Introduction to FPGAs

Chun-Jen Tsai and Lan-Da Van
Department of Computer Science
National Yang Ming Chiao Tung University
Taiwan, R.O.C.

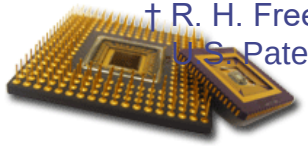
Fall, 2025

History of the FPGAs

- Before the invention of Field Programmable Gate Array (FPGA), there are many different programmable IC such as PLA, PAL, CPLD, etc.
  - These programmable IC's have very limited logic capacity.
  - CPLD IC's are still popular in industry because their "digital logic programs" do not disappear when the power is off.

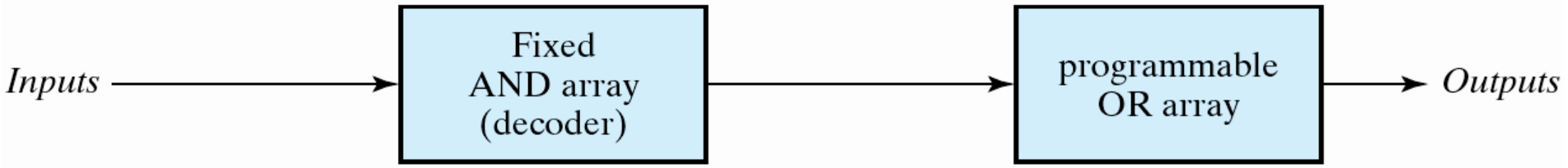
- R. H. Freeman invented the first practical FPGA in 1985†.
  - Freeman and Vonderschmitt co-funded Xilinx in 1984.
  - Xilinx is the first fabless IC design house in the world.
  - TSMC enters the IC OEM business for fabless IC design houses in 1987.

† R. H. Freeman, "Configurable Electrical Circuit Having Configurable Logic Elements and Configurable Interconnect," U.S. Patent 4,870,302, Sep. 26, 1989

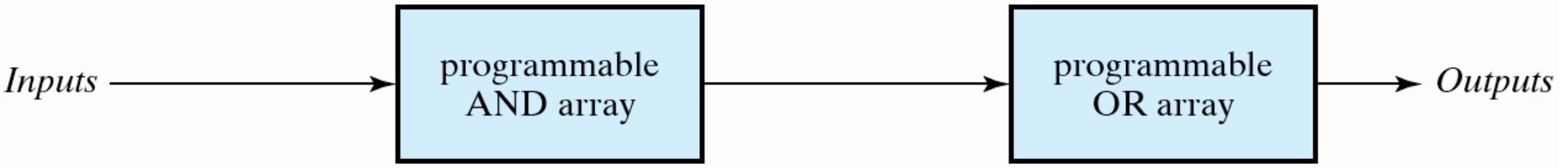


Combinational PLDs

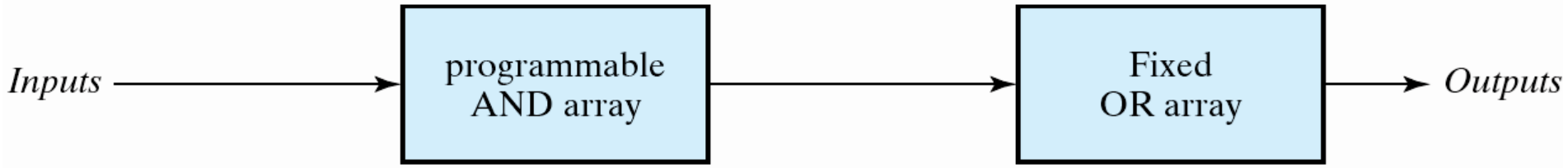
- Programmable two-level logic
  - an AND array and an OR array



(a) Programmable read-only memory (PROM)



(c) Programmable logic array (PLA)



(b) Programmable array logic (PAL)

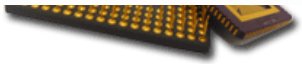
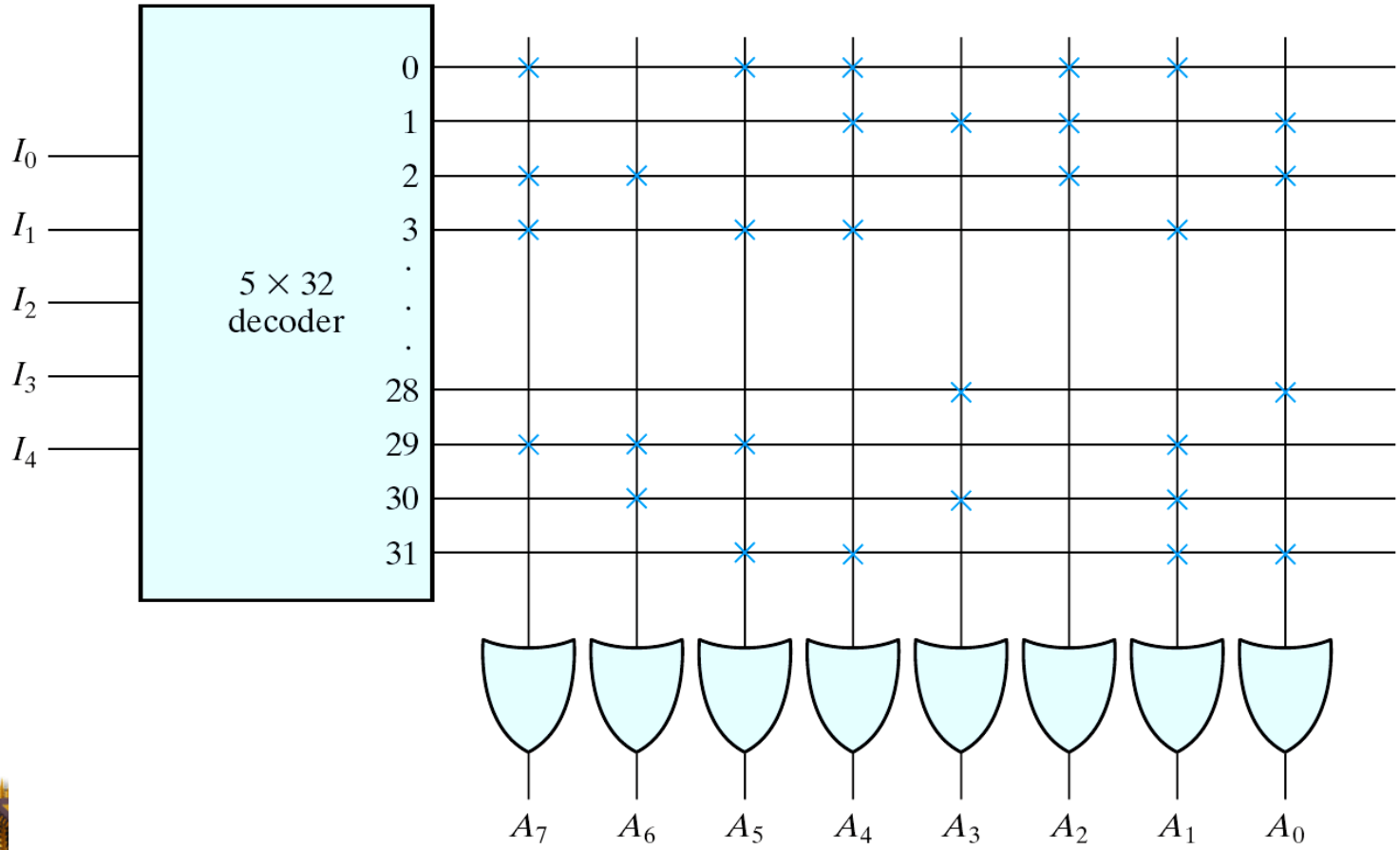
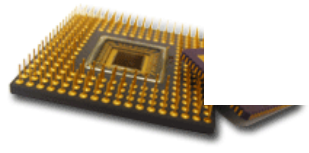


Diagram of 32x8 ROM

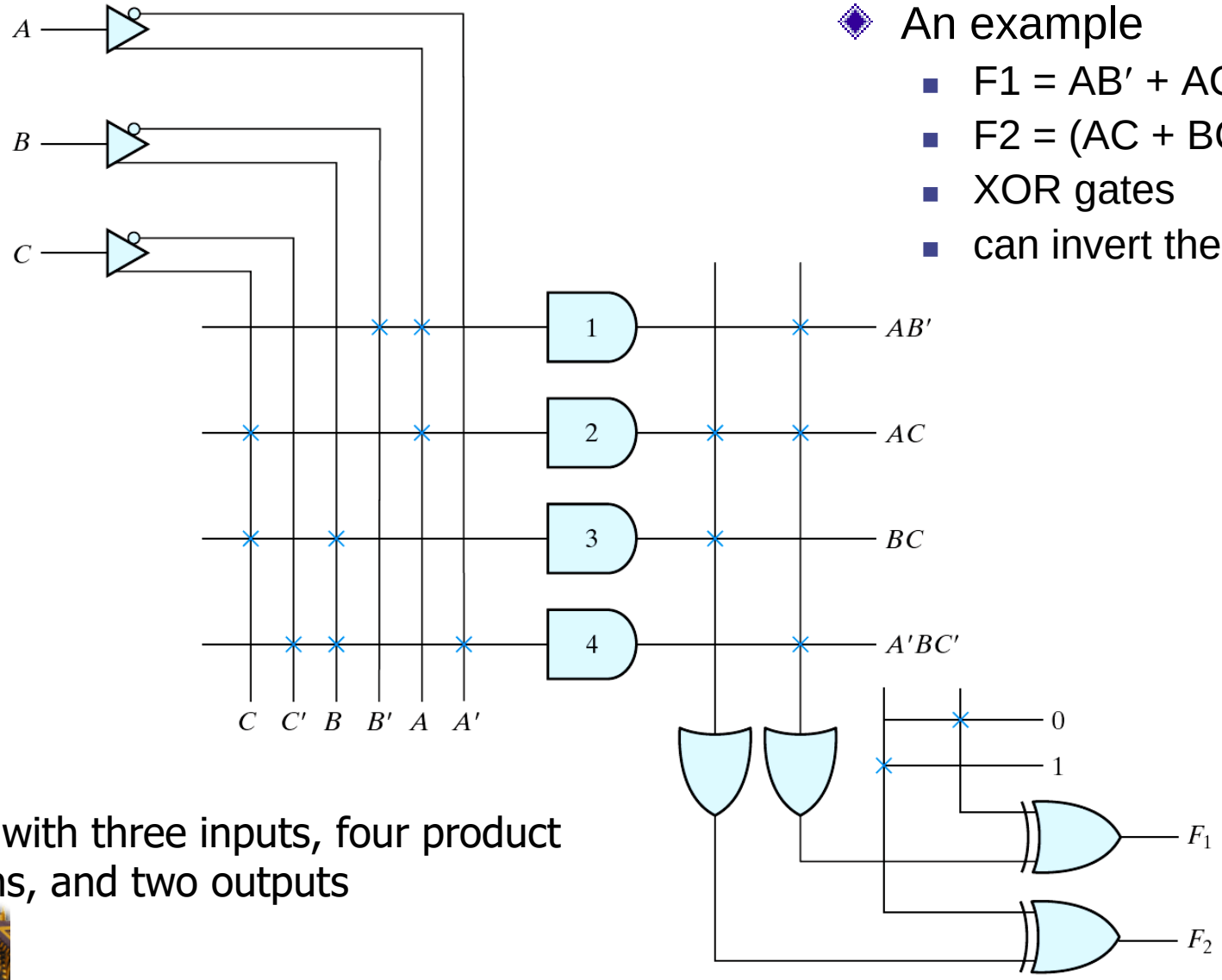
- programmable interconnections
  - close (two lines are connected)
  - or open
  - A fuse that can be blown by applying a high voltage pulse



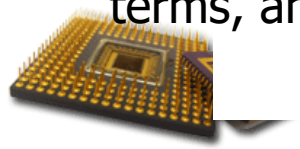


PLA Example

- An example
  - F1 = AB′ + AC + A′BC′
  - F2 = (AC + BC)′
  - XOR gates
  - can invert the outputs



PLA with three inputs, four product terms, and two outputs



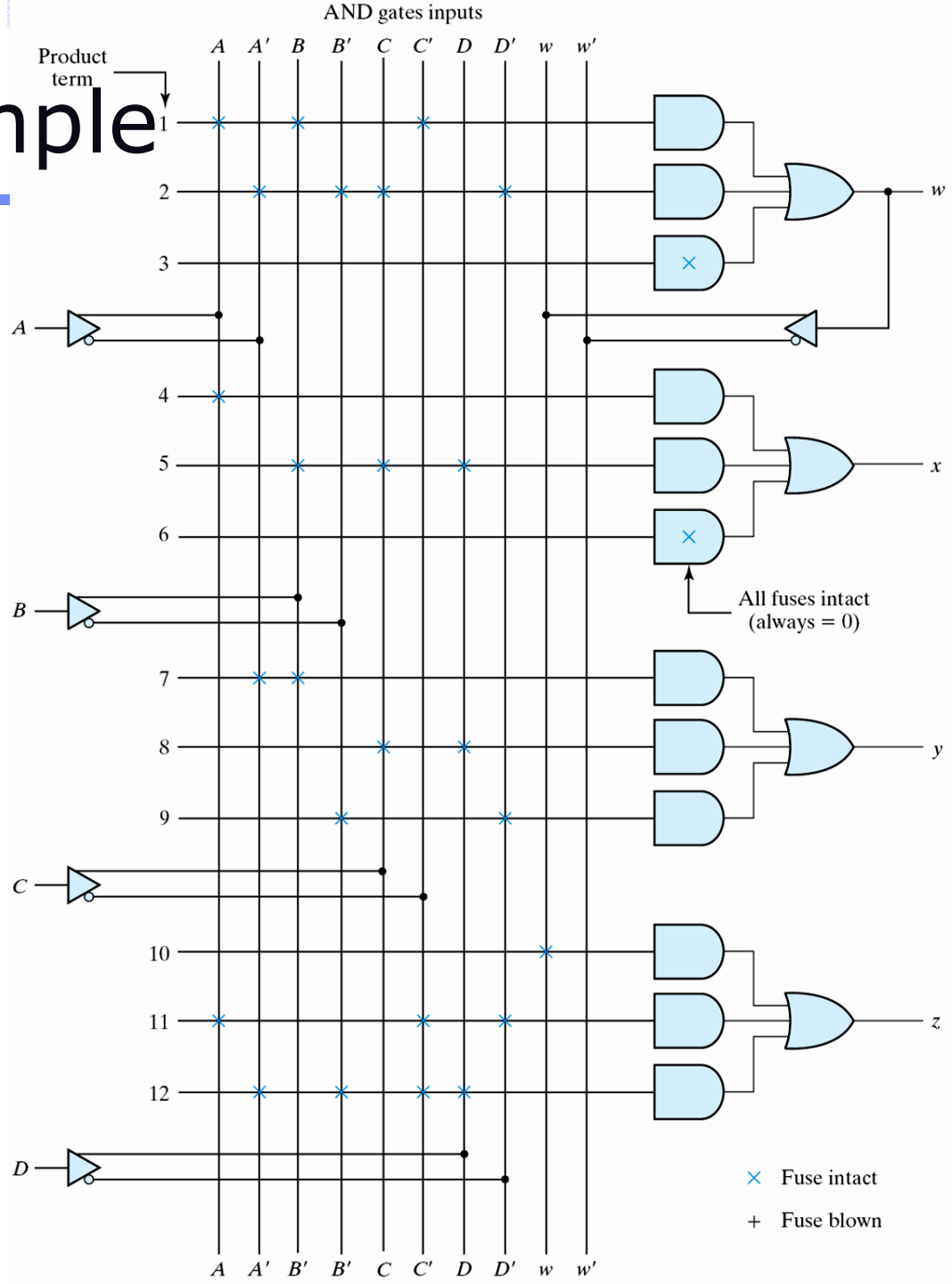
PAL Example

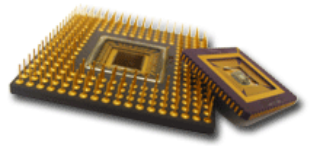
$w = ABC' + A'B'CD'$

$x = A + BCD$

$y = A'B + CD + B'D'$

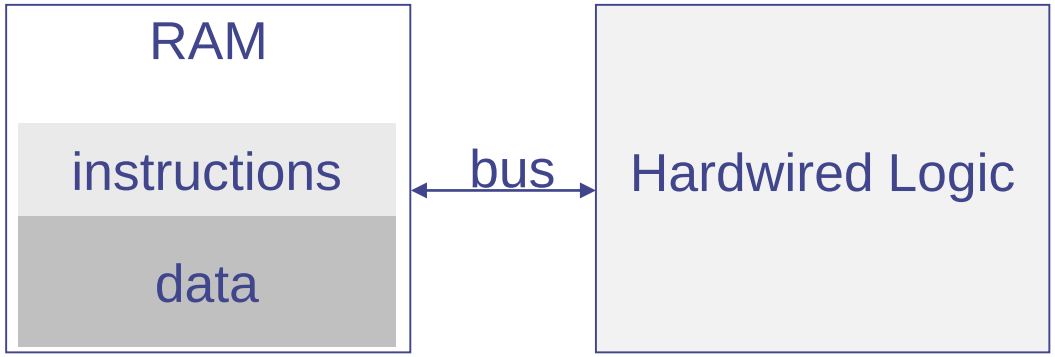
$z = w + AC'D' + A'B'C'D$

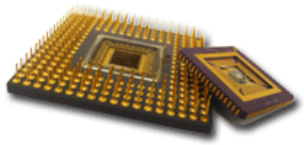




Programming Concept of the FPGA

- Theoretically speaking, creating a digital logic on an FPGA strictly follows the “stored program concept” of software programming.
  - Thus, FPGA coding should be regarded as “software coding”.
- Stored program concept:
  The computing behavior of a piece of hardware is controlled by information stored in memory devices.





Digital Circuits in an FPGA

- The basic ideas of FPGA's is to inter-connect small "truth tables" to emulate complex digital circuits.
  - In an FPGA, these tables are called "Lookup Table" (LUT).

Table 1

input	output
0000	1
0001	0
…	…
1111	1

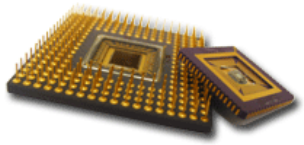
Table 4

input	output
0000	0
0001	1
…	…
1111	1

Table 5

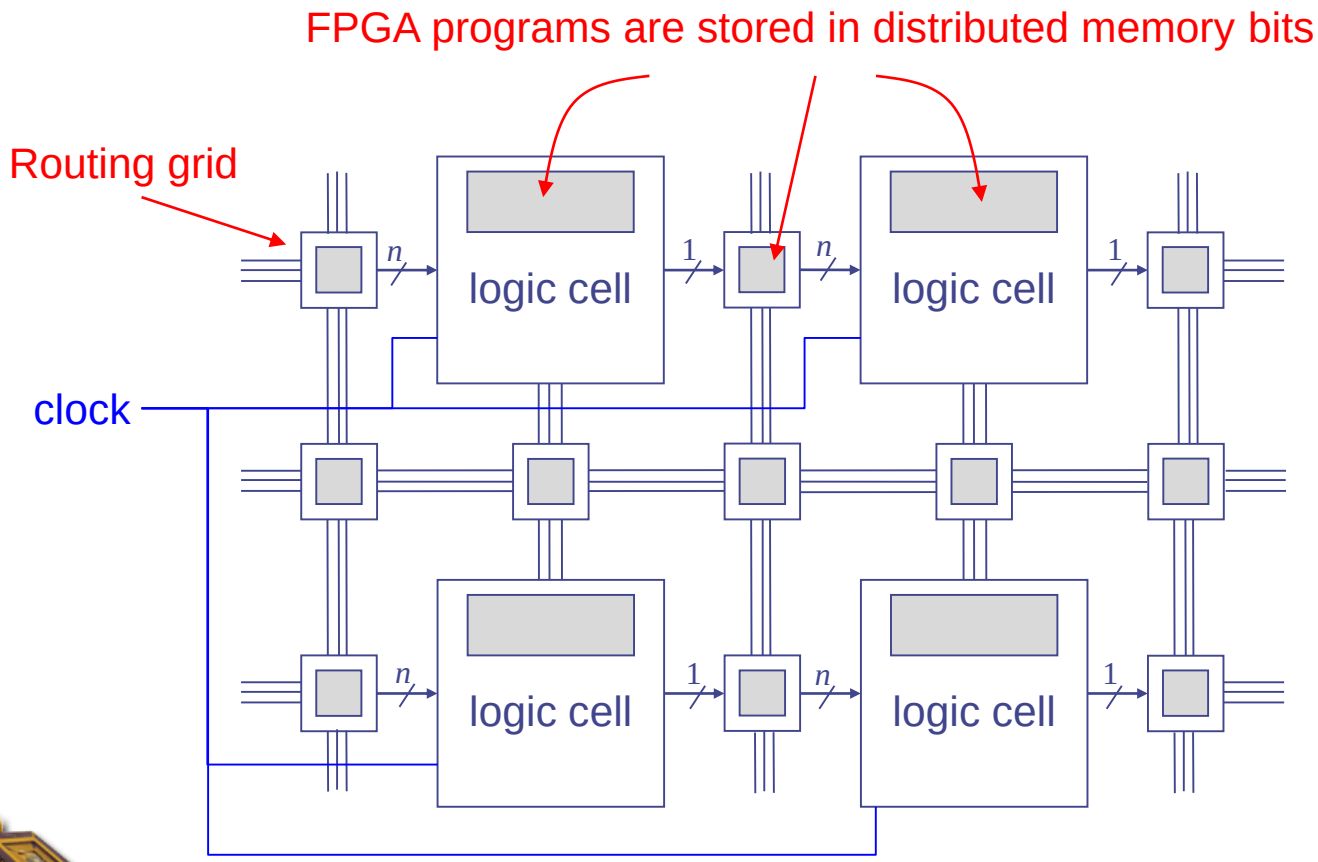
input	output
0000	0
0001	0
…	…
1111	1

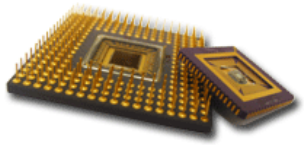
Each table is equivalent to a n-to-1 gate, where n = 4 or 6 for Xilinx FPGA's.



Programmability of the FPGA

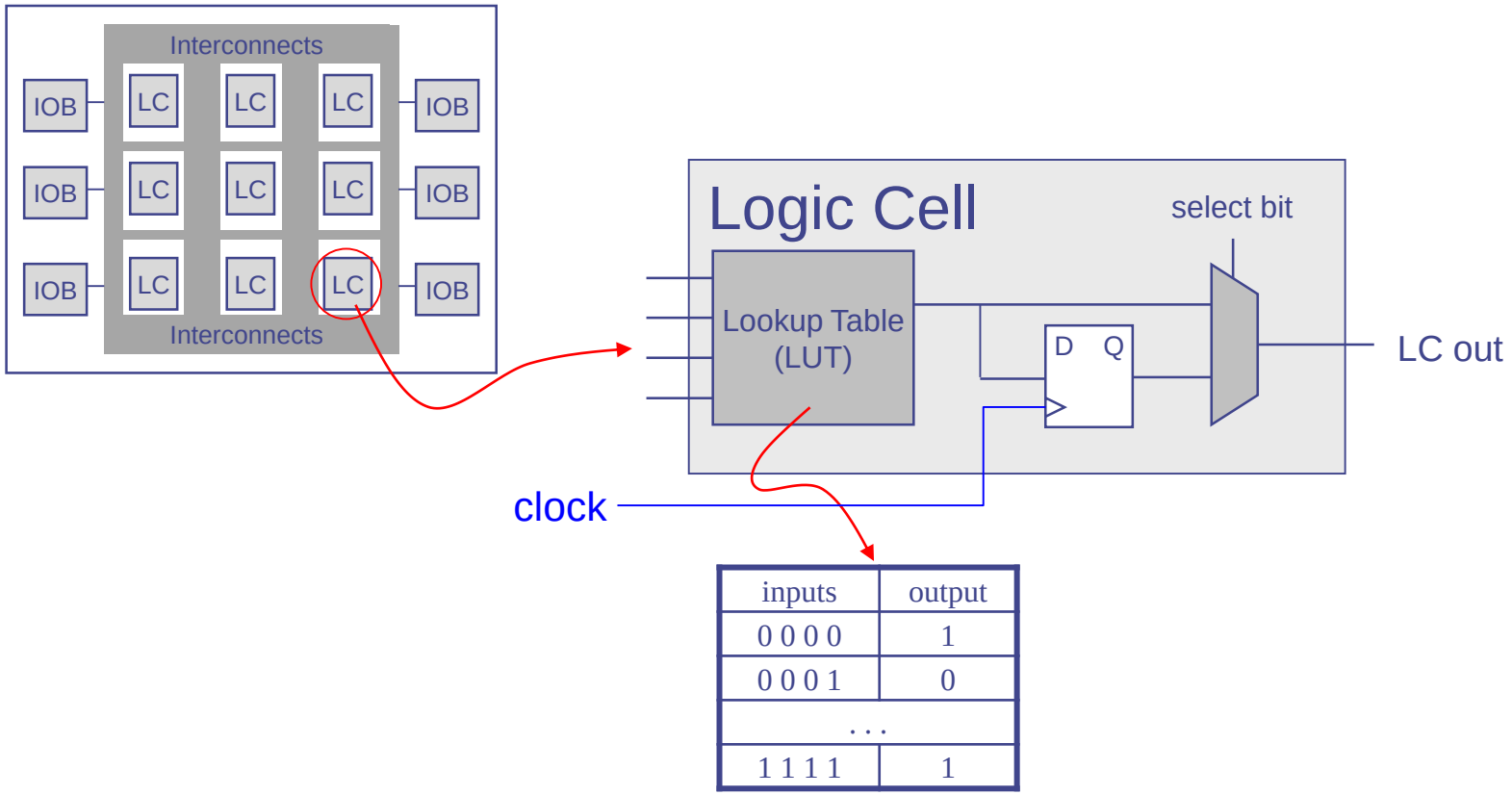
- FPGA still adopts the “stored program concept”.
  - Both **the n-to-1 gates** and **the wires** can be programmed.

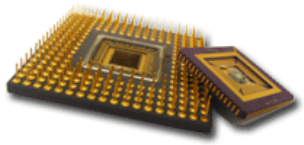




A Generic Logic Cell

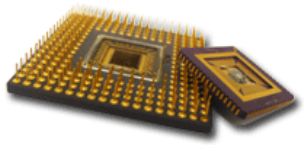
- Basic logic cell (LC) structure:





Features of SRAM-based LUT

- SRAM is often used to implement LUT, the properties of this type of LUT is as follows:
  - An n-bit input LUT can handle function of 2^n different inputs.
  - All logic functions take the same amount of area.
  - All functions have the same delay.
    - For CMOS custom logics, XOR is much slower than NAND.
  - Burn power even at idle
- An LUT is more “powerful” than a two-input gate.
  - Gate-count is not a good measure of FPGA logic cost.
  - For a static gate, an n input NAND/NOR gate has ~ $2n$ transistors.
  - For an FPGA LC, 4-input LUT has 128 transistors in SRAM, 96 in multiplexer.

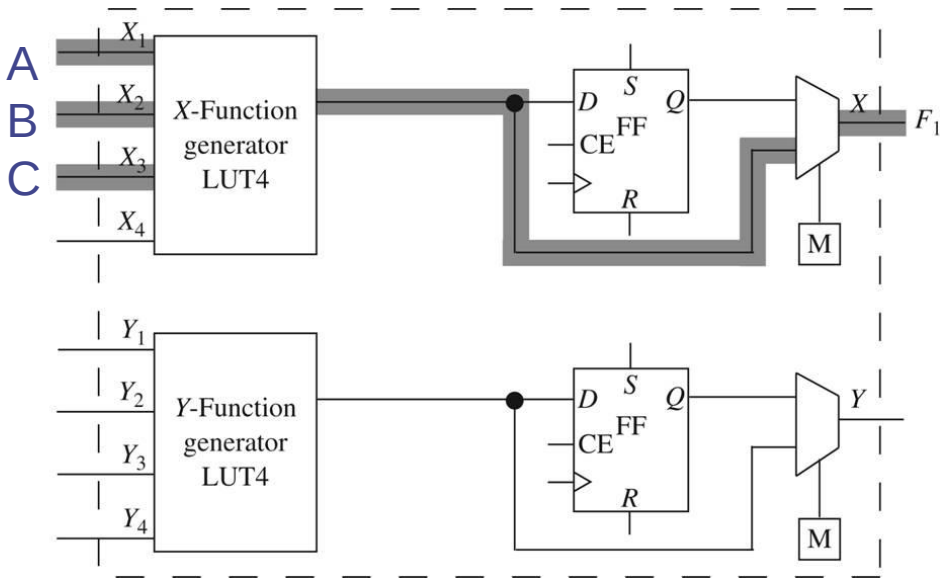


Function Implementation with LUTs

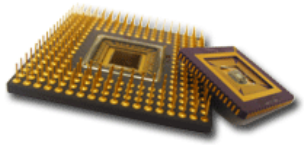
- The datapath that implements $F = A'B'C + A'BC' + AB$ is as follows, the LUT4 has entries as follows:

$X_1 X_2 X_3 X_4$	F
0000	0
0001	0
0010	1
0011	1
...	...
1111	1

LUT4 table entries
(red means don't care)

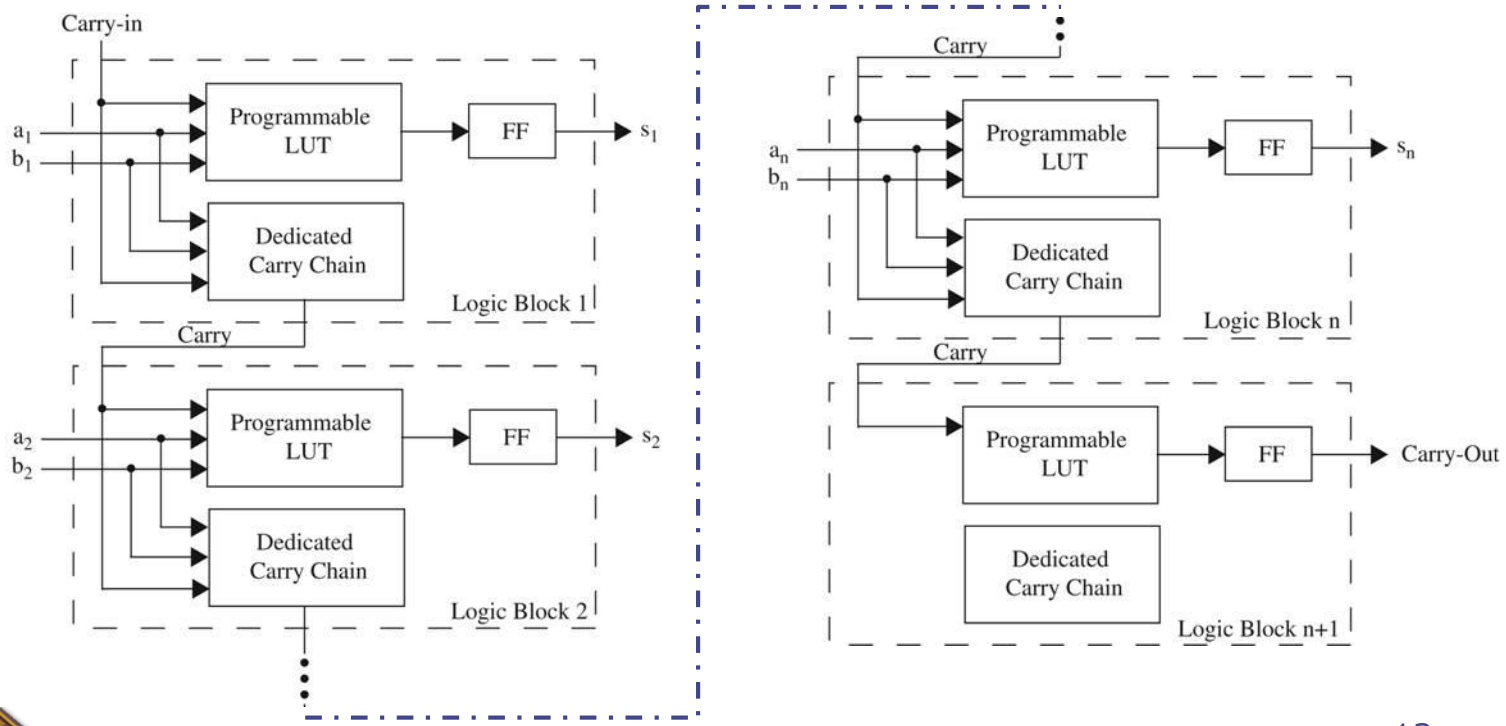


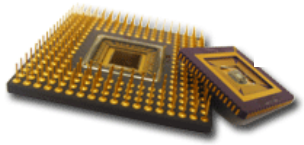
- A function with more than 4 variables can always be decomposed to the sum (OR) of 4-variable function.



Carry Chains in FPGA

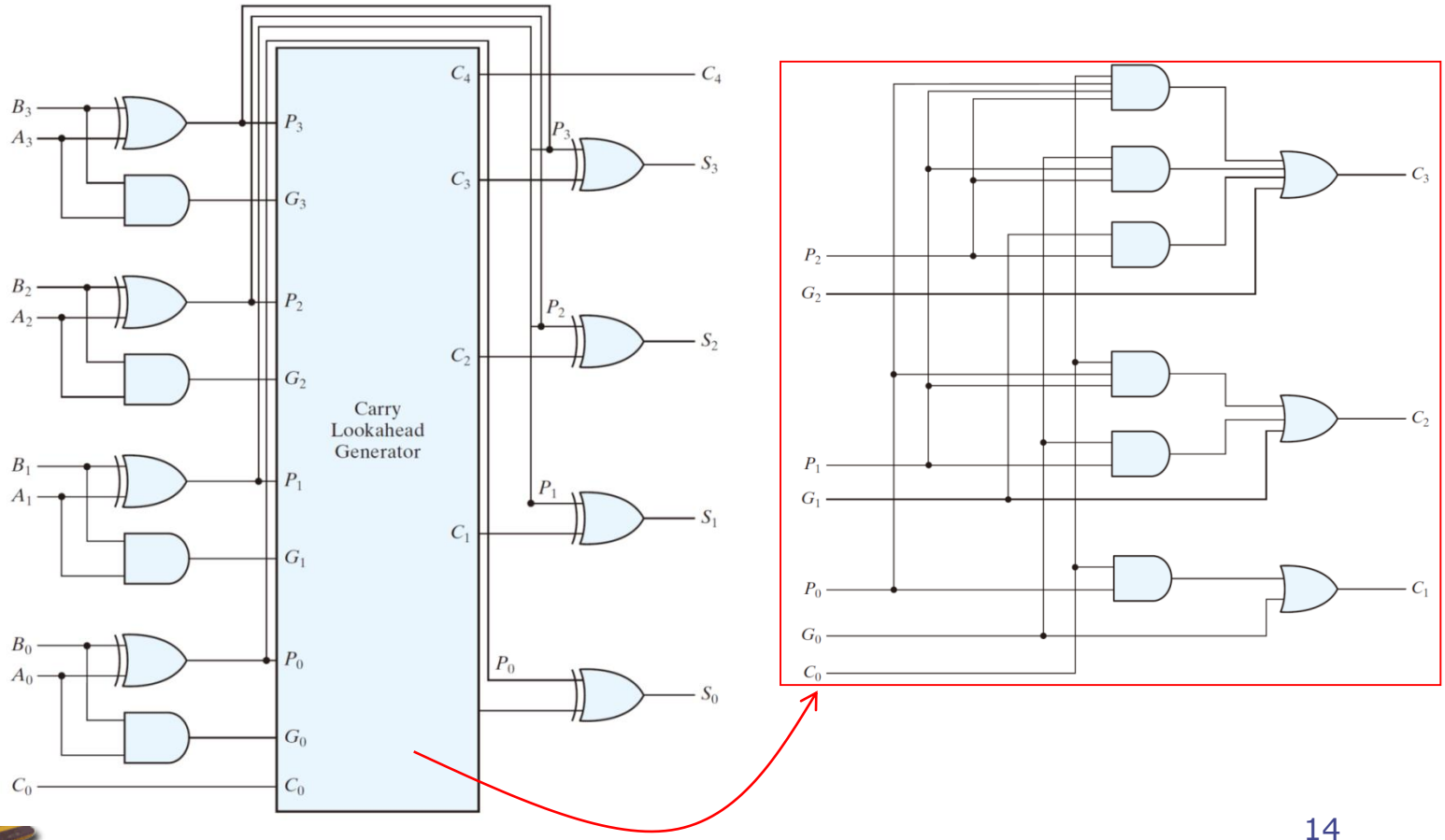
- Since additions are very important, many FPGAs have dedicated circuits for carry calculation and propagation.
  - Both carry look-ahead and carry-ripple adders can be implemented efficiently on FPGAs.



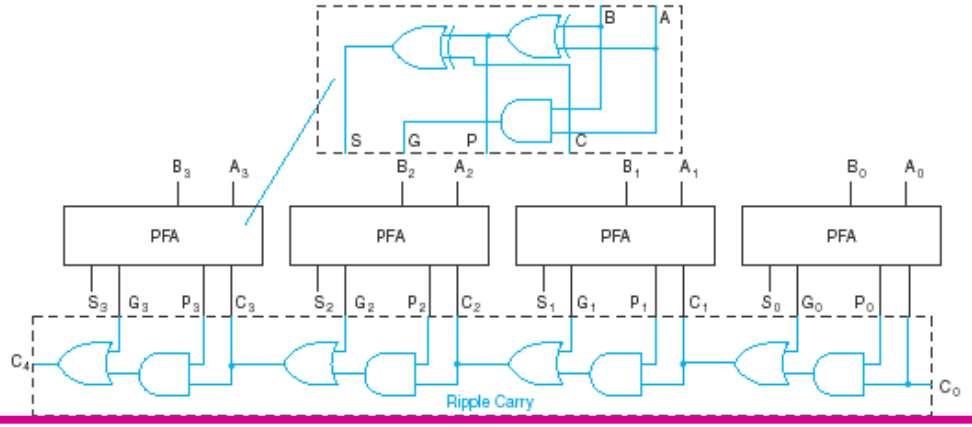


Review: Look-ahead Carry Generator

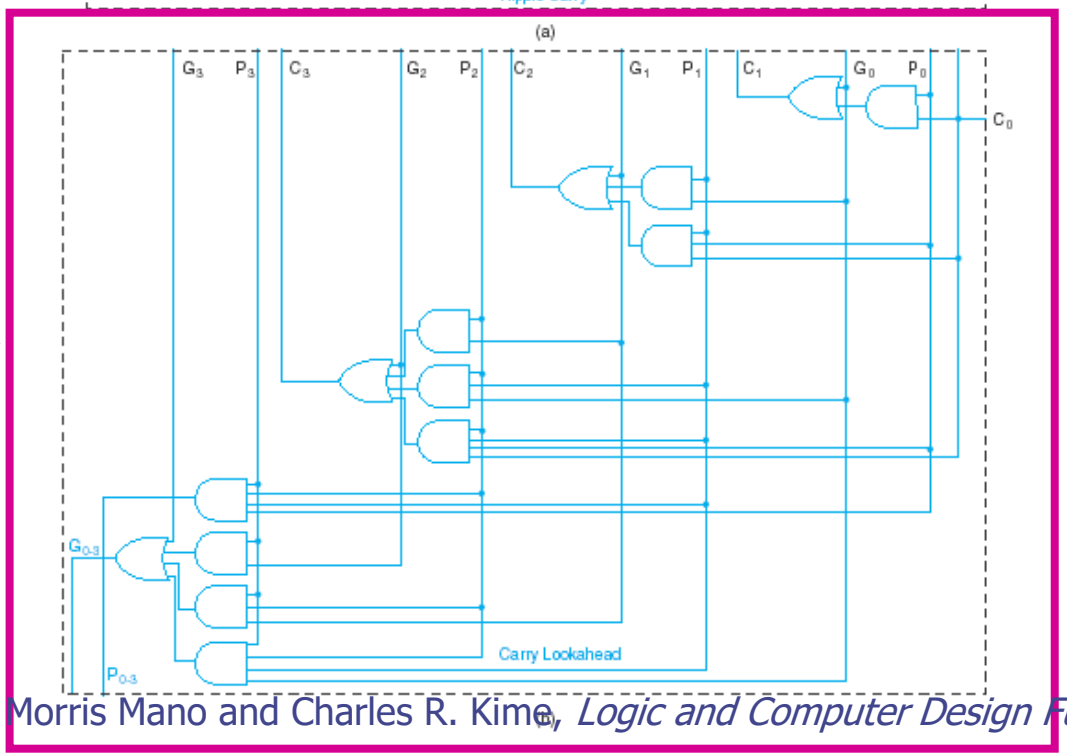
- A look-ahead carry adder is more efficient than a carry-ripple adder:

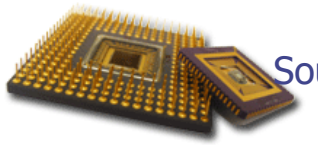


Review: Look-ahead Carry Generator



CLA GEN



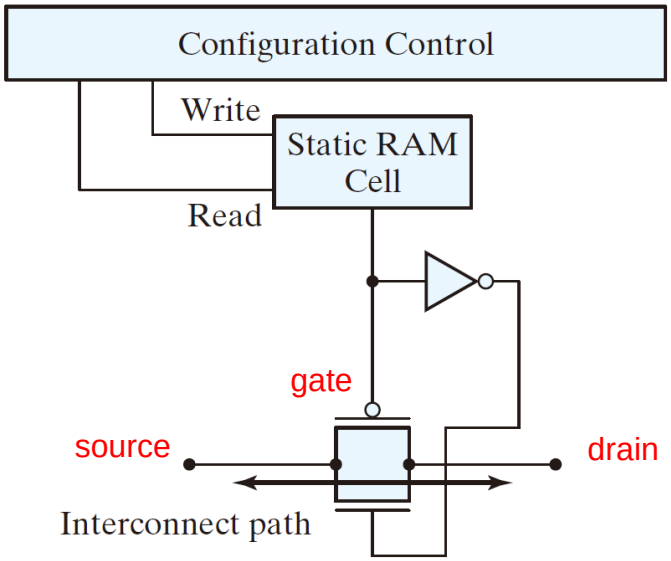


Source: M. Morris Mano and Charles R. Kime, *Logic and Computer Design Fundamentals*.

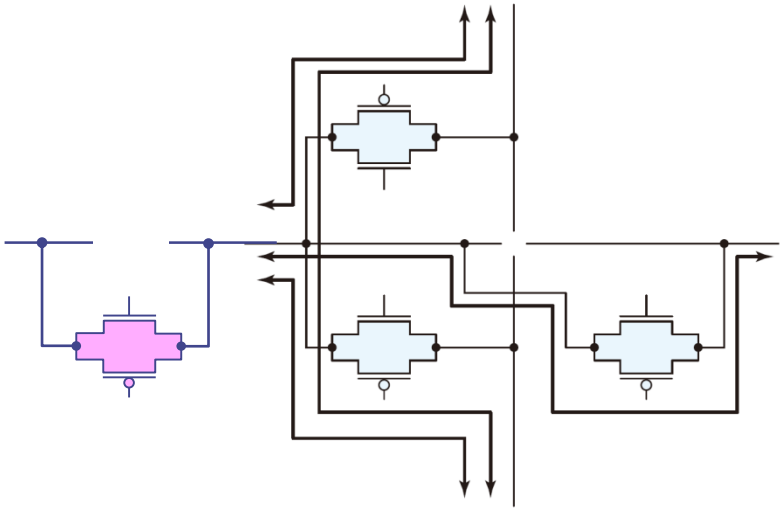
Programmable Interconnect Points

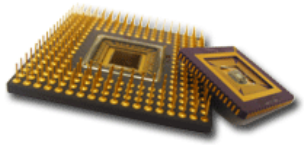
- The routing wire in an FPGA is programmed using a single-bit SRAM cell connected to a transistor:

Connect/disconnect a single wire:



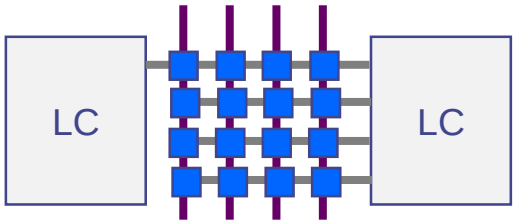
Routing a signal:

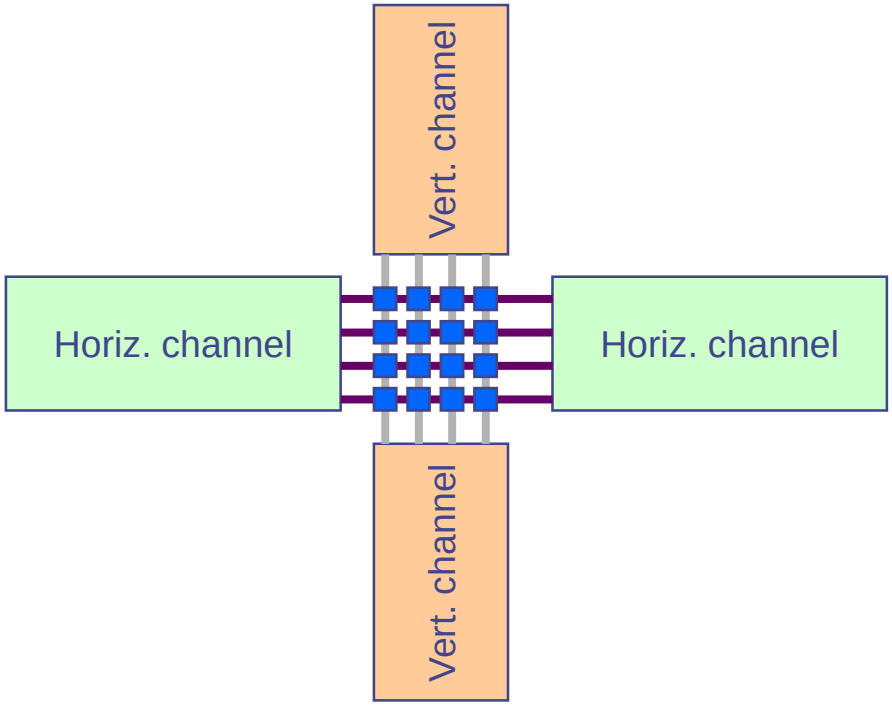


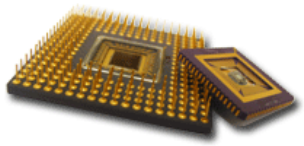


Interconnect Architecture

- On an FPGA, we must be able to control
  - Connections from wiring channels to LCs
  - Connections between wires in the wiring channels

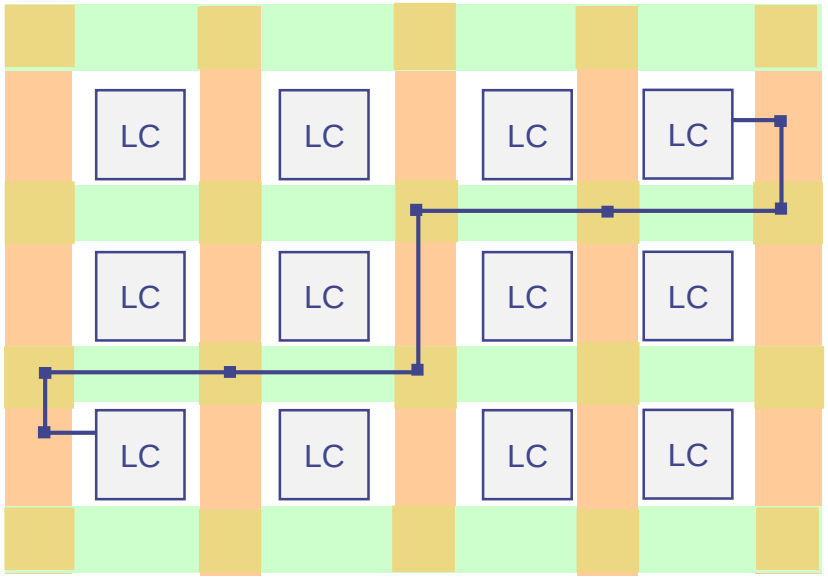


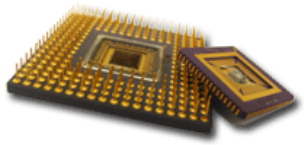




Place and Route Problem

- Wiring among LCs are organized into channels.
  - Channels are arranged horizontally and vertically on the chip.
  - There are many wires per channel.
- Connections between wires made at programmable interconnection points
- An EDA tool must choose:
  - Channels from source to destination
  - Wires within the channels





Xilinx FPGA Family

- The Xilinx FPGA generations are coupled with the TSMC manufacturing process.
  - Usually two lines of product before the TSMC 28nm process
- Old products before the 28 nm TSMC process
  - Virtex-II / Spartan-2 (130 nm)
  - Virtex-4 / Spartan-3 (90nm)

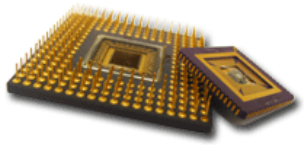
  } 4-input LUT

  - Virtex-5 (65 nm)
  - Virtex-6 (40 nm)/ Spartan-6 (45 nm)

  } 6-input LUT

- New products after the 28 nm TSMC process
  - Virtex-7 / Kintex-7 / Artix-7 / Zynq-7 (28 nm)
  - Virtex UltraScale / Kintex UltraScale (20nm)
  - Virtex UltraScale+ / Zynq UltraScale+ (16 nm FinFET+)

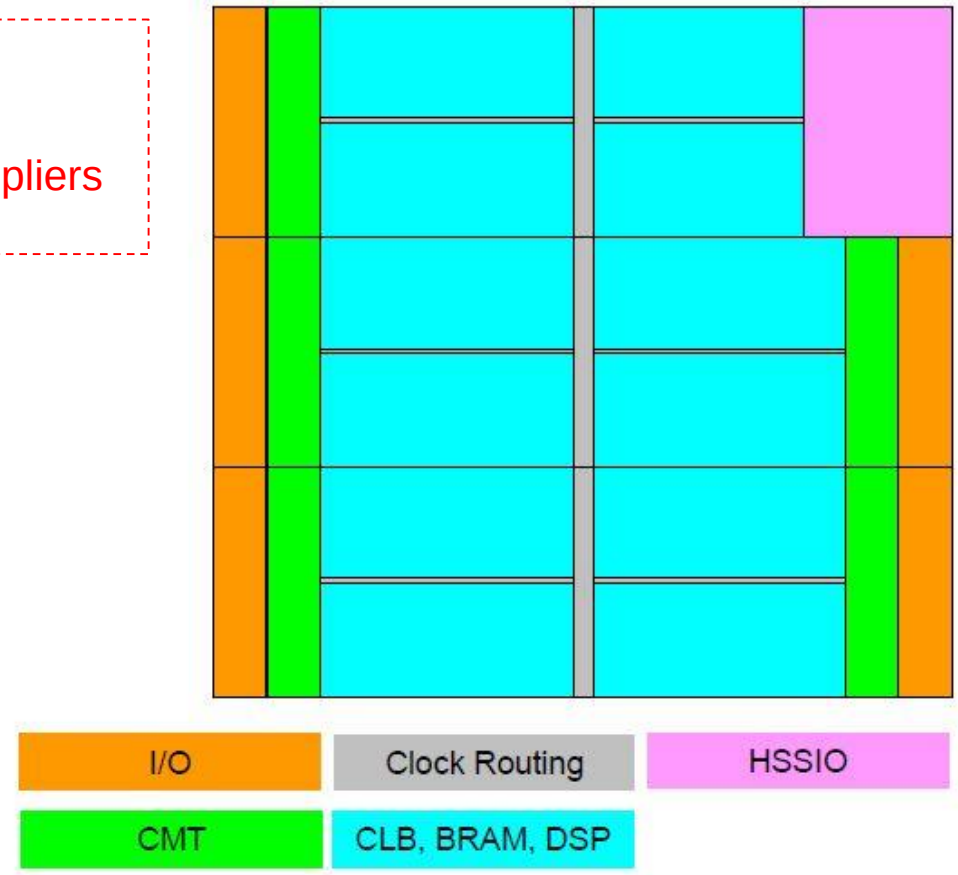
  } 6-input LUT

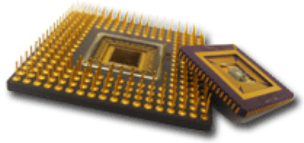


Xilinx FPGA Architecture

- Xilinx FPGAs are composed of CLBs, BRAMs, Multipliers (in DSP slices), CMTs, and IOBs.

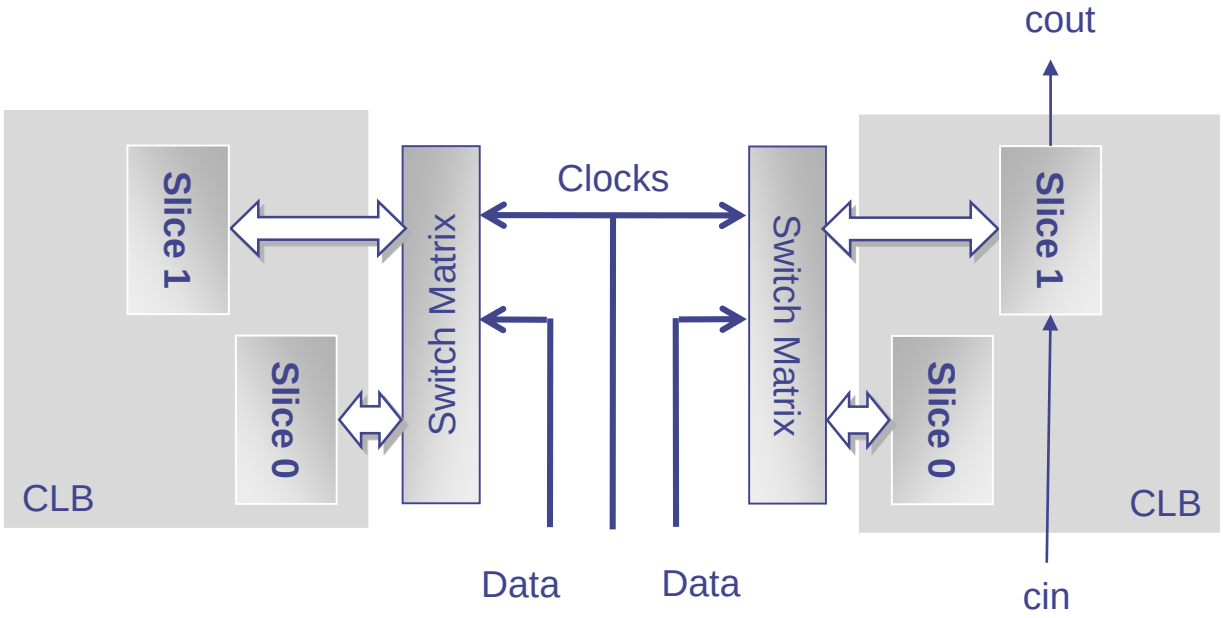
XC7A35T:
- 2,600 CLBs
- 1800 Kb BRAMs
- 90 25x18-bit multipliers
- 5 CMTs

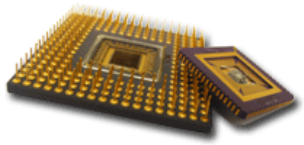




CLB Architecture of Artix-7

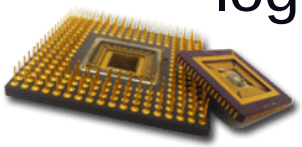
- The CLB architecture of an Artix-7 FPGA:
  - The main resources for logic synthesis
  - Each CLB contains two slices, each slice four logic cells.
  - Each CLB is connected to a switch matrix.
  - Carry chain runs vertically in a column from one slice to the one above.

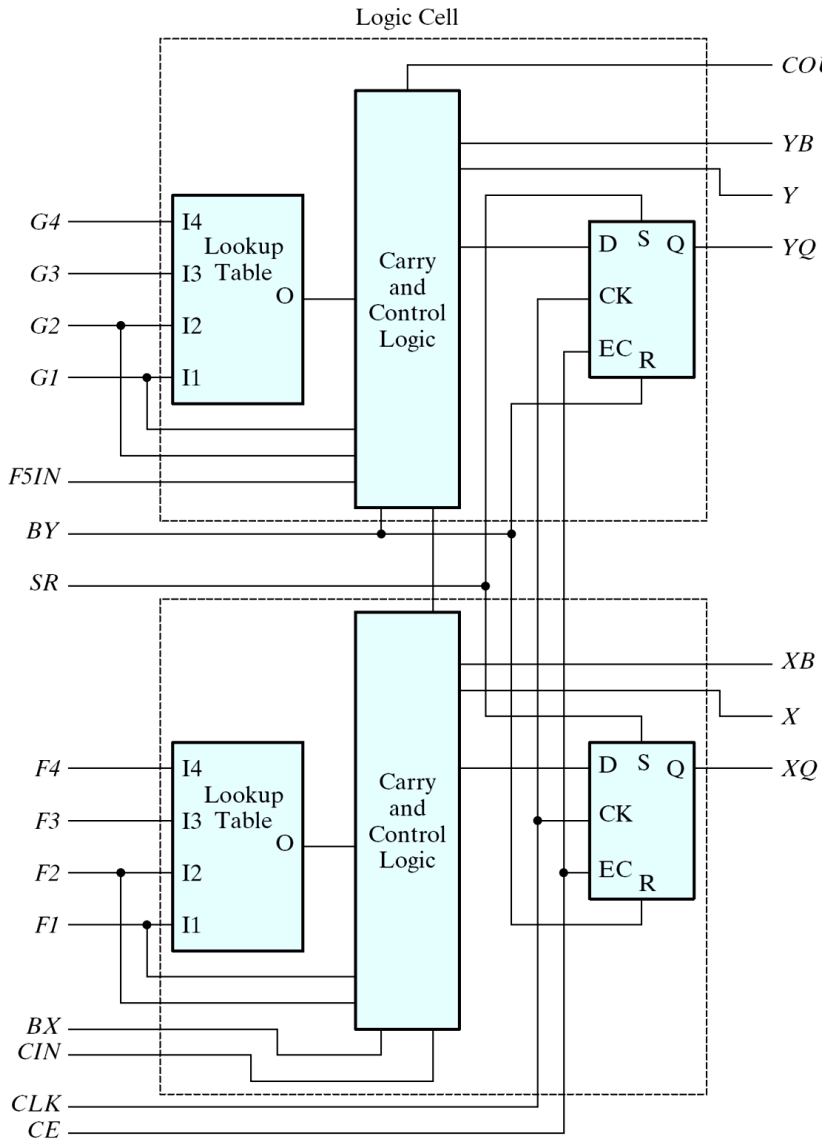




CLB Architecture of Spartan II

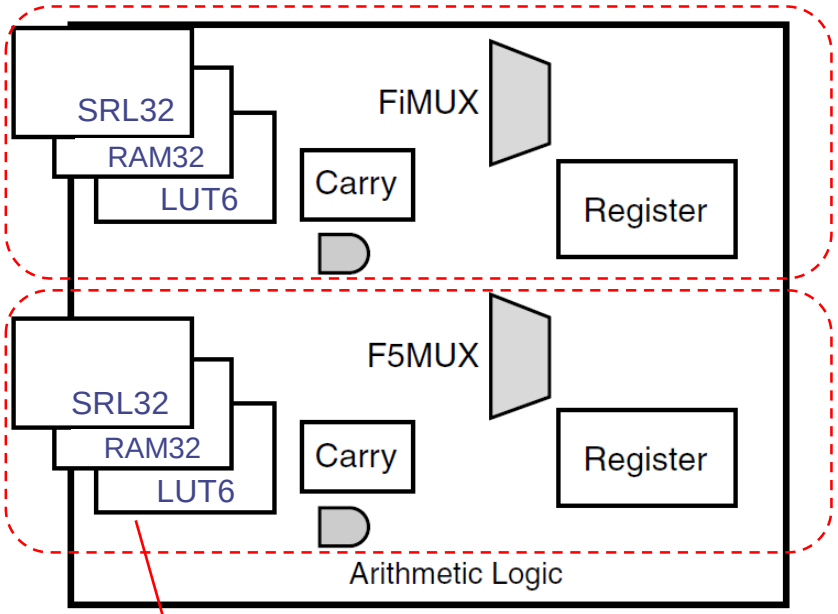
- Each CLB contains four logic cells, organized as a pair of slices.
- Each slice has a four-input lookup table, logic for carry and control, and a D-type flip-flop.
- Spartan II family provides the flexibility and capacity of an on-chip block RAM.
- 1 CLB = 2 slices = 4 logic cells



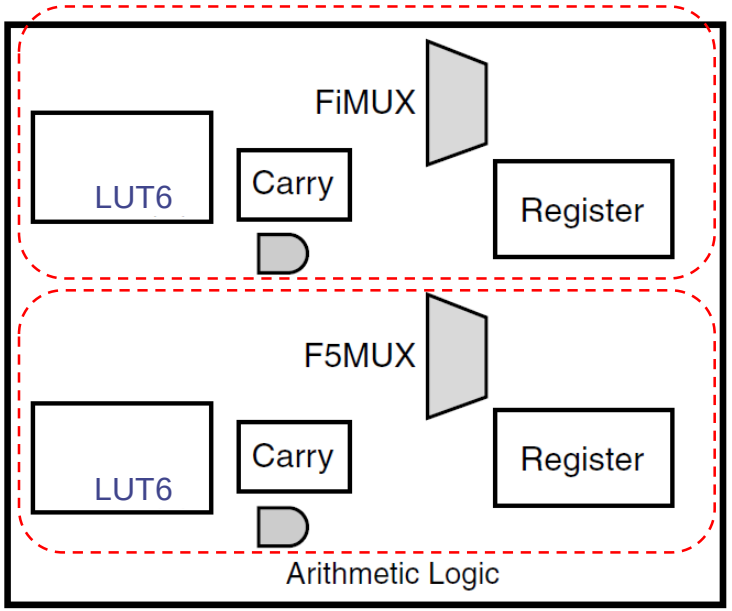


Slice Structure

- The left-hand and right-hand slices have different architecture:



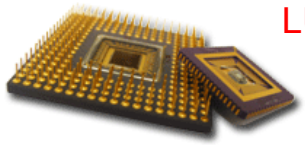
SLICEM
(M means memory)



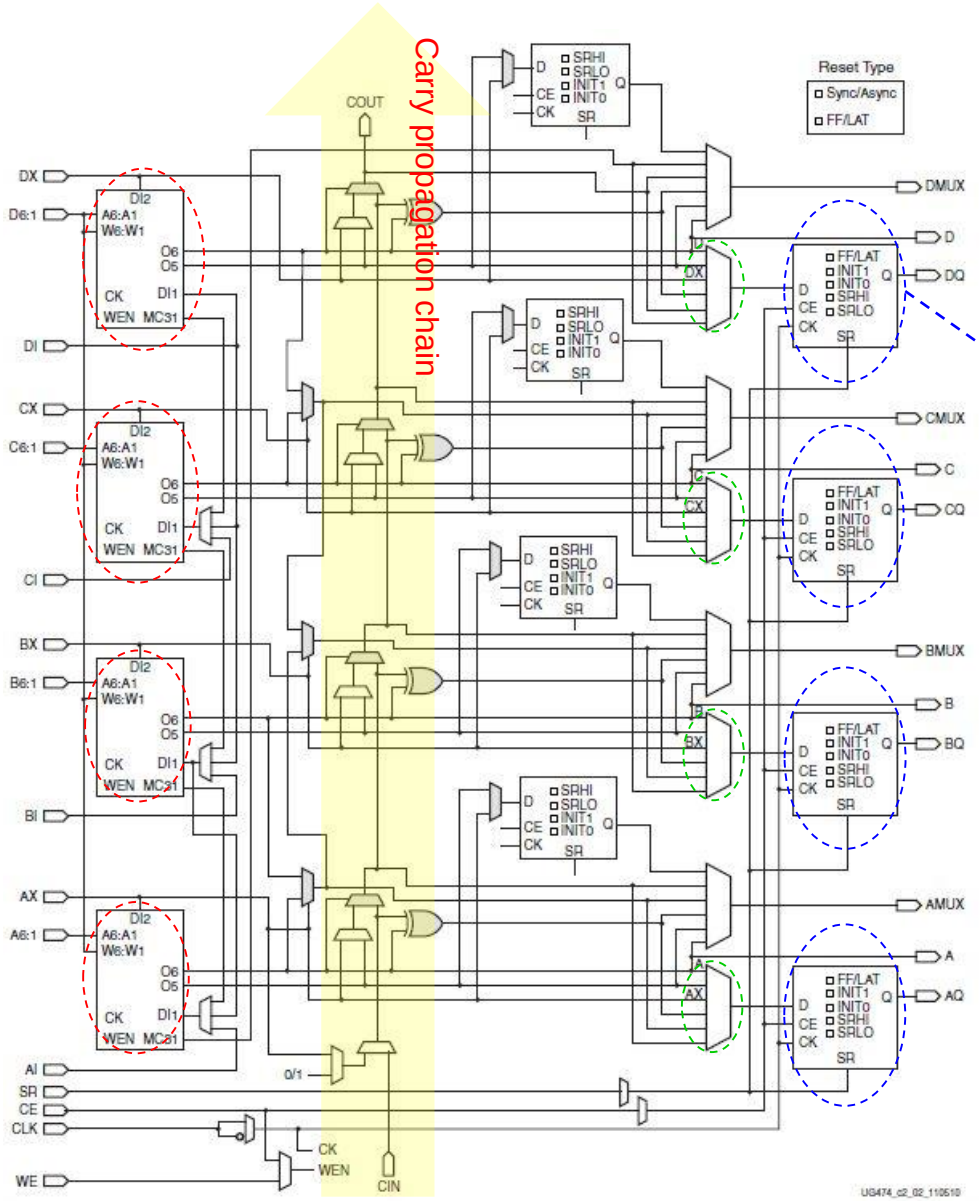
SLICEL
(L means logic)

DS312-2_13_020905

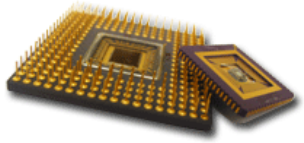
LUT6 can be used as a shift-register or a small RAM block, i.e., a distributed RAM



Artix-7 Slice Structure (SliceM)

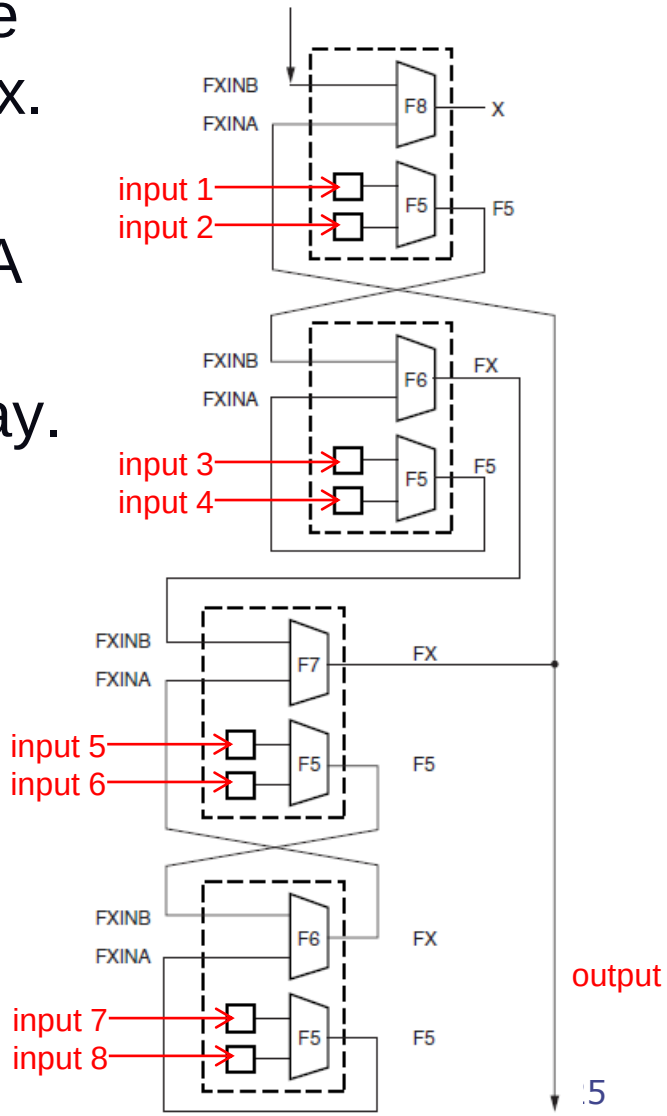


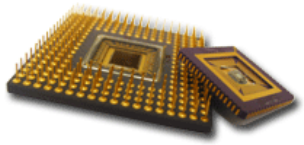
Output flip-flop can be configured as a latch!



Synthesis of a Large Multiplexor

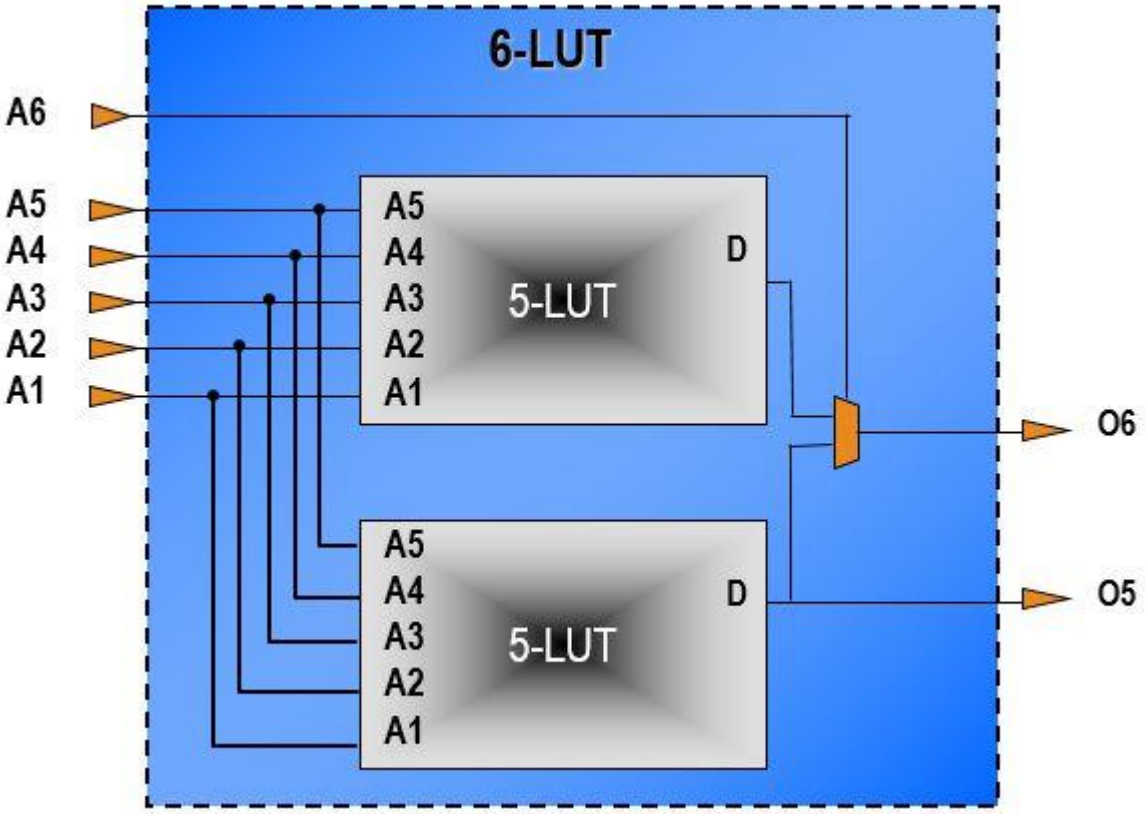
- Several 2-to-1 MUXes can be combined to form a large mux.
- A large mux consumes FPGA resources significantly and creating long signal path delay.
  - Please use them wisely.

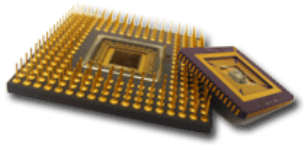




Configurable LUT6 or LUT5

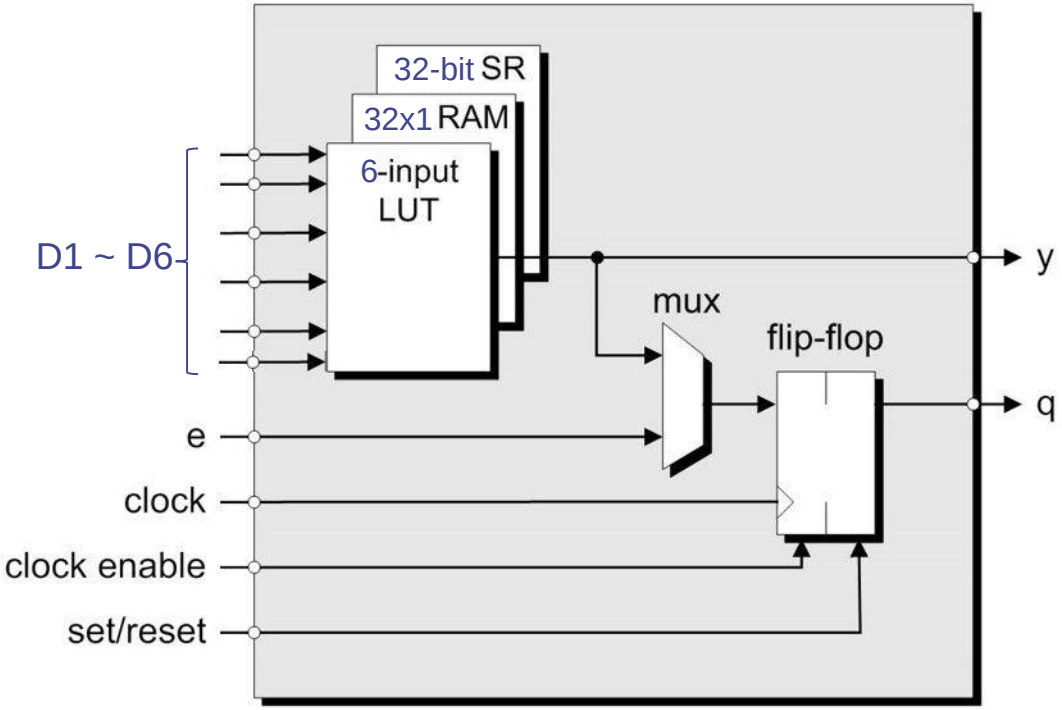
- Each 6-input LUT can also be configured as two 5-input LUT's for logic synthesis:

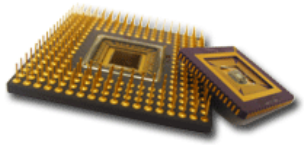




Multi-Purpose LUT in SliceM

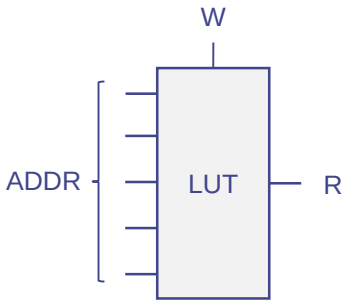
- The LUTs in SliceM can also be configured as either 32-bit shift registers or 32-bit RAM blocks (called distributed RAM).

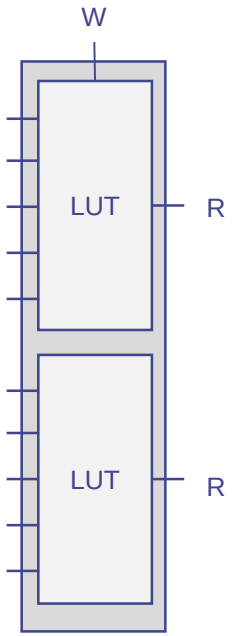


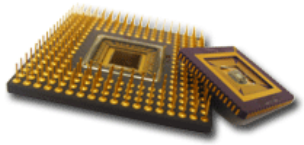


LUT as a Distributed RAM

- LUT can be used as a distributed RAM.
  - Each unit is a 32×1 RAM, where 5 of the LUT input lines becomes the address lines of a RAM block.
  - LUTs can be cascaded to increase RAM size.
  - Each LUTs can be used to Implement either
    - a 64×1-bit RAM
    - a 32×2-bit RAM
    - a 32×1-bit RAM with dual output ports
- Synchronous write
- Synchronous / Asynchronous read
  - Output flip-flop is used for synchronous read.

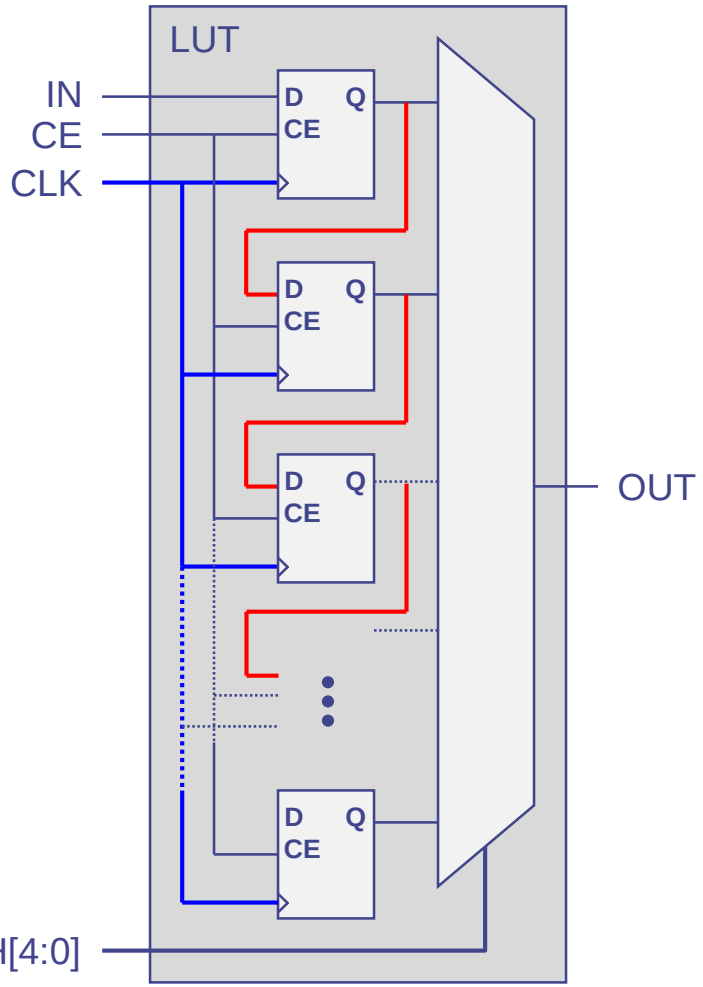


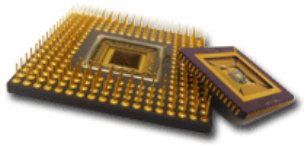




LUT as a Shift Register

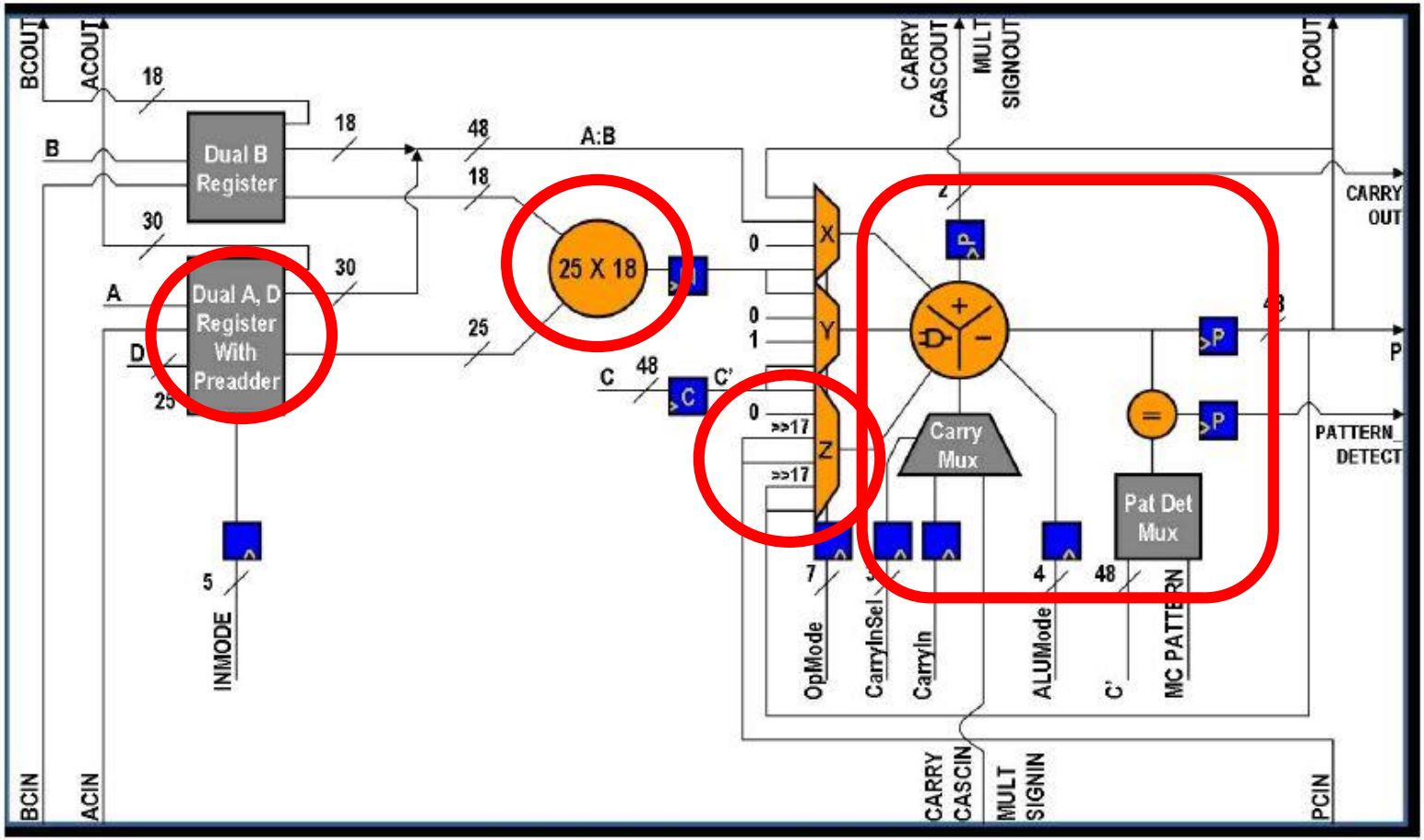
- Each LUT can be configured as a shift register.
  - Bit depth of the shift register is configurable from 1 to 32 bits.
  - The 5 of the input lines of the LUT is used to set the depth of the shift register.
- Multiple LUTs can be cascaded to form a larger shift register.





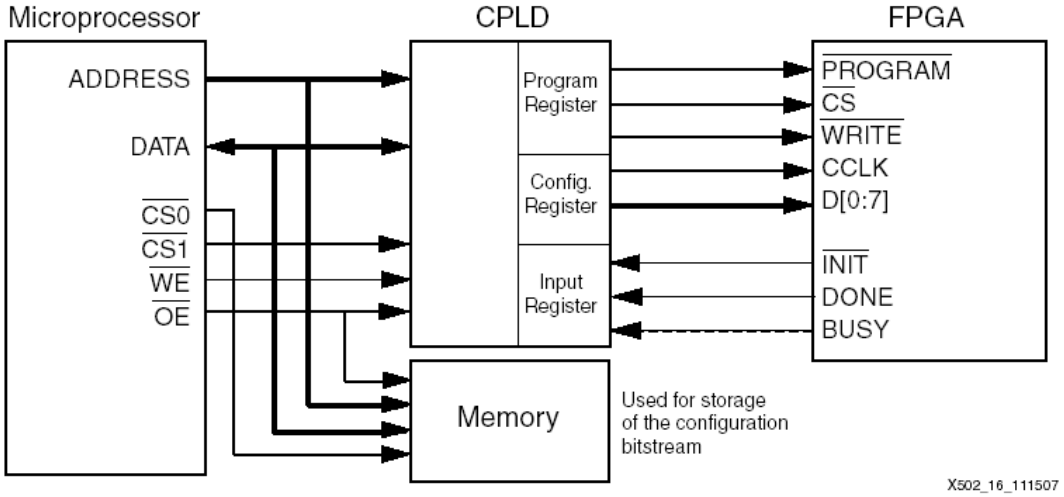
DSP Slice

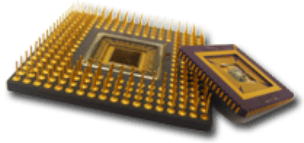
- Each DSP slice contains:
  - 25x18 multiplier, 25-bit pre-adder, 48-bit ALU, 17-bit shifter
  - Configurable pipeline



Configuring SRAM-based FPGA

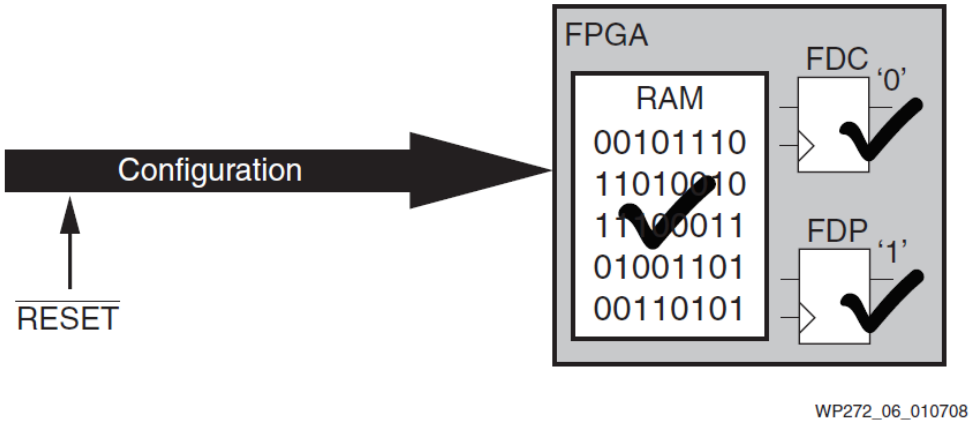
- There are several ways to configure an FPGA.
  - A host PC configures FPGA using JTAG interface, not good for “turn-key” systems.
  - FPGA boots in master mode, reads configuration data from a flash ROM (or SD card) through the SPI bus.
  - FPGA boots in slave mode, a microcontroller (through GPIO) or a CPLD can be used to configure an FPGA.



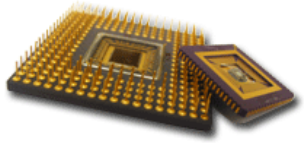


Behavior of FPGA Configuration

- When a Xilinx FPGA is configured or reconfigured, every cells are initialized.
  - Configuration has the same effect as a global reset and it sets/presets all the flip-flops and initializes all RAM cells.



FDC – D Flip-flops with asynchronous clear
FDP – D Flip-flops with asynchronous preset



References

- Mentor Graphics, *The Design Warrior's Guide to FPGAs Devices, Tools, and Flows*, ISBN 0750676043, 2004.
- Xilinx, *7 Series FPGAs Configurable Logic Block User Guide*, Xilinx UG474, v1.8, Sep. 27, 2016.
- P. Garrault and B. Philofsky, *HDL Coding Practices to Accelerate Design Performance*, Xilinx WP231, Jan. 6, 2006.
- Xilinx, *Using Digital Clock Managers (DCMs) in Spartan-3 FPGAs*, XAPP462, Jan. 2006.
- M. Peattie, *Using a Microprocessor to Configure Xilinx FPGAs via Slave Serial or SelectMAP Mode*, Xilinx AN502, Aug. 2009.