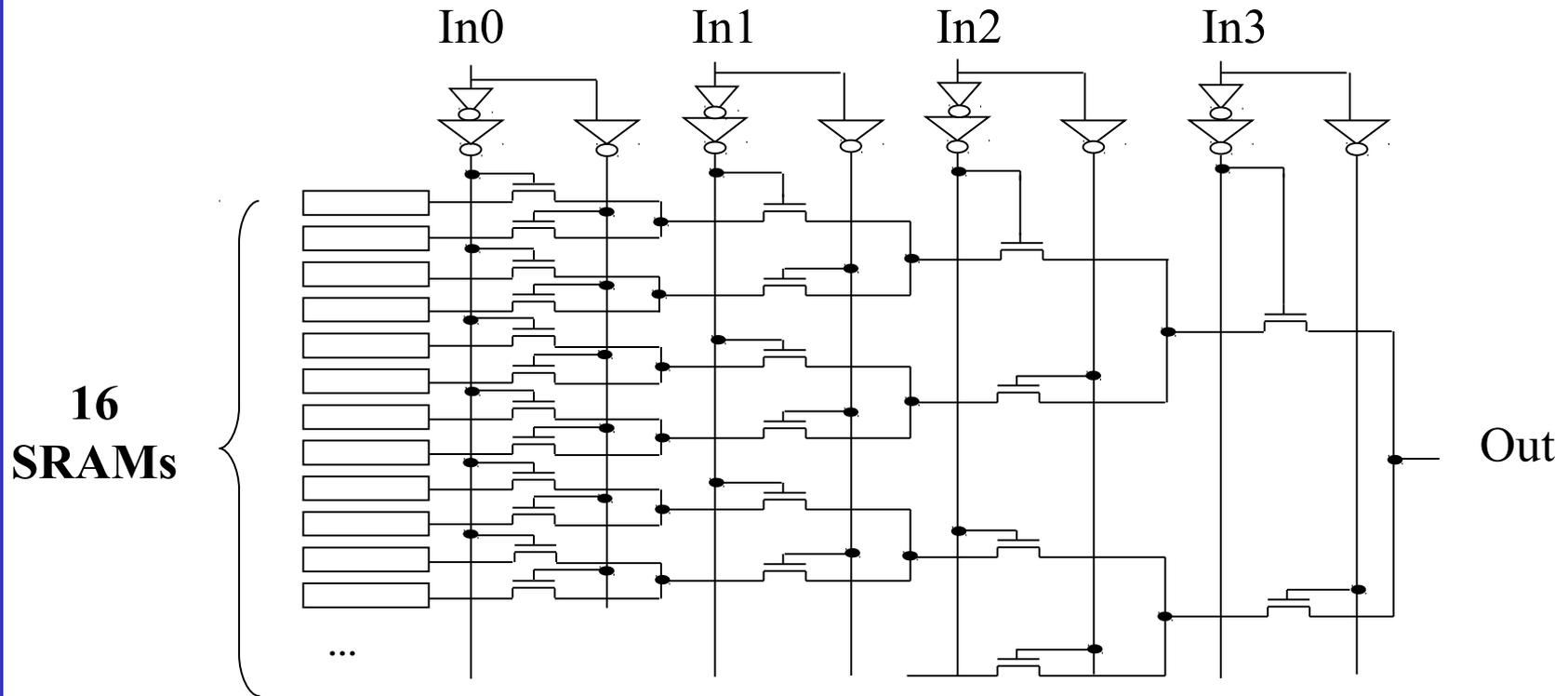
A Generic FPGA Architecture



Conceptual Structure of a Logic Block and Its Peripheral



An Implementation of a 4-input Look-up Table (4-LUT)

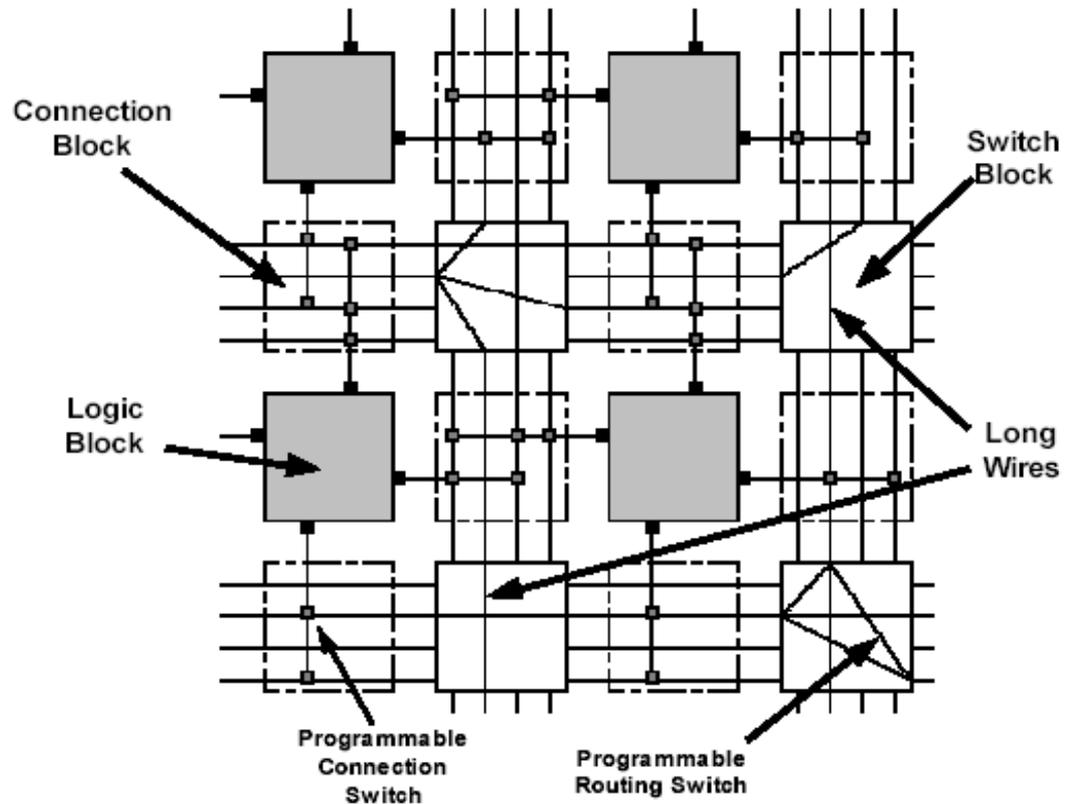


$$Out = f(in0, in1, in2, in3)$$

Interconnect

- 2-Dimensional mesh of wires, with switching elements at wire crossings to control routing
 - Bit patterns stored into the switch SRAMs determine routing
 - Switch connections programmed as part of configuring array
- To optimize for speed, many designs include multiple lengths of wire
 - Single-length (connect adjacent switches)
 - Double-length (connect to switches two hops away)
 - Long lines (run entire length/width of array)

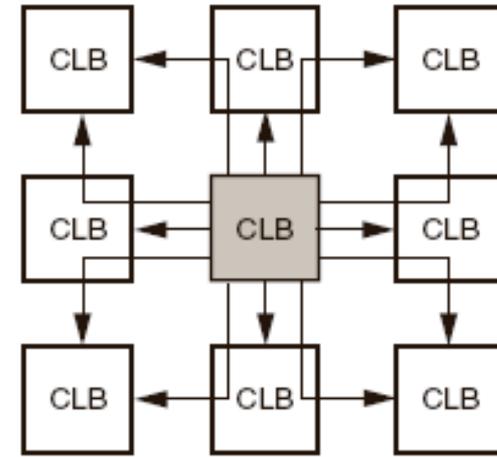
Interconnects



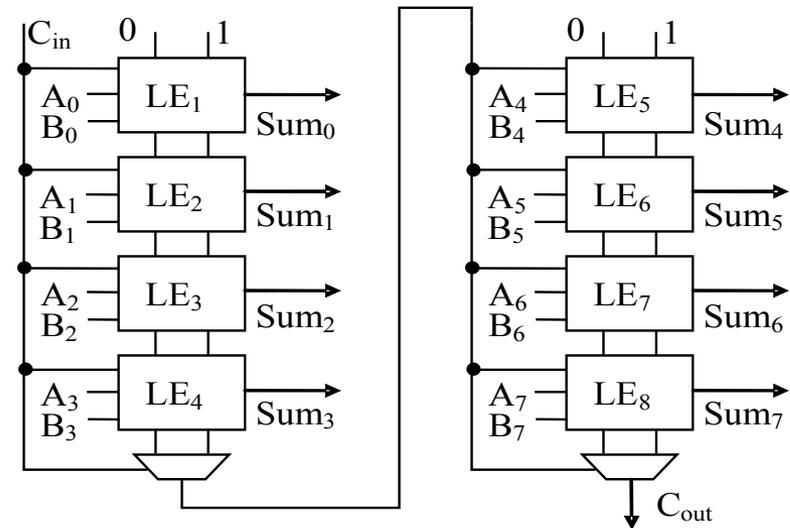
- Architecture parameters channel width (W),
- switch block flexibility (F_s – the number of wires to which each incoming wire can connect in a switch block),
- connection block flexibility (F_c – the number of wires in each channel to which a logic block input or output pin can connect),
- and segmented wire lengths (the number of logic blocks a wire segment spans).

Commercial FPGAs

- Commercial FPGA chips contain a large amount of dedicated interconnects with different fixed lengths.
- These interconnects are usually point-to-point and uni-directional connections for performance improvement.
- There are dedicated carry chain and register chain interconnects within and between logic blocks as well.



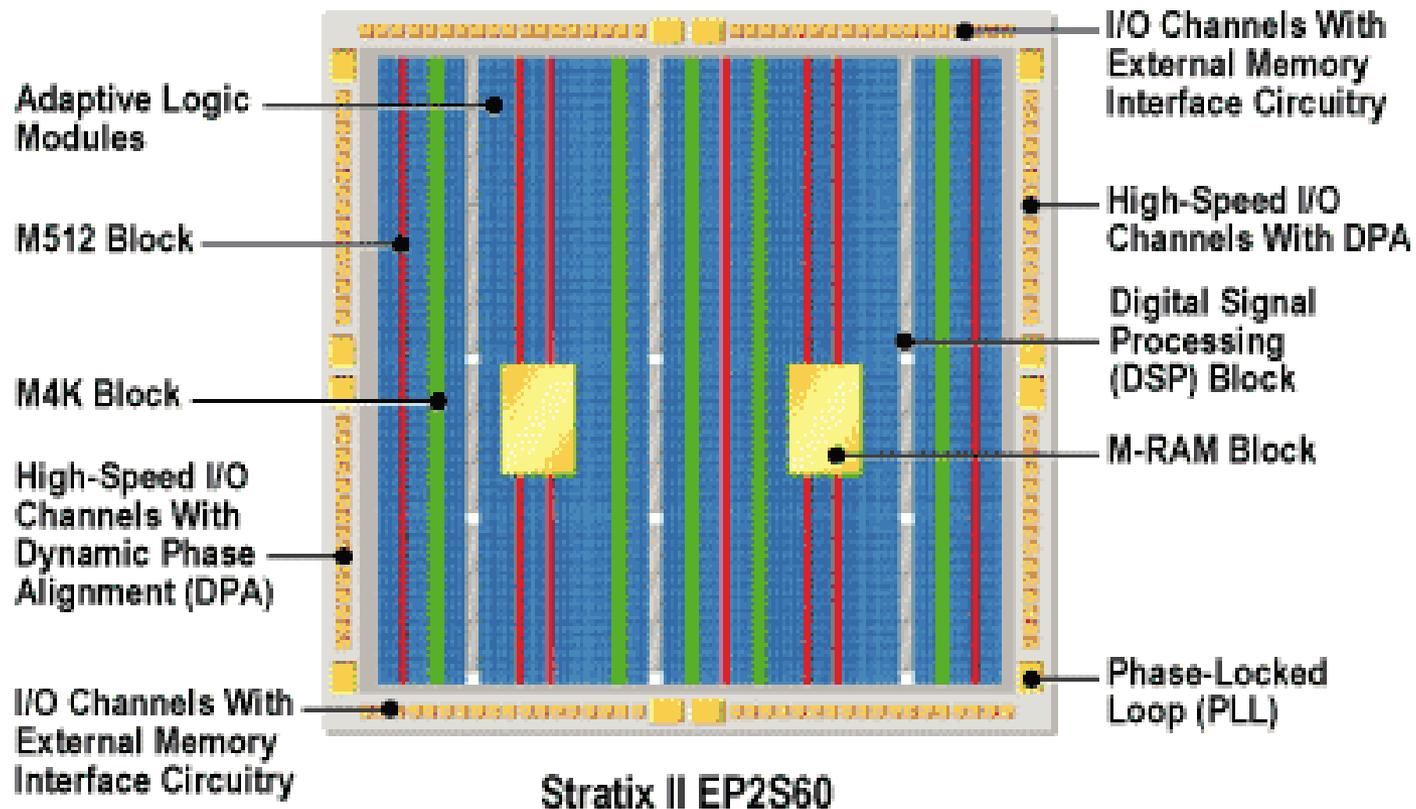
Dedicated interconnects in Xilinx's Spartan-3E



An 8-bit carry-select adder in Altera's Stratix

Commercial FPGAs

- Modern FPGAs also provide embedded IP cores, such as memories, DSP blocks, and processors, to facilitate the implementation of system-on-a-chip (SoC) designs.



Xilinx CLB – a.k.a. “Slice”

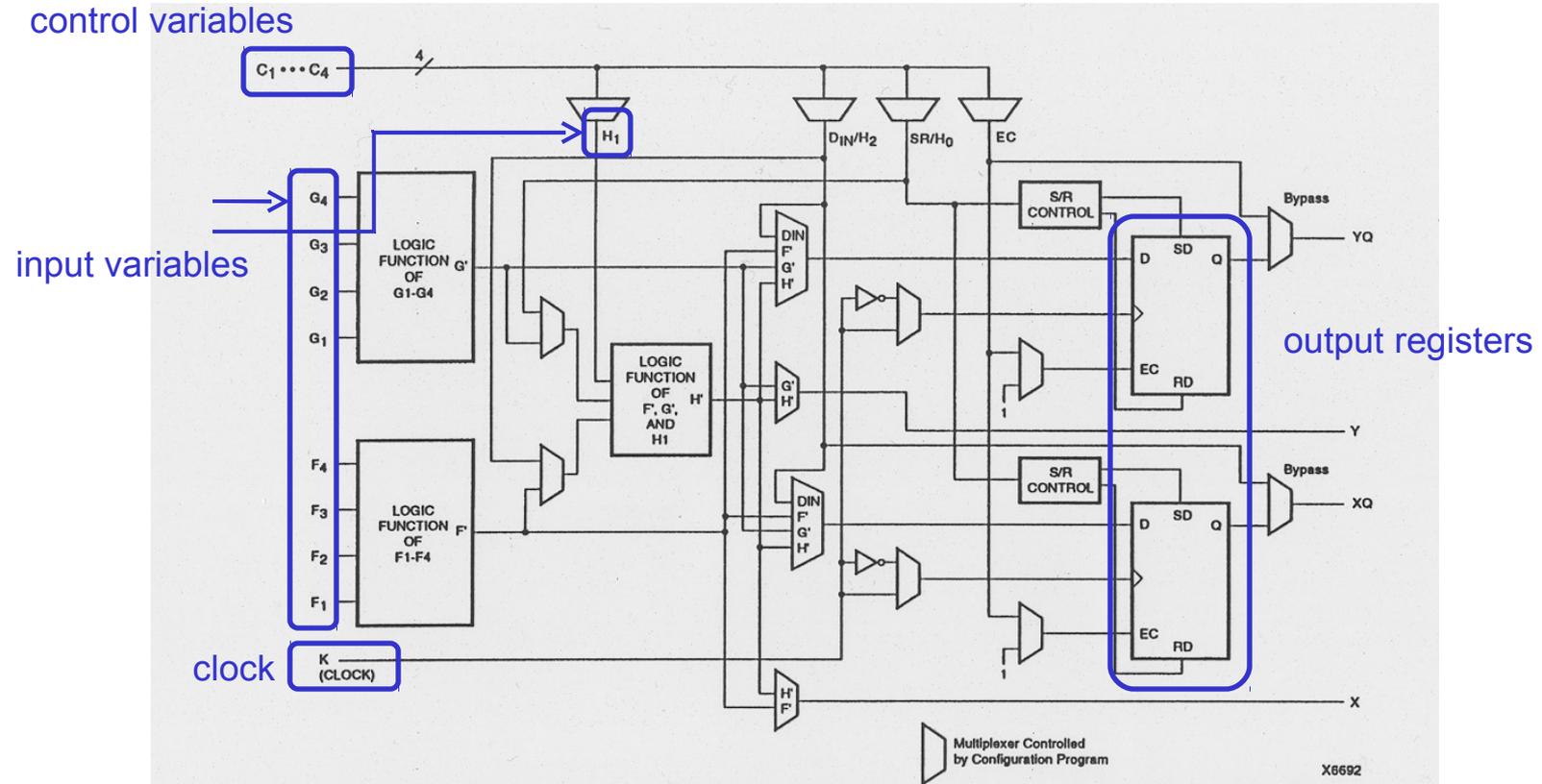


Figure 1: Simplified Block Diagram of XC4000-Series CLB (RAM and Carry Logic functions not shown)

Input-Output Blocks

- One IOB per FPGA pin
 - Allows pin to be used as input, output, or bidirectional (tri-state)
- Inputs
 - Direct
 - Registered
 - Drive dedicated decoder logic for address recognition
- IOB may also include logic for boundary scan (JTAG)

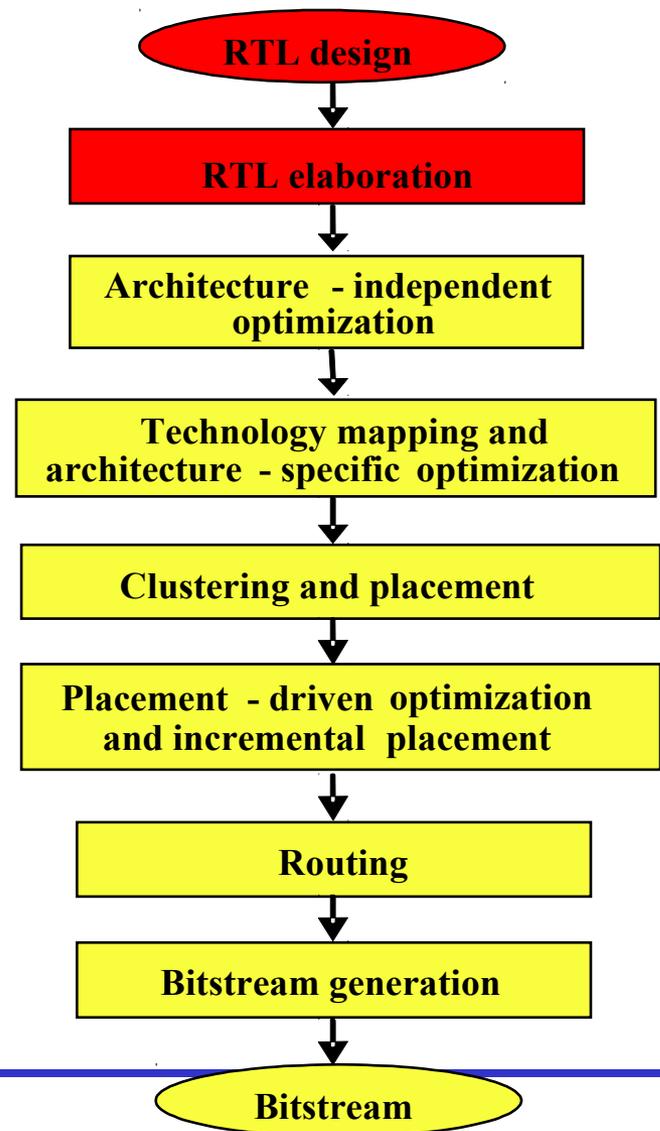
A Typical FPGA Design Flow (1)

■ RTL design

- ◆ The most widely used design specification languages are Verilog or VHDL at the register transfer (RT) level
- ◆ New trend toward moving to specification at a higher level of abstraction, where a behavior synthesis tool is used to generate the RTL specification in Verilog or VHDL

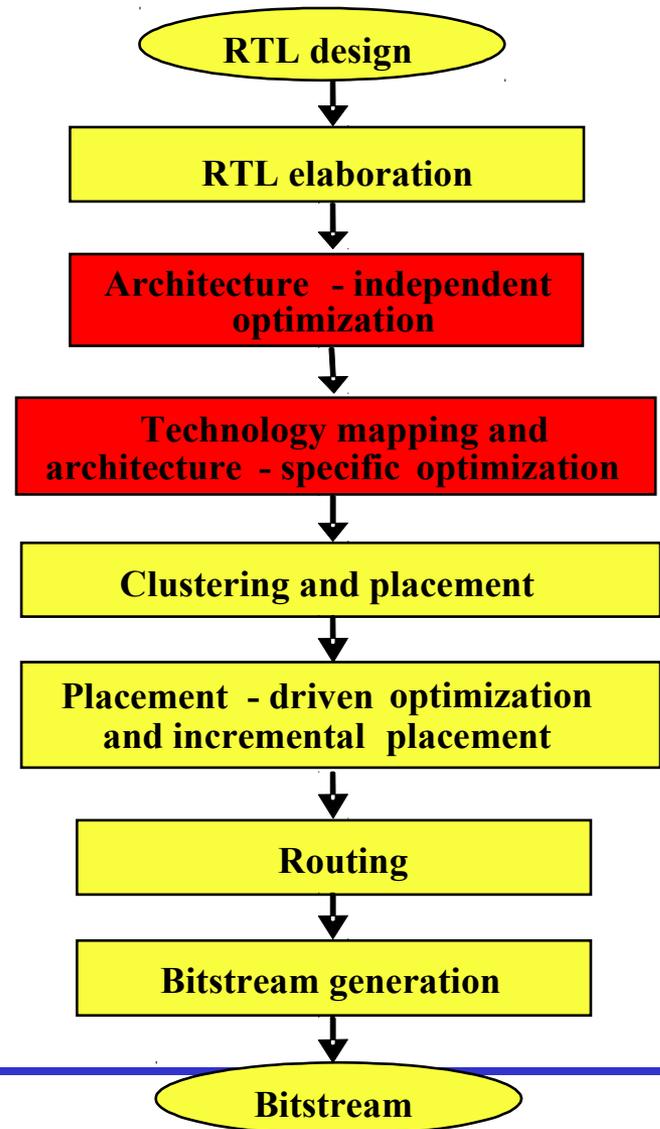
■ RTL elaboration

- ◆ This identifies and/or infers *datapath* operations, such as additions, multiplications, register files, and/or memory blocks, and *control logic*



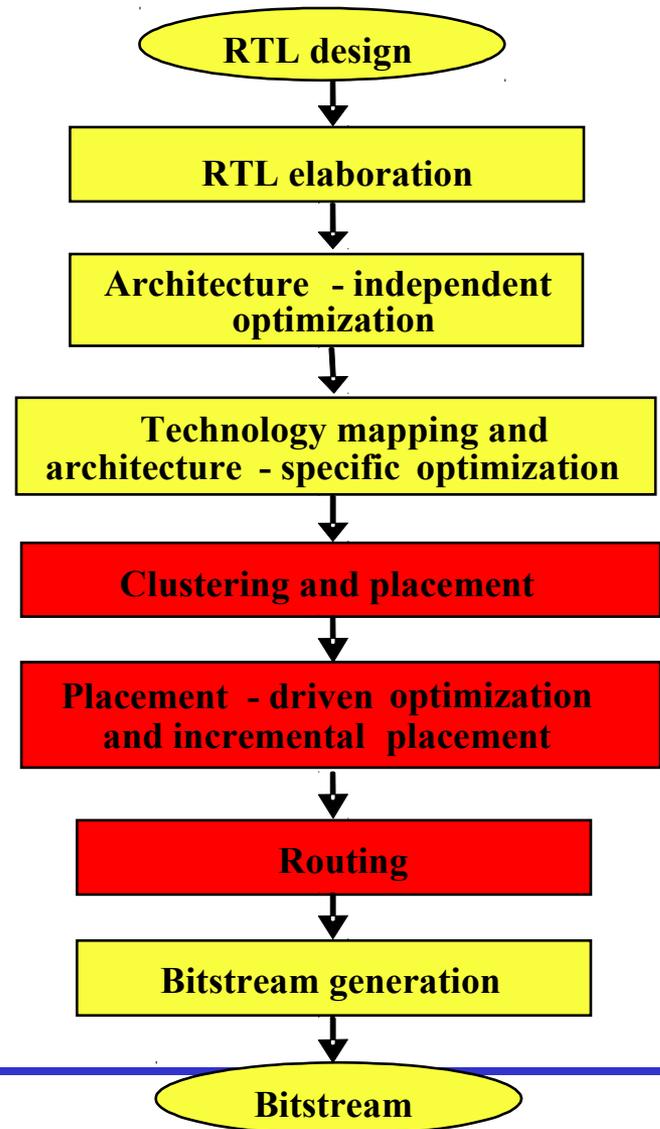
A Typical FPGA Design Flow (2)

- **Architecture-independent optimization**
 - Datapath optimization
 - Control logic optimization
- **Technology mapping and architecture-specific optimization**
 - Datapath to on-chip dedicated circuit structures
 - Control logic to basic programmable logic elements (BLEs)
 - Datapath operations to BLEs



A Typical FPGA Design Flow (3)

- **Clustering and placement**
 - Clustering and placement can be carried out separately or simultaneously
- **Placement-driven optimization and incremental placement**
 - Deal with interconnects bottleneck
 - Incremental placement to legalize
- **Routing**
 - Global routing and detail routing



FPGAs -- Pros

- Reasonably Cheap at low volume
 - Good for low-volume parts, more expensive than IC for high-volume parts
 - Can migrate from SRAM based to fuse based when volume ramps up
- Short Design Cycle (~1sec programming time)
- Reprogrammable
 - Can download bug fix into units you've already shipped
- Large capacity (millions of gates, though we won't use any that big)
 - FPGAs in the lab are "rated" at ~1M gates for 30K LE's
- More flexible than PLDs -- can have internal state

FPGA's -- Cons

- Lower capacity, speed and higher power consumption than building an ASIC
 - Sub-optimal mapping of logic into CLB's – often 60% utilization
 - Much lower clock frequency than max CLB toggle rate – often 40%
 - Less dense layout and placement and slower operation due to programmability
 - Overhead of configurable interconnect and logic blocks
- CPLDs may be faster than FPGA for designs they can handle

Evolution of the FPGA

- Early FPGAs used mainly for “glue logic” between other components
 - Simple CLBs, small number of inputs
 - Focus was on implementing “random” logic efficiently
- As capacities grew, other applications emerged
 - FPGAs as alternative to custom IC’s for entire applications
 - Computing with FPGAs
- FPGAs have changed to meet new application demands
 - Carry chains, better support for multi-bit operations
 - Integrated memories, such as the block RAMs in the devices we’ll use
 - Specialized units, such as multipliers, to implement functions that are slow/inefficient in CLBs
 - Newer devices incorporate entire CPUs: Xilinx Virtex device family has 1-4 Power PC CPUs
 - Devices that don’t have CPU hardware generally support synthesized CPUs

Next Lecture

- Introduction to VHDL