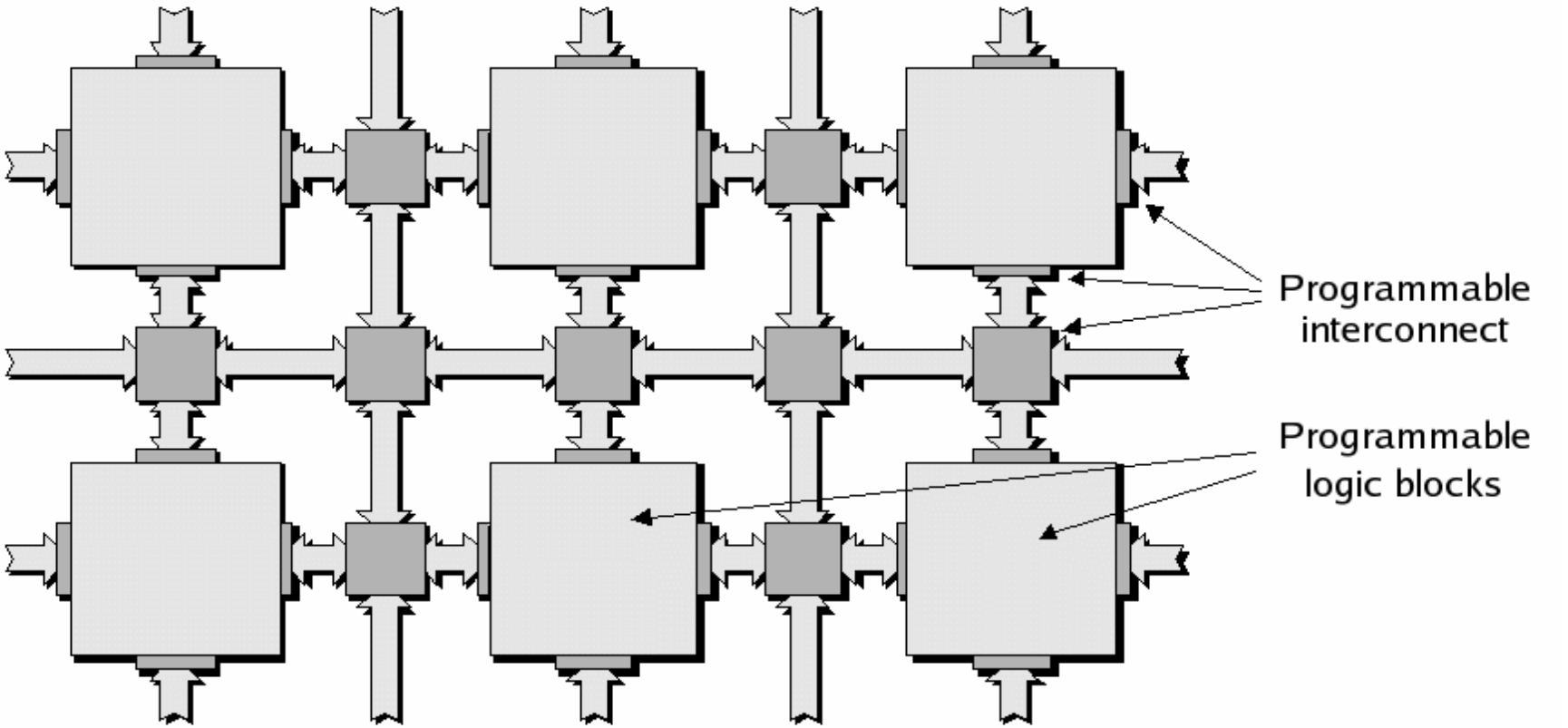
# FPGAs 1

CMPE691/491: Advanced FPGA Design

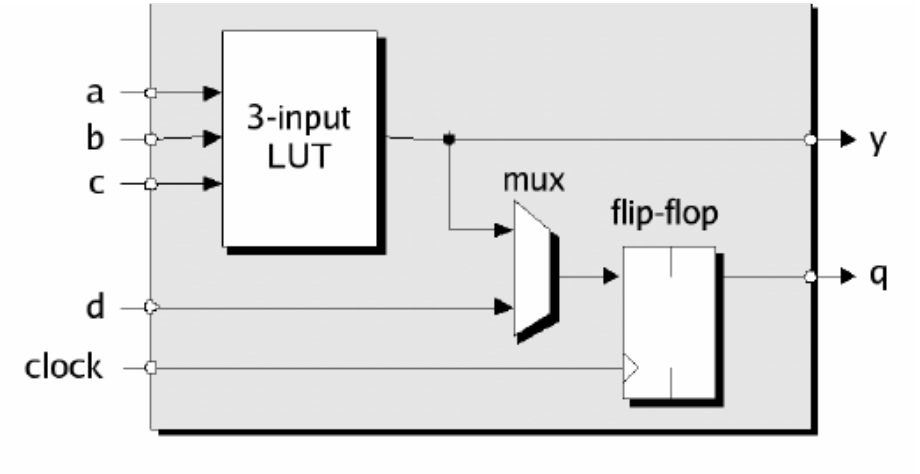
# FPGAs

- Large array of configurable logic blocks (CLB) connected via programmable interconnects




The Design Warrior's Guide to FPGAs,
ISBN 0750676043,
Copyright(C) 2004 Mentor Graphics Corp

# Features and Specifications of FPGAs




The Design Warrior's Guide to FPGAs,
ISBN 0750676043,
Copyright(C) 2004 Mentor Graphics Corp

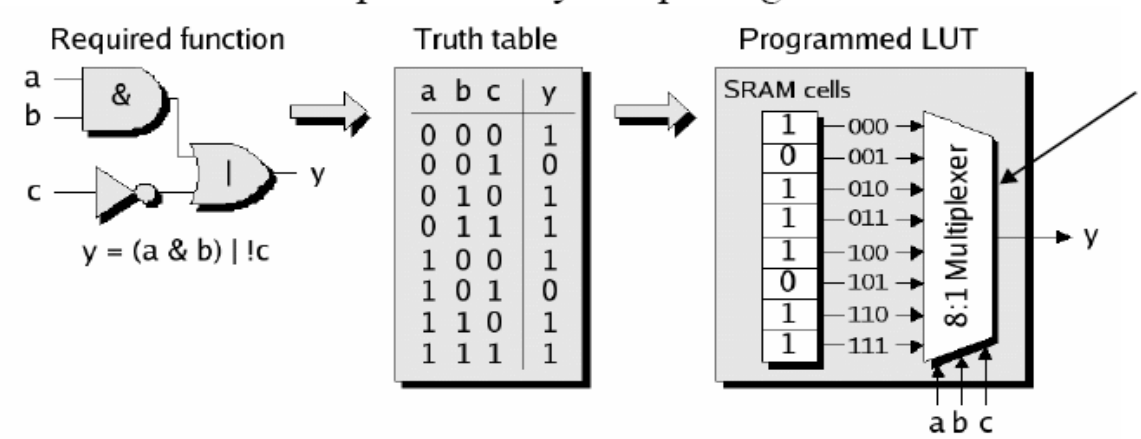
Each PLB can be programmed individually to perform a unique function.

The FF can be triggered by a positive or negative-going clk.

The MUX allows selection of the LUT output or an external input.

# Basic Programmable Devices

The LUT can implement any 3-input logic function.



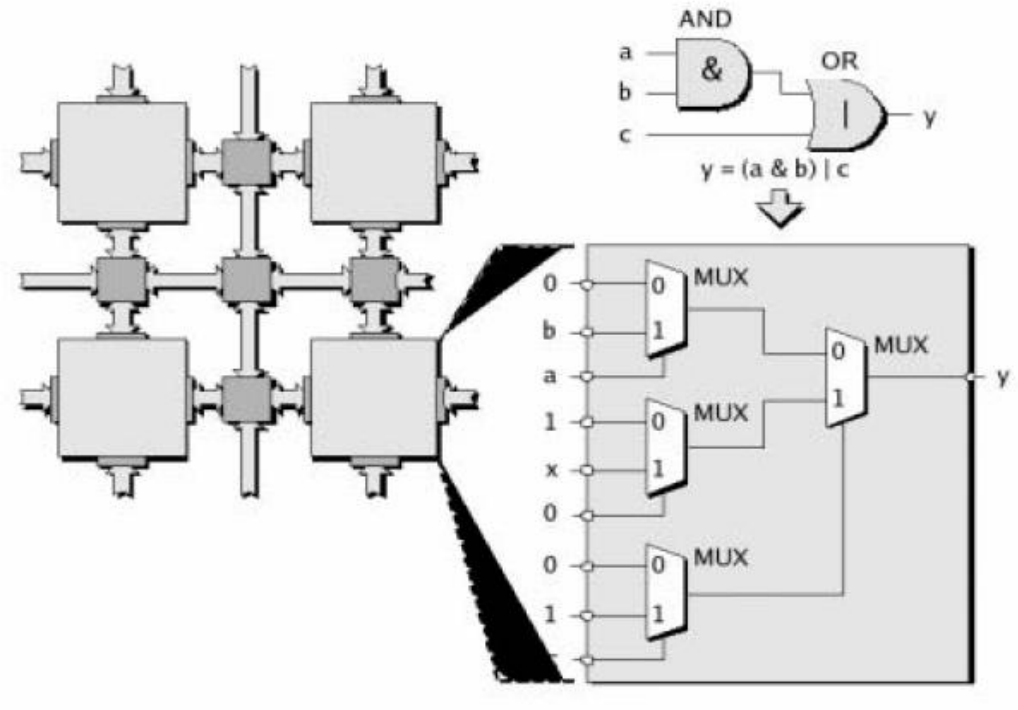

(Actual implementation does not use a MUX here -- more on this later).

# Features and Specifications of FPGAs

**MUX- vs. LUT-based Logic Blocks**

There are 2 basic flavors of PLBs for *medium-grained* architectures, *multiplexer* (MUX) and *lookup table* (LUT).

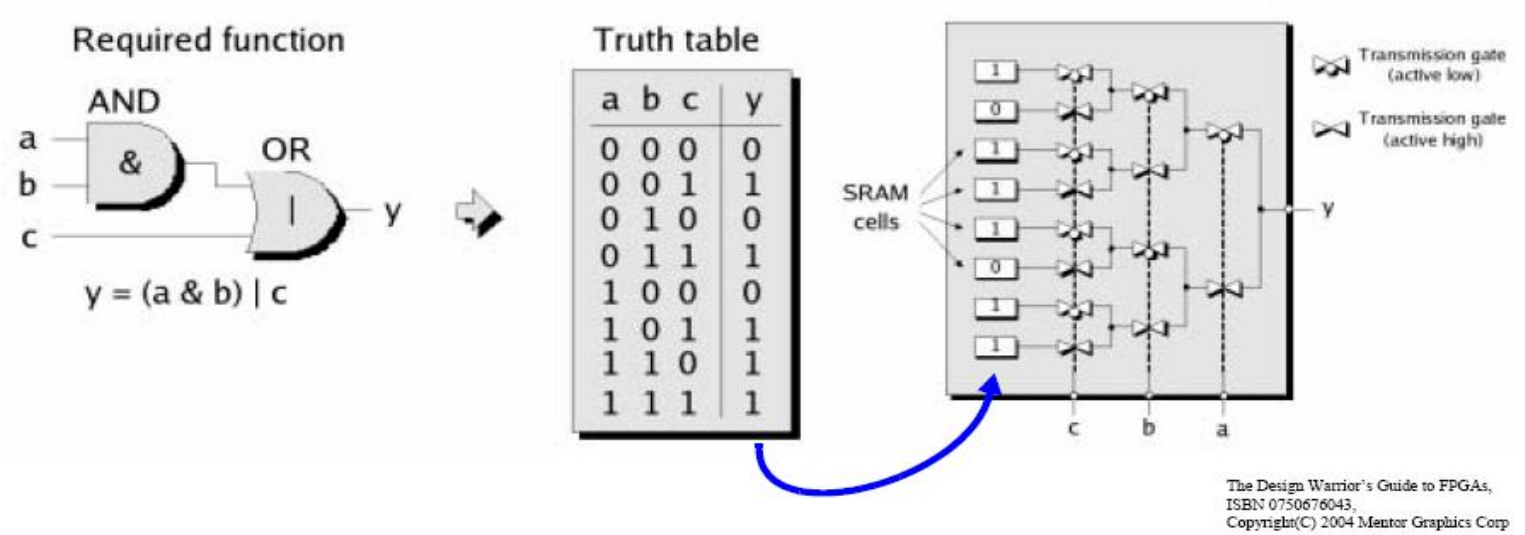
In the MUX-based version, each input can be programmed with a logic 0, 1, or the true or inverted version of a variable.



# Features and Specifications of FPGAs

**MUX- vs. LUT-based Logic Blocks**

Most FPGAs today are LUT-based -- here, the input signals are used as a pointer into a lookup table.




Input signals can be decoded using a hierarchy of *transmission-gate* MUXs.

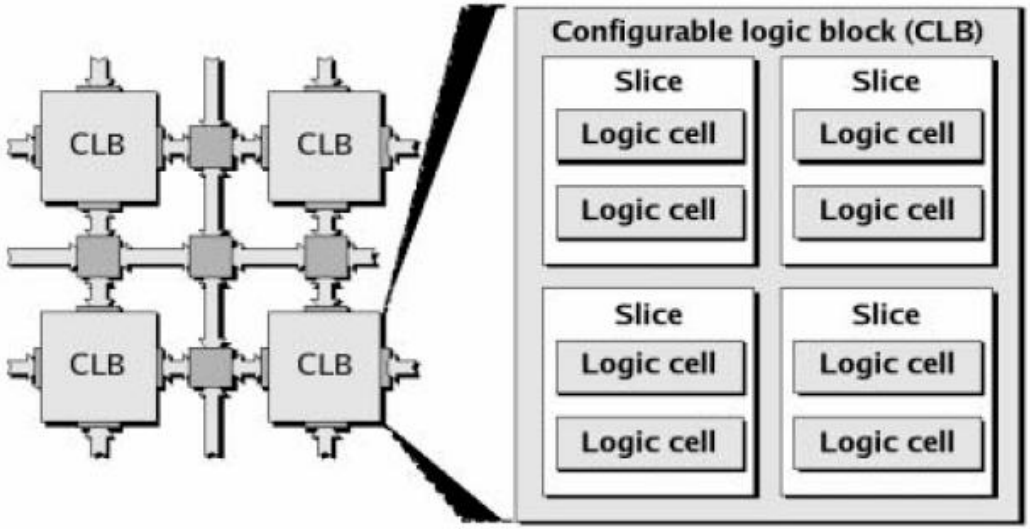
Transmission gates *pass* the value on their inputs or are **high-impedance**.

Note that the diagram does not show the serial connection of the cells (scan chain) for simplicity.

# Features and Specifications of FPGAs

**Terminology and Hierarchy**

The CLB also has some fast interconnect (not shown), that is used to connect neighboring slices.



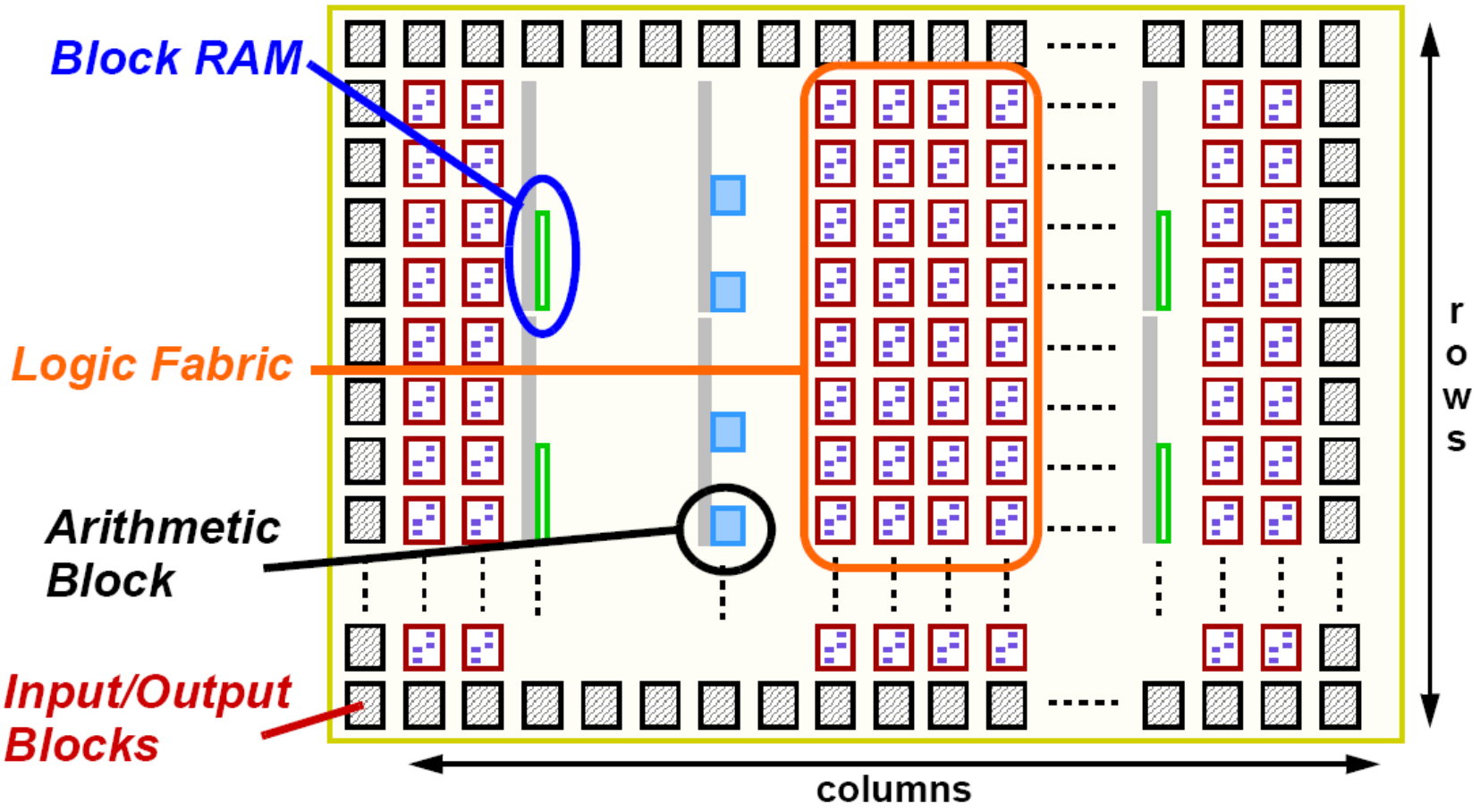



The organization of *LC -> Slice -> CLB* is complemented by an equivalent hierarchy in the interconnect.

That is, fast interconnect between LCs in a slice, slightly slower between slices in a CLB, followed by the interconnect between CLBs.

# Generic Xilinx FPGA Architecture

- FPGAs provide a highly parallel and flexible implementation platform for DSP... this diagram is representative of Xilinx DSP series devices.



# Features and Specifications of FPGAs

**Embedded Processor Cores (Hard and Soft)**

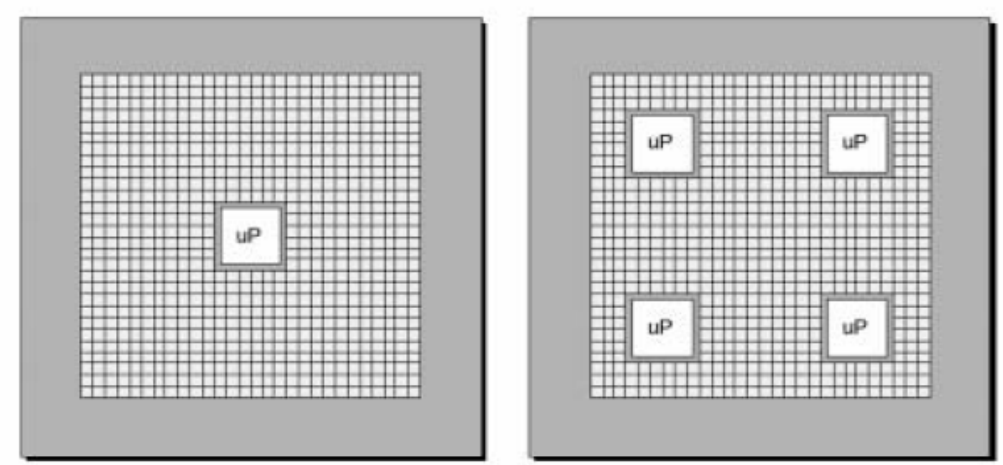
- In a strip (called the Stripe) to the side of the FPGA fabric (cont).

  Advantages:

  The main FPGA fabric is identical for chips with and without the microprocessor.

  The FPGA vendor can bundle other features in the *stripe*, such as memory, special peripherals, etc.

- Embed directly into the FPGA fabric




(a) One embedded core (b) Four embedded cores

# Virtex FPGA family name

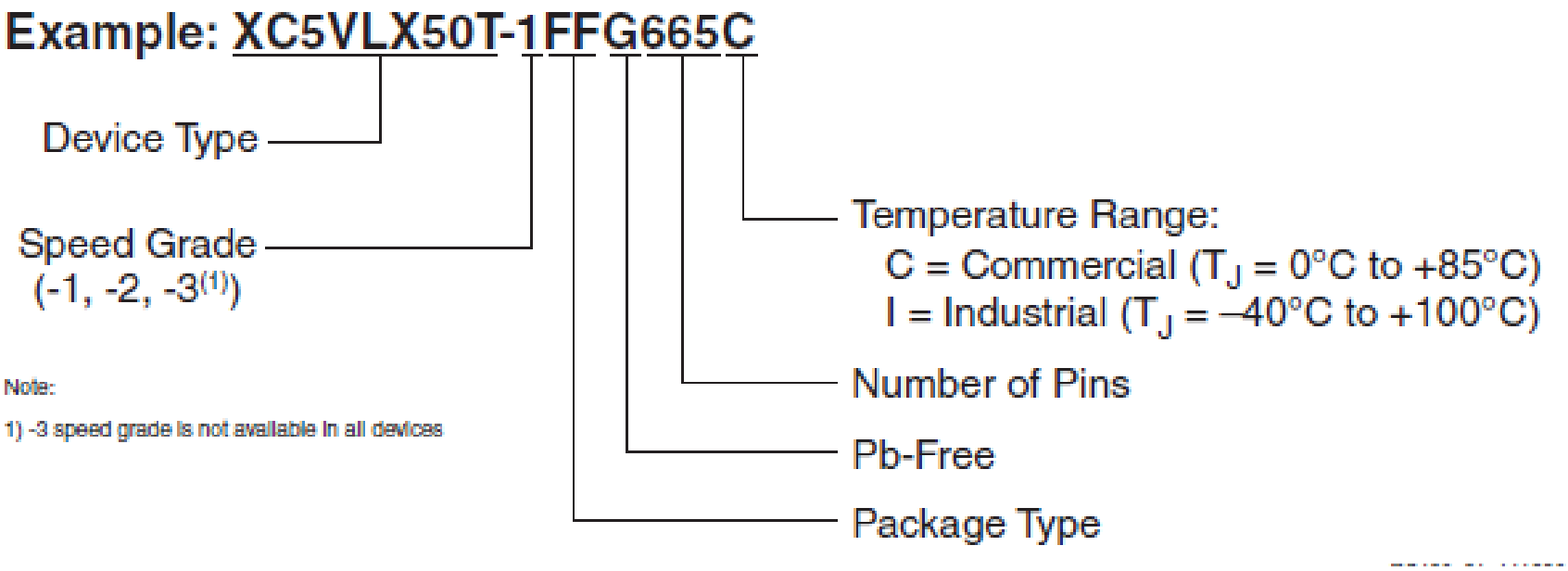


*Table 1:* **Virtex-5 FPGA Family Members**

| Device | Configurable Logic Blocks (CLBs) | | | DSP48E Slices[2] | Block RAM Blocks | | | CMTs[4] | PowerPC Processor Blocks | Endpoint Blocks for PCI Express | Ethernet MACs[5] | Max RocketIO Transceivers[6] | | Total I/O Banks[8] | Max User I/O[7] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Array (Row x Col) | Virtex-5 Slices[1] | Max Distributed RAM (Kb) | | 18 Kb[3] | 36 Kb | Max (Kb) | | | | | GTP | GTX | | |
| XC5VLX30 | 80 x 30 | 4,800 | 320 | 32 | 64 | 32 | 1,152 | 2 | N/A | N/A | N/A | N/A | N/A | 13 | 400 |
| XC5VLX50 | 120 x 30 | 7,200 | 480 | 48 | 96 | 48 | 1,728 | 6 | N/A | N/A | N/A | N/A | N/A | 17 | 560 |
| XC5VLX85 | 120 x 54 | 12,960 | 840 | 48 | 192 | 96 | 3,456 | 6 | N/A | N/A | N/A | N/A | N/A | 17 | 560 |
| XC5VLX110 | 160 x 54 | 17,280 | 1,120 | 64 | 256 | 128 | 4,608 | 6 | N/A | N/A | N/A | N/A | N/A | 23 | 800 |
| XC5VLX155 | 160 x 76 | 24,320 | 1,640 | 128 | 384 | 192 | 6,912 | 6 | N/A | N/A | N/A | N/A | N/A | 23 | 800 |
| XC5VLX220 | 160 x 108 | 34,560 | 2,280 | 128 | 384 | 192 | 6,912 | 6 | N/A | N/A | N/A | N/A | N/A | 23 | 800 |
| XC5VLX330 | 240 x 108 | 51,840 | 3,420 | 192 | 576 | 288 | 10,368 | 6 | N/A | N/A | N/A | N/A | N/A | 33 | 1,200 |
| XC5VLX20T | 60 x 26 | 3,120 | 210 | 24 | 52 | 26 | 936 | 1 | N/A | 1 | 2 | 4 | N/A | 7 | 172 |
| XC5VLX30T | 80 x 30 | 4,800 | 320 | 32 | 72 | 36 | 1,296 | 2 | N/A | 1 | 4 | 8 | N/A | 12 | 360 |
| XC5VLX50T | 120 x 30 | 7,200 | 480 | 48 | 120 | 60 | 2,160 | 6 | N/A | 1 | 4 | 12 | N/A | 15 | 480 |
| XC5VLX85T | 120 x 54 | 12,960 | 840 | 48 | 216 | 108 | 3,888 | 6 | N/A | 1 | 4 | 12 | N/A | 15 | 480 |
| XC5VLX110T | 160 x 54 | 17,280 | 1,120 | 64 | 296 | 148 | 5,328 | 6 | N/A | 1 | 4 | 16 | N/A | 20 | 680 |
| XC5VLX155T | 160 x 76 | 24,320 | 1,640 | 128 | 424 | 212 | 7,632 | 6 | N/A | 1 | 4 | 16 | N/A | 20 | 680 |
| XC5VLX220T | 160 x 108 | 34,560 | 2,280 | 128 | 424 | 212 | 7,632 | 6 | N/A | 1 | 4 | 16 | N/A | 20 | 680 |
| XC5VLX330T | 240 x 108 | 51,840 | 3,420 | 192 | 648 | 324 | 11,664 | 6 | N/A | 1 | 4 | 24 | N/A | 27 | 960 |
| XC5VSX35T | 80 x 34 | 5,440 | 520 | 192 | 168 | 84 | 3,024 | 2 | N/A | 1 | 4 | 8 | N/A | 12 | 360 |
| XC5VSX50T | 120 x 34 | 8,160 | 780 | 288 | 264 | 132 | 4,752 | 6 | N/A | 1 | 4 | 12 | N/A | 15 | 480 |
| XC5VSX95T | 160 x 46 | 14,720 | 1,520 | 640 | 488 | 244 | 8,784 | 6 | N/A | 1 | 4 | 16 | N/A | 19 | 640 |
| XC5VSX240T | 240 x 78 | 37,440 | 4,200 | 1,056 | 1,032 | 516 | 18,576 | 6 | N/A | 1 | 4 | 24 | N/A | 27 | 960 |
| XC5VTX150T | 200 x 58 | 23,200 | 1,500 | 80 | 456 | 228 | 8,208 | 6 | N/A | 1 | 4 | N/A | 40 | 20 | 680 |
| XC5VTX240T | 240 x 78 | 37,440 | 2,400 | 96 | 648 | 324 | 11,664 | 6 | N/A | 1 | 4 | N/A | 48 | 20 | 680 |
| XC5VFX30T | 80 x 38 | 5,120 | 380 | 64 | 136 | 68 | 2,448 | 2 | 1 | 1 | 4 | N/A | 8 | 12 | 360 |
| XC5VFX70T | 160 x 38 | 11,200 | 820 | 128 | 296 | 148 | 5,328 | 6 | 1 | 3 | 4 | N/A | 16 | 19 | 640 |
| XC5VFX100T | 160 x 56 | 16,000 | 1,240 | 256 | 456 | 228 | 8,208 | 6 | 2 | 3 | 4 | N/A | 16 | 20 | 680 |
| XC5VFX130T | 200 x 56 | 20,480 | 1,580 | 320 | 596 | 298 | 10,728 | 6 | 2 | 3 | 6 | N/A | 20 | 24 | 840 |
| XC5VFX200T | 240 x 68 | 30,720 | 2,280 | 384 | 912 | 456 | 16,416 | 6 | 2 | 4 | 8 | N/A | 24 | 27 | 960 |

# FPGA vs ASIC

UMBC

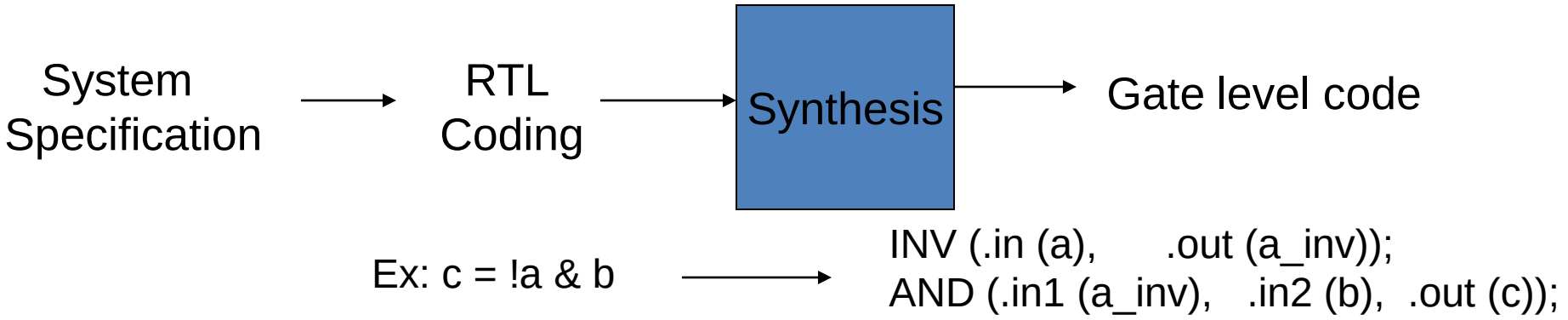
# Standard cell based IC vs. Custom design IC

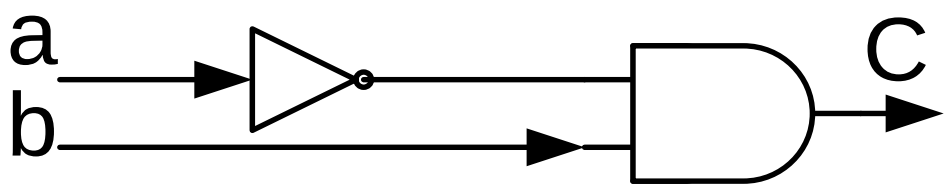
- Standard cell based IC:
  - Design using standard cells
  - Standard cells come from library provider
  - Many different choices for cell size, delay, leakage power
  - Many EDA tools to automate this flow
  - Shorter design time
- Custom design IC:
  - Design all by yourself
  - Higher performance

# Standard cell based VLSI design flow

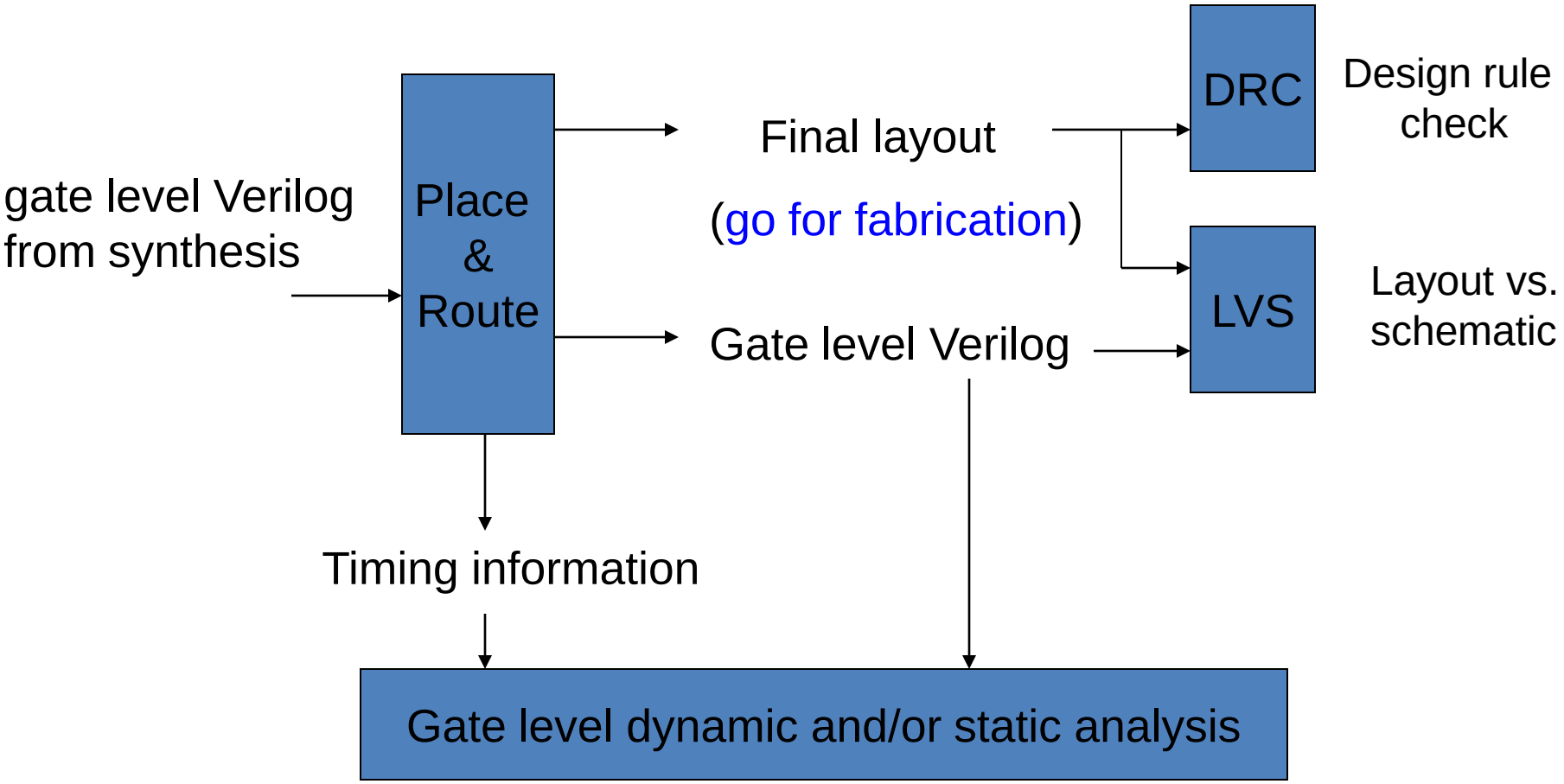
- Front end
  - System specification and architecture
  - HDL coding & behavioral simulation
  - Synthesis & gate level simulation
- Back end
  - Placement and routing
  - DRC (Design Rule Check), LVS (Layout vs Schematic)
  - dynamic simulation and static analysis

# Simple diagram of the front-end design flow



# Simple diagram of the back-end design flow



# Flow of placement and routing

- Floorplan (place macros, do power planning)
- Placement and in-place optimization
- Clock tree generation
- Routing

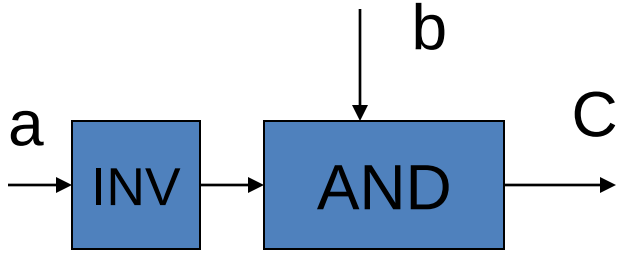
# Import needed files

- Gate level verilog (.v)
- Geometry information (.lef)
- Timing information (.lib)

```
INV (.in (a),      .out (a_inv));
AND (.in1 (a_inv),   .in2 (b),  .out (c));
```

INV: 1um width  AND: 2 um width

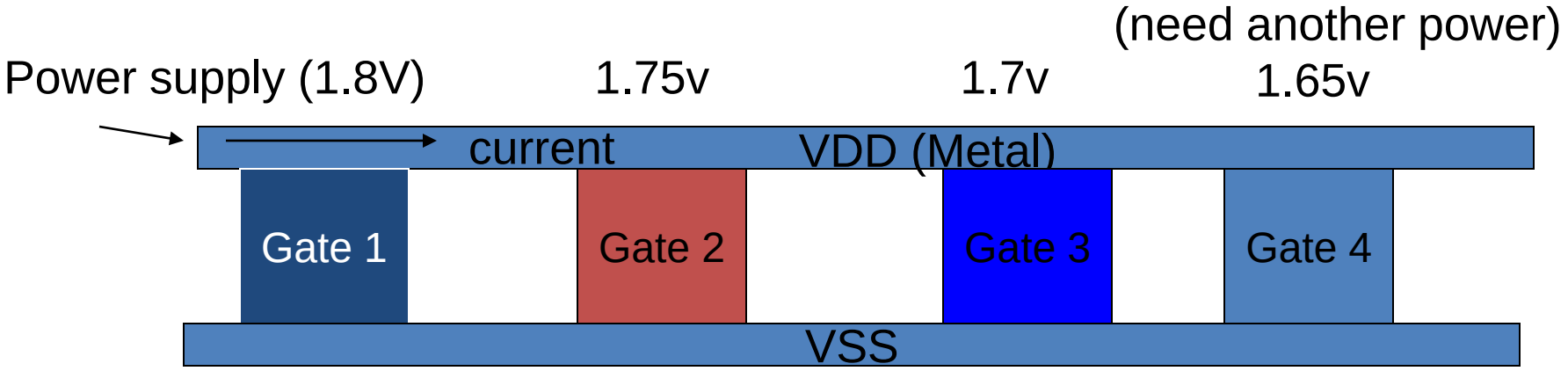
INV: 1ns delay;  AND: 2 ns delay




Delay (a->c): 1ns + 2ns = 3ns

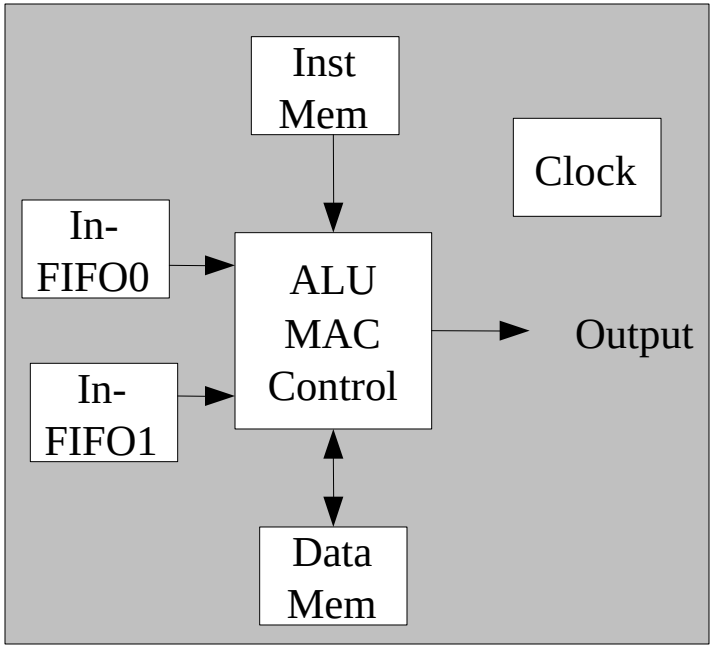
# Floorplan

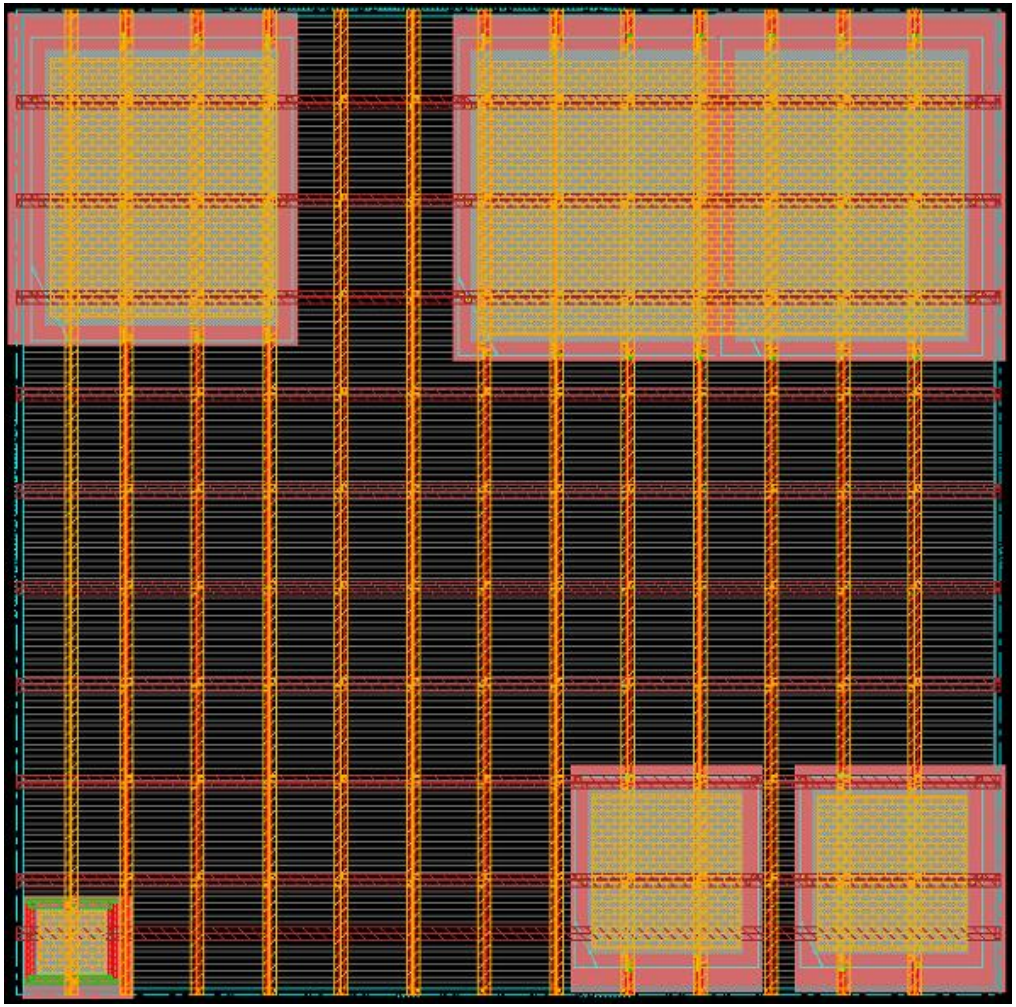
- Size of chip
- Location of Pins
- Location of main blocks
- Power supply: give enough power for each gate




Voltage drop equation: $V2 = V1 - I * R$

# Floorplan of a single processor

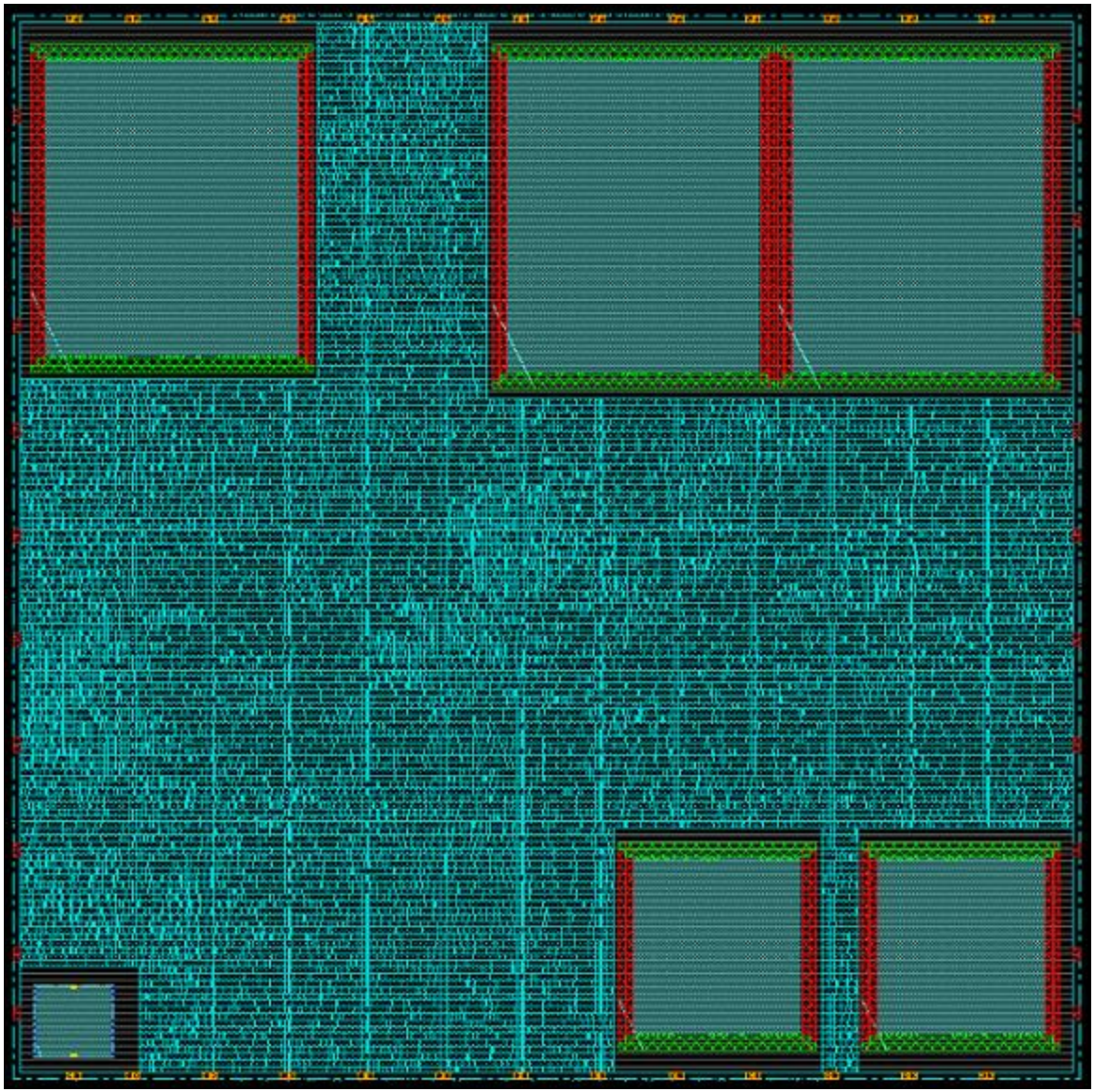


# Placement & in-placement optimization

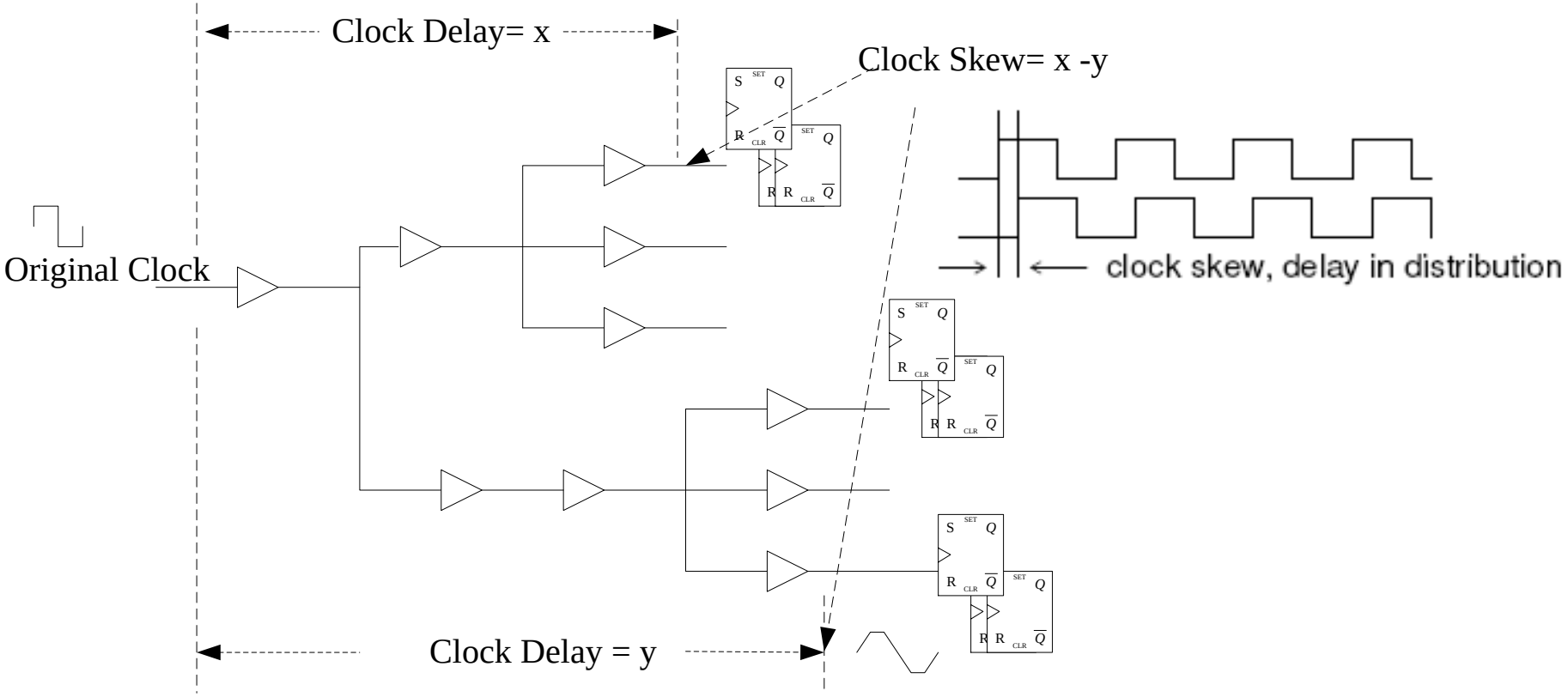
- Placement: place the gates
- In-placement optimization
  - Why: timing information difference between synthesis and layout (wire delay)
  - How: change gate size, insert buffers
  - Should not change the circuit function!!

# Placement of a single processor

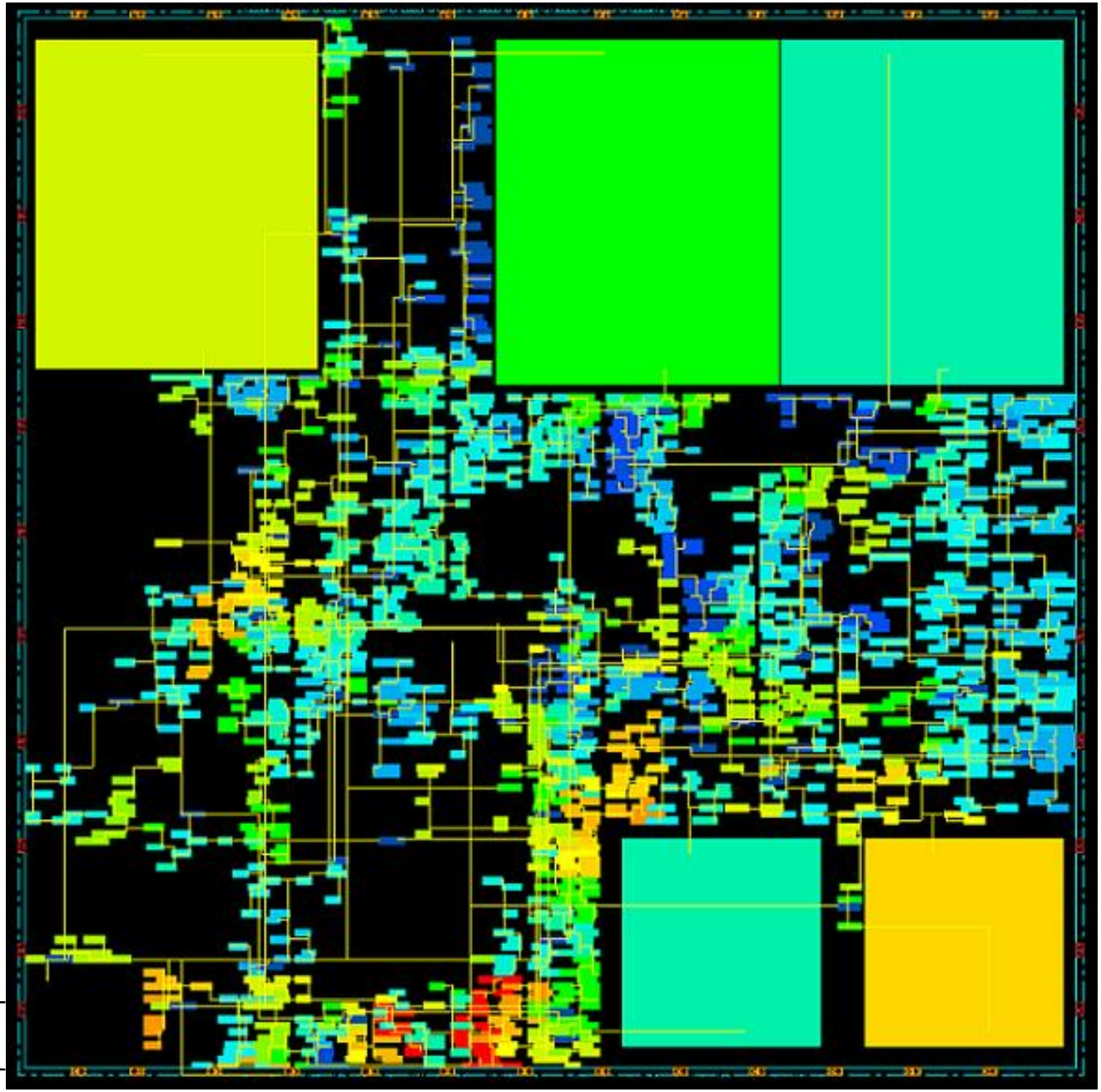


# Clock tree

- Main parameters: skew, delay, transition time



# Clock tree of single processor

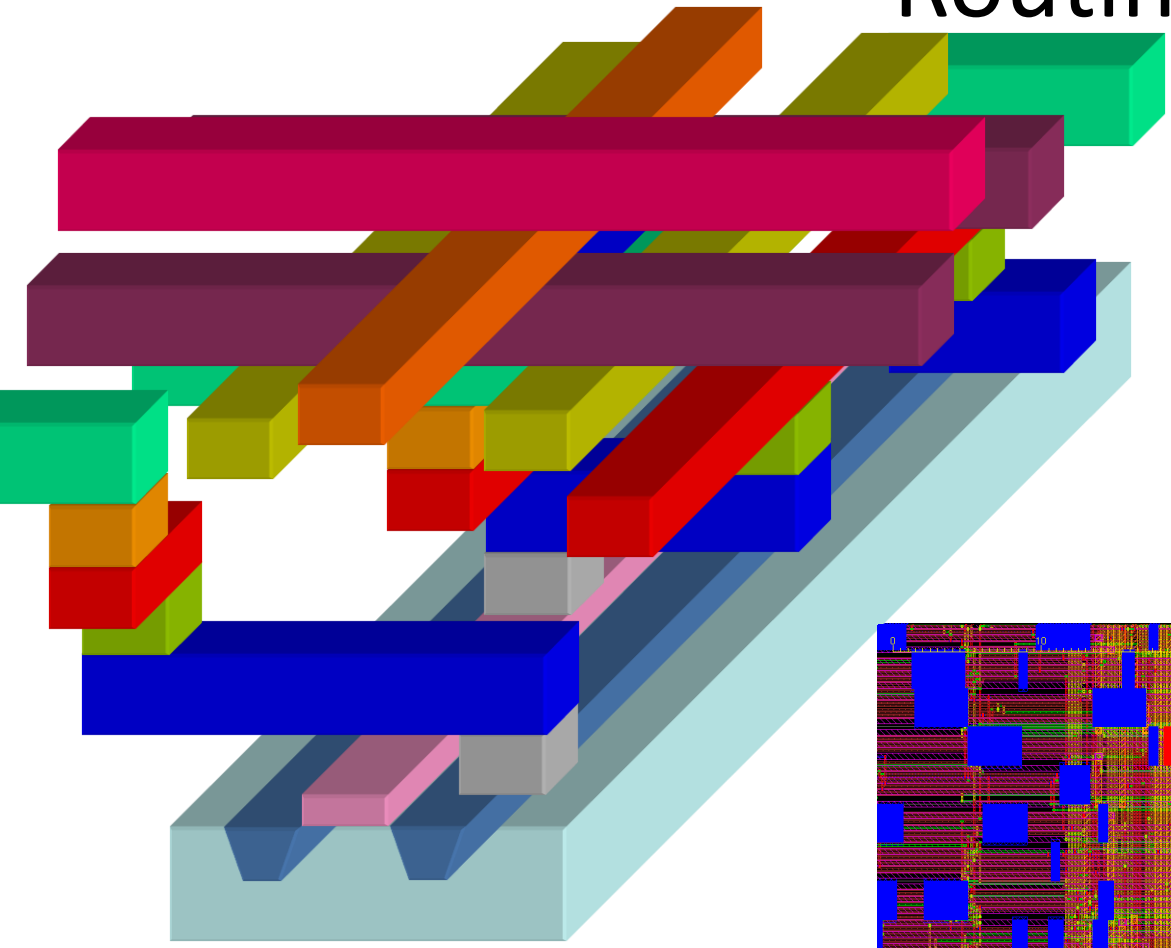


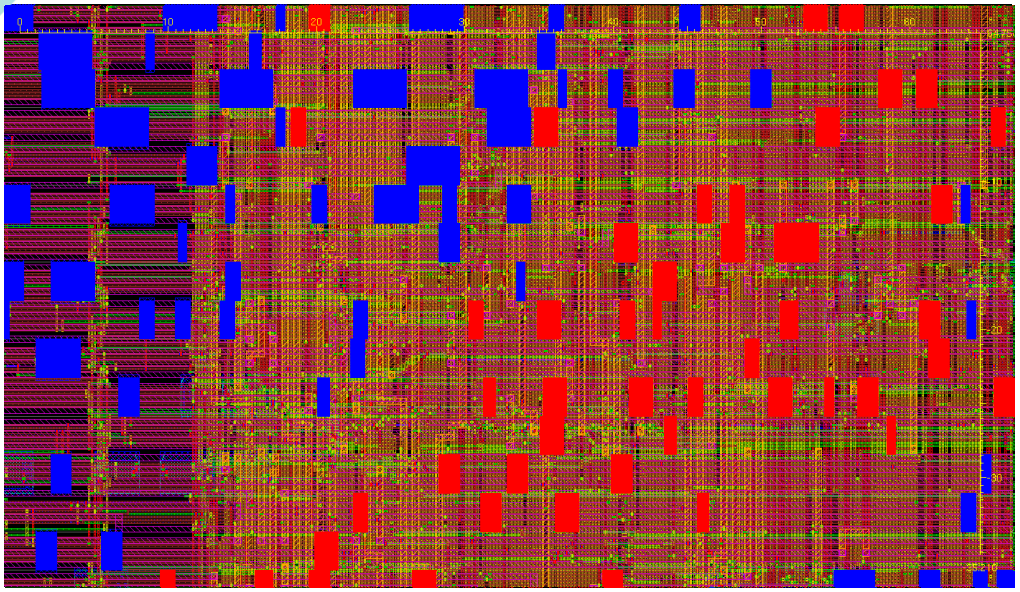
UM

# Routing

- Connect the gates using wires
- Two steps
  - Connect the global signals (power)
  - Connect other signals

# Routing

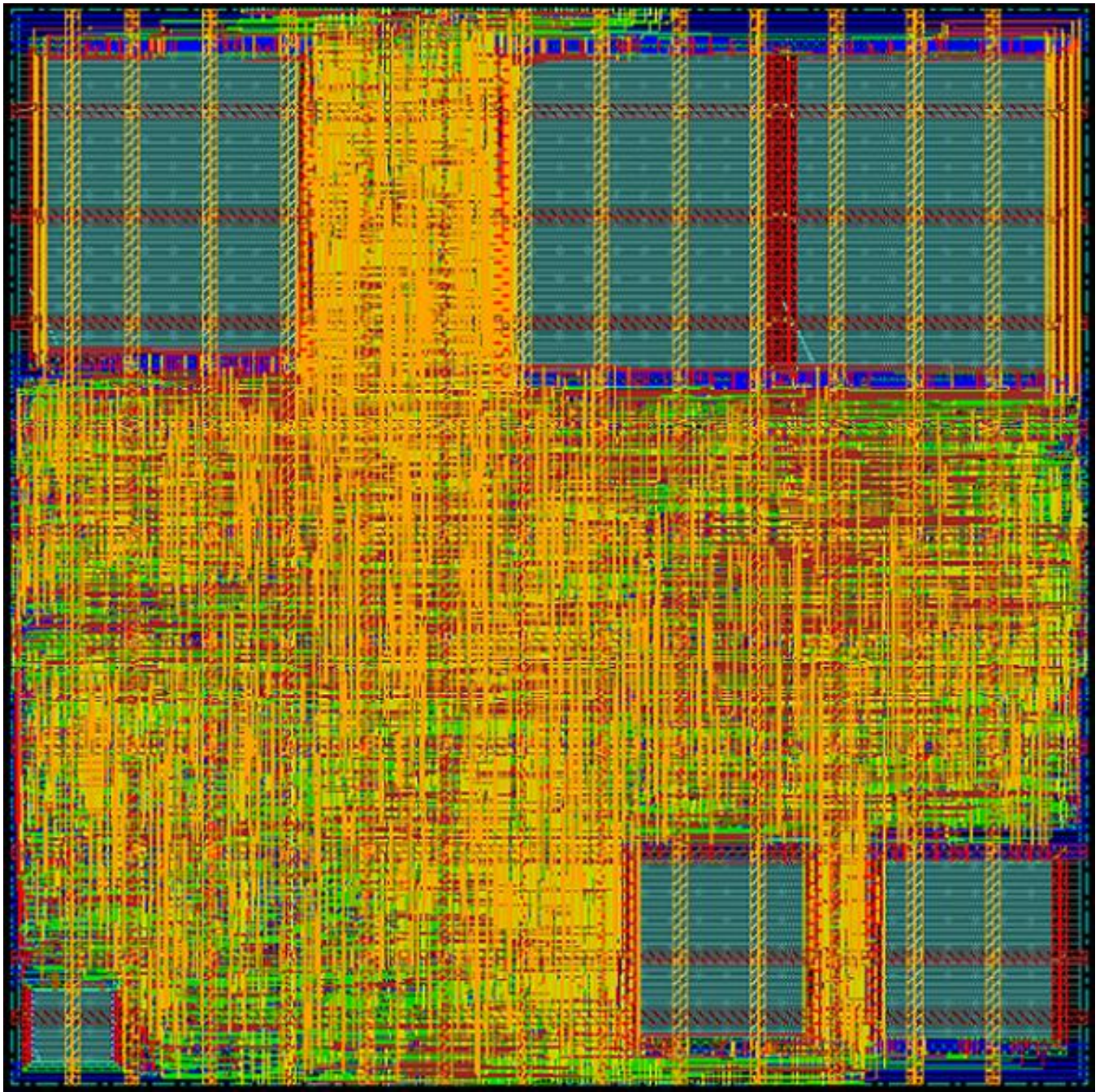




Metal Layer Topology

UMBC

# Layout of a single processor

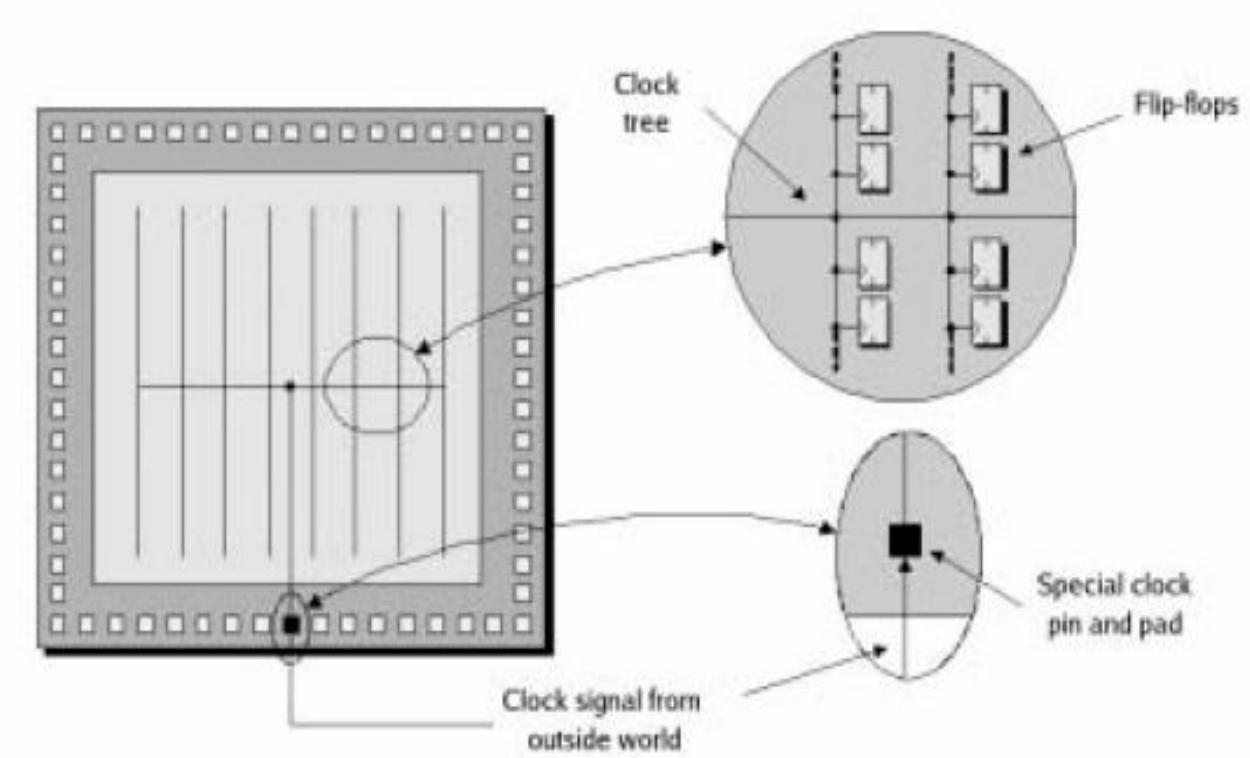


Area:
0.8mm x 0.8mm

Estimated speed:
450 MHz

# Clock Tree in FPGAs

- Everything is preplaced and routed (there is no space for improvement)
- There is no gate sizing to enhance performance



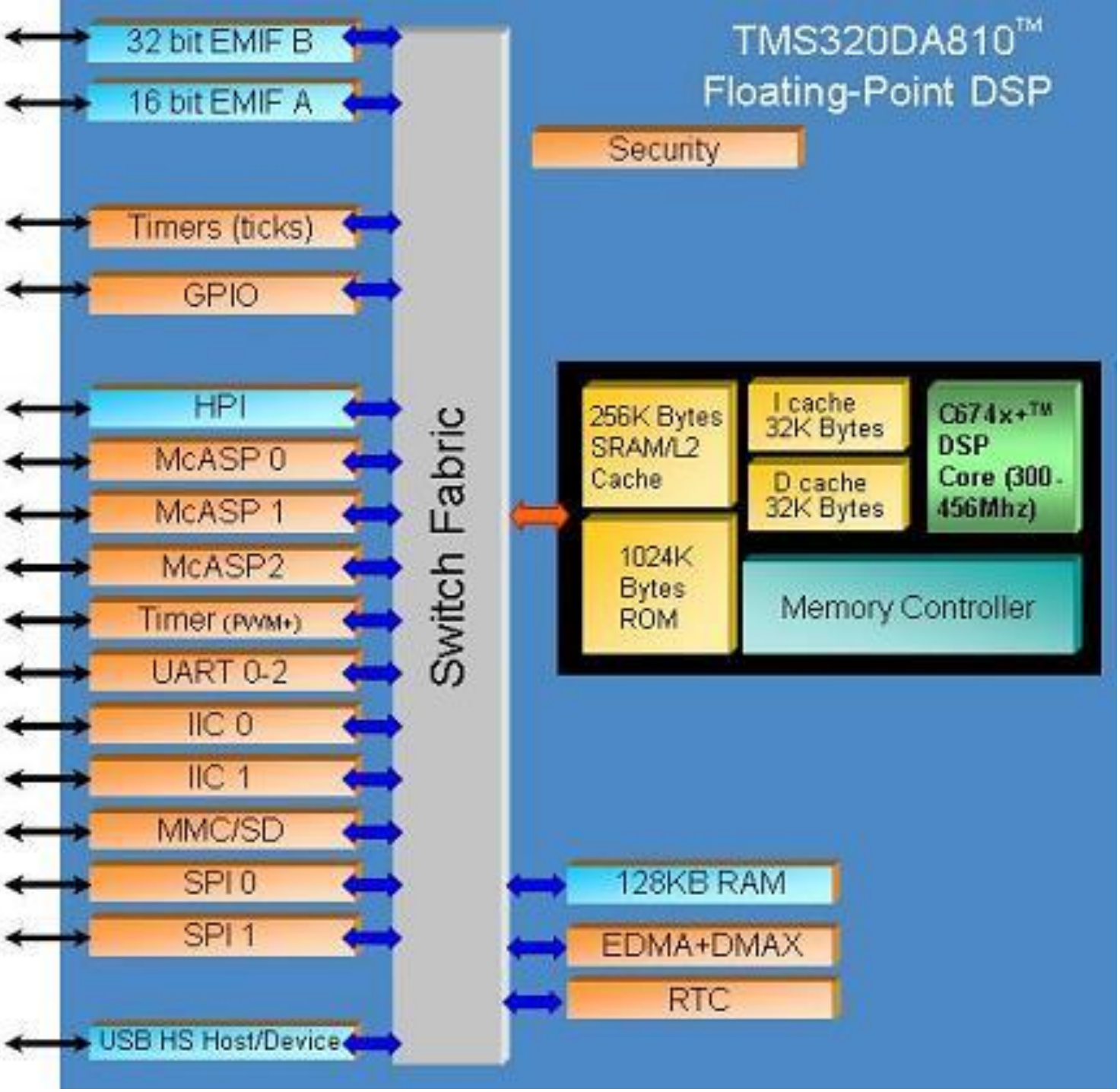

The Design Warrior's Guide to FPGAs,
ISBN 0750676043,
Copyright(C) 2004 Mentor Graphics Corp

# FPGA vs ASIC summary

- Front-end  design flow is almost the same for both
- Back-end design flow optimization is different
  - ASIC design: freedom in routing, gate sizing, power gating and clock tree optimization.
  - FPGA design: everything is preplaced, clock tree is pre-routed, no power gating
  - Designs implemented in FPGAs are slower and consume more power than ASIC

# FPGA vs DSP

UMBC

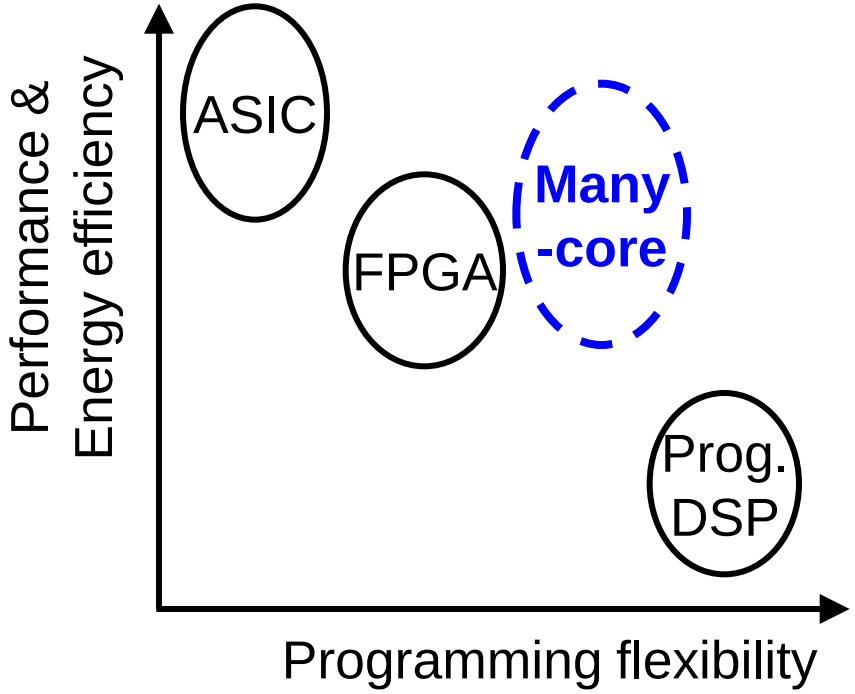

TMS320DA810™
Floating-Point DSP
32 bit EMIF B
16 bit EMIF A
Security
Timers (ticks)
GPIO
HPI
McASP 0
McASP 1
McASP2
Timer (PWM+)
UART 0-2
IIC 0
IIC 1
MMC/SD
SPI 0
SPI 1
USB HS Host/Device
Switch Fabric
256K Bytes SRAM/L2 Cache
I cache 32K Bytes
D cache 32K Bytes
C674x+™ DSP Core (300-456Mhz)
1024K Bytes ROM
Memory Controller
128KB RAM
EDMA+DMAX
RTC

# FPGA vs DSP

- DSP:
  - Easy to program (usually standard C)
  - Very efficient for complex sequential math-intensive tasks
  - Fixed datapath-width. Ex: 24-bit adder, is not efficient for 5-bit addition
  - Limited resources
- FPGA
  - Requires HDL language programming
  - Efficient for highly parallel applications
  - Efficient for bit-level operations
  - Large number of gates and resources
  - Does not support floating point, must construct your own.

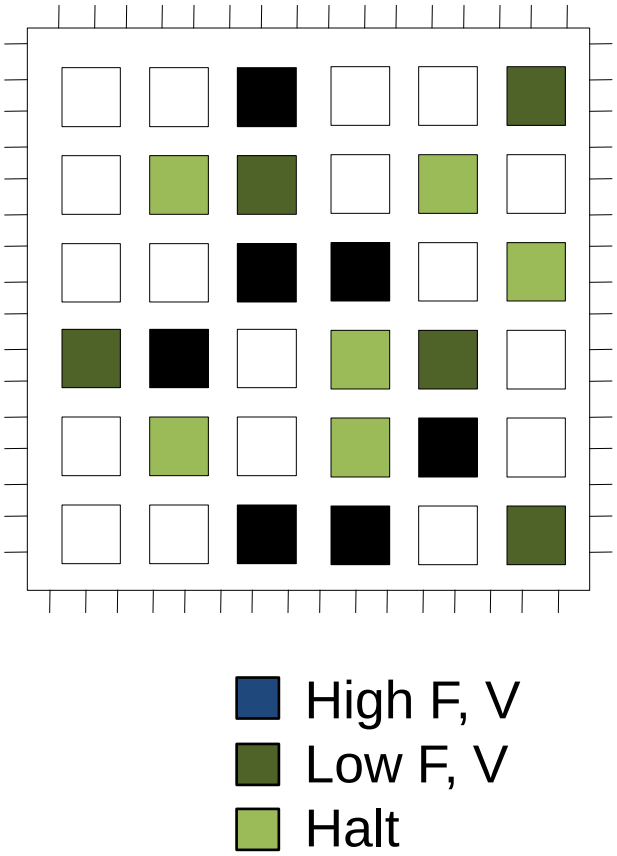
# Current trend

- Programming flexibility
- High performance
  - Throughput
  - Latency
- High energy efficiency
- Suitable for future fabrication technologies



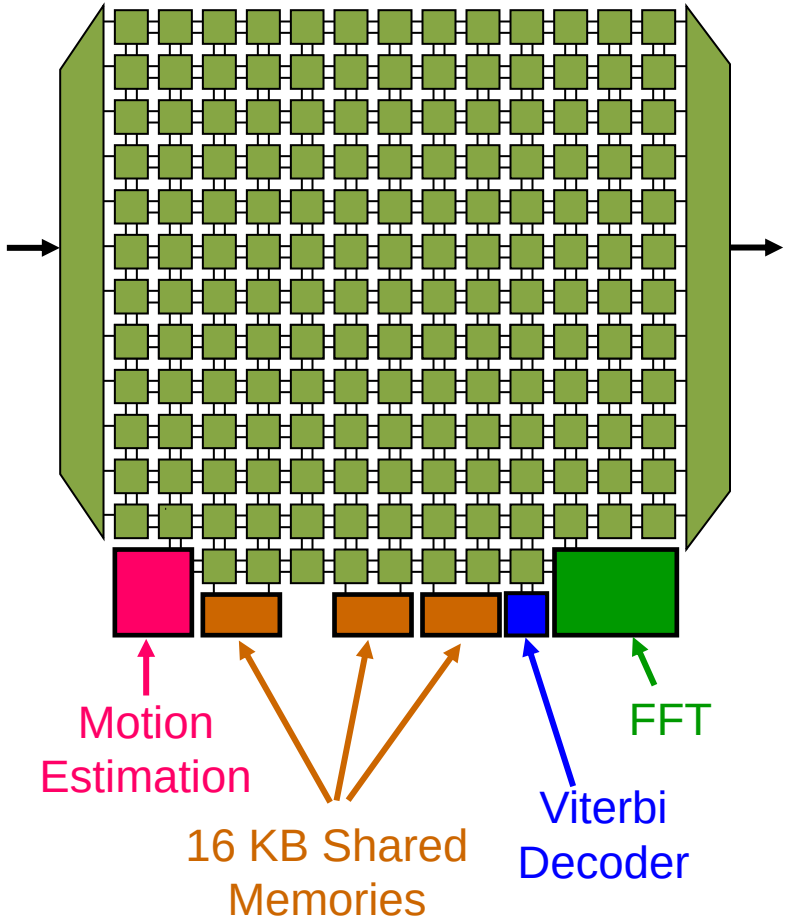
# Target Many-core Architecture

- High performance
  - Exploit task-level parallelism in digital signal processing and multimedia
  - Large number of processors per chip to support multiple applications
- High energy efficiency
  - Voltage and frequency scaling capability per processor



# 167-processor Multi-voltage Computational Chip

- 164 programmable procs.
- Three dedicated-purpose procs.
- Per processor Dynamic Voltage and Frequency Scaling (DVFS)
  - Selects between two voltages (VDD High and VDD Low)
  - Programmable local oscillator



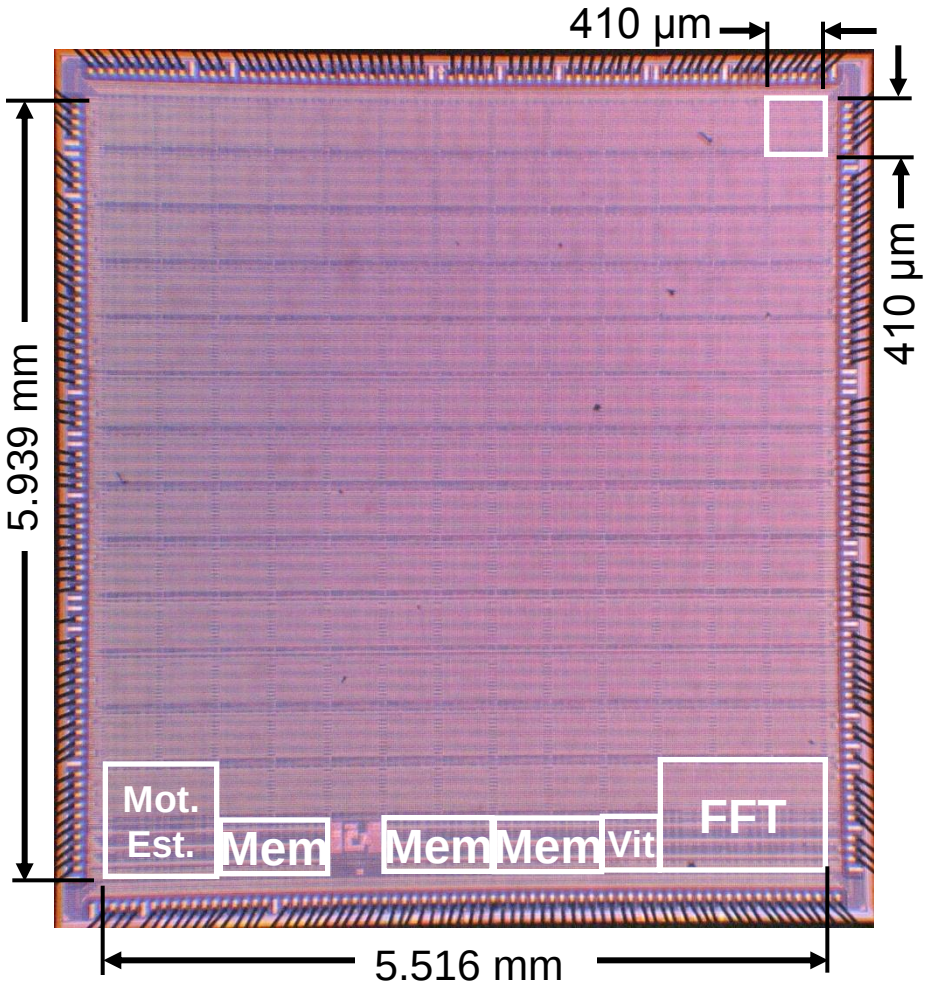



D. Truong, W. Cheng, T. Mohsenin, Z. Yu, A. Jacobson, G. Landge, M. Meeuwsen, C. Watnik, A. Tran, Z. Xiao, E. Work, J. Webb, P. Mejia, B. Baas, VLSI Symp. 2008, JSSC 2009

# Summary of the 167 Many-core Chip

| Single Tile | |
|---|---|
| Transistors | 325,000 |
| Area | 0.17 $mm^2$ |
| CMOS Tech. | 65 nm ST Microelectronics low-leakage |
| Max. frequency | 1.19 GHz @ 1.3 V |
| Power (100% active) | 59 mW @ 1.19 GHz, 1.3 V |
| | 47 mW @ 1.06 GHz, 1.2 V |
| | 608 µW @ 66 MHz, 0.675 V |
| App. power (802.11a rx) | 16 mW @ 590 MHz, 1.3 V |

55 million transistors, 39.4 $mm^2$

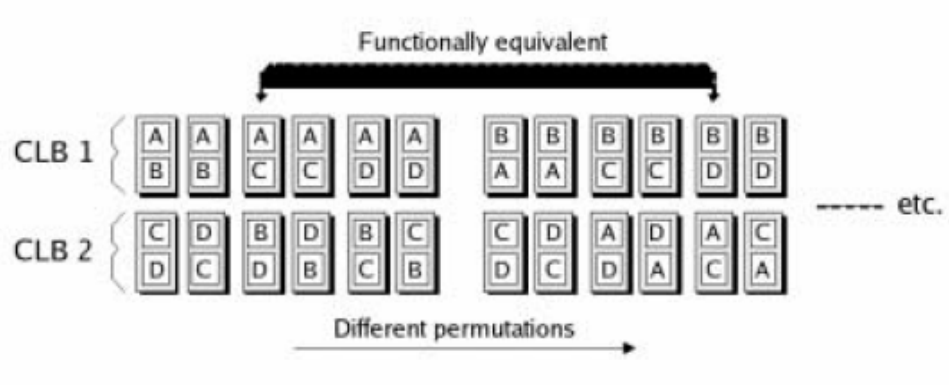


# Design Flow

**Packing** follows the *mapping* step in which the LUTs and registers are packed into CLBs.

This step is also non-trivial b/c there are many ways to combine and permute the LUTs to CLBs.

For example, assume we have a netlist that maps to 4 LUTs and the FPGA can contain 2 LUTs per CLB, then there are 4! different ways to pack the LUTs.



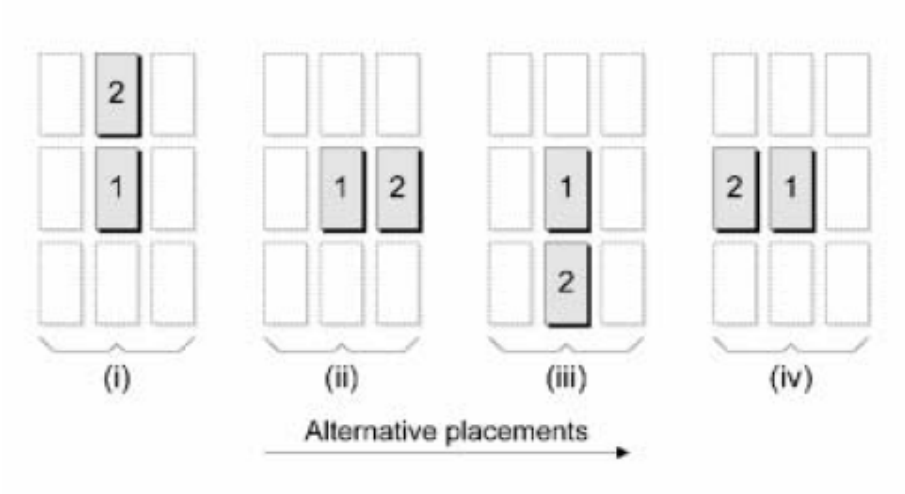



Although some of these are equivalent, e.g., AC-BD and BD-AC, the number of possibilities is still very large.

# Design Flow

**Place-and-Route** refers to selecting the CLBs and configuring the interconnect.

Assume there are 2 CLBs that need to be connected together and, further, that they need to be adjacent to each other.






Since there are only 2 LUTs and 4 possibilities here, this task is fairly simple.

However, with hundreds of thousands of CLBs, each of which may connect to one or more CLBs, finding optimal placement is non-trivial.

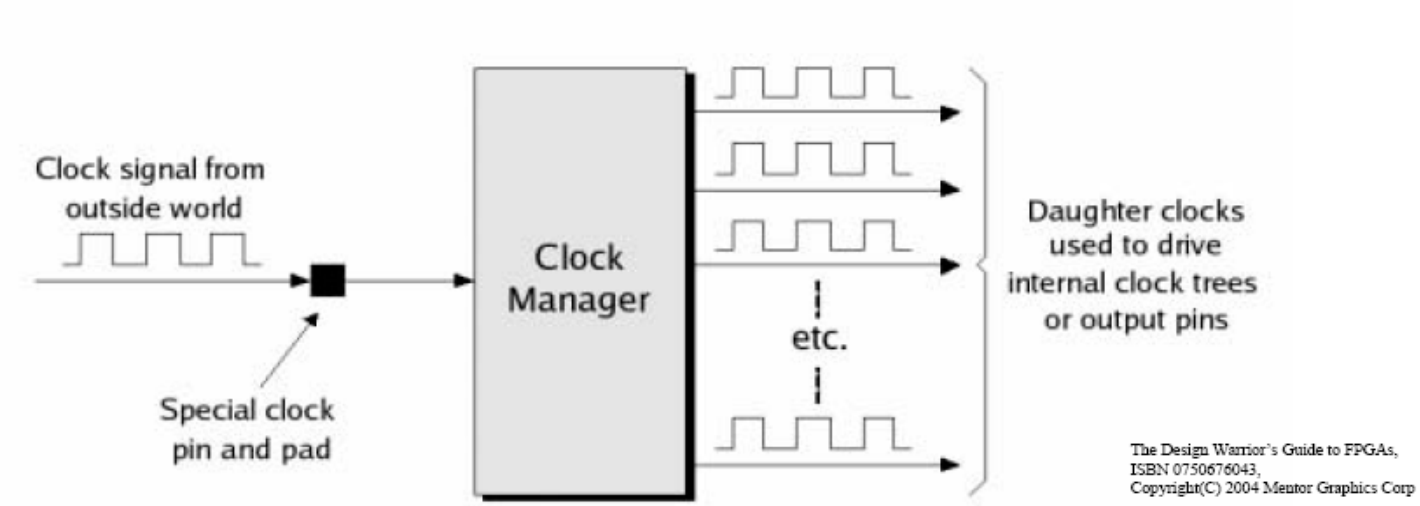
# Features and Specifications of FPGAs

**Clock Trees and Clock Managers**

The clock tree is routed using special tracks in the FPGA, and is designed to minimize **clock skew**.

In reality, there are multiple clock pins to support *multiple clock domains* within the FPGA.

The *clock manager* is a special hard-wired function that can receive the external clock signal and generate **daughter clocks**.



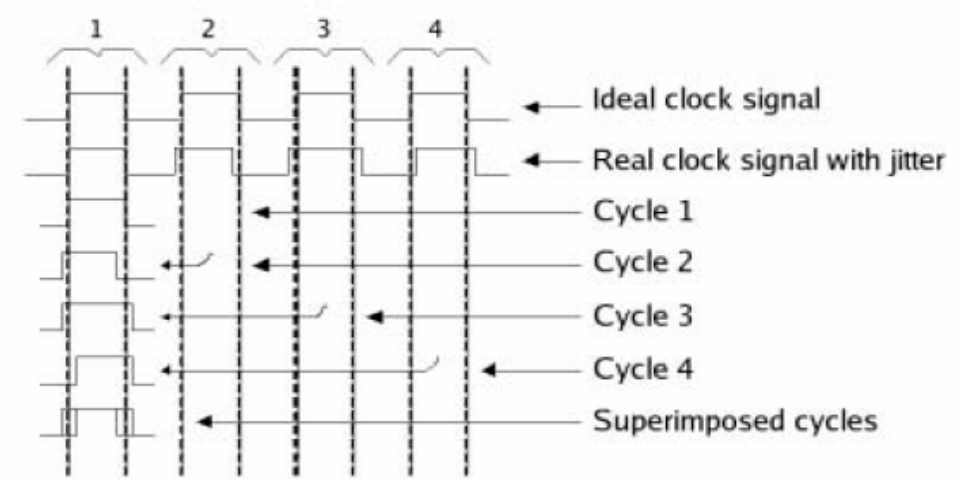
# Features and Specifications of FPGAs

**Clock Trees and Clock Managers**

The daughter clocks can then be used to drive internal clock trees, or output pins for distribution to other chips on the PCB.

Clock managers can support the following features:

- *Jitter removal*: Uncertainty in the exact arrival time of each clock edge results in jitter.



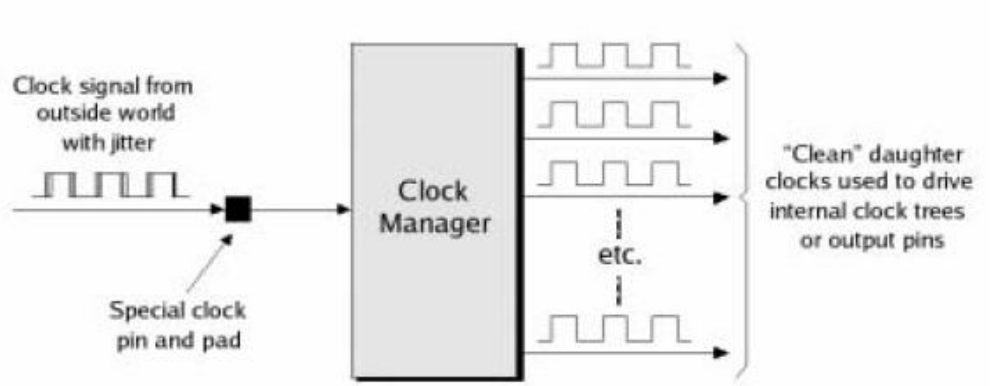

The Design Warrior's Guide to FPGAs,
ISBN 0750676043,
Copyright(C) 2004 Mentor Graphics Corp

The clock manager can be used to *detect* and *correct* for this jitter, and therefore provide **clean** daughter clock signals.

# Features and Specifications of FPGAs

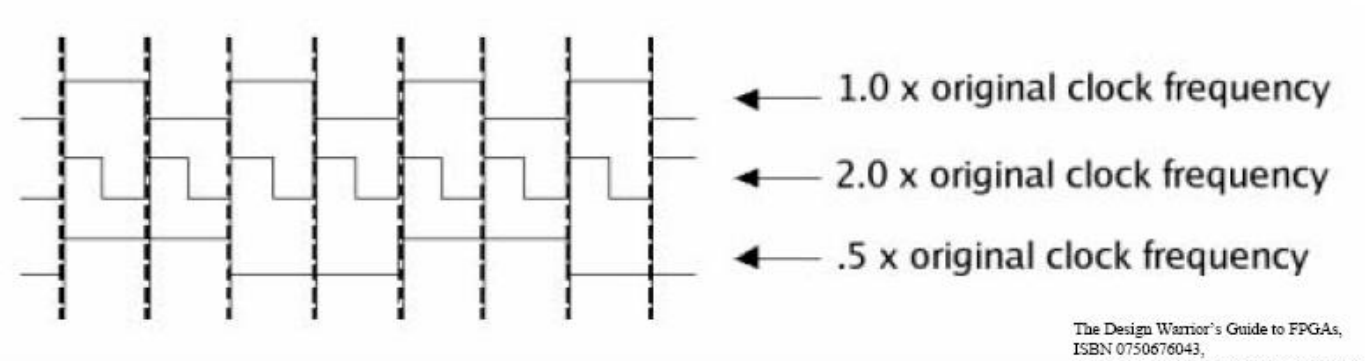
**Clock Trees and Clock Managers**

- *Jitter removal*





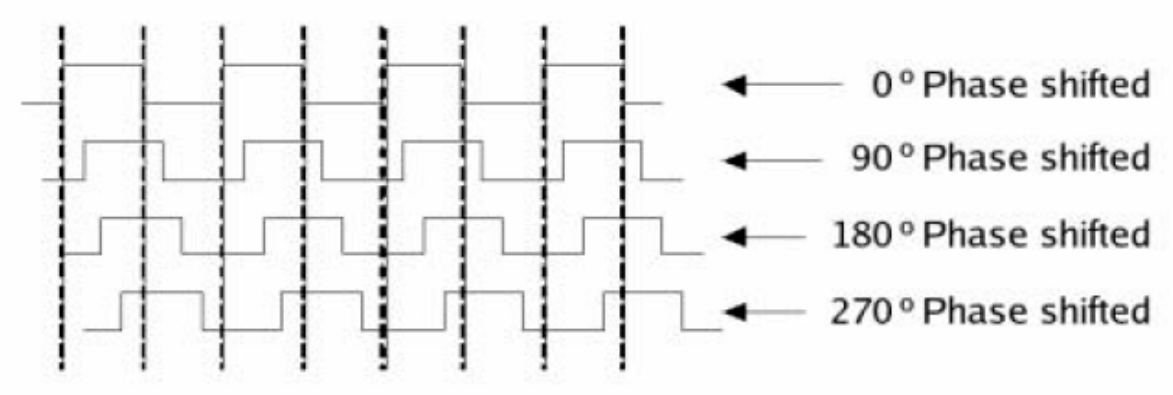

- *Frequency synthesis*: Allows the external clock frequency to divided or multiplied.



# Features and Specifications of FPGAs

**Clock Trees and Clock Managers**

- *Phase shifting*: Phase shifting a clock with respect to another adds delay to it.




The Design Warrior's Guide to FPGAs,
ISBN 0750676043,
Copyright(C) 2004 Mentor Graphics Corp

Common phase shifts are 120 and 240 degrees (for 3-phase clk schemes) and 90, 180 and 270 degrees for 4-phase schemes.

Some clock managers allow *any* value to be set for each daughter clk.

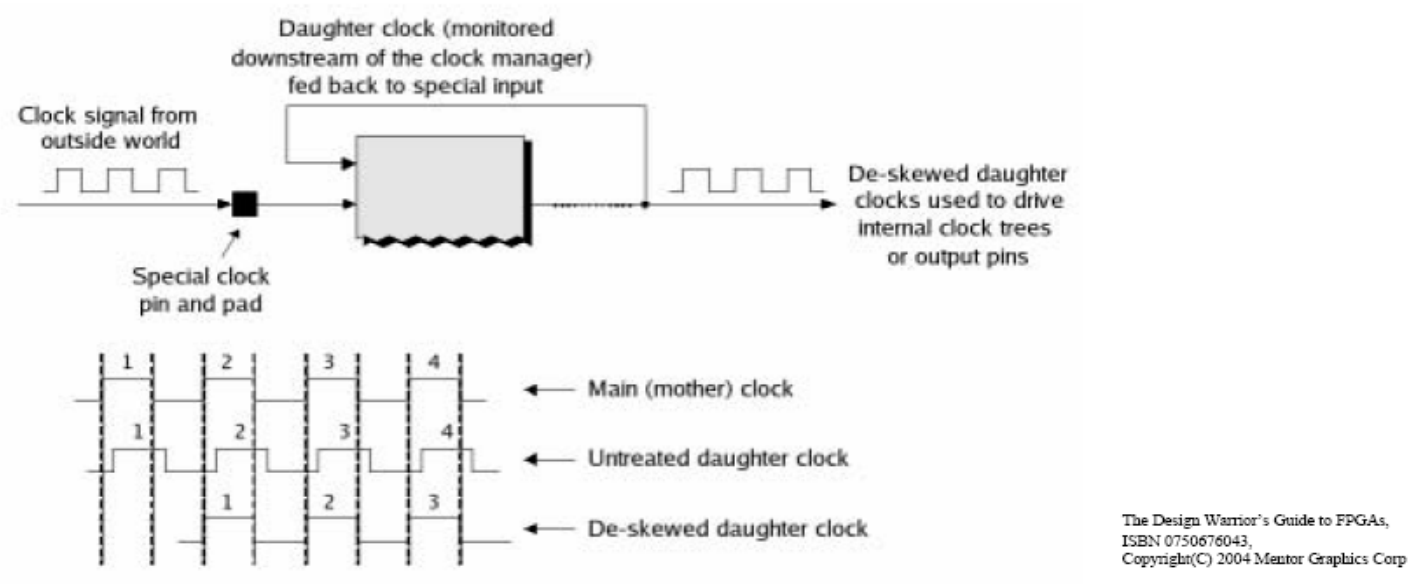
- *Auto-skew correction*: Allows for automatic correction of delays introduced by the clock manager and interconnect.

  This is accomplished by "feeding back" the daughter clock to be corrected.

# Features and Specifications of FPGAs

**Clock Trees and Clock Managers**

- *Auto-skew correction* (cont.)




The Design Warrior's Guide to FPGAs,
ISBN 0750676043,
Copyright(C) 2004 Mentor Graphics Corp

Skew in the daughter is removed by comparing it with the external clk and delaying it until the edges re-align.

Some clock managers are based on *phase-locked loops* (PLLs) and others on *digital delay-locked loops* (DLLs).

Trade-offs include precision, stability and noise insensitivity.

# Features and Specifications of FPGAs

**General-Purpose I/O**

With 1000 or more pins on today's FPGAs, special packages that allow for *area arrays* are used (2-dimensional distribution of pads across the chip).

For simplicity, let's assume the older style of packaging with peripheral I/Os.

*Configurable I/O standards:*

Given the wide variety of standards that exist for representing logic 0 and logic 1 at the board level, FPGA have *general purpose I/Os*.

They can be configured to accept and generate signals conforming to any one of these standards.

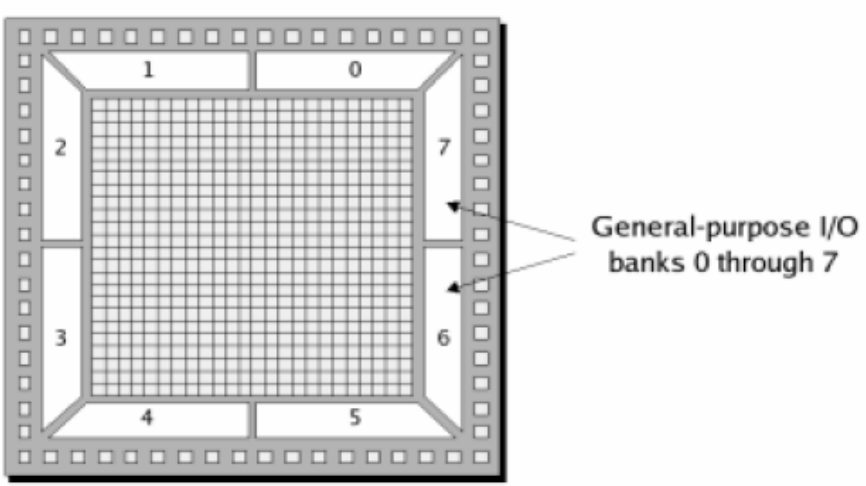
The I/Os are organized into a set of *banks*, each of which can be configured individually to support a particular I/O standard.

This increases the versatility of the FPGA and allows it to act as an interface between different I/O standards.

# Features and Specifications of FPGAs

**General-Purpose I/O**

*Configurable I/O standards:*






*Configurable I/O impedances*:

FPGAs I/O pins can be configured with specific impedances (terminating resistances) to cancel signal reflections.

Signal reflections result from very fast edge rates, that cause signals to bounce back and forth across the wires connecting chips.

# Backup

UMBC