Lessons Learned from Designing a 65 nm ASIC for Third Round SHA-3 Candidates

Frank K. Gürkaynak, Kris Gaj, Beat Muheim,
Ekawat Homsirikamol, Christoph Keller, Marcin Rogawski,
Hubert Kaeslin, Jens-Peter Kaps

ETH Zurich - George Mason University

22-23 March 2012

Motivation

Present

comparative ASIC performance results on all SHA-3 third round candidates

Motivation

Present

comparative ASIC performance results on all SHA-3 third round candidates

In this work

- No claims about the cryptographic security
- Authors' recommendations for SHA-2-256 equivalent security have been followed

Two Groups, Two Different Approaches

George Mason University

- Academic approach
- Optimized for maximum: **Throughput per Area**
- Taken VHDL codes from extensive architecture evaluations for FPGAs

Two Groups, Two Different Approaches

George Mason University

- Academic approach
- Optimized for maximum: **Throughput per Area**
- Taken VHDL codes from extensive architecture evaluations for FPGAs

ETH Zurich

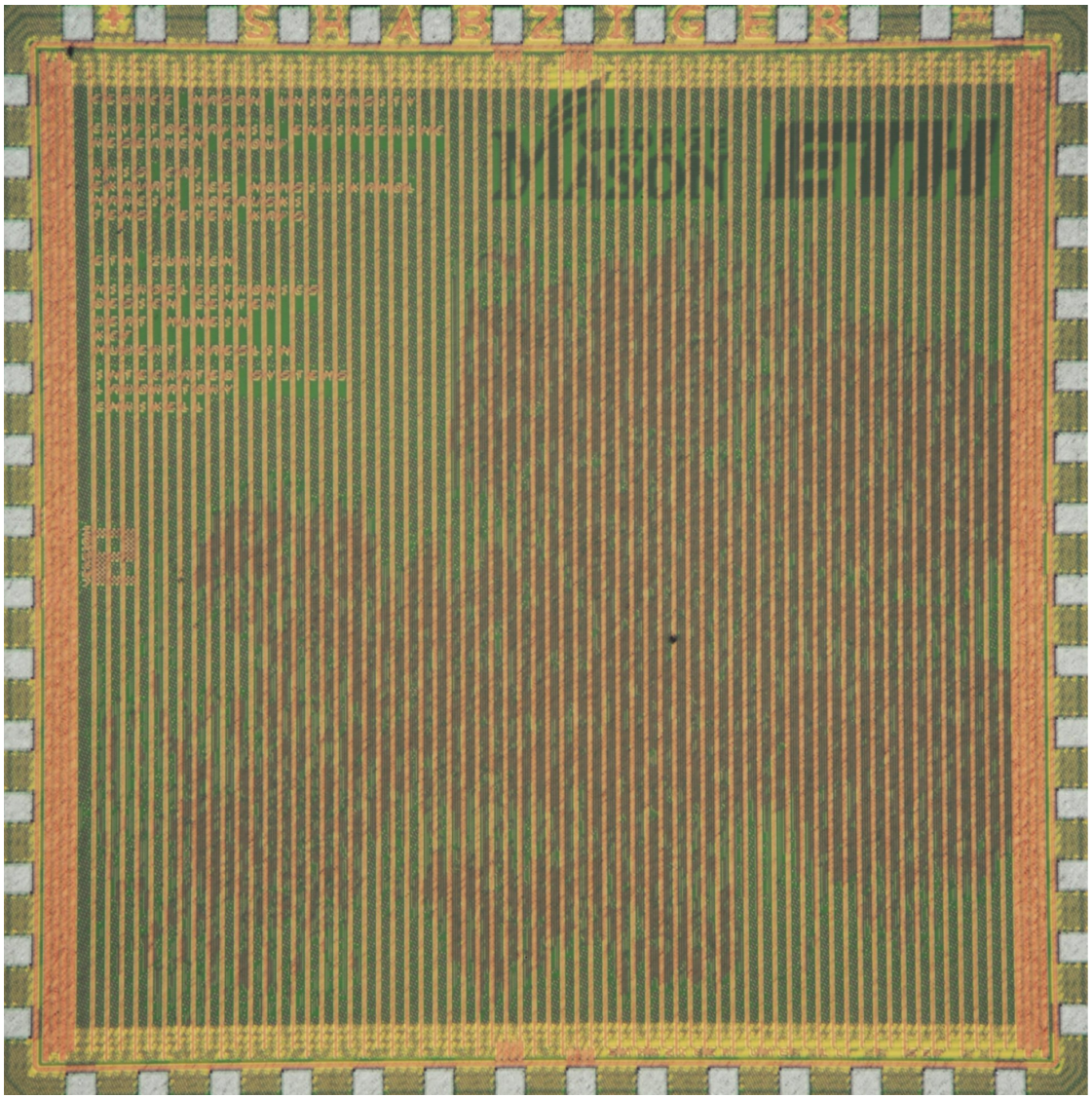
- Quasi industrial approach
- Specific throughput target: **2.488 Gbit/s**
- Selected smallest design for the throughput
- Deliberately tried to increase architectural diversity

Background

Timeline

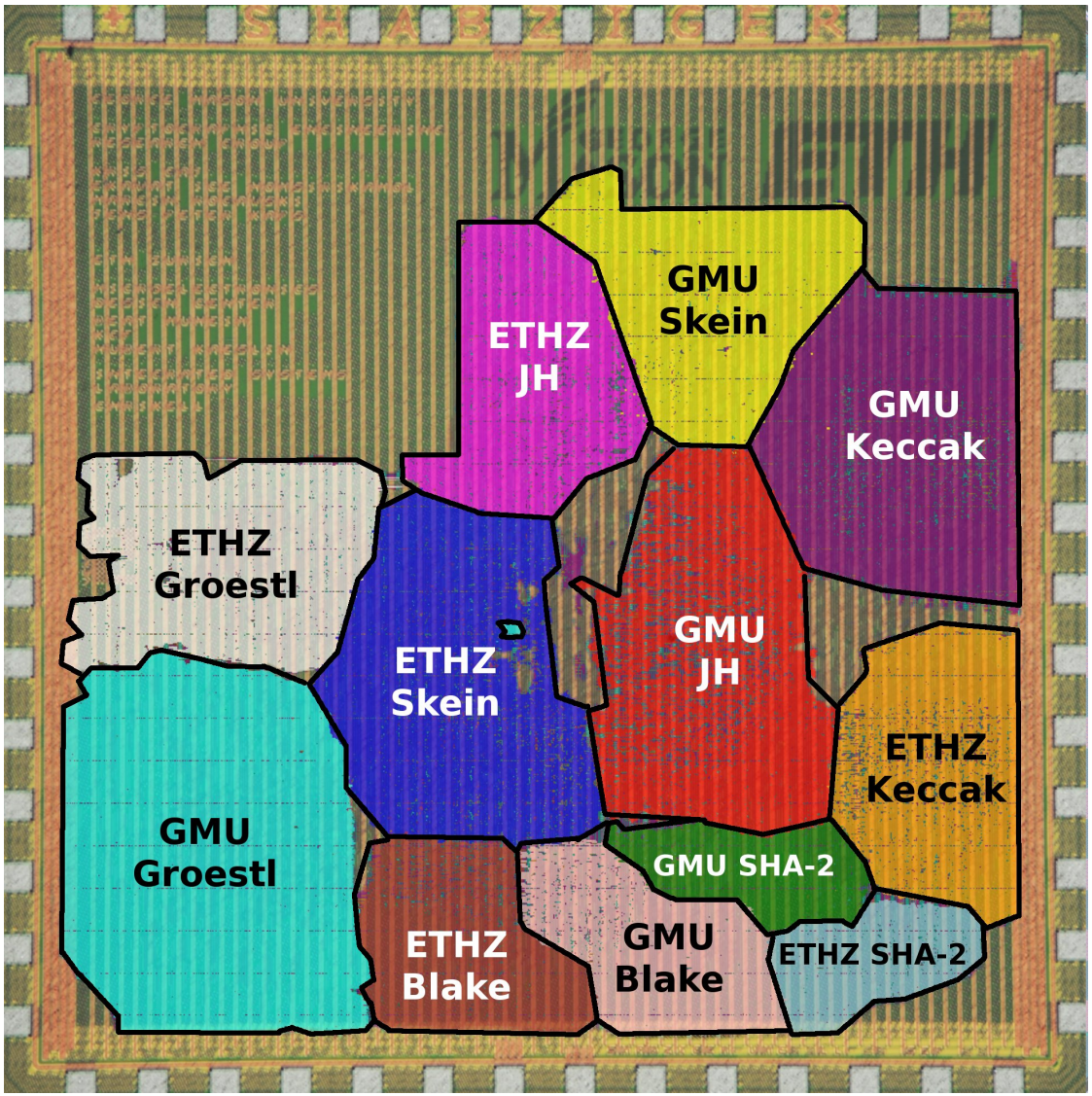
earlier	GMU releases ATHENa, a database for FPGA results ETH publishes study on 2nd round candidates
May 2011	Quo Vadis 2011 Wokshop in Warsaw Start of collaboration
Jun 2011	Start of project
Aug 2011	Common interface, all cores (ETH Zurich-GMU) compatible
Oct 2011	Tape-out
Dec 2011	Production problem with I/O transistors
Feb 2012	Measured 5 ASICs from first batch

SHABZIGER: Our ASIC with all SHA-3 Candidates



- **Techology**
  UMCLL65nm
- **Supply**
  1.2V VDD
- **Metallization**
  8-Metal
- **Package**
  56pin QFN56
- **Total Size**
  1.825mm x 1.825mm
- **Area Unit**
  1 GE=1.44µm^2

SHABZIGER: Our ASIC with all SHA-3 Candidates



- **Techology**
  UMCLL65nm
- **Supply**
  1.2V VDD
- **Metallization**
  8-Metal
- **Package**
  56pin QFN56
- **Total Size**
  1.825mm x 1.825mm
- **Area Unit**
  1 GE=1.44μm^2

Main Problem

EDA tools are designed for industry requirements

- Constraints for **worst** case conditions.
- Tools **not** designed for finding peak (faster/smaller) performance.

Main Problem

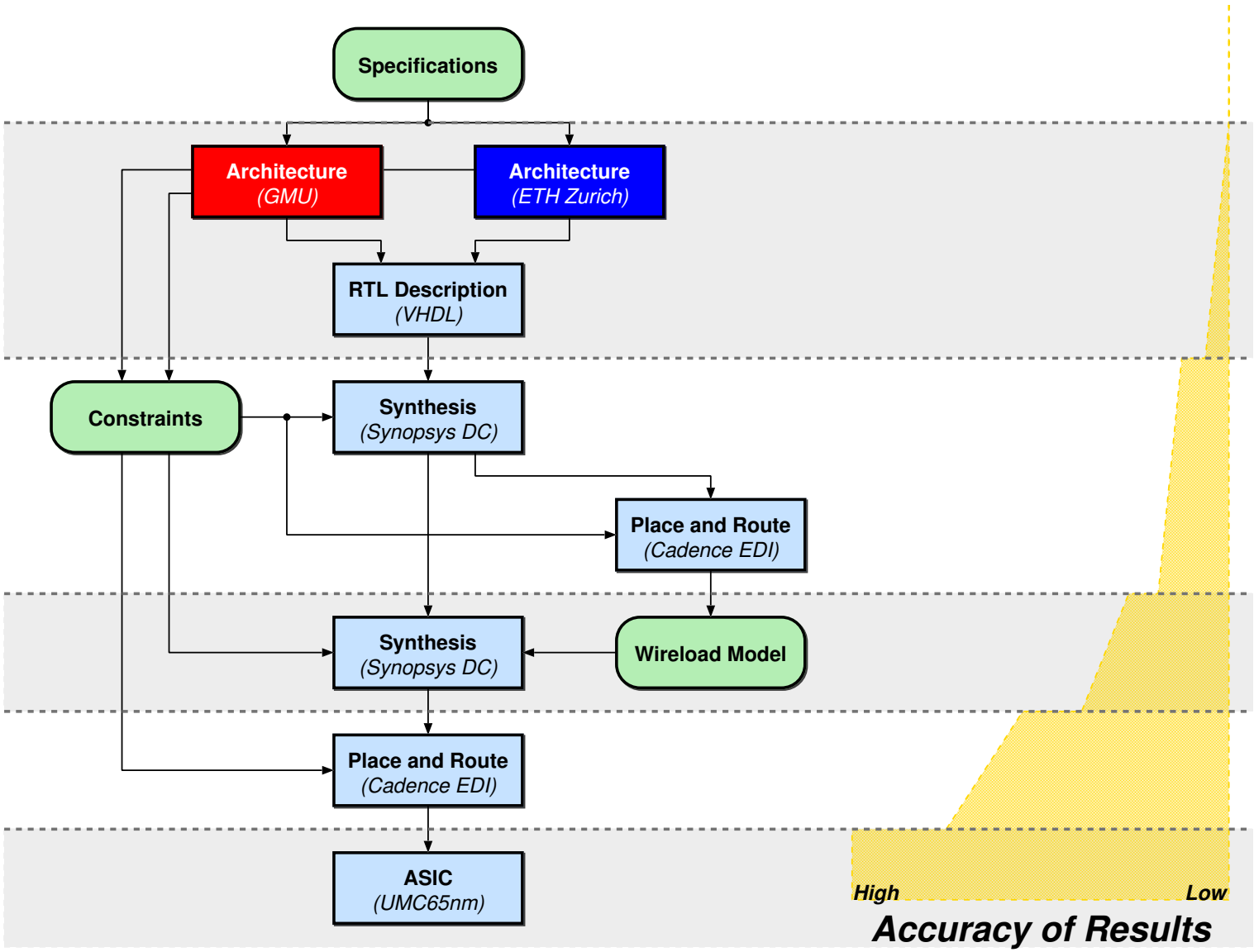
EDA tools are designed for industry requirements

- Constraints for **worst** case conditions.
- Tools **not** designed for finding peak (faster/smaller) performance.

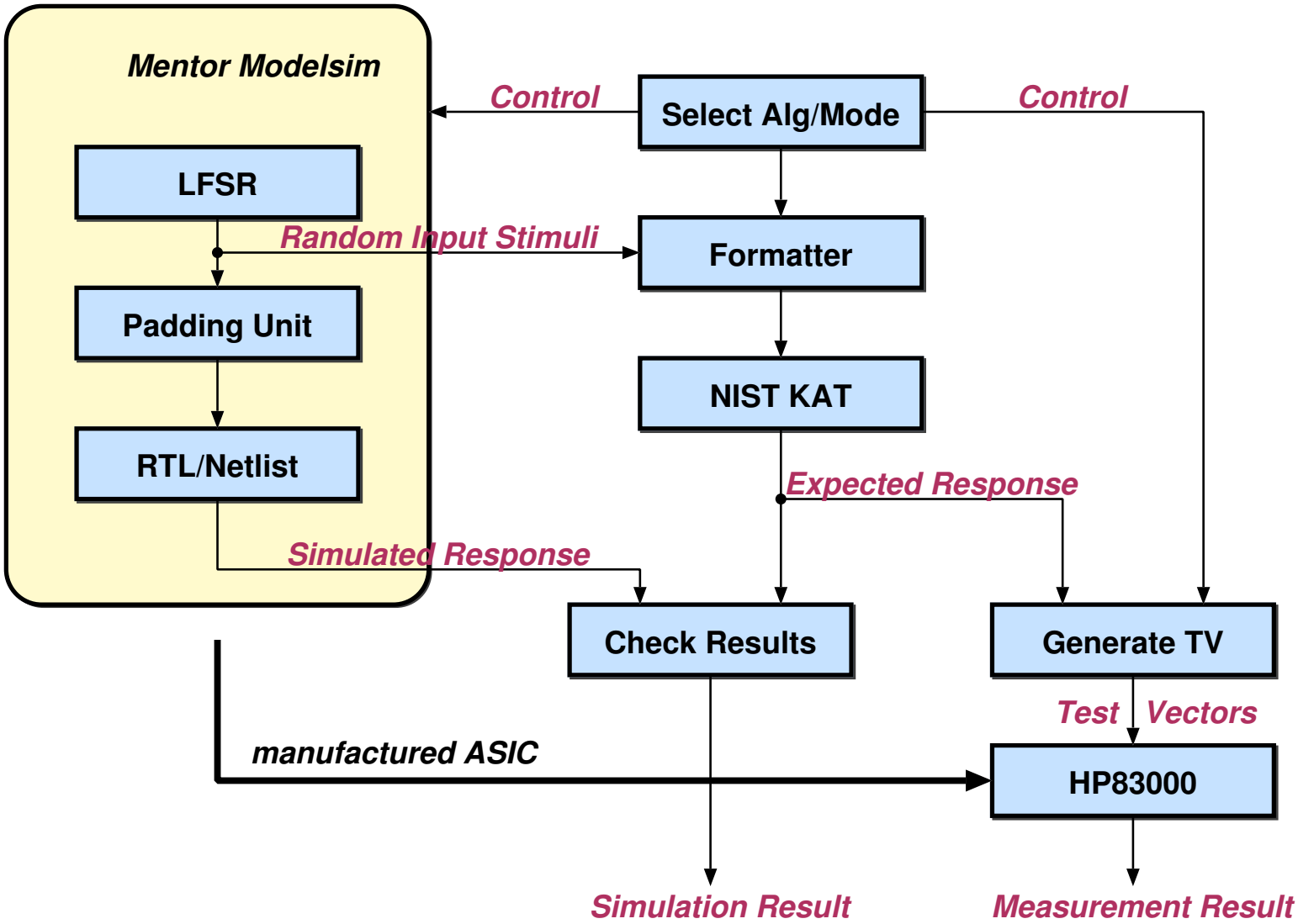
In general, Academia is interested in limits

- Not easy to get **fair** numbers from industrial tools.
- Constraints are **mis-used** for exploration.

The Design Flow



The Verification Flow



Reporting Performance: Area

How much silicon area is used by the circuit

- Area is reported in Gate Equivalents (GE).
- For the UMC65 technology and the standard cell library used

$$1\,\text{GE} = 1.44\,\mu\text{m}^2$$

- Includes overhead for clock trees, scan chains, reset circuitry.

Reporting Performance: Area

How much silicon area is used by the circuit

- Area is reported in Gate Equivalents (GE).
- For the UMC65 technology and the standard cell library used

$$1\,\text{GE} = 1.44\,\mu\text{m}^2$$

- Includes overhead for clock trees, scan chains, reset circuitry.

Area in Gate Equivalents is not very accurate

Additional overhead for :

- Power
- Routability
- Signal integrity

These depend on circuit and operating conditions.

Reporting Performance: Time, Speed, Throughput

Finding the correct unit

- **Clock period** [ns]
  Main constraint for speed in a digital circuit.

Reporting Performance: Time, Speed, Throughput

Finding the correct unit

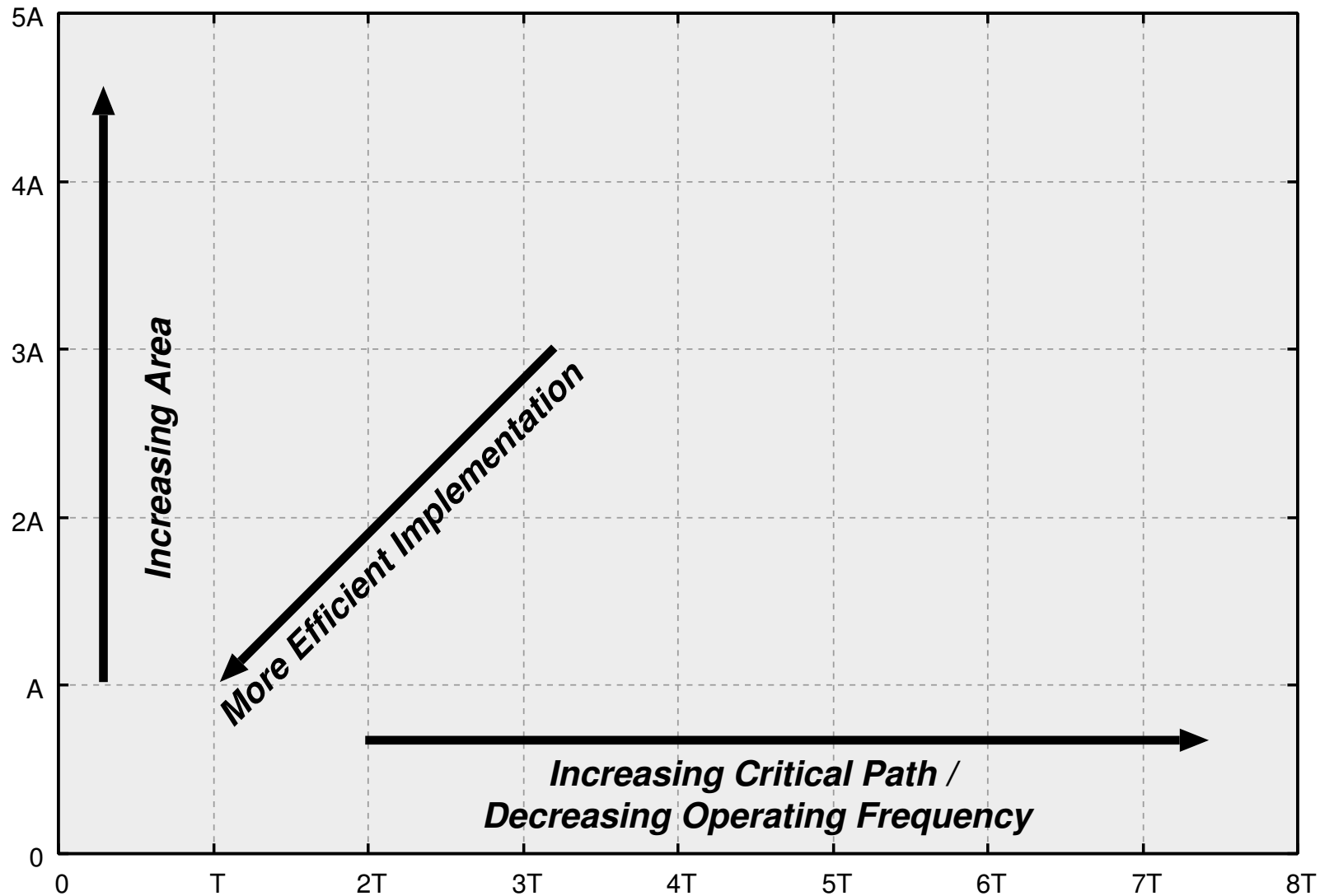
- **Clock period** [ns]
  Main constraint for speed in a digital circuit.
- **Throughput** [Gbit/s]
  Useful when comparing **different architectures**
  In this work: long message hashing performance.

Reporting Performance: Time, Speed, Throughput

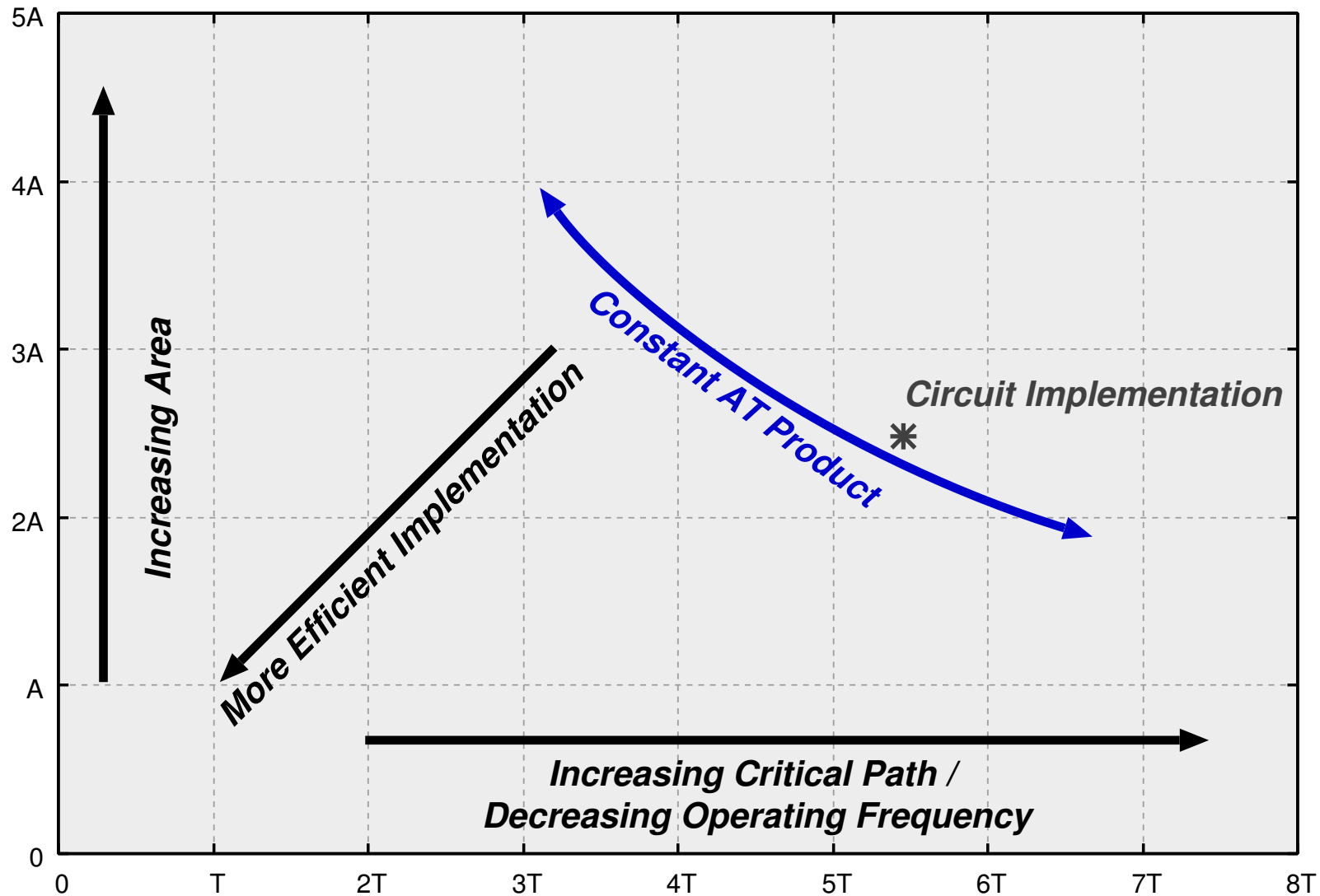
Finding the correct unit

- **Clock period** [ns]
  Main constraint for speed in a digital circuit.
- **Throughput** [Gbit/s]
  Useful when comparing **different architectures**
  In this work: long message hashing performance.
- **Time per data item** [ns/bit]
  More practical for AT (Area-Time) plots, one axis is time.
  Similar to [cycles/byte] used for software performance

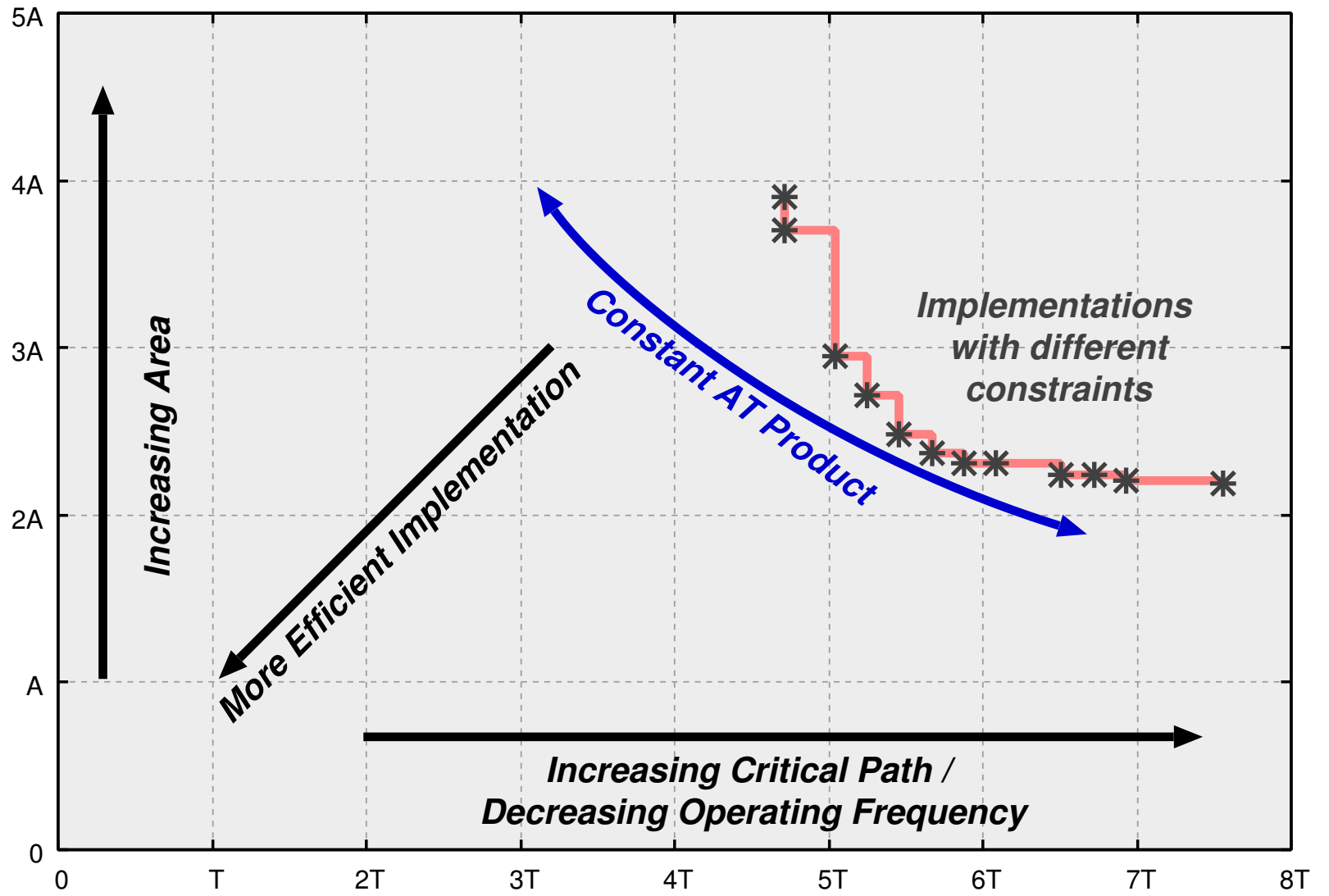
The AT plot


5A
4A
3A
2A
A
0
0
T
2T
3T
4T
5T
6T
7T
8T
Increasing Area
More Efficient Implementation
Increasing Critical Path /
Decreasing Operating Frequency

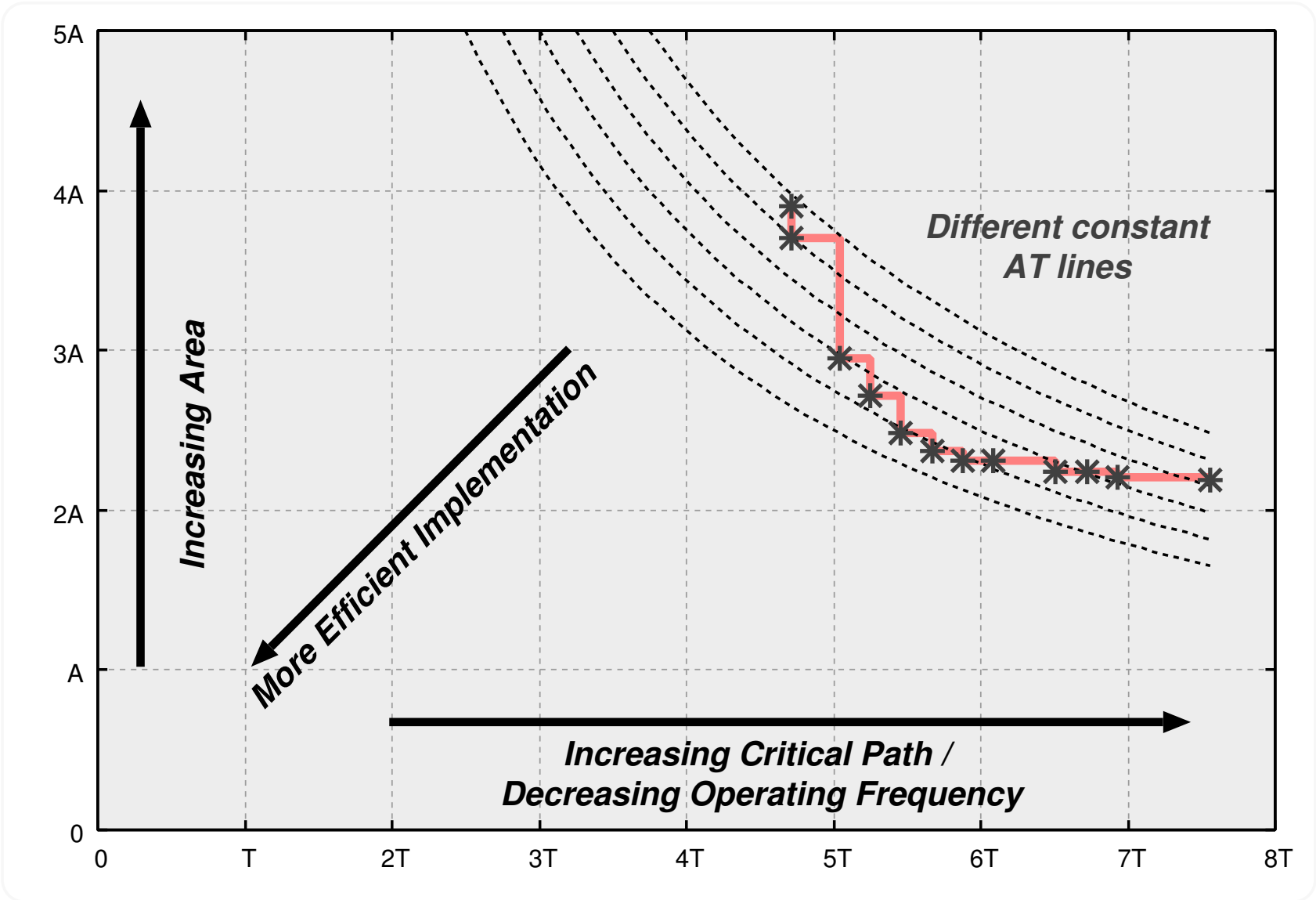
The AT plot


5A
4A
3A
2A
A
0
0
T
2T
3T
4T
5T
6T
7T
8T
Increasing Area
More Efficient Implementation
Constant AT Product
Circuit Implementation
Increasing Critical Path /
Decreasing Operating Frequency

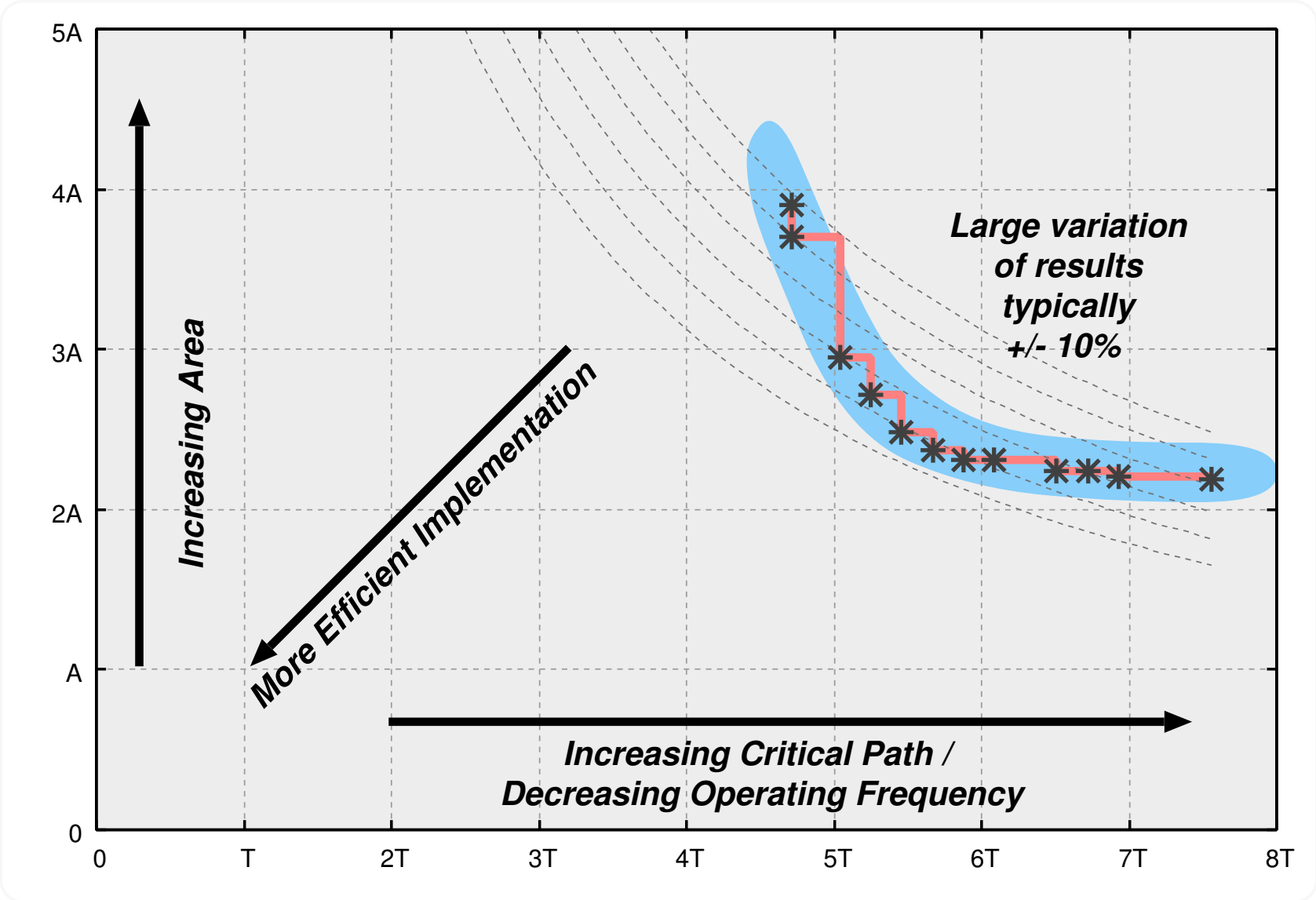
The AT plot


5A
4A
3A
2A
A
0
0
T
2T
3T
4T
5T
6T
7T
8T
Increasing Area
More Efficient Implementation
Constant AT Product
Implementations with different constraints
Increasing Critical Path / Decreasing Operating Frequency

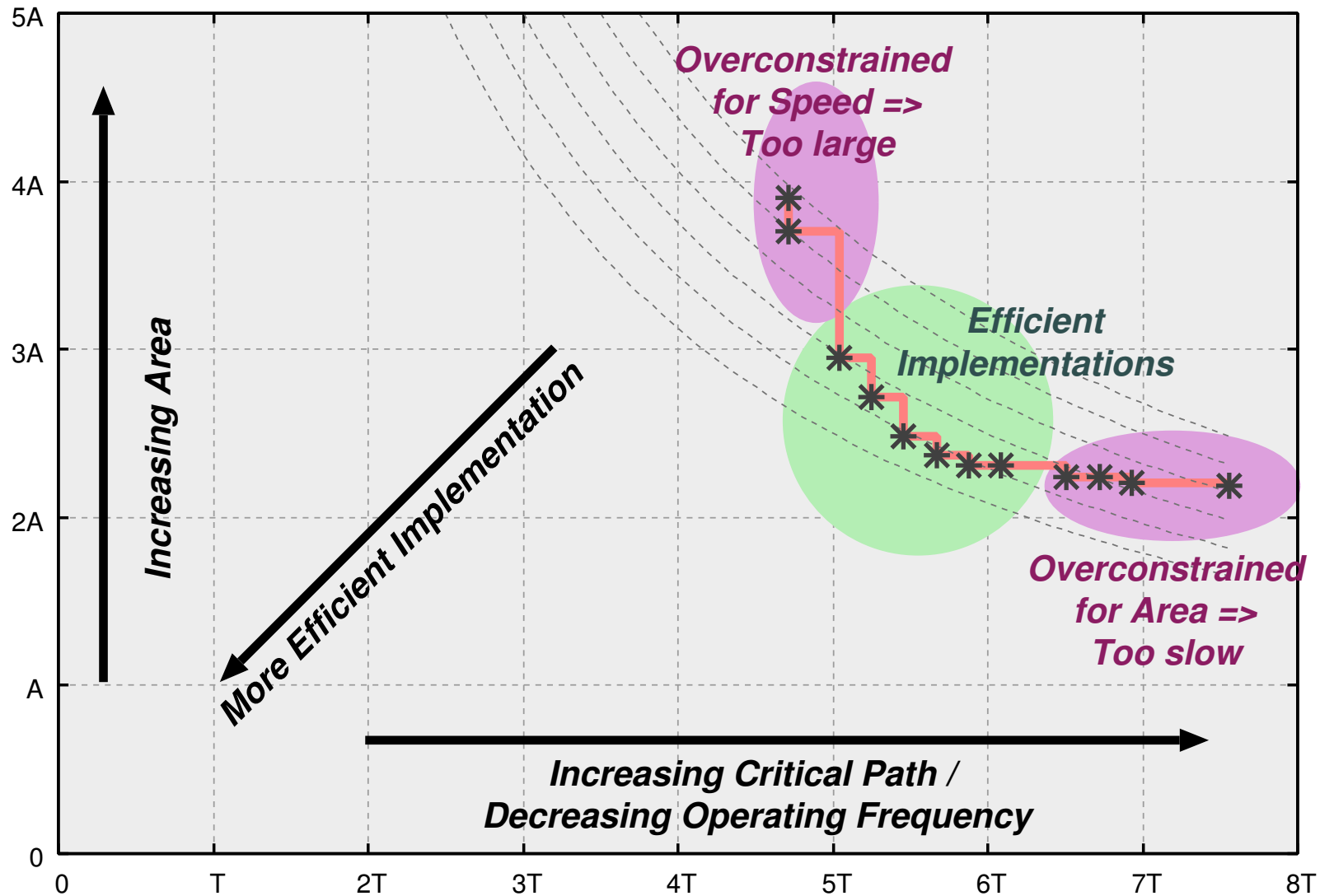
The AT plot


5A
4A
3A
2A
A
0
0
T
2T
3T
4T
5T
6T
7T
8T
Increasing Area
More Efficient Implementation
Different constant
AT lines
Increasing Critical Path /
Decreasing Operating Frequency

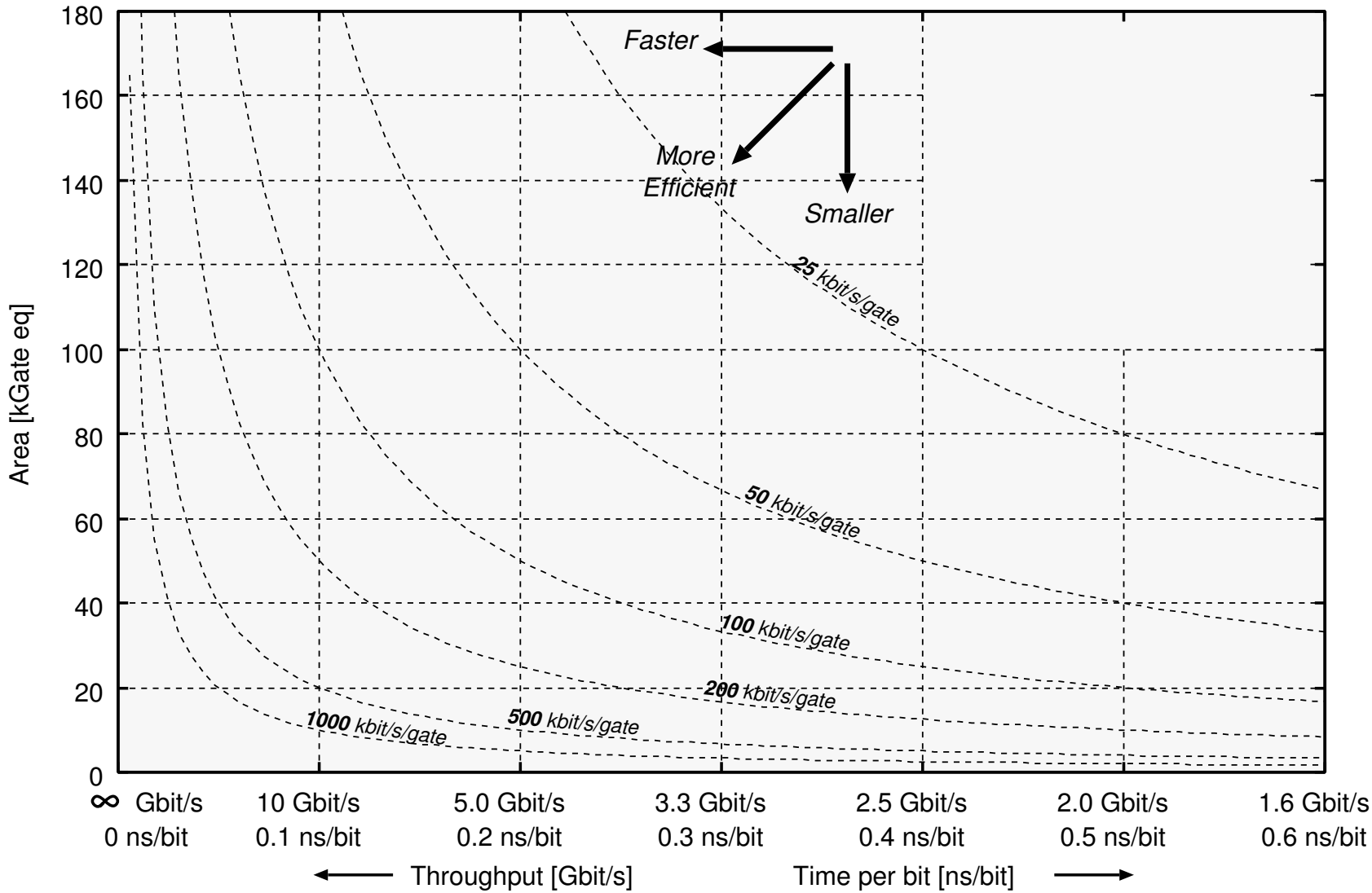
The AT plot


5A
4A
3A
2A
A
0
0
T
2T
3T
4T
5T
6T
7T
8T
Increasing Area
More Efficient Implementation
Large variation of results typically +/- 10%
Increasing Critical Path / Decreasing Operating Frequency

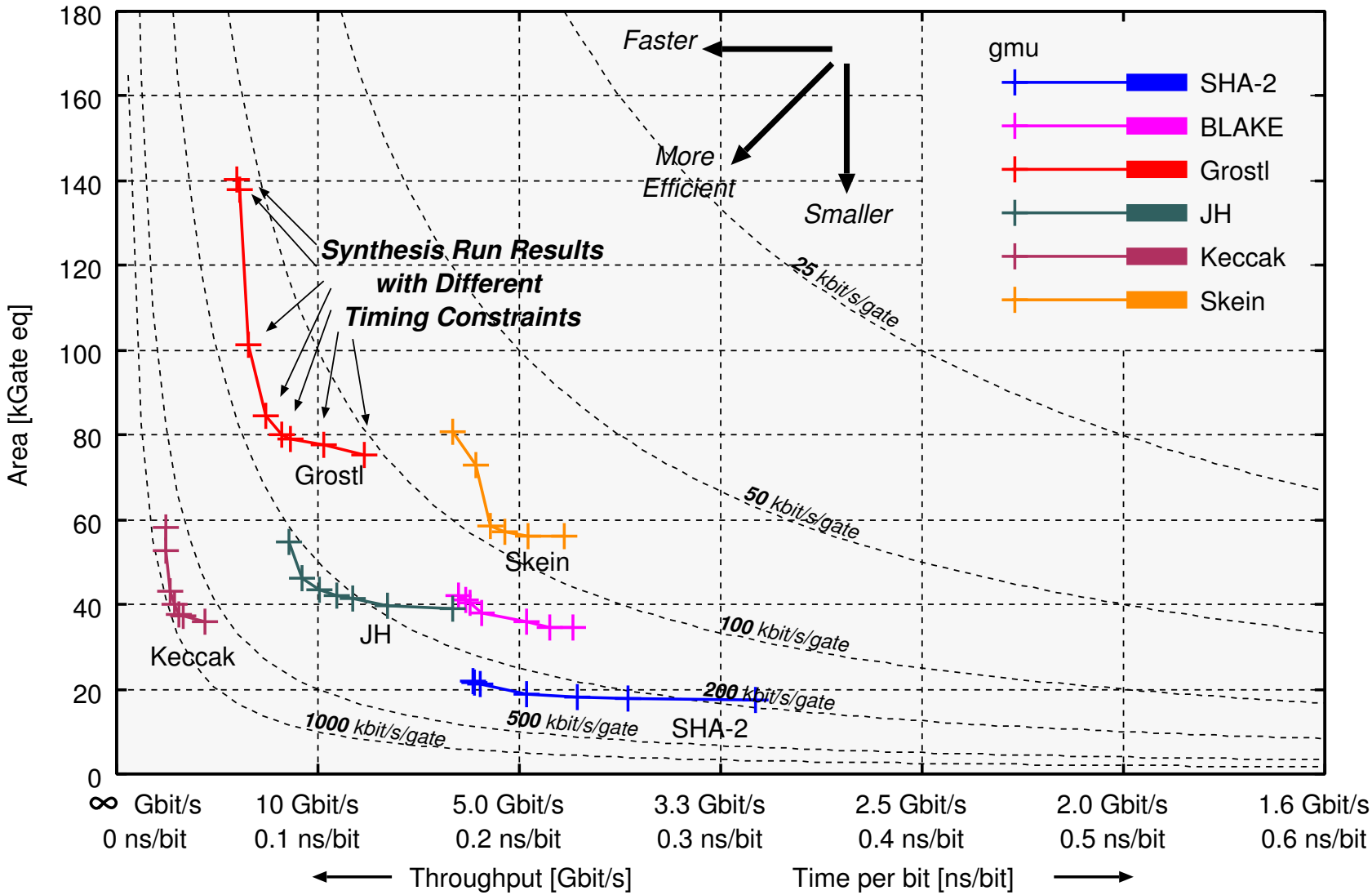
The AT plot


5A
4A
3A
2A
A
0
0
T
2T
3T
4T
5T
6T
7T
8T
Overconstrained
for Speed =>
Too large
Efficient
Implementations
Overconstrained
for Area =>
Too slow
Increasing Area
More Efficient Implementation
Increasing Critical Path /
Decreasing Operating Frequency

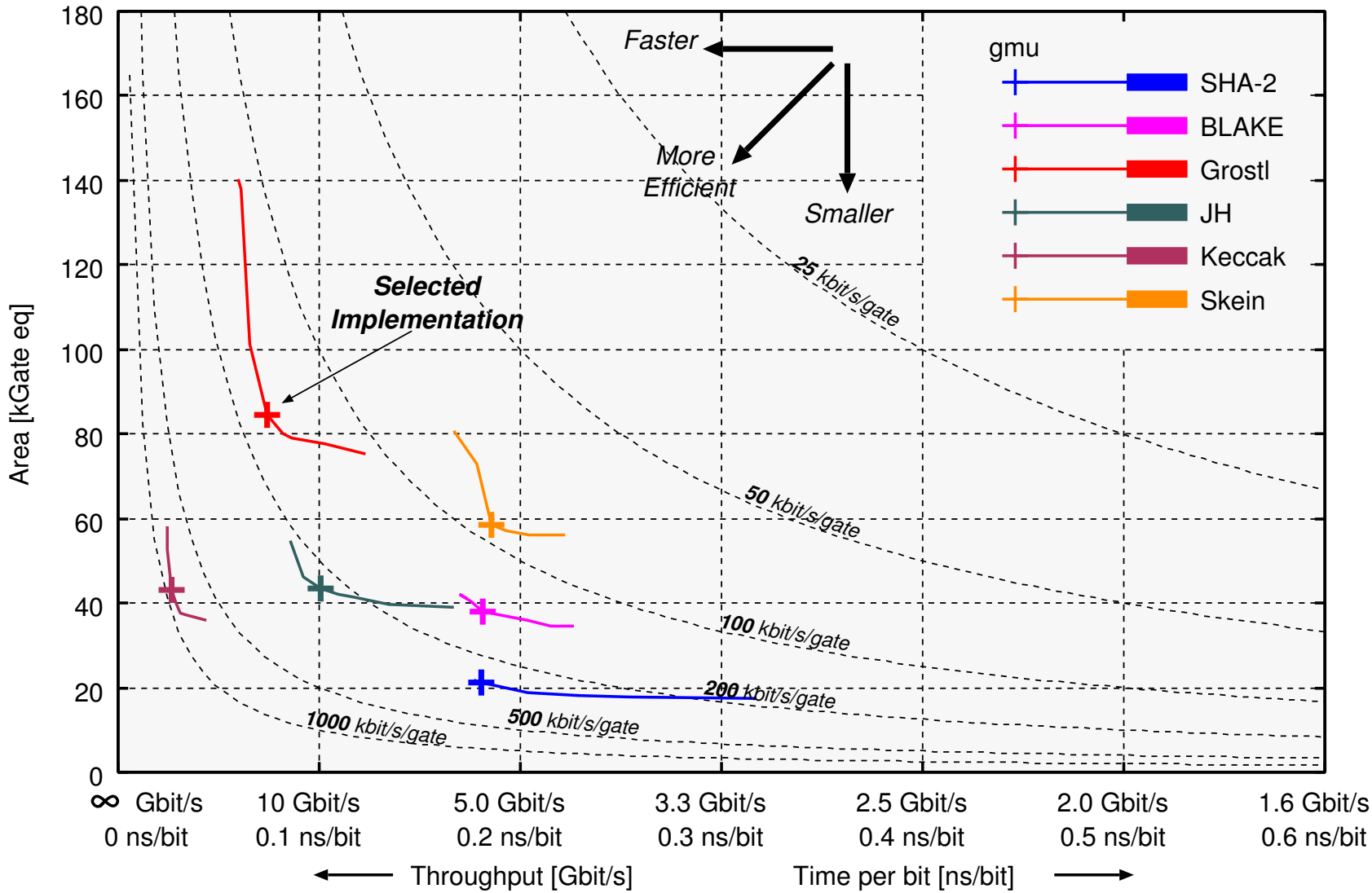
Synthesis Results



Synthesis Results



Synthesis Results



The Story of Wireload Models

Wireload models reflect the routing overhead of the circuit

- **Parasitic effects** are major contributors to overall delay.

The Story of Wireload Models

Wireload models reflect the routing overhead of the circuit

- **Parasitic effects** are major contributors to overall delay.
- During synthesis, wireload models **approximate** this delay.

The Story of Wireload Models

Wireload models reflect the routing overhead of the circuit

- **Parasitic effects** are major contributors to overall delay.
- During synthesis, wireload models **approximate** this delay.
- Each circuit is different, will require a different wireload.

The Story of Wireload Models

Wireload models reflect the routing overhead of the circuit

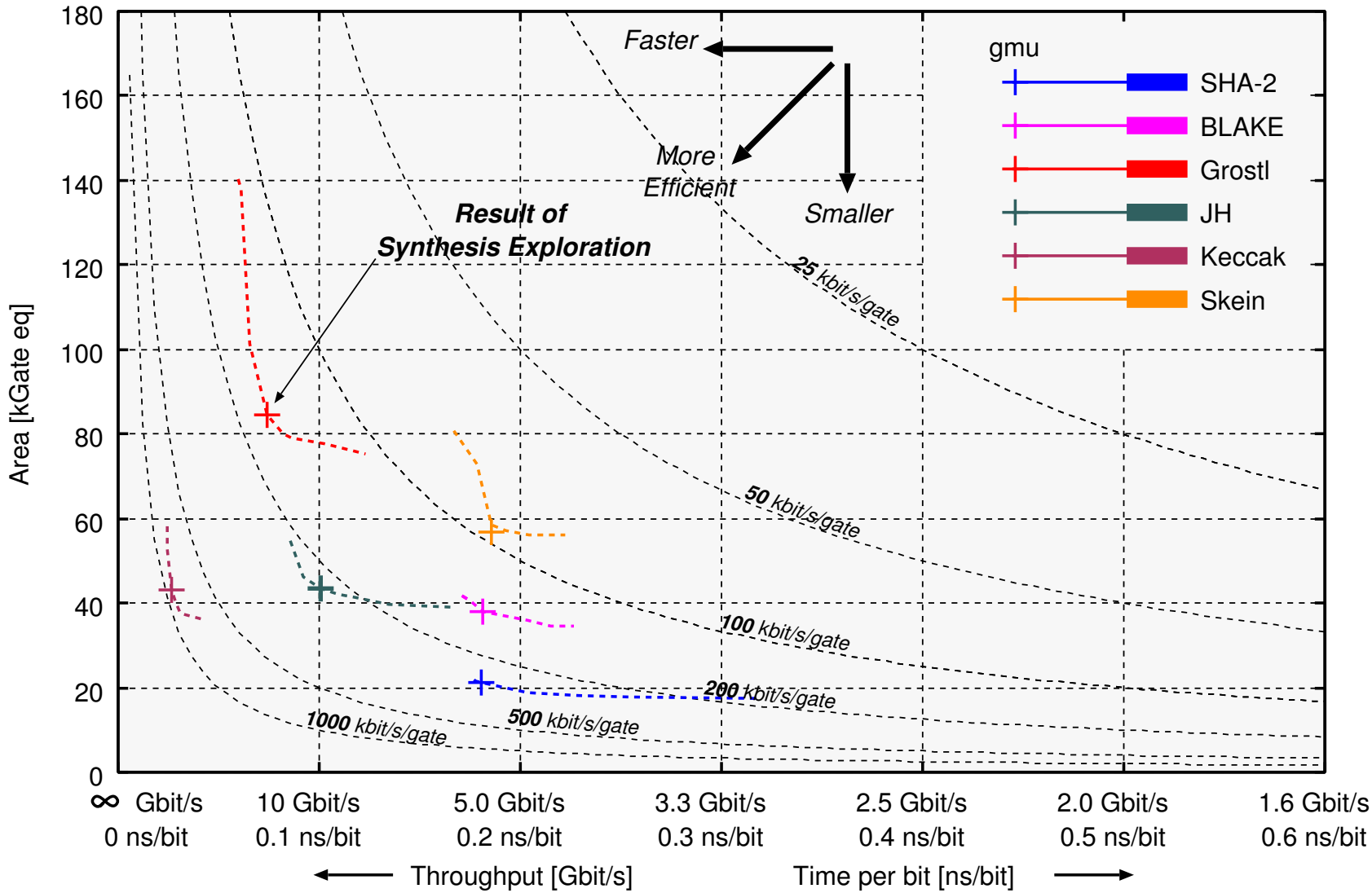
- **Parasitic effects** are major contributors to overall delay.
- During synthesis, wireload models **approximate** this delay.
- Each circuit is different, will require a different wireload.
- Wireload can be **extracted** after place and route.

The Story of Wireload Models

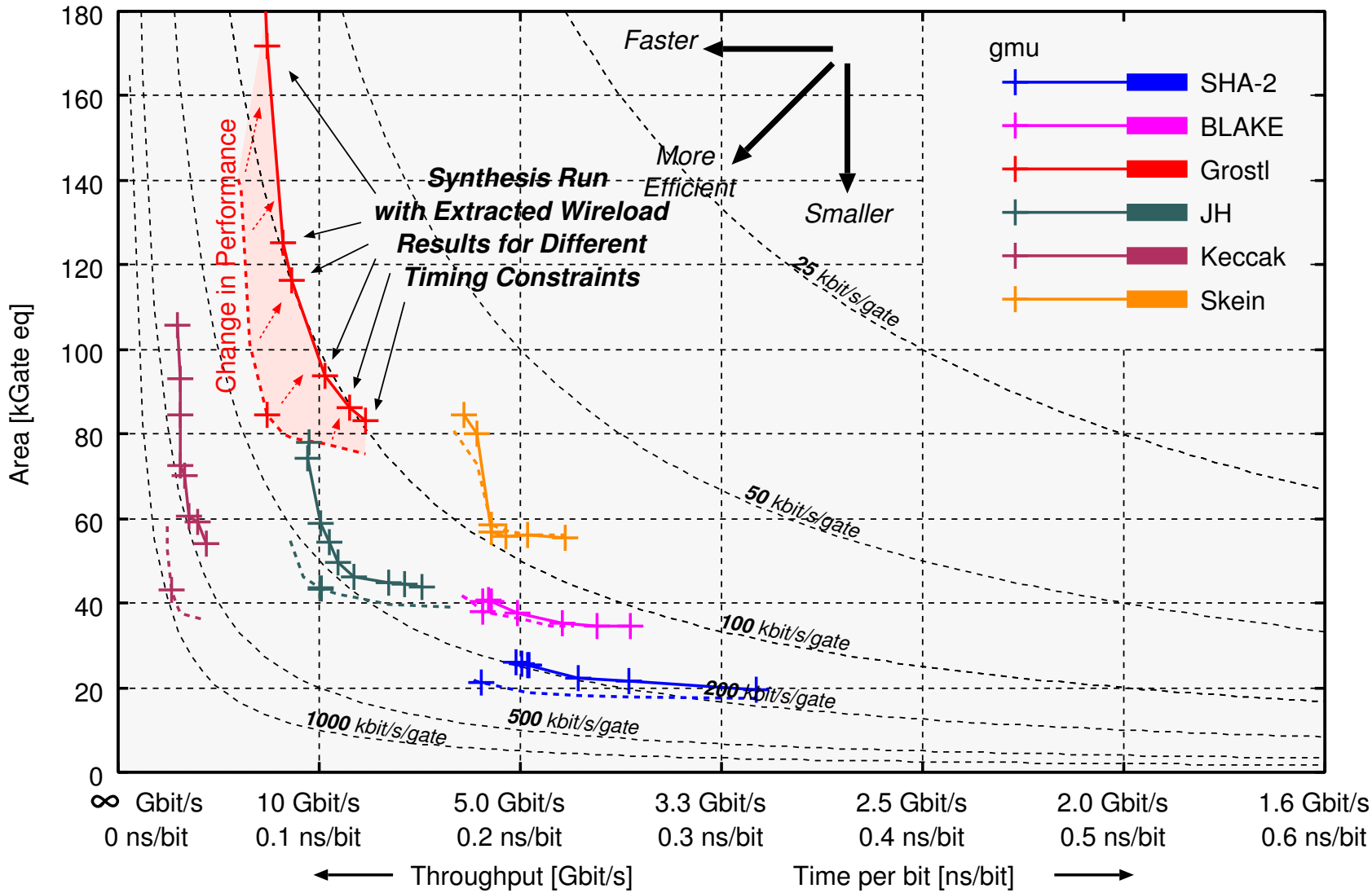
Wireload models reflect the routing overhead of the circuit

- **Parasitic effects** are major contributors to overall delay.
- During synthesis, wireload models **approximate** this delay.
- Each circuit is different, will require a different wireload.
- Wireload can be **extracted** after place and route.
- Subsequent synthesis runs will be **more accurate**.

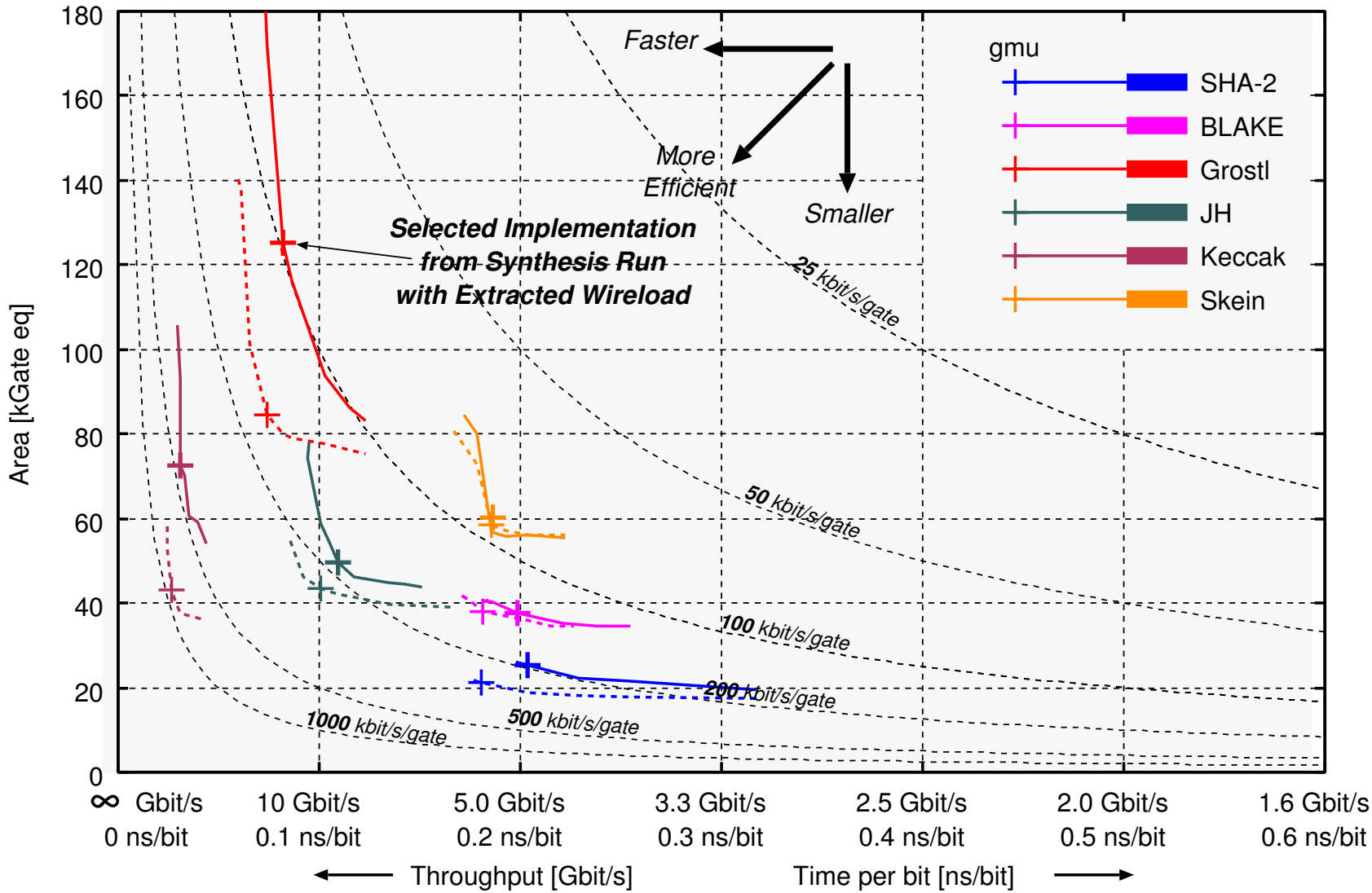
Synthesis Results with Extracted Wireload



Synthesis Results with Extracted Wireload



Synthesis Results with Extracted Wireload



Obtaining Postlayout Results

Cores synthetized separately, combined during backend

- Constraints specified **individually** for each core.

Obtaining Postlayout Results

Cores synthetized separately, combined during backend

- Constraints specified **individually** for each core.
- SoC Encounter can optimize all modes simultaneously.

Obtaining Postlayout Results

Cores synthetized separately, combined during backend

- Constraints specified **individually** for each core.
- SoC Encounter can optimize all modes simultaneously.
- Due to parasitic effects, constraints are relaxed for P&R.

Obtaining Postlayout Results

Cores synthetized separately, combined during backend

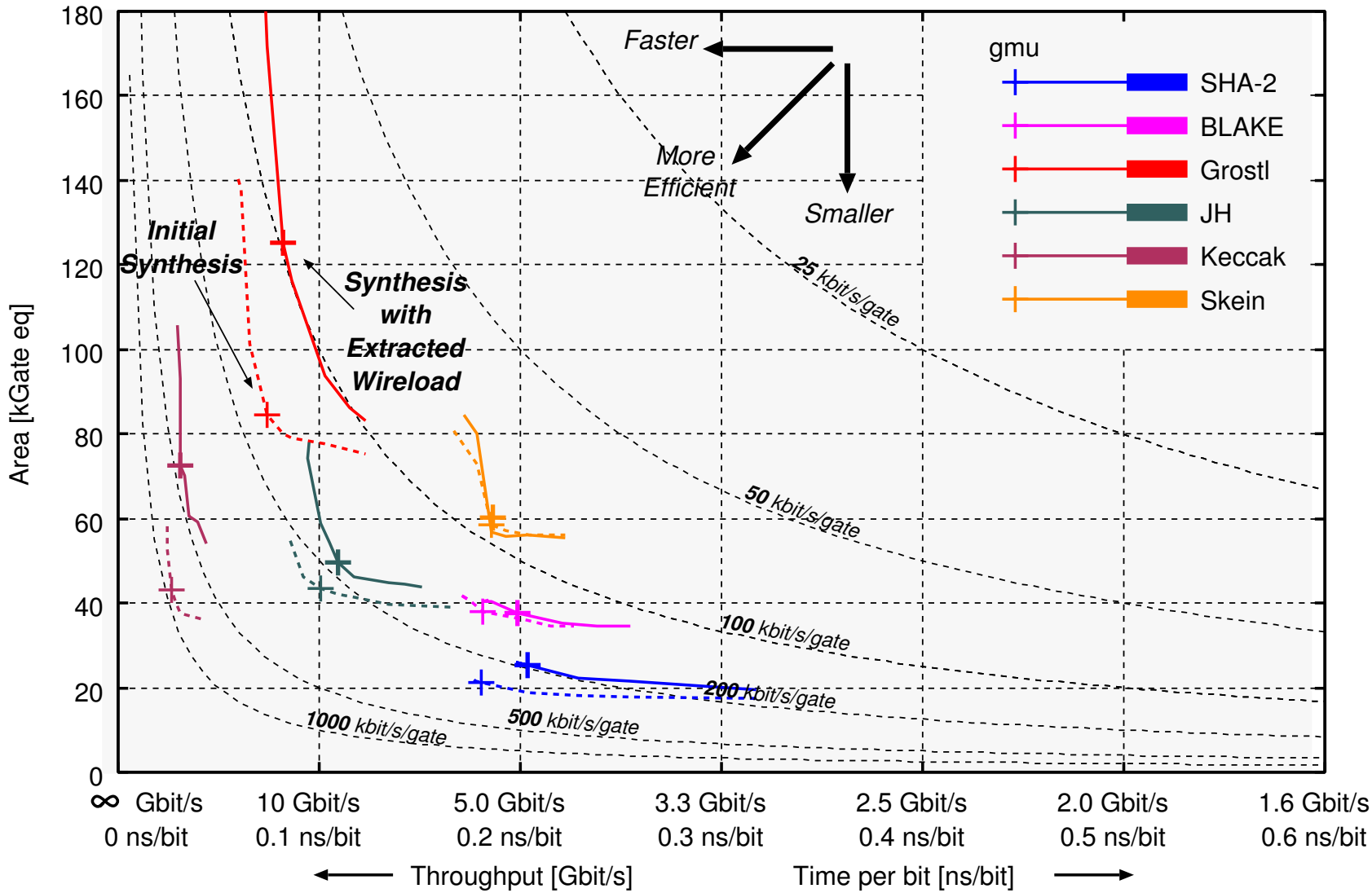
- Constraints specified **individually** for each core.
- SoC Encounter can optimize all modes simultaneously.
- Due to parasitic effects, constraints are relaxed for P&R.
- Backend affects each circui differently.

Obtaining Postlayout Results

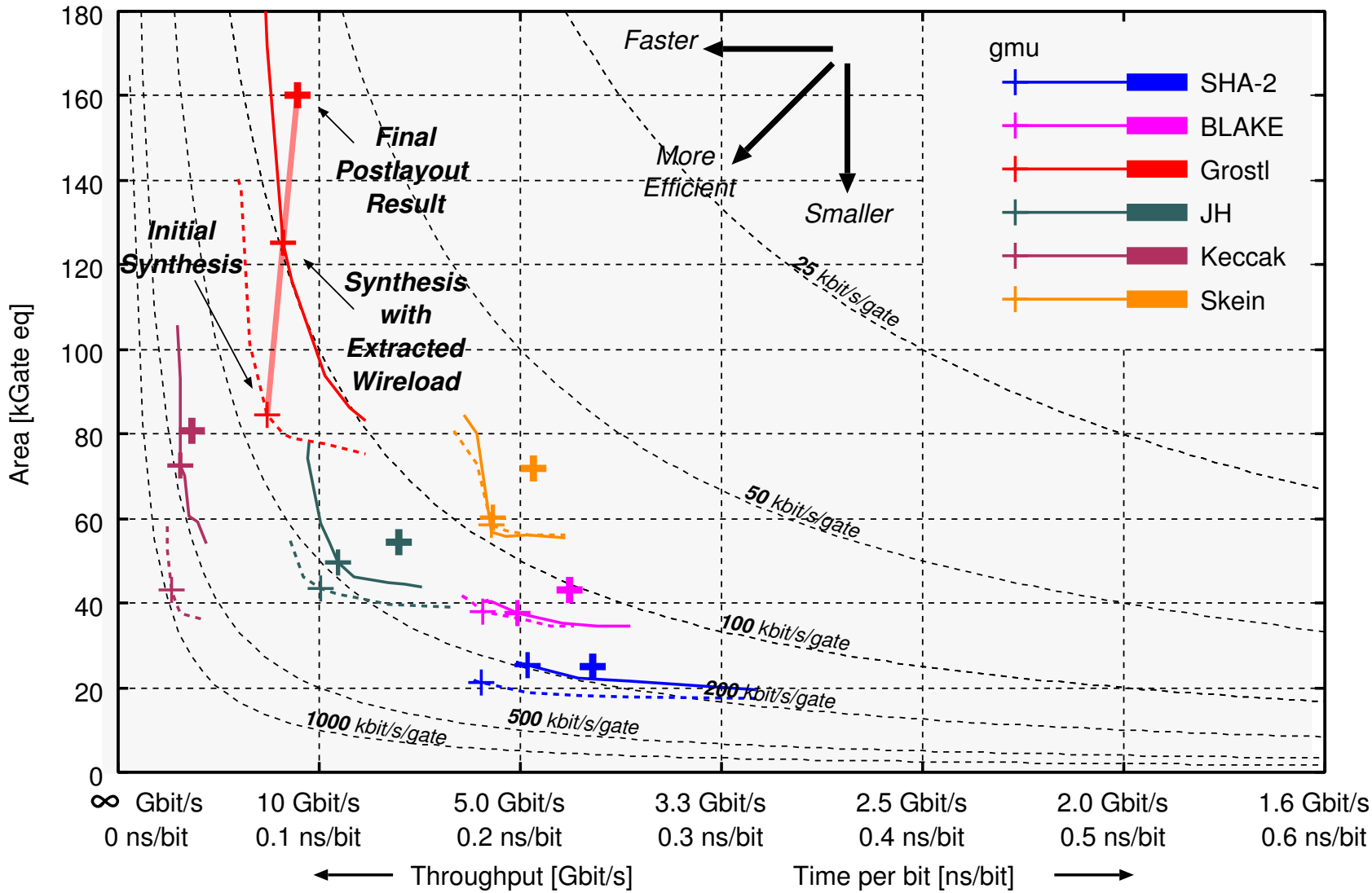
Cores synthetized separately, combined during backend

- Constraints specified **individually** for each core.
- SoC Encounter can optimize all modes simultaneously.
- Due to parasitic effects, constraints are relaxed for P&R.
- Backend affects each circui differently.
- Used several runs to find an **acceptable** solution.

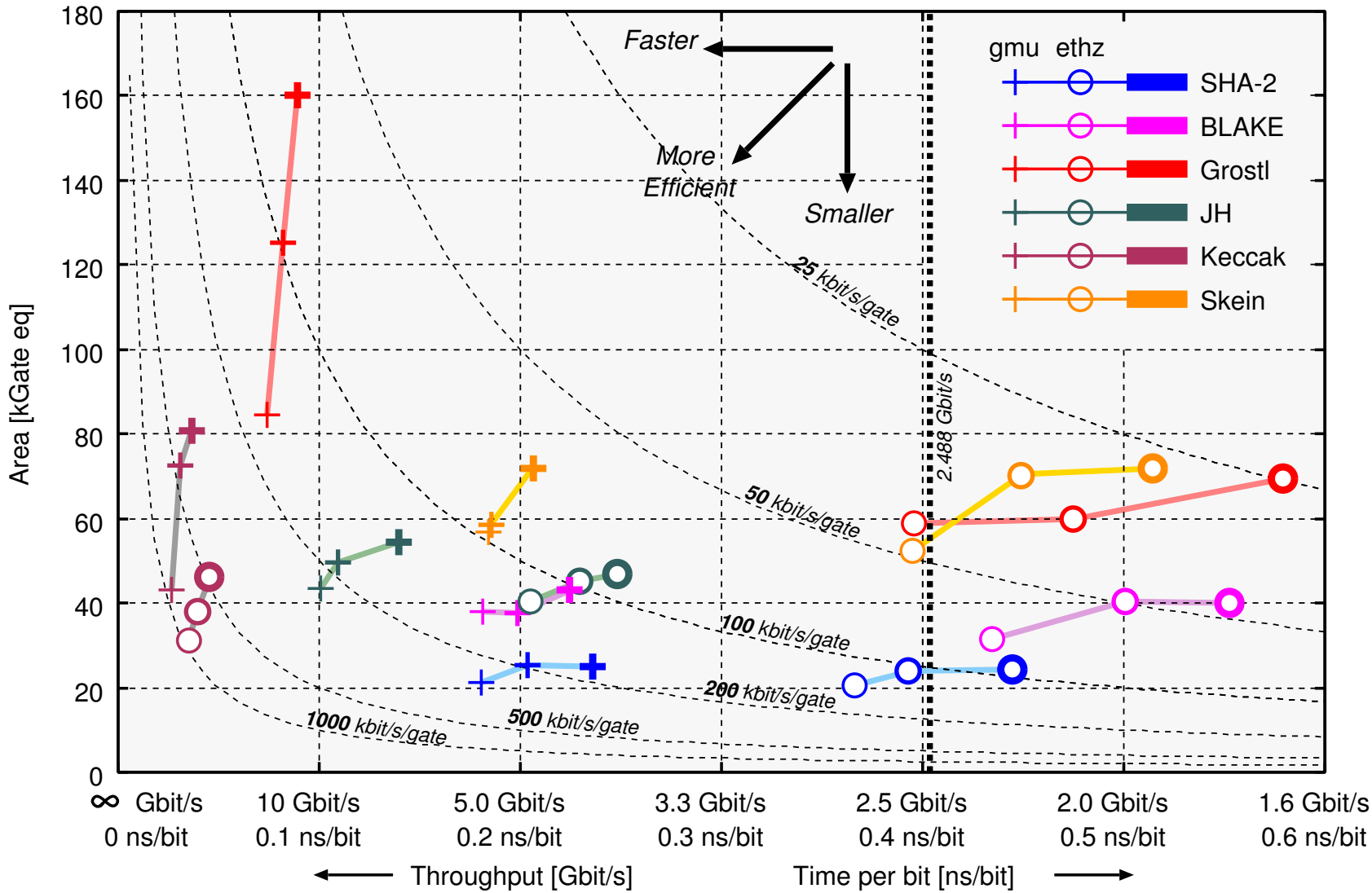
Postlayout Results



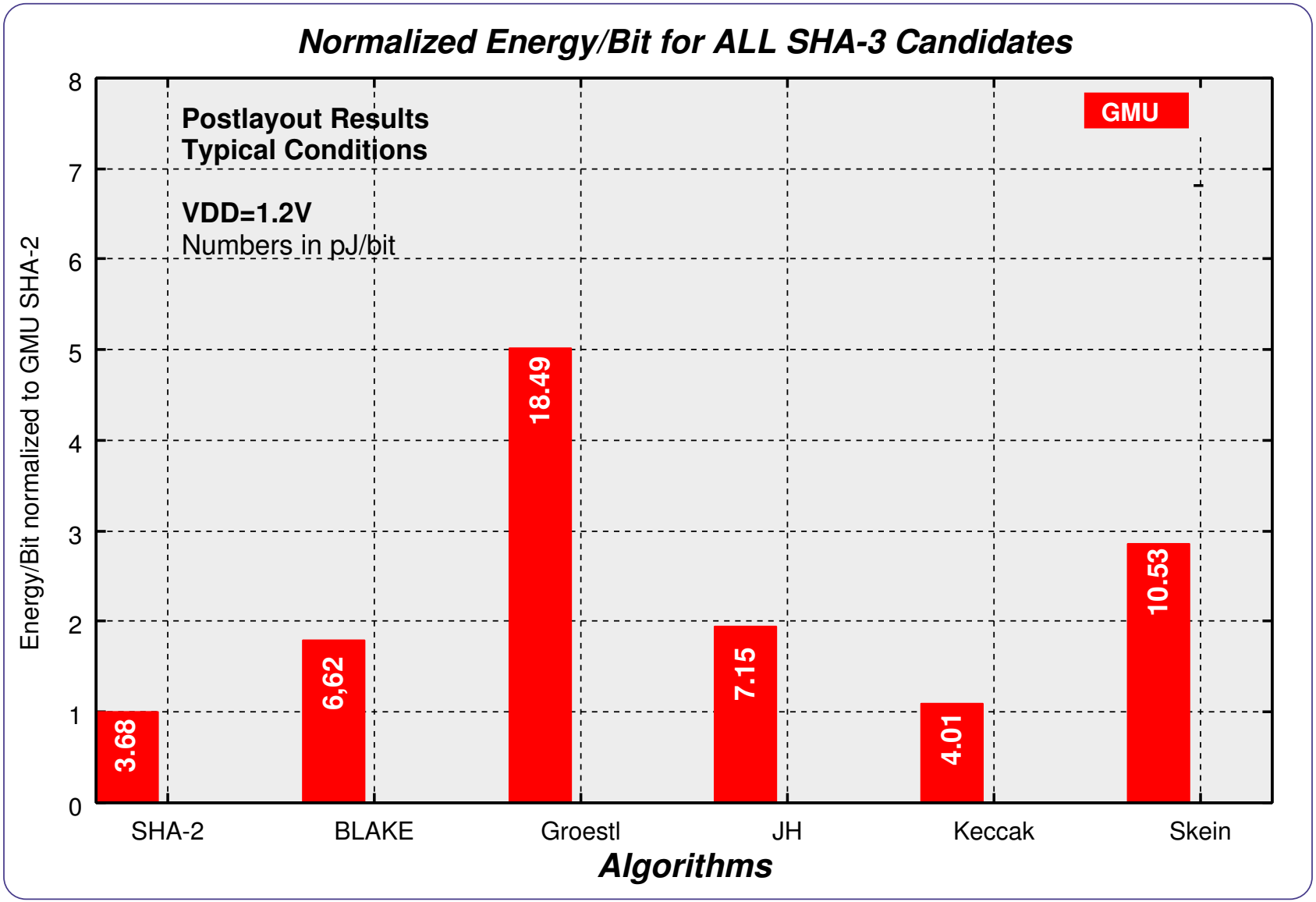
Postlayout Results



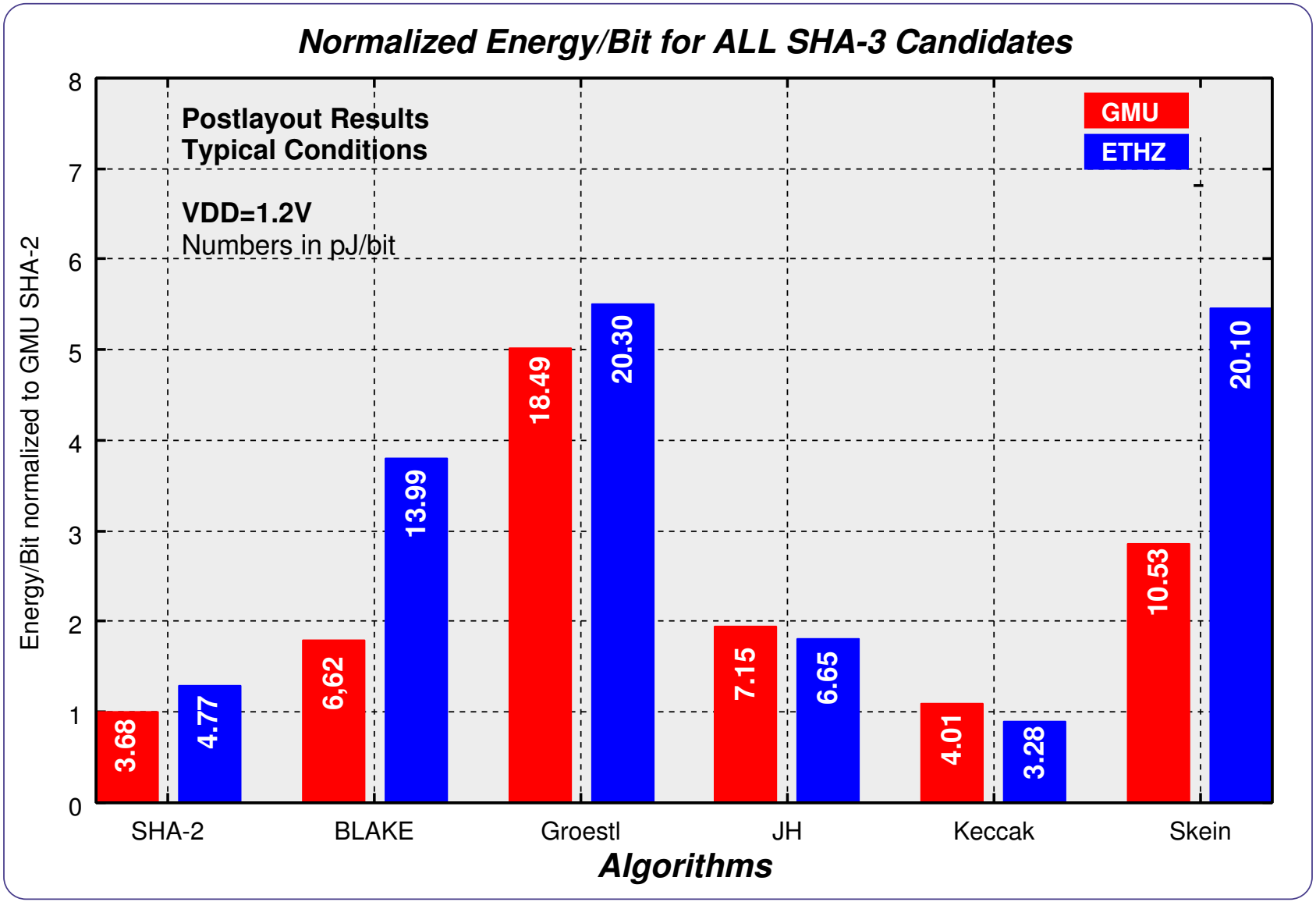
Postlayout Results



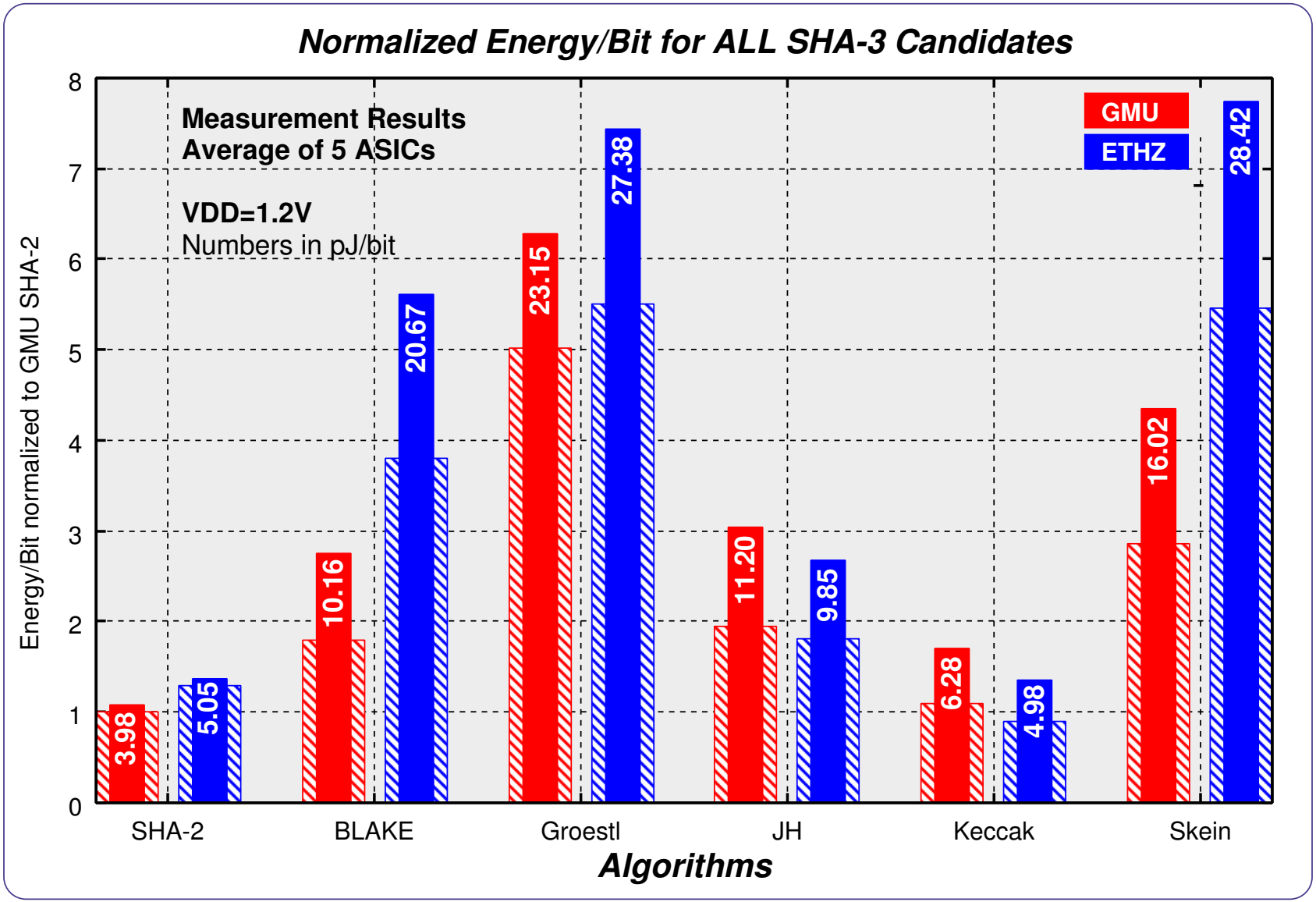
Normalized Energy/bit, Measurement vs Estimation



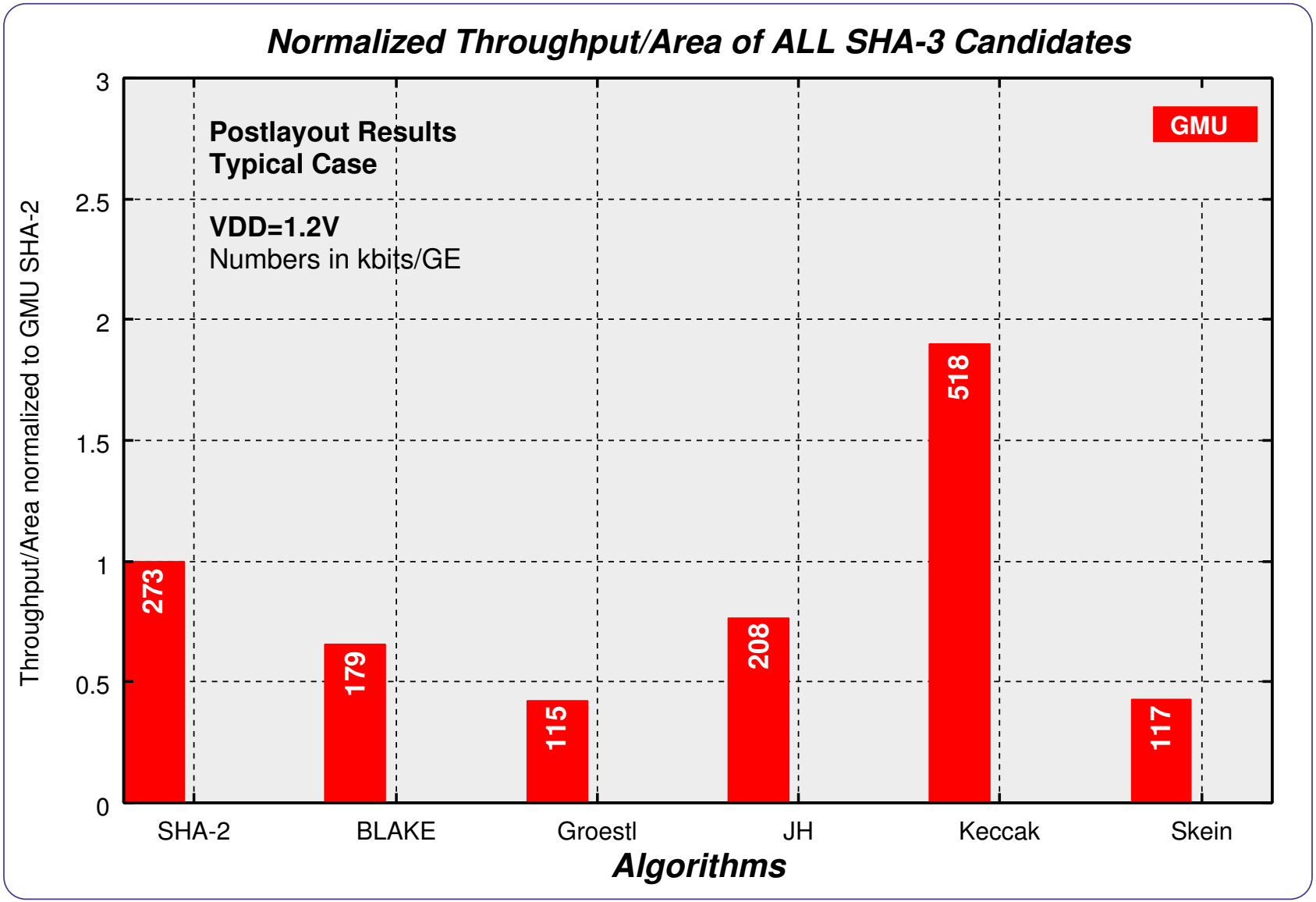
Normalized Energy/bit, Measurement vs Estimation



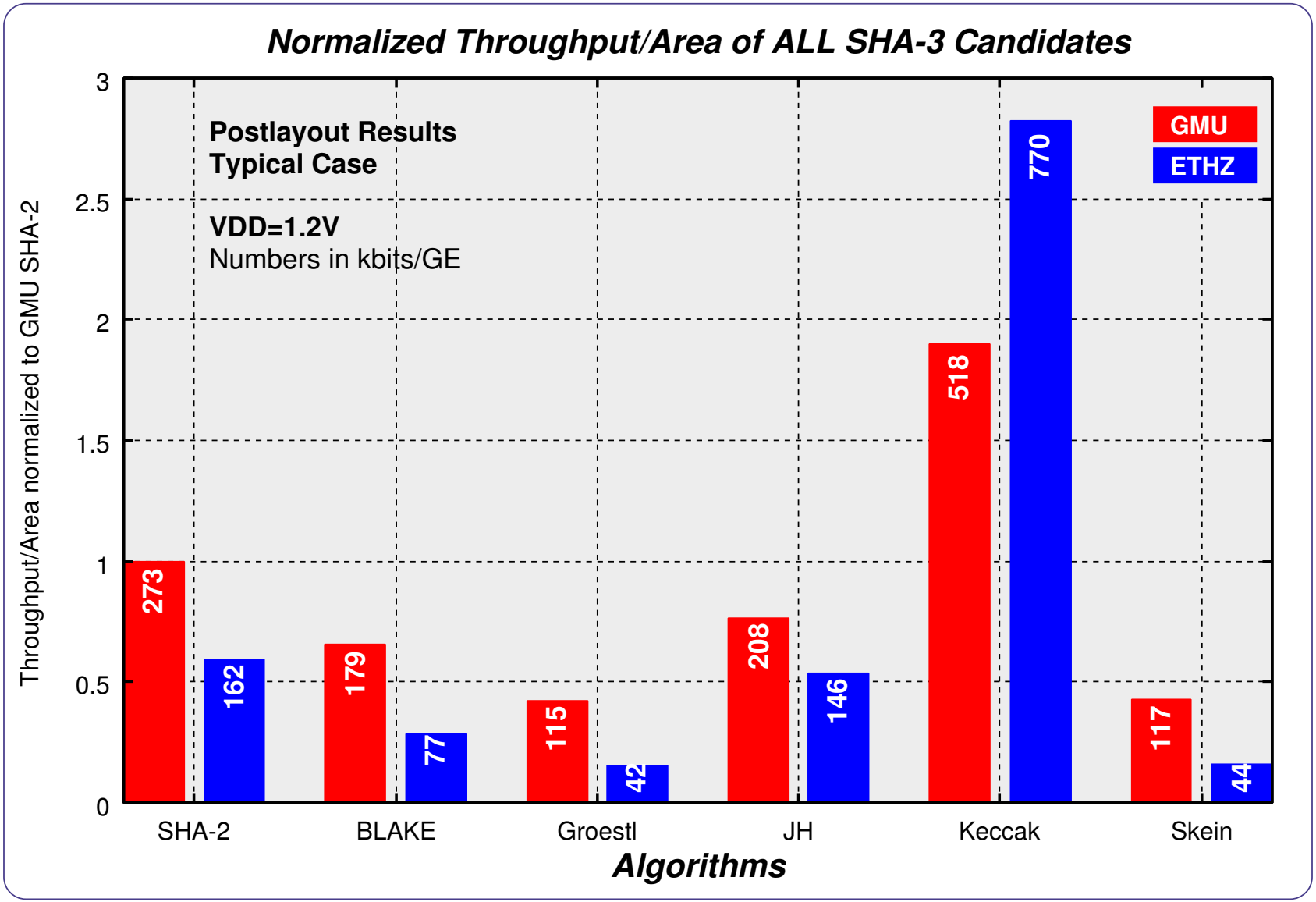
Normalized Energy/bit, Measurement vs Estimation



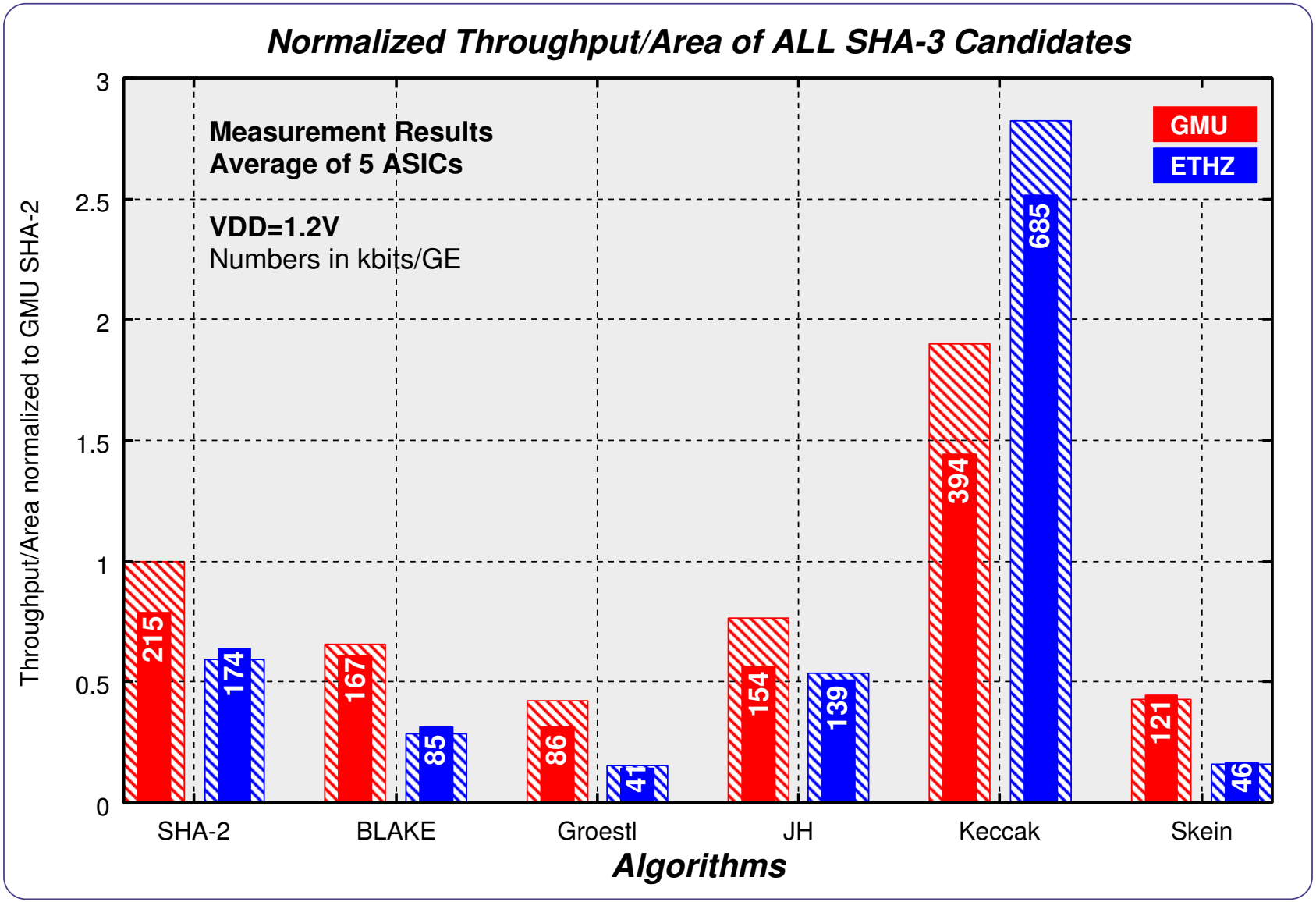
Throughput/Area, Measurement vs Estimation



Throughput/Area, Measurement vs Estimation



Throughput/Area, Measurement vs Estimation



Concluding Remarks (I)

SHA-2

- Very efficient in hardware
- By far the smallest
- Algorithm has been around longer,
  perhaps reason for more optimized implementations

Concluding Remarks (I)

SHA-2

- Very efficient in hardware
- By far the smallest
- Algorithm has been around longer,
  perhaps reason for more optimized implementations

BLAKE

- Compact, easy to implement
- Allows good scalability
- Not the fastest

Concluding Remarks (II)

Grøstl

- Best scalability (Speed/Area tradeoff)
- Low throughput per area
- Cumbersome for hardware

Concluding Remarks (II)

Grøstl

- Best scalability (Speed/Area tradeoff)
- Low throughput per area
- Cumbersome for hardware

JH

- Consistently ranks in the middle
- So far, unable to find good scaling options
- All modes use identical hardware

Concluding Remarks (III)

Keccak

- Hands down fastest algorithm
- Large block size, and small latency key to speed
- Not very good Area/Speed trade-off

Concluding Remarks (III)

Keccak

- Hands down fastest algorithm
- Large block size, and small latency key to speed
- Not very good Area/Speed trade-off

Skein

- Low throughput per area
- Interesting hardware trade-offs due to adder
- Longer combinational delay per clock cycle,
  perhaps reason for better match between expectation and measurement.

Lessons Learned

- **Synthesis results can be far from actual performance**

Lessons Learned

- Synthesis results can be far from actual performance
- Measurement on ASIC is necessary

Lessons Learned

- Synthesis results can be far from actual performance
- Measurement on ASIC is necessary
- Industrial EDA tools ill suited for best performance

Lessons Learned

- Synthesis results can be far from actual performance
- Measurement on ASIC is necessary
- Industrial EDA tools ill suited for best performance
- Different implementations should be compared

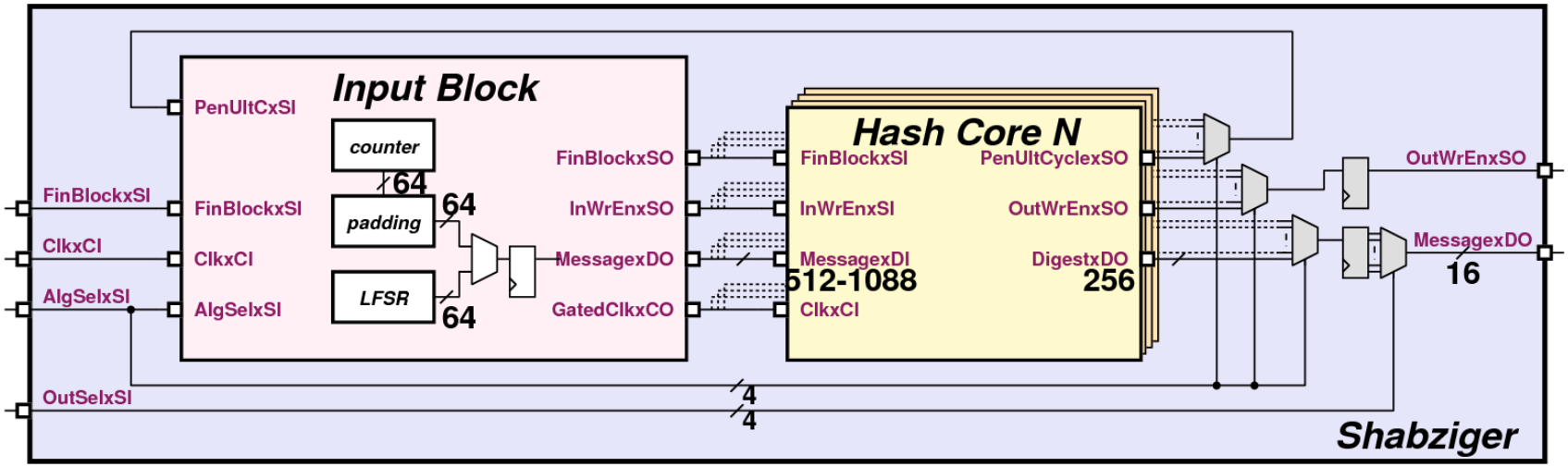
Thank you...

Additional Material

All sources and scripts:

http://www.iis.ee.ethz.ch/~sha3

One ASIC, Many Cores



A common I/O interface for all cores

- LFSR based input assembles random input message
- `FinalBlock` signal tells that current message block is last
- Last message block is padded (fixed padding length)
- All inputs applied parallel, 1088 bits for Keccak, 512 for others
- Multiplexer selects 16-bits out of 256 output bits

Post Layout Results: Speed, Typical Case

Alg.	**Block Size** [bits]	**Impl.**	**Area** (FFs) [kGE]	**Max. Clk** [MHz]	**Tput** [Gbit/s]	**TpA** [kbit/s·GE]
SHA-2	512	ETHZ	**24.30** (29%)	516.00	3.943	162.255
		GMU	25.14 (35%)	870.32	6.855	272.691
BLAKE	512	ETHZ	39.96 (26%)	344.12	3.091	77.347
		GMU	43.02 (34%)	436.30	7.703	179.039
Grøstl	512	ETHZ	69.39 (17%)	460.83	*2.913*	*41.977*
		GMU	*160.28* (9%)	757.58	18.470	115.239
JH	512	ETHZ	46.79 (27%)	558.97	6.814	145.626
		GMU	54.35 (31%)	947.87	11.286	207.655
Keccak	1088	ETHZ	46.31 (25%)	786.16	35.639	**769.550**
		GMU	80.65 (19%)	920.81	**41.743**	517.587
Skein	512	ETHZ	71.87 (19%)	564.33	3.141	43.697
		GMU	71.90 (22%)	312.11	8.411	116.977

Measurement Results: Speed, Average of 5 ASICs

Alg.	**Block Size** [bits]	**Impl.**	**Area** (FFs) [kGE]	**Max. Clk** [MHz]	**Tput** [Gbit/s]	**TpA** [kbit/s·GE]
SHA-2	512	ETHZ	**24.30** (29%)	552.79	4.224	173.826
		GMU	25.14 (35%)	685.40	5.399	214.751
BLAKE	512	ETHZ	39.96 (26%)	377.93	3.395	84.947
		GMU	43.02 (34%)	405.84	7.165	166.541
Grøstl	512	ETHZ	69.39 (17%)	445.63	*2.817*	*40.593*
		GMU	*160.28* (9%)	563.70	13.743	85.747
JH	512	ETHZ	46.79 (27%)	532.48	6.491	138.725
		GMU	54.35 (31%)	704.72	8.391	154.387
Keccak	1088	ETHZ	46.31 (25%)	700.28	31.746	**685.482**
		GMU	80.65 (19%)	701.75	**31.813**	394.456
Skein	512	ETHZ	71.87 (19%)	588.24	3.274	45.548
		GMU	71.90 (22%)	323.21	8.710	121.036

Post Layout Results: Power @2.488 Gb/s, Typical

Algorithm	Block Size [bits]	Imp.	Latency [cycles]	Clk Freq. [MHz]	Power [mW]	Energy/bit [pJ/bit]
SHA-2	512	ETHZ	67	324	11.86	4.76
		GMU	65	316	9.16	3.68
BLAKE	512	ETHZ	57	276	34.80	13.99
		GMU	29	140	16.47	6.62
Grøstl	512	ETHZ	81	392	*50.50*	20.30
		GMU	21	102	46.01	18.49
JH	512	ETHZ	42	204	16.54	6.67
		GMU	43	209	17.80	7.15
Keccak	1088	ETHZ	24	54	**8.16**	3.28
		GMU	24	54	9.98	4.01
Skein	512	ETHZ	92	446	50.00	20.10
		GMU	19	92	26.19	10.53

Measurement Results: Power @2.488 Gb/s - 1.2V

Algorithm	Block Size [bits]	Imp.	Latency [cycles]	Clk Freq. [MHz]	Power [mW]	Energy/bit [pJ/bit]
SHA-2	512	ETHZ	67	324	12.57	5.05
		GMU	65	316	**9.90**	3.98
BLAKE	512	ETHZ	57	276	51.42	20.67
		GMU	29	140	25.27	10.16
Grøstl	512	ETHZ	81	392	68.12	27.38
		GMU	21	102	57.59	23.15
JH	512	ETHZ	42	204	24.51	9.85
		GMU	43	209	27.89	11.20
Keccak	1088	ETHZ	24	54	12.38	4.98
		GMU	24	54	15.62	6.28
Skein	512	ETHZ	92	446	*70.71*	28.42
		GMU	19	92	39.86	16.02