



L13: Reconfigurable Logic Architectures



Acknowledgements:

R. Katz, "*Contemporary Logic Design*", Addison Wesley Publishing Company, Reading, MA, 1993.

Frank Honore



History of Computational Fabrics



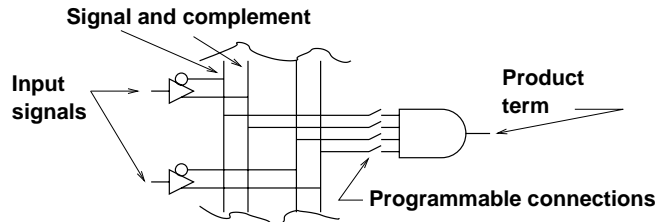
- Discrete devices: relays, transistors (1940s-50s)
- Discrete logic gates (1950s-60s)
- Integrated circuits (1960s-70s)
 - e.g. TTL packages: Data Book for 100s of different parts
- Gate Arrays (IBM 1970s)
 - Transistors are pre-placed on the chip and Place and Route software puts the chip together automatically – only program the interconnect (mask programming).
- Software-Based Schemes (1970s- present)
 - Run instructions on a general purpose core.
- ASIC Design (1980s to present)
 - Turn VHDL directly into layout using a library of standard cells
 - Effective for high-volume and efficient use of silicon area
- Programmable Logic (1980s to present)
 - A chip that be reprogrammed after it has been fabricated
 - Examples: EPROM, EEPROM, PALs, PLDs, FPGAs
 - Excellent support for mapping from VHDL



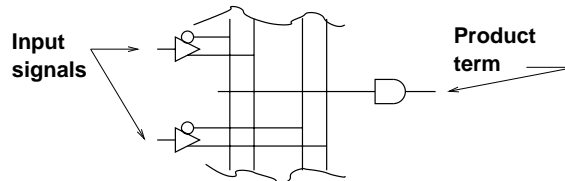
Programmable Logic



The basic element of all PALs (PLDs and CPLDs) is an AND gate which can be driven by each input and its complement. The unprogrammed state is that all connections are intact; therefore, the product term is zero. When all connections are destroyed (temporarily) the output of the AND floats high and is thus a one.



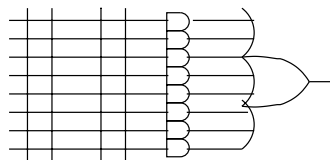
A shorthand notation (especially useful if there are a lot of inputs) is:



OR of ANDs



The product terms produced by the ANDs are combined in large ORs.



And the output from the OR can usually be registered or simply combinational. Additionally, either can be inverted or not.

Pin	20V8(1)		20V8(2)		22V10	
	Use	Terms	Use	Terms	Use	Terms
23	I		I		I/O	8
22	O	7	I/O	8	I/O	10
21	I/O	7	I/O	8	I/O	12
20	I/O	7	I/O	8	I/O	14
19	I/O	7	I/O	8	I/O	16
18	I/O	7	I/O	8	I/O	16
17	I/O	7	I/O	8	I/O	14
16	I/O	7	I/O	8	I/O	12
15	O	7	I/O	8	I/O	10
14	I		I		I/O	8
13	I		/OE		I	

20V8 (1): All outputs are combinatorial (none registered).
 (2): Some outputs registered: 8 product terms for registered outputs only.

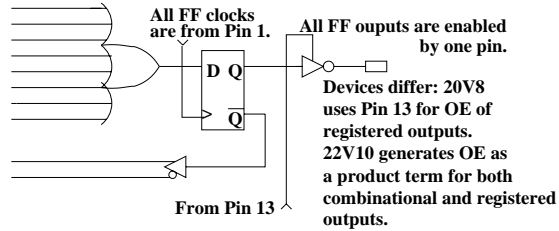


Outputs

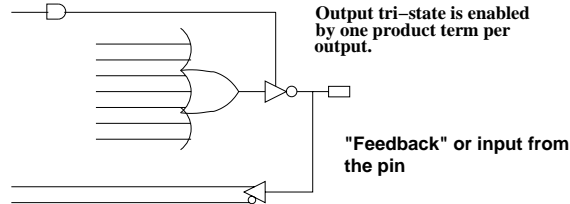


This is how the 20V8 works.

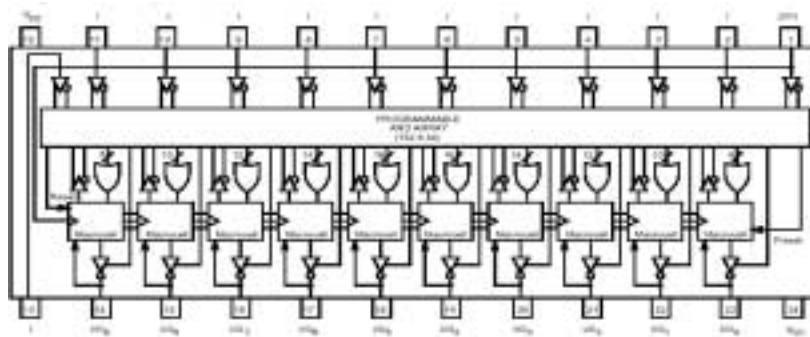
The output can be registered.

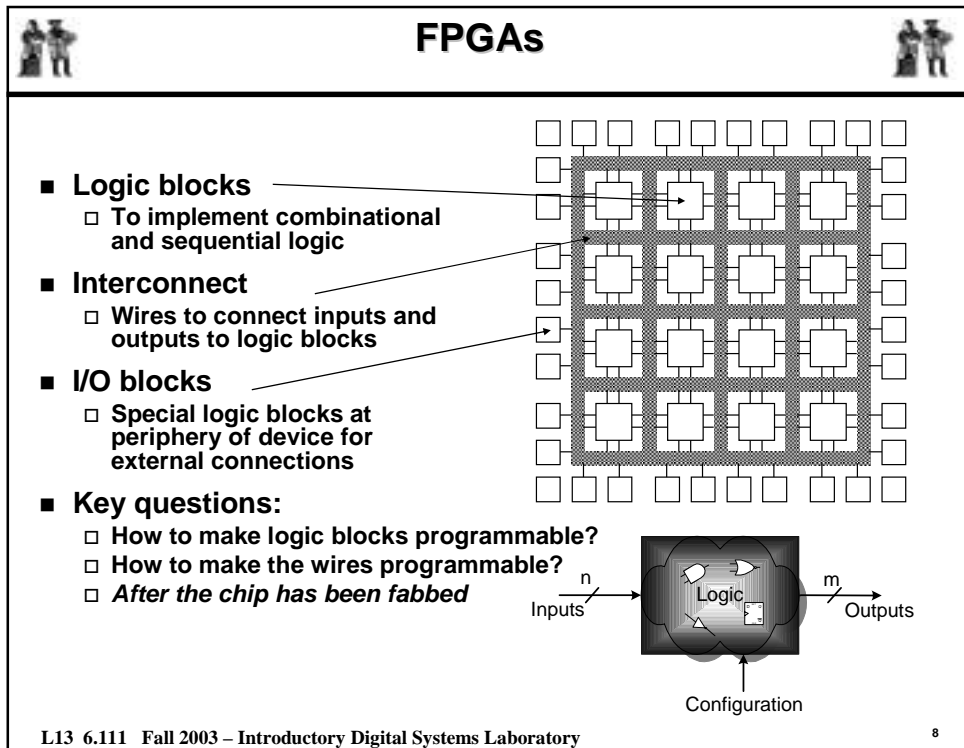
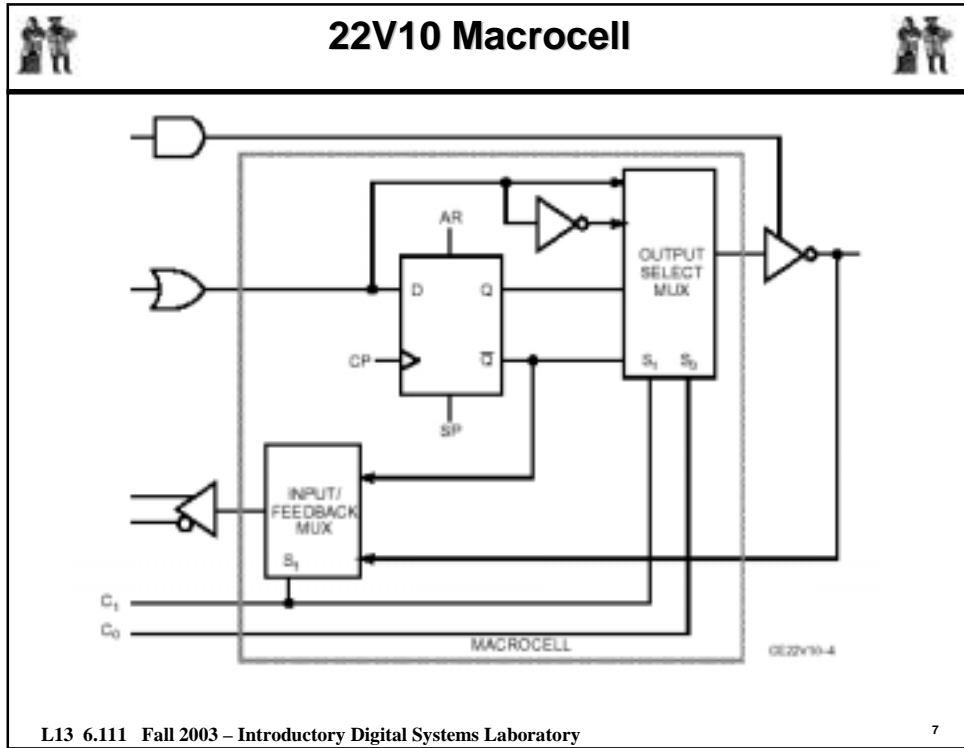


The output can be sent directly to a pin.



22v10 PAL







Trade-offs in FPGA



- **Logic block - how are functions implemented: fixed functions (manipulate inputs) or programmable?**
 - Support complex functions, need fewer blocks, but they are bigger so less of them on chip
 - Support simple functions, need more blocks, but they are smaller so more of them on chip
- **Interconnect**
 - How are logic blocks arranged?
 - How many inputs/outputs must be routed to/from each logic block?
 - Programmability slows wires down – are some wires specialized to long distances?
 - What utilization are we willing to accept? 50%? 20%? 90%?



Anti-Fuse-Based Approach (Actel)

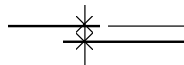


Rows of programmable logic building blocks

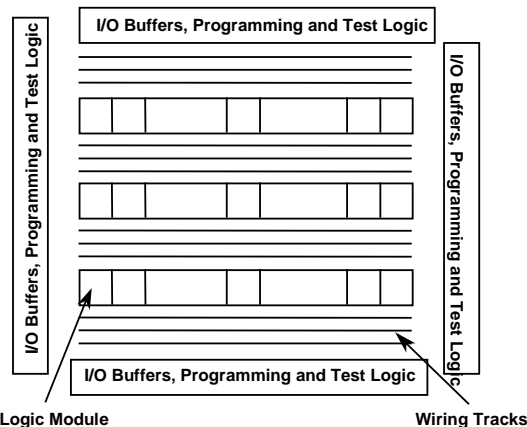
+

rows of interconnect

Anti-fuse Technology:
Program Once



Use Anti-fuses to build up long wiring runs from short segments.



8-input, single-output combinational logic blocks

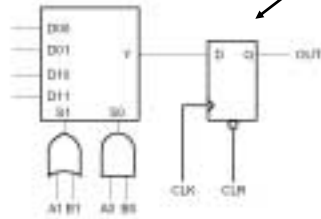
FFs constructed from discrete cross coupled gates.



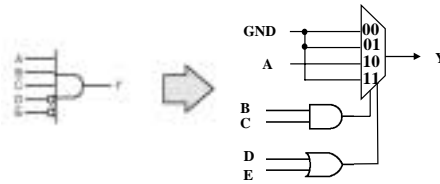
Actel Logic Module



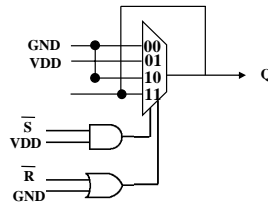
Combinational block does not have the output FF



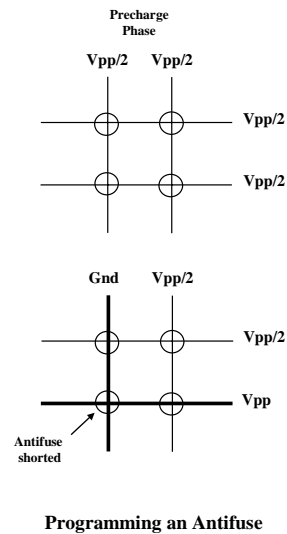
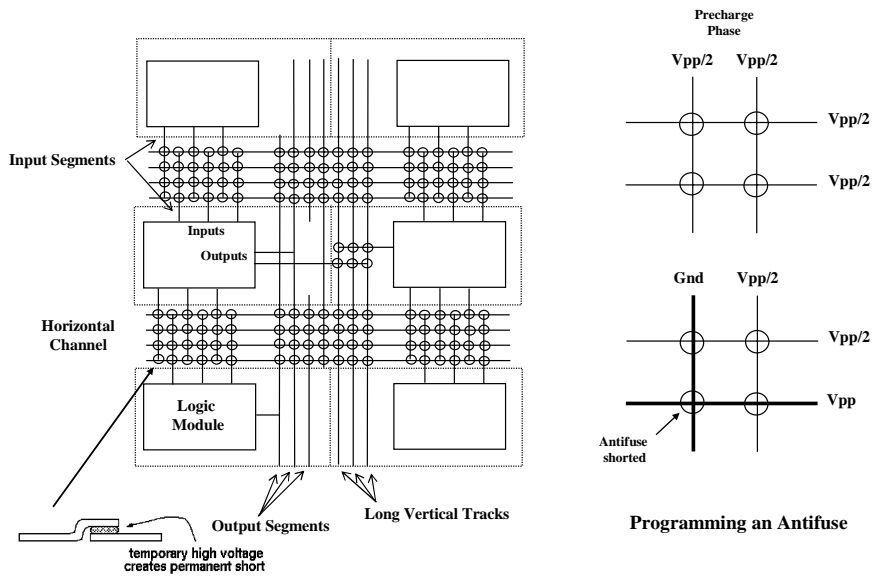
Example Gate Mapping



S-R Latch

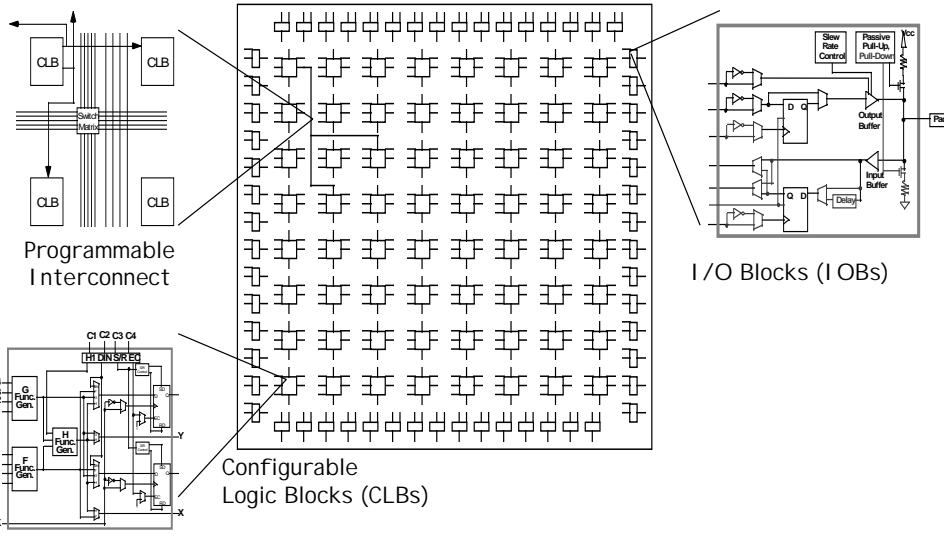


Actel Routing & Programming

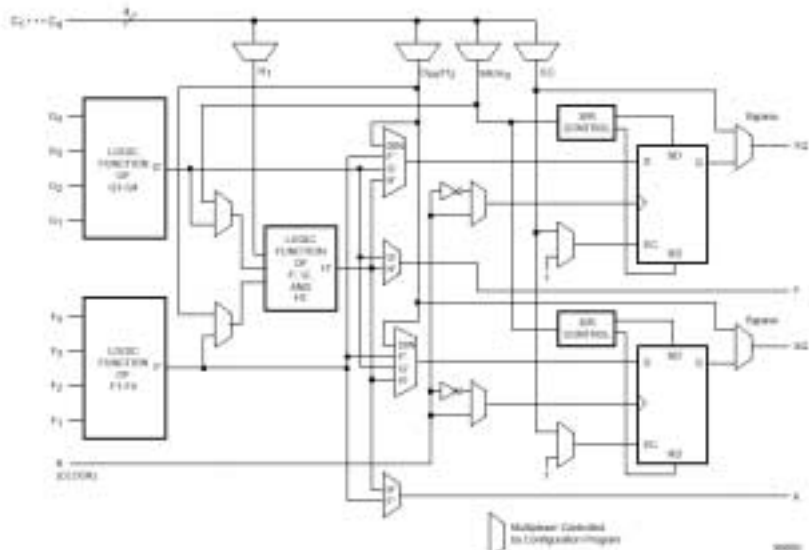




RAM Based Field Programmable Logic – Xilinx and Altera



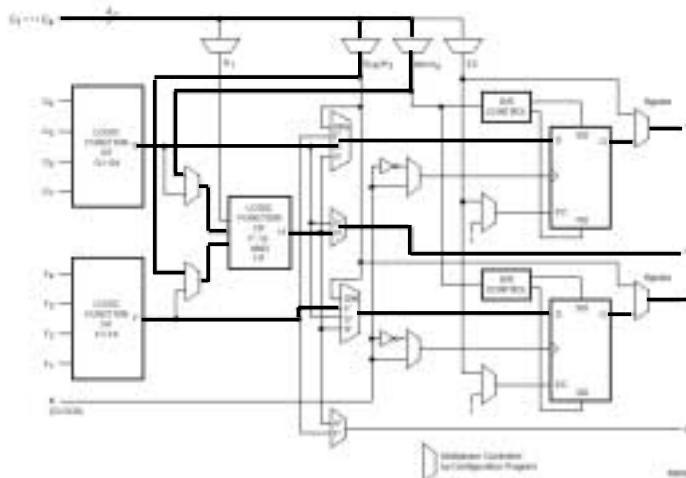
The Xilinx 4000 CLB



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



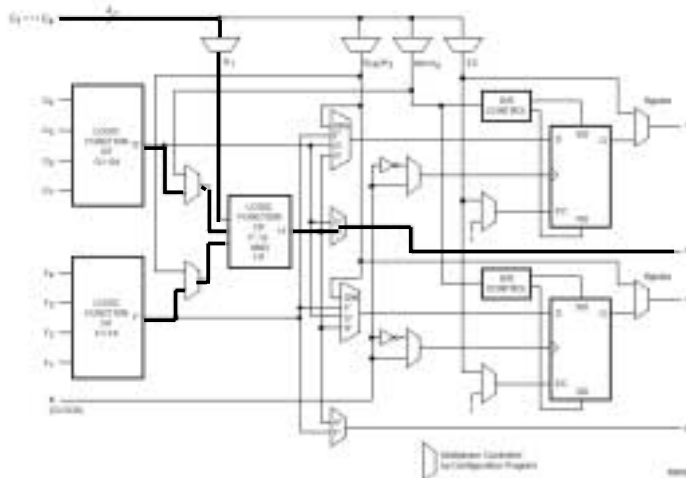
Two 4-input Functions, Registered Output



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



5-input Function, Combinational Output



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

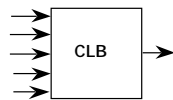


Logic Examples

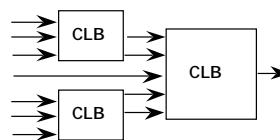


- **Key: General functions are limited to 5 inputs**
 - *No limitation on function complexity (4 even better - 1/2 CLB)*
- **2-bit comparator:**
 - $A B = C D$ and $A B > C D$ implemented with 1 CLB
 - (GT) $F = A C' + A B D' + B C' D'$
 - (EQ) $G = A'B'C'D' + A'B C'D + A B'C D' + A B C D$
- **N-input majority function: 1 whenever n/2 or more inputs are 1**

5-input Majority Circuit



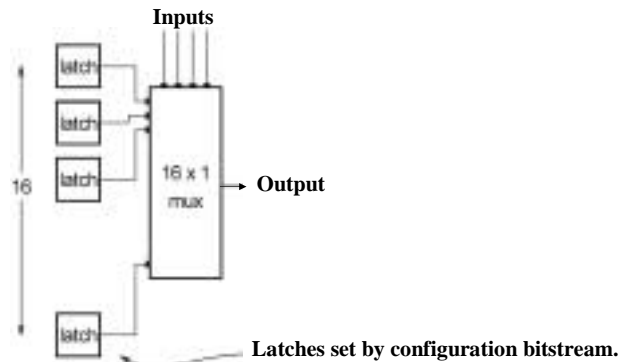
7-input Majority Circuit



LUT Mapping



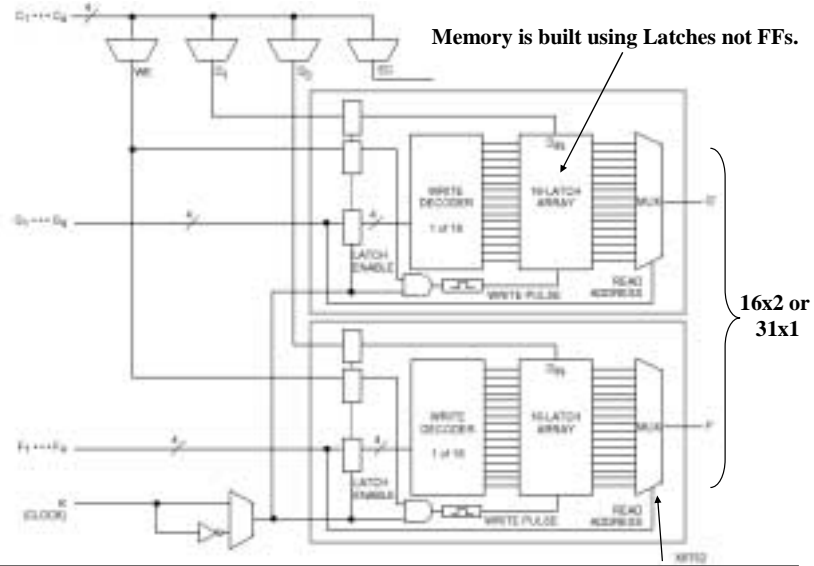
- **N-LUT is a direct implementation of a truth table: any function of n-inputs.**
- **N-LUT requires 2^N storage elements (latches).**
- **N-inputs select one latch location (like a memory).**



4LUT example



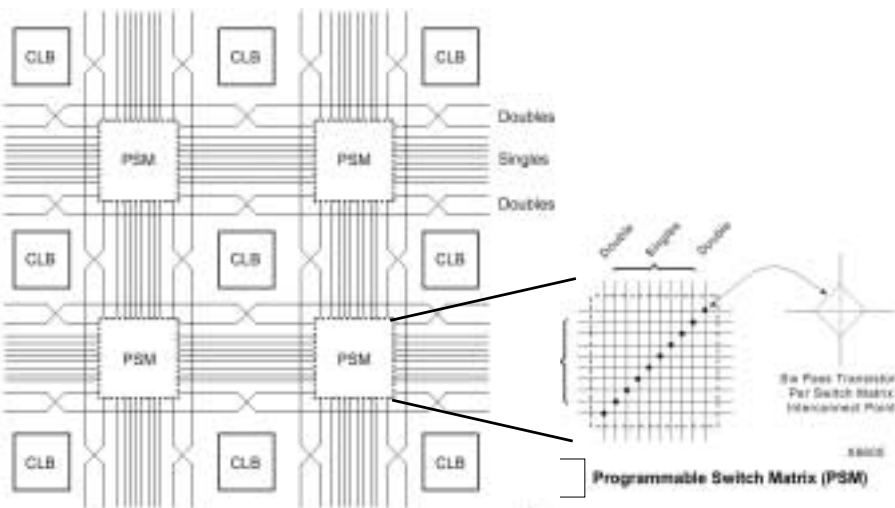
Configuring the CLB as a RAM



Read is same as a LUT Function!



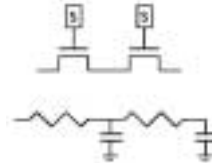
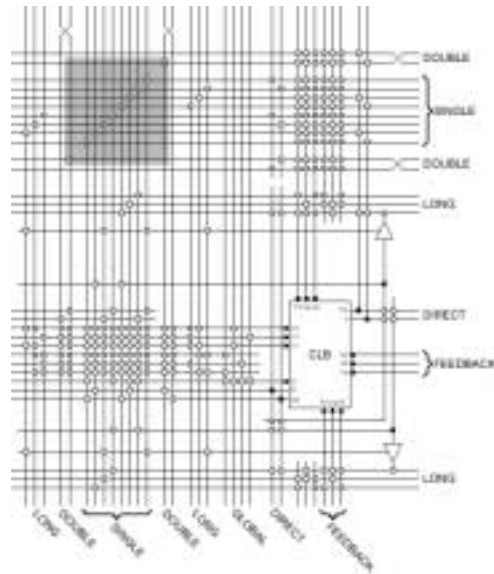
Xilinx 4000 Interconnect



Single- and Double-Length Lines, with Programmable Switch Matrices (PSMs)



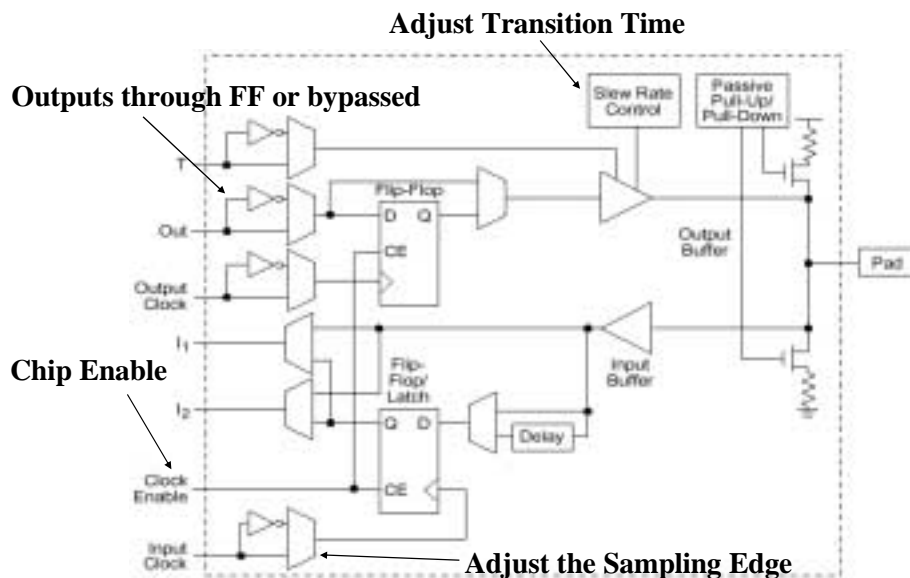
Xilinx 4000 Interconnect Details



Wires are not ideal!



Xilinx 4000 Flexible IOB





Altera's FLEX 10K Devices

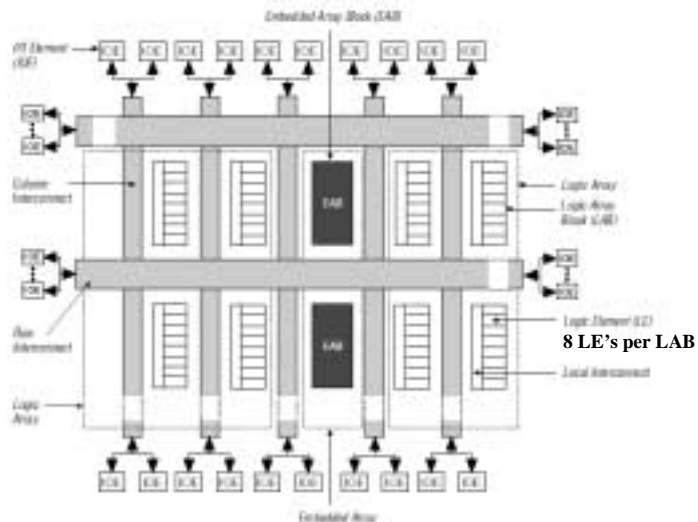


- Based on reconfigurable CMOS SRAM elements
 - This enables 100% testing prior to shipment.
 - They can (must) be configured on their pc board.
 - This facilitates field changes and special test modes.

- Each FLEX device contains both embedded and logic arrays.
 - The embedded array consists of EABs.
 - Each of which provides 2,048 bits
 - The EAB can be used to create RAM, ROM, dual-port RAM or a FIFO.
 - EABs can be used independently or they can be combined to implement larger logic functions.
 - The logic array consists of LABs.
 - They are used for general logic such as counters, adders, state machines, etc.
 - Each LAB consists of eight LEs and local interconnect.
 - An LE consists of a 4-input look-up table, a programmable FF, and dedicated signal paths for carry and cascade paths.



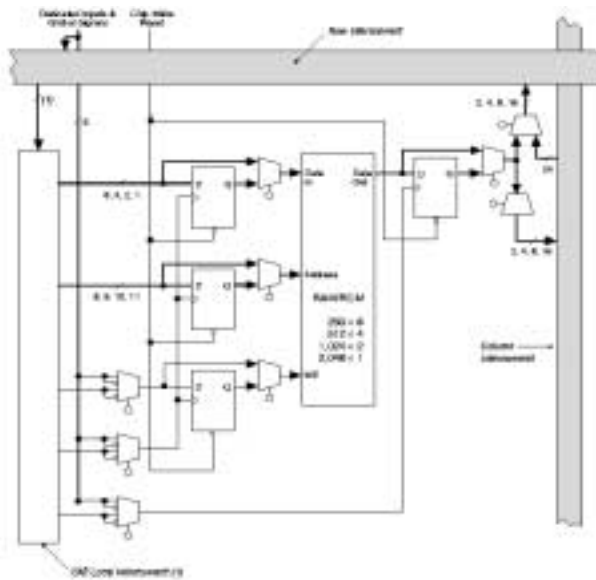
Altera FLEX 10K Block Diagram



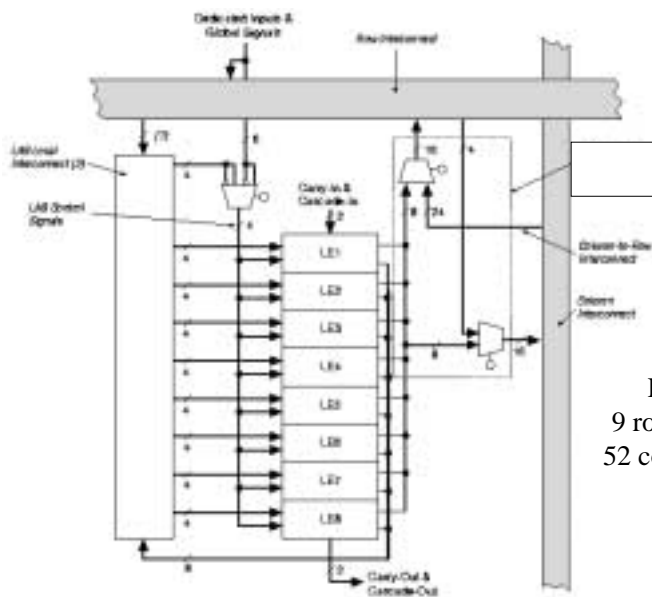
SRAM-based programming



FLEX 10K Embedded Array Block



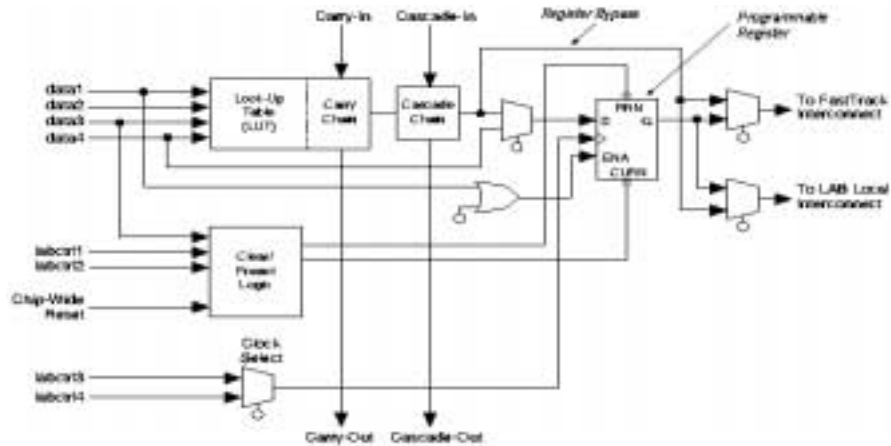
FLEX 10K Logic Array Block



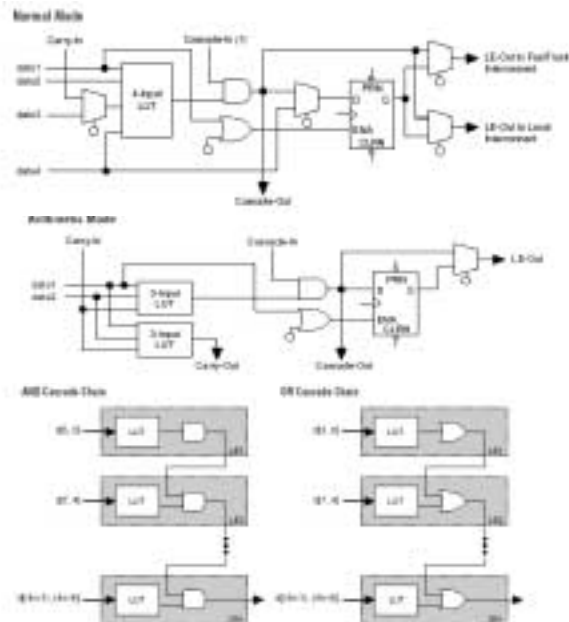
FLEX 10K70:
 9 rows (312 chan/row),
 52 columns (24 chan/col)



FLEX 10K Logic Element

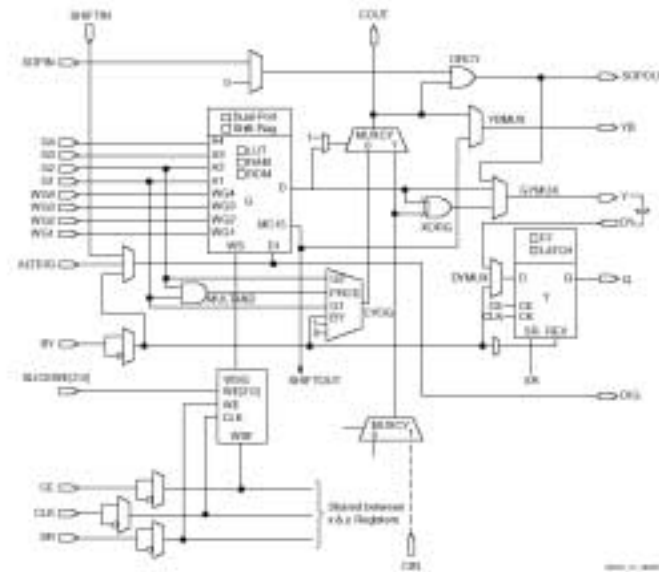


FLEX10K Operating Mode Examples

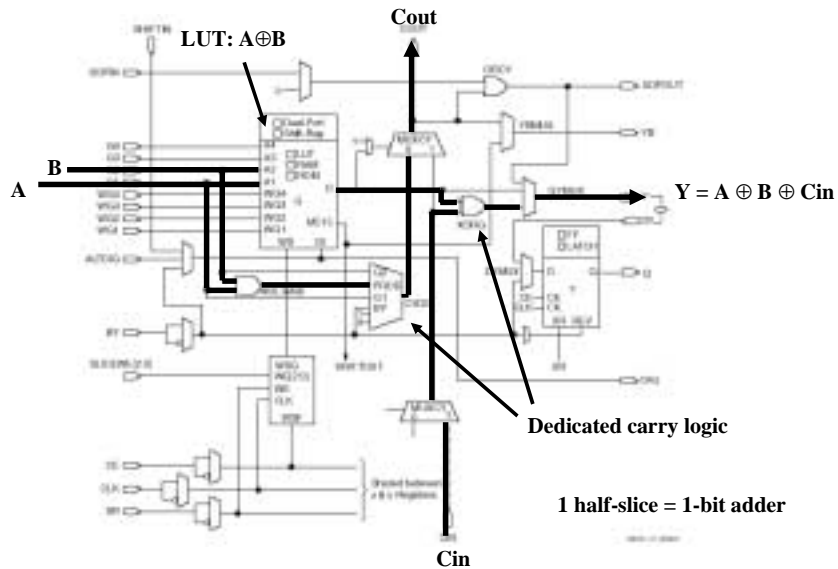




The Virtex II CLB (Half-Slice Shown)

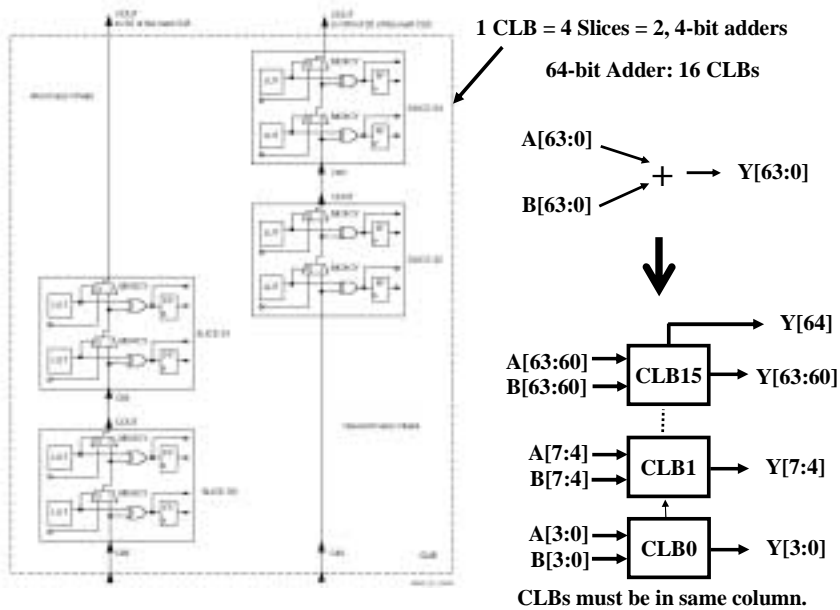


Adder Implementation

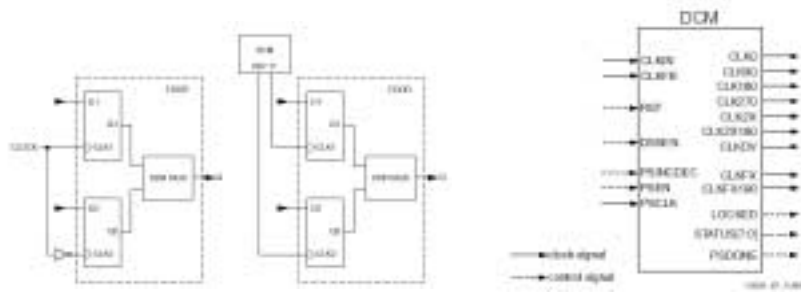




Carry Chain

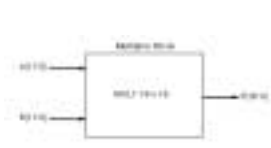


Virtex II Features

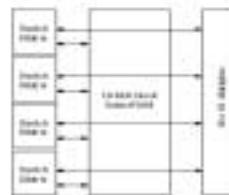


Double Data Rate Registers

Digital Clock Manager



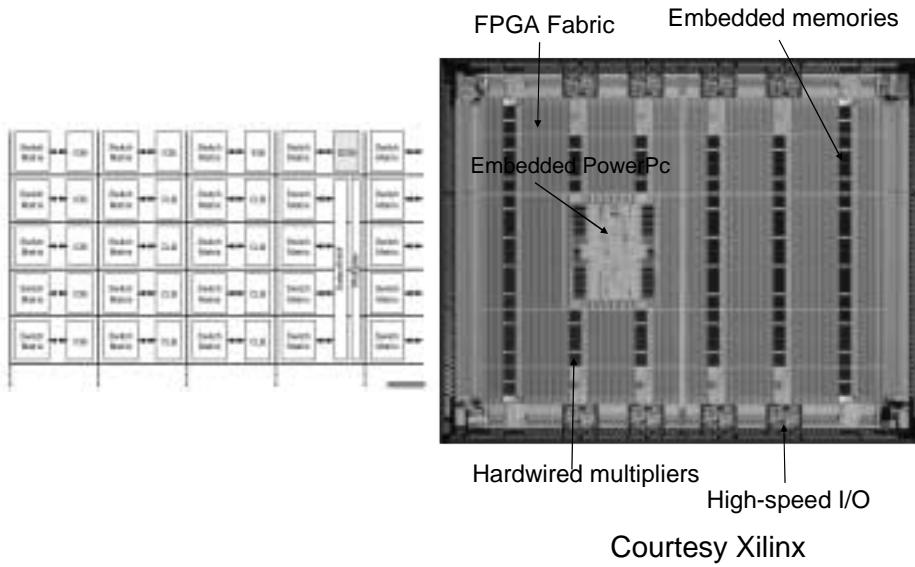
Embedded Multiplier



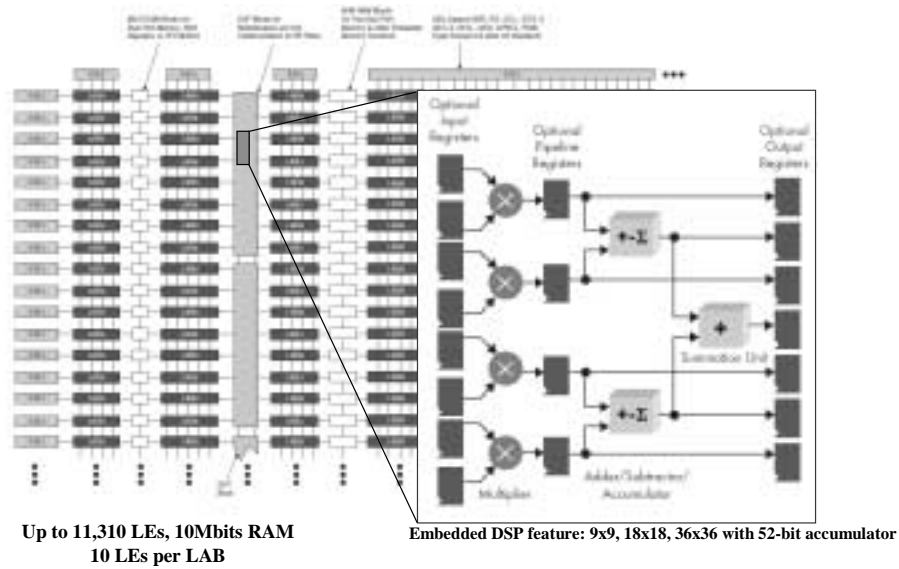
Block Select RAM



The Latest Generation: Virtex-II Pro

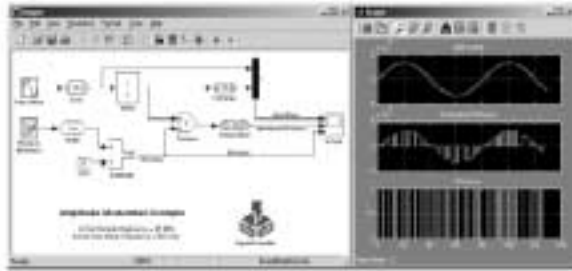


Altera's New Stratix Architecture

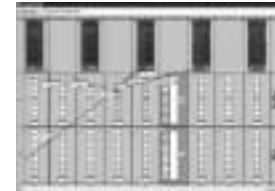




MegaCore Wizard and IP Cores



DSP Builder



Floorplan View

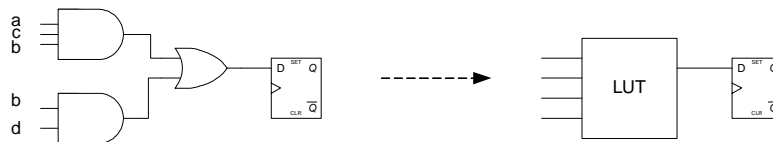
- Generates hard macro from parameterized modules
 - Already placed and routed blocks are relocatable in design.
 - Simulation models are provided for Matlab and VHDL simulators.
- Third parties can deliver Intellectual Property as black boxes.
- Allows design reuse without recompiling



Design Flow - Mapping



- **Technology Mapping: Schematic/HDL to Physical Logic units**
- **Compile functions into basic LUT-based groups (function of target architecture)**



```

process(Clock, Reset)
begin
  if (Reset = '0') then
    q <= 0;
  elsif rising_edge(clock) then
    q <= (a AND b AND c) OR (b AND d);
  end if;
end process;

```

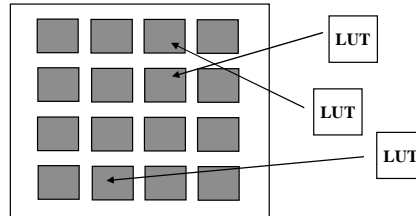
Behavioral description – fast simulation



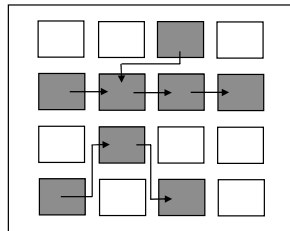
Design Flow – Place and Route



- Placement – assign logic location on a particular device



- Routing – iterative process to connect CLB inputs/outputs and IOBs. Optimizes critical path delay – can take hours or days for large, dense designs



Iterate placement if timing not met

Satisfy timing? → Generate Bitstream to config device

Challenge! Cannot use full chip for reasonable speeds (wires are not ideal).

Typically no more than 50% utilization



Example: VHDL to FPGA



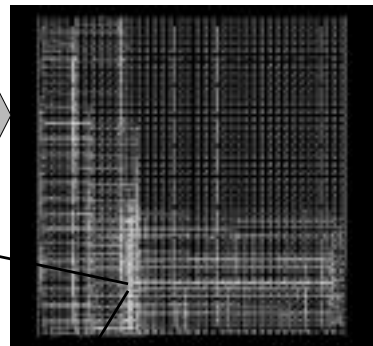
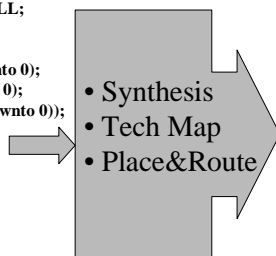
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
```

```
entity top is
  Port ( A : in std_logic_vector(63 downto 0);
        B : in std_logic_vector(63 downto 0);
        SUM : out std_logic_vector(64 downto 0));
end top;
```

```
architecture Behavioral of top is
```

```
begin
  process (A, B)
  begin
    SUM <= A + B;
  end process;
end Behavioral;
```

64-bit Adder Example



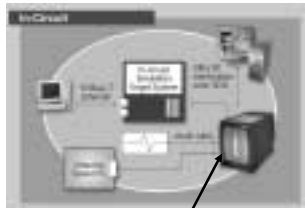
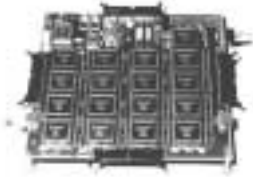
Virtex II – XC2V2000



How are FPGAs Used?



Logic Emulation



FPGA-based Emulator
(courtesy of IKOS)

■ Prototyping

- Ensemble of gate arrays used to emulate a circuit to be manufactured
- Get more/better/faster debugging done than with simulation

■ Small scale manufacturing

- When volume is not enough to justify an ASIC or a custom VLSI chip

■ Reconfigurable hardware

- One hardware block used to implement more than one function

■ Special-purpose computation engines

- Hardware dedicated to solving one problem (or class of problems)
- Accelerators attached to general-purpose computers (e.g., in a cell phone!)