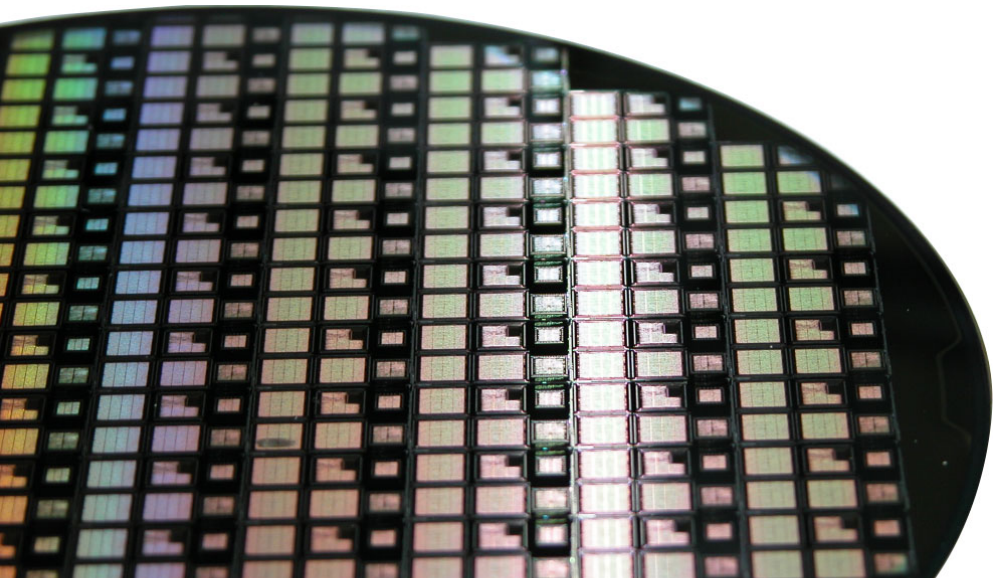
Understanding performance numbers in Integrated Circuit Design

Oprecomp summer school 2019, Perugia Italy

5 September 2019

Frank K. Gürkaynak
kgf@ee.ethz.ch



ETH zürich
Integrated Systems Laboratory

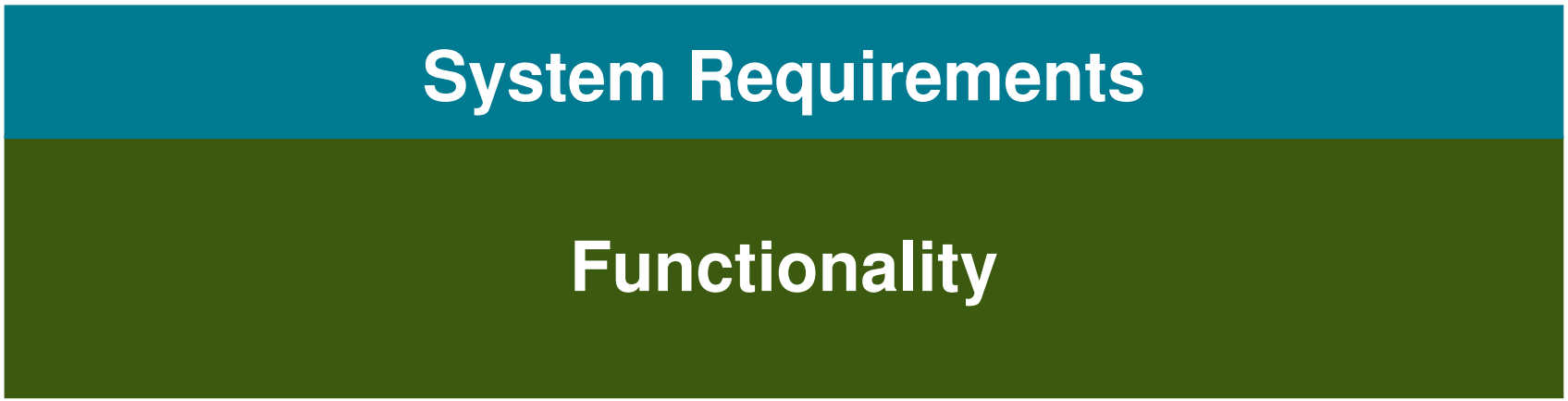
Who Am I?

- Born in Istanbul, Turkey
- Studied and worked at:
  - Istanbul Technical University, Istanbul, Turkey
  - EPFL, Lausanne, Switzerland
  - Worcester Polytechnic Institute, Worcester MA, USA
- Since 2008: Integrated Systems Laboratory, ETH Zurich
  - Director, Microelectronics Design Center
  - Senior Scientist, group of Prof. Luca Benini
- Interests:
  - Digital Integrated Circuits
  - Cryptographic Hardware Design
  - Design Flows for Digital Design
  - Processor Design
  - Open Source Hardware

What Will We Discuss Today?

- **Introduction**
  Cost Structure of Integrated Circuits (ICs)
- **Measuring performance of ICs**
  Why is it difficult? EDA tools should give us a number
- **Area**
  How do people report area? Is that fair?
- **Speed**
  How fast does my circuit actually work?
- **Power**
  These days much more important, but also much harder to get right

System Design Requirements

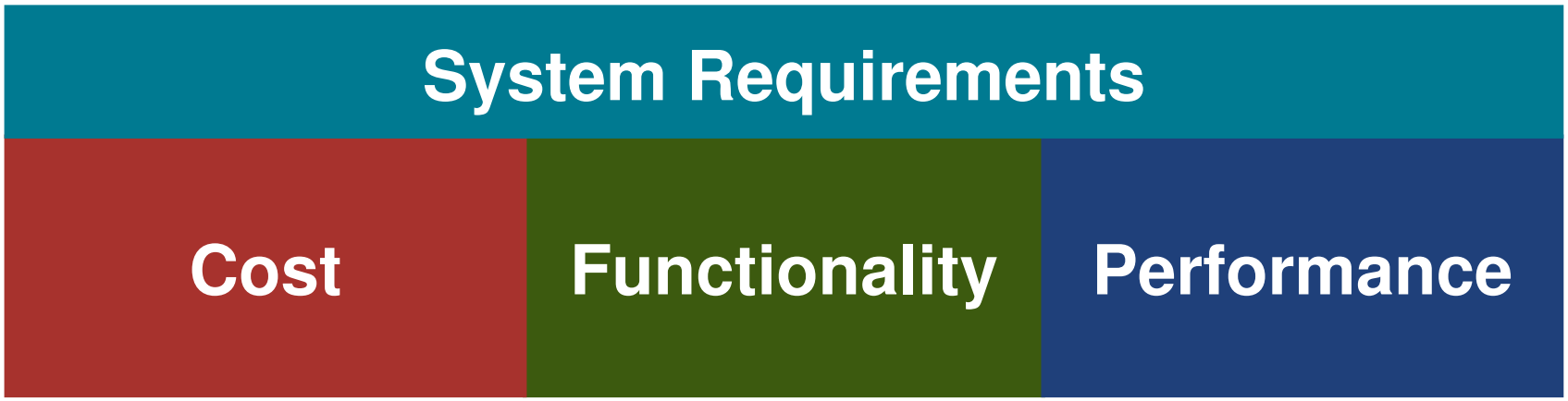


- **Functionality** determines what the system will do

System Design Requirements

- **Functionality** determines what the system will do
- The **performance** establishes the solution space

System Design Requirements



- **Functionality** determines what the system will do
- The **performance** establishes the solution space
- Finally the **cost** sets a limit to what is possible

The Focus of this Talk is on Digital Systems

- **Custom Integrated Circuits, ASICs**
  - ✓ Best performance, cheap for mass market
  - ✗ Difficult to design, long manufacturing time

The Focus of this Talk is on Digital Systems

- **Custom Integrated Circuits, ASICs**
  - ✓ Best performance, cheap for mass market
  - ✗ Difficult to design, long manufacturing time
- **Programmable Logic, FPGAs**
  - ✓ Very flexible, simpler design flow than ASICs, many IPs
  - ✗ High unit cost, not as fast as ASICs

The Focus of this Talk is on Digital Systems

- **Custom Integrated Circuits, ASICs**
  - ✓ Best performance, cheap for mass market
  - ✗ Difficult to design, long manufacturing time
- **Programmable Logic, FPGAs**
  - ✓ Very flexible, simpler design flow than ASICs, many IPs
  - ✗ High unit cost, not as fast as ASICs
- **Processors**
  - **Microcontrollers**
    - ✓ Relatively easy to program, includes peripherals, cheap
    - ✗ Poor performance

The Focus of this Talk is on Digital Systems

- **Custom Integrated Circuits, ASICs**
  - ✓ Best performance, cheap for mass market
  - ✗ Difficult to design, long manufacturing time
- **Programmable Logic, FPGAs**
  - ✓ Very flexible, simpler design flow than ASICs, many IPs
  - ✗ High unit cost, not as fast as ASICs
- **Processors**
  - **Microcontrollers**
    - ✓ Relatively easy to program, includes peripherals, cheap
    - ✗ Poor performance
  - **General Purpose Processors**
    - ✓ Simple to program, no need to know hardware design
    - ✗ Sequential processing, needs many peripherals, high cost

The Focus of this Talk is on Digital Systems

- **Custom Integrated Circuits, ASICs**
  - ✓ Best performance, cheap for mass market
  - ✗ Difficult to design, long manufacturing time
- **Programmable Logic, FPGAs**
  - ✓ Very flexible, simpler design flow than ASICs, many IPs
  - ✗ High unit cost, not as fast as ASICs
- **Processors**
  - **Microcontrollers**
    - ✓ Relatively easy to program, includes peripherals, cheap
    - ✗ Poor performance
  - **General Purpose Processors**
    - ✓ Simple to program, no need to know hardware design
    - ✗ Sequential processing, needs many peripherals, high cost
  - **Parallel/Graphics Processors**
    - ✓ Allow parallel processing
    - ✗ High cost, no peripherals, not so easy to program

Digital Solutions are Cost Efficient

For example:

Main motivation to move from VHS to DVD was **cost!**

How Much Does a Chip Cost?

Non-recurring costs

- **Engineering**
  Designers are not cheap (app. 100 k$/yr)
- **IP costs**
  Some of the IP are bought
- **EDA tools + Infrastructure**
  EDA tools are expensive, i.e. 100 k$/yr/seat
- **Masks for production**
  Chips may need 20-50 masks. Technology dependent, may cost 50 k$-5,000 k$

How Much Does a Chip Cost?

Non-recurring costs

- **Engineering**
  Designers are not cheap (app. 100 k$/yr)
- **IP costs**
  Some of the IP are bought
- **EDA tools + Infrastructure**
  EDA tools are expensive, i.e. 100 k$/yr/seat
- **Masks for production**
  Chips may need 20-50 masks. Technology dependent, may cost 50 k$-5,000 k$

Recurring costs (per chip)

- **Silicon cost**
  Cost of wafer + processing
- **Licensing cost**
  Some of the IP require a licensing cost per chip sold
- **Packaging**
  The plastic package that holds the chip + costs of bonding
- **Testing**
  Each manufactured chip needs to be tested. Costs both money and time

A Simplified Example

A 4 mm × 4 mm chip in 90 nm technology, 300 mm wafers

- **Silicon Cost**
  - About 4,000 dies per wafer
  - At 85 % yield, around 3400 working dies
  - About 2,500 $ per wafer (including processing)
  - **around 0.75 $** per working die

Adapted from H. Kaeslin "*Digital Integrated Circuit Design*", Cambridge University Press, 2008

A Simplified Example

A 4 mm × 4 mm chip in 90 nm technology, 300 mm wafers

- **Silicon Cost**
  - About 4,000 dies per wafer
  - At 85 % yield, around 3400 working dies
  - About 2,500 $ per wafer (including processing)
  - **around 0.75 $** per working die
- **Packaging + Licensing + Testing**
  - Design dependent from a few cents, to several of dollars (**1.00 $**)

Adapted from H. Kaeslin "*Digital Integrated Circuit Design*", Cambridge University Press, 2008

A Simplified Example

A 4 mm × 4 mm chip in 90 nm technology, 300 mm wafers

- **Silicon Cost**
  - About 4,000 dies per wafer
  - At 85 % yield, around 3400 working dies
  - About 2,500 $ per wafer (including processing)
  - **around 0.75 $** per working die
- **Packaging + Licensing + Testing**
  - Design dependent from a few cents, to several of dollars (**1.00 $**)
- **Non-recurring costs**
  - 200 MM of engineering, around 2 M$
  - Production mask set around 1 M$
  - IPs + EDA tools + administration + infrastructure another 1 M$
  - Volume dependent. i.e for 5,000,000 chips sold, **around 0.80 $** per chip

Adapted from H. Kaeslin "*Digital Integrated Circuit Design*", Cambridge University Press, 2008

The key to IC Design success is volume

Previous example, cost of one chip

- **If we sell 5,000,000 chips**
  - 4 M$ non-recurring costs, 0.80 $ per chip
  - Total chip costs 2.55 $ (31% non-recurring)

Adapted from H. Kaeslin "*Digital Integrated Circuit Design*", Cambridge University Press, 2008

The key to IC Design success is volume

Previous example, cost of one chip

- **If we sell 5,000,000 chips**
  - 4 M$ non-recurring costs, 0.80 $ per chip
  - Total chip costs 2.55 $ (31% non-recurring)
- **If we sell 5'000 chips**
  - 4 M$ non-recurring costs, 800 $ per chip
  - Total chip costs 801.75 $ (99.78% non-recurring)

Adapted from H. Kaeslin "*Digital Integrated Circuit Design*", Cambridge University Press, 2008

The key to IC Design success is volume

Previous example, cost of one chip

- **If we sell 5,000,000 chips**
  - 4 M$ non-recurring costs, 0.80 $ per chip
  - Total chip costs 2.55 $ (31% non-recurring)
- **If we sell 5'000 chips**
  - 4 M$ non-recurring costs, 800 $ per chip
  - Total chip costs 801.75 $ (99.78% non-recurring)
- **If we sell 500'000'000 chips**
  - 4 M$ non-recurring costs, 0.008 $ per chip
  - Total chip costs 1.76 $ (0.5% non-recurring)

Adapted from H. Kaeslin "*Digital Integrated Circuit Design*", Cambridge University Press, 2008

How Can We Make Chips Cheaper

Non-recurring costs

Sell more chips!

- **Engineering**
  Buy IP (**increases IP costs**)
- **IP costs**
  Redesign it yourself (**increases engineering costs**)
- **EDA tools + Infrastructure**
  Outsource to design houses
- **Masks for production**
  Use fewer masks. Use older (cheaper) technologies

How Can We Make Chips Cheaper

Non-recurring costs

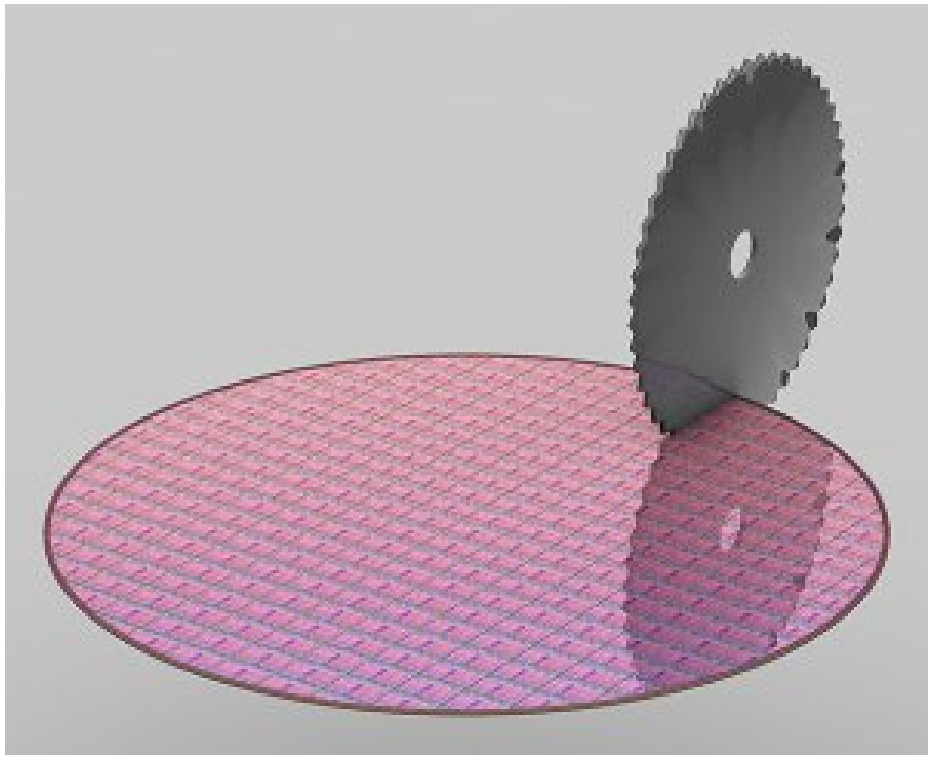
Sell more chips!

- **Engineering**
  Buy IP (**increases IP costs**)
- **IP costs**
  Redesign it yourself (**increases engineering costs**)
- **EDA tools + Infrastructure**
  Outsource to design houses
- **Masks for production**
  Use fewer masks. Use older (cheaper) technologies

Recurring costs (per chip)

- **Silicon cost**
  Reduce die area
- **Licensing cost**
  Do not buy IP (**increases engineering cost**)
- **Packaging**
  Simpler packages, reduce number of I/Os, thermal requirements
- **Testing**
  Reduce testing time, improve DFT.

How are Individual Dies Cut From a Wafer?

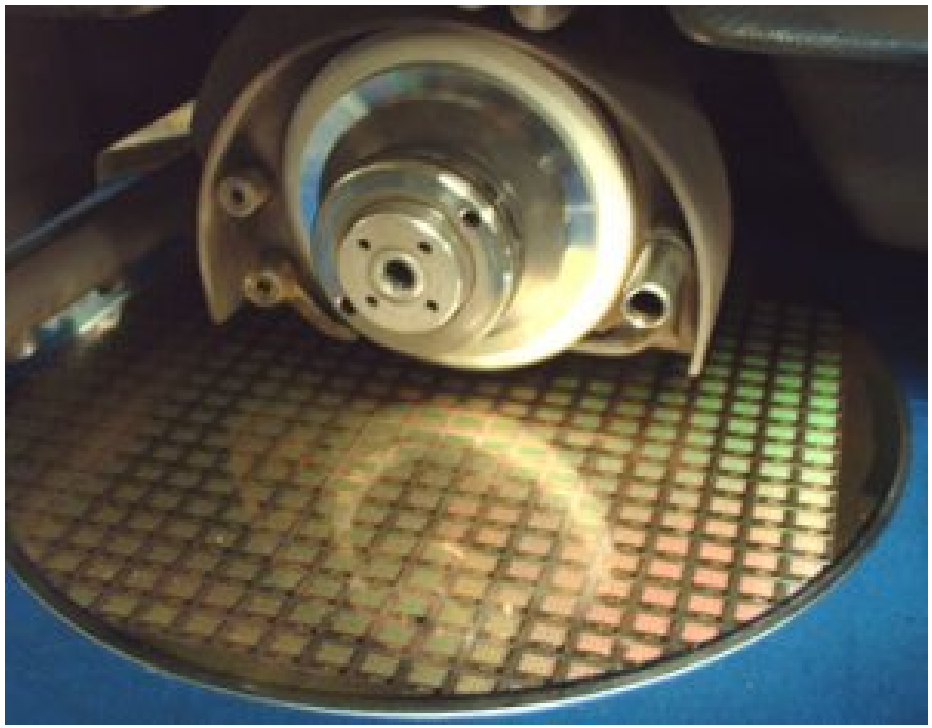


Dicing is a mechanical process

There needs to be some space (typically 50-200 µm) between dies on a wafer so that we can reliably cut them.

Images taken from `http://wonderfulworldofwafers.wordpress.com/`

How are Individual Dies Cut From a Wafer?

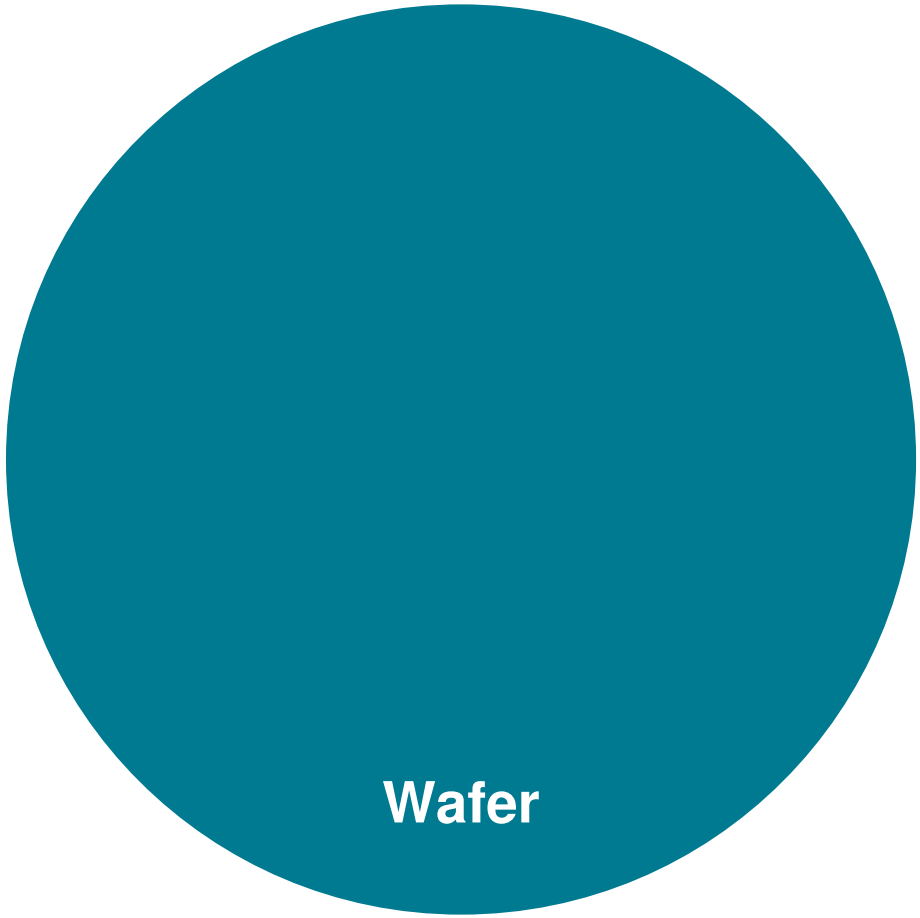


The sawline sets a lower limit for the practical die size

If the sawline is 100 µm, and the die is 1 mm×1 mm in size:
more than **20 %** of the wafer will be **wasted** for sawlines

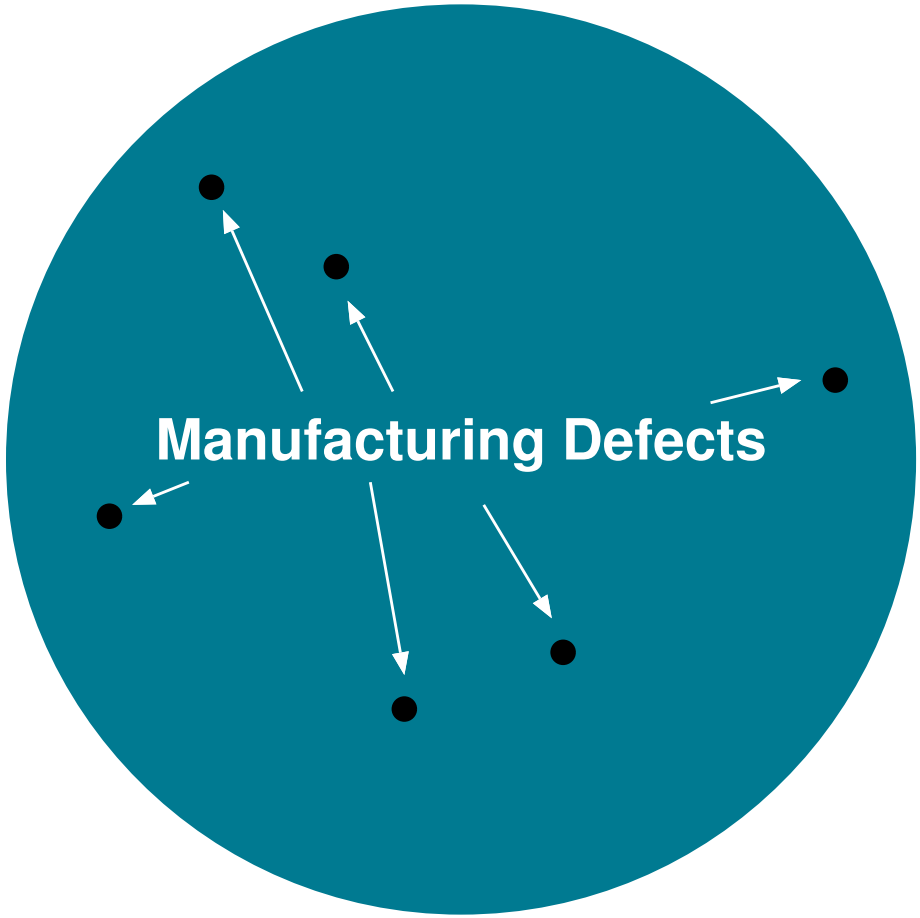
Images taken from `http://wonderfulworldofwafers.wordpress.com/`

The Story of Yield



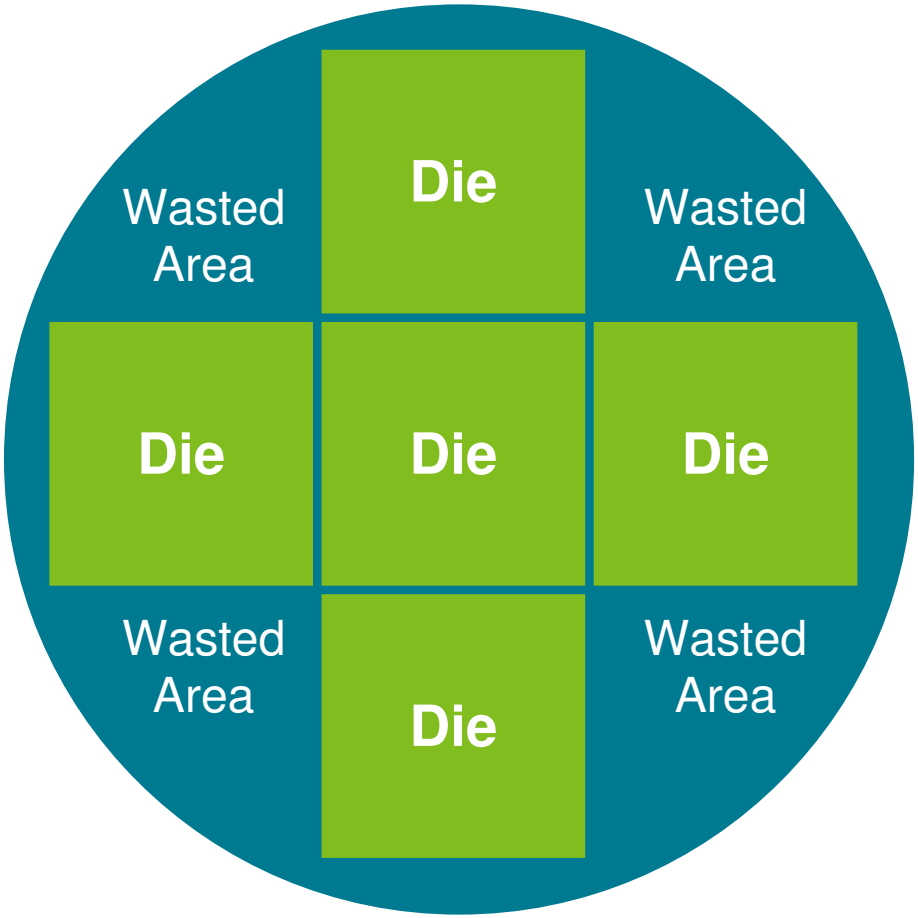
Single wafer during production

The Story of Yield



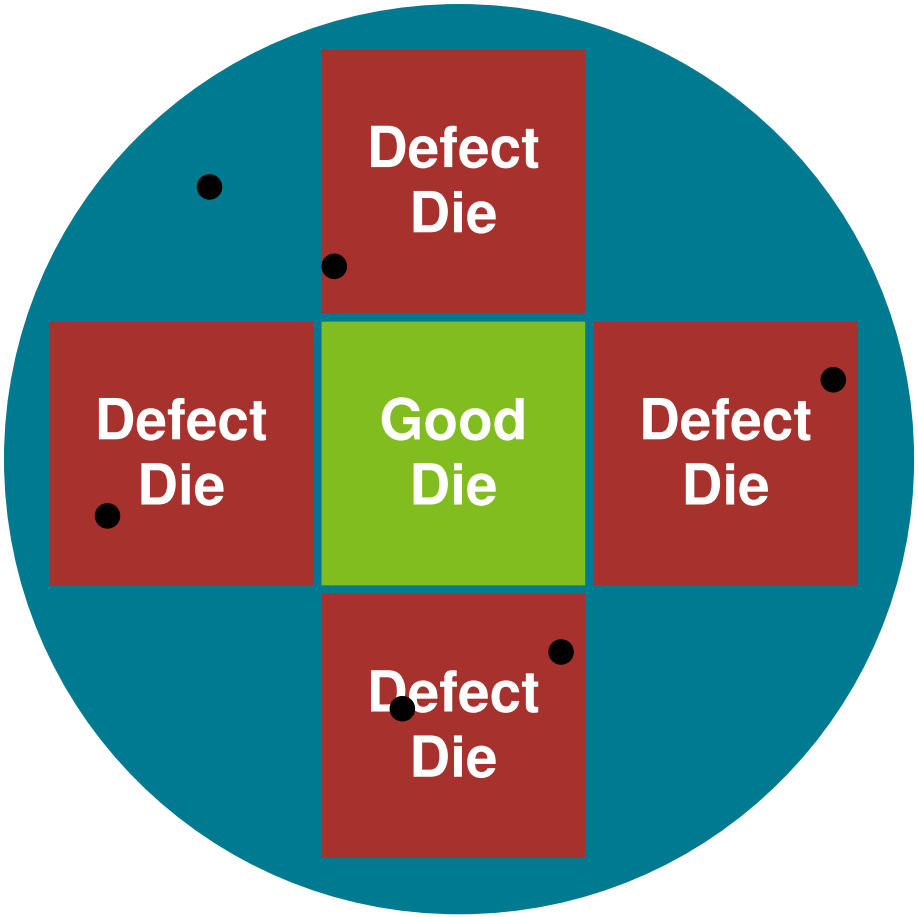
Inevitably there will be some **defects** on the wafer

The Story of Yield



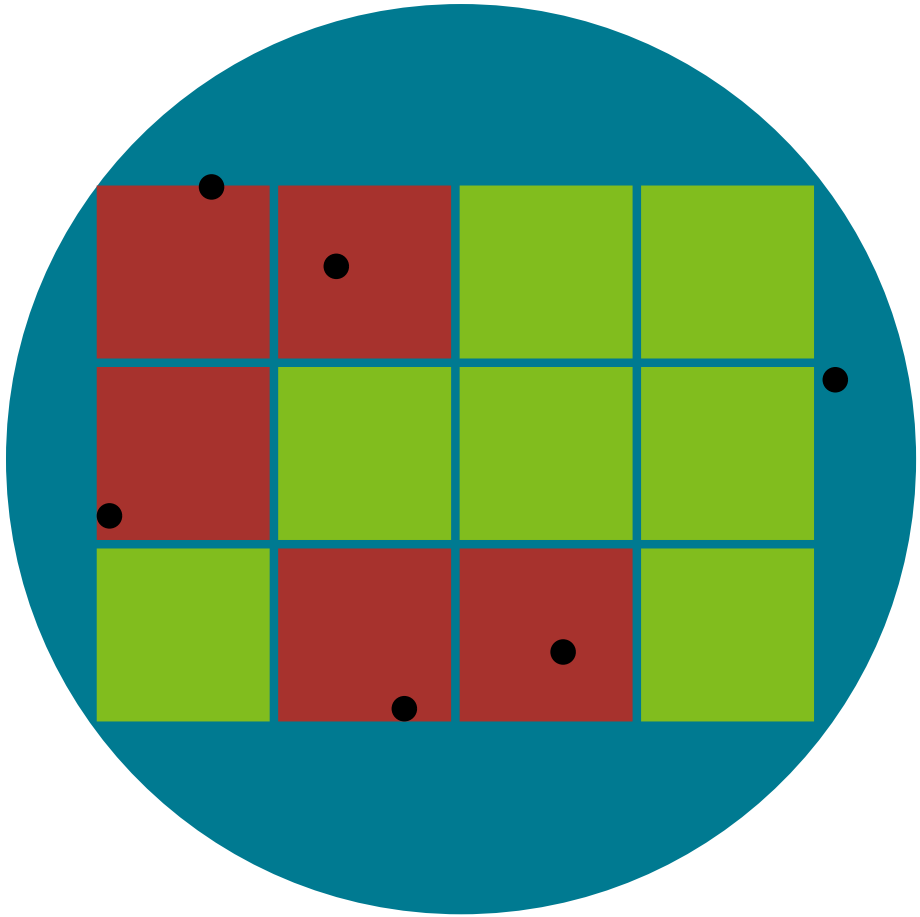
If a large die is used, only few dies can be manufactured per each wafer

The Story of Yield



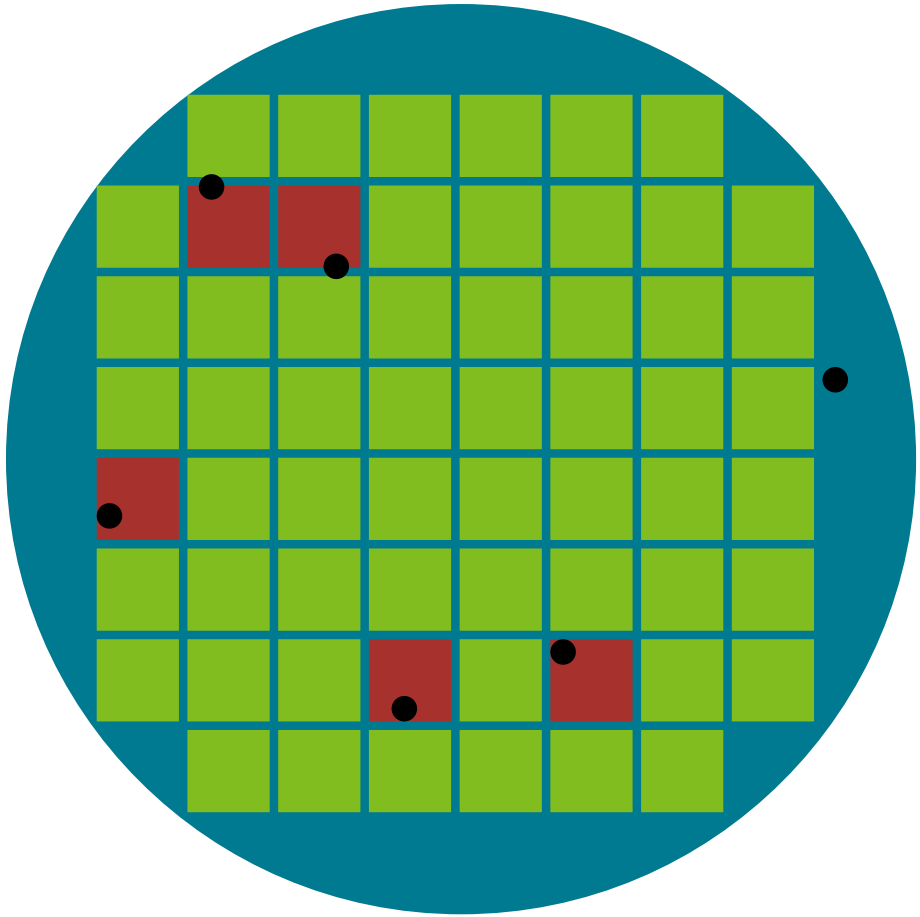
In this example **4** out of 5 dies are defect. Yield $= 1/5 =$ **20 %**

The Story of Yield



Smaller dies improve yield: Yield $= 7/12 =$ **58 %**

The Story of Yield



The smaller the dies, the better the yield: Yield $= 5/60 =$ **92 %**

You Need To Test Every Chip You Manufacture

A certain amount of chips will always be defective

- Keeps production costs down. 100 % yield **economically not feasible**
- Yield typically improves over time for a specific process
  - Cutting edge technologies may have as low as 10 % yield
  - Established technologies have 90 % or more yield
- The smaller the dies, the higher the yield

You Need To Test Every Chip You Manufacture

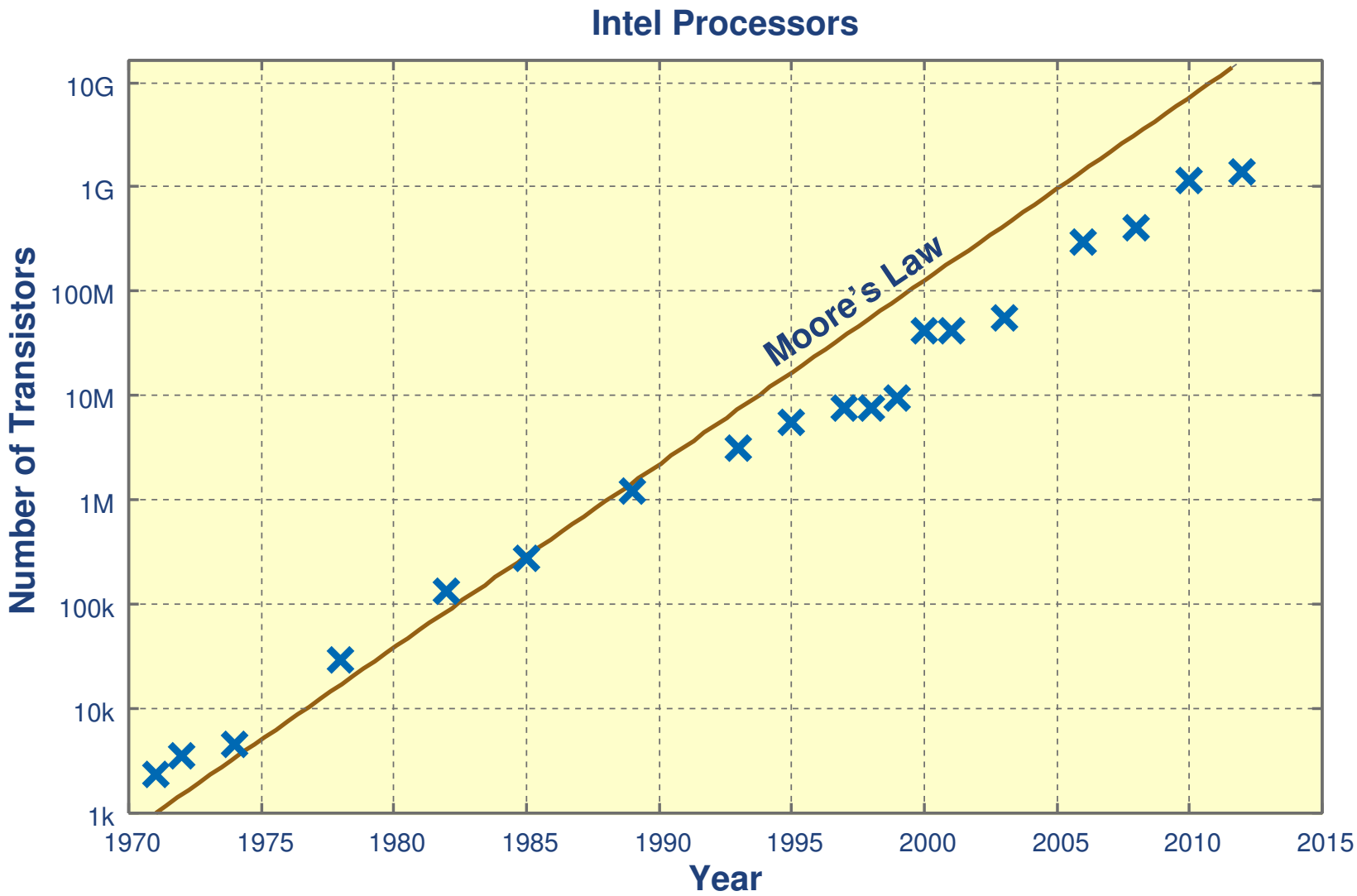
A certain amount of chips will always be defective

- Keeps production costs down. 100 % yield **economically not feasible**
- Yield typically improves over time for a specific process
  - Cutting edge technologies may have as low as 10 % yield
  - Established technologies have 90 % or more yield
- The smaller the dies, the higher the yield

Testing can be expensive and time consuming

- Typical tester costs more than 1 M$
- Test costs expressed usually per pin and per time
  - Make sure you spend as little time as possible on tester
  - Test at speed within the chip (Built-in Self-Test)
- Most testing is outsourced to dedicated test houses

Moore's Law: 2x Transistors Every 18 Months



http://www.intel.com/content/www/us/en/history/history-intel-chips-timeline-poster.html

What Moore's Law Doesn't Say

- **Chips are not necessarily getting smaller**
  - We can put **more transistors** per unit area
  - But very small (and very large chips) are not feasible
  - We need to **find use** for more transistors

What Moore's Law Doesn't Say

- **Chips are not necessarily getting smaller**
  - We can put **more transistors** per unit area
  - But very small (and very large chips) are not feasible
  - We need to **find use** for more transistors
- **How long will it hold up?**
  - Has been predicted to end multiple times (even by Moore himself)
  - Self fullfilling prophecy, sets goal for industry
  - Process options become more complex and more expensive
  - Physical limits (single atom) approaching fast

What Moore's Law Doesn't Say

- **Chips are not necessarily getting smaller**
  - We can put **more transistors** per unit area
  - But very small (and very large chips) are not feasible
  - We need to **find use** for more transistors
- **How long will it hold up?**
  - Has been predicted to end multiple times (even by Moore himself)
  - Self fullfilling prophecy, sets goal for industry
  - Process options become more complex and more expensive
  - Physical limits (single atom) approaching fast
- **Chips are not necessarily getting faster**
  - Side effect of scaling: **smaller transistors switch faster**
  - Power density issues: **voltage is reduced**
  - Reduced voltage: **reduces operating speed**
  - Trade-off between speed/power

Die Area from a Business Perspective

Type	Approximate Size	Notes
very small	$< 1\,mm^2$	**Sawlines** are a problem
small	$1\,mm^2$ - $10\,mm^2$	**Low die cost**
large	$10\,mm^2$ - $100\,mm^2$	More design effort required
very large	$> 100\,mm^2$	**Yield** an issue

- If dies are very small, silicon area is wasted on sawlines
- If dies are very large, yield decreases, die cost increases
- Sweetspot for die area between **$1\,mm^2$ - $100\,mm^2$**

Die Area from a Business Perspective

Type	Approximate Size	Notes
very small	$< 1\,mm^2$	**Sawlines** are a problem
small	$1\,mm^2$ - $10\,mm^2$	**Low die cost**
large	$10\,mm^2$ - $100\,mm^2$	More design effort required
very large	$> 100\,mm^2$	**Yield** an issue

- If dies are very small, silicon area is wasted on sawlines
- If dies are very large, yield decreases, die cost increases
- Sweetspot for die area between **$1\,mm^2$ - $100\,mm^2$**

Not all chips will profit equally from Moore's Law

If your chip is very small, making it even smaller will not reduce costs

The Ultimate Goal of System Designers/Managers

Powerpoint to Chip (GDSII)

The Ultimate Goal of System Designers/Managers

Powerpoint to Chip (GDSII)

Why doesn't this work yet?

- Perhaps there is a conspiracy?
- Maybe we are lazy!
- It could be that we want to keep our jobs.

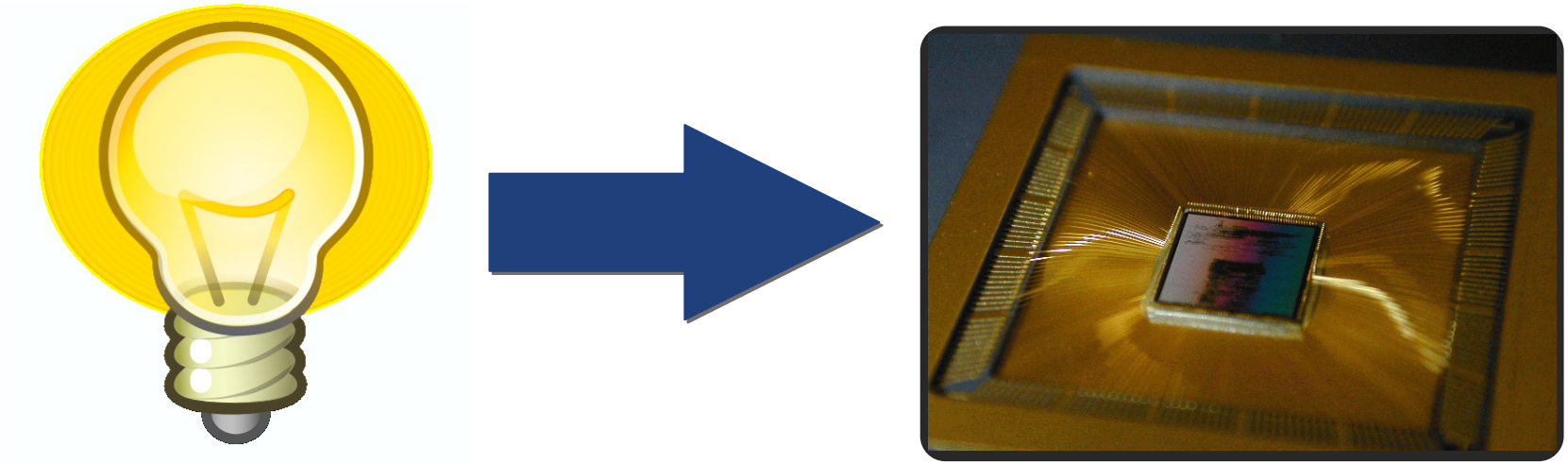
The Ultimate Goal of System Designers/Managers

Powerpoint to Chip (GDSII)

Why doesn't this work yet?

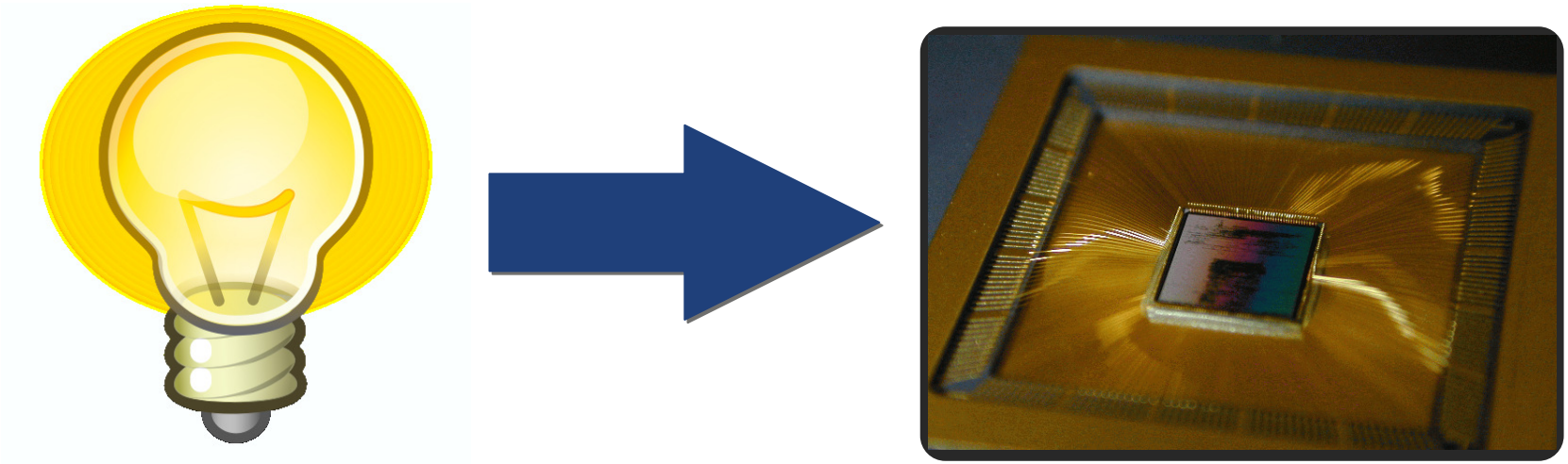
- Perhaps there is a conspiracy?
- Maybe we are lazy!
- It could be that we want to keep our jobs.
- .. or it is actually **not so easy**.

From Idea to ASIC: The Design Flow



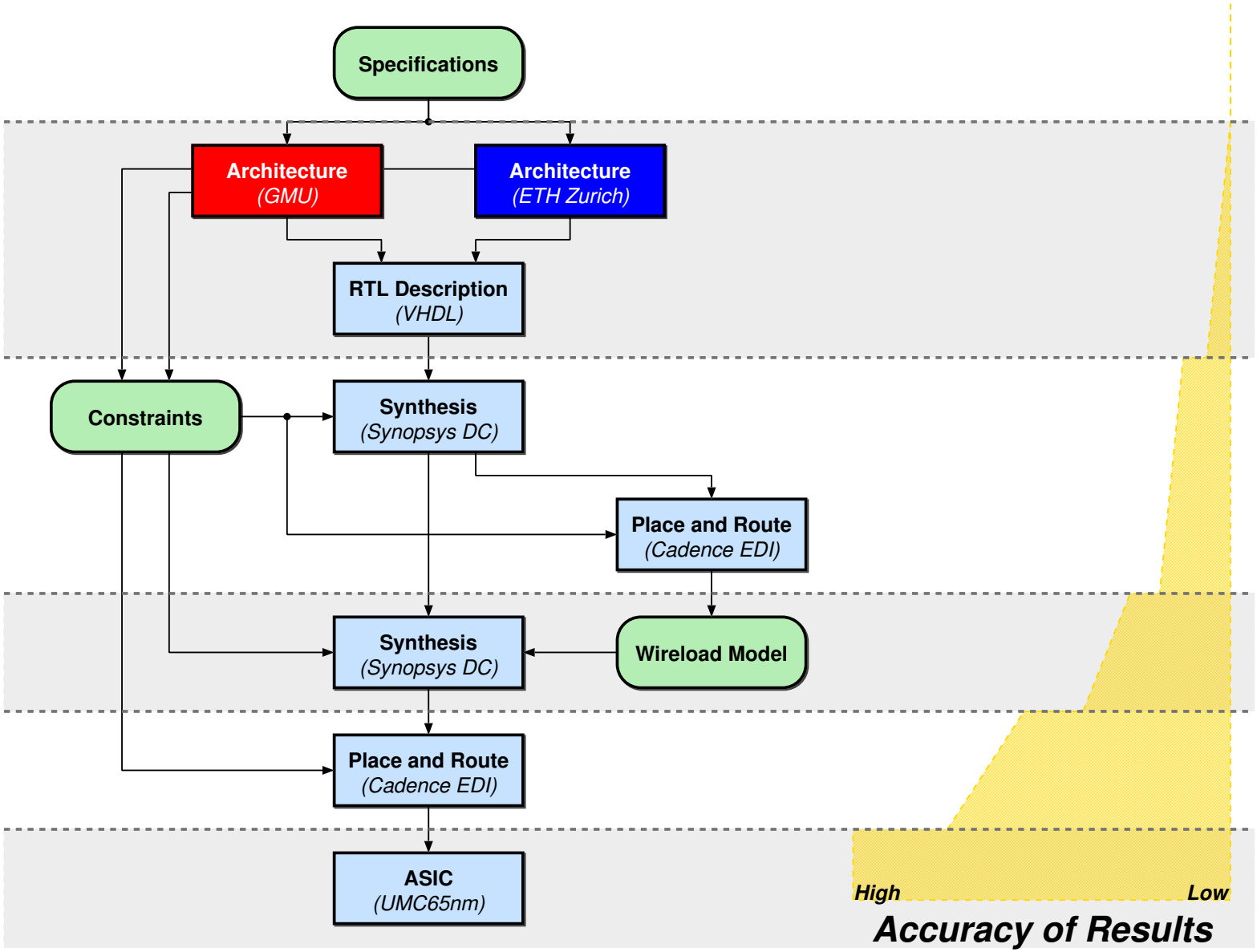
- Front-end Design Flow
  - **Design**: Coming up with an architecture
  - **Design Entry**: Description in HDL
  - **Verification**: Making sure it does what it is supposed to
  - **Synthesis**: Mapping to target technology
  - **Scan Insertion**: Adding structures for testing

From Idea to ASIC: The Design Flow



- Back-end Design Flow
  - **Placement**: Putting all logic gates / macros on chip
  - **Clock Tree Insertion**: Distributing clock signal on the chip
  - **Routing**: Establishing connections on the chip
  - **Timing Verification**: Making sure the setup/hold times are met
  - **Chip Finishing**: Pads, logos, seal ring
  - **DRC and LVS**: Making sure that the physical layout is correct

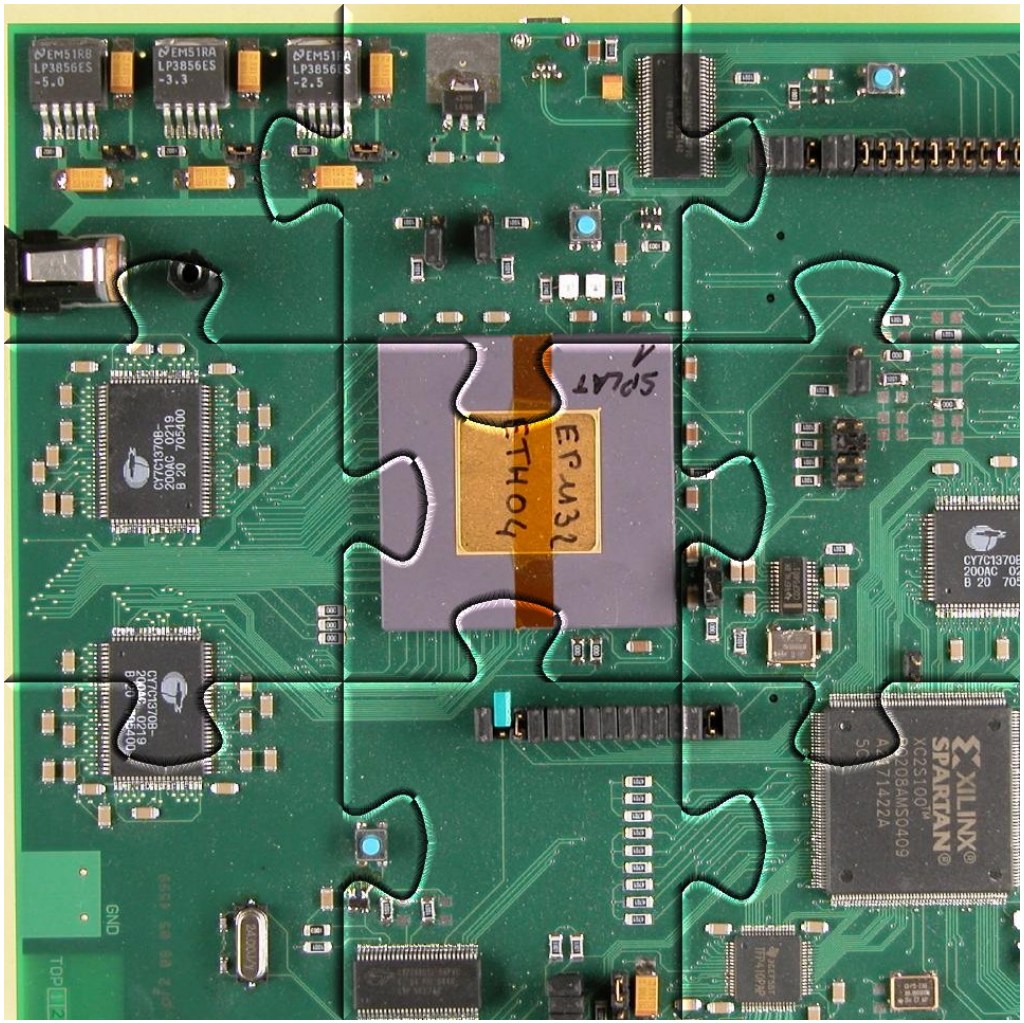
Confidence in the Results Increase with Flow



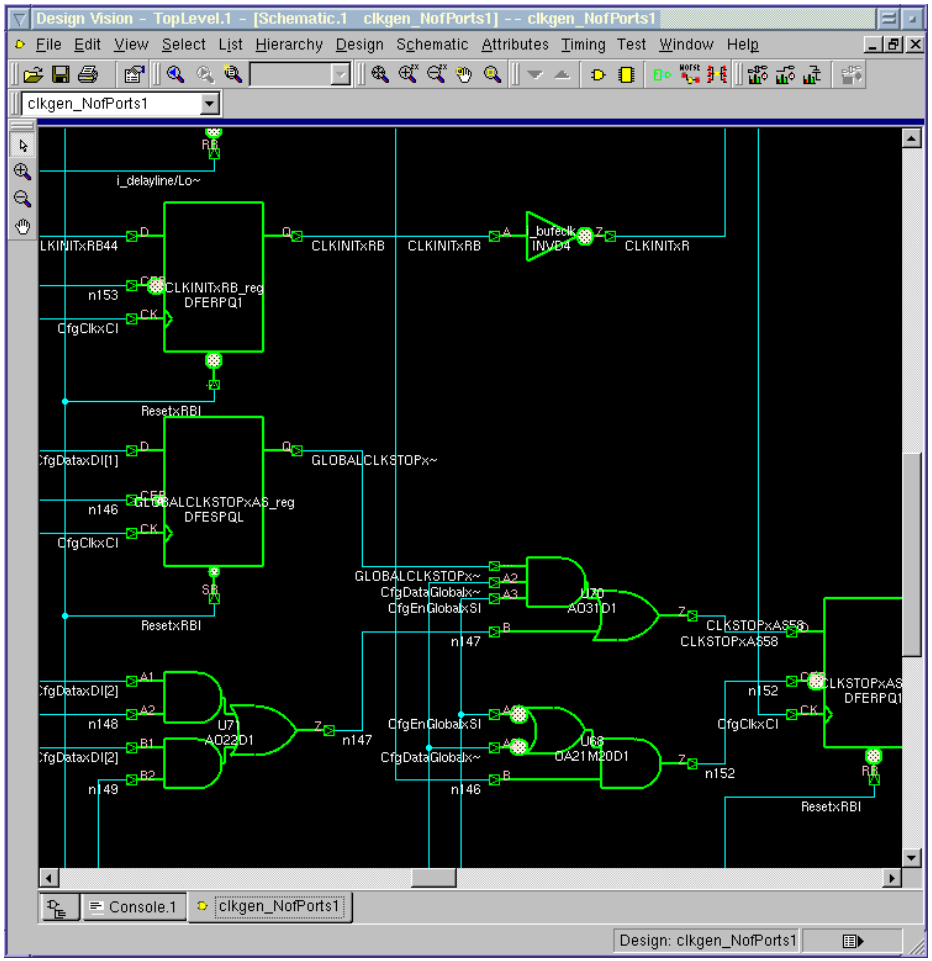
Design Specifications

A List of Requirements

- **Function**
  What exactly is expected from the chip?
- **Performance**
  How fast, what area, how much power?
- **I/O requirements**
  How will the ASIC fit together with the rest of the system?



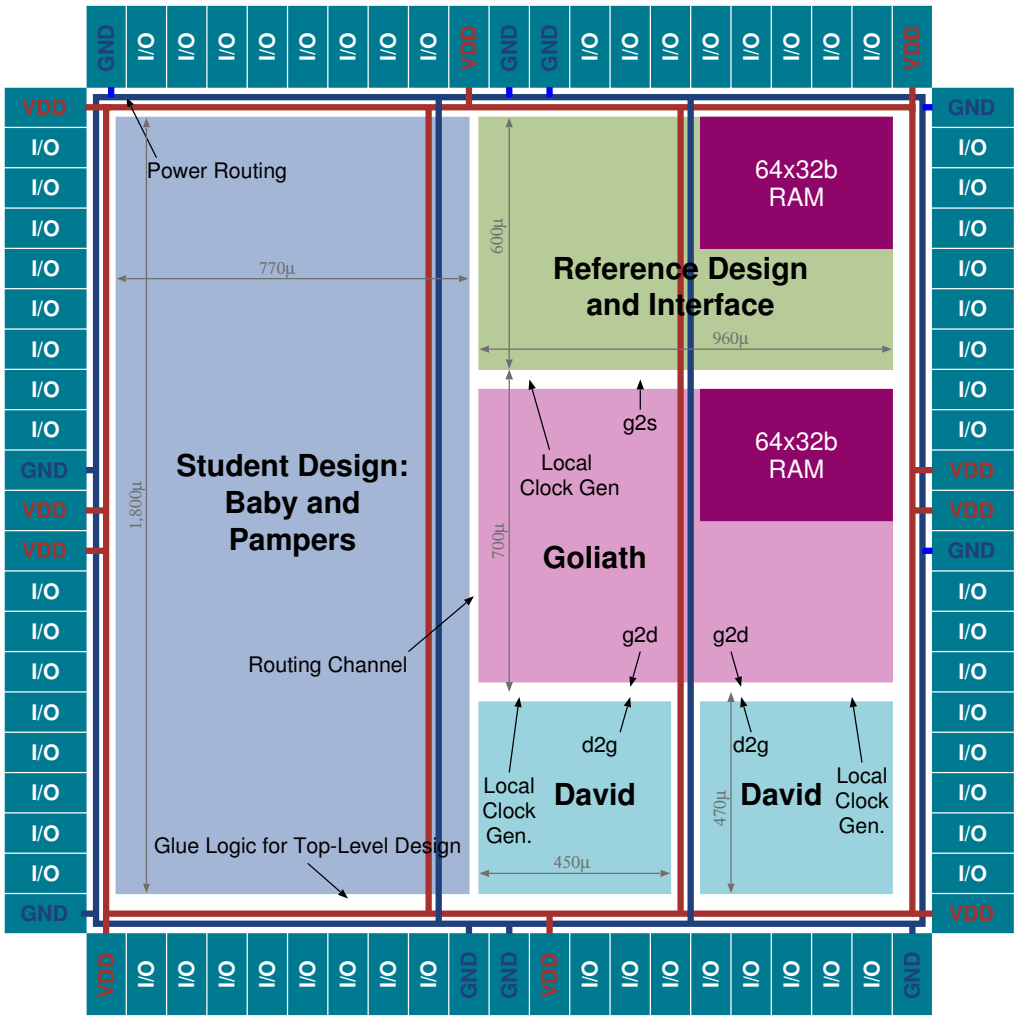
Synthesize HDL into Gates



Acquiring Physical Properties

- **Technology**
  Decided by choosing the target library
- **Area**
  Gates occupy area
- **Speed**
  Propagation through gates requires time
- **Power**
  Gates consume power during switching and when idle

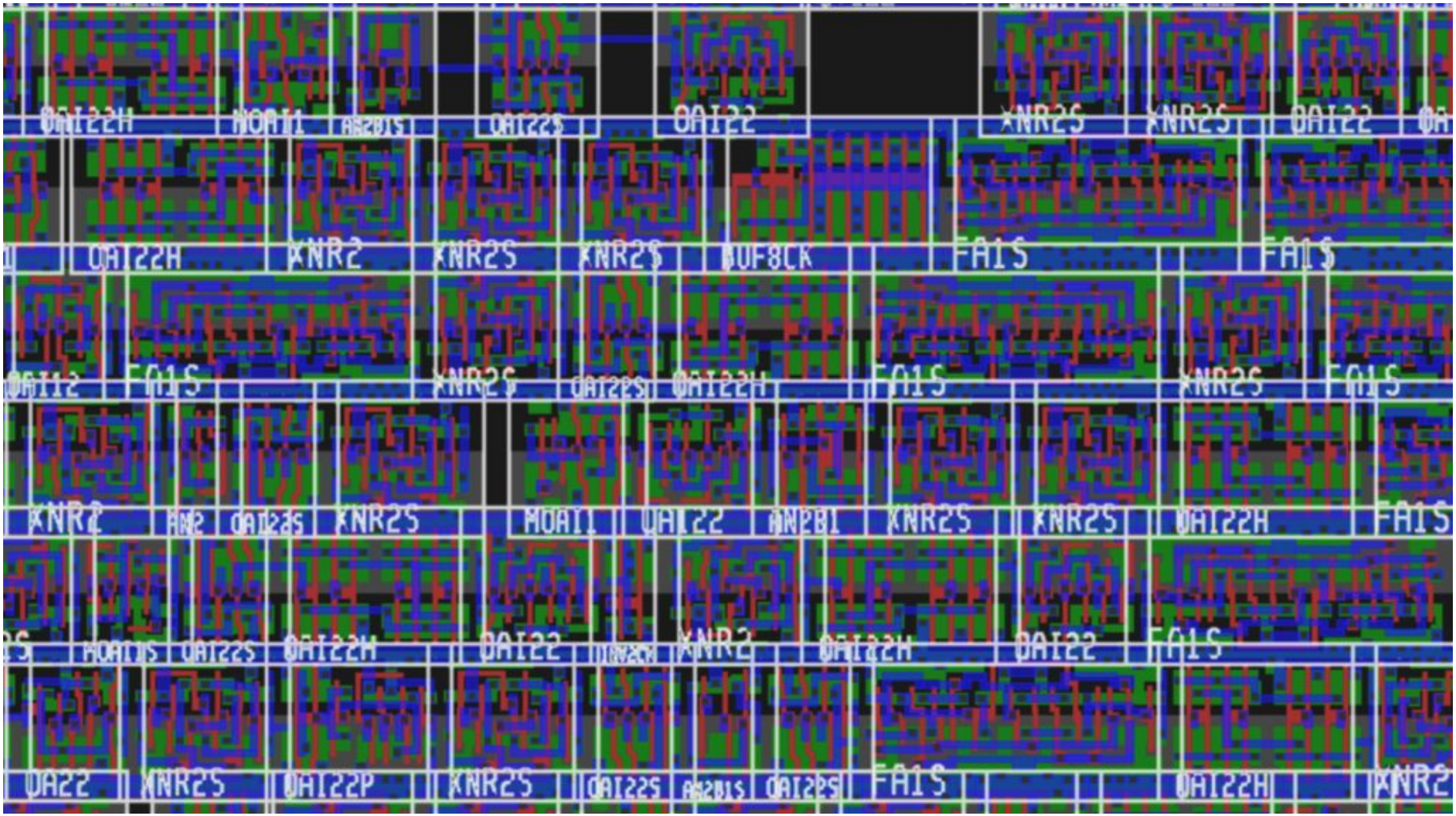
Floorplanning



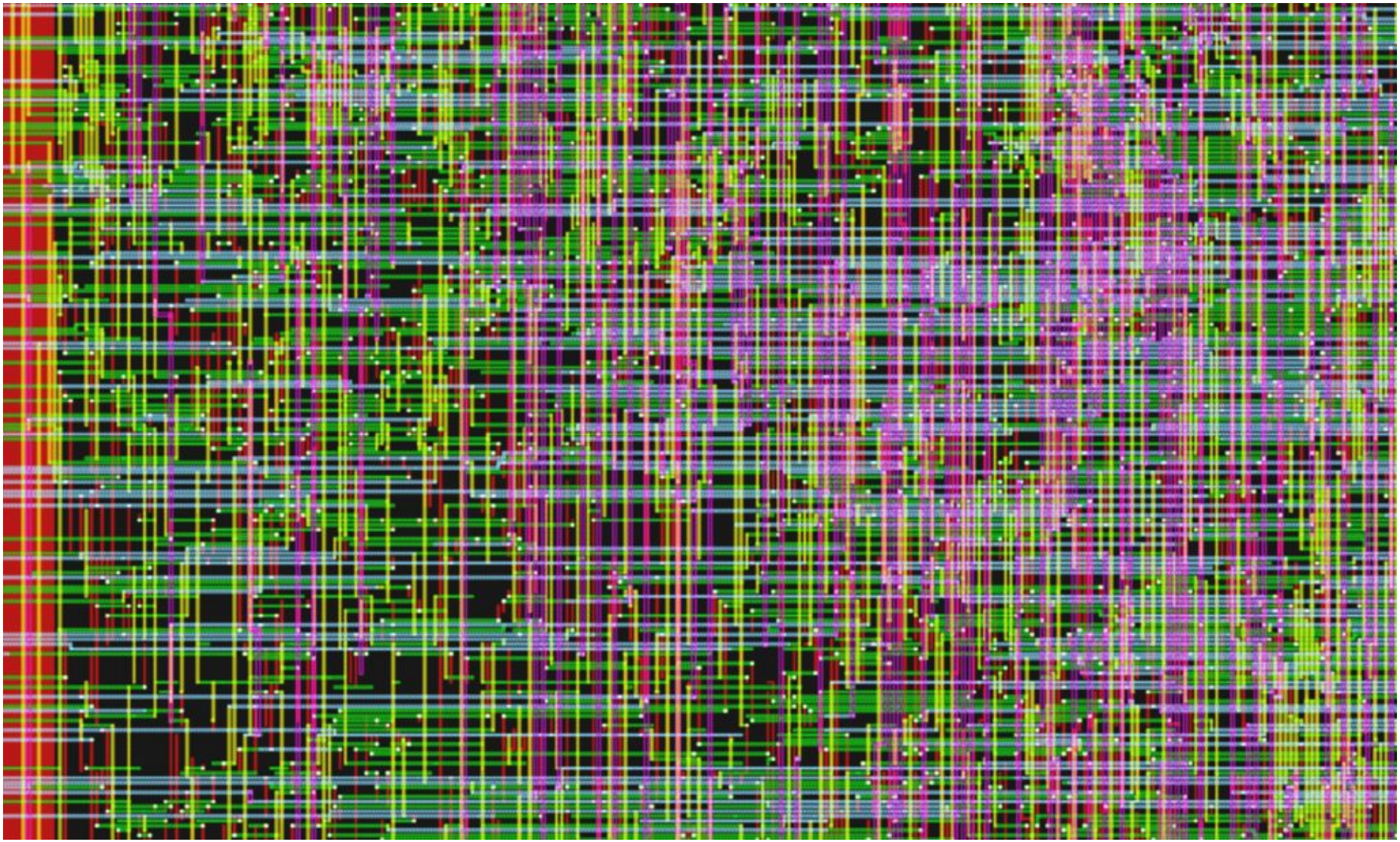
How will the chip look

- Determine area/geometry of the chip
- Place macro and I/O cells
- Design the power connections
- Leave room for routing optimizations
- **Iterative process**, can not determine the *perfect* floorplan from beginning.

Placement is a NP-Hard Problem



Routing Determines the Parasitic Load



What Are Performance Parameters?

- **Area**
  Total area occupied by circuit
- **Throughput**
  Number of data items processed per unit time
- **Power**
  Peak power for operation
- **Time to Market**
  How long the design will take

What Are Performance Parameters?

- **Area**
  Total area occupied by circuit
- **Throughput**
  Number of data items processed per unit time
- **Power**
  Peak power for operation
- **Time to Market**
  How long the design will take

- **Cost**
  Total silicon cost
- **Latency**
  Amount of time to process a given data item
- **Energy**
  Total energy to finish task
- **Man Months to Design**
  How much engineers cost

What Are Performance Parameters?

- **Area**
  Total area occupied by circuit
- **Throughput**
  Number of data items processed per unit time
- **Power**
  Peak power for operation
- **Time to Market**
  How long the design will take

- **Cost**
  Total silicon cost
- **Latency**
  Amount of time to process a given data item
- **Energy**
  Total energy to finish task
- **Man Months to Design**
  How much engineers cost

- These are related, but they are **not the same**
- Important to know, which one we are really interested in!

How to Improve Performance

How to Improve Performance

- **Use a more advanced technology**
  - Easy way
  - Not always feasible
  - Costs will increase
  - Additional problems may arise (adapting, lack of IPs)

How to Improve Performance

- **Use a more advanced technology**
  - Easy way
  - Not always feasible
  - Costs will increase
  - Additional problems may arise (adapting, lack of IPs)
- **Make a better implementation**
  - Harder, needs smart (expensive) engineers
  - Not always possible, there is a limit to what can be done
  - Academia should concentrate on this, show what is the limit

Academic vs Industrial IC Design

Both of them

- have similar design flows
- use the same tools
- define performance similarly

Academic vs Industrial IC Design

Both of them

- have similar design flows
- use the same tools
- define performance similarly

..but in Industry

The chips need to work within specifications.

- Goal is to **sell a product**
- **Cost and time** to design important
- **Manufacturability** is paramount
- Mostly **conservative design** practices are used

EDA tools are Designed for Industry Requirements

For the industry:

- Circuit has to function to specification even in **worst** conditions
- **Constraints** are used for to define specifications
- EDA tools make sure circuit performance meets specifications
  Not designed to get *peak* performance

EDA tools are Designed for Industry Requirements

For the industry:

- Circuit has to function to specification even in **worst** conditions
- **Constraints** are used for to define specifications
- EDA tools make sure circuit performance meets specifications
  Not designed to get *peak* performance

In academia:

- Interested only in the limits (*fastest*, *smallest* etc.)
- Performance numbers needed to show how we compare against others
- It is not easy to get **fair** numbers from these tools
- Constraints are **misused** to explore design space

Real-World Problems will Effect Performance

- Parasitic RLC effects on timing
- Clock distribution (skew, jitter)
- Cross talk, signal integrity
- I/O speed limitations
- IR drop (more area for power routing)
- Maximum current density limits (more area for power routing)
- PVT variations (die-to-die, inter-die), reliability
- Thermal issues, temperature variation during operation

Real-World Problems will Effect Performance

- Parasitic RLC effects on timing
- Clock distribution (skew, jitter)
- Cross talk, signal integrity
- I/O speed limitations
- IR drop (more area for power routing)
- Maximum current density limits (more area for power routing)
- PVT variations (die-to-die, inter-die), reliability
- Thermal issues, temperature variation during operation

All of these problems have solutions...

... but they come at a **cost!**

Academic Reporting Neglects Some Problems

- **Overhead of Solutions not Always Included**

  For example:
  - Large I/O bandwidth (*serial I/O*, *pads*, *buffers*)
  - Very fast clocks (*PLLs*, *costly clock tree*)
  - Dynamic Voltage/Frequency Scaling (*synchronizers*, *level shifters*)
- **Additional Verification Effort**

  Circuits with:
  - Many operational modes
  - Interfaces to different clock domains

  Can be costly in terms of verification regardless of the circuit size
- **Testing Overhead**

  Some circuits are difficult to test (*i.e. asynchronous circuits*), or require extensive test modes

Quality of Results Depend on Design State

Different Levels, Different Accuracy

- **Synthesis**
  - First level with physical properties
  - Routing overhead unknown, estimated by wireload models
  - Timing and power models are mean values, not that accurate

Quality of Results Depend on Design State

Different Levels, Different Accuracy

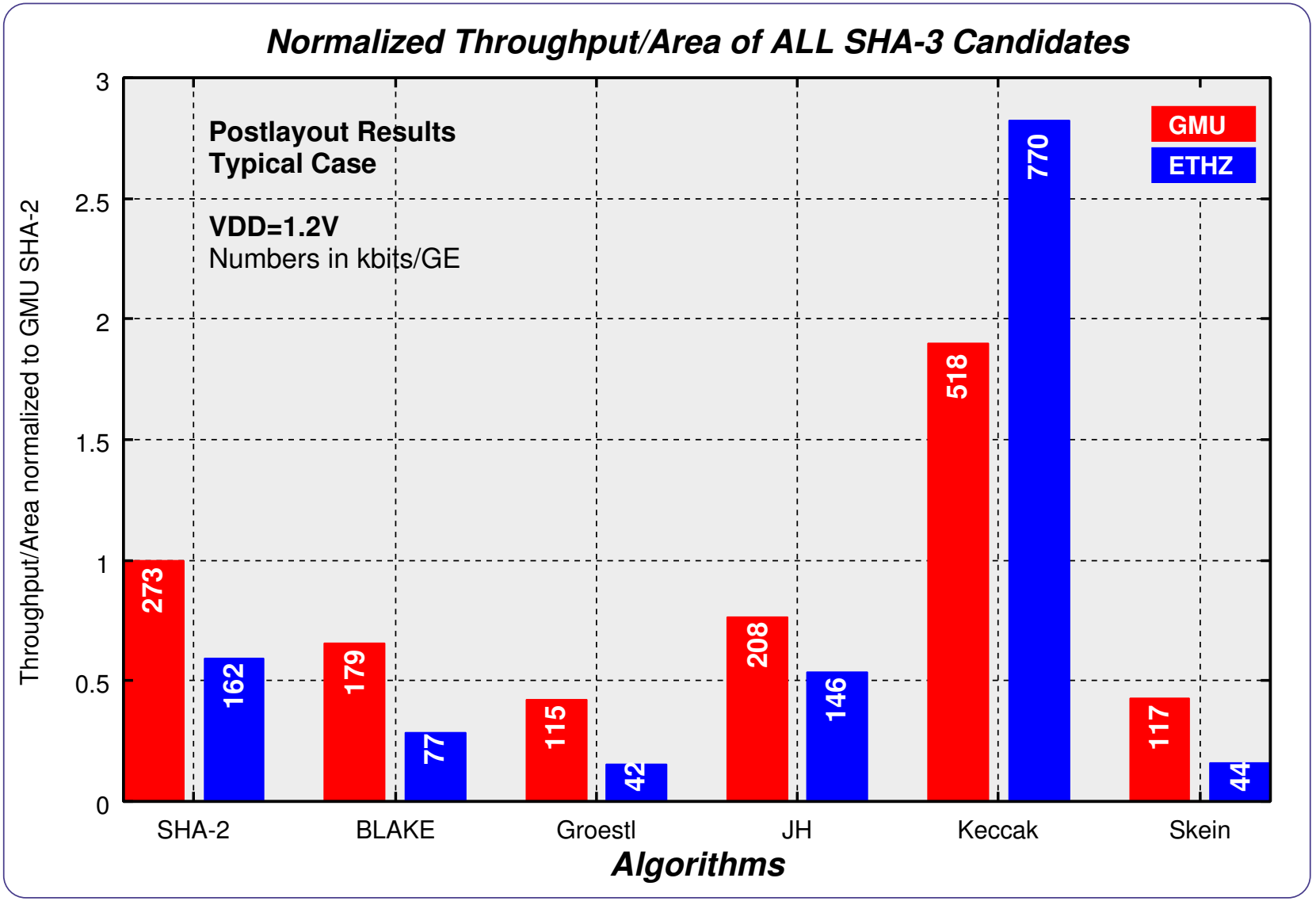
- **Synthesis**
  - First level with physical properties
  - Routing overhead unknown, estimated by wireload models
  - Timing and power models are mean values, not that accurate
- **Post-Layout**
  - Routing overhead known, more accurate timing
  - Not all post-layout designs include proper provisions for: power, signal integrity, testing

Quality of Results Depend on Design State

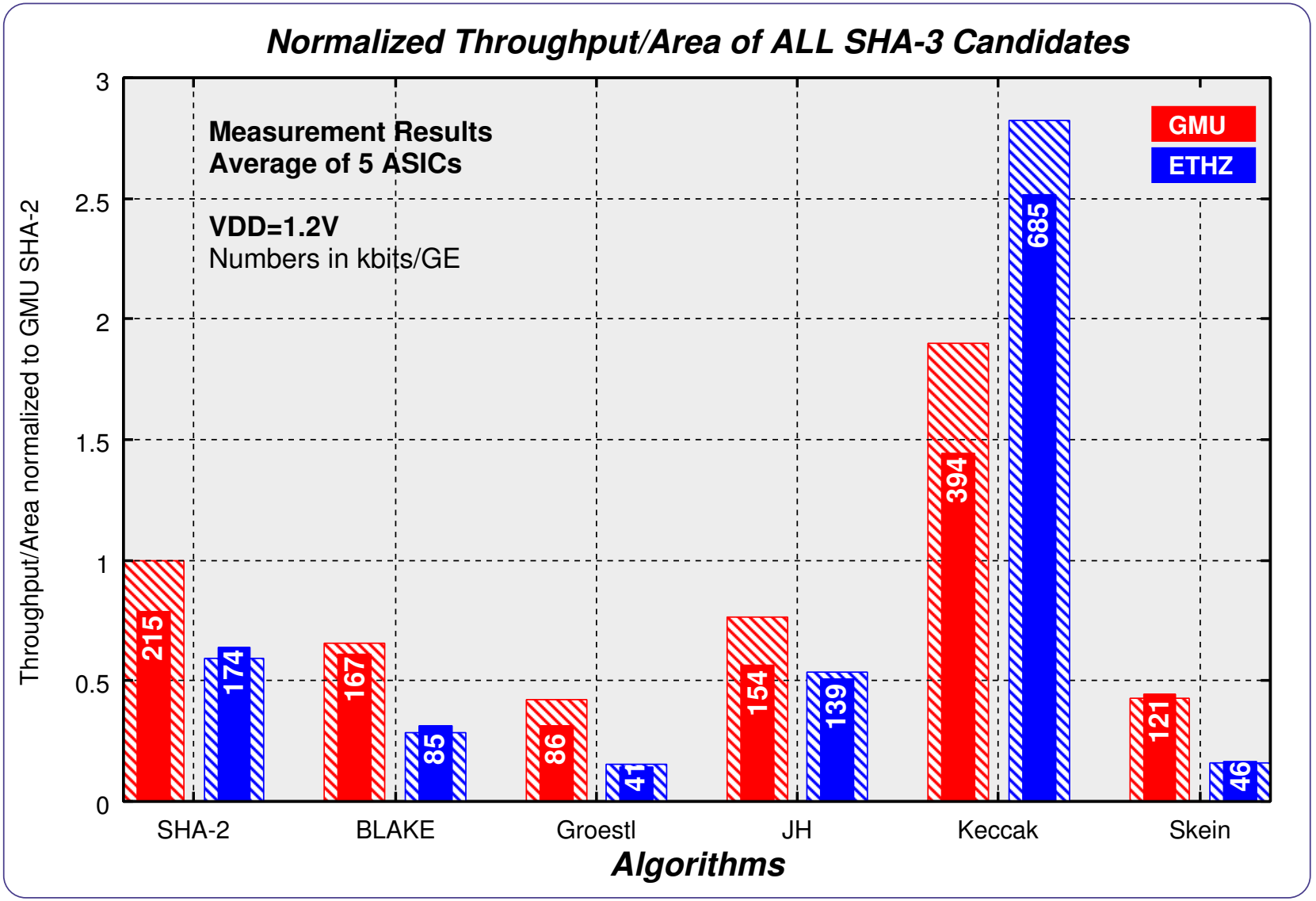
Different Levels, Different Accuracy

- **Synthesis**
  - First level with physical properties
  - Routing overhead unknown, estimated by wireload models
  - Timing and power models are mean values, not that accurate
- **Post-Layout**
  - Routing overhead known, more accurate timing
  - Not all post-layout designs include proper provisions for: power, signal integrity, testing
- **Actual Measurement**
  - Real proof of concept
  - Performance affected by practical problems, usually worse than expected
  - Requires test infrastructure, costly

Throughput/Area: Post-Layout → Measurement



Throughput/Area: Post-Layout → Measurement



Let Us Talk About Area

How much silicon area will be used for the circuit?

Let Us Talk About Area

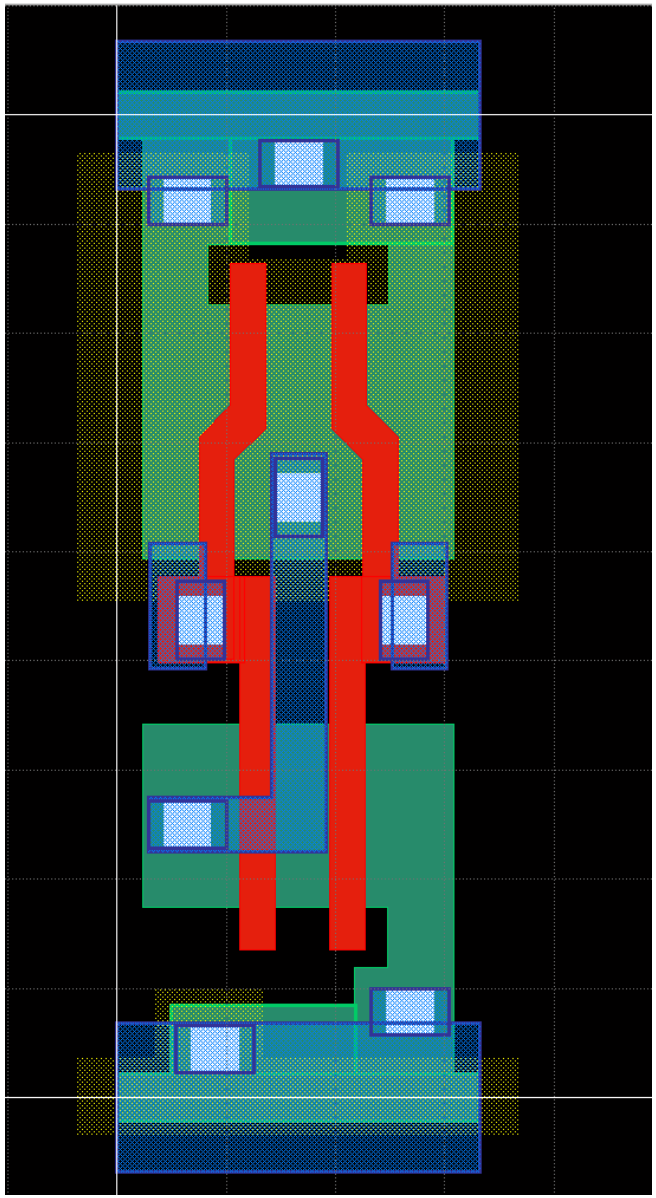
How much silicon area will be used for the circuit?

- **Why do we care?**
  - Silicon Cost
  - Feasibility (*will it fit?*)
- **Should be easy to determine**
  - Does not change after/during manufacturing
  - Reliable and accurate post-layout numbers
- **Units**
  - **mm^2** correct unit, technology dependent
  - **GE** commonly used, not accurate at all

The Story of Gate Equivalents

Area of 2-input NAND gate

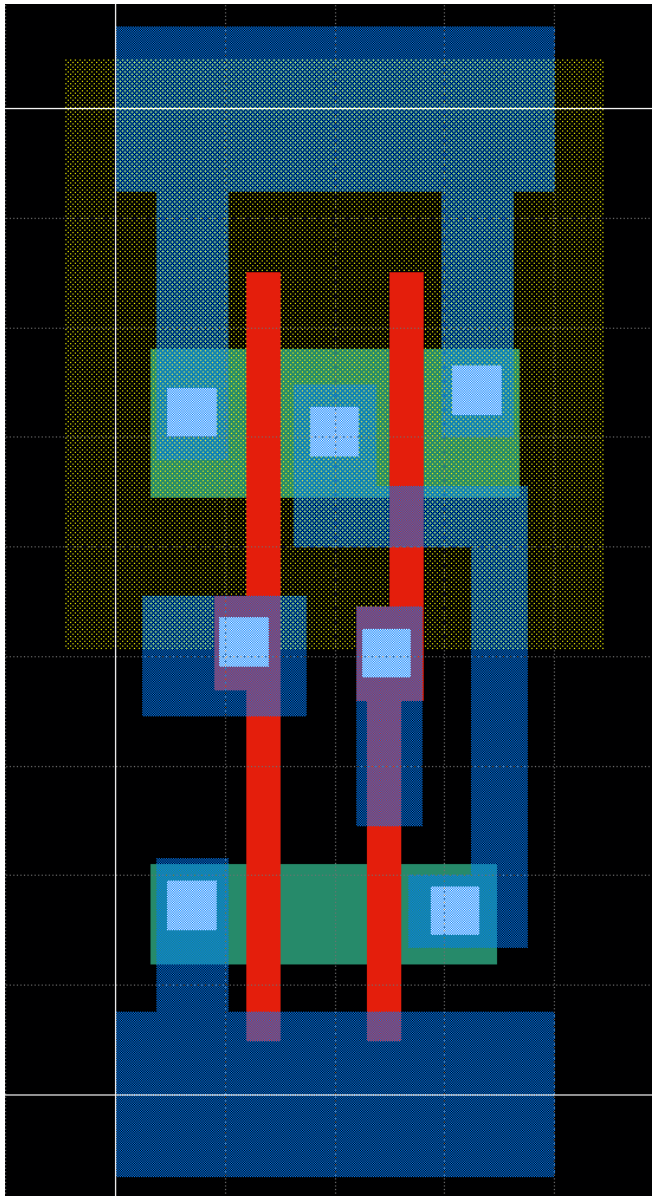
- **Historical reasons**
  Was used to compare circuits in different styles: CMOS, TTL, ECL..
  For CMOS 4 transistors is 1 GE
- **Not easy to standardize**
  Everyone can interpret it differently
- **Many 4 transistor gates**
  Buffer, 2-input NAND, NOR..
  Different drive strengths



The Story of Gate Equivalents

Area of 2-input NAND gate

- **Historical reasons**
  Was used to compare circuits in different styles: CMOS, TTL, ECL..
  For CMOS 4 transistors is 1 GE
- **Not easy to standardize**
  Everyone can interpret it differently
- **Many 4 transistor gates**
  Buffer, 2-input NAND, NOR..
  Different drive strengths
- **Library dependent**
  Width, style, height can differ



Same Designs, Different Technologies

Technology		Library		Design A		Design B	
Foundry	Node	Vendor	Tracks	[mm^2]	[kGE]	[mm^2]	[kGE]
C	28 nm	I	12	0.014	**28.1**	0.129	263.6
C	45 nm	I	9	0.016	23.2	**0.168**	**247.3**
C	45 nm	I	14	0.019	18.3	**0.220**	**208.3**
C	65 nm	I	13	0.035	**17.0**	0.407	195.5
A	65 nm	IV	9	0.030	21.0	0.311	216.0
D	65 nm	VI	9	0.029	20.1	0.305	211.7
A	90 nm	V	10	0.060	19.3	0.672	214.2
A	130 nm	V	8	0.104	20.3	1.057	206.5
E	130 nm	II	9	0.109	24.1	1.179	259.7
C	130 nm	I	12	0.115	19.0	1.299	214.6
B	150 nm	III	9	0.214	23.7	2.070	229.0
A	180 nm	V	9	0.211	22.5	2.151	229.5
A	180 nm	VII	9	0.198	20.5	2.056	212.5

GE is a very Coarse Measure for Area

Area in gate equivalents will vary at least

$\pm$ **10%**

between different technologies

In other words

- **Do not use overly precise area descriptions:**
  - 8,421 GE
  - 235,481 GE

GE is a very Coarse Measure for Area

Area in gate equivalents will vary at least

$\pm$ 10%

between different technologies

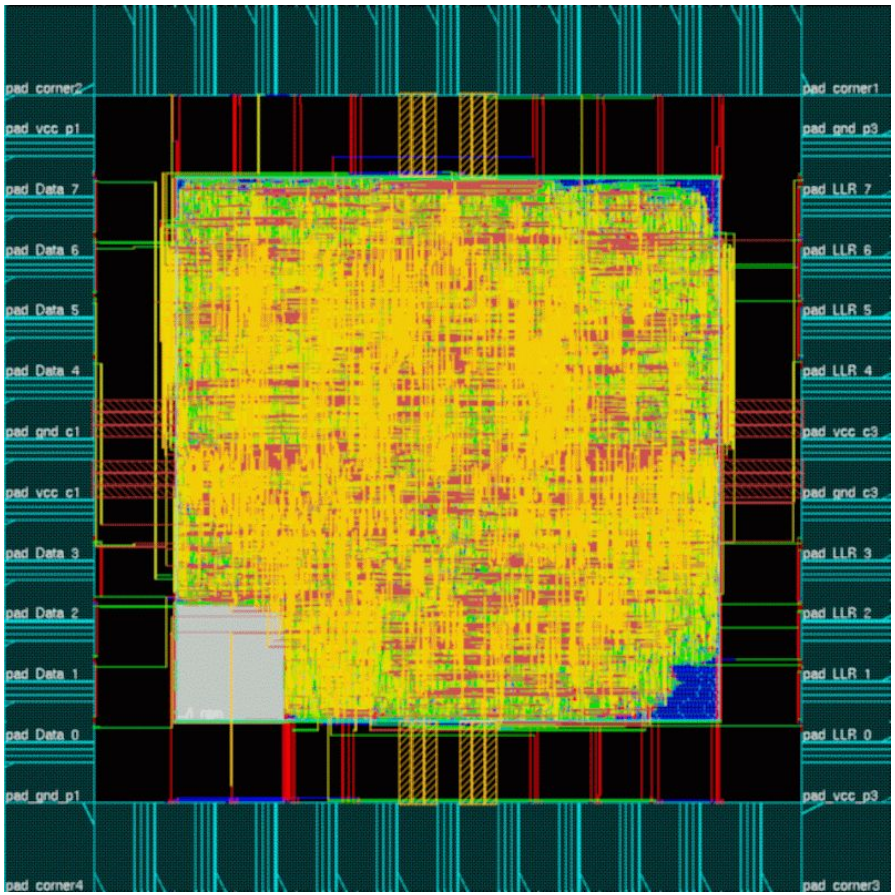
In other words

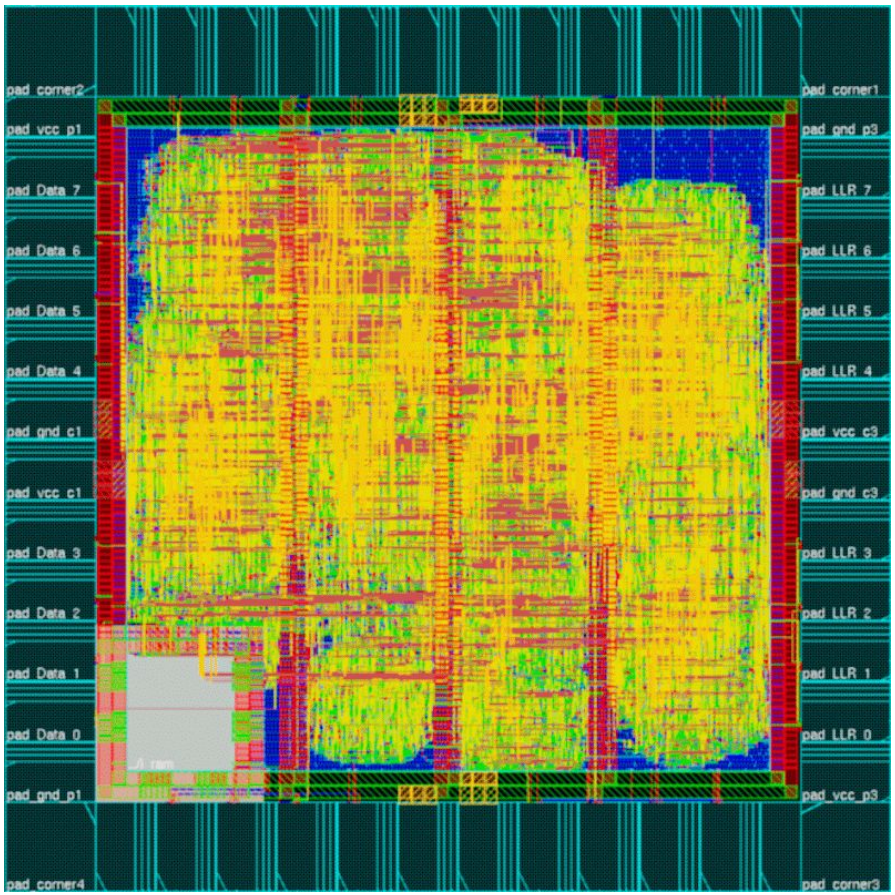
- **Do not use overly precise area descriptions:**
  - 8,421 GE
  - 235,481 GE
- **Round your results to not more than 2 significant digits!**
  - 8 kGE
  - 250 kGE

What Exactly Is the Circuit Area?

- **The number reported by the synthesizer?**
  - Does not include routing, power, clock overhead

Design With and Without Power Routing





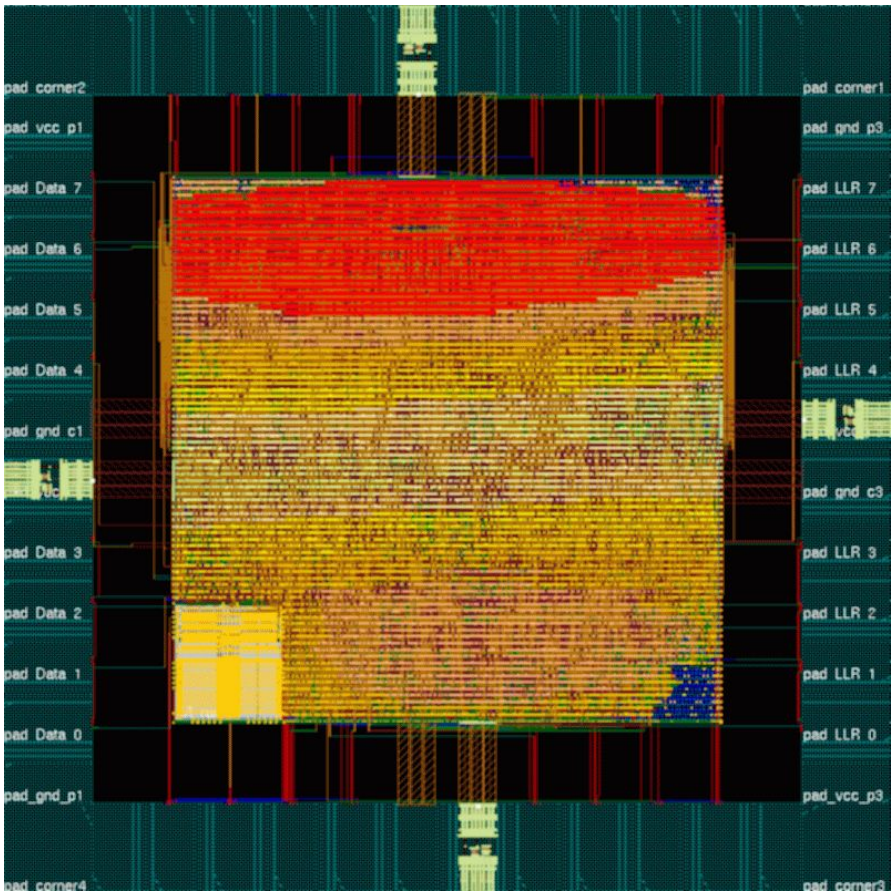
Small design

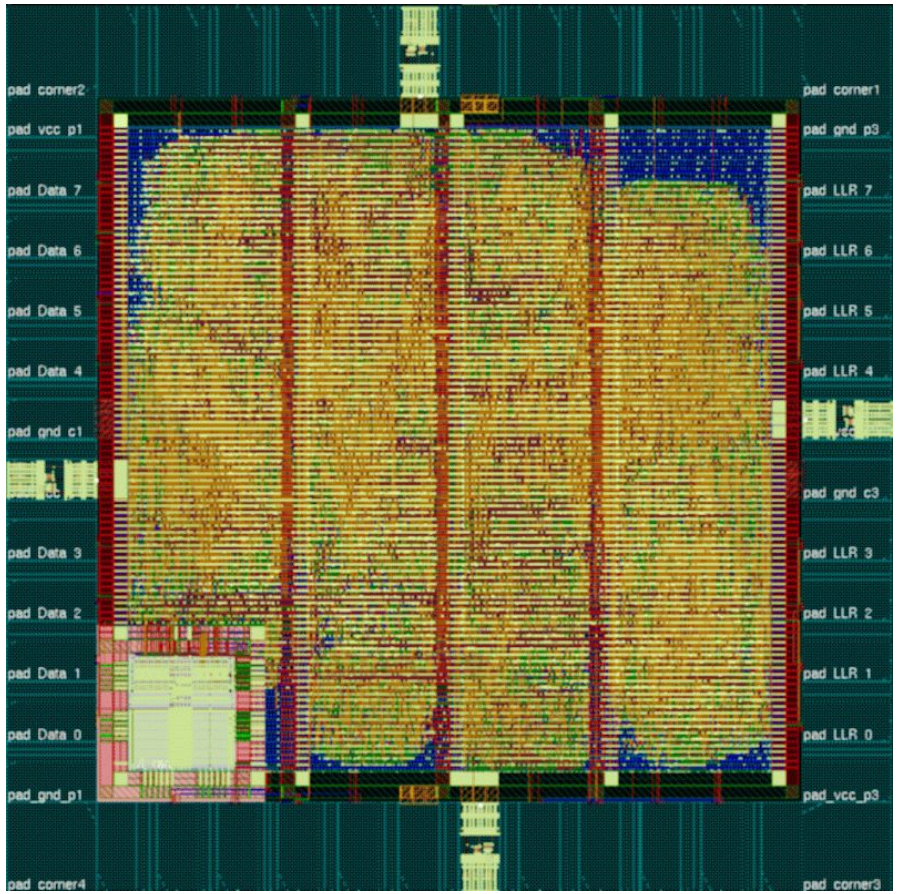
GE: 69'042,
area: $0.719\,\mathrm{mm}^2$

Same design, larger

GE: 70'906,
area: $1.183\,\mathrm{mm}^2$

Design With and Without Power Routing





Small design

Would **NOT** work at speed more than 20% IR

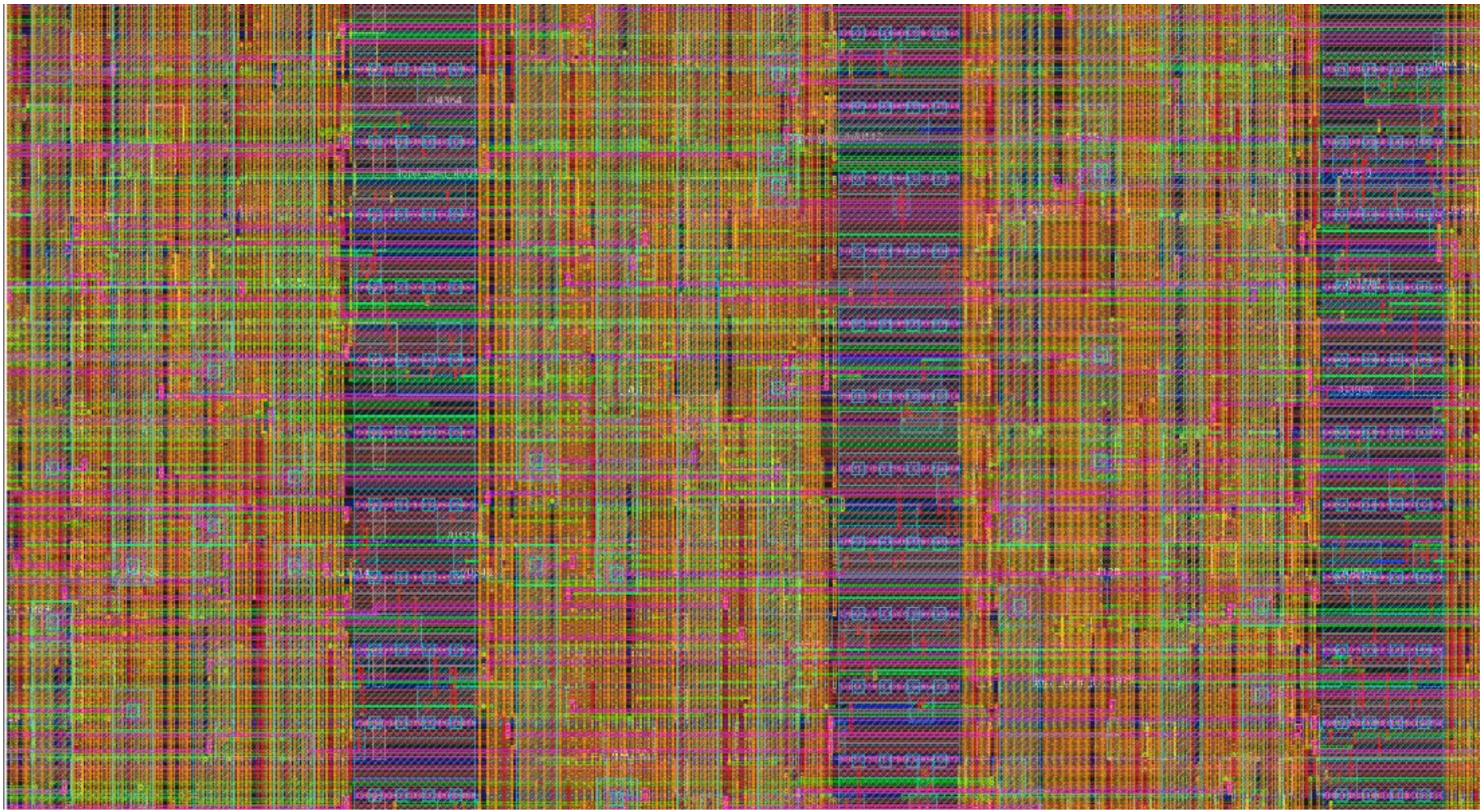
Same design, larger

Would work fine at the specified frequency

What Exactly is the Circuit Area?

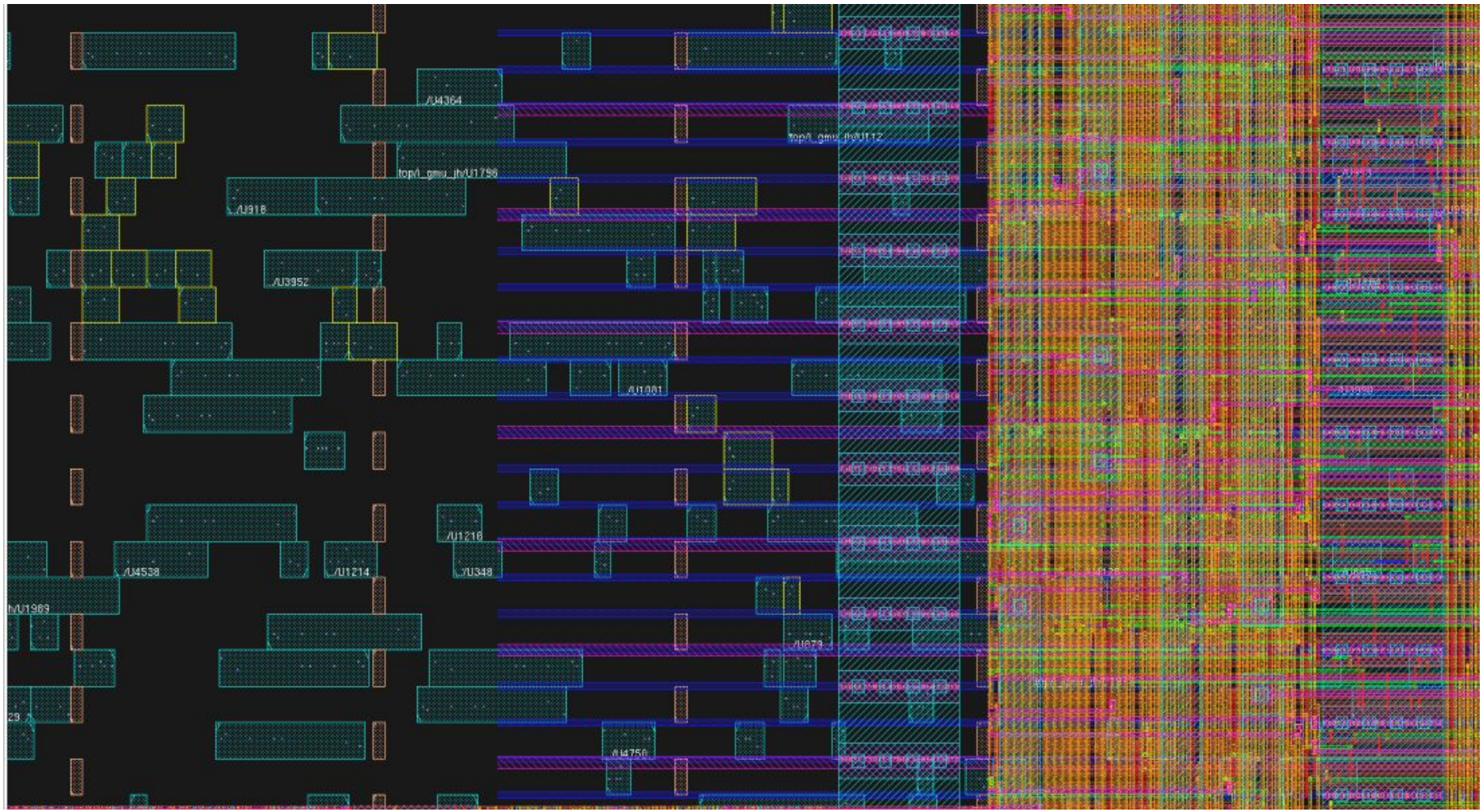
- **The number reported by the synthesizer?**
  - Does not include routing, power, clock overhead
- **The post-layout gate count?**
  - The gate count is just the area occupied by the gates
  - You need extra space for routing, signal integrity, and power routing

How Cells are Actually Placed on Chip



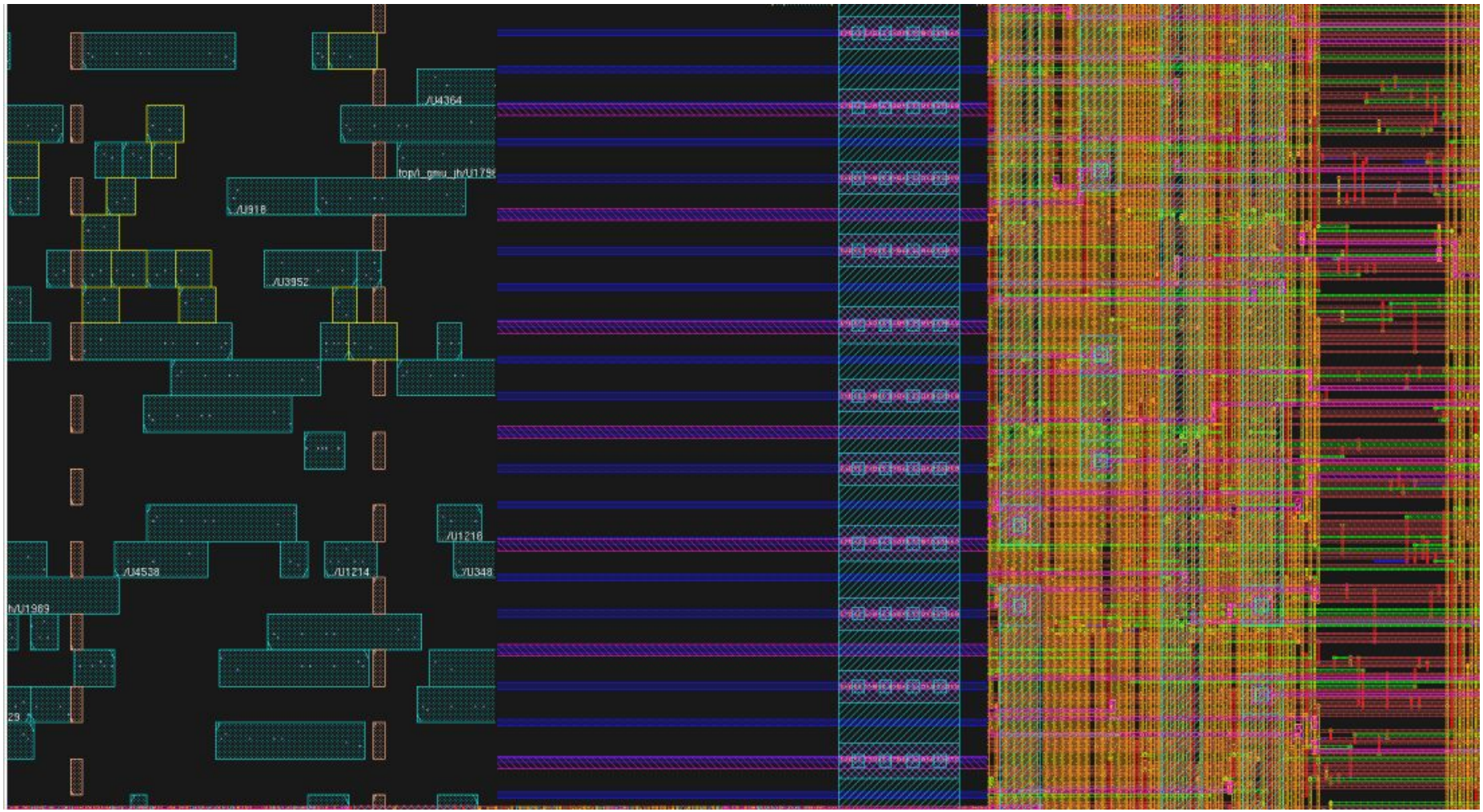
Close-up of a placed and routed design

How Cells are Actually Placed on Chip



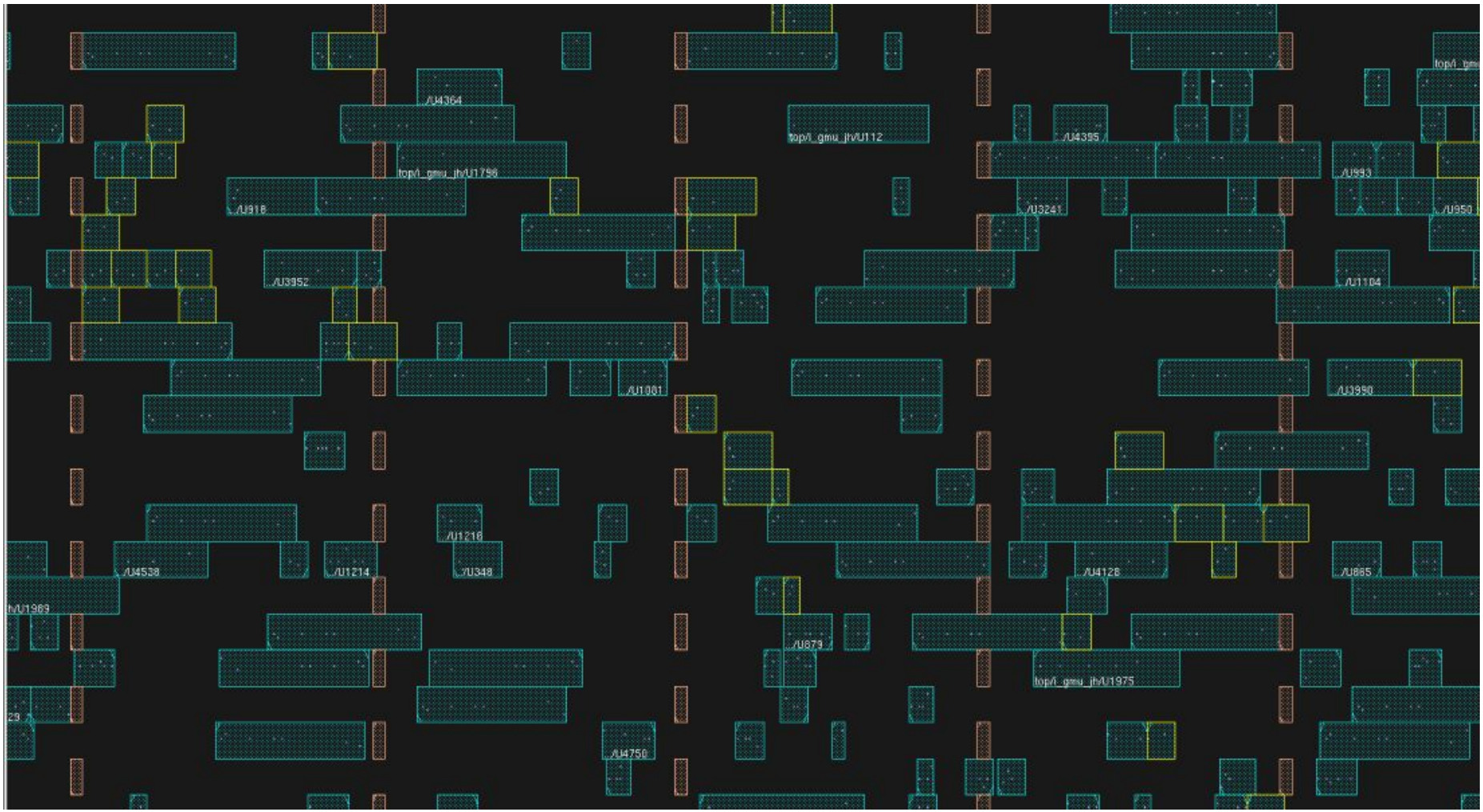
Partially stripped layout to show cells and power routing

How Cells are Actually Placed on Chip



Standard cells, power routing and signal routing separated

How Cells are Actually Placed on Chip

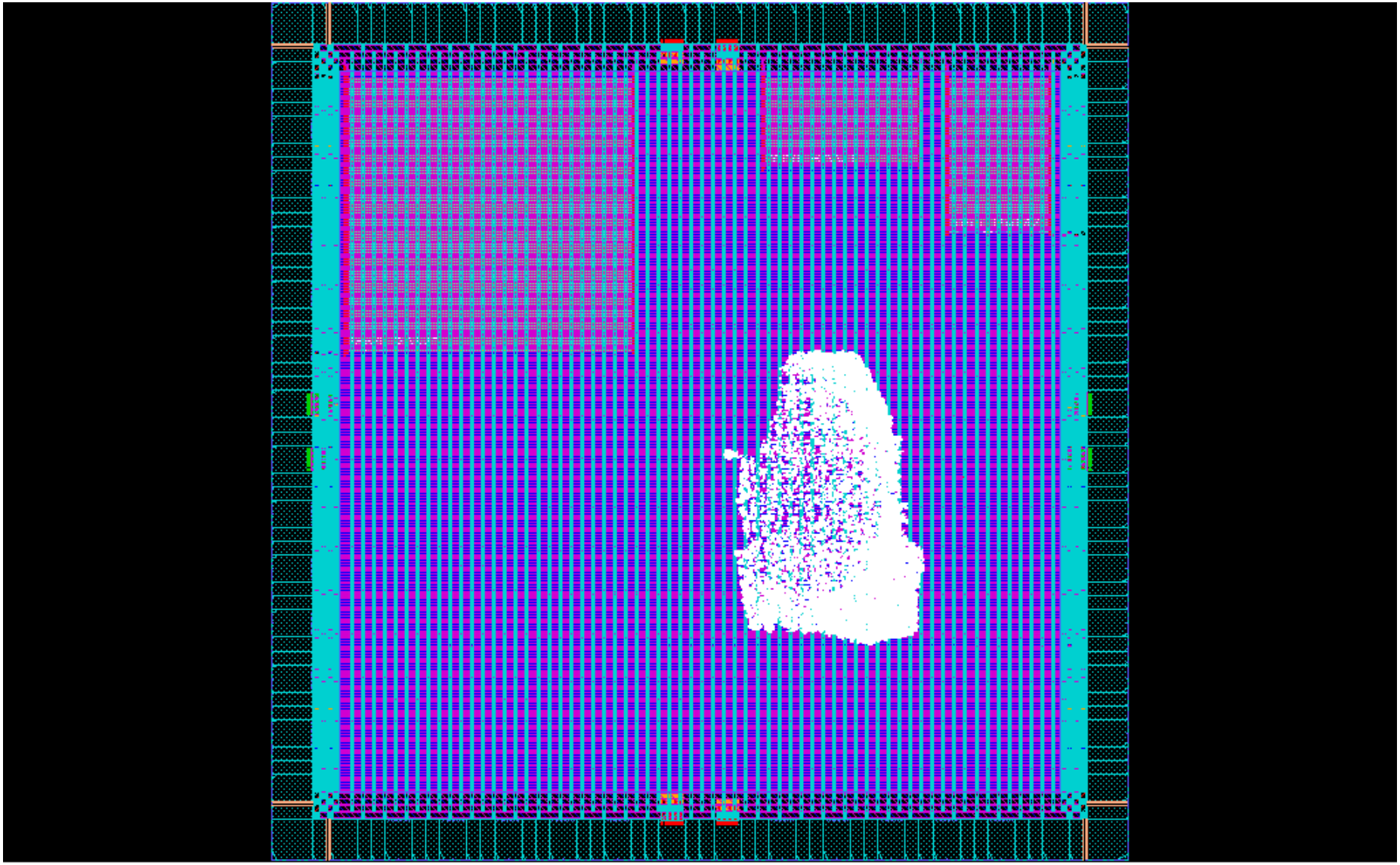


Cell placement showing standard cells, body tie cells, buffers

What Exactly Is the Circuit Area?

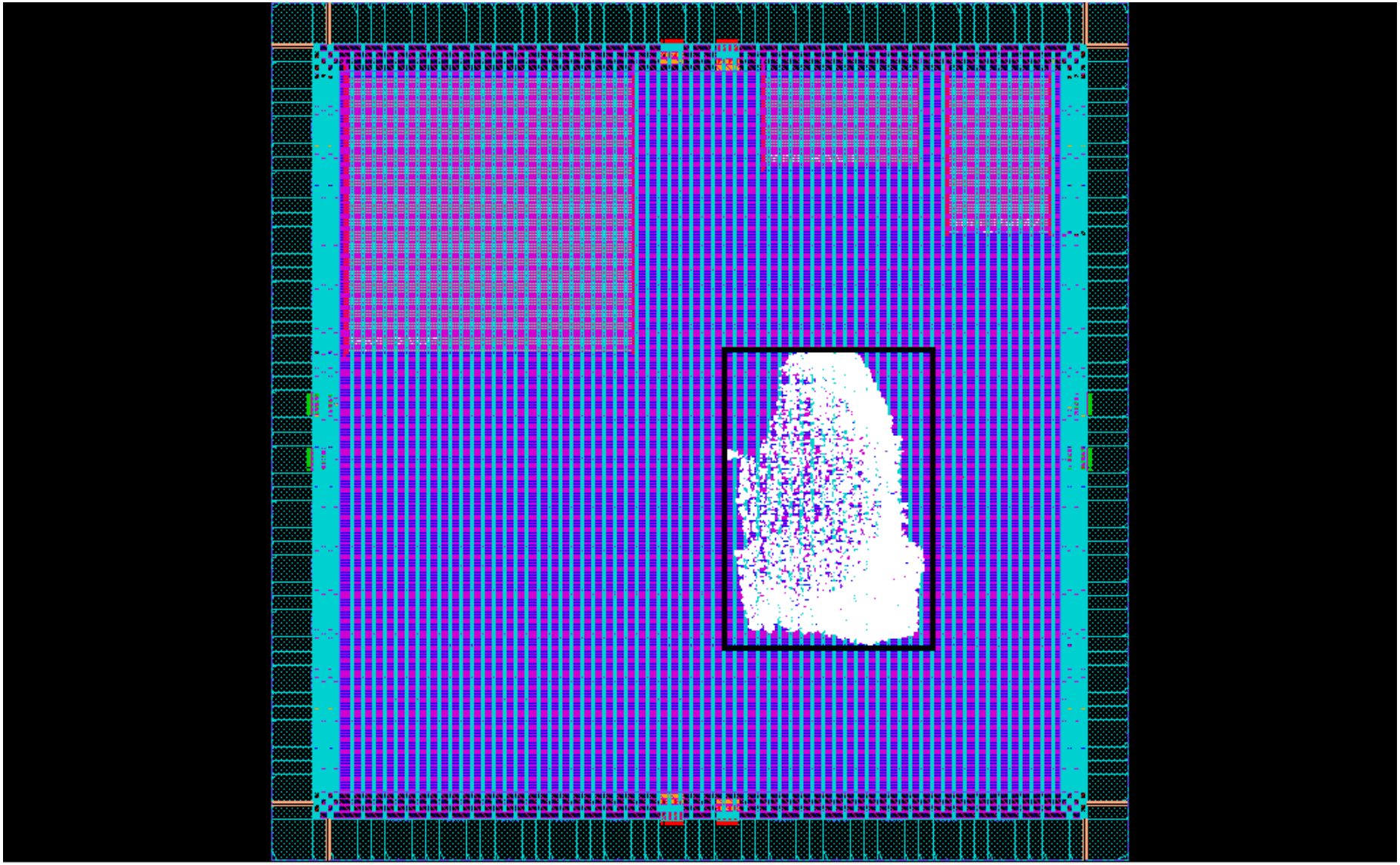
- **The number reported by the synthesizer?**
  - Does not include routing, power, clock overhead
- **The post-layout gate count?**
  - The gate count is just the area occupied by the gates
  - You need extra space for routing, signal integrity, and power routing
- **The total die area of the manufactured chip?**
  - You may have some additional structures for testing in the chip
  - Practical limits may require specific die sizes (*Europractice mini@sic*)
  - What if the chip contains more than one design?

What is the Real Area of this Block?



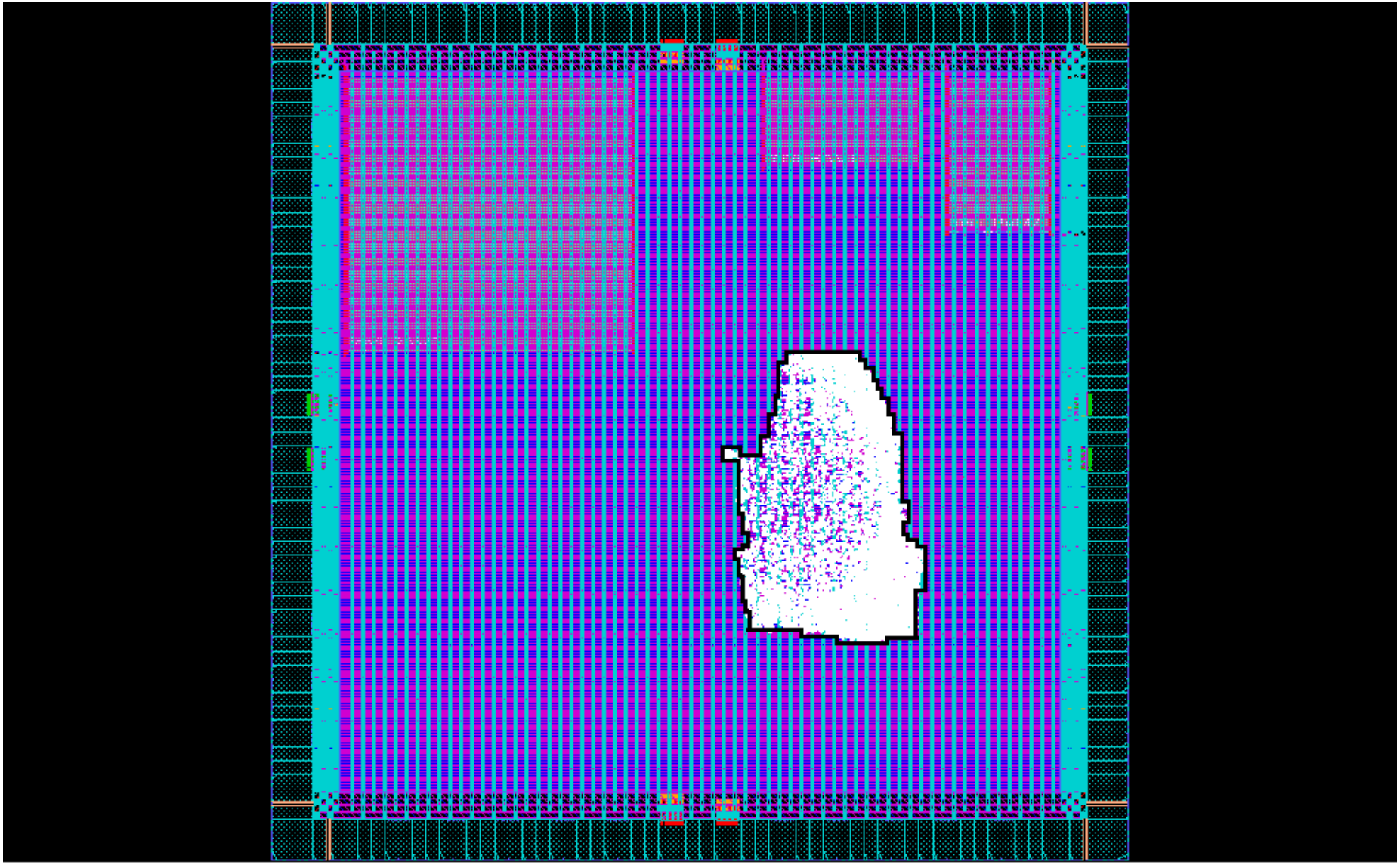
One block within a larger ASIC

What is the Real Area of this Block?



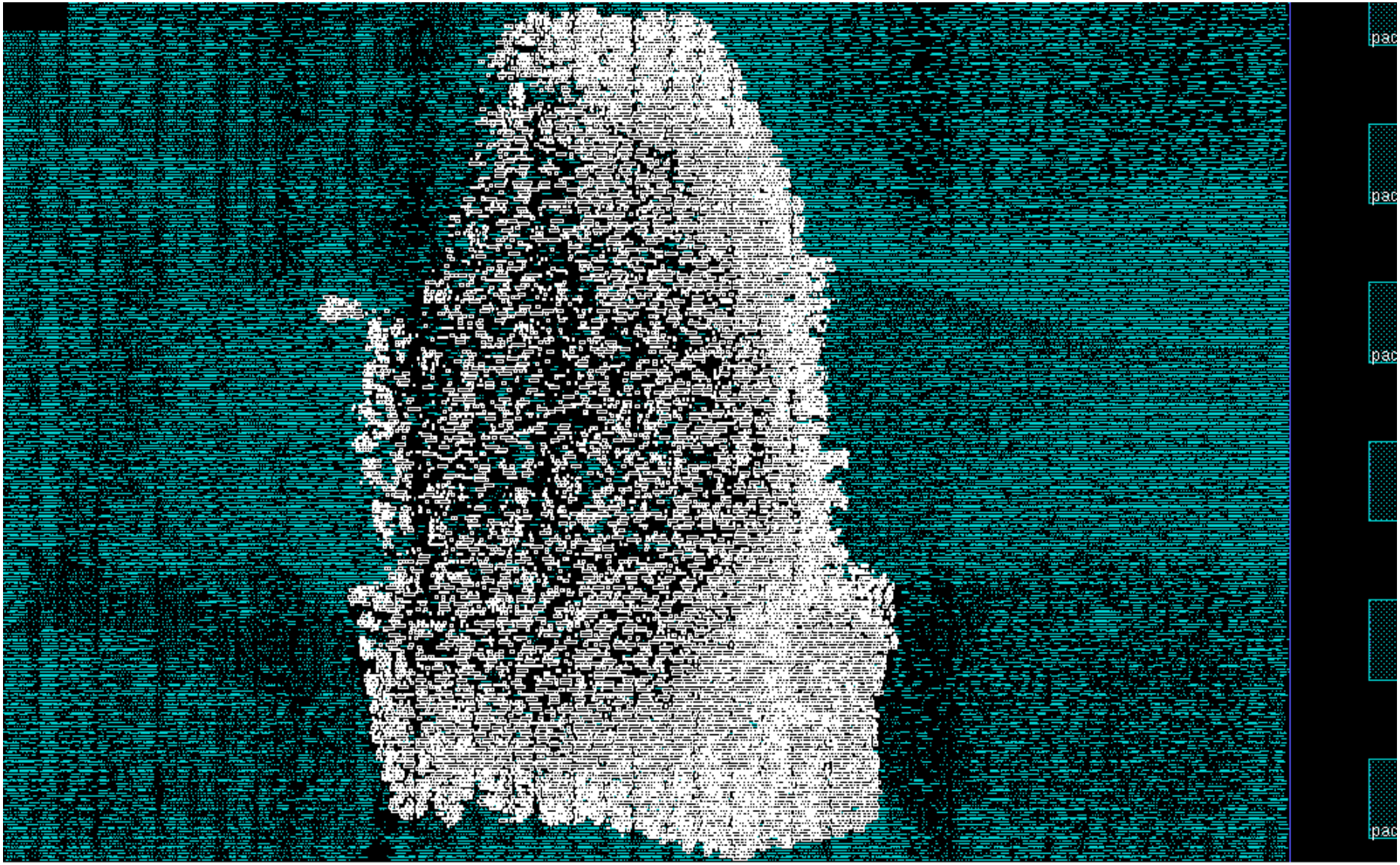
Area of smallest rectangle into which the block fits completely?

What is the Real Area of this Block?



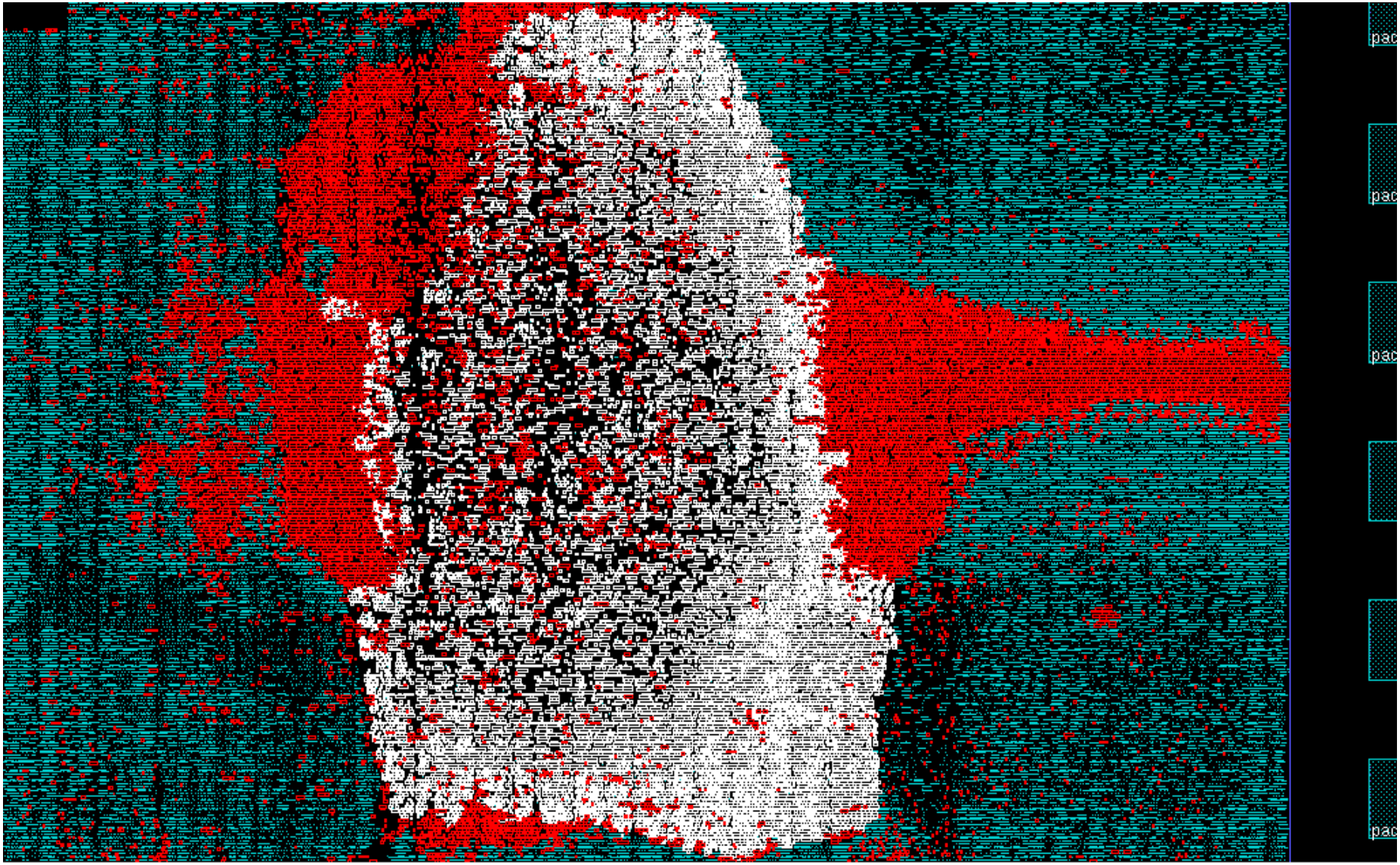
Area covering the maximum reach of the cells?

What is the Real Area of this Block?



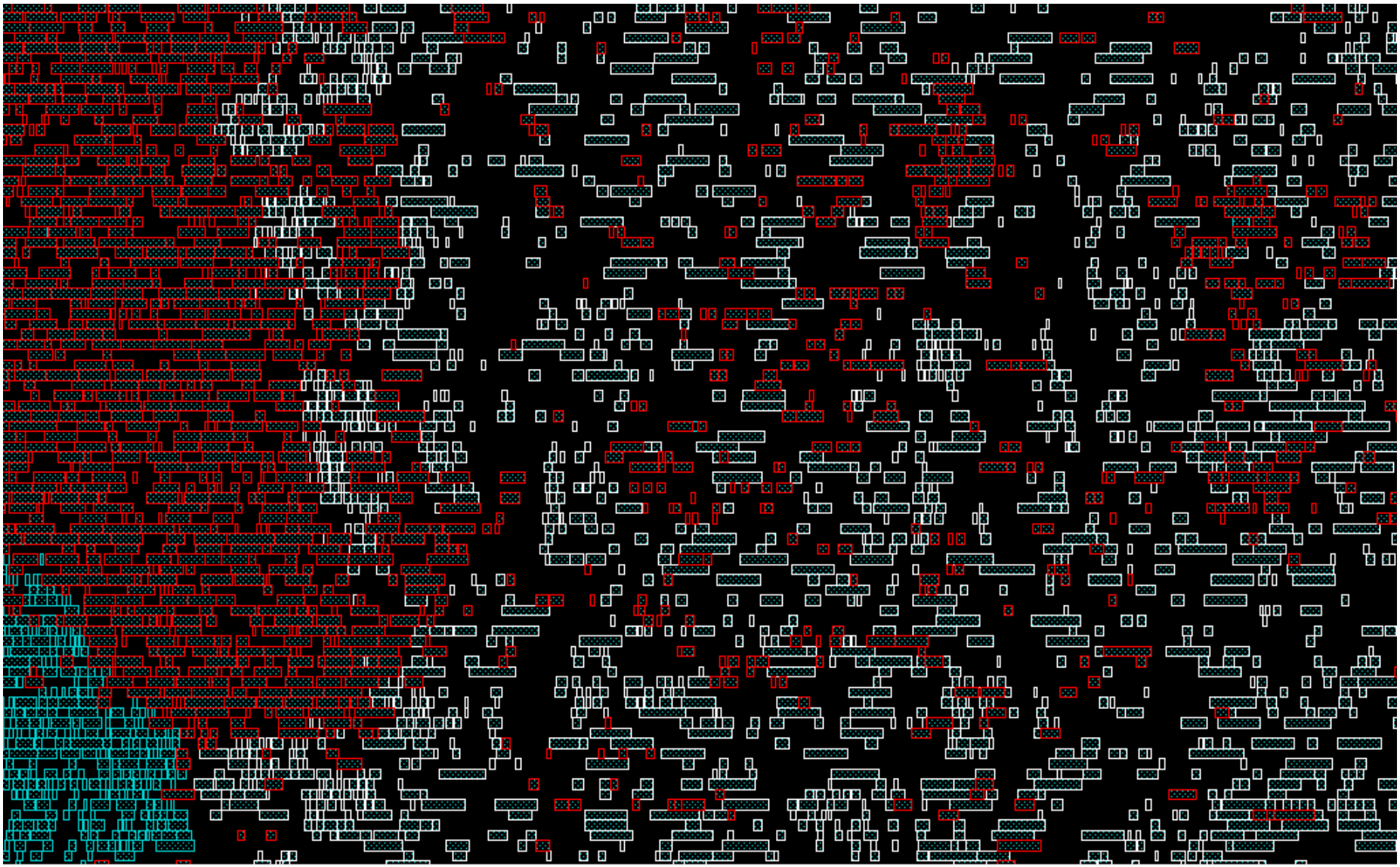
Close-up of the same block

What is the Real Area of this Block?



Block of interest shares area with a second block (in red)

What is the Real Area of this Block?



Closer look shows cells of two blocks with empty area in between

Area Numbers are Approximate

Determining the area is not so easy

- **Synthesis numbers do not tell the whole story**
  - Routing, power overhead is **design dependent**.
  - Can be only 10 % (LFSR)
  - Can be 500 % (LDPC decoder)
- **Postlayout numbers are more reliable**
  - Can still be misleading
  - There are many factors (IR drop, crosstalk) that need to be taken care of
  - The higher the clock rate, the more additional problems you get
- **The total chip area is easy to determine**
  - Not often the case that the whole chip is of interest
  - Difficult to determine the exact area of one block within a chip

What About Speed?

How fast is my circuit?

What About Speed?

How fast is my circuit?

- **Throughput?**
  - How many data items can we process per unit time
  - Do more per unit time (*parallelization*)
  - Divide the process into multiple shorter steps (*pipelining*)
  - Do each processing step faster (*increase clock frequency*)
- **Latency?**
  - Amount of time required to process one data item
  - Can not be made faster by parallelization, and pipelining
  - Necessary for feedback systems

Units that tell us the Speed:

- **Clock frequency [MHz]**
  - Determined by the critical path
  - All other things being equal, would compare actually the speed
  - How much is done in one clock cycle can differ from circuit to circuit

Units that tell us the Speed:

- **Clock frequency [MHz]**
  - Determined by the critical path
  - All other things being equal, would compare actually the speed
  - How much is done in one clock cycle can differ from circuit to circuit
- **Clock period [ns]**
  - 1/clock frequency
  - Is more physical, relates to the sum of gate delays in the critical path

Units that tell us the Speed:

- **Clock frequency [MHz]**
  - Determined by the critical path
  - All other things being equal, would compare actually the speed
  - How much is done in one clock cycle can differ from circuit to circuit
- **Clock period [ns]**
  - 1/clock frequency
  - Is more physical, relates to the sum of gate delays in the critical path
- **Throughput [bits/s]**
  - How many bits are processed per unit time
  - Good for streaming architectures, bad for processors

Units that tell us the Speed:

- **Clock frequency [MHz]**
  - Determined by the critical path
  - All other things being equal, would compare actually the speed
  - How much is done in one clock cycle can differ from circuit to circuit
- **Clock period [ns]**
  - 1/clock frequency
  - Is more physical, relates to the sum of gate delays in the critical path
- **Throughput [bits/s]**
  - How many bits are processed per unit time
  - Good for streaming architectures, bad for processors
- **(FP) Operations per second (GFLOPS) for processors**
  - Could be given as peak, minimum or mean values
  - Depends on the complexity of the operation (i.e. `NOP`)

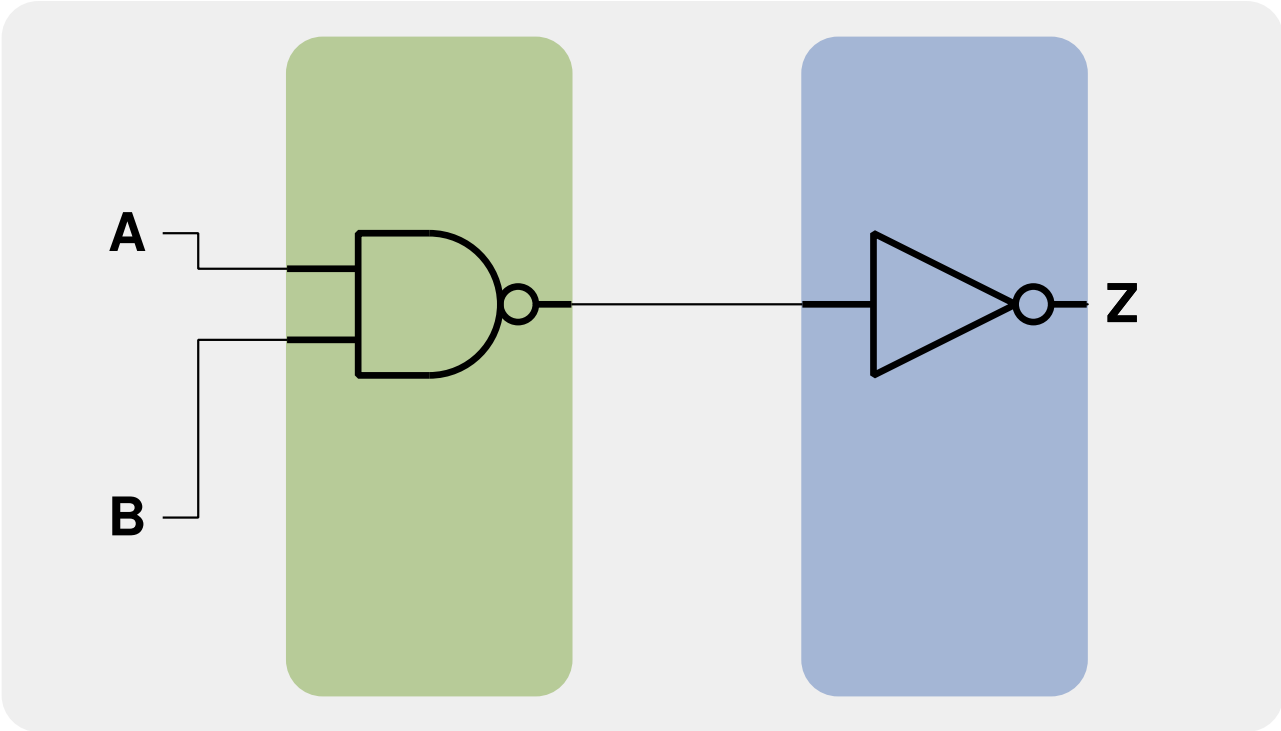
Units that tell us the Speed:

- **Clock frequency [MHz]**
  - Determined by the critical path
  - All other things being equal, would compare actually the speed
  - How much is done in one clock cycle can differ from circuit to circuit
- **Clock period [ns]**
  - 1/clock frequency
  - Is more physical, relates to the sum of gate delays in the critical path
- **Throughput [bits/s]**
  - How many bits are processed per unit time
  - Good for streaming architectures, bad for processors
- **(FP) Operations per second (GFLOPS) for processors**
  - Could be given as peak, minimum or mean values
  - Depends on the complexity of the operation (i.e. `NOP`)
- **Clocks per instruction (CPI) for processors**
  - Reflects the parallelization of the instructions.
  - Needs the clock frequency to determine actual performance

Timing Analysis Investigates All Paths

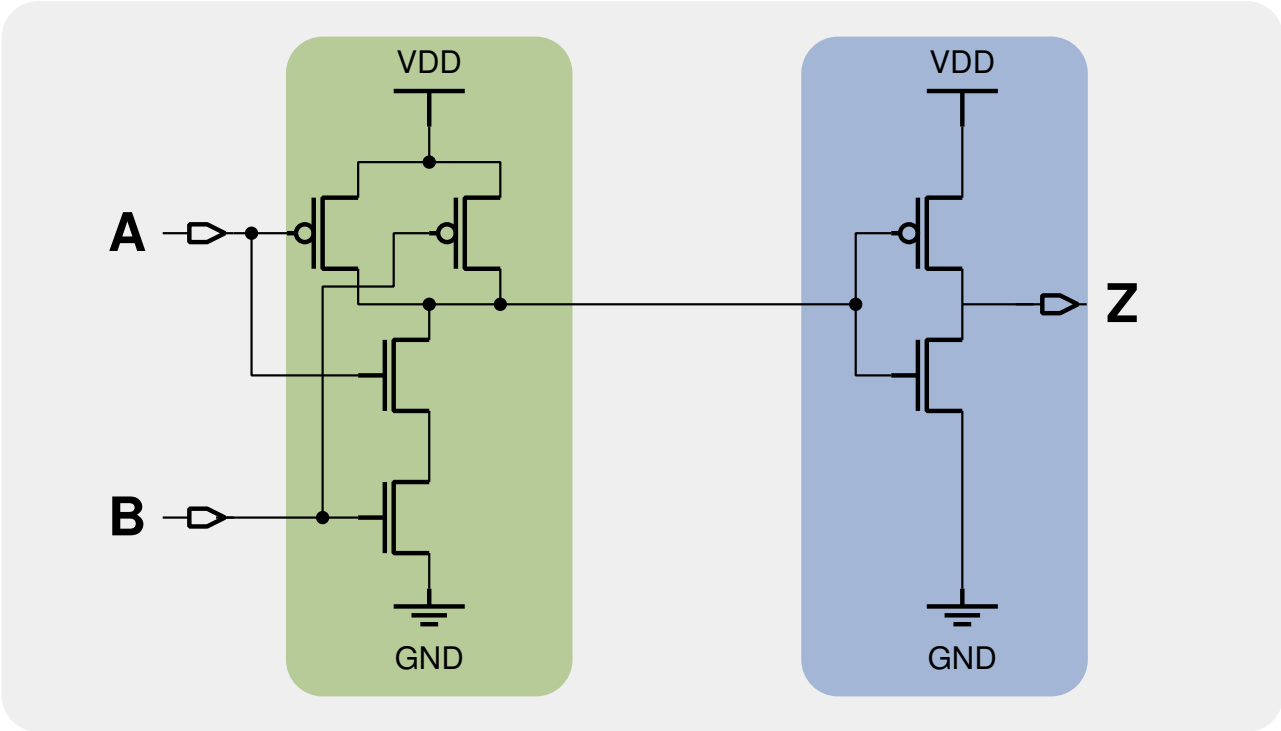
```
Des/Clust/Port     Wire Load Model          Library
------------------------------------------------------------
sbox               enG5K                    generic_lib

Point                                          Incr       Path
------------------------------------------------------------
InpxDI[0] (in)                                 0.0987     0.0987 f
i_s_lut_0100/AddrxDI[0] (sub_lut_0100)         0.0000     0.0987 f
i_s_lut_0100/U1/O (INV1S)                      0.4191     0.5178 r
i_s_lut_0100/U53/O (NR2)                       0.3683     0.8861 f
i_s_lut_0100/U83/O (NR2)                       0.4559     1.3420 r
i_s_lut_0100/U25/O (INV1S)                     0.4845     1.8265 f
i_s_lut_0100/U140/O (AOI22S)                   0.2411     2.0676 r
i_s_lut_0100/U19/O (ND3S)                      0.0821     2.1497 f
i_s_lut_0100/U225/O (MOAI1S)                   0.1802     2.3299 f
i_s_lut_0100/U226/O (AOI112S)                  0.2397     2.5696 r
i_s_lut_0100/U227/O (OAI222S)                  0.1254     2.6951 f
i_s_lut_0100/DataxDO[7] (sub_lut_0100)         0.0000     2.6951 f
U16/O (MUX2)                                   0.2020     2.8971 f
OupxDO[7] (out)                                0.0000     2.8971 f
data arrival time                                         2.8971
------------------------------------------------------------
```

CMOS Timing Depends on Capacitive Loading



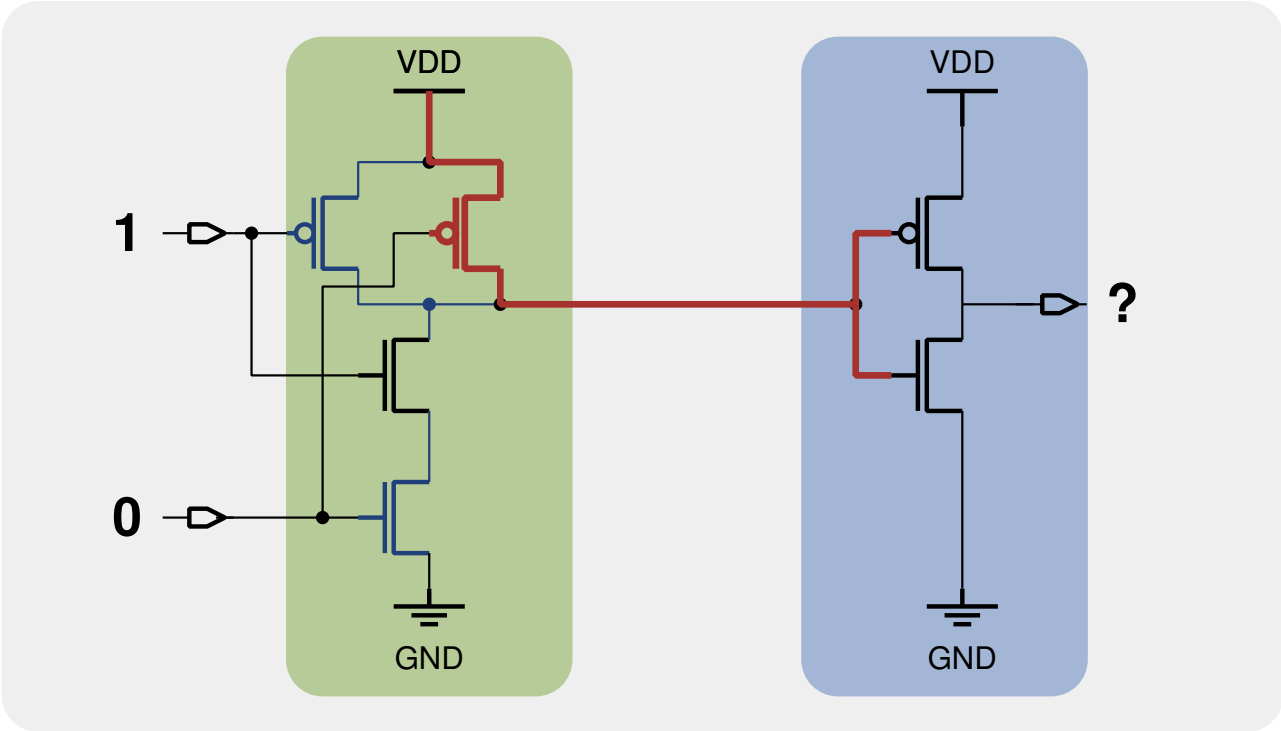
Simple Example:
2-input NAND gate driving an inverter

CMOS Timing Depends on Capacitive Loading



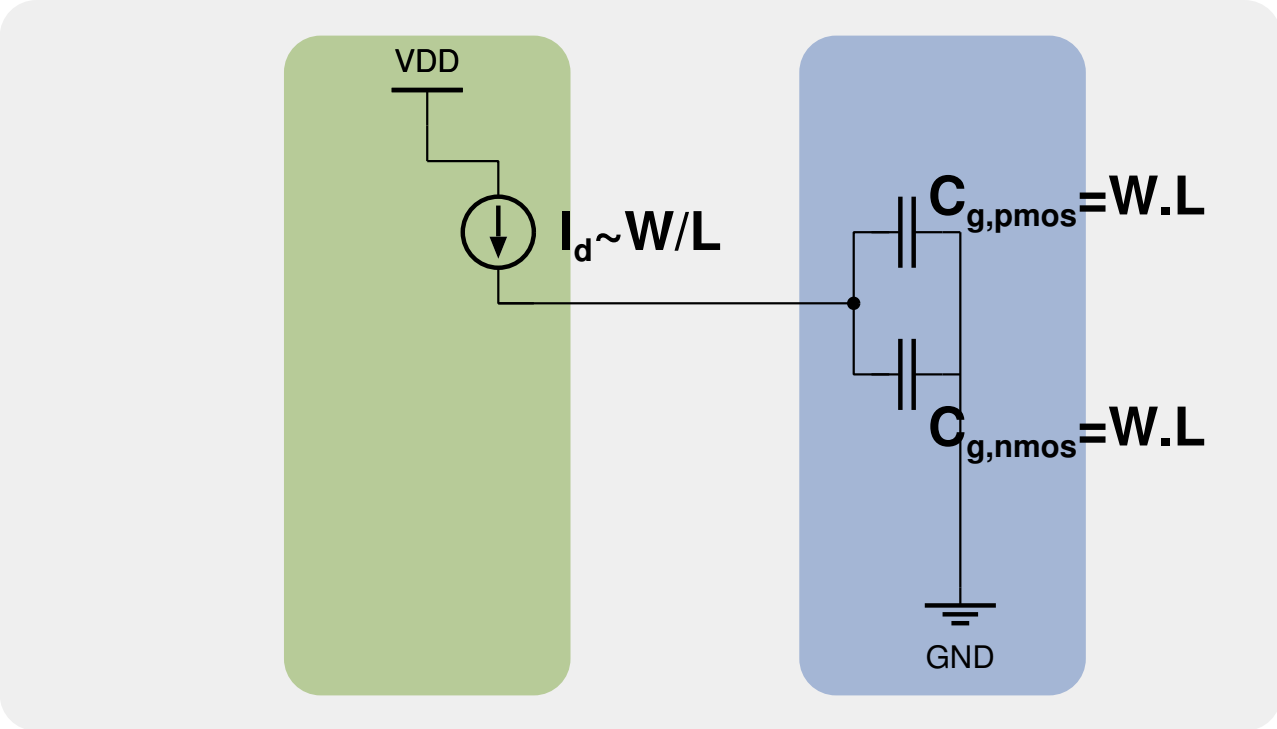
Transistor level schematic:
The gates when implemented using CMOS logic

CMOS Timing Depends on Capacitive Loading



For example:
When A=1, B=0, one pMOS charges up the input of the inverter

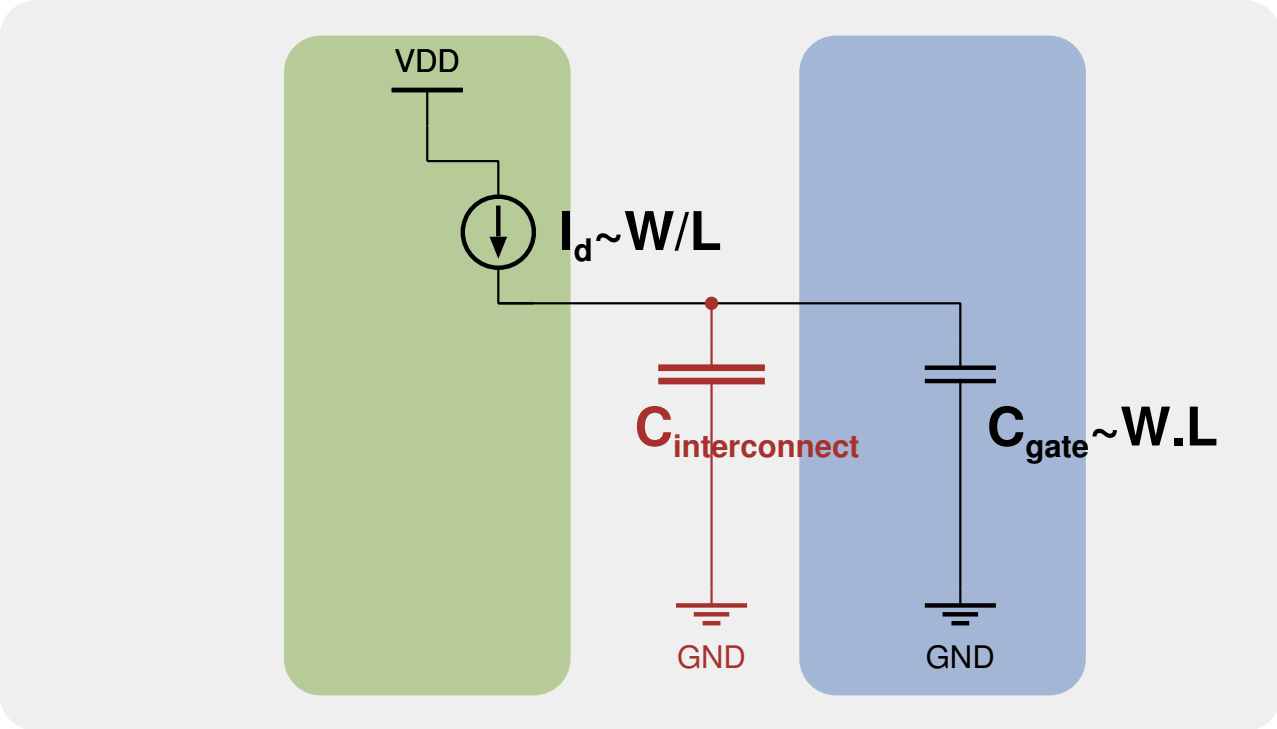
CMOS Timing Depends on Capacitive Loading



In ideal case, it is a current source driving two capacitors

Faster circuit = **larger current source** or **smaller capacitance**

CMOS Timing Depends on Capacitive Loading



Parasitic interconnect capacitance is major contributor to delay

Not possible to determine the speed if $C_{interconnect}$ is not known

How Do I Know What the Parasitic Load Is?

- **Ignore parasitic loads**
  - Naive approach
  - Especially in modern processes ($<$90 nm) parasitics are dominant

How Do I Know What the Parasitic Load Is?

- **Ignore parasitic loads**
  - Naive approach
  - Especially in modern processes ($<$90 nm) parasitics are dominant
- **Wireload models for synthesis**
  - Statistical lookup tables to estimate the load
  - Capacitive load as a function of the output fanout
  - Circuit dependent, for best results needs to be extracted for each circuit

How Do I Know What the Parasitic Load Is?

- **Ignore parasitic loads**
  - Naive approach
  - Especially in modern processes ($<$90 nm) parasitics are dominant
- **Wireload models for synthesis**
  - Statistical lookup tables to estimate the load
  - Capacitive load as a function of the output fanout
  - Circuit dependent, for best results needs to be extracted for each circuit
- **Parasitic extraction**
  - Only after placement and routing has been done
  - All interconnections are known, capacitance can be extracted
  - Different accuracy levels: 2D, 2.5D, full 3D
  - Results are used by the timing analysis engine to calculate delay

Impact of Process Voltage Temperature Corners

	Worst Case	Typical Case	Best Case
Voltage	1.08 V	1.2 V	1.32 V
Temperature	125 °C	27 °C	-40 °C
Critical Path	3.49 ns	2.24 ns	1.59 ns
Throughput	13.75 Gbit/s	21.42 Gbit/s	30.19 Gbit/s
Relative Performance	64.2 %	100 %	140.9 %

A crypto-algorithm implemented in the 90 nm UMC process

This is exactly the same circuit, everytime a different PVT corner is loaded and the timing report is repeated

Every Technology has a Speed Range

Example numbers for a 180 nm process

Speed	Num. of Gates	Frequency Range	Notes
Slow	>120	<50 MHz	No problem
Normal	120-50	50-150 MHz	Standard design
Fast	50-20	150-300 MHz	Needs attention
Very Fast	20-10	300-600 MHz	Involved design
Ultra fast	<10	>600 MHz	Full custom design

- For every process there is a speed range that can achieved easily
- Slower circuits do not take advantage of the process capabilities
- Faster circuits need disproportionately more effort for design

Accurate Timing is Difficult to Determine

- **Industry is interested in corner performance**
  - Even in the **worst** case no **setup** violations
  - Even in the **best** case no **hold** violations
  - Statistical models. Large variation between best and worst case
- **Wiring capacitance needs to be considered**
  - Synthesis tools rely on wireload models. Needs to be customized
  - Reliable estimates only after final routing
- **Short periods/Faster clock rates more problematic**
  - **f=1/T**, small changes in critical path, large changes in frequency
  - Problems with Clock distribution (skew), IR drop, thermal problems all increase with faster clocks

Trade-off between Area and Speed

Generally you can always make a circuit faster by sacrificing area

Trade-off between Area and Speed

Generally you can always make a circuit faster by sacrificing area

We say generally because

- It does not scale infinitely,
  there are upper and lower limits

Trade-off between Area and Speed

Generally you can always make a circuit faster by sacrificing area

We say generally because

- It does not scale infinitely,
  there are upper and lower limits
- Not all points are equally efficient

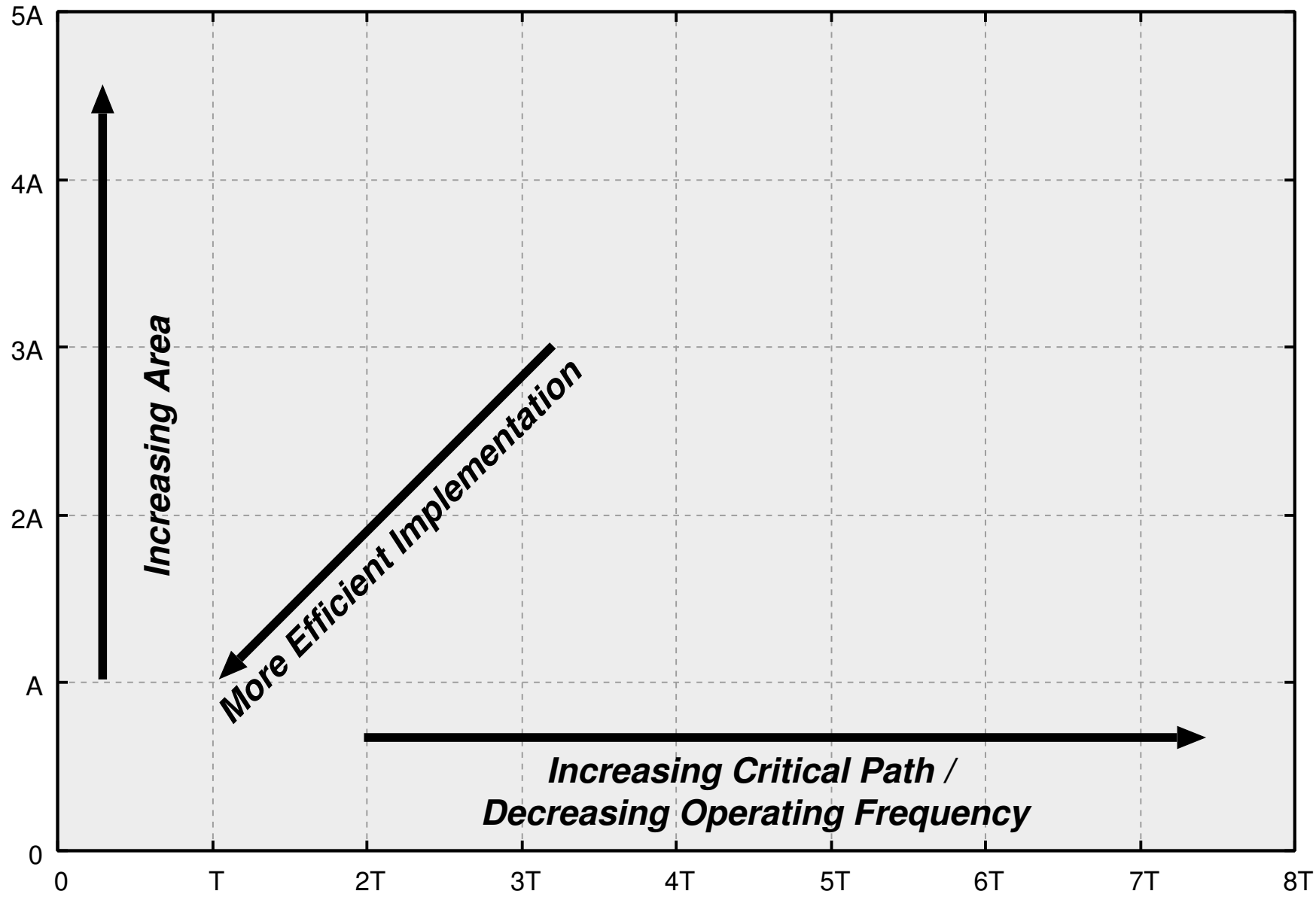
Trade-off between Area and Speed

Generally you can always make a circuit faster by sacrificing area

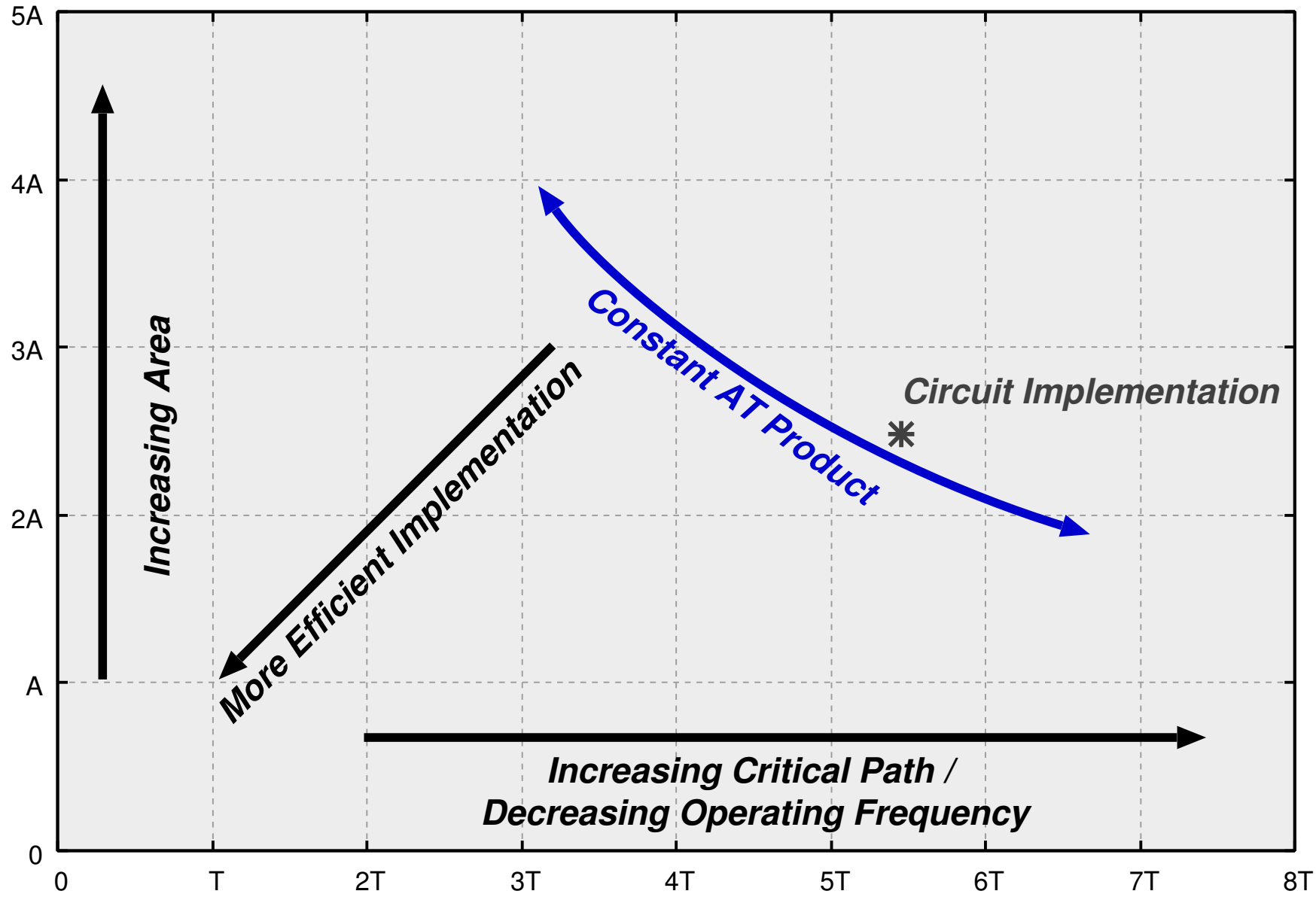
We say generally because

- It does not scale infinitely,
  there are upper and lower limits
- Not all points are equally efficient
- AT graphs visualize this trade-off

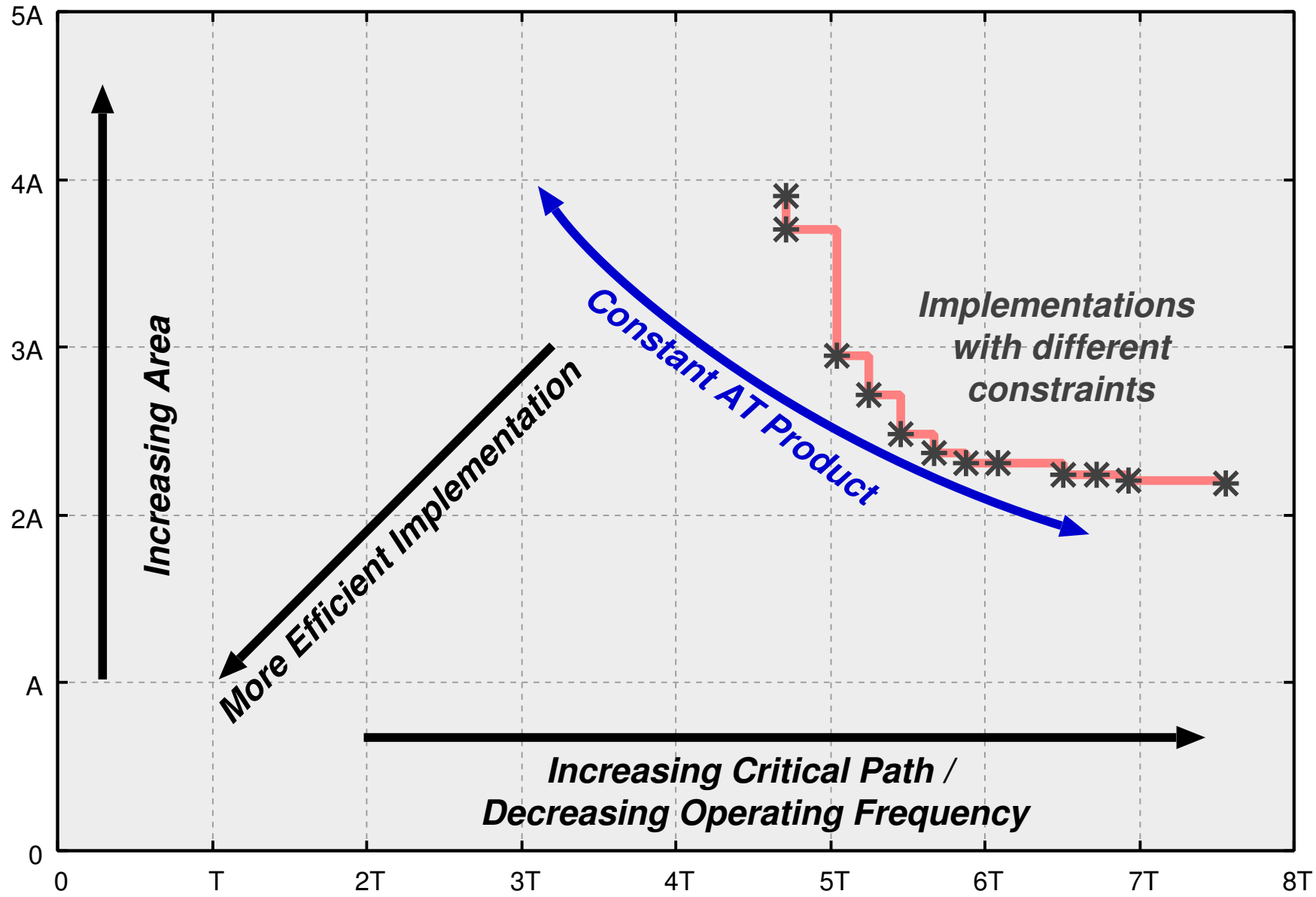
AT Graphs Explained



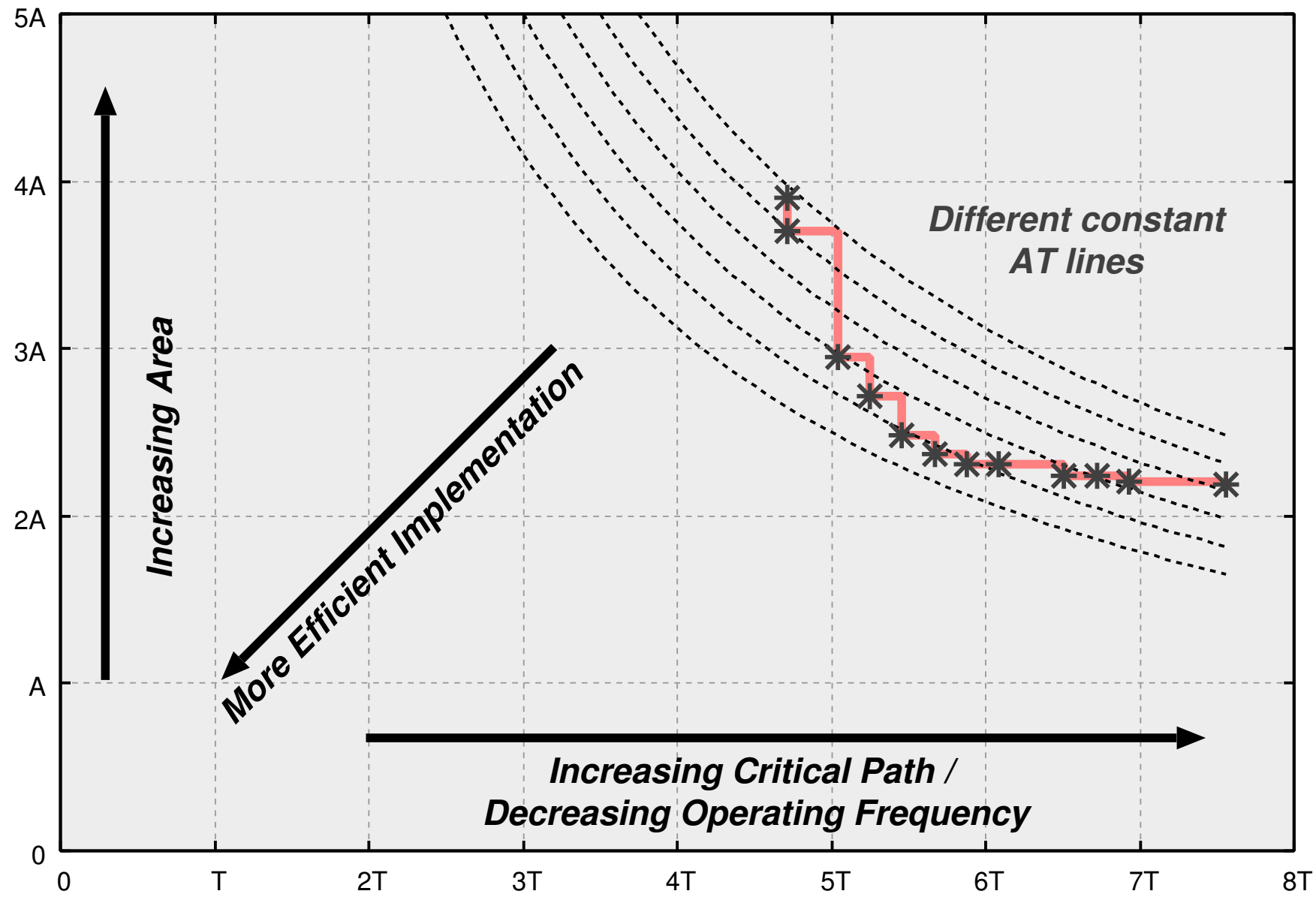
AT Graphs Explained



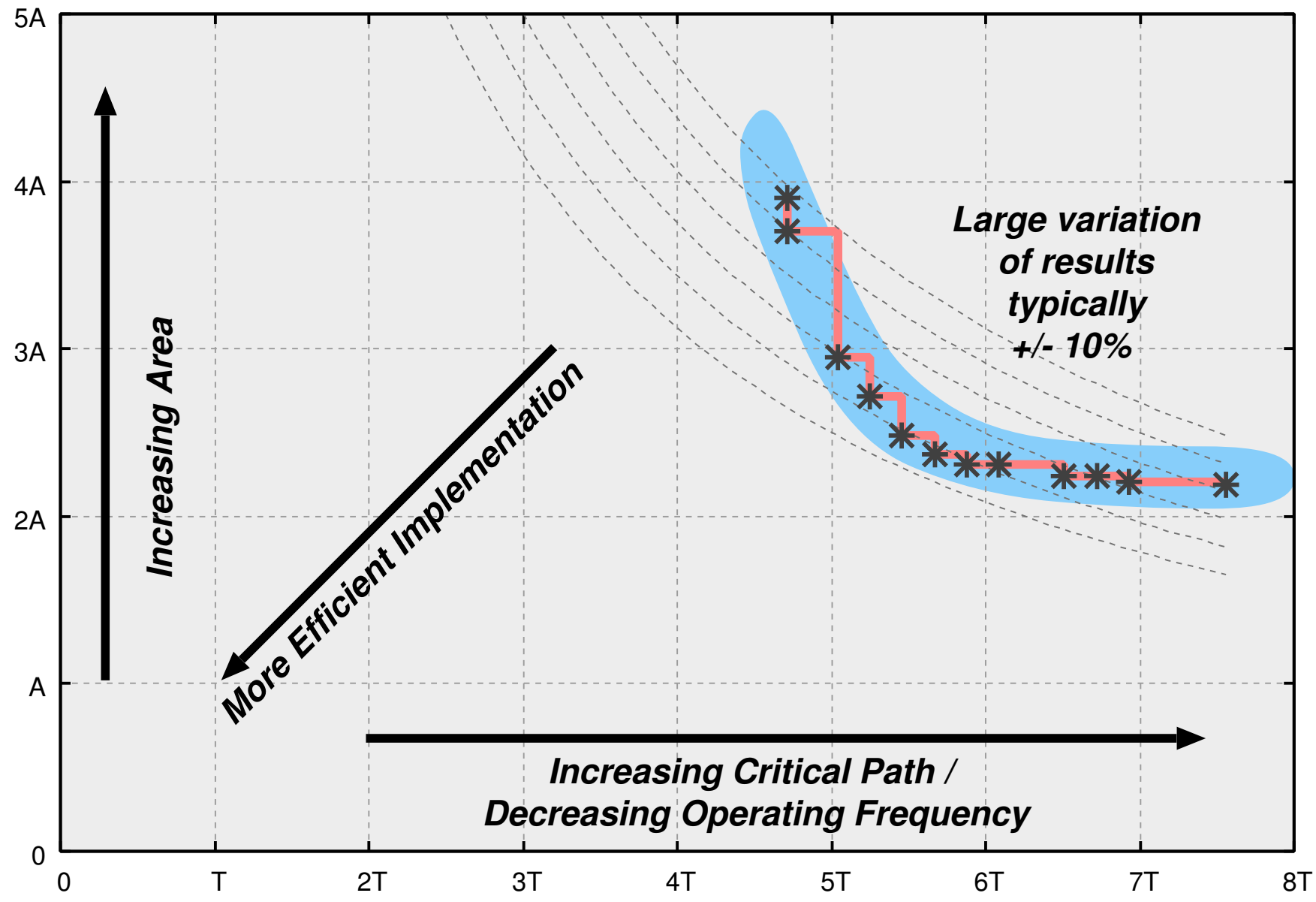
AT Graphs Explained



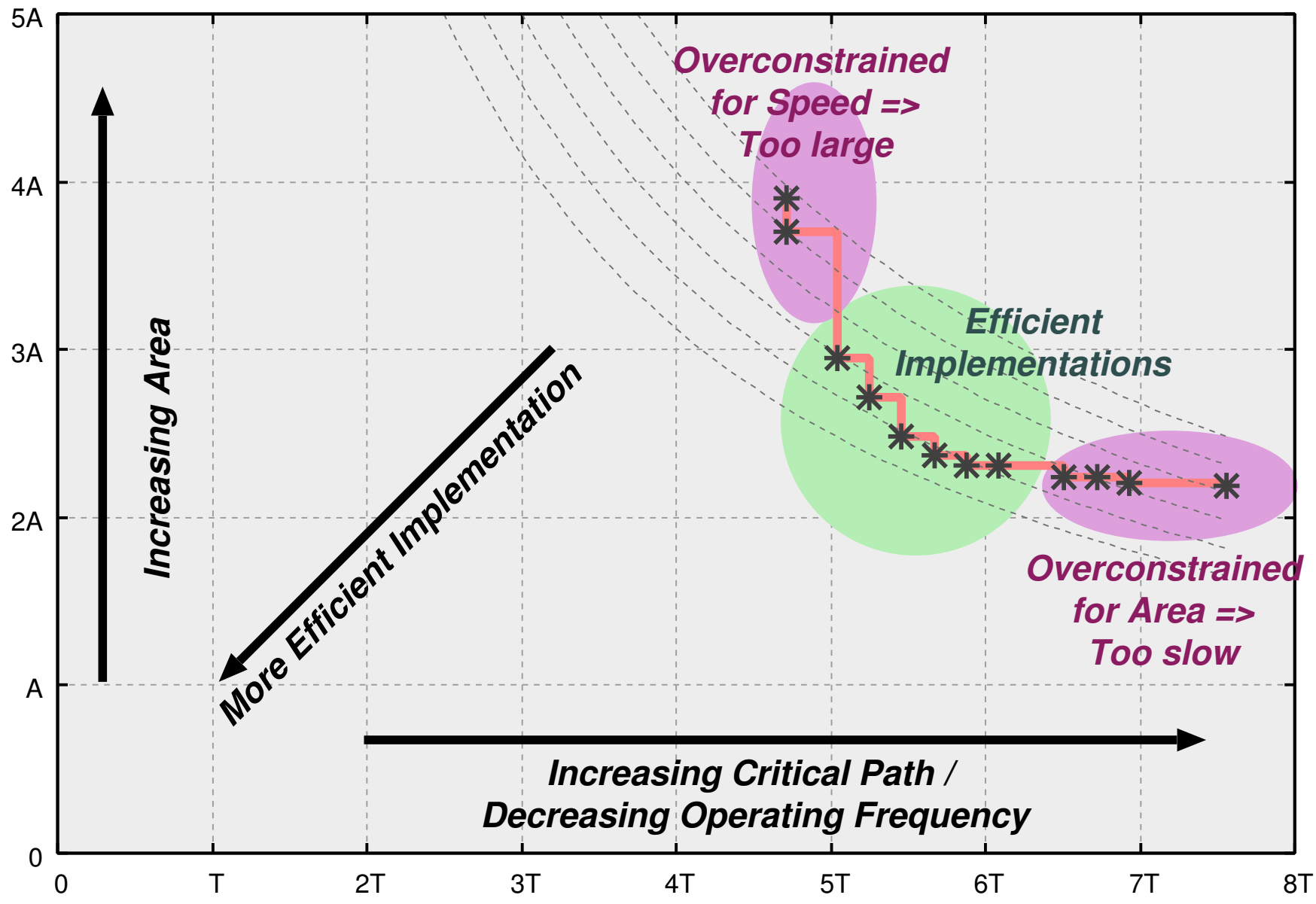
AT Graphs Explained



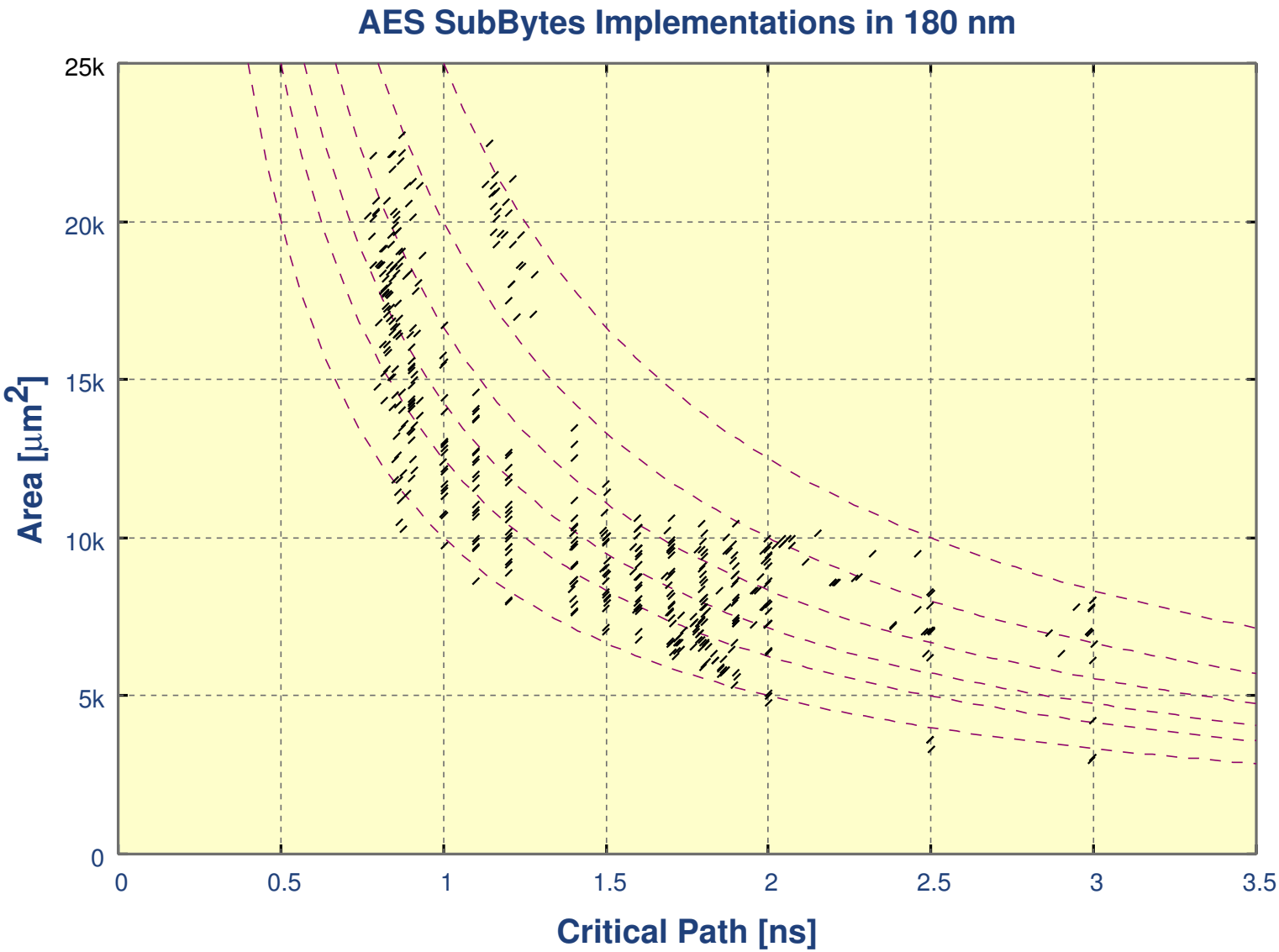
AT Graphs Explained



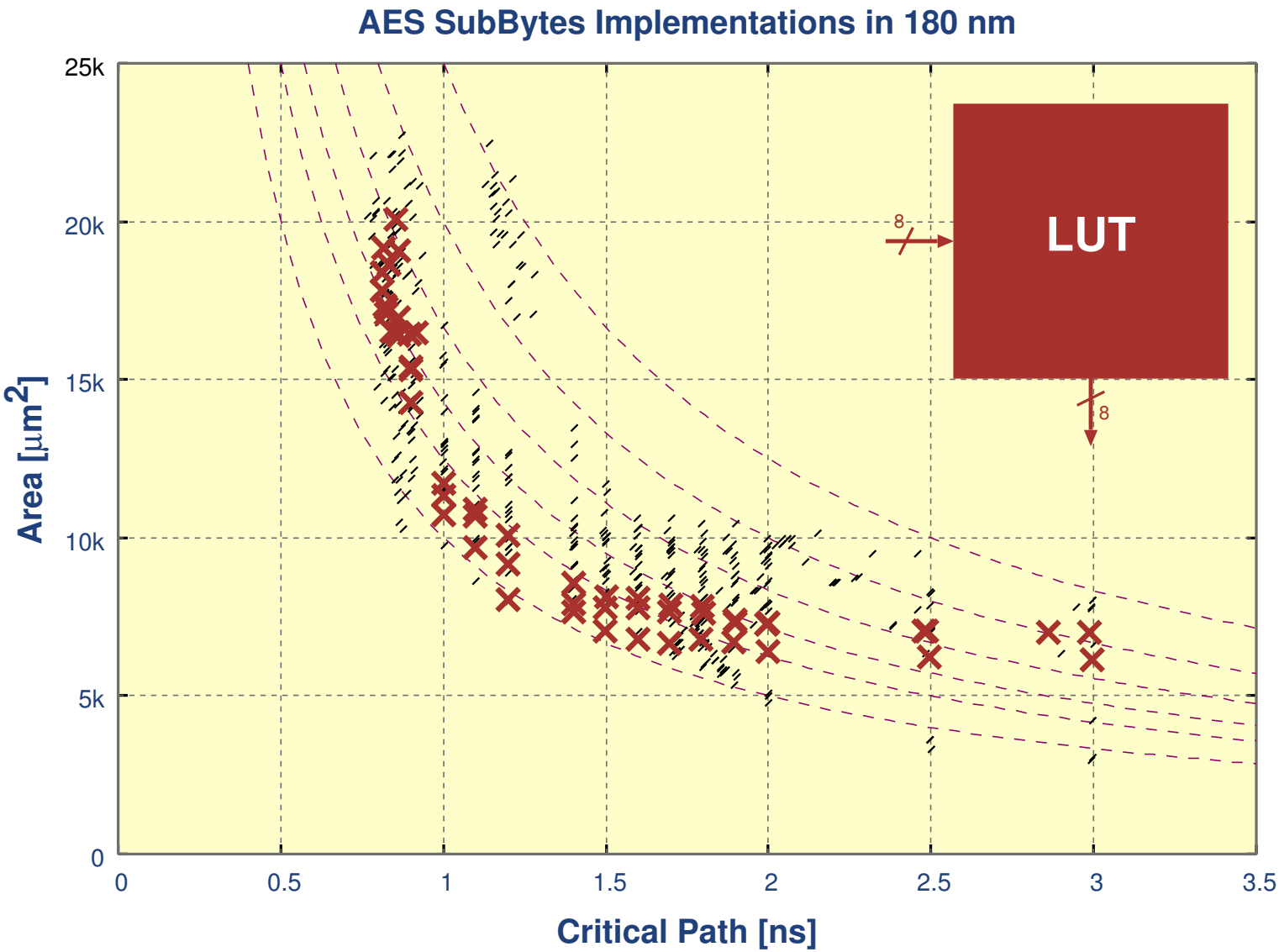
AT Graphs Explained



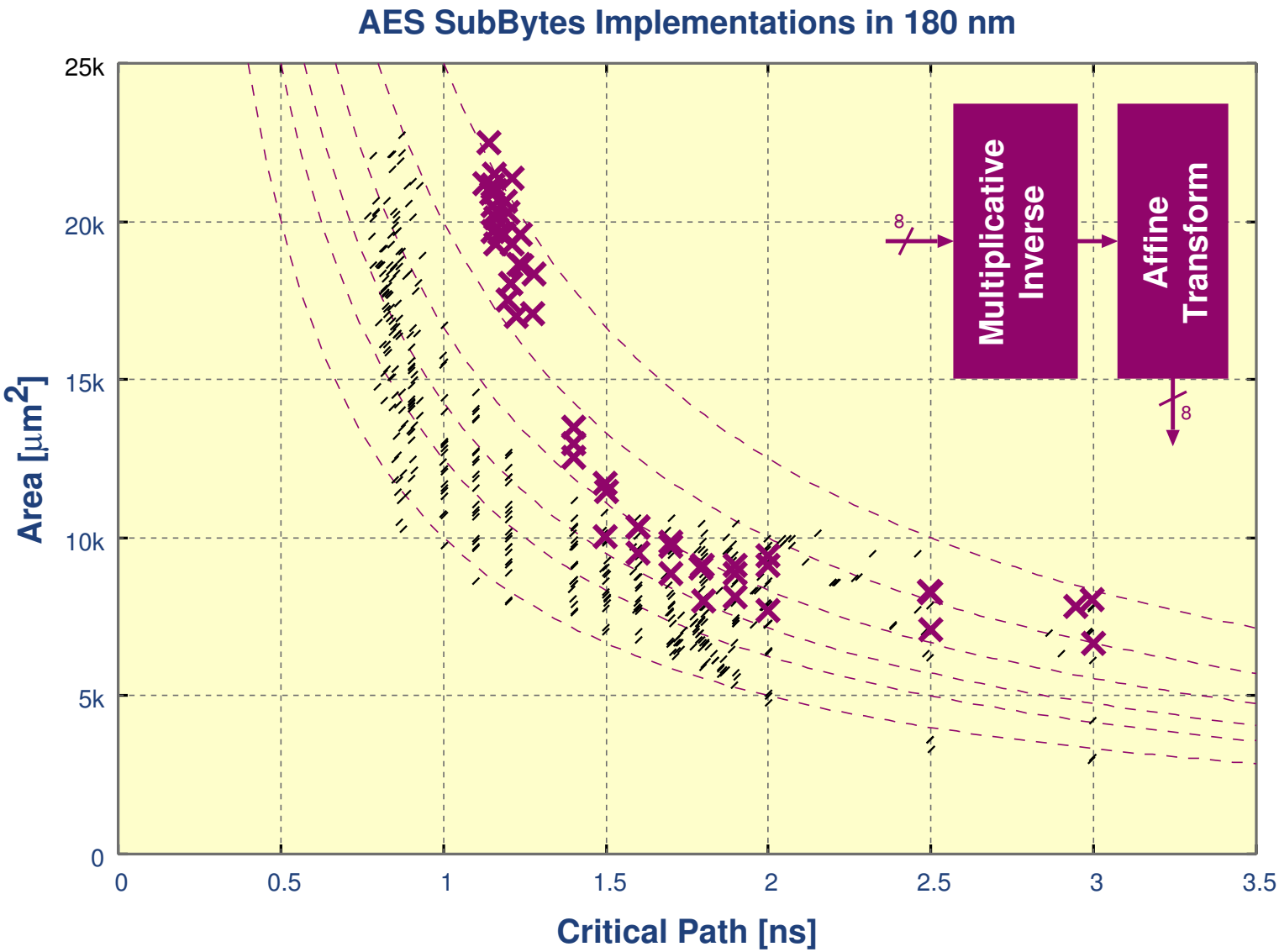
Synthesis Results Depend on The HDL Code



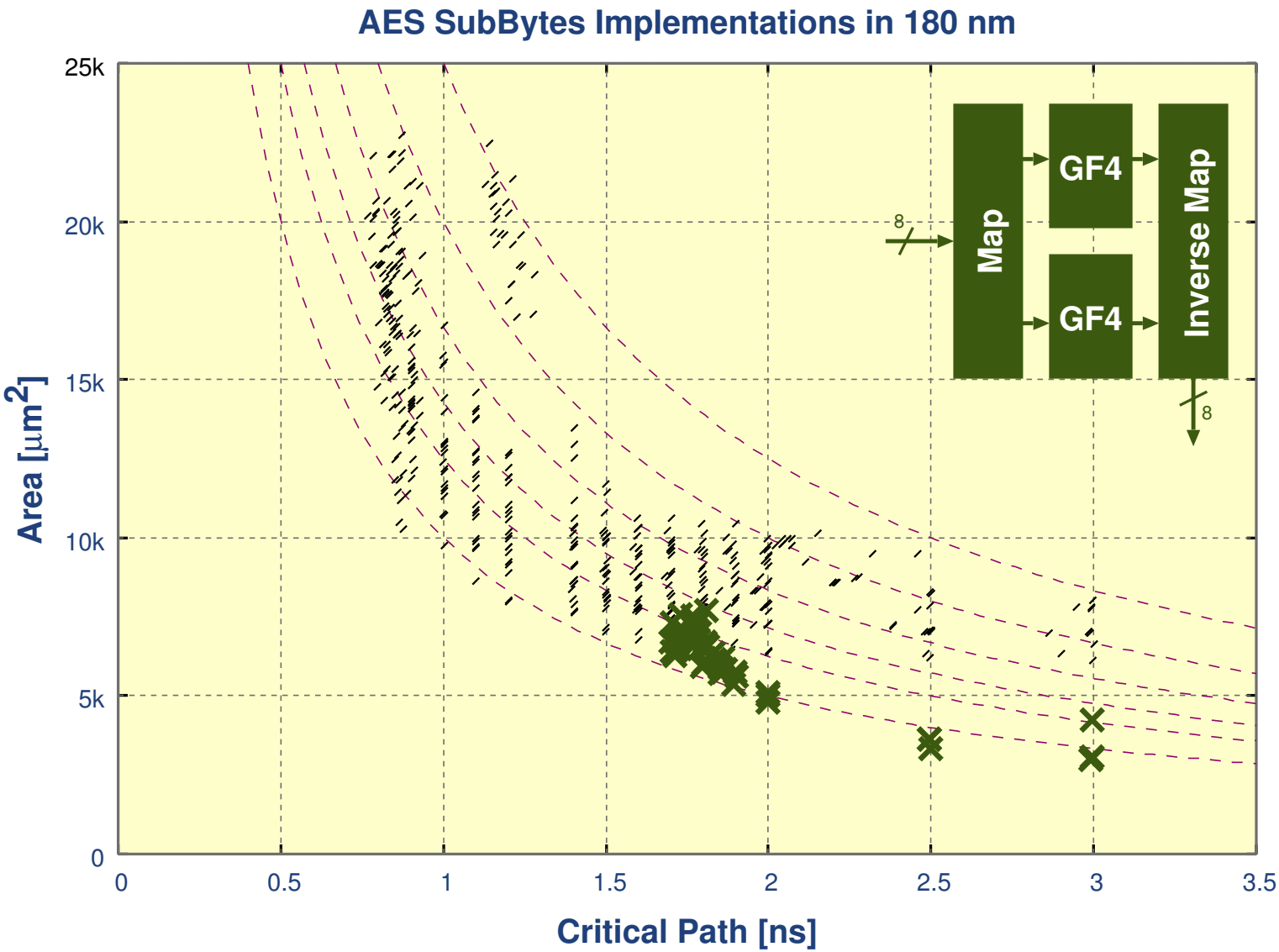
Synthesis Results Depend on The HDL Code



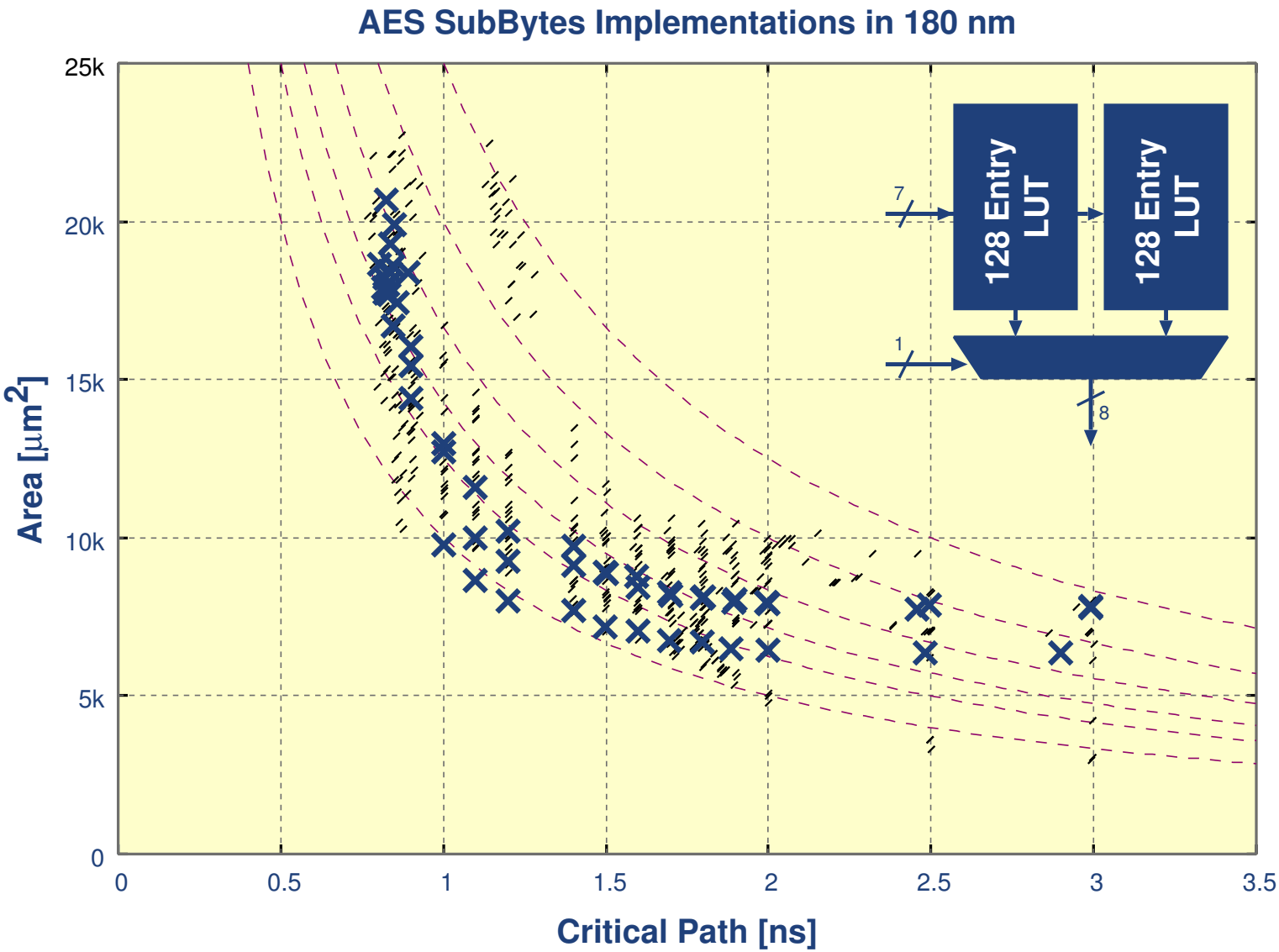
Synthesis Results Depend on The HDL Code



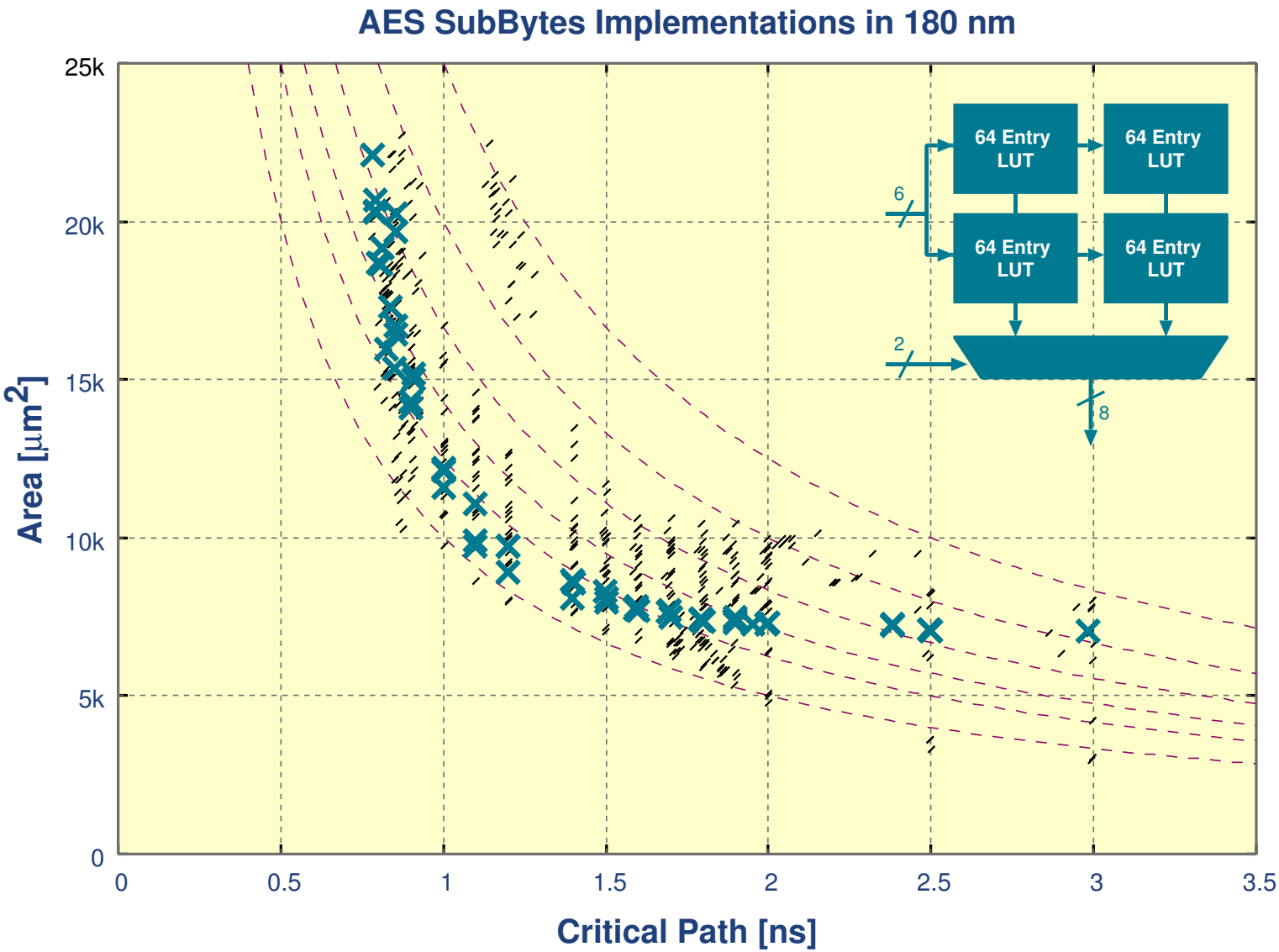
Synthesis Results Depend on The HDL Code



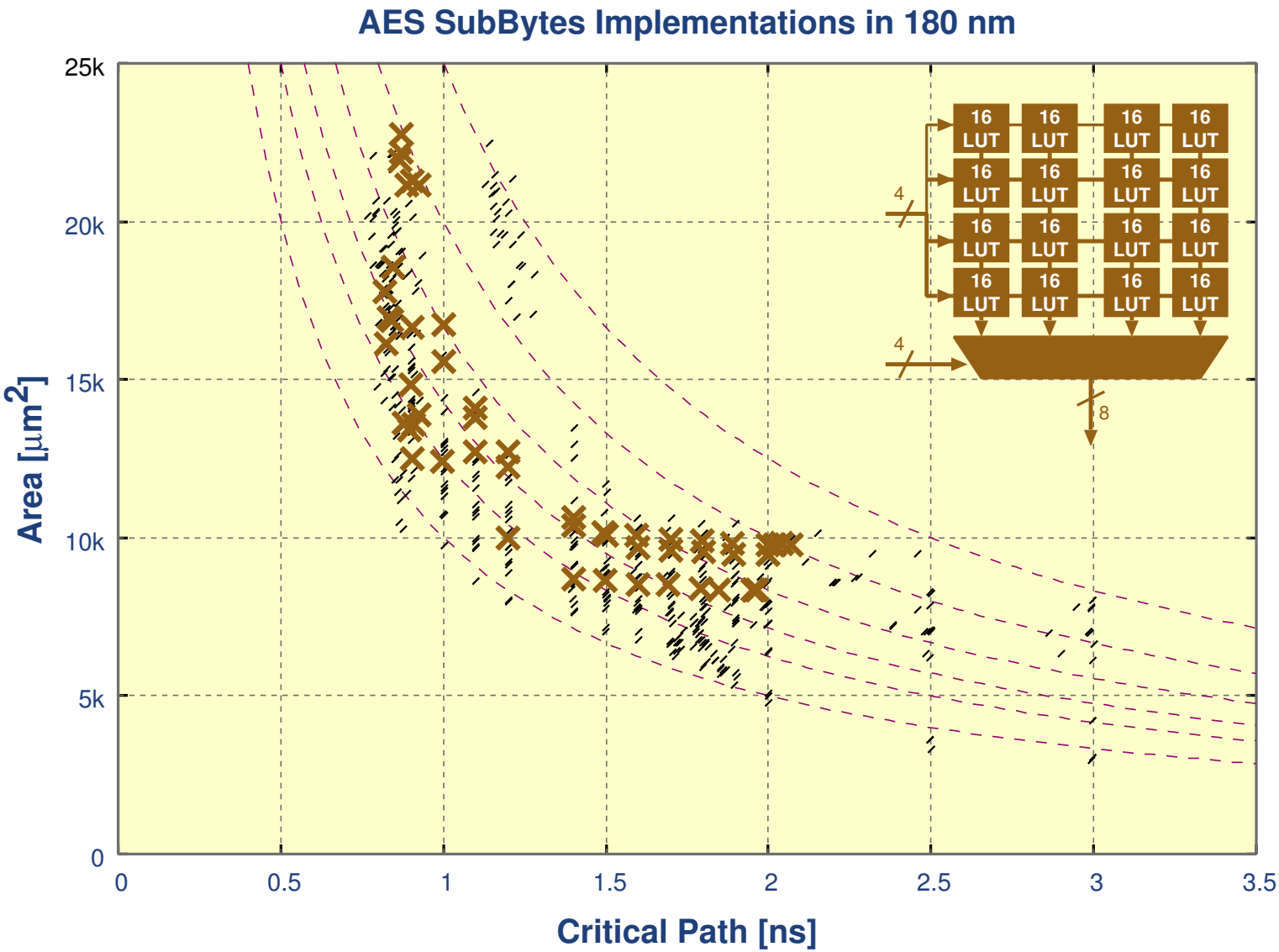
Synthesis Results Depend on The HDL Code



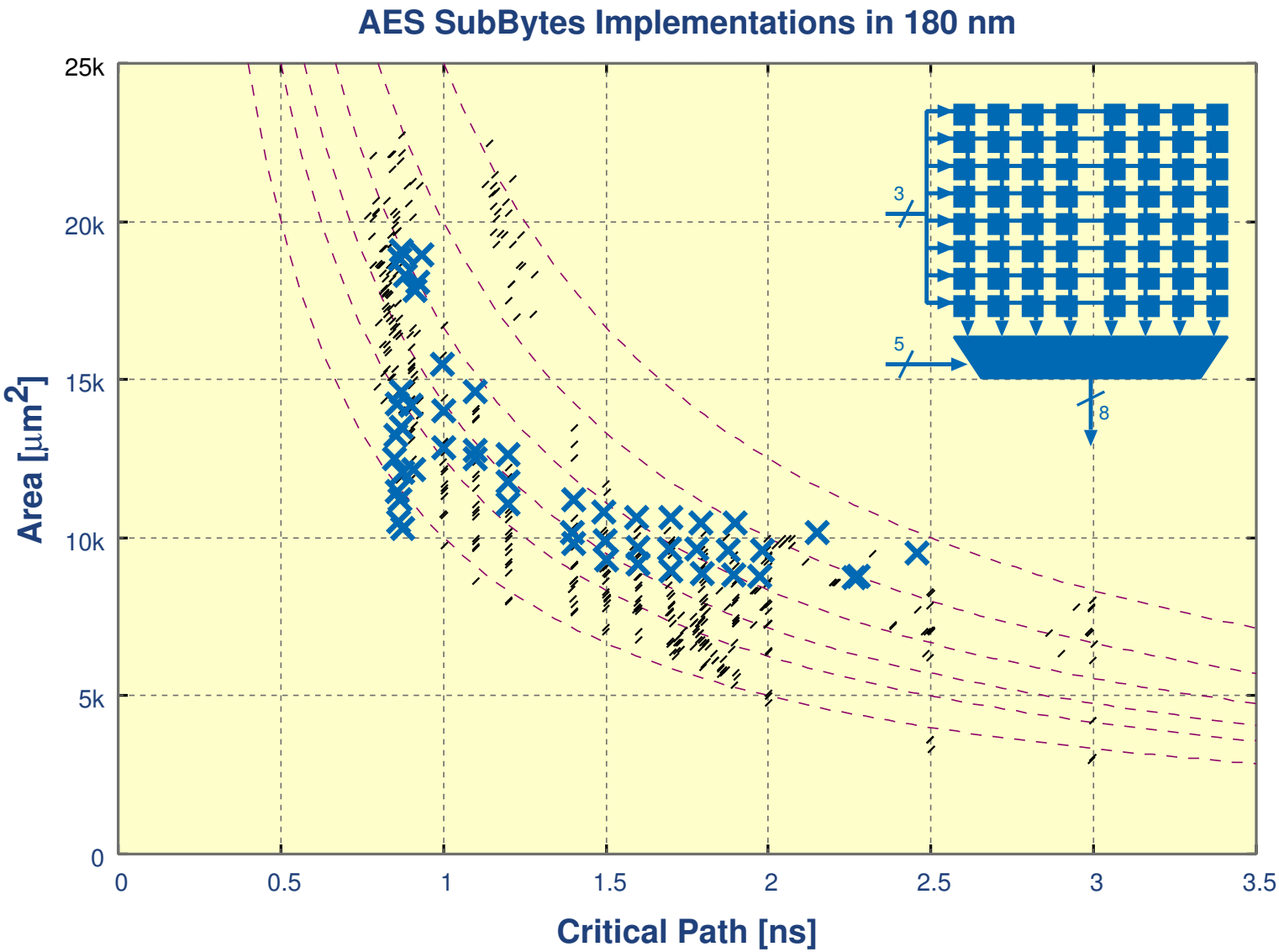
Synthesis Results Depend on The HDL Code



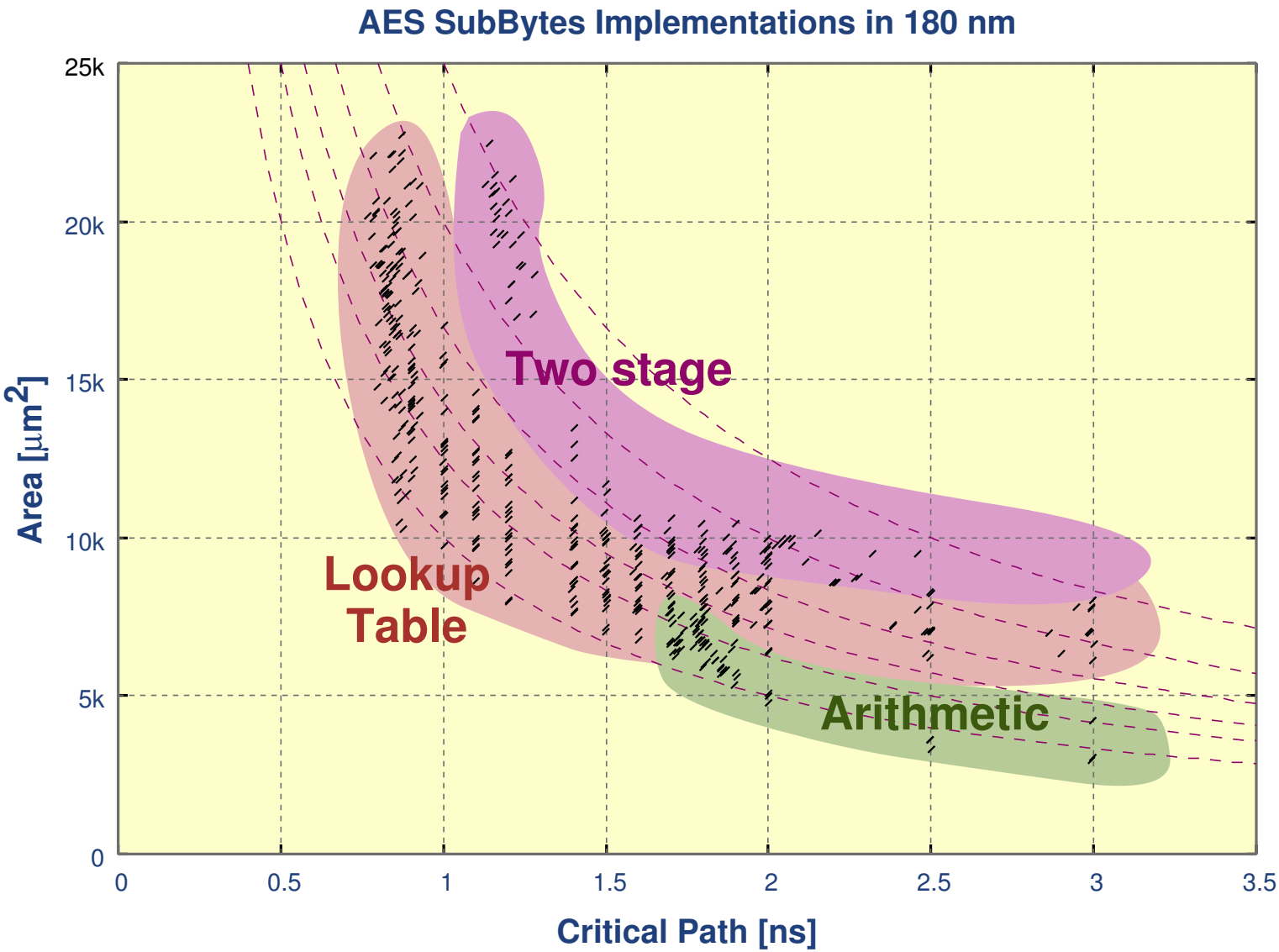
Synthesis Results Depend on The HDL Code



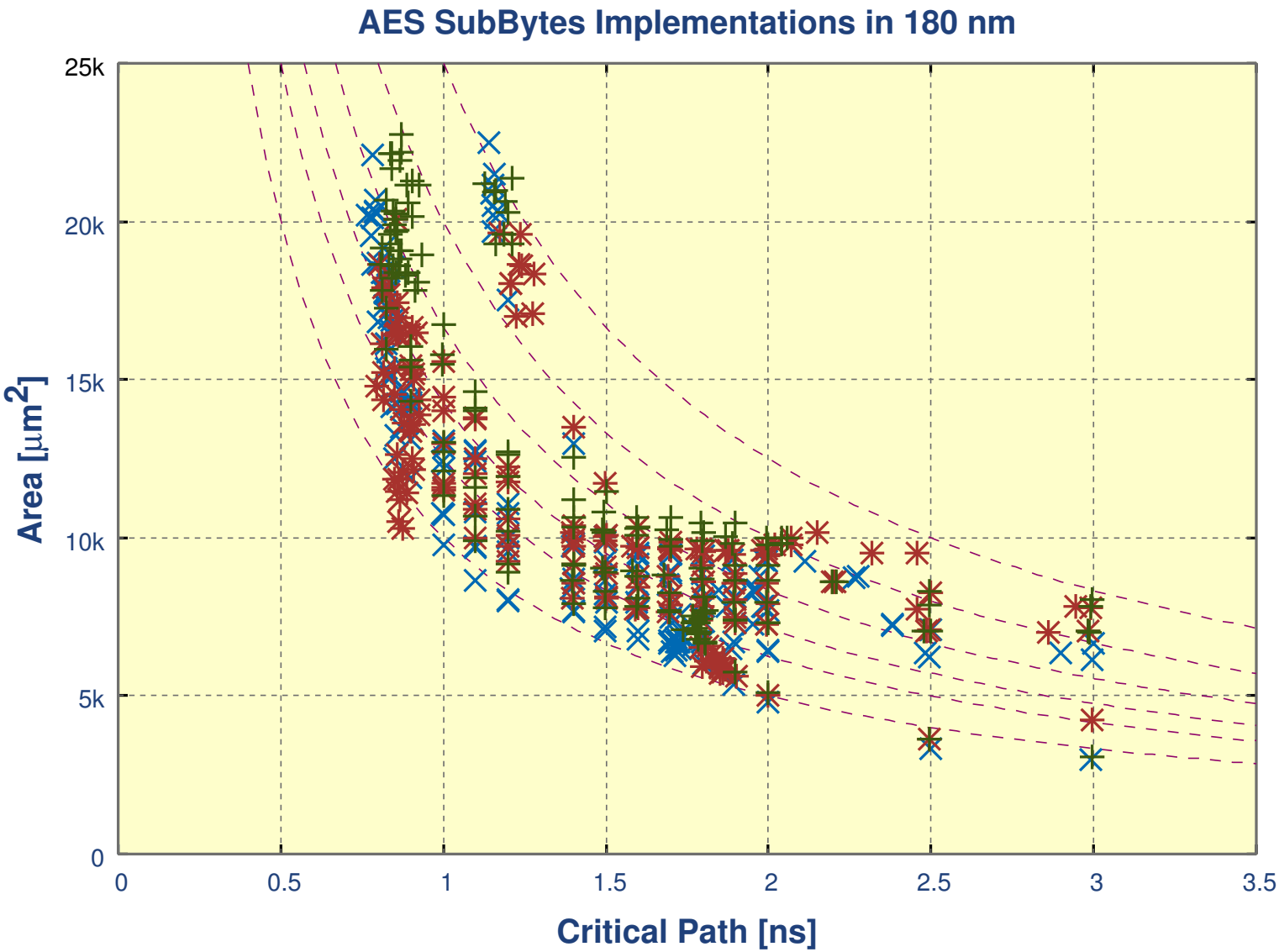
Synthesis Results Depend on The HDL Code



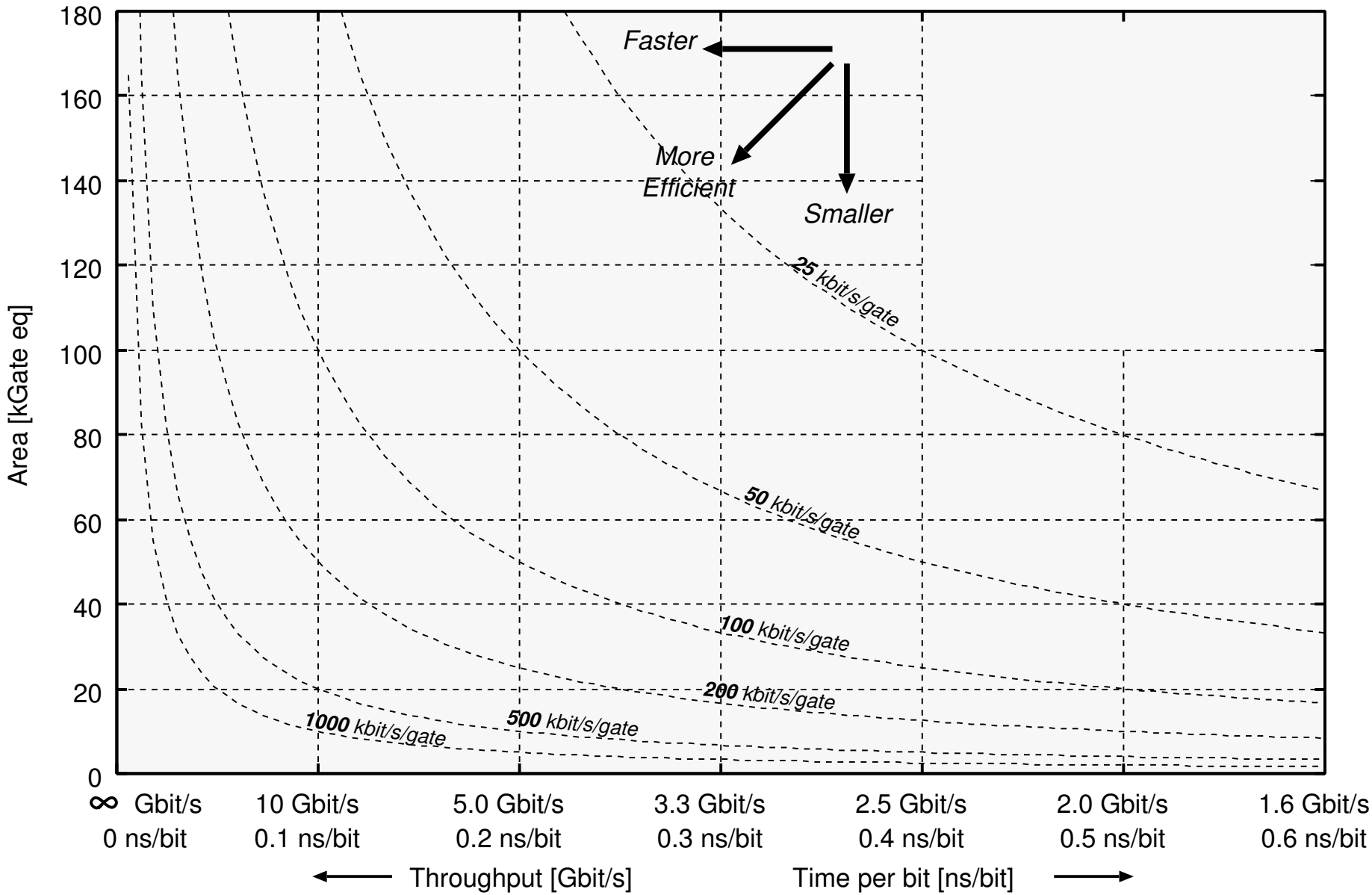
Synthesis Results Depend on The HDL Code



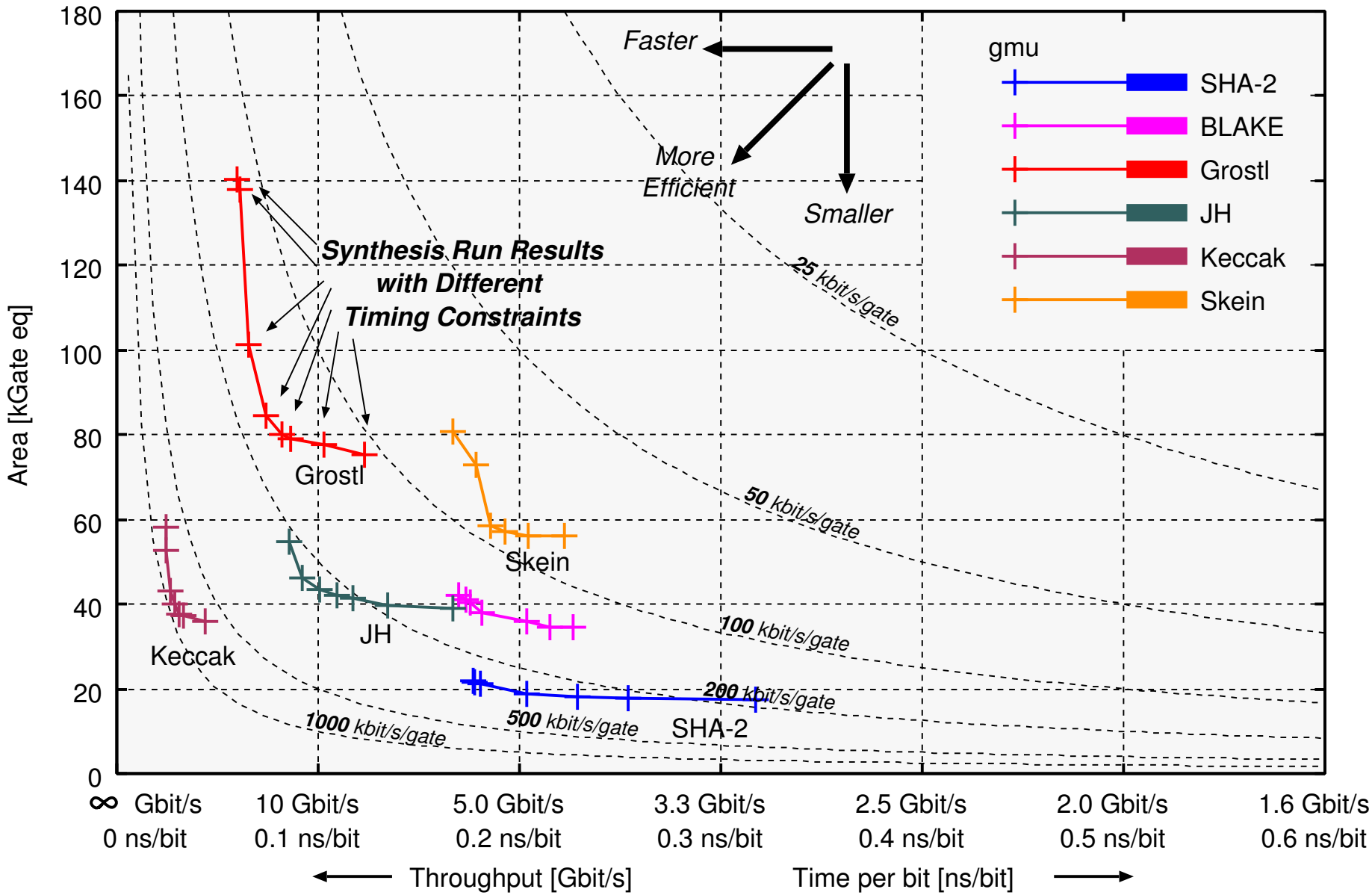
As Well As the Version of Synthesis Software



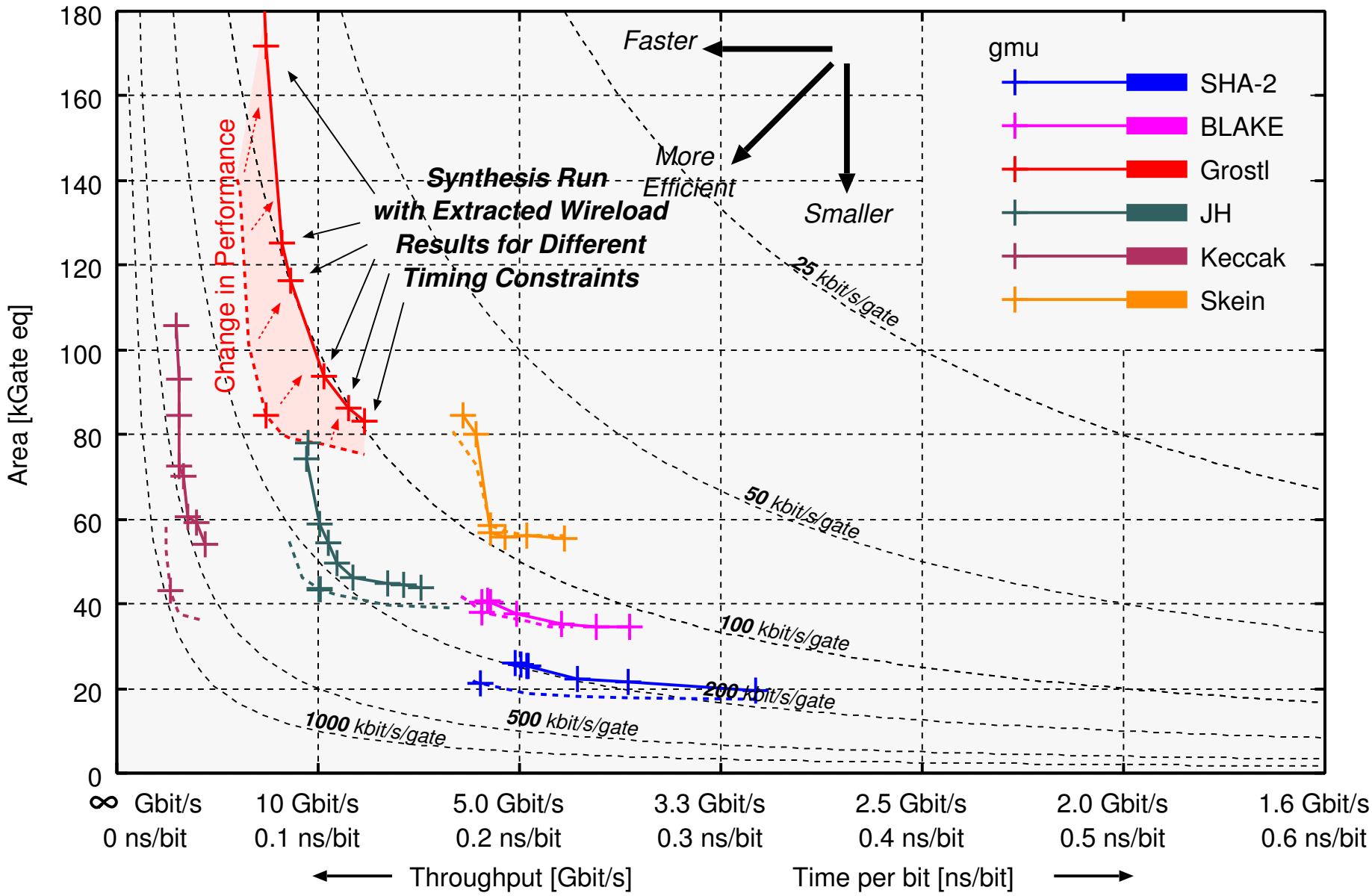
Synthesis Results Depend on the Parasitics



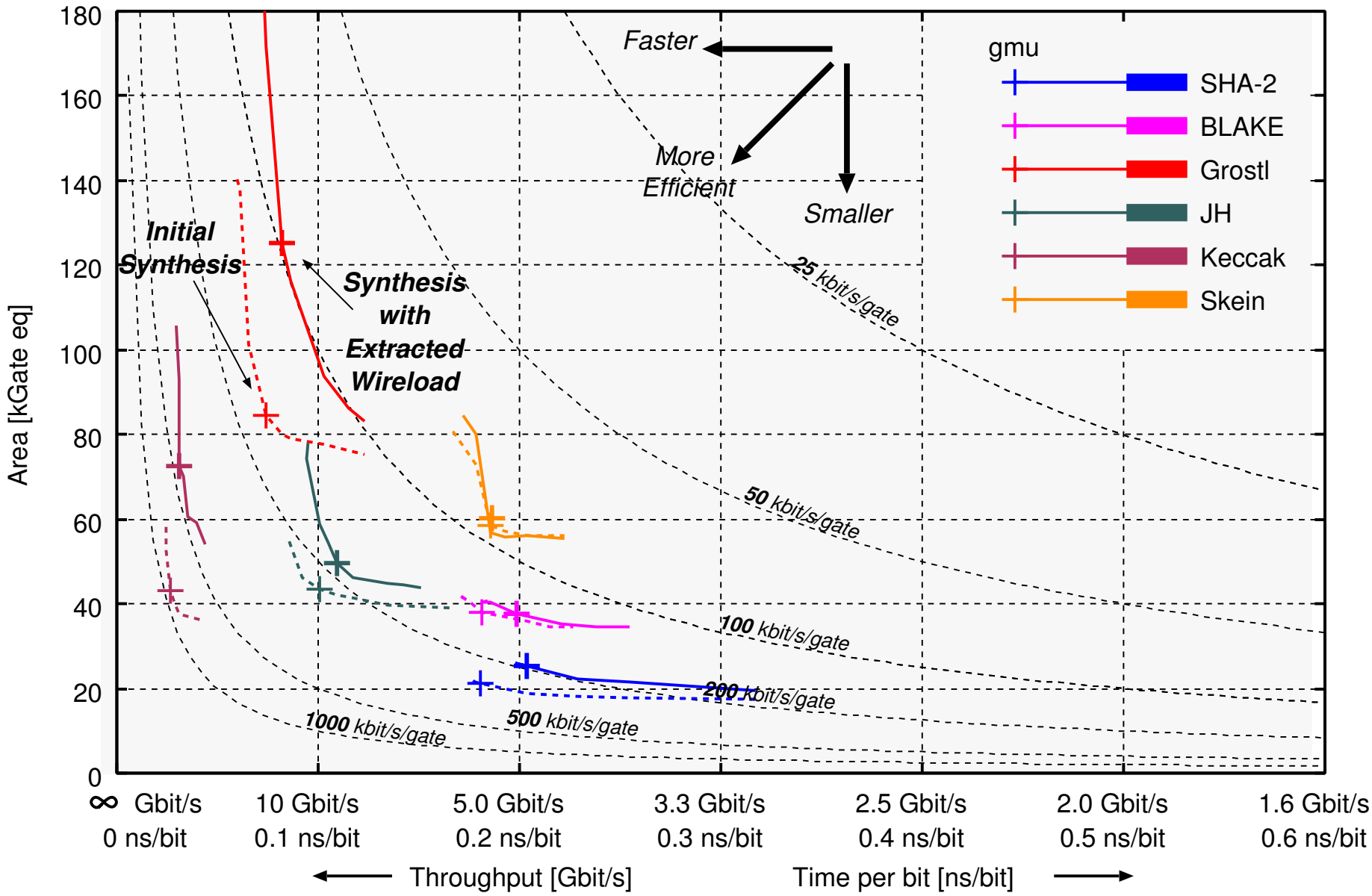
Synthesis Results Depend on the Parasitics



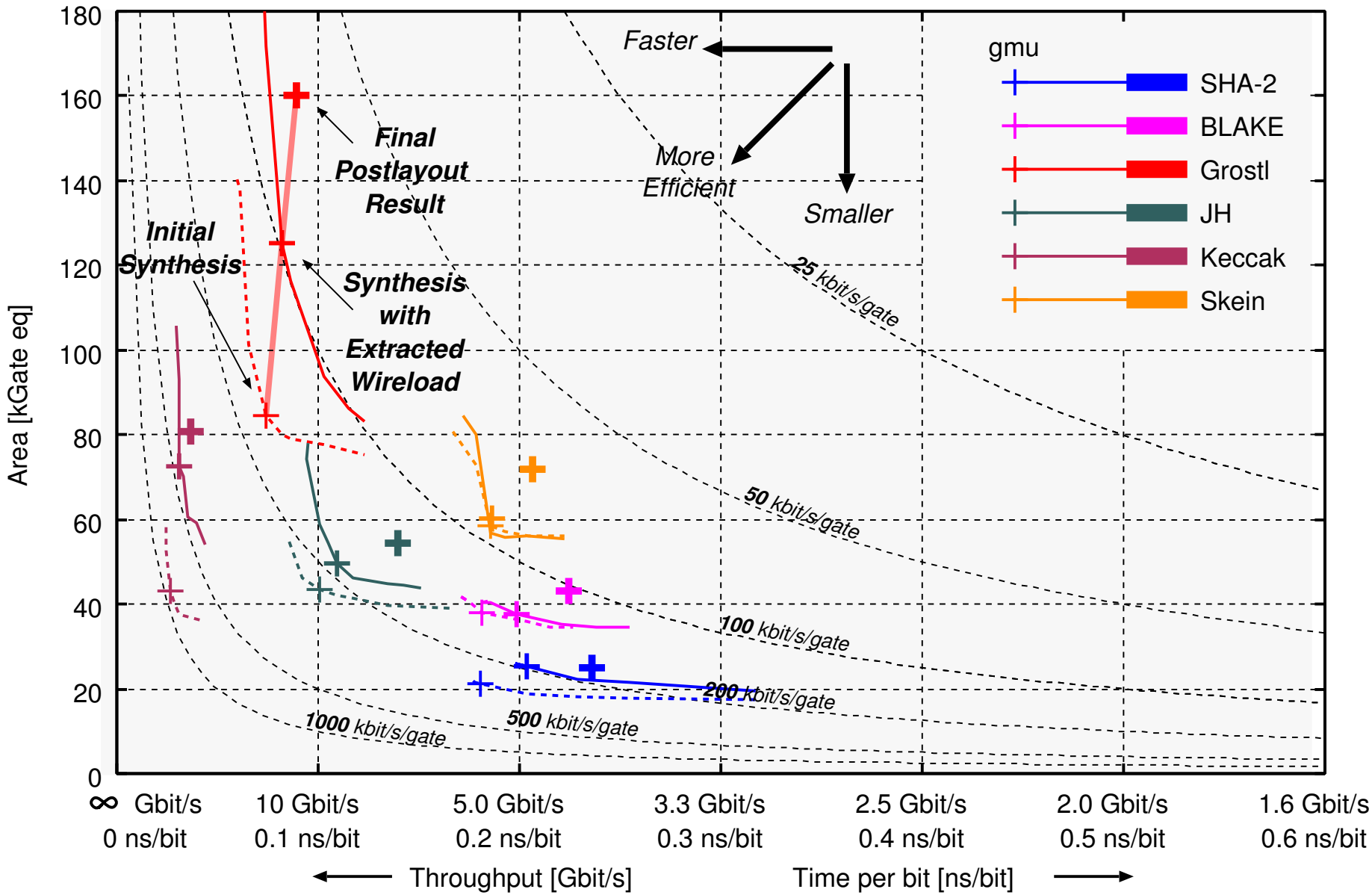
Synthesis Results Depend on the Parasitics



Synthesis Results Depend on the Parasitics



Synthesis Results Depend on the Parasitics



Example: Adding a Crypto Block

Evolved EDGE Receiver System

- Digital front end of a receiver system
- Accepts data from the RF module
- Decodes data at a rate of up to 600 kbit/s
- Want to add a crypto module that optionally generates a digest of the received messages using a hash algorithm

Example: Adding a Crypto Block

Evolved EDGE Receiver System

- Digital front end of a receiver system
- Accepts data from the RF module
- Decodes data at a rate of up to 600 kbit/s
- Want to add a crypto module that optionally generates a digest of the received messages using a hash algorithm

A Suitable SHA-3 Module

- Optimized for small area (10 kGE)
- Runs up to 350 MHz
- Is able to generate digests for messages at a rate of 24 Mbit/s

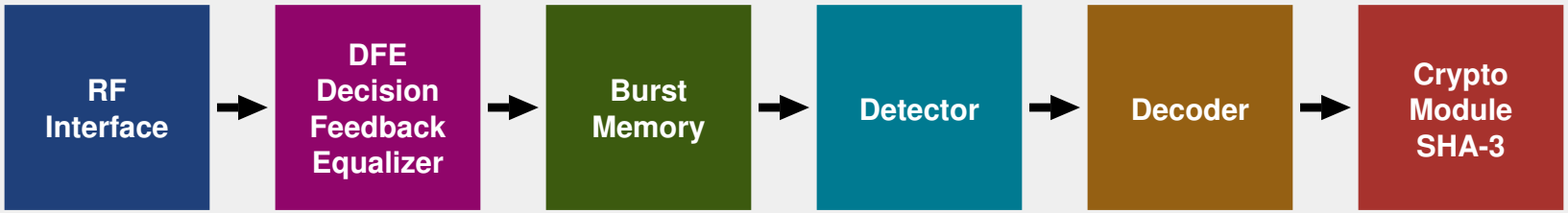
Example: Adding a Crypto Block

Digital Front End for Evolved Edge, Receiver Chain

Evolved EDGE receiver system

Simplified block diagram of the receiver system

Example: Adding a Crypto Block

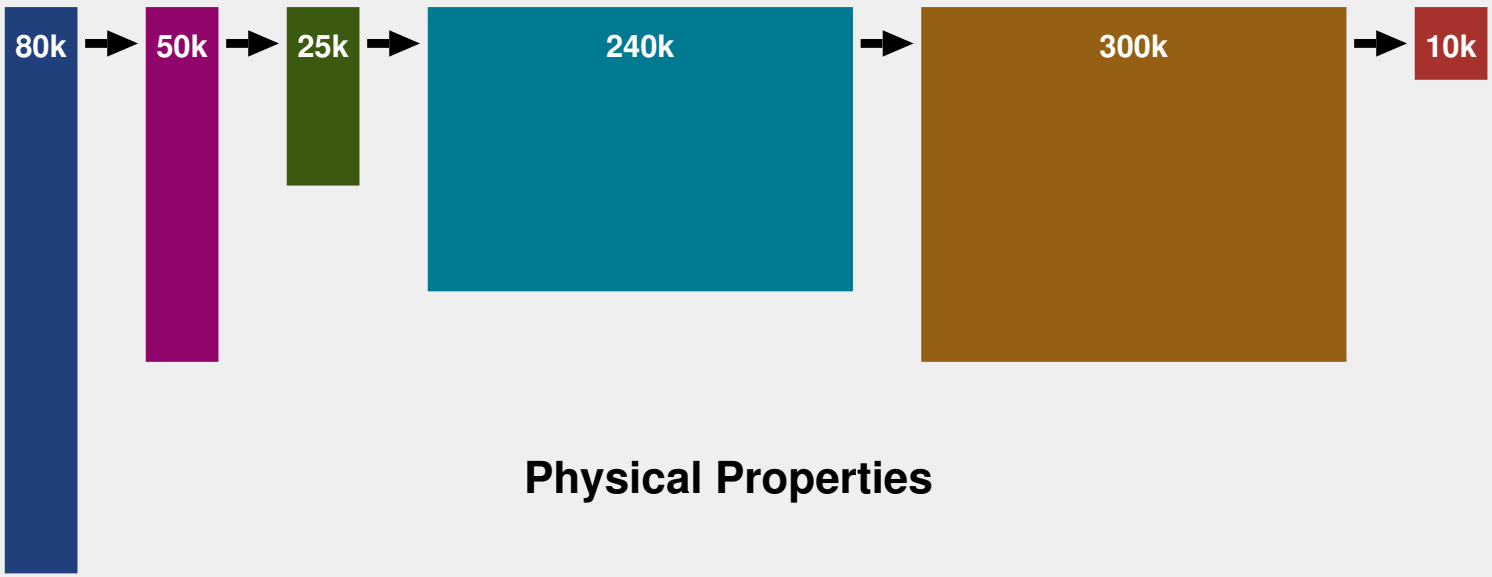


Digital Front End for Evolved Edge, Receiver Chain

Evolved EDGE receiver system

Our goal: to add a SHA-3 module to the end

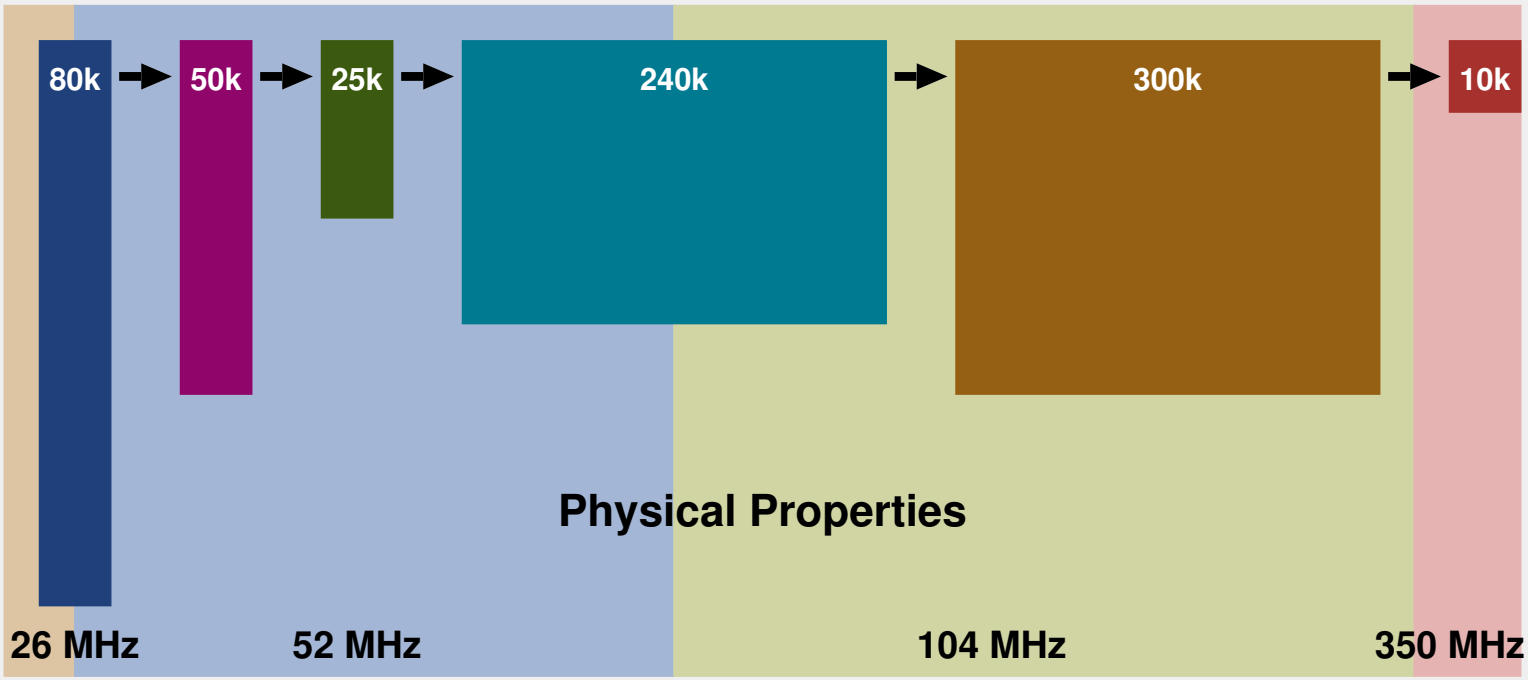
Example: Adding a Crypto Block



Evolved EDGE receiver system

Relative sizes of individual blocks, numbers in gate equivalents

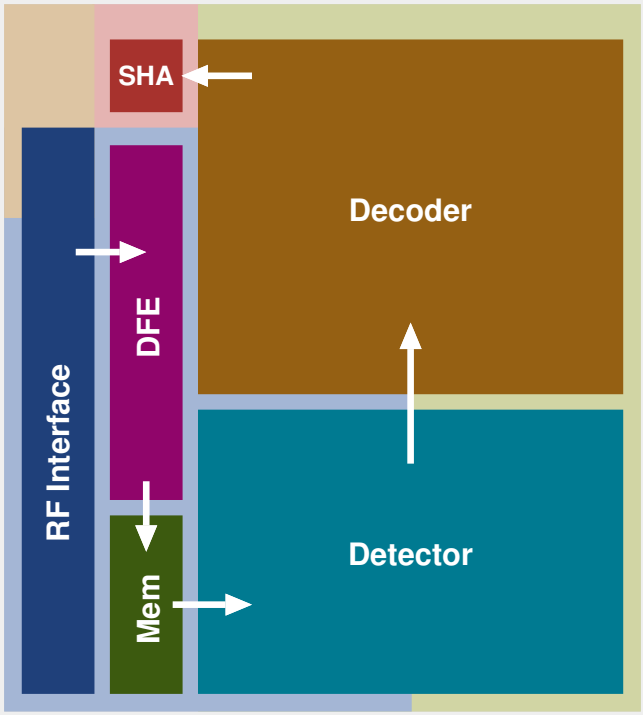
Example: Adding a Crypto Block



Evolved EDGE receiver system

Clock rates at which each block is designed to operate

Example: Adding a Crypto Block



Evolved EDGE receiver system

Possible placement of the modules on an actual die

Academic Performance not Always Relevant

In the previous Evolved EDGE example:

- **SHA-3 block was way too fast**
  - Already as small as possible, no speed/area trade-off
  - Could reduce voltage to save power, too much overhead

Academic Performance not Always Relevant

In the previous Evolved EDGE example:

- **SHA-3 block was way too fast**
  - Already as small as possible, no speed/area trade-off
  - Could reduce voltage to save power, too much overhead
- **Performance uncritical**
  - SHA-3 block very small part of overall system ($<2\dot{\%}$)
  - Gains in area/speed/power will not improve overall performance much

Academic Performance not Always Relevant

In the previous Evolved EDGE example:

- **SHA-3 block was way too fast**
  - Already as small as possible, no speed/area trade-off
  - Could reduce voltage to save power, too much overhead
- **Performance uncritical**
  - SHA-3 block very small part of overall system ($<2\dot{\%}$)
  - Gains in area/speed/power will not improve overall performance much
- **Should not cause too much extra work**
  - Goal: reduce the engineering overhead to add it to system
  - Needs to adapt to the rest of the system as much as possible
  - Use the same voltages and clock frequencies as the rest
  - The interface, dft solutions more of a problem than anything else

Energy and/or Power

How green is my circuit?

Energy and/or Power

How green is my circuit?

- **Power [W]**
  - Could be given as peak, minimum or mean values
  - Important to determine: power supply, rail thicknesses number of power pins and distribution
  - **Power density** [W/cm^2] limiting factor in large chips
  - Does not tell us how much is done in unit time, favors serialization
  - Low-power **does not necessarily mean** your battery will last longer

Energy and/or Power

How green is my circuit?

- **Power [W]**
  - Could be given as peak, minimum or mean values
  - Important to determine: power supply, rail thicknesses number of power pins and distribution
  - **Power density** [W/cm^2] limiting factor in large chips
  - Does not tell us how much is done in unit time, favors serialization
  - Low-power **does not necessarily mean** your battery will last longer
- **Energy [J]**
  - Does not tell us how long the operation will take
  - Usually favors parallelism
  - Low-energy means the battery **will last longer**

Two Separate Energy/Power Related Problems

Increase efficiency of constrained devices

- Circuit power supplied by battery, radiated (RFID), or harvested
- **Energy** determines operation period for battery
- Harvesting, radiation efficiency determines **peak power**
- Usual goal: **more functionality with same energy/power**
- Not so much: increasing battery life, or reducing power

Two Separate Energy/Power Related Problems

Increase efficiency of constrained devices

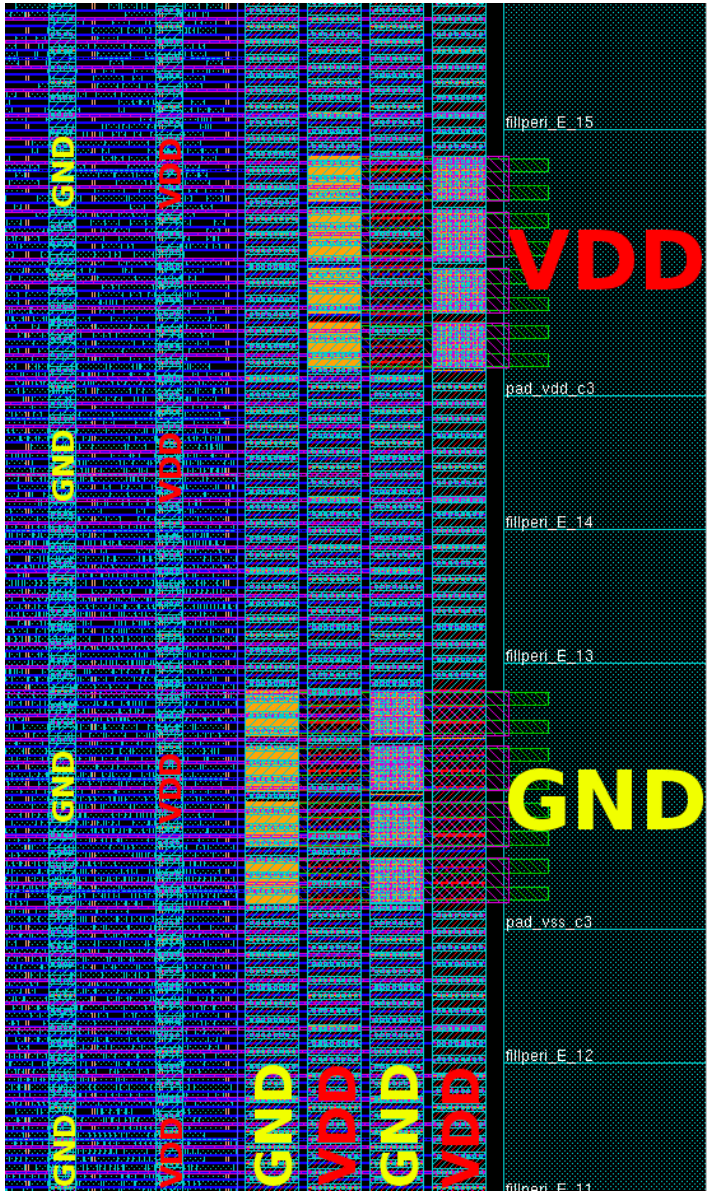
- Circuit power supplied by battery, radiated (RFID), or harvested
- **Energy** determines operation period for battery
- Harvesting, radiation efficiency determines **peak power**
- Usual goal: **more functionality with same energy/power**
- Not so much: increasing battery life, or reducing power

Ensure safe operation of high performance systems

- Moore's Law gives us more transistors per unit area
- Power density increases: $100\,\mathrm{W/cm^2}$ or even more
- Most of this power is dissipated as heat
- If not removed in time, chip will melt
- Solutions: More efficient cooling, **reducing power** consumption

Power Consumption Determines Power Routing

- **Maximum current density**
  1-2 mA per 1 µm of wire
- **Skin effect**
  High frequency current flows on the outside. Extreme wide wires are not helpful
- **Resistance of wires**
  Voltage drop on the power rails, not all cells have the same VDD, work slower
- **Via current density**
  Vias connect different metal layers, fixed geometry, narrower than the metal



Two Components of Power/Energy Consumption

- **Dynamic**
  - Directly caused by activity in circuit (*application dependent*)
  - Depends also on total capacitance switched by the activity (*area*)
  - The rate of the changing activity (*frequency*)
  - And the **square** of the supply voltage
  - Reducing voltage is most promising, however **not much available range**

Two Components of Power/Energy Consumption

- **Dynamic**
  - Directly caused by activity in circuit (*application dependent*)
  - Depends also on total capacitance switched by the activity (*area*)
  - The rate of the changing activity (*frequency*)
  - And the **square** of the supply voltage
  - Reducing voltage is most promising, however **not much available range**
- **Static**
  - Mainly caused by leakage in transistors (*area dependent*)
  - **Increases** as transistors get faster (*smaller technology, lower VT*)
  - Consumed even if circuit is not doing anything
  - Many technology options to counter, body biasing, high VT transistors

Two Components of Power/Energy Consumption

- **Dynamic**
  - Directly caused by activity in circuit (*application dependent*)
  - Depends also on total capacitance switched by the activity (*area*)
  - The rate of the changing activity (*frequency*)
  - And the **square** of the supply voltage
  - Reducing voltage is most promising, however **not much available range**
- **Static**
  - Mainly caused by leakage in transistors (*area dependent*)
  - **Increases** as transistors get faster (*smaller technology, lower VT*)
  - Consumed even if circuit is not doing anything
  - Many technology options to counter, body biasing, high VT transistors

Modern low power/energy designs try to

Balance dynamic and static consumption at around 50 %

How to Determine Power

$$P_{total} = P_{static} + P_{dynamic} \sim \alpha \cdot C \cdot f \cdot {V_{DD}}^2$$

- P_{static} **is relatively simple**
  As soon as you have the netlist, you can estimate it
- f **and** ${V_{DD}}^2$ **are known**
  Both are design parameters, that would be known after synthesis

How to Determine Power

$$P_{total} = P_{static} + P_{dynamic} \sim \alpha \cdot C \cdot f \cdot {V_{DD}}^2$$

- P_{static} **is relatively simple**
  As soon as you have the netlist, you can estimate it
- f **and** ${V_{DD}}^2$ **are known**
  Both are design parameters, that would be known after synthesis
- **For** C **you need the placed and routed design**
  This is problematic, only available at the end of design flow. Circuit needs to be simulated with back-annotated netlist, **time consuming**

How to Determine Power

$$P_{total} = P_{static} + P_{dynamic} \sim \alpha \cdot C \cdot f \cdot {V_{DD}}^2$$

- P_{static} **is relatively simple**
  As soon as you have the netlist, you can estimate it
- f **and** ${V_{DD}}^2$ **are known**
  Both are design parameters, that would be known after synthesis
- **For** C **you need the placed and routed design**
  This is problematic, only available at the end of design flow. Circuit needs to be simulated with back-annotated netlist, **time consuming**
- α **activity depends on the vectors**
  Power directly depends on the activity.
  Finding **correct vectors** not trivial

It is Easy to Make a Mistake Estimating Power

Make sure that you:

- **Specify voltage and clock frequency**
  Any power figure without these is useless
- **Explain how circuit activity was obtained**
  - Assumed a **global activity** value
  - Derived from **input activity**
  - Actual **functional simulation vectors**
- **Disclose how the power was measured**
  - Estimated during synthesis
  - Back-annotated delays with post-layout netlist
  - Actual measurement on manufactured chip
- **For power estimations (not actual measurements)**
  State operating conditions as well: best, typical, worst case
- **Actual measurements not easy**
  Difficult to measure current for fast switching circuits

What About Compound Performance Measures?

Quite common technique to compare different architectures

- **Throughput per area**
  Used commonly in cryptographic hardware papers, shows trade-off between area and speed.

- **Energy delay product**
  Popular with computer architectures, shows the trade-off between energy and speed. Few variants of $\mathrm{E}^n \cdot \mathrm{D}^m$, with different n and m exist.

What About Compound Performance Measures?

Quite common technique to compare different architectures

- **Throughput per area**
  Used commonly in cryptographic hardware papers, shows trade-off between area and speed.

- **Energy delay product**
  Popular with computer architectures, shows the trade-off between energy and speed. Few variants of $\mathrm{E}^n \cdot \mathrm{D}^m$, with different n and m exist.

Meaningless, unless the circuit has been designed for this measure

Example: Throughput Per Area in 65 nm

Keccak

- **TpA: 395 kbit/s·GE**
  Keccak is almost twice as fast
- **Area: ?**
  Is it the same area...
- **Speed: ?**
  .. but double the speed ?

SHA-2

- **TpA: 215 kbit/s·GE**
  SHA-2 is slower/larger version
- **Area: ?**
  Is it twice as large as Keccak...
- **Speed: ?**
  ... at the same speed ?

Compound Performance Metrics can be Misleading

- TpA shows neither the speed nor the area of each design
- It implies that Throughput can be traded-off against Area without limitation. This is not true.

Example: Throughput Per Area in 65 nm

Keccak

- **TpA: 395 kbit/s·GE**
  Keccak is almost twice as fast
- **Area: 80 kGE**
  More than 3 times the area
- **Speed: 32 Gbit/s**
  and 6 times the speed

SHA-2

- **TpA: 215 kbit/s·GE**
  SHA-2 is slower/larger version
- **Area: 25 kGE**
  In reality SHA-2 is small
- **Speed: 5.4 Gbit/s**
  but much slower

Compound Performance Metrics can be Misleading

- TpA shows neither the speed nor the area of each design
- It implies that Throughput can be traded-off against Area without limitation. This is not true.
- Can not get a 1 Gbit/s Keccak using only 2.5 kGEs

Comparing FPGAs and ASICs

- **Started as programmable logic**
  Alternative for glue logic, replacement for 74xx families

Comparing FPGAs and ASICs

- **Started as programmable logic**
  Alternative for glue logic, replacement for 74xx families
- **Became something more**
  Added block RAMs, programmable I/Os, PLLs, DSP slices

Comparing FPGAs and ASICs

- **Started as programmable logic**
  Alternative for glue logic, replacement for 74xx families
- **Became something more**
  Added block RAMs, programmable I/Os, PLLs, DSP slices
- **Newest generation is even more powerful**
  I.e. the Xilinx Zynq contains a dual core ARM A9.

Resource allocation problem

- FPGAs have a collection of resources
- If you can make use of them good, if not you don't save anything
- In some cases, you can use a smaller (cheaper FPGA), cost benefit

FPGAs have a Fast Design Flow

Typically a single tool for the entire design flow

- **Good post-layout results**
  All resources on FPGA characterized, good speed/power estimations
- **Fast implementation**
  The complete design can be measured/tested immediately

Estimating Area Usage on FPGAs Problematic

FPGAs contain a set of resources

- **Not all resources used by the same amount**
  Design uses x % of LUTs, and y % of BRAMs and z % of DSPs
- **FPGA families differ in their resources**
  The type and particular mix of resources in an FPGA varies
- **In practice most resources can not be used 100 %**
  In most cases you will not be able to put 5 designs that use 20 % of the LUTs in to the same FPGA
- **Resources are there no matter what**
  Pipelining FFs are in every LUT (ALM). If you do not use them they will be by-passed. A circuit with 20 pipeline stages is not larger than one with only one stage.

Last Words

What Did We Discuss?

- **Cost structure of IC design**
- **Area and Speed of IC circuits**
  What are the common pitfalls
- **Academic vs Industrial use of EDA tools**
  Different goals in both approaches, but EDA tools designed for industrial goals
- **Energy/Power**
  Why is it the more difficult parameter to determine
- **If you review a paper**
  Now you know which ones you can reject.