L15: VLSI Integration and Performance Transformations

Acknowledgement:

Materials in this lecture are courtesy of the following sources and are used with permission.

Curt Schurgers

J. Rabaey, A. Chandrakasan, B. Nikolic. *Digital Integrated Circuits: A Design Perspective.* Prentice Hall/Pearson, 2003.

Layout 101

3-D Cross-Section

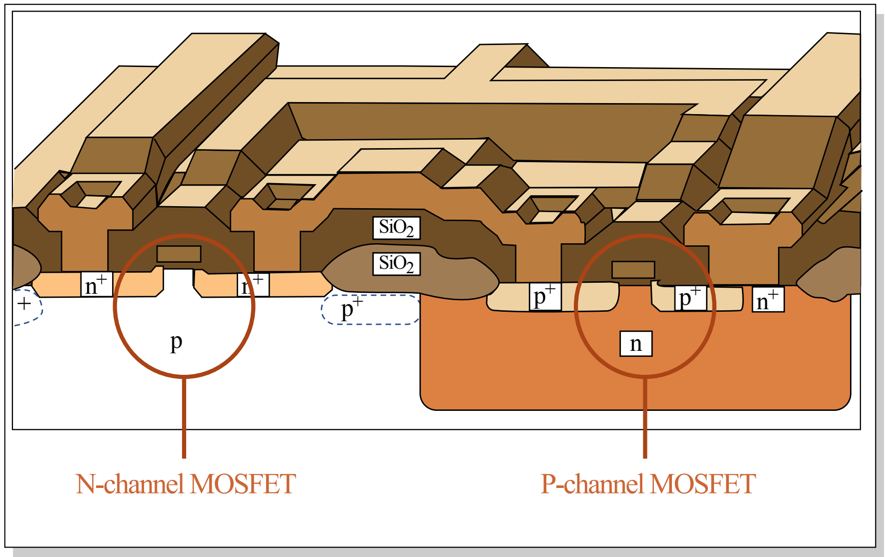
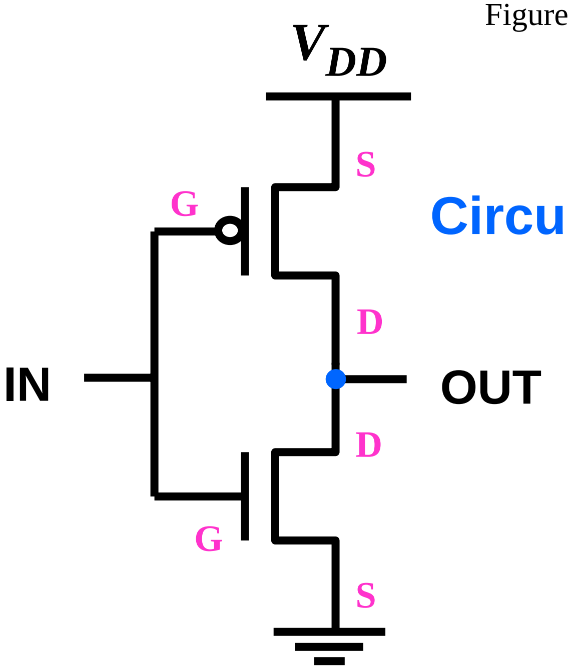
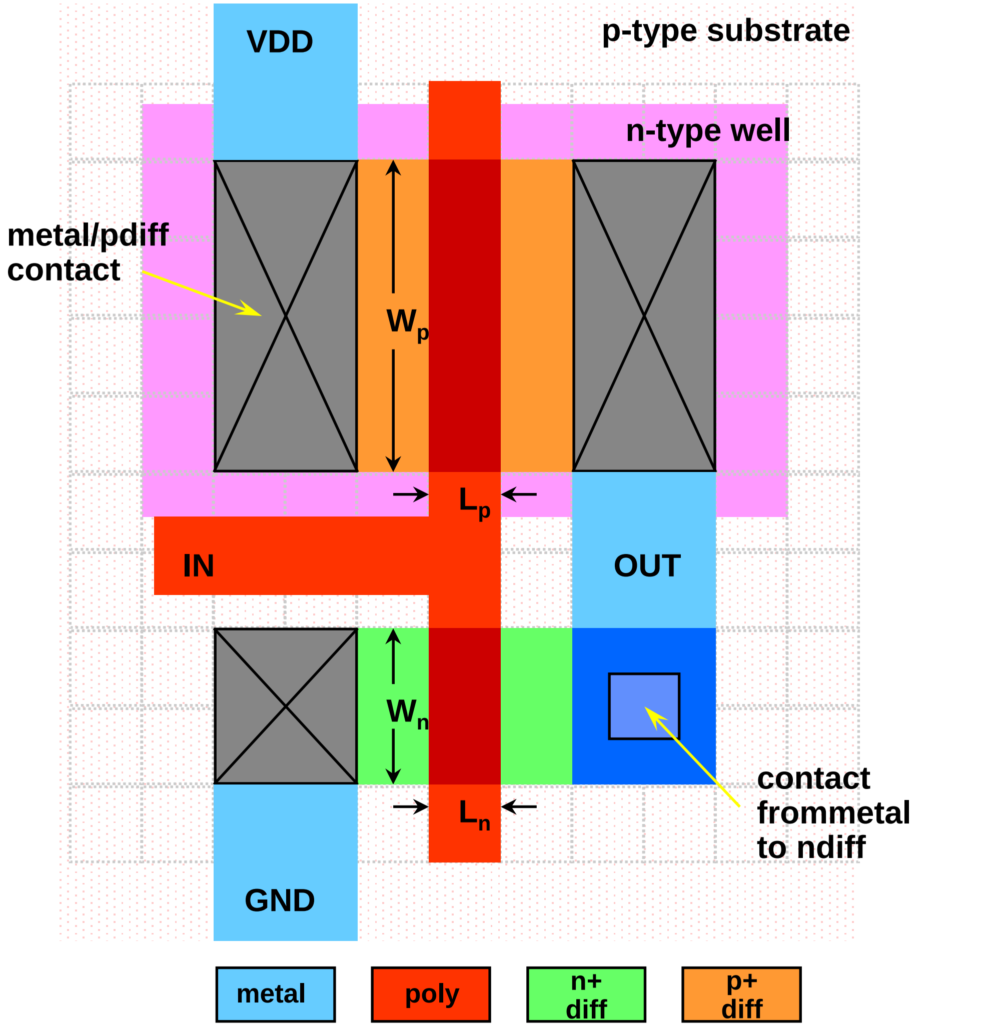


Figure by MIT OpenCourseWare.



Circuit Representation

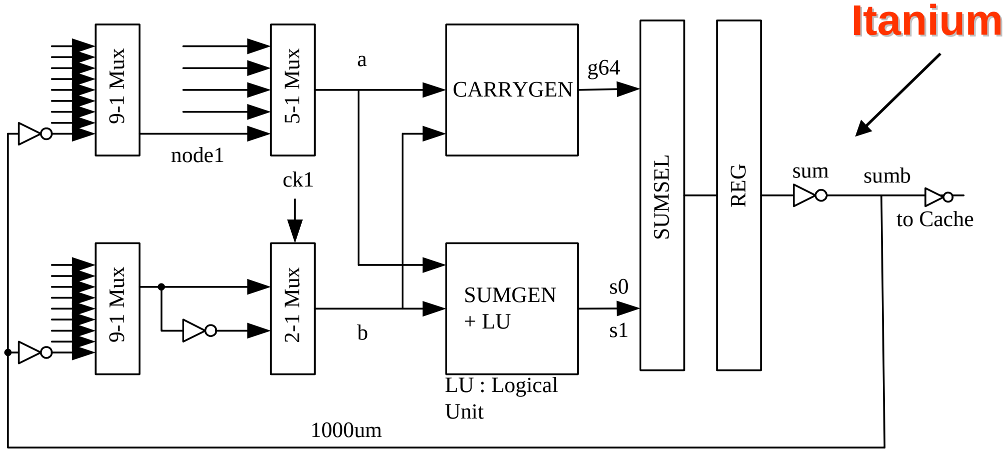


Layout

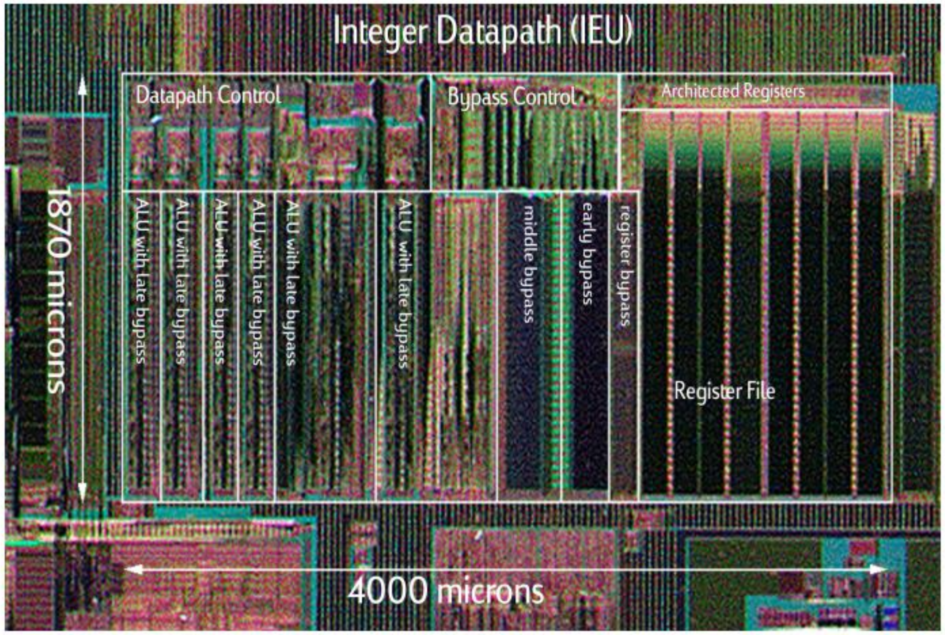
Used with permission.

- **Follow simple design rules (contract between process and circuit designers)**

Custom Design/Layout



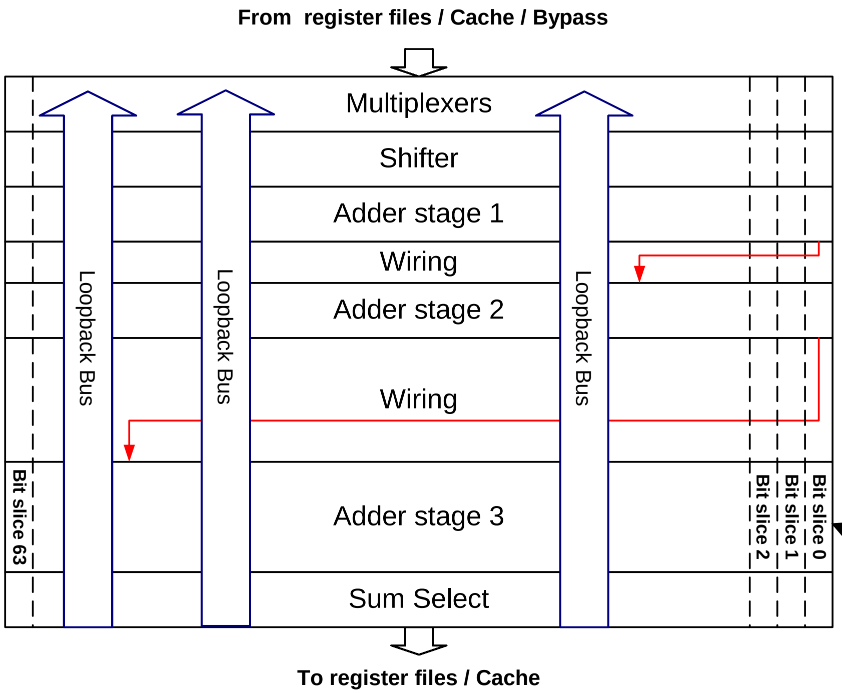
Itanium has 6 integer execution units like this



Die photograph of the

Itanium integer datapath

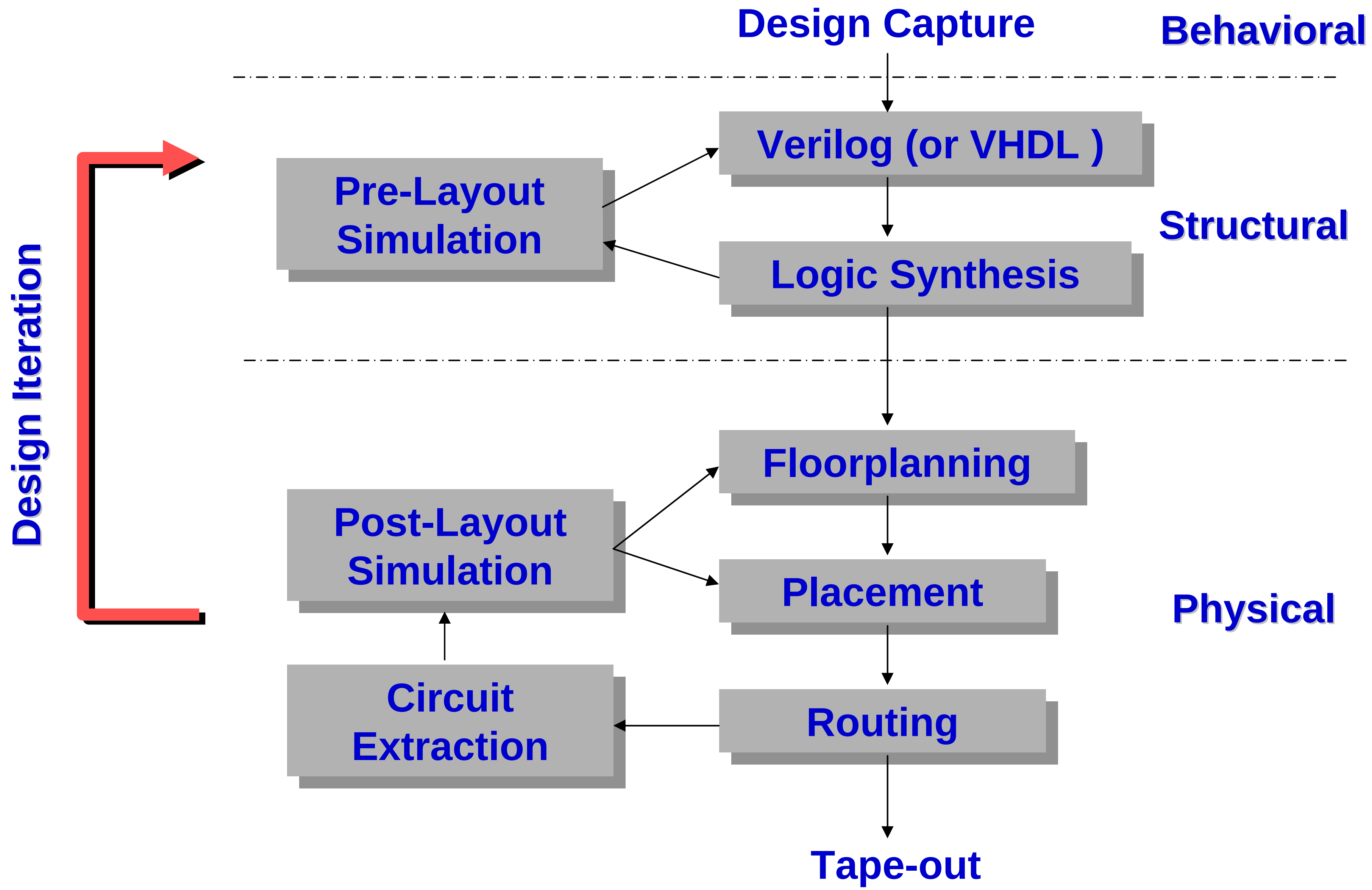
Courtesy Intel, as reprinted in Rabaey, et al. "Digital Integrated Circuits".



Bit-slice Design Methodology

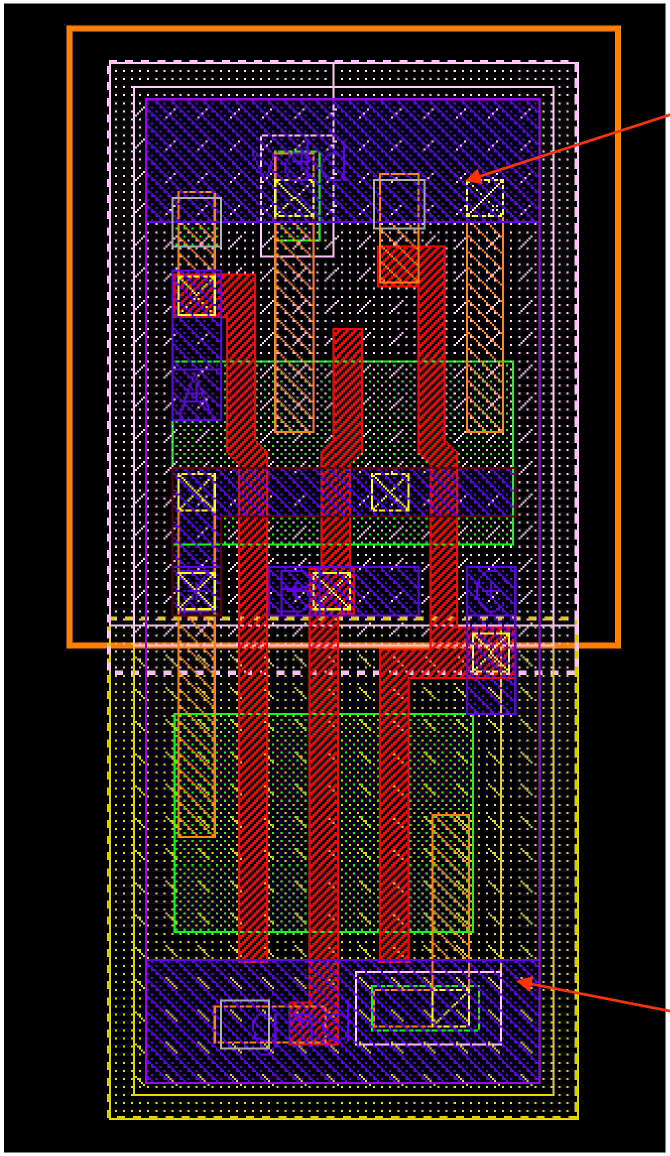
- Hand crafting the layout to achieve maximum clock rates (> 1Ghz)
- Exploits regularity in datapath structure to optimize interconnects

The ASIC Approach



Most Common Design Approach for Designs up to 500Mhz Clock Rates

Standard Cell Example



Power Supply Line (V_{DD})

Delay in (ns)!!

Path	1.2V - 125°C	1.6V - 40°C
$In1$—t_{pLH}	$0.073+7.98C+0.317T$	$0.020+2.73C+0.253T$
$In1$—t_{pHL}	$0.069+8.43C+0.364T$	$0.018+2.14C+0.292T$
$In2$—t_{pLH}	$0.101+7.97C+0.318T$	$0.026+2.38C+0.255T$
$In2$—t_{pHL}	$0.097+8.42C+0.325T$	$0.023+2.14C+0.269T$
$In3$—t_{pLH}	$0.120+8.00C+0.318T$	$0.031+2.37C+0.258T$
$In3$—t_{pHL}	$0.110+8.41C+0.280T$	$0.027+2.15C+0.223T$

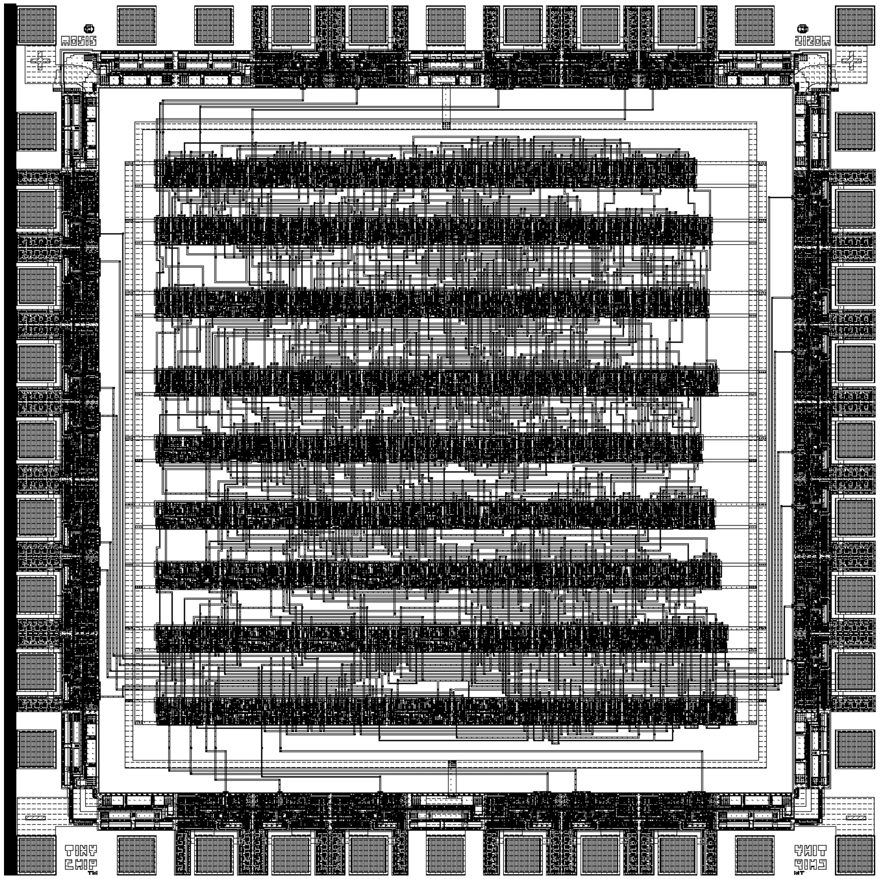
3-input NAND cell
(from ST Microelectronics):
C = Load capacitance
T = input rise/fall time

Ground Supply Line (GND)

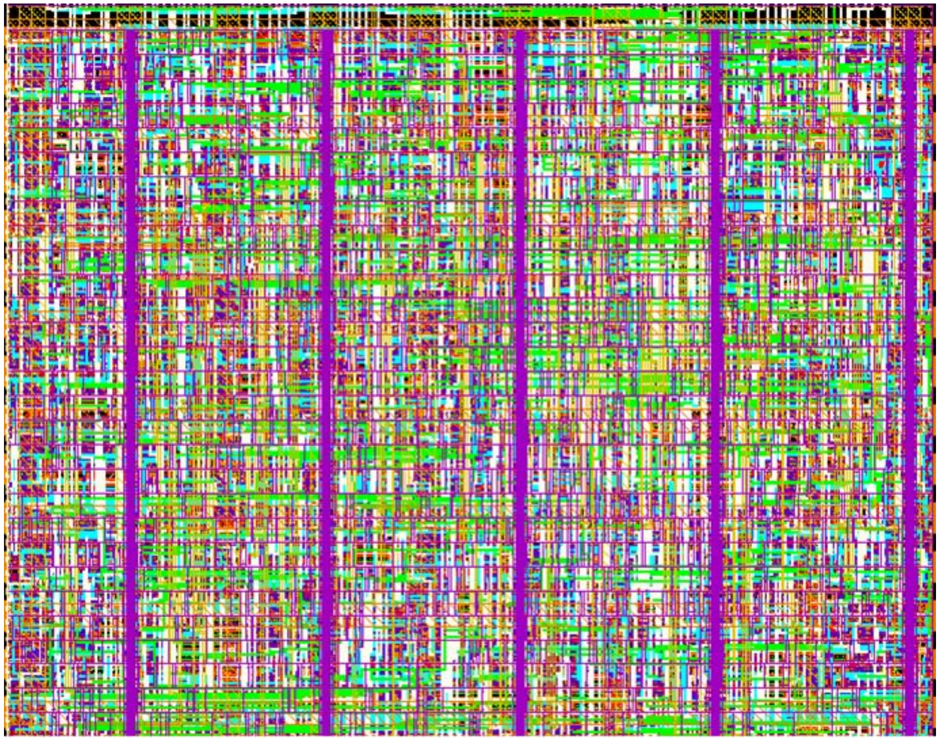
- **Each library cell (FF, NAND, NOR, INV, etc.) and the variations on size (strength of the gate) is fully characterized across temperature, loading, etc.**

Standard Cell Layout Methodology

2-level metal technology



Current Day Technology



Cell-structure hidden under interconnect layers

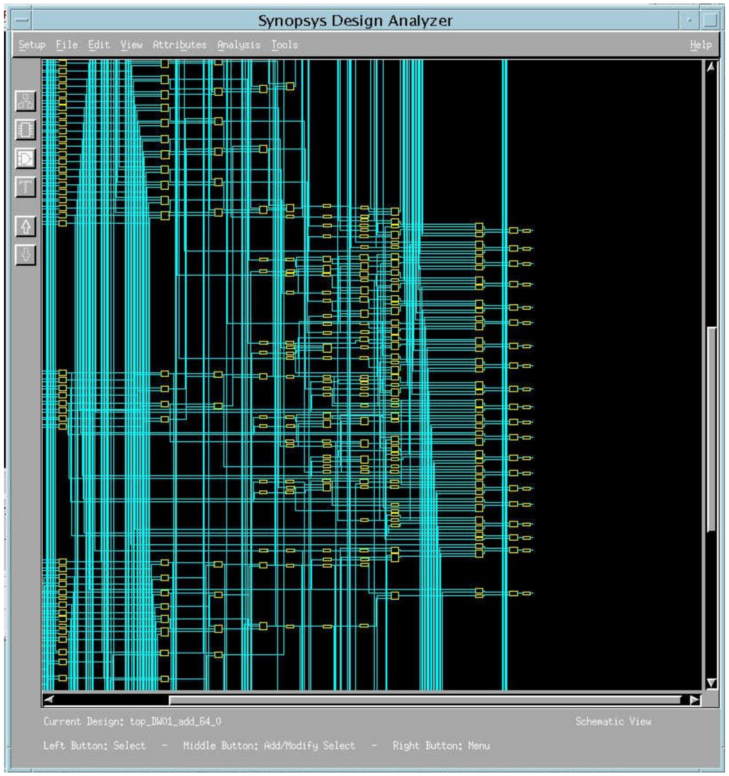
- With limited interconnect layers, dedicated routing channels between rows of standard cells are needed
- Width of the cell allowed to vary to accommodate complexity
- Interconnect plays a significant role in speed of a digital circuit

Verilog to ASIC Layout (the push button approach)

```
module adder64 (a, b, sum);
  input  [63:0] a, b;
  output [63:0] sum;

  assign sum = a + b;
endmodule
```

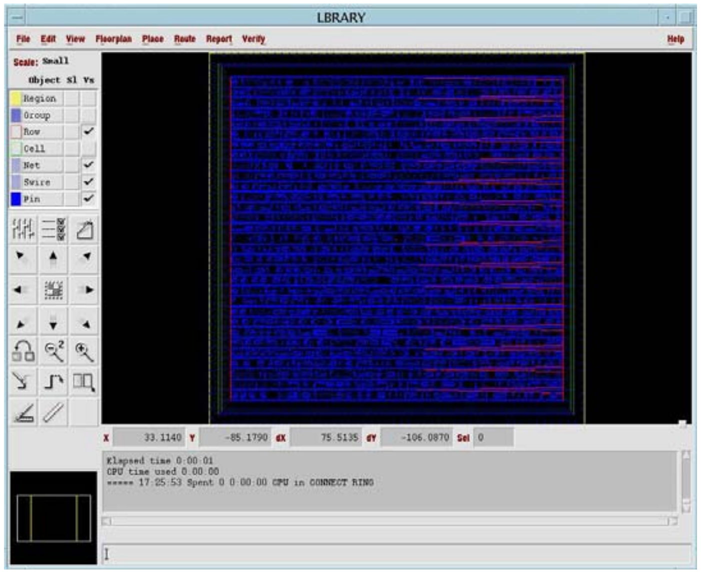
After Synthesis



Current Design: top_DW01_add_64_0   Schematic View

Left Button: Select - Middle Button: Add/Modify Select - Right Button: Menu

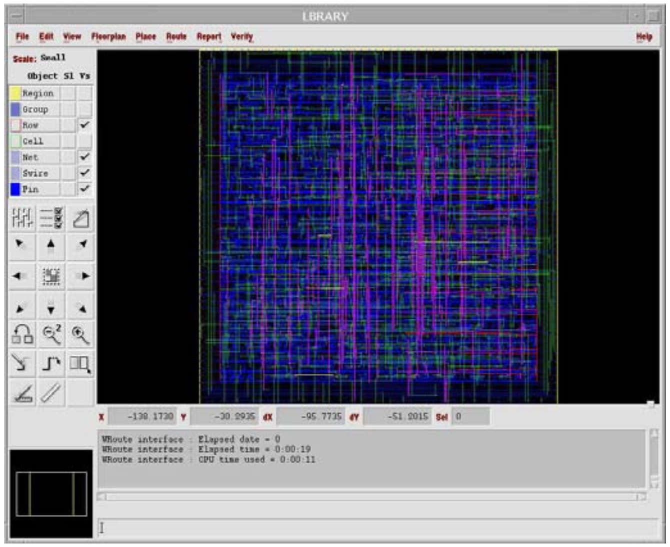
After Placement



X 33.1140 Y -85.1790 dX 75.5135 dY -106.0870 Sel 0

```
Elapsed time 0:00:01
CPU time used 0.00.00
===== 17:25:53 Spent 0 0:00:00 CPU in CONNECT RING
```

After Routing

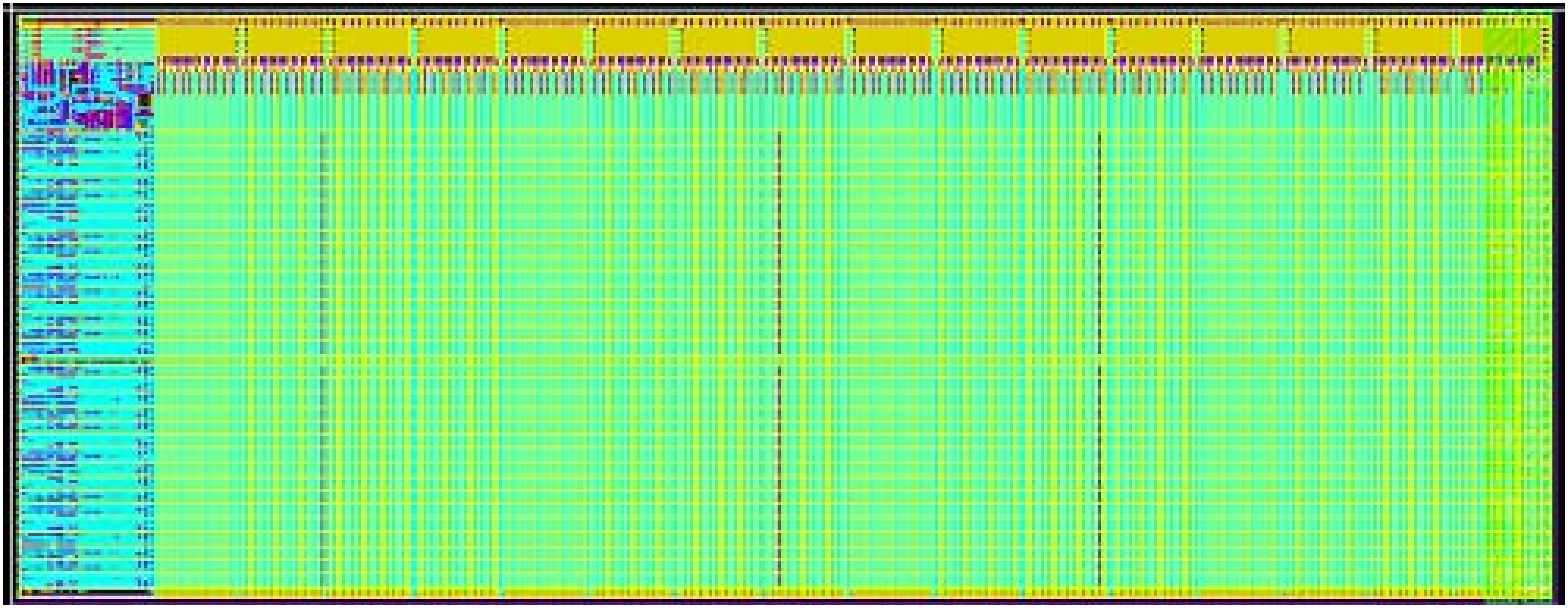


X -138.1730 Y -30.2935 dX -95.7735 dY -51.2015 Sel 0

```
WRoute interface : Elapsed date = 0
WRoute interface : Elapsed time = 0:00:19
WRoute interface : CPU time used = 0:00:11
```

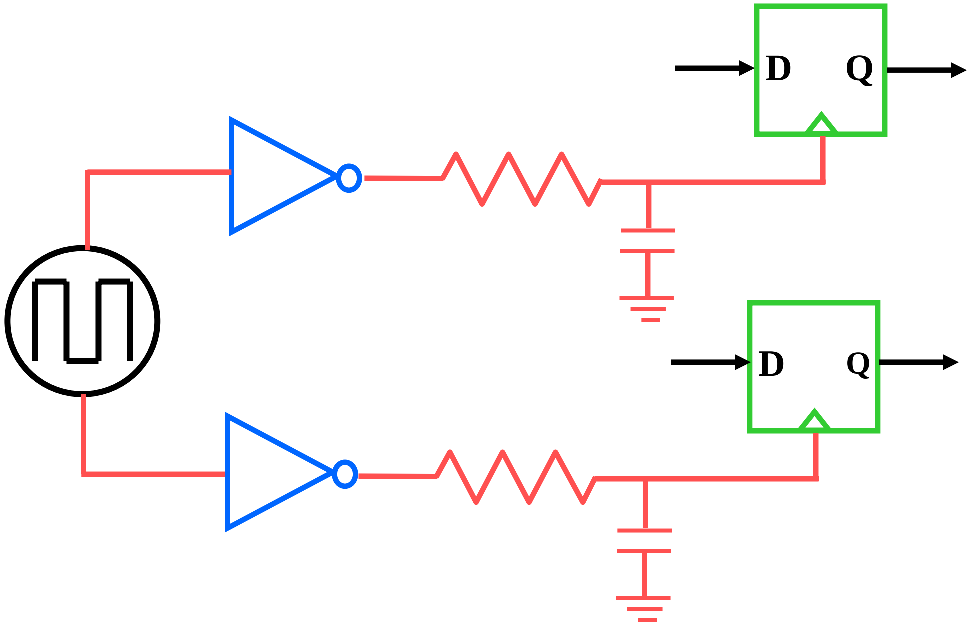
Macro Modules

256×32 (or 8192 bit) SRAM Generated by hard-macro module generator



- Generate highly regular structures (entire memories, multipliers, etc.) with a few lines of code
- Verilog models for memories automatically generated based on size

Clock Distribution



Clock skew

Image removed due to copyright restrictions.

For 1Ghz clock, skew budget is 100ps.

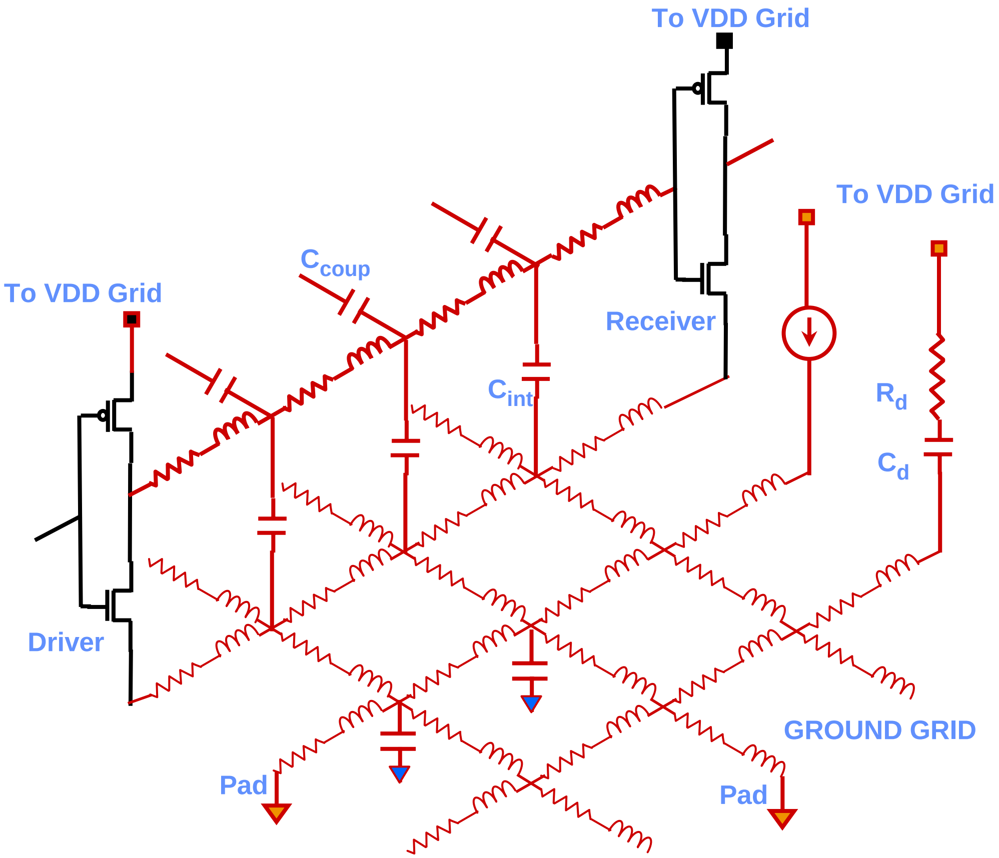
Variations along different paths arise from:

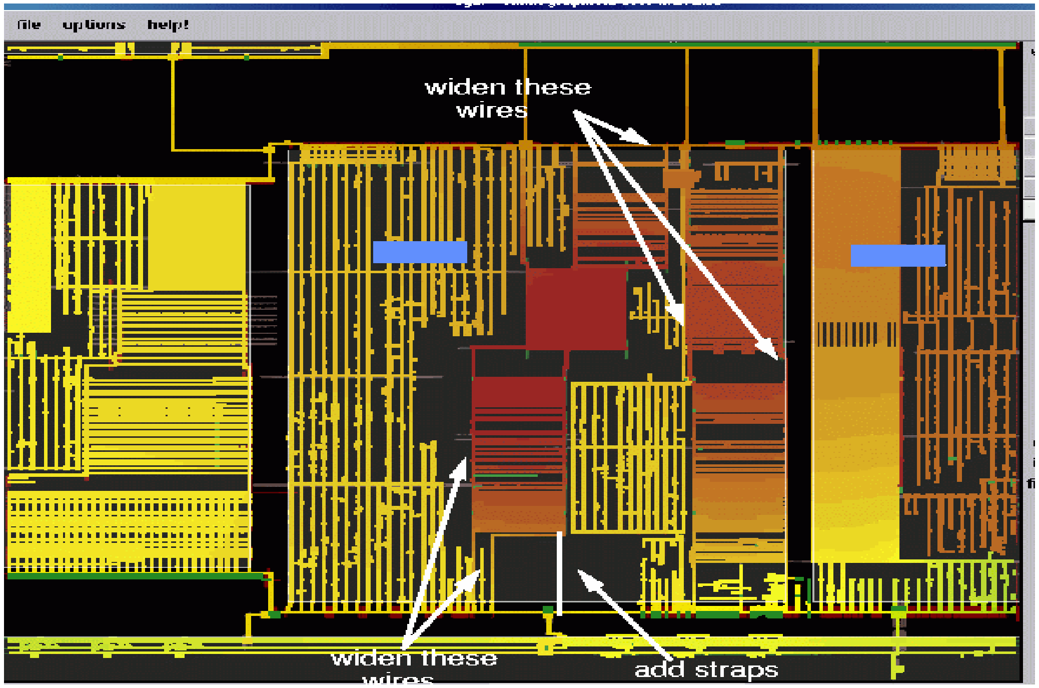
- Device: V_T, W/L, etc.
- Environment: V_{DD}, °C
- Interconnect: dielectric thickness variation

IBM Clock Routing

Image removed due to copyright restrictions.

The Power Supply Wires are Not Ideal!

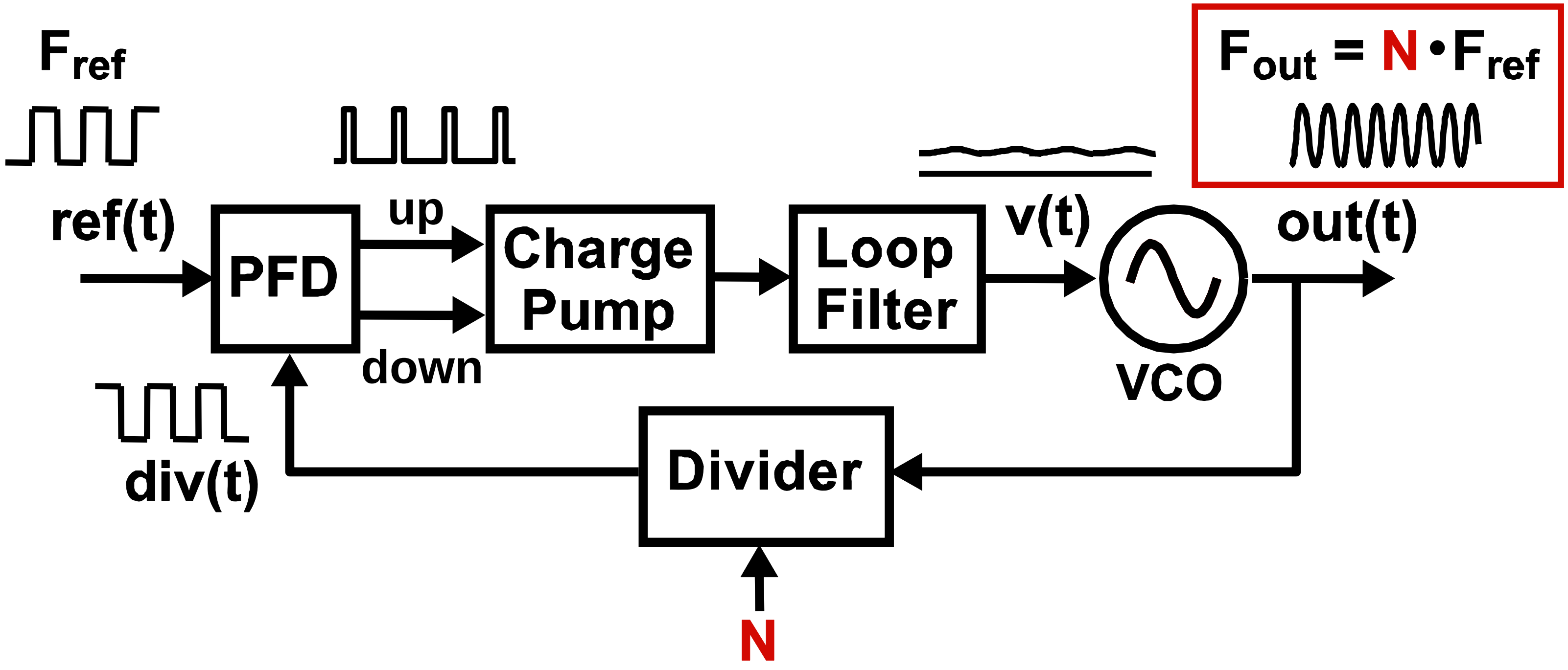




The IR-drop problem causes internal power supply voltage to be less than the external source

Used with permission.

Analog Circuits: Clock Frequency Multiplication (Phase Locked Loop)

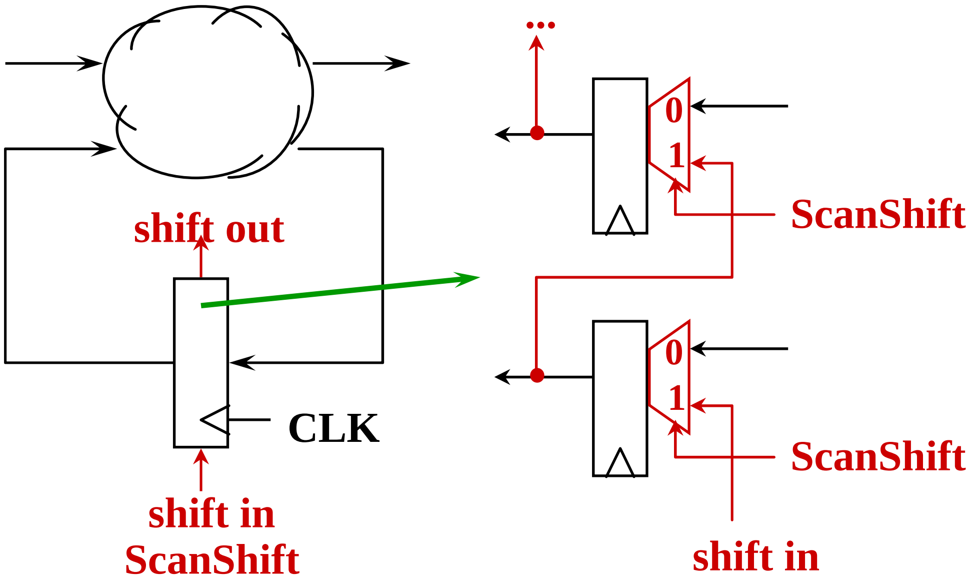


- VCO ⇨ produces high frequency square wave
- Divider ⇨ divides down VCO frequency
- PFD ⇨ compares phase of ref and div
- Loop filter ⇨ extracts phase error information

Used widely in digital systems for clock synthesis (a standard IP block in most ASIC flows)

Courtesy Michael Perrott. Used with permission.

Scan Testing



Used with permission

Idea: have a mode in which all registers are chained into one giant shift register which can be loaded/ read-out bit serially. Test remaining (combinational) logic by

(1) in “test” mode, shift in new values for all register bits thus setting up the inputs to the combinational logic

(2) clock the circuit once in “normal” mode, latching the outputs of the combinational logic back into the registers

(3) in “test” mode, shift out the values of all register bits and compare against expected results.

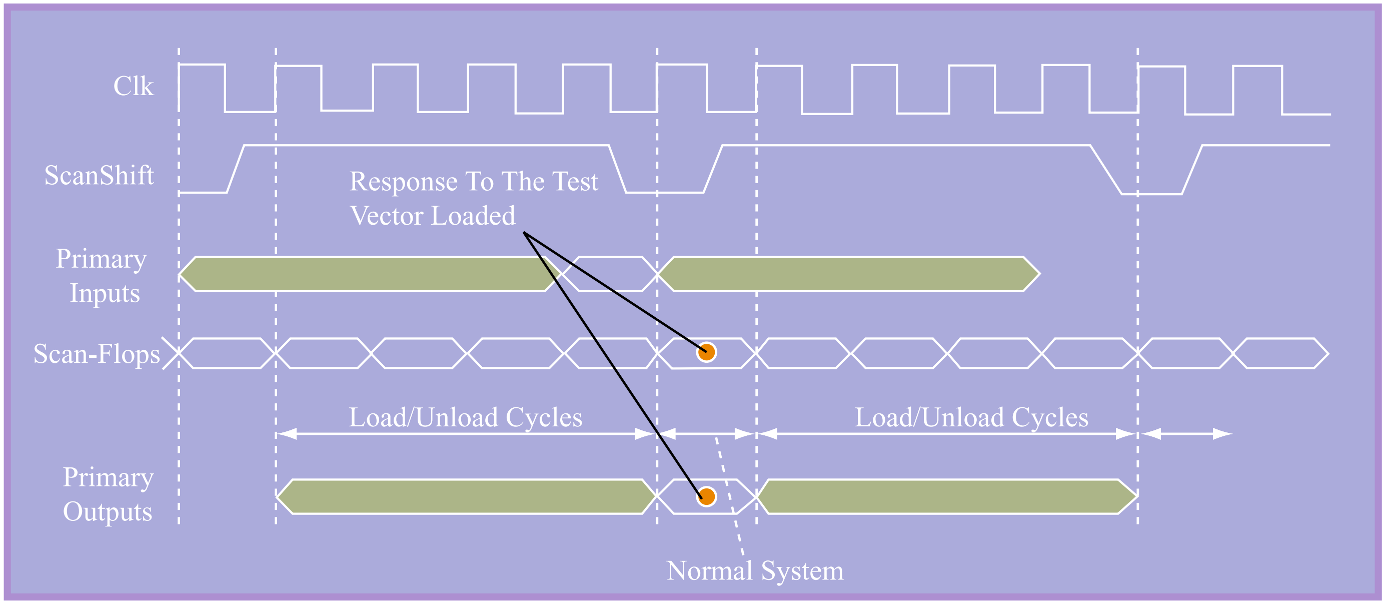


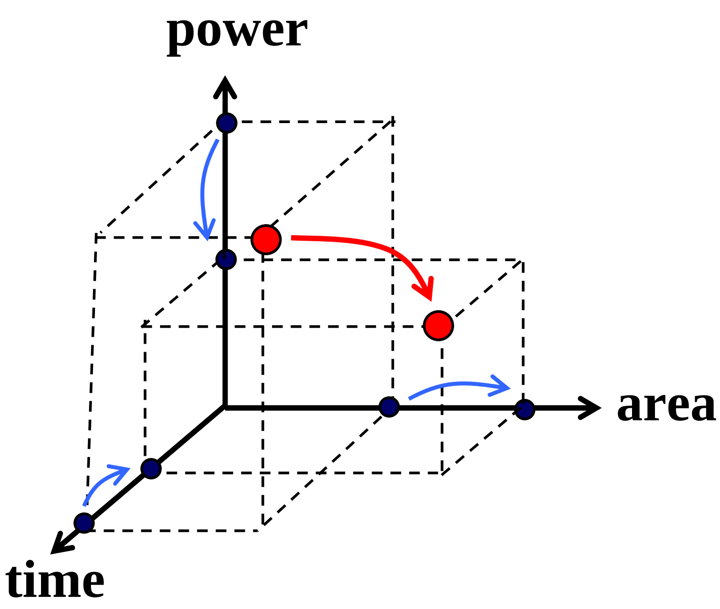
Figure by MIT OpenCourseWare.

Behavioral Transformations

- There are a large number of implementations of the same functionality
- These implementations present a different point in the area-time-power design space
- Behavioral transformations allow exploring the design space a high-level

Optimization metrics:

1. **Area** of the design
2. **Throughput** or sample time T_S
3. **Latency**: clock cycles between the input and associated output change
4. **Power** consumption
5. **Energy** of executing a task
6. …



Fixed-Coefficient Multiplication

Conventional Multiplication

$Z = X \cdot Y$

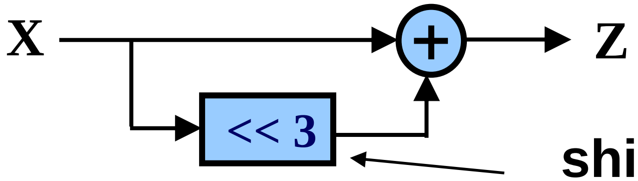
$$
\begin{array}{cccccccc}
 & & & & X_3 & X_2 & X_1 & X_0 \\
 & & & & Y_3 & Y_2 & Y_1 & Y_0 \\
\hline
 & & & & X_3 \cdot Y_0 & X_2 \cdot Y_0 & X_1 \cdot Y_0 & X_0 \cdot Y_0 \\
 & & & X_3 \cdot Y_1 & X_2 \cdot Y_1 & X_1 \cdot Y_1 & X_0 \cdot Y_1 & \\
 & & X_3 \cdot Y_2 & X_2 \cdot Y_2 & X_1 \cdot Y_2 & X_0 \cdot Y_2 & & \\
 & X_3 \cdot Y_3 & X_2 \cdot Y_3 & X_1 \cdot Y_3 & X_0 \cdot Y_3 & & & \\
\hline
Z_7 & Z_6 & Z_5 & Z_4 & Z_3 & Z_2 & Z_1 & Z_0
\end{array}
$$

Constant multiplication (become hardwired shifts and adds)

$Z = X \cdot (1001)_2$

$$
\begin{array}{cccccccc}
 & & & & X_3 & X_2 & X_1 & X_0 \\
 & & & & 1 & 0 & 0 & 1 \\
\hline
 & & & & X_3 & X_2 & X_1 & X_0 \\
 & X_3 & X_2 & X_1 & X_0 & & & \\
\hline
Z_7 & Z_6 & Z_5 & Z_4 & Z_3 & Z_2 & Z_1 & Z_0
\end{array}
$$

$Y = (1001)_2 = 2^3 + 2^0$



shifts using wiring

Transform: Canonical Signed Digits (CSD)

Canonical signed digit representation is used to increase the number of zeros. It uses digits {-1, 0, 1} instead of only {0, 1}.

Iterative encoding: replace string of consecutive 1's

0	1	1	…	1	1

➡

1	0	0	…	0	-1

$2^{N-2} + \ldots + 2^1 + 2^0$ $\quad 2^{N-1} - 2^0$

Worst case CSD has 50% non zero bits

01101111

0	1	1	0	1	1	1	1

➡

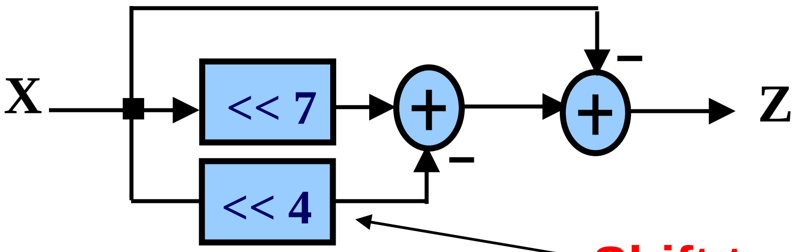
0	1	1	1	0	0	0	-1

⬇

=

$100\bar{1}000\bar{1}$

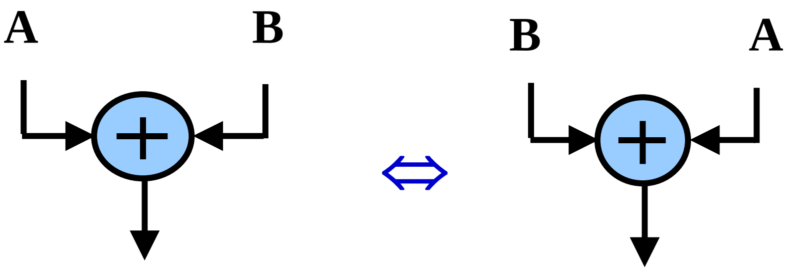
1	0	0	-1	0	0	0	-1



Shift translates to re-wiring

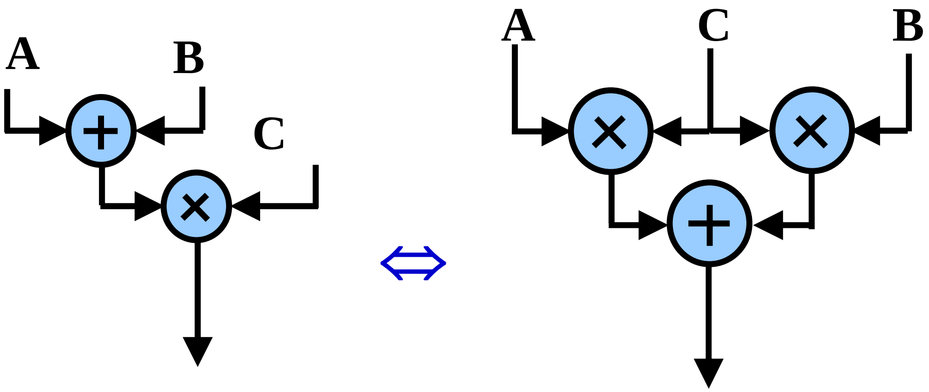
Algebraic Transformations

Commutativity



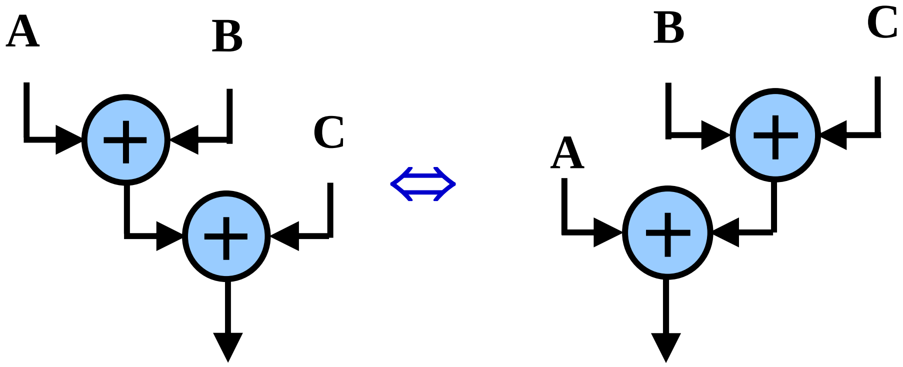
A + B = B + A

Distributivity



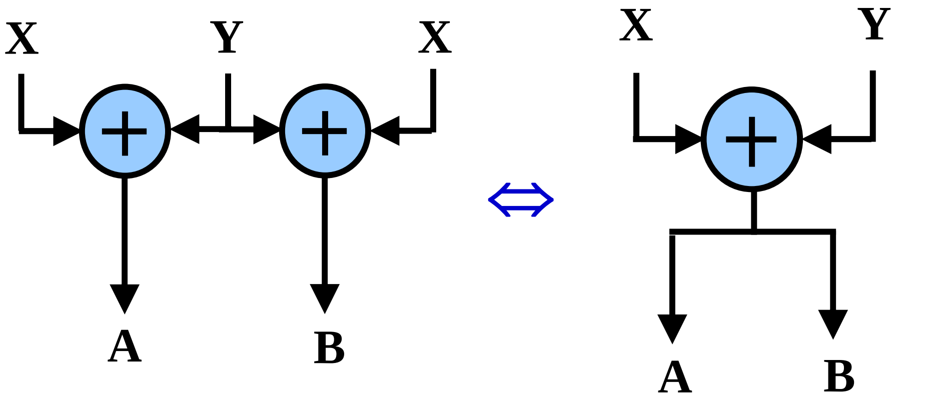
(A + B) C = AB + BC

Associativity

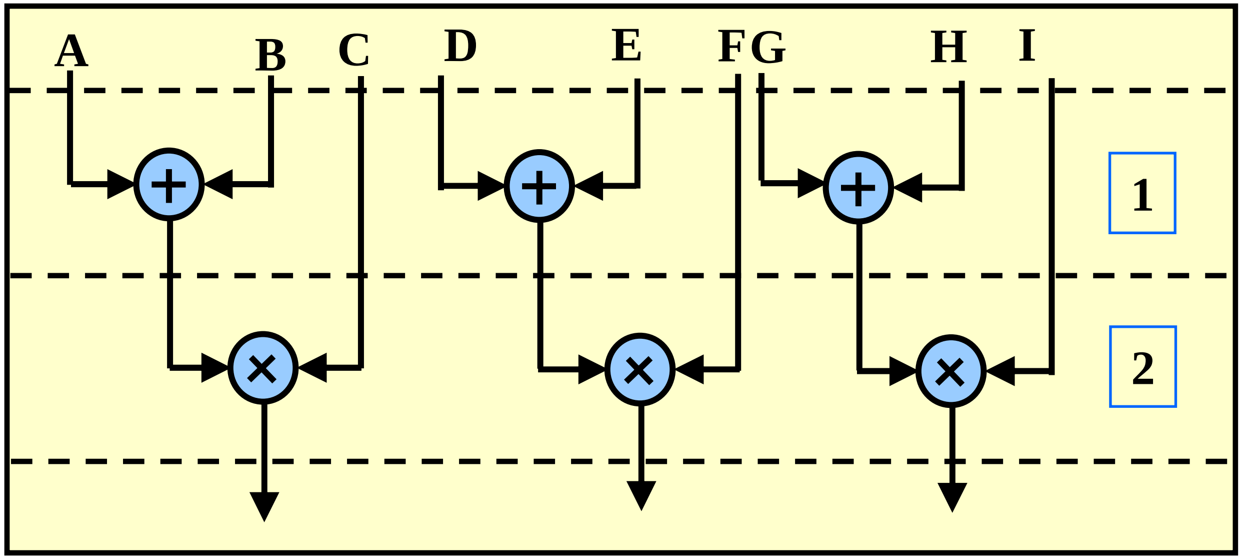


(A + B) + C = A + (B+C)

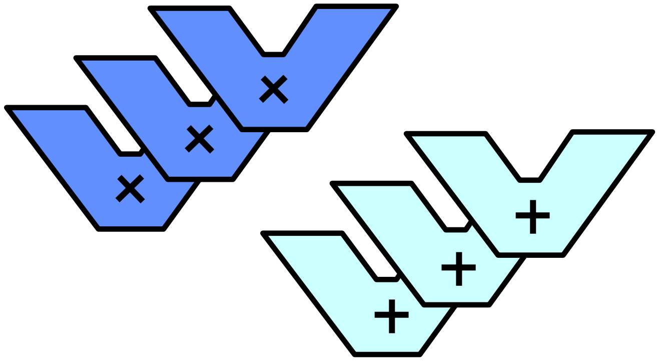
Common sub-expressions



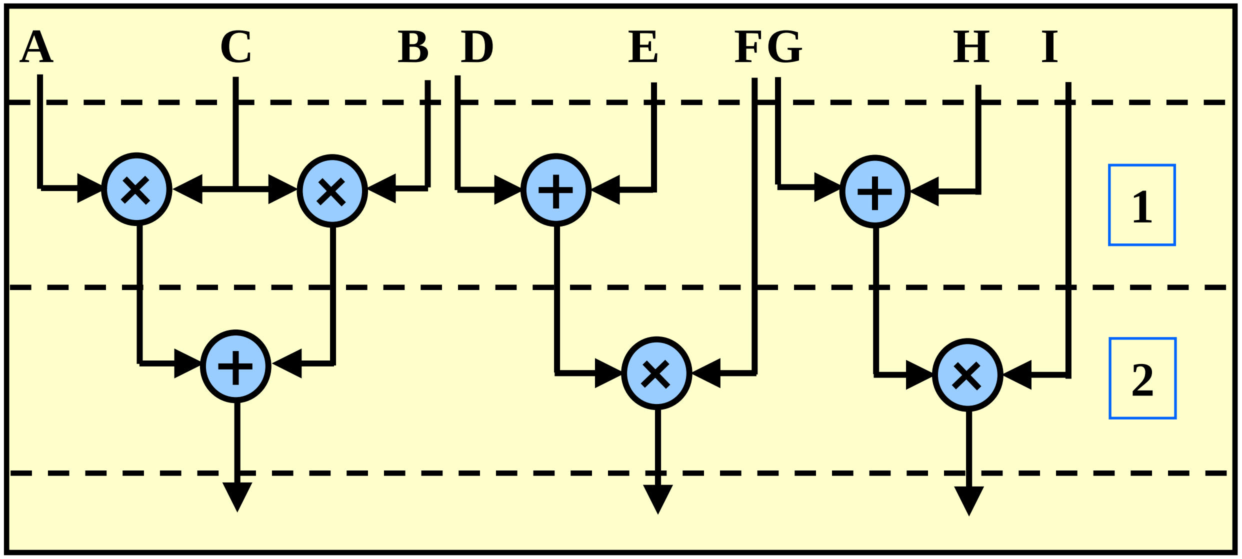
Transforms for Efficient Resource Utilization



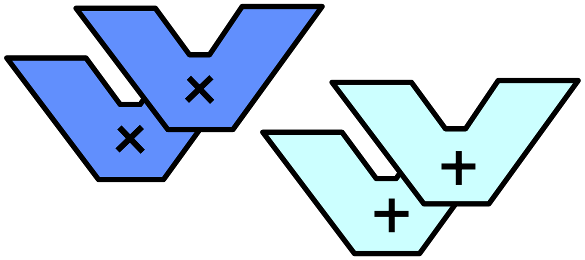
Time multiplexing: mapped to 3 multipliers and 3 adders



distributivity



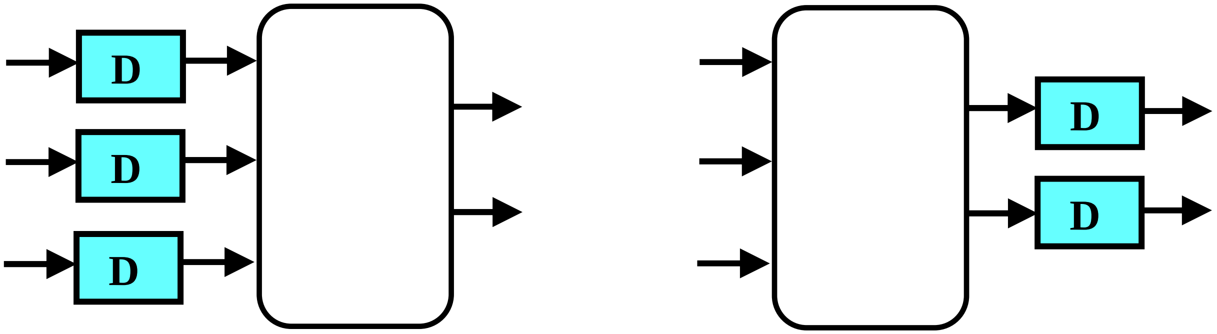
Reduce number of operators to 2 multipliers and 2 adders



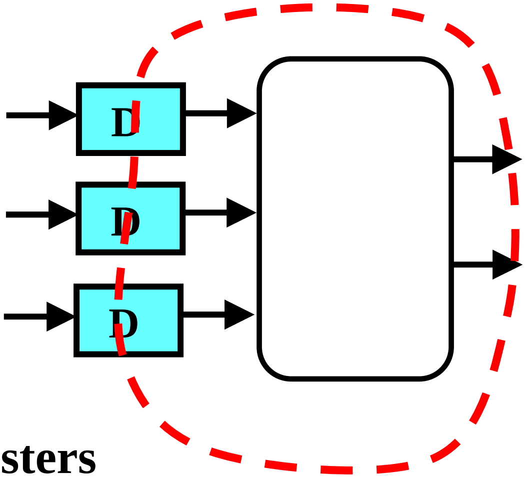
A Very Useful Transform: Retiming

Retiming is the action of moving delay around in the systems

- **Delays have to be moved from ALL inputs to ALL outputs or vice versa**

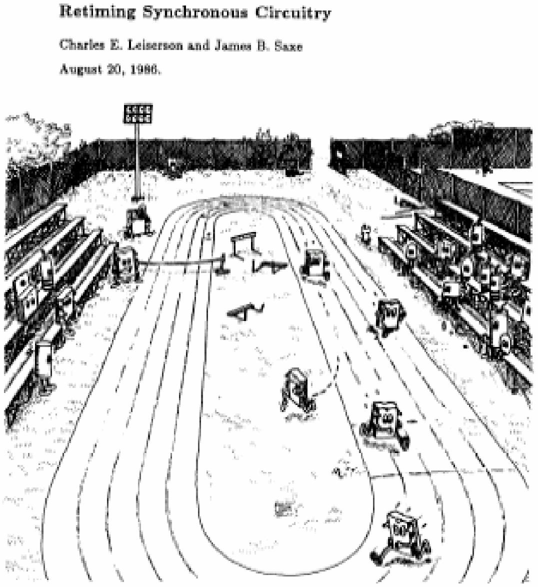


Cutset retiming: **A cutset intersects the edges, such that this would result in two disjoint partitions of these edges being cut. To retime, delays are moved from the ingoing to the outgoing edges or vice versa.**



Benefits of retiming:

- **Modify critical path delay**
- **Reduce total number of registers**



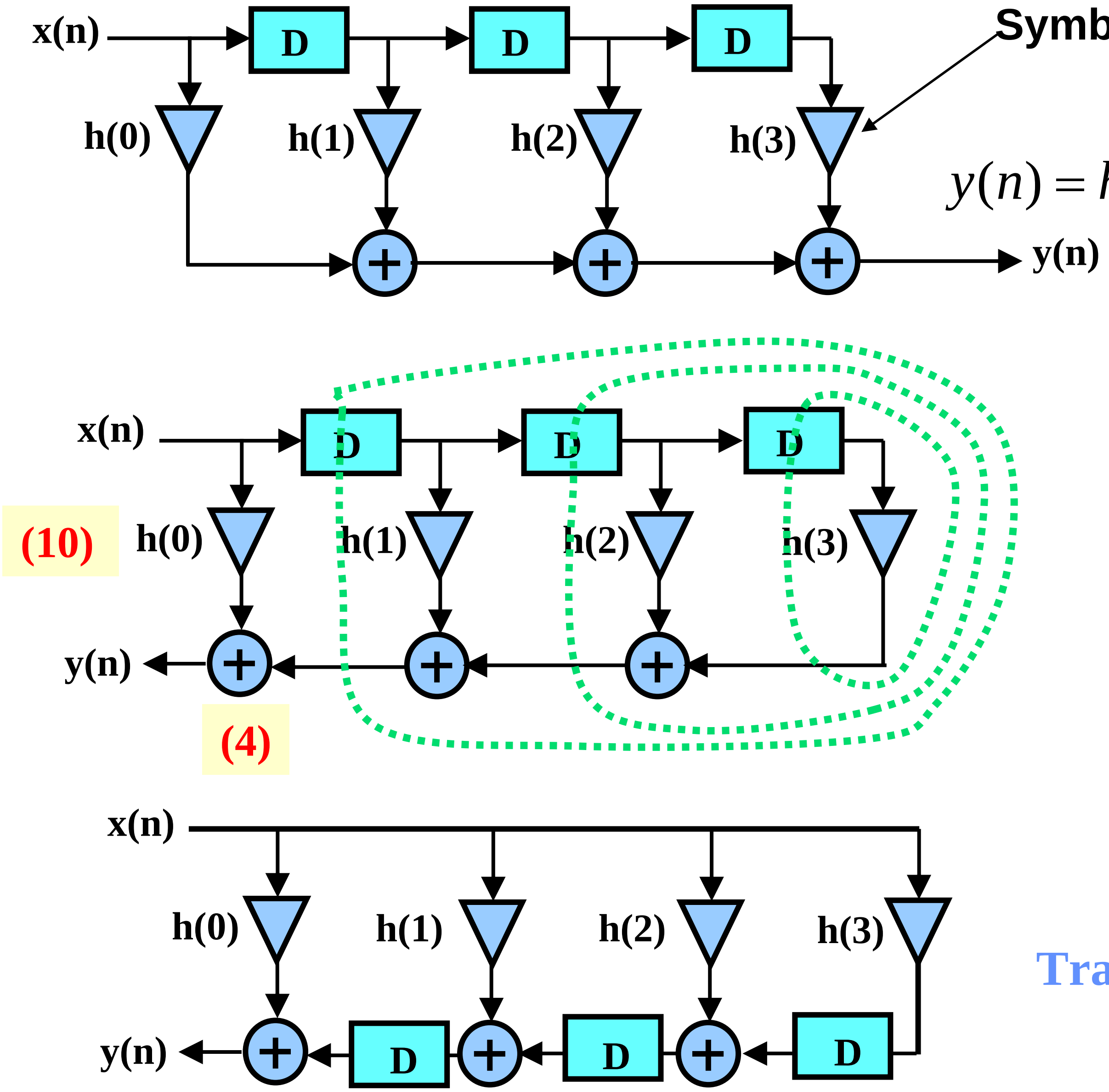
Retiming Synchronous Circuitry

Charles E. Leiserson and James B. Saxe

August 20, 1986.

Courtesy of Prof. Charles E. Leiserson.

Retiming Example: FIR Filter



Symbol for multiplication

Direct form

$$y(n) = h(n) \otimes x(n) = \sum_{i=0}^{K} x(n-i) \cdot h(i)$$

associativity of the addition

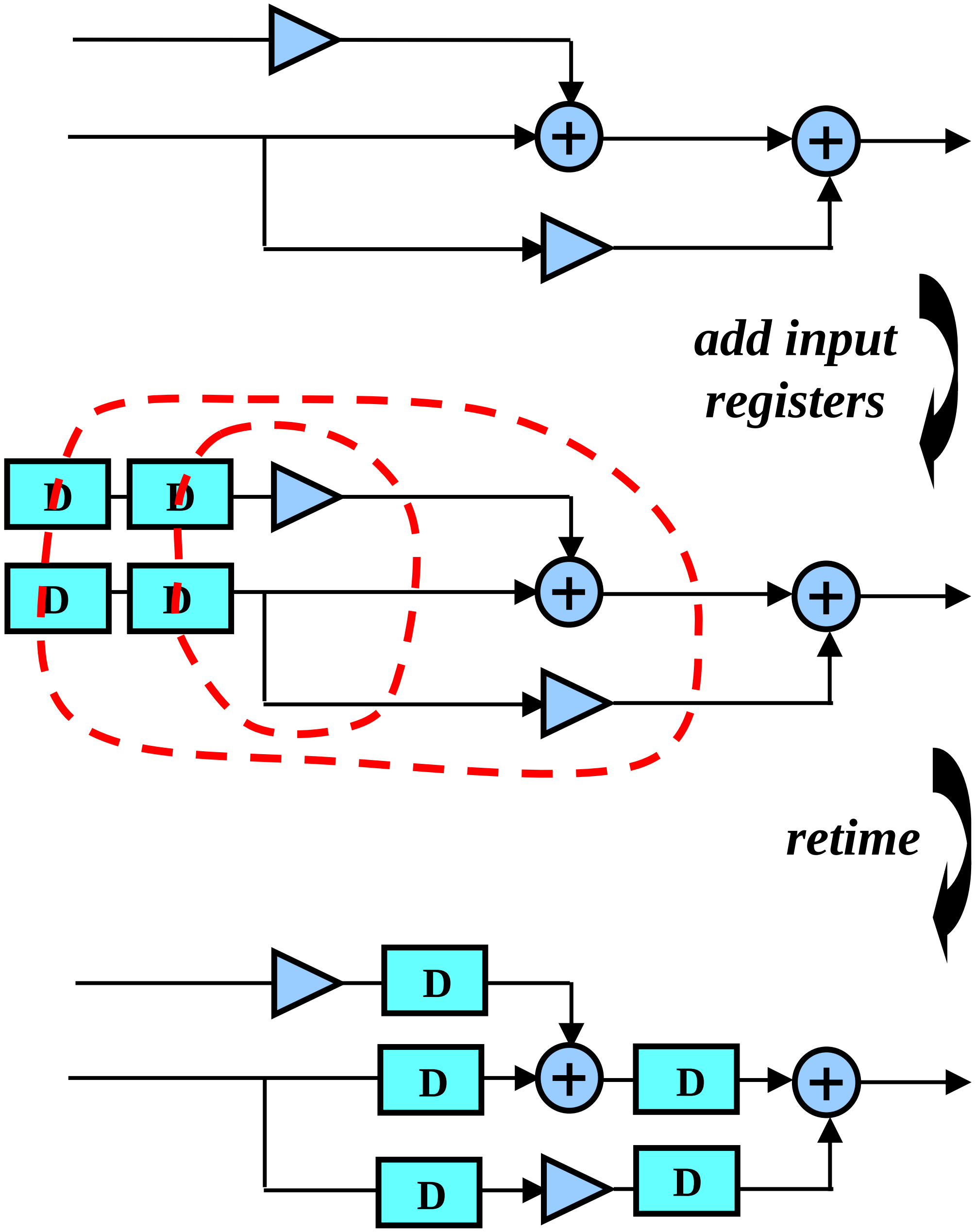
T_{clk} = 22 ns

retime

Transposed form T_{clk} = 14 ns

Note: here we use a first cut analysis that assumes the delay of a chain of operators is the sum of their individual delays. This is not accurate.

Pipelining, Just Another Transformation (Pipelining = Adding Delays + Retiming)



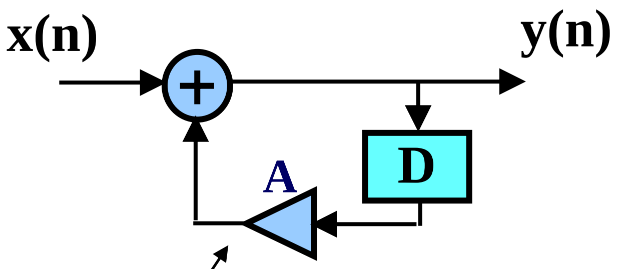
Contrary to retiming, pipelining adds extra registers to the system

How to pipeline:

1. Add extra registers at *all* inputs
2. Retime

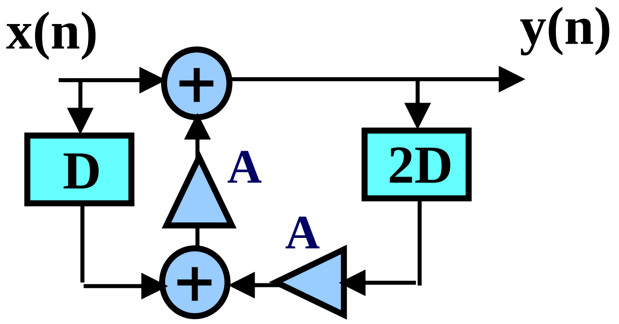
The Power of Transforms: Lookahead

$y(n) = x(n) + A\ y(n-1)$



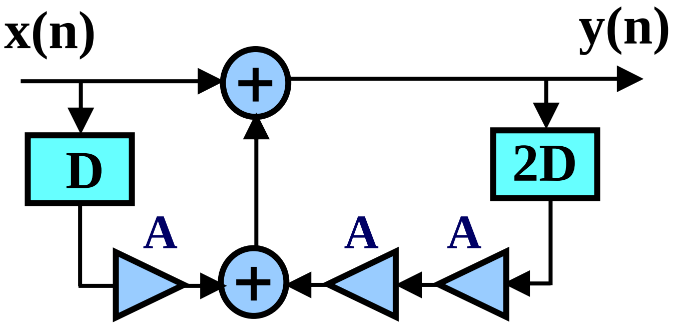
Try pipelining this structure

loop unrolling



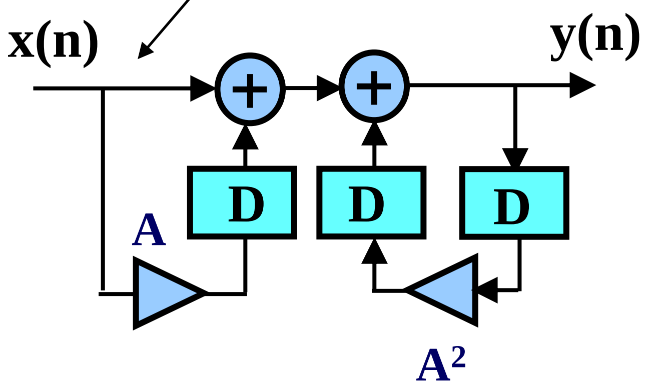
$y(n) = x(n) + A[x(n-1) + A\ y(n-2)]$

distributivity

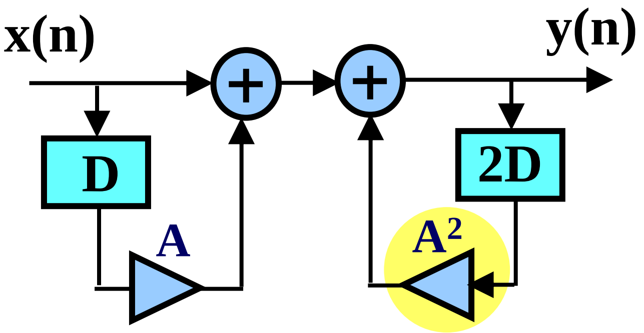


How about pipelining this structure!

associativity

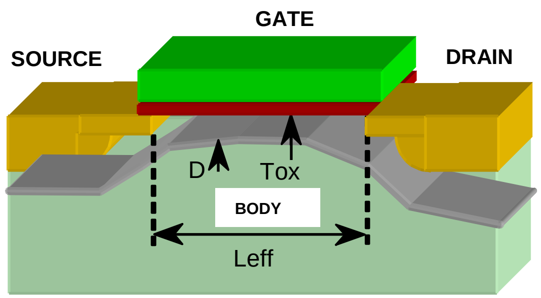


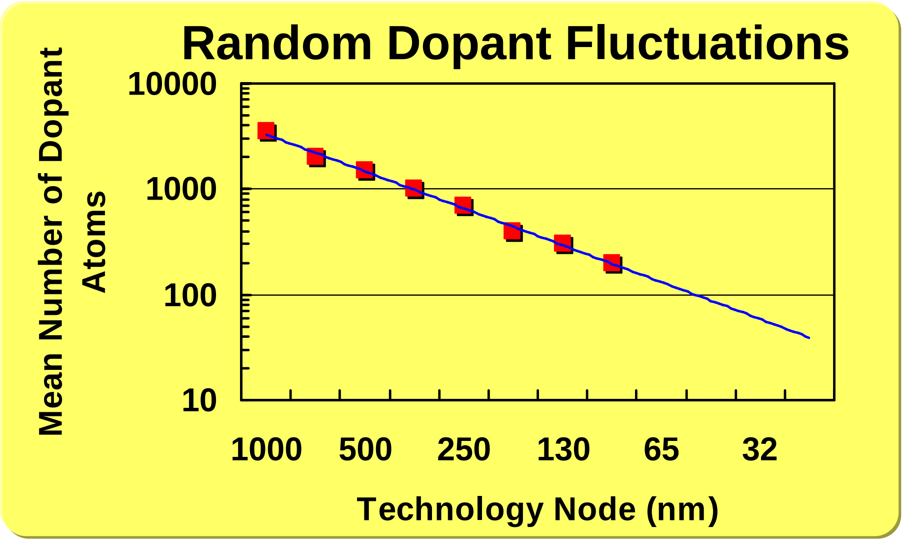
retiming

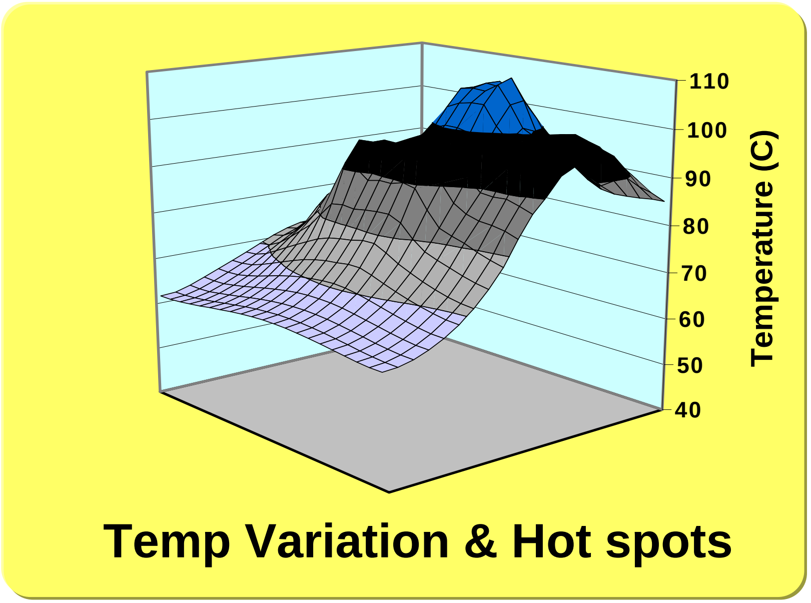


precomputed

Key Concern in Modern VLSI: Variations!

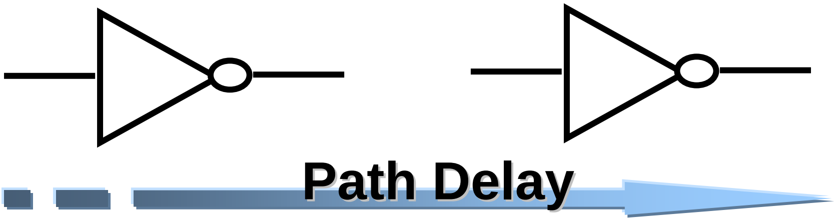


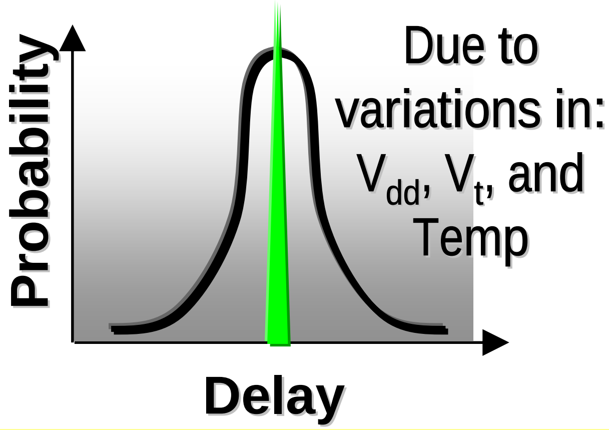




Temp Variation & Hot spots

With 100b transistors,
1b unusable (variations)



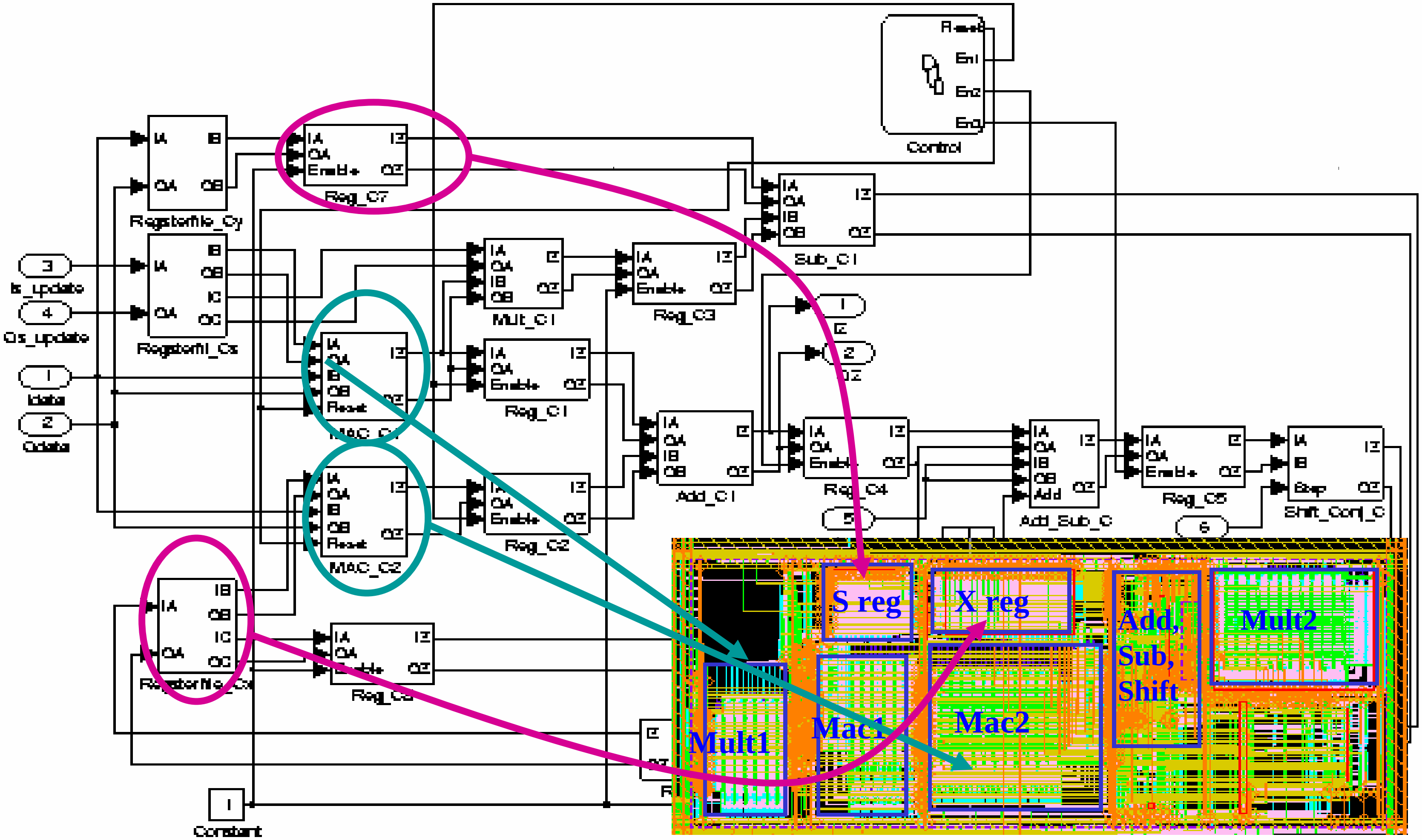


Due to variations in: V_{dd}, V_t, and Temp

Deterministic design techniques inadequate in the future

Courtesy of Shekhar Y. Borkar (Intel). Used with permission.

Trends: “Chip in a Day” (Matlab/Simulink to Silicon…)



Map algorithms directly to silicon - bypass writing Verilog!

(Courtesy of R. Brodersen. Used with permission.)

Trends: Watermarking of Digital Designs

Fingerprinting is a technique to deter people from illegally redistributing legally obtained IP by enabling the author of the IP to uniquely identify the original buyer of the resold copy.

The essence of the **watermarking** approach is to encode the author's signature. The selection, encoding, and embedding of the signature must result in minimal performance and storage overhead.

Image removed due to copyright restrictions.

Image removed due to copyright restrictions.