- 1. Implementation technologies
- 2. Full Custom and Gate Arrays
- 3. PLD, EPLD, CPLD
- 4. FPGA
- 5. Xilinx XC6200

Available implementation technologies

- Full Custom
- Standard Cell
- Gate Array
- Field Programmable Gate Arrays (FPGAs)
- Complex PLDs (CPLDs)
- Programmable Logic Devices (PLDs)

Implementation Choices

Integrated Circuit could be:-

– PLD	100's
– Gate Array	1000's
– Standard Cell	10,000's
– Full Custom	millions

- With increasing numbers of logic gates going from 100 gates to 10 M gates

Implementation Technologies

- We can implement a design with many different implementation technologies -
  - different implementation technologies offer different tradeoffs
  - VHDL Synthesis offers an easy way to target a model towards **different implementations**
  - There are also *retargetting tools* which will convert a netlist from one technology to another
    - (from a standard cell implementation to a Field Programmable Gate Array implementation).

Full Custom

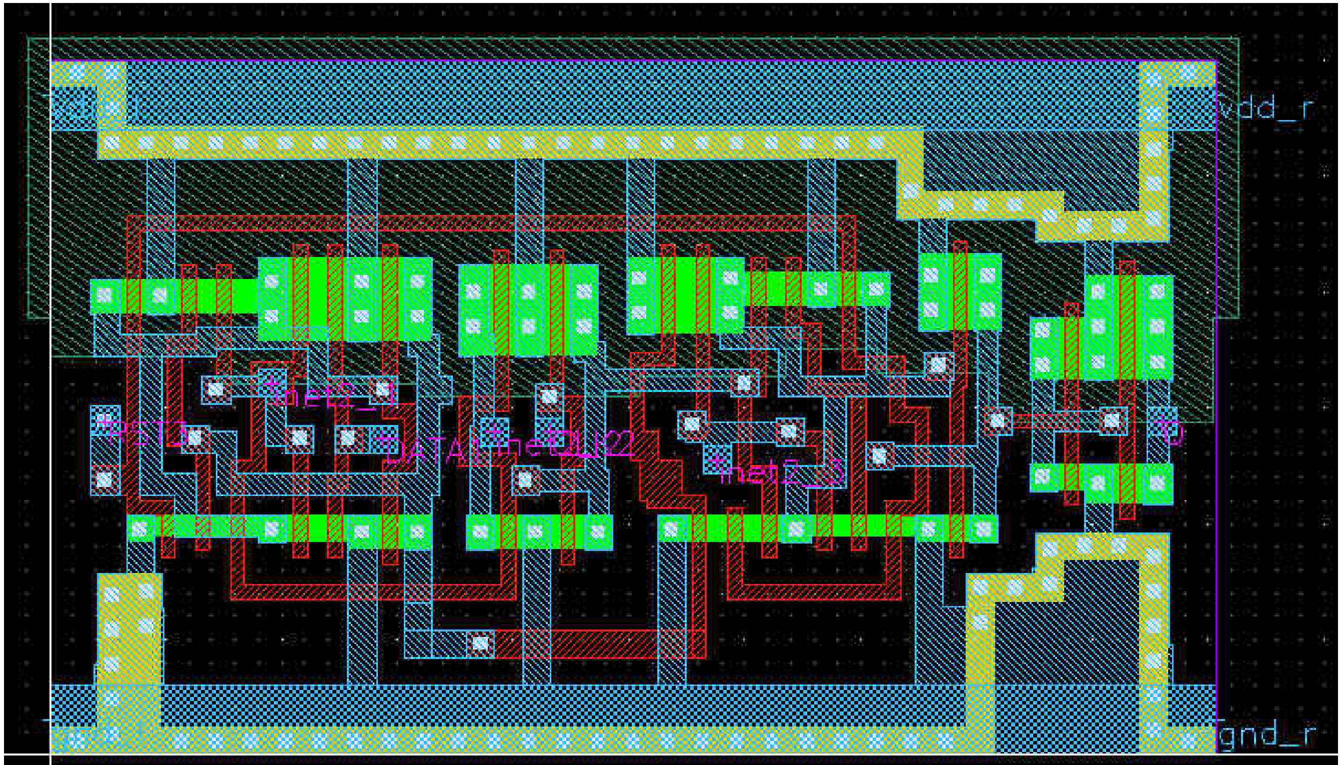
- Designer hand draws geometries which specify transistors and other devices for an integrated circuit.
  - Designer must be an expert in VLSI (Very Large Scale Integration) design.
- Can achieve **very high transistor density** (transistors per square micron); unfortunately, design time can be very long (multiple months).
- Involves the creation of a a completely new chip, which consists of about a dozen masks (for the photolitographic manufacturing process).
  - Mask creation is the expensive part.

Full Custom (cont)

- Offers the chance for **optimum performance**.
  - Performance is based on available process technology, designer skill, and CAD tool assistance.
- Fabrication costs are high - all custom masks must be made so non-recurring engineering costs (NRE) is high (in the thousands of dollars).
  - If required number of chips is high then can spread these NRE costs across the chips.
- The first custom chip costs you about $200,000, but each additional one is much cheaper.

Full Custom (cont)

- Fabrication time from geometry submission to returned chips is at least 6-8 weeks.
- Full custom is currently the only option for mixed Analog/Digital chips.
- An example VLSI layout is shown below.

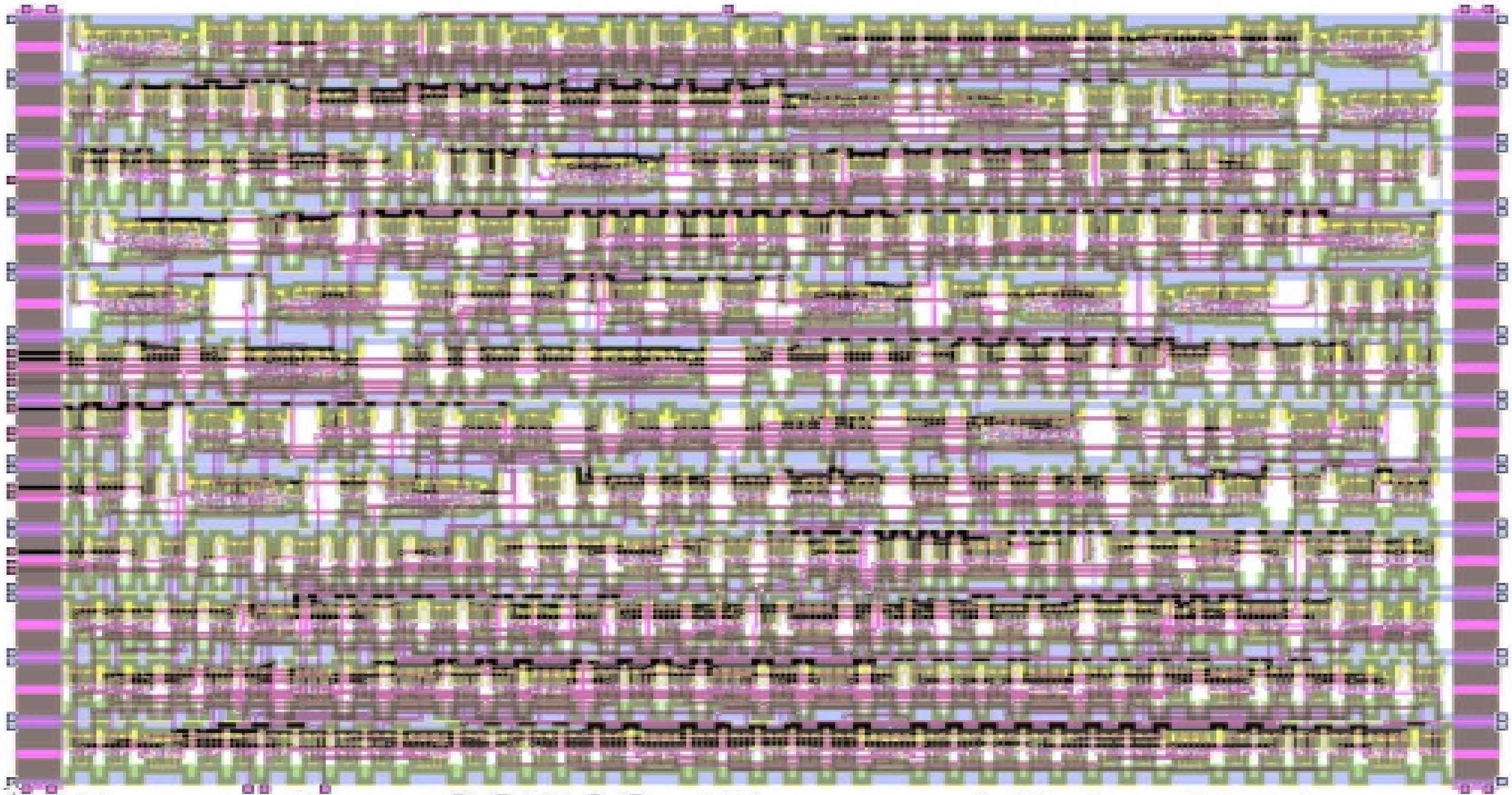


Standard Cell

- Designer uses a library of standard cells; an automatic place and route tool does the layout.
  - Designer does not have to be a VLSI expert.
- *Transistor density* and performance degradation depends on type of design being done.
  - Not bad for random logic, can be significant for data path type designs.
    - Quality of available library and tools make a significant difference.
- Design time can be much faster than full custom because layout is automatically generated.

Standard Cell (cont)

- Still involves creation of custom chip so all masks must still be made; manufacturing costs <u>same as full custom.</u>
- Fabrication time <u>same as full custom</u>.



Bus arbiter - GCMOS - Mentor 619.2 x 385.2 um

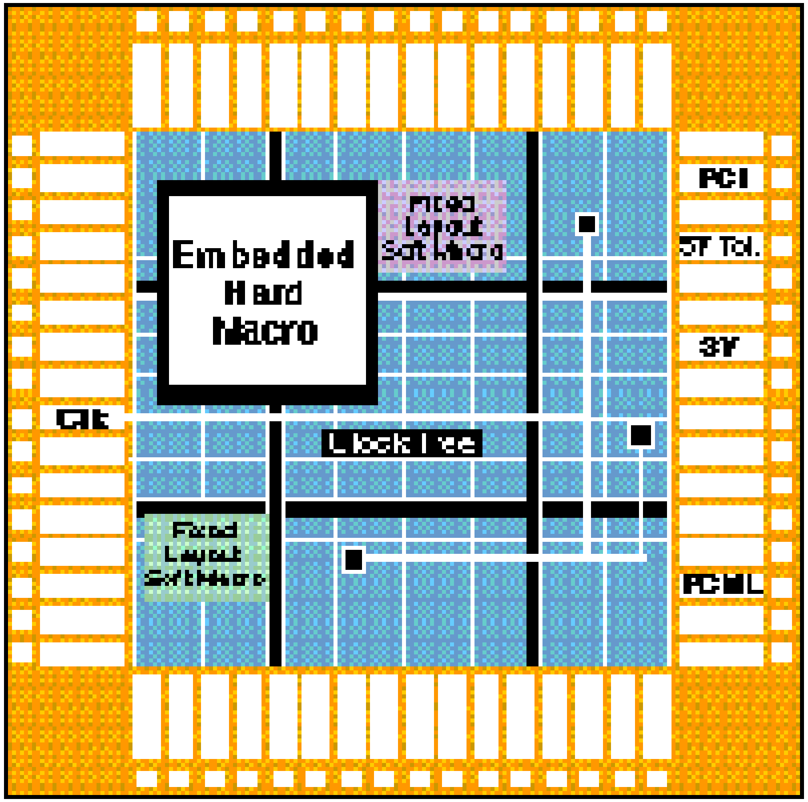
Gate Array

- Designer uses a library of standard cells.
- The design is mapped onto an array of transistors which is already created on a wafer; wafers with transistor arrays can be created ahead of time.
- A *routing tool* creates the masks for the routing layers and "customizes" the pre-created gate array for the user's design.
- Transistor density can be almost as good as standard cell.
- Design time advantages are the same as for standard cell.
- Performance can be very good;
  - again, depends on quality of available library and routing tools.

Gate Array (cont)

- Fabrication costs are cheaper than standard cell or full custom because the gate array wafers are mass produced;
- the non recurring engineering costs are lower
  - because only a few (1-3) unique routing masks have to be created for each design.
- Fabrication time can be extremely short (1-2 weeks)
  - because the wafers are already created and are only missing the routing layers.
  - The more routing layers, the higher the cost, the longer the fabrication time, but the better usage of the available transistors on the gate array.
- Almost all high volume production of complex digital designs are done in either Standard Cell or Gate Array
  - **Gate arrays** used to be more popular, but recently Standard cells has shown a resurgence in use.

Cores & Macros



- Fujitsu provides system-on-a-chip solutions.
- The company's diverse offering of reusable building block ranging from complex microprocessors to mixed signal functionality cores that allow customers to introduce products with a competitive edge and in a timely manner.

PALs, EPLDs, etc.

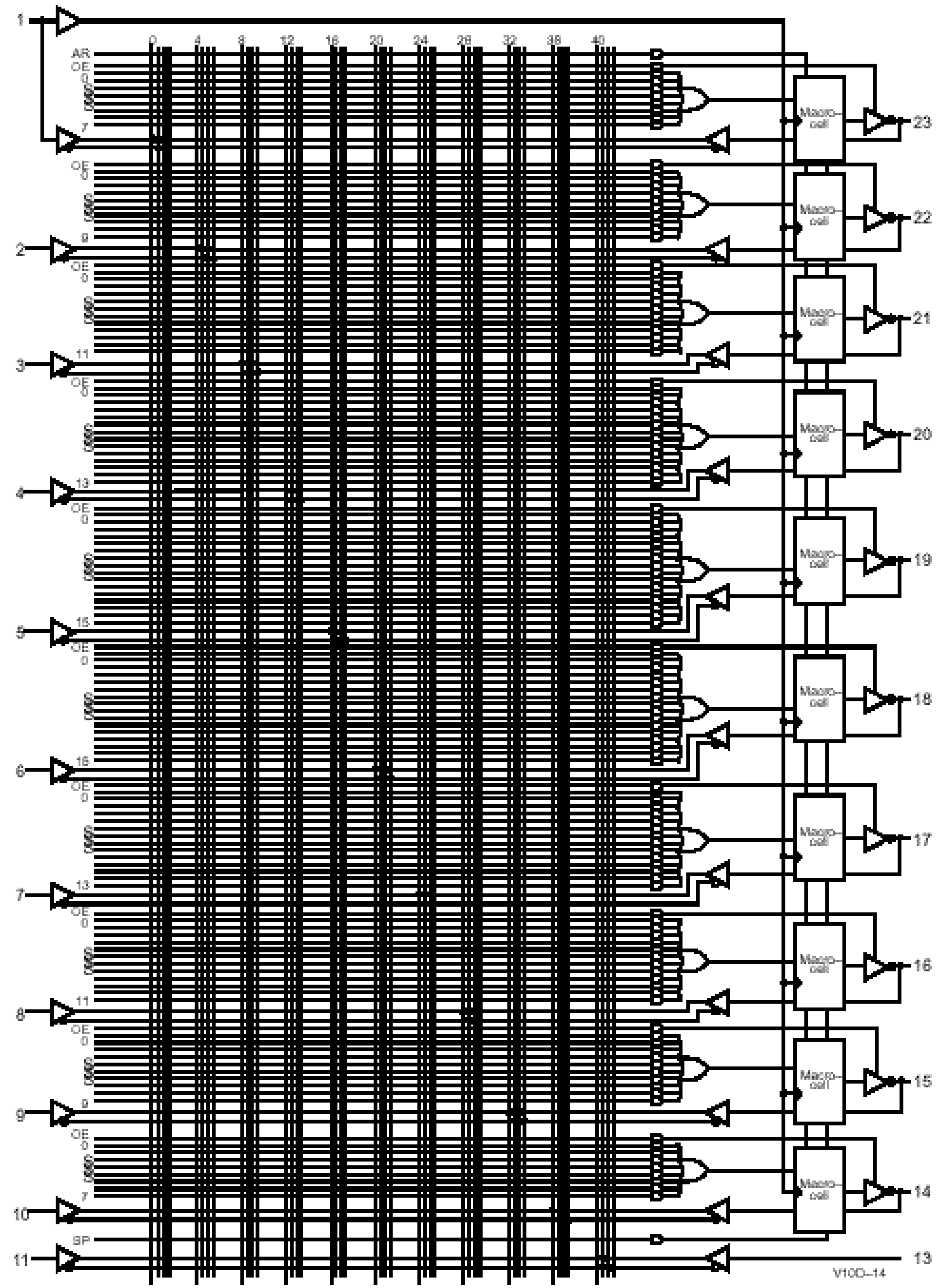
- So far, have only talked about PALs (see 22V10 figure next page).
- What is the next step in the evolution of PLDs?
  - More gates!
- How do we get more gates? We could put several PALs on one chip and put an interconnection matrix between them!!
  - This is called a Complex PLD (CPLD).

PALs (Programmable Array Logic)

- An early type of programmable logic - still in common use today.
- Logic is represented in SOP form (Sum of Products)
- The number of PRODUCTs in an SOP form will be limited to a fixed number (usually 4-10 Product terms).
- The number of VARIABLEs in each product term limited by number of input pins on PLD (usually a LOT, minimum of 10 inputs
- The number of independent functions limited by number of OUTPUT pins.

22V10 PLD

Functional Logic Diagram for PALC22V10D



Programmable Logic

- Logic devices which can be programmed/configured **on the desktop.**
- Three families (in increasing density)
  - PALS (Programmable Array Logic), Programmable Logic Devices
  - Complex PLDs
  - Field Programmable Gate Arrays
- It should be noted that memories are the earliest type of programmable logic

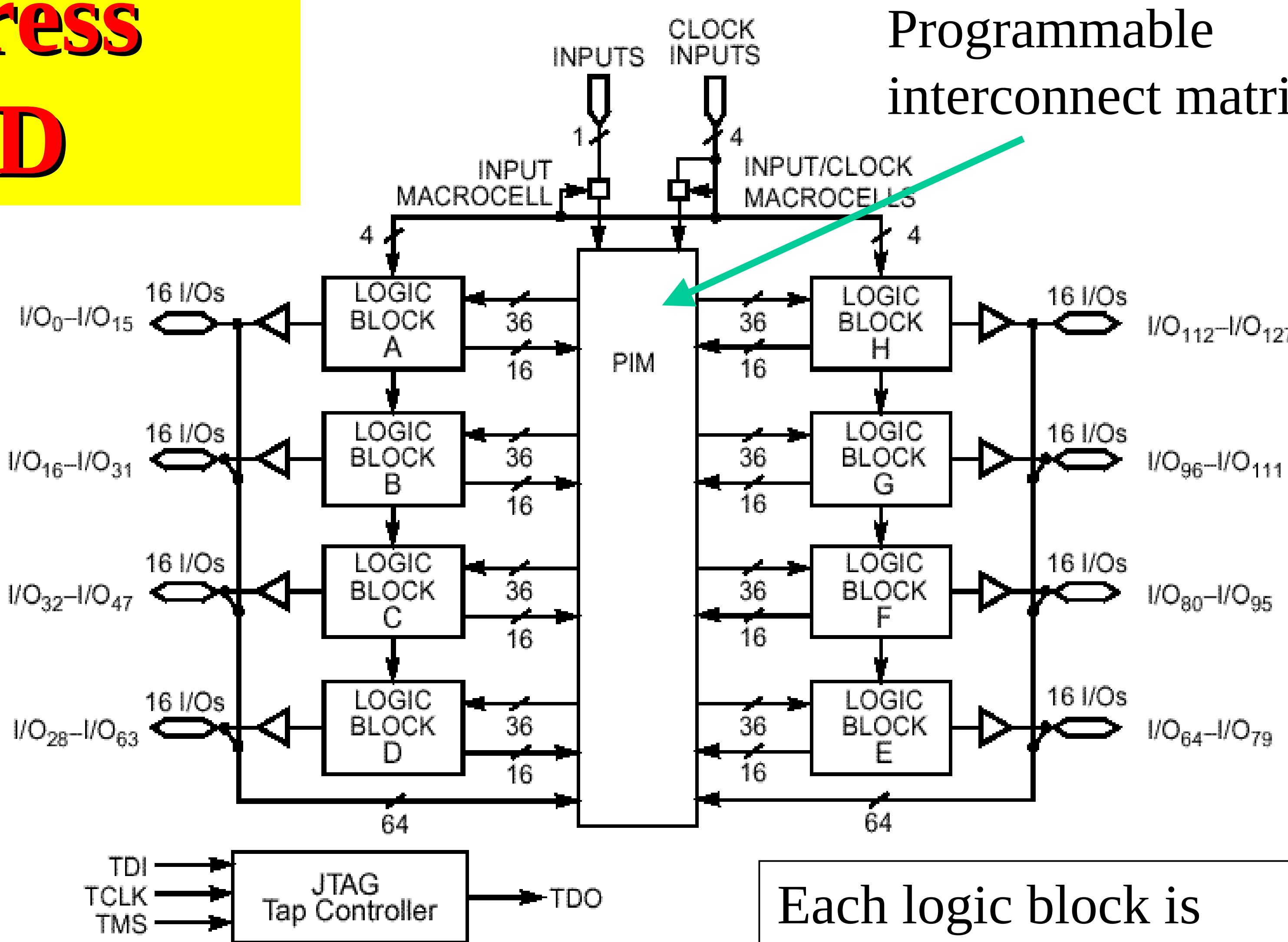
CPLD

Complex PLDs

- What is the next step in the evolution of programmable logic?
  - **More gates!**
- How do we get more gates?
- We could put several PALs on one chip and put an interconnection matrix between them!!
  - This is called a ***Complex PLD (CPLD).***

Cypress CPLD

Programmable interconnect matrix.



Each logic block is similar to a 22V10.

Logic block diagram

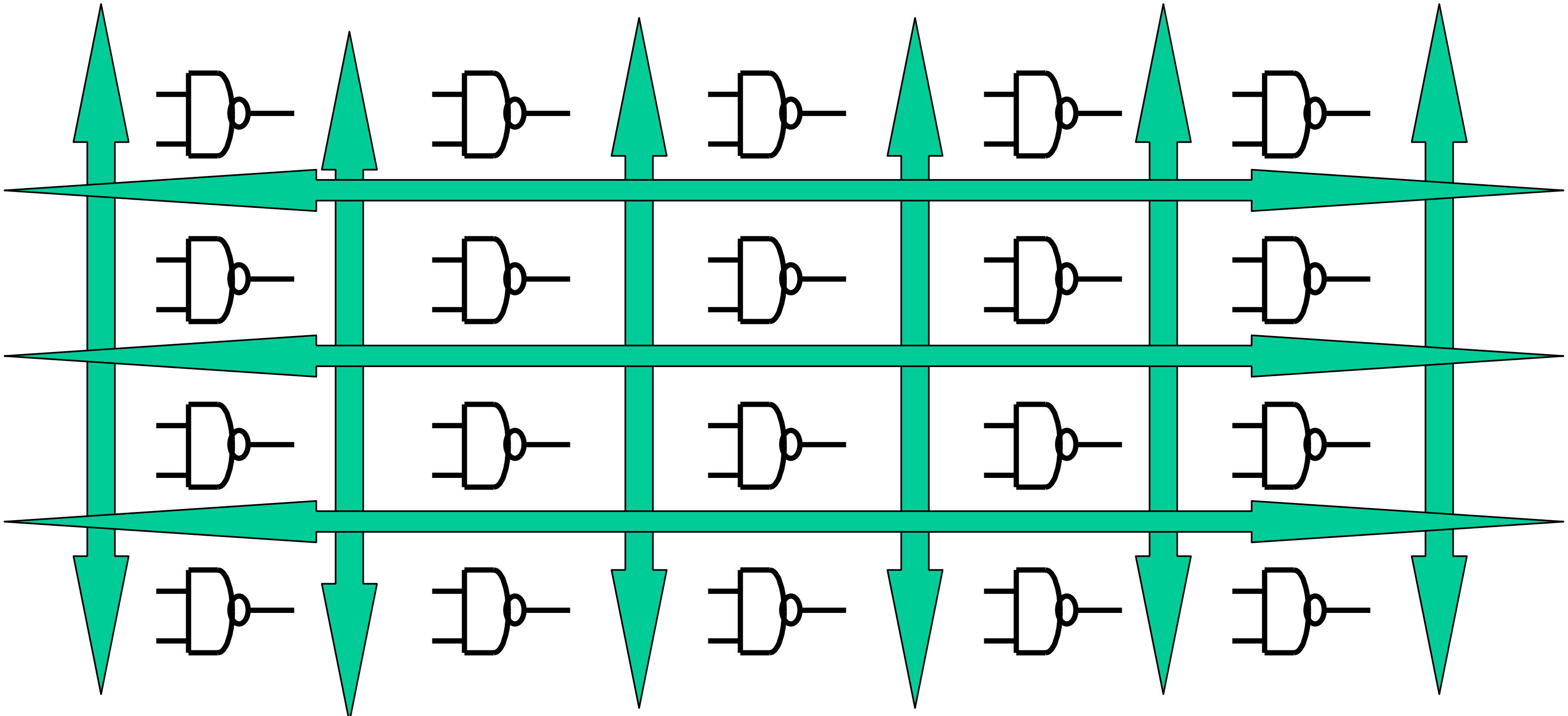
Figure 1. Ultra37128 Block Diagram

Cypress CPLDs

- Ultra37000 Family
  - 32 to 512 Macrocells
  - Fast (Tpd 5 to 10ns depending on number of macrocells)
  - Very good routing resources for a CPLD

Other approaches and Issues

- Another approach to building a “better” PLD is
    - place a lot of primitive gates on a die,
    - and then place programmable interconnect between them:



Other FPGA features

- Besides primitive logic elements and programmable routing, some FPGA families add **other features**
- ***Embedded memory***
  - Many hardware applications need memory for data storage. Many FPGAs include blocks of RAM for this purpose
- Dedicated logic for carry generation, or other arithmetic functions
- Phase locked loops for clock synchronization, division, multiplication.

Other FPGA Comments

- **Performance** is usually several factors to an order of magnitude lower than standard cell.
  - Performance depends heavily on quality of FPGA technology.
- **Design time** advantages are the same as for standard cell (use same type of cell/macro library).
- **Densities** are an order of magnitude lower than standard cell but an order of magnitude higher than normal PLDs.
- Very good for **prototype design** because many FPGAs are re-usable.
- Can be used to prototype and verify designs before investing in technologies with high start-up costs (e.g. full custom).

Programmability Options

- PLDs, CPLDs, and FPGAs have different types of programmability.
- **One time programmable:**
  - Part is programmed once and holds its programming "forever".
  - Not reusable, but usually the cheapest.
- **UV-Erasable:**
  - Erasable with UV light.
  - Needs a ceramic package with window; package adds expense to part.
  - Programming retained after power down.
  - Programming/Erasing limited to 1000s of cycles.
- **Electrically Erasable:**
  - Both reprogramming and erasing is electrical.
  - Part can programmed/erased on circuit board, no special packaging needed.
  - Erase time much faster than UV erase.
  - Programming retained after power down.
  - Programming/Erasing limited to 1000s of cycles.

Programmability Options (cont.)

- **Static Random Access Memory** (SRAM) Programming:
  - Configuration bits are stored in SRAM.
    - Can be reprogrammed infinite number of times.
  - Programming contents NOT retained after power down;
    - FPGA must be 'configured' every time on power up.
  - External non-volatile memory device required to hold device programming;
    - on power up contents of external device transferred to FPGA to configure the device.
  - Altera, Xilinx corporations offer this type of FPGAs.
- Highest density FPGAs use SRAM for configuration bits.

FPGA

What is an FPGA?

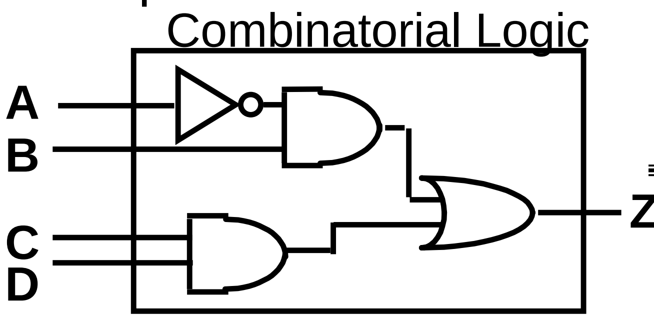
- Field Programmable Gate Array
- Fully programmable alternative to a customized chip
- Used to implement functions in hardware
- Also called a Reconfigurable Processing Unit (RPU)

Reasons to use an FPGA

- Hardwired logic is very fast
- Can interface to outside world
  - Custom hardware/peripherals
  - “Glue logic” to custom co/processors
- Can perform bit-level and systolic operations not suited for traditional CPU/MPU

Look Up Tables

- Combinatorial Logic is stored in 16x1 SRAM Look Up Tables (LUTs) in a CLB
- Example:



Look Up Table

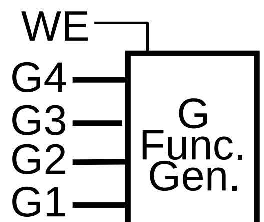
4-bit address

A	B	C	D	Z
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
	.	.	.	
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

$2^{(2^4)}$

$= 64K$!

- Capacity is limited by number of inputs, not complexity
- Choose to use each function generator as 4 input logic (LUT) or as high speed sync.dual port RAM



Field Programmable Gate Arrays

The FPGA approach to arrange primitive logic elements (logic cells) arrange in rows/columns with programmable routing between them.

What constitutes a primitive logic element?

Lots of different choices can be made! Primitive element must be classified as a "complete logic family".

- A primitive gate like a NAND gate
- A 2/1 mux (this happens to be a complete logic family)
- A Lookup table (I.e, 16x1 lookup table can implement any 4 input logic function).

Often combine <u>one of the above</u> with a DFF to form the primitive logic element.

Issues in FPGA Technologies

- Complexity of Logic Element
  – How many inputs/outputs for the logic element?
  – Does the basic logic element contain a FF? What type?
- Interconnect
  – How fast is it? Does it offer 'high speed' paths that cross the chip? How many of these?
  – Can I have on-chip tri-state busses?
  – How routable is the design? If 95% of the logic elements are used, can I route the design?
    - More routing means more routability, but less room for logic elements

Issues in FPGA Technologies (cont)

- **Macro elements**
  - Are there SRAM blocks? Is the SRAM dual ported?
  - Is there fast adder support (i.e. fast carry chains?)
  - Is there fast logic support (i.e. cascade chains)
  - What other types of macro blocks are available (fast decoders? register files?)
- **Clock support**
  - How many global clocks can I have?
  - Are there any on-chip Phase Logic Loops (PLLs) or Delay Locked Loops (DLLs) for clock synchronization, clock multiplication?

Issues in FPGA Technologies (cont)

- What type of **IO support** do I have?
  - TTL, CMOS are a given
  - Support for mixed 5V, 3.3v IOs?
    - 3.3 v internal, but 5V tolerant inputs?
  - Support for new low voltage signaling standards?
    - GTL+, GTL (Gunning Tranceiver Logic) - used on Pentium II
    - HSTL - High Speed Transceiver Logic
    - SSTL - Stub Series-Terminate Logic
    - USB - IO used for Universal Serial Bus (differential signaling)
    - AGP - IO used for Advanced Graphics Port
  - Maximum number of IO? Package types?
    - Ball Grid Array (BGA) for high density IO

Altera FPGA Family

- Altera Flex10K/10KE
  - LEs (Logic elements) have 4-input LUTS (look-up tables) +1 FF
  - Fast Carry Chain between LE's, Cascade chain for logic operations
  - Large blocks of SRAM available as well
- Altera Max7000/Max7000A
  - EEPROM based, very fast (Tpd = 7.5 ns)
  - Basically a PLD architecture with programmable interconnect.
  - Max 7000A family is 3.3 v

Altera Flex 10K FPGA Family

Table 1. FLEX 10K Device Features

Feature	EPF10K10 EPF10K10A	EPF10K20	EPF10K30 EPF10K30A	EPF10K40	EPF10K50 EPF10K50V
Typical gates (logic and RAM), *Note (1)*	10,000	20,000	30,000	40,000	50,000
Usable gates	7,000 to 31,000	15,000 to 63,000	22,000 to 69,000	29,000 to 93,000	36,000 to 116,000
Logic elements (LEs)	576	1,152	1,728	2,304	2,880
Logic array blocks (LABs)	72	144	216	288	360
Embedded array blocks (EABs)	3	6	6	8	10
Total RAM bits	6,144	12,288	12,288	16,384	20,480
Maximum user I/O pins	134	189	246	189	310

Altera Flex 10K FPGA Family (cont)

Table 2. FLEX 10K Device Features

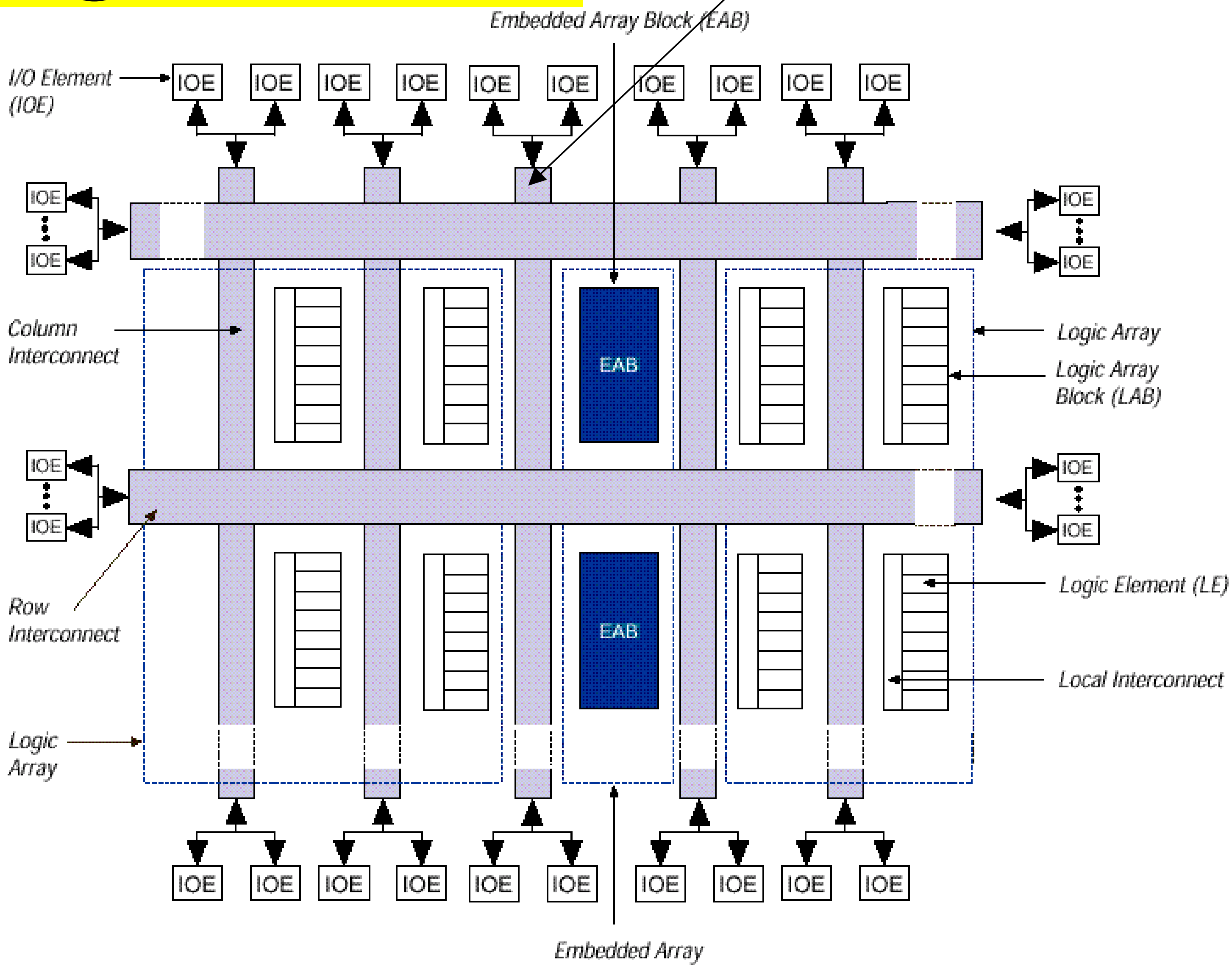
Feature	EPF10K70	EPF10K100 EPF10K100A	EPF10K130V	EPF10K250A
Typical gates (logic and RAM), *Note (1)*	70,000	100,000	130,000	250,000
Usable gates	46,000 to 118,000	62,000 to 158,000	82,000 to 211,000	149,000 to 310,000
LEs	3,744	4,992	6,656	12,160
LABs	468	624	832	1,520
EABs	9	12	16	20
Total RAM bits	18,432	24,576	32,768	40,960
Maximum user I/O pins	358	406	470	470

Note to tables:

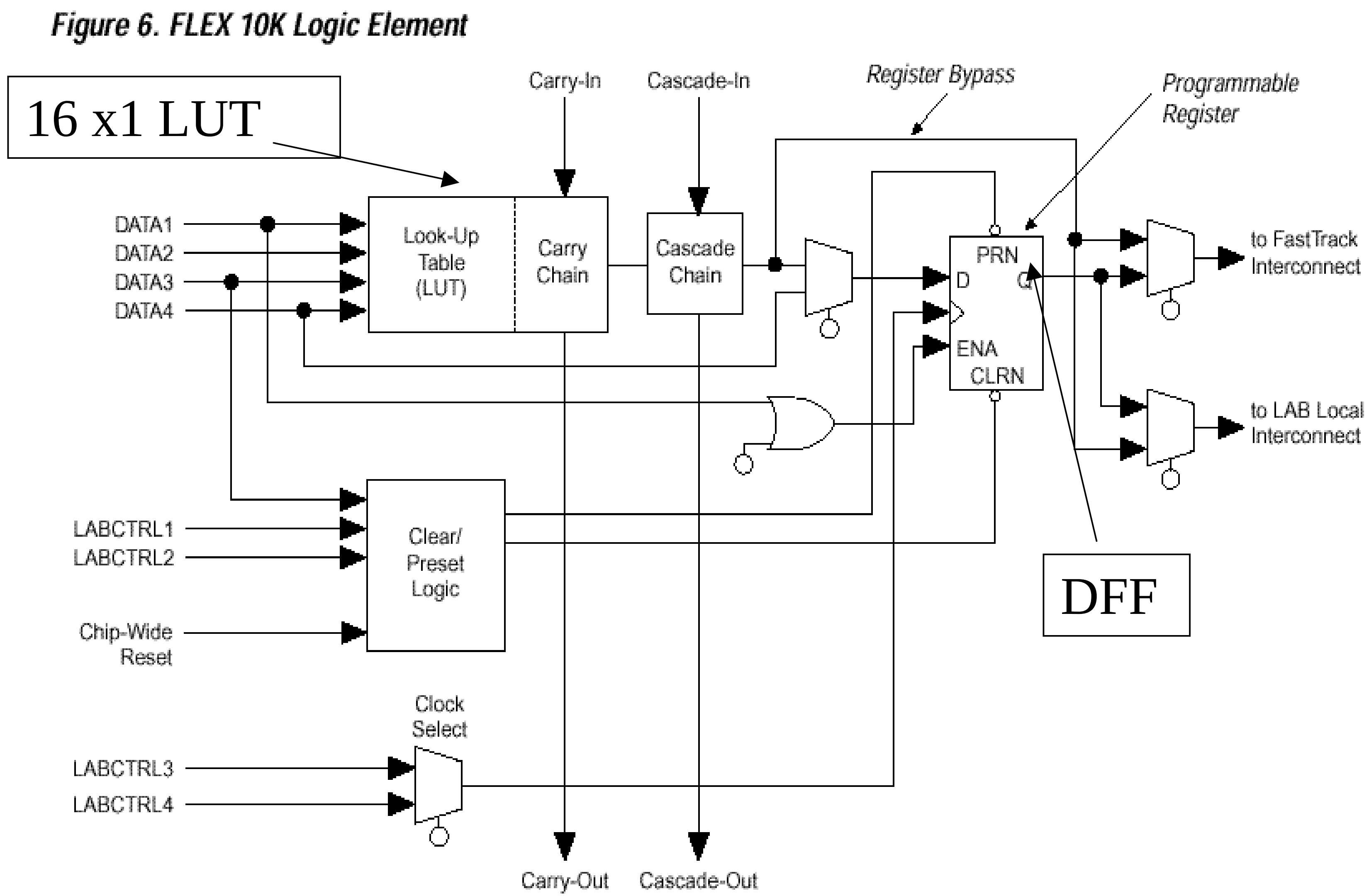
(1) For designs that require JTAG boundary-scan testing, the built-in JTAG circuitry contributes up to 31,250 additional

FLEX 10K Device Block Diagram

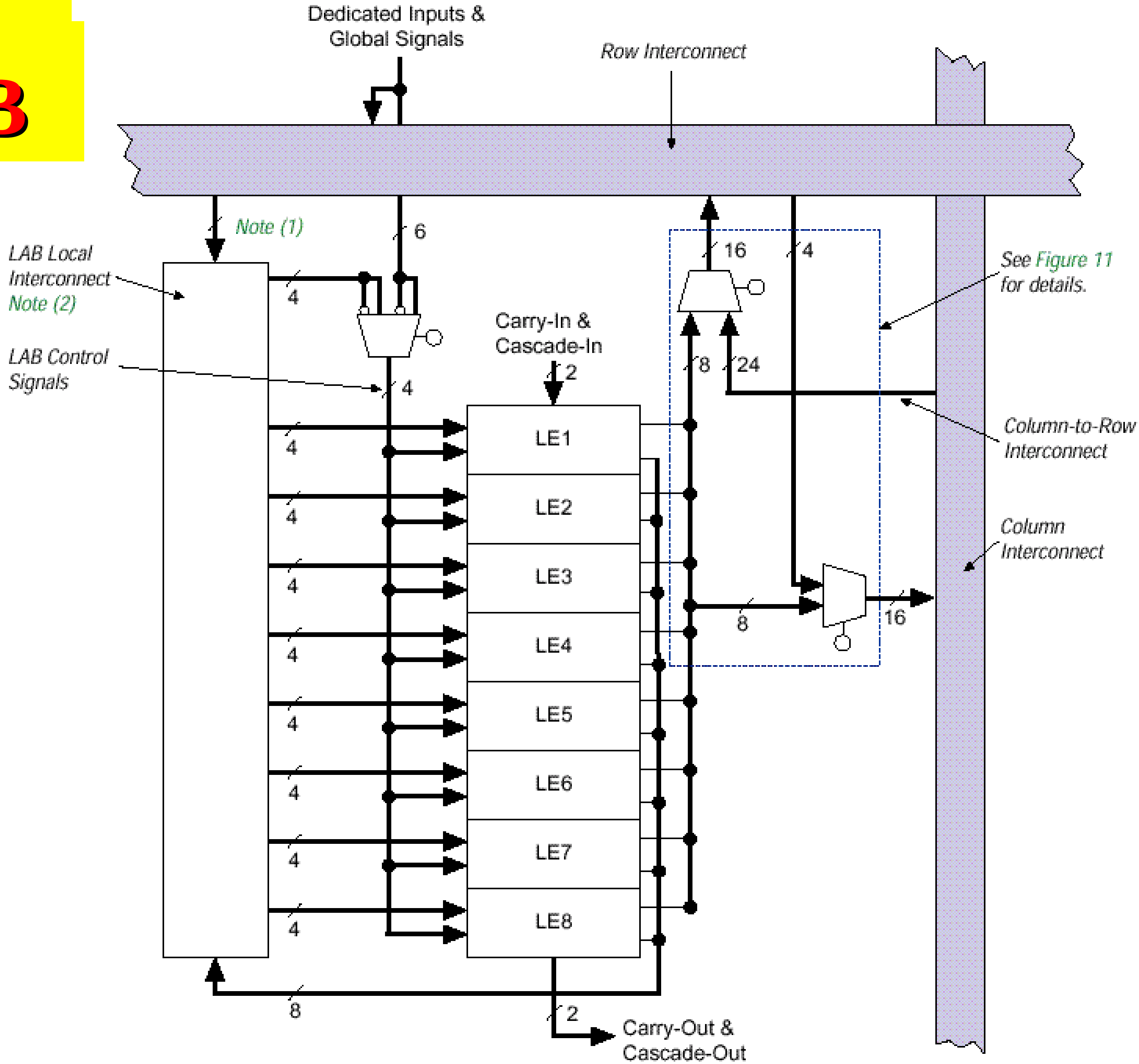
Dedicated memory



FLEX 10K Logic Element



FLEX 10K LAB

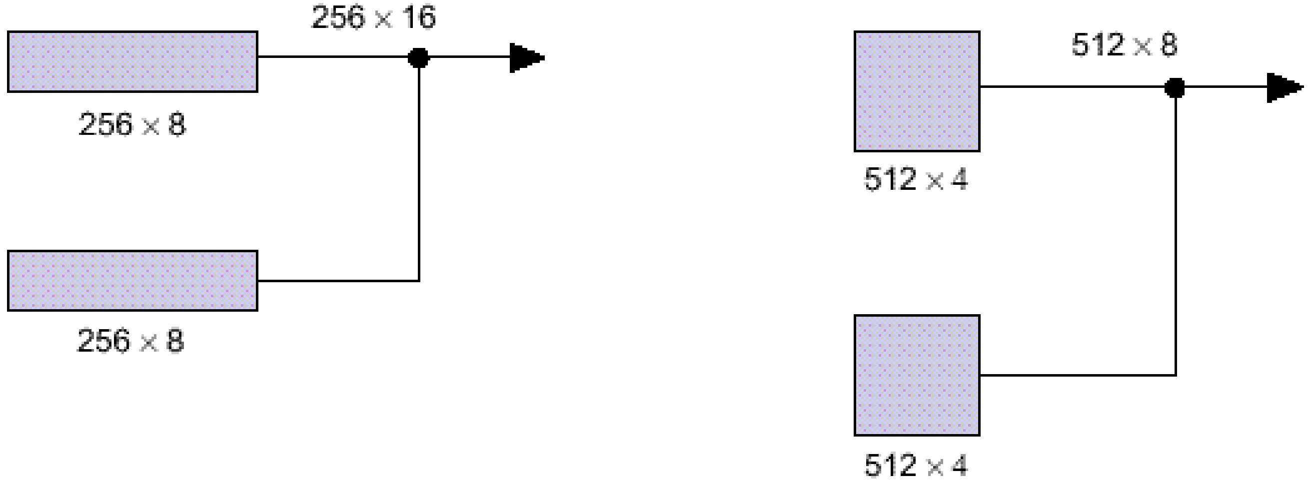


Dedicated Inputs & Global Signals
Row Interconnect
Note (1)
6
LAB Local Interconnect Note (2)
4
LAB Control Signals
4
Carry-In & Cascade-In
2
16
4
See Figure 11 for details.
8
24
Column-to-Row Interconnect
LE1
LE2
LE3
LE4
LE5
LE6
LE7
LE8
Column Interconnect
8
16
8
2
Carry-Out & Cascade-Out

Emedded Array Block

- Memory block, Can be configured:
  - 256 x 8, 512 x 4, 1024 x 2, 2048 x 1

Figure 3. Examples of Combining EABs

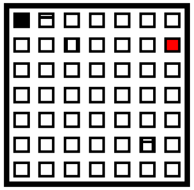


Actel FPGA Family

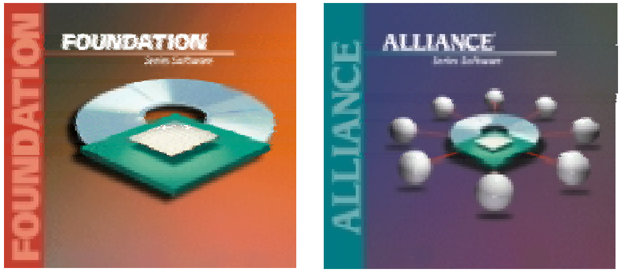
- MXDS Family
  - Fine grain Logic Elements that contain Mux logic + DFF
  - Embedded Dual Port SRAM
  - One Time Programmable (OTP) - means that no configuration loading on powerup, no external serial ROM
  - AntiFuse technology for programming (AntiFuse means that you program the fuse to make the connection).
  - Fast (Tpd = 7.5 ns)
  - Low density compared to Altera, Xilinx - maximum number of gates is 36,000

Who is Xilinx?

- Provides programmable logic solutions



Programmable
Logic Chips



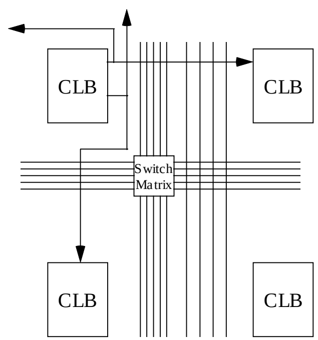
Foundation and Alliance Series
Design Software

- Inventor of the Field Programmable Gate Array
- $900M Annual Revenues; 36+% annual growth

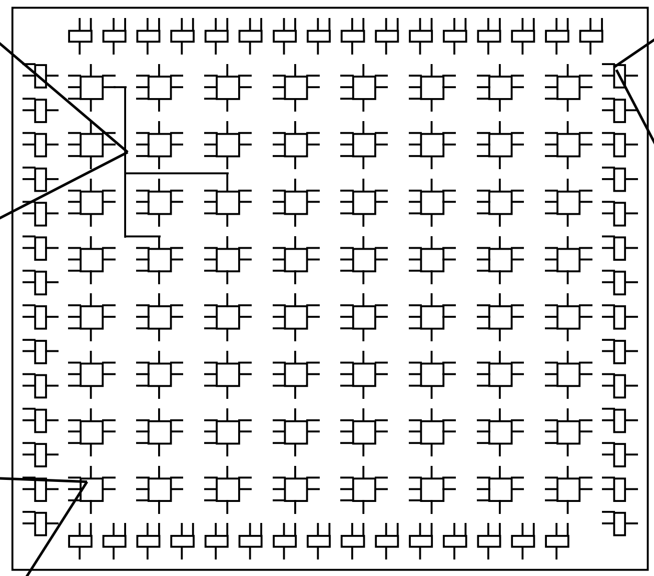
Xilinx FPGA Family

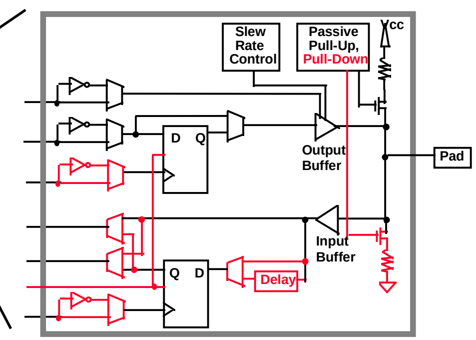
- Virtex Family
  - SRAM Based
  - Largest device has 1M gates
  - Configurable Logic Blocks (CLBs) have two 4-input LUTS, 2 DFFs
  - Four onboard Delay Locked Loops (DLLs) for clock synchronization
  - Dedicated RAM blocks (LUTs can also function as RAM).
  - Fast Carry Logic
- XC4000 Family
  - Previous version of Virtex
  - No DLLs, No dedicated RAM blocks

XC4000 Architecture

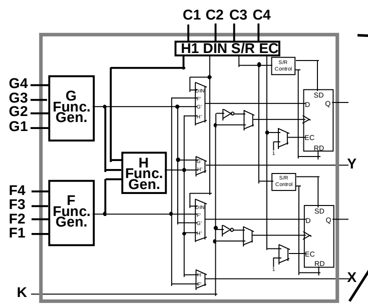


Programmable Interconnect





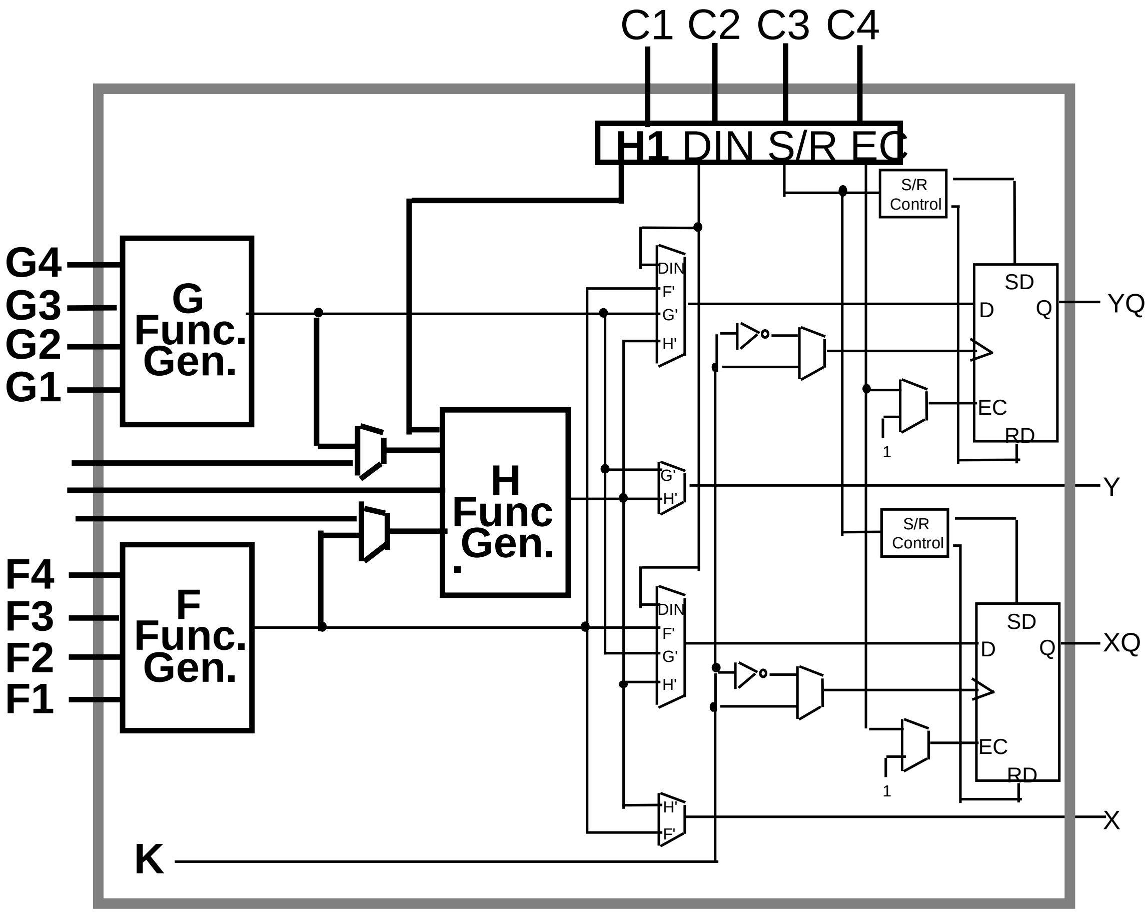
I/O Blocks (IOBs)



Configurable Logic Blocks (CLBs)

XC4000E/X Configurable Logic Blocks

- 2 Four-input function generators (Look Up Tables)
  - 16x1 RAM or Logic function
- 2 Registers
  - Each can be configured as Flip Flop or Latch
  - Independent clock polarity
  - Synchronous and asynchronous Set/Reset



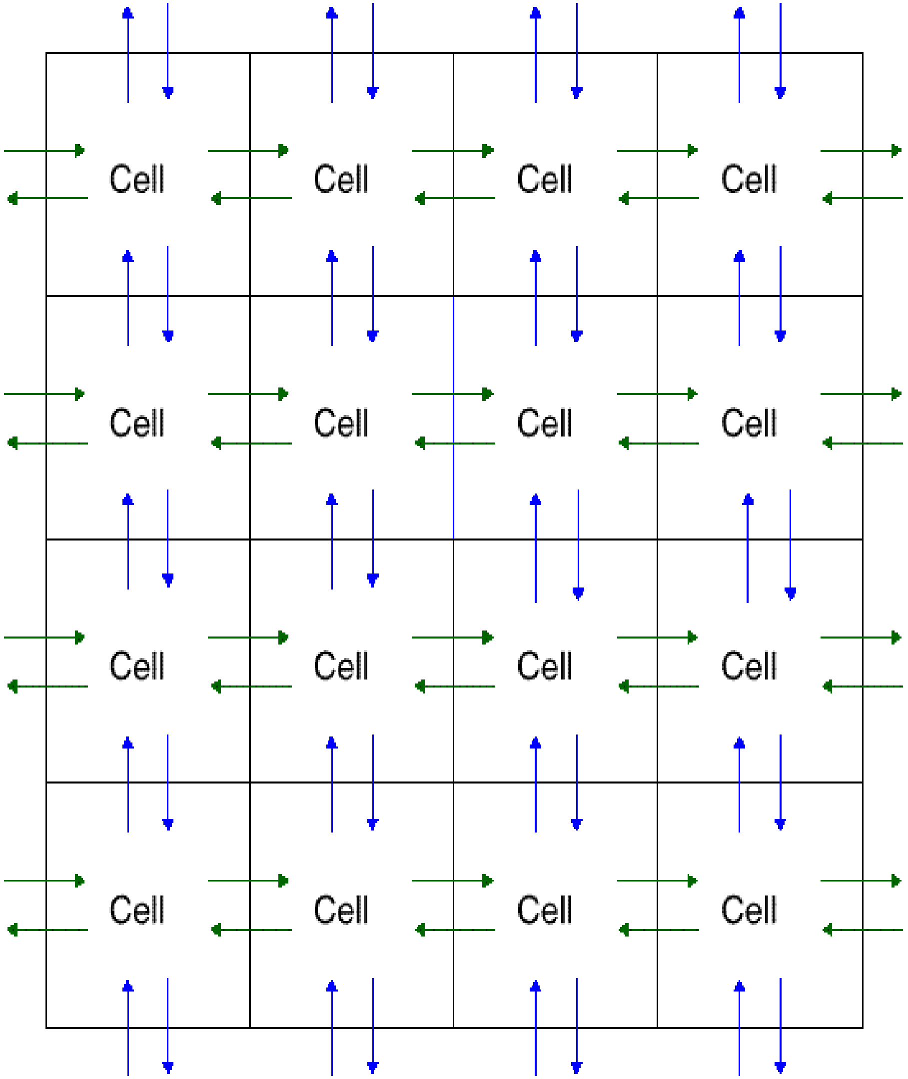
The *Xilinx* *XC6200* and the *H.O.T. Works* *Development* *System*

The Xilinx XC6200 RPU

- SRAM-based FPGA
  - Fast, unlimited reconfiguration
  - Dynamic and partially reconfigurable logic
- Microprocessor interface
- Symmetrical, hierarchical and regular structure

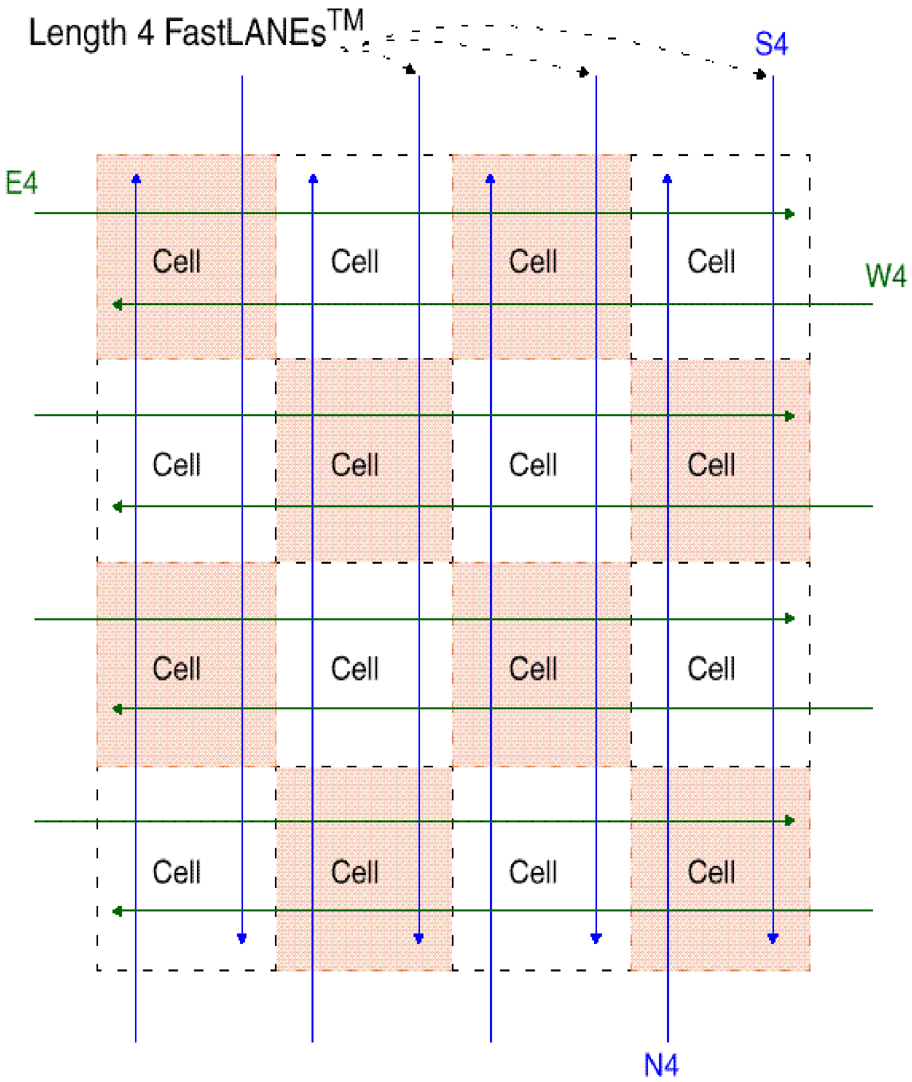
XC6200 Architecture

- Large array of simple, configurable cells (sea of gates)
- Each cell:
  - D-Type register
  - Logic function
  - Nearest-neighbor interconnection
  - Grouped in 4x4 block



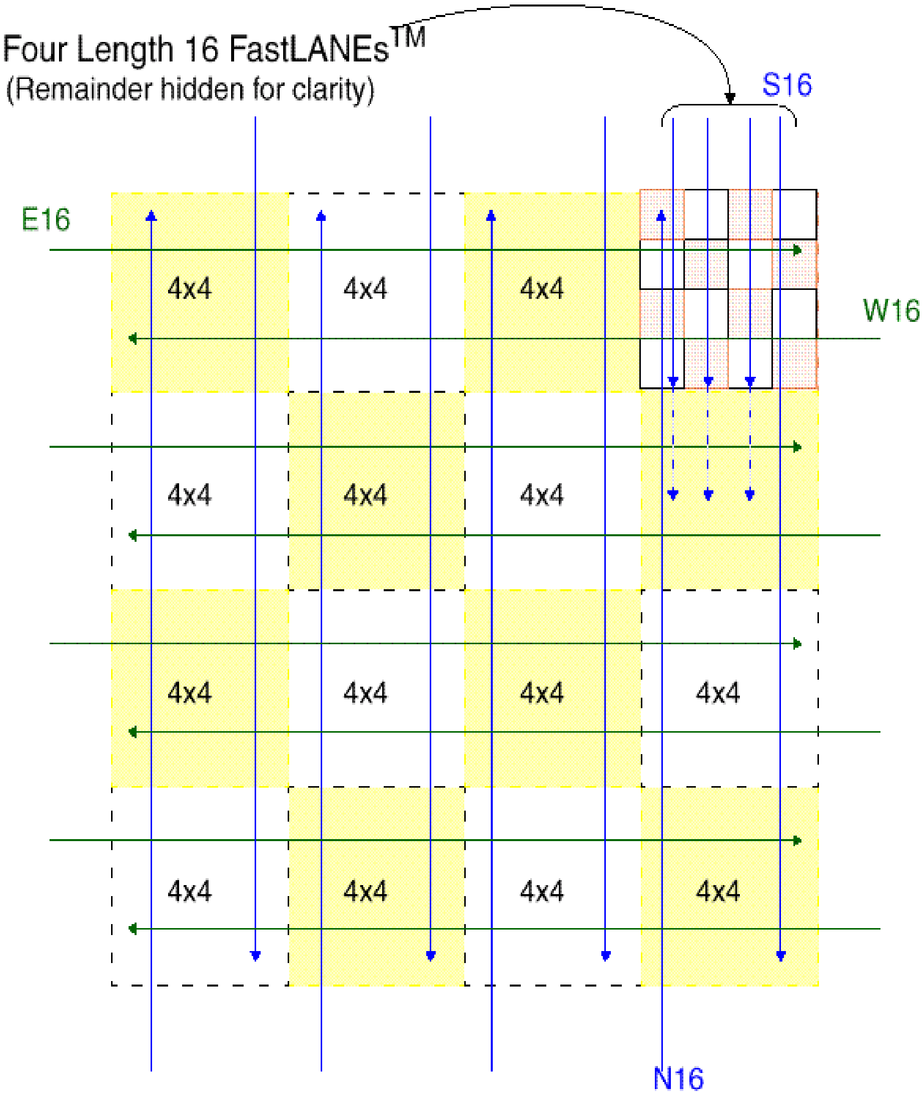
XC6200 Architecture

- 16 (4x4) neighbor-connected cells are grouped together to form a larger cellular array
- Communication “lanes” available between neighboring 4x4 cell blocks



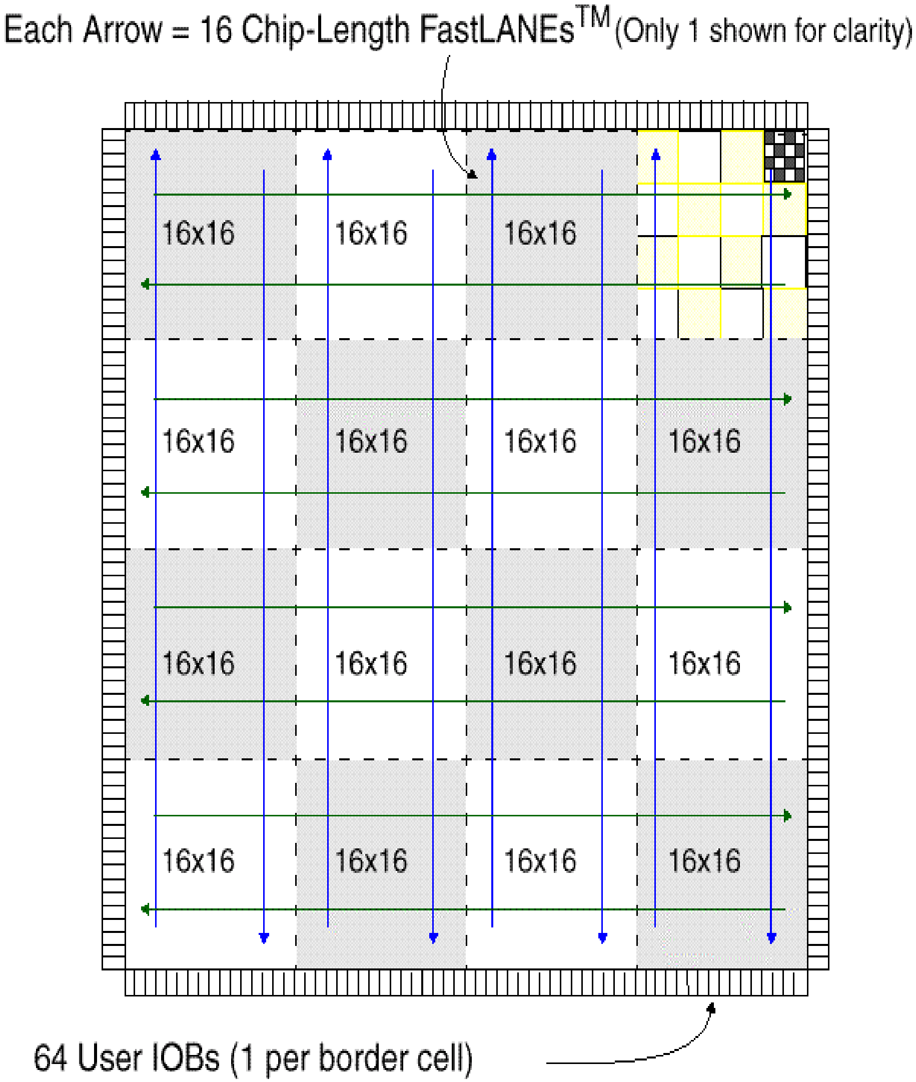
XC6200 Architecture

- A 4x4 array of the previously shown 4x4 blocks forms a 16x16 block
- Length 16 FastLANEs connect these larger arrays



XC6200 Architecture

- A 4x4 array of the 16x16 blocks forms the central 64x64 cell array
- Chip-Length FastLANEs connect
- Central block surrounded by I/O pads

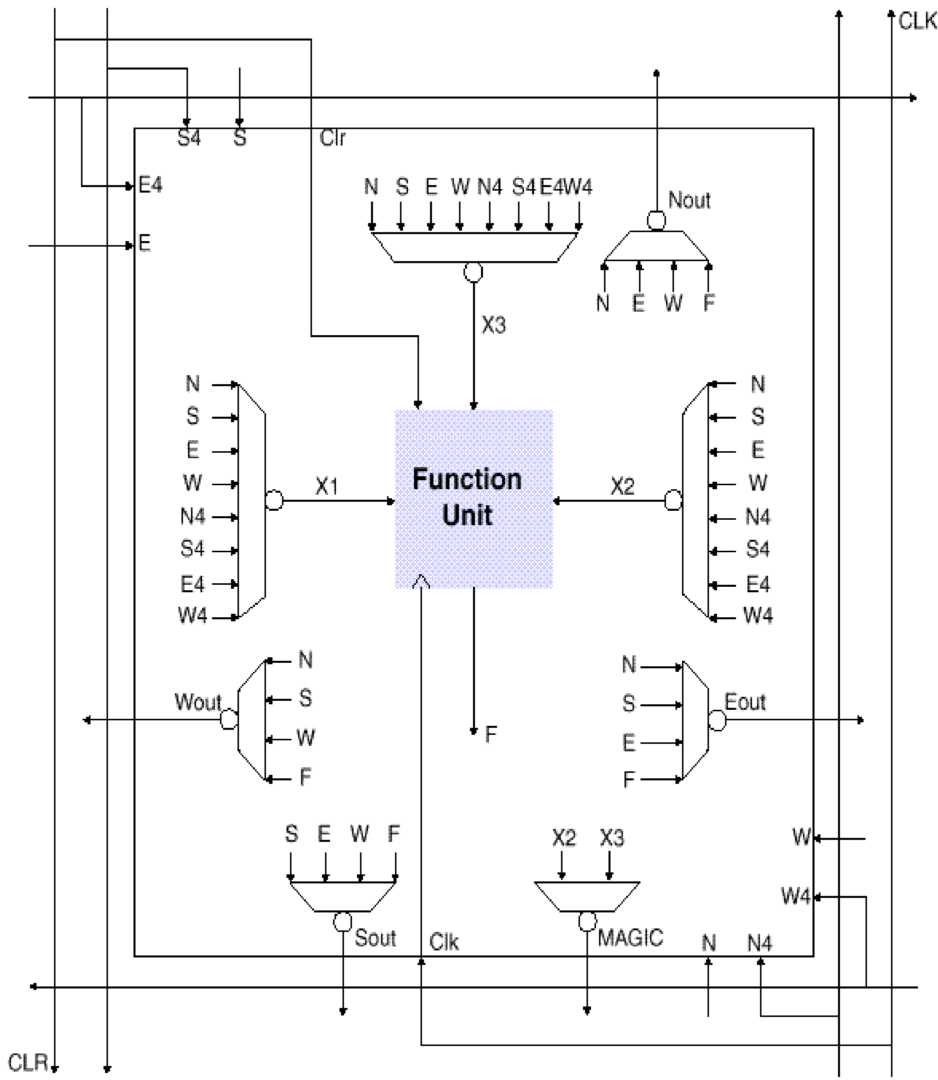


XC6200 Routing

- Each level of hierarchy has its own associated routing resources
  - Unit cells, 4x4, 16x16, 64x64 cell blocks
- Routing does not use a unit cell's resources
- Switches at the edge of the blocks provide for connections between the levels of interconnect

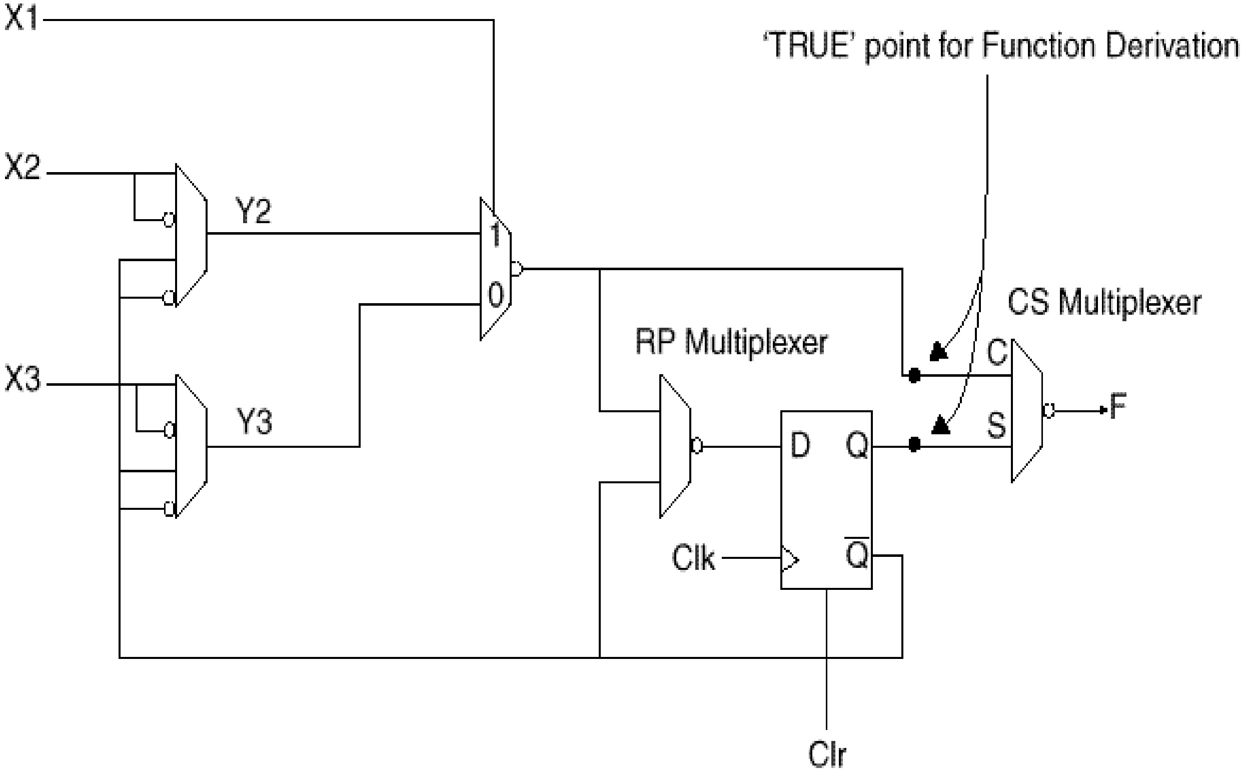
XC6200 Unit Cell

- Each unit cell contains a computation unit:
  - D-type register
  - 2-input logic function
  - Nearest neighbor interconnection
  - Individually programmable from host interface (uP)



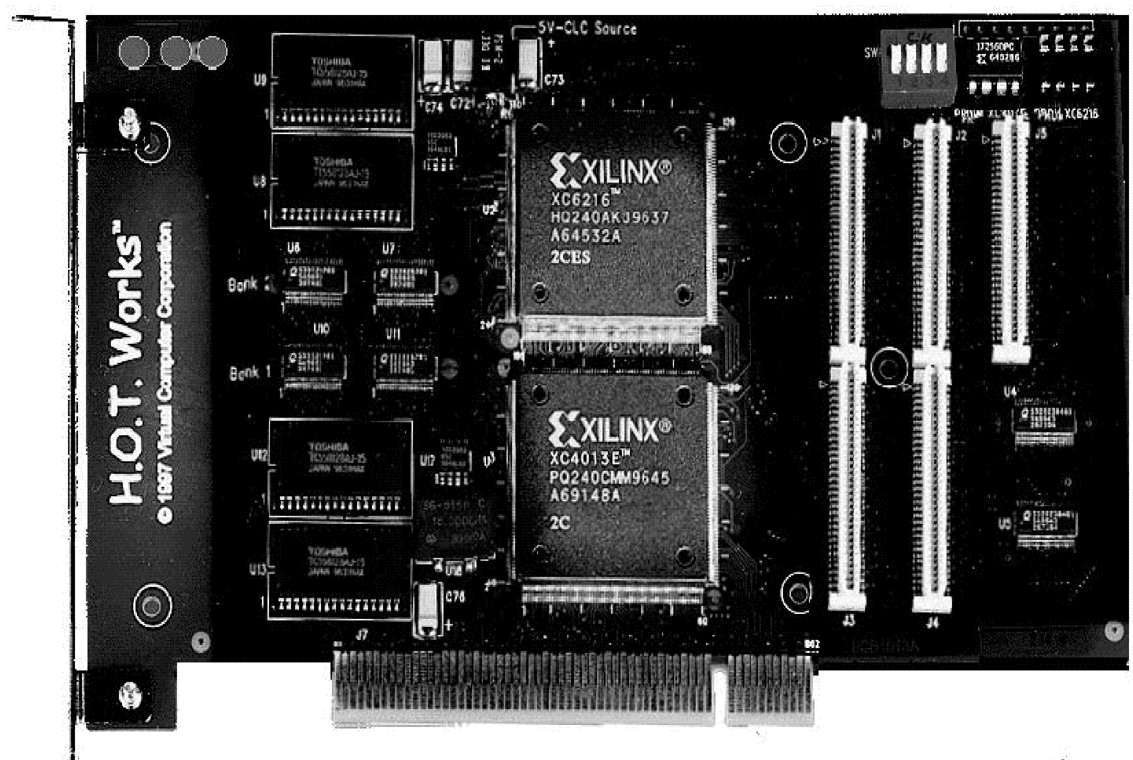
XC6200 Functional Unit

- Design based on the fact that any function of two Boolean variables can be computed by a 2:1 MUX.



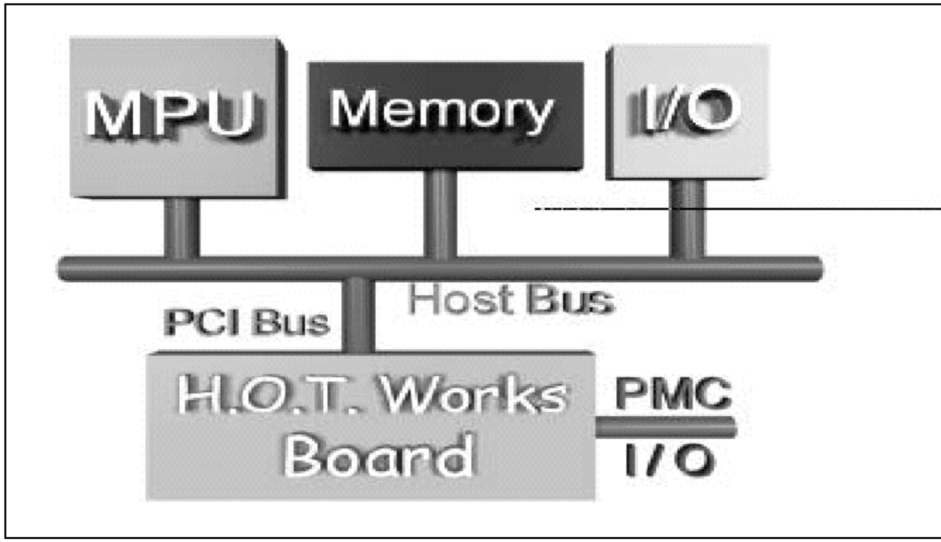
H.O.T. Works

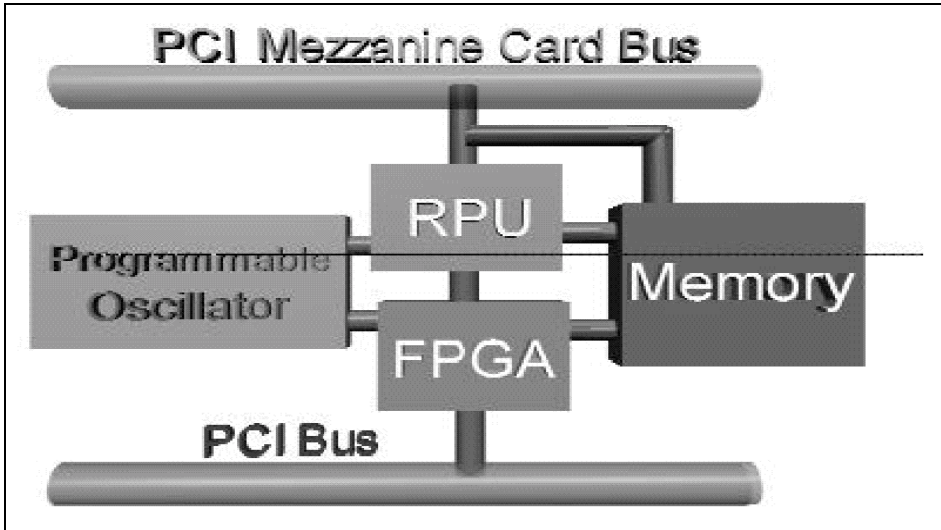
- Development system based on the Xilinx XC6200-series RPU
- Includes:
  - H.O.T. Works Configurable Computer Board
  - H.O.T. Works Development System Software



H.O.T. Works Board

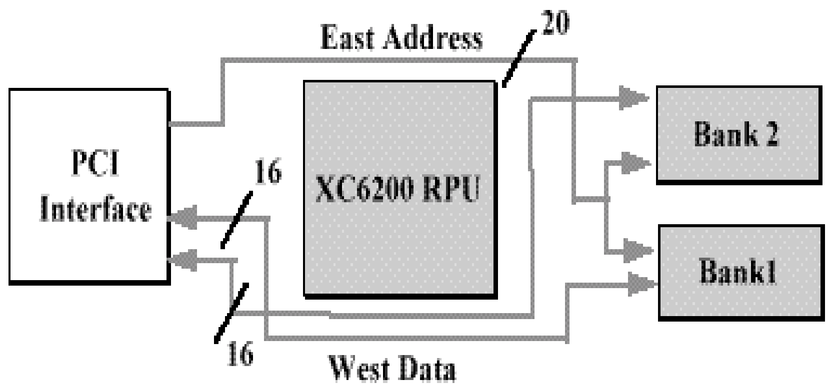
- Interfaces with a host system (Windows95-based PC) on PCI bus
  - 2MB SRAM (memory)
  - XC6200 (RPU)
  - PCI controller on XC4000 (FPGA)
  - Expansion through Mezzanine connector



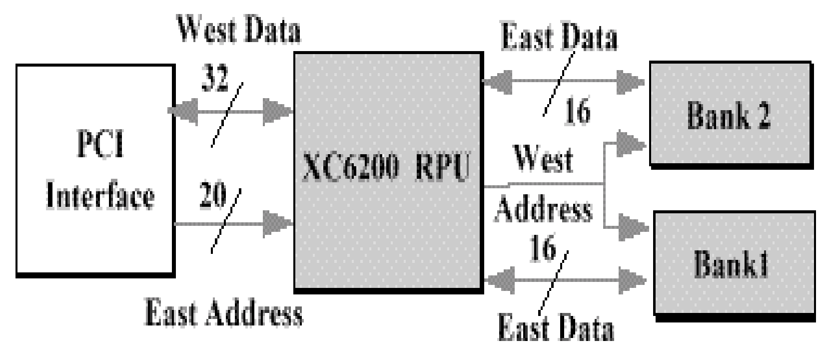


Memory Access Modes

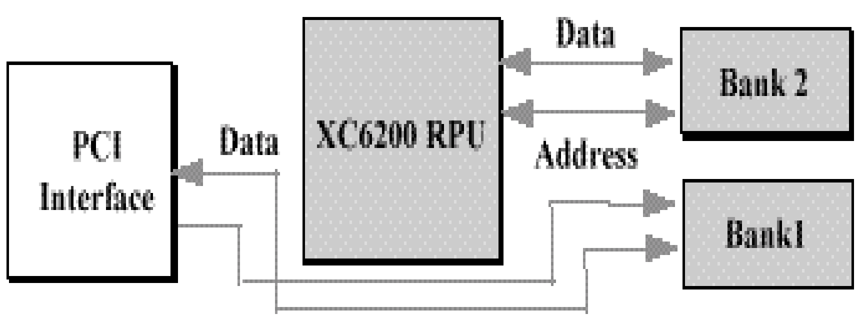
Mode 1 PCI to 32 bit RAM (Boot Default Mode)



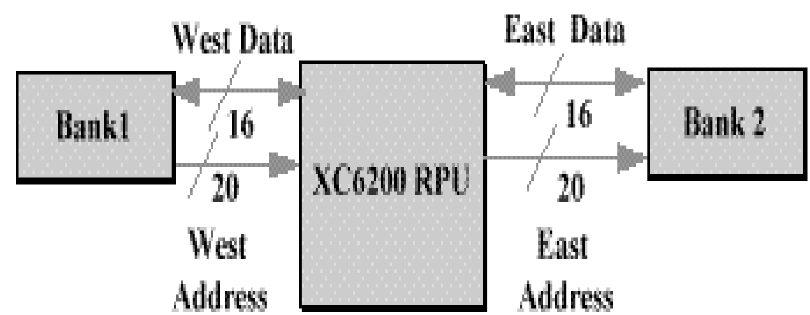
Mode 2 PCI to RPU, RPU to 32 bit RAM



Mode 3a & 3b PCI & RPU to 16 bit RAM



Mode 4 16 bit RAM to RPU to 16 bit RAM

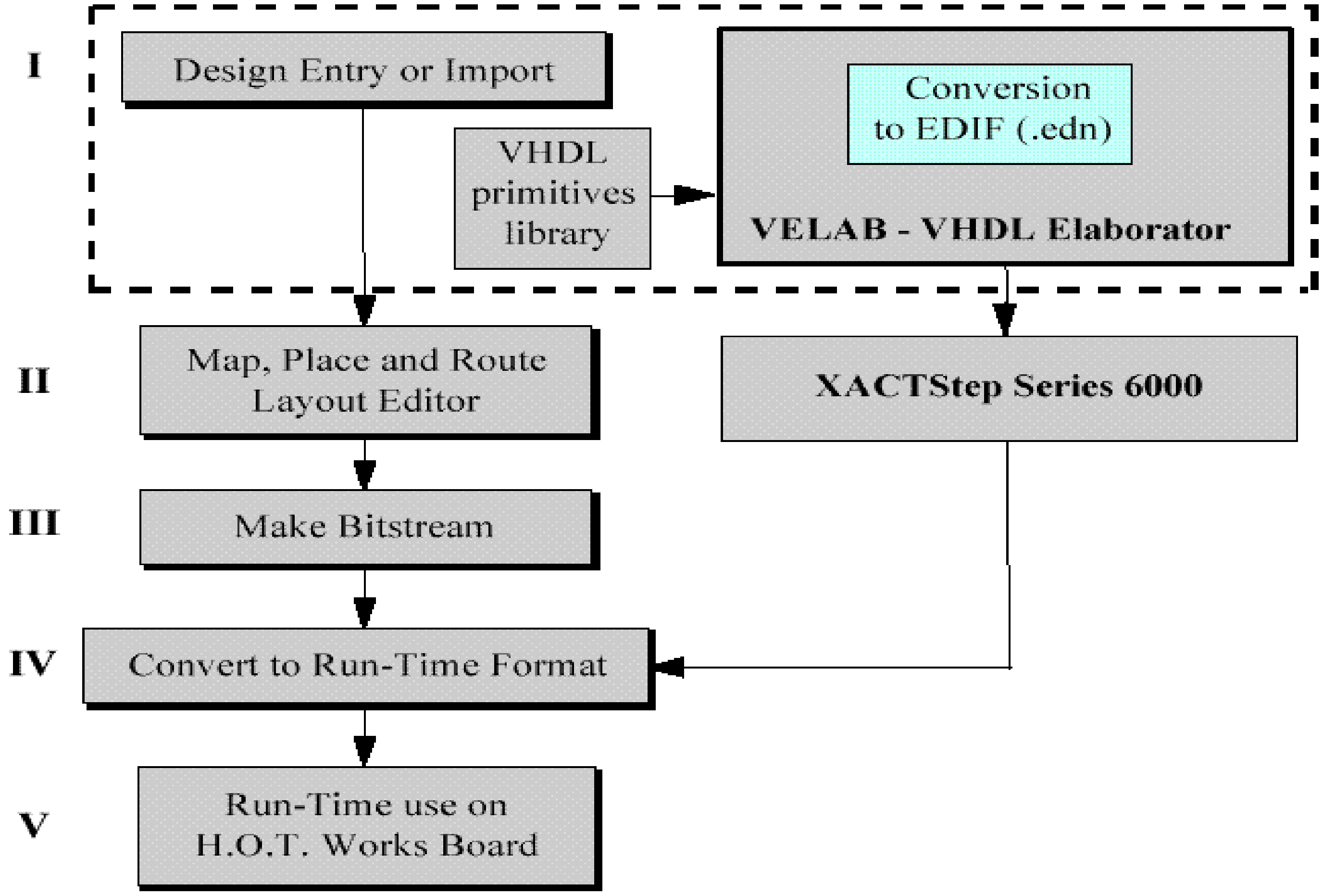


H.O.T. Works Software

- Xilinx XACT*Step*
  – Map, Place and Router for XC6200
- Velab
  – Structural VHDL elaborator
- WebScope
  – Java-based debug tool
- H.O.T. Works Development System
  – C++-based API for board interfacing

Design Flow

Design Flow for VHDL Entry Method



Run-Time Programming

- C++ support software is provided for low-level board interface and device configuration
- Digital design is downloaded to the board at execution time
- User-level routines must be written to conduct data input/output and control

Addendum:

Cell logic function table

Refer to Slide 11

Function	X1	X2	X3	Y2	Y3	RP	CS	Q
0	A	A	A	X2	$\overline{X3}$	X	C	X
1	A	A	A	$\overline{X2}$	X3	X	C	X
BUF (Fast)	A	X	X	Q	$\overline{Q}$	Q	C	0
BUF	X	A	A	$\overline{X2}$	$\overline{X3}$	X	C	X
INV (Fast)	A	X	X	$\overline{Q}$	Q	Q	C	0
INV	X	A	A	X2	X3	X	C	X
A.B (Fast)	A	B	X	$\overline{X2}$	$\overline{Q}$	Q	C	0
A.B	A	B	A	$\overline{X2}$	$\overline{X3}$	X	C	X
$\overline{A}$.B (Fast)	A	X	B	$\overline{Q}$	$\overline{X3}$	Q	C	0
$\overline{A}$.B	A	A	B	X2	$\overline{X3}$	X	C	X
$\overline{A.B}$ (Fast)	A	B	X	X2	Q	Q	C	0
$\overline{A.B}$	A	B	A	X2	X3	X	C	X
A+B (Fast)	A	X	B	Q	$\overline{X3}$	Q	C	0
A+B	A	A	B	$\overline{X2}$	$\overline{X3}$	X	C	X
$\overline{A}$+B (Fast)	A	B	X	$\overline{X2}$	Q	Q	C	0
$\overline{A}$+B	A	B	A	$\overline{X2}$	X3	X	C	X
$\overline{A+B}$ (Fast)	A	X	B	$\overline{Q}$	X3	Q	C	0
$\overline{A+B}$	A	A	B	X2	X3	X	C	X
A⊕B	A	B	B	X2	$\overline{X3}$	X	C	X
$\overline{A \oplus B}$	A	B	B	$\overline{X2}$	X3	X	C	X
M2_1	SEL	A	B	$\overline{X2}$	$\overline{X3}$	X	C	X
M2_1B1A	SEL	A	B	X2	$\overline{X3}$	X	C	X
M2_1B1B	SEL	A	B	$\overline{X2}$	X3	X	C	X
M2_1B2	SEL	A	B	X2	X3	X	C	X

Conclusions on XC5200

- Xilinx XC6200 provides a fast and inexpensive method to obtain great speedups in certain classes of algorithms
- H.O.T. Works provides a useable development platform to go from structural VHDL to digital design, and a programmable run-time interface in C++.

Comparing Technologies - Density (gates per chip)

- Highest to lowest density:
  - Full Custom,
  - Standard Cell,
  - Gate Array,
  - FPGAs,
  - CPLD,
  - PLD
- Full Custom, Standard Cell, Gate Array are called ASIC technologies (Application Specific Integrated Circuit).
- Large Density gap between ASIC technologies and Programmable logic technologies (FPGAs, CPLD, PLD).
- Highest end FPGA density is now equal to low-end ASIC density (i.e., hundreds of thousands of gates with embedded SRAMs).

Comparing Technologies - Speed

- Highest to lowest performance: Full Custom, Standard Cell, Gate Array, PLDs, CPLDs, FPGAs.
- Again, large performance gap between ASIC technologies and programmable technologies.
- Performance of programmable technologies is in reverse order of their densities.

Comparing Technologies - Cost

- Depends heavily on volume.
- If only need a few hundred, then FPGAs can be cheaper.
- If need thousands, then ASIC technologies are cheaper.
- NRE cost (non-recurring engineering costs) are higher for ASIC techologies than FPGAs
- Per-unit-cost (chip cost) higher for FPGAs

Summary

- Full custom can give best density and performance
- Faster design time and ease of design are principle advantages of gate array and standard cell over full custom.
- Fast fabrication time and lower cost are principle advantages of gate arrays over standard cell.
- Gate arrays offer much higher density over FPGAs and are cheaper than FPGAs in volume production.

Summary (cont.)

- FPGAs principle advantage over gate arrays is 'instant' fabrication time (programmed on desktop). FPGAs are also cheaper than gate arrays in low volume.
- Densities are reaching 100's of thousands of gates/chip.
- Can be used to prototype full custom/standard cell designs.
- PLDs still hold a speed advantage over most FPGAs are useful primarily for high speed decoding and speed critical glue logic.

Sources

- Bob Reese
- **Rob Yates**
- Sheffield Hallam University
- Mark L. Chang <mchang@ece.nwu.edu>