Suggestions for Hardware Evaluation of Cryptographic Algorithms

Frank K. Gürkaynak

Microelectronics Design Center, ETH Zürich

6 July 2012

My Goal

To improve the quality of
hardware evaluations for crypto algorithms.

My Goal

To improve the quality of
hardware evaluations for crypto algorithms.

Problems

- Many degrees of freedom in Hardware Design

My Goal

To improve the quality of
hardware evaluations for crypto algorithms.

Problems

- Many degrees of freedom in Hardware Design
- Is costly (time/money)
  not many are performed

My Goal

To improve the quality of
hardware evaluations for crypto algorithms.

Problems

- Many degrees of freedom in Hardware Design
- Is costly (time/money)
  not many are performed
- Different goals, technologies, metrics

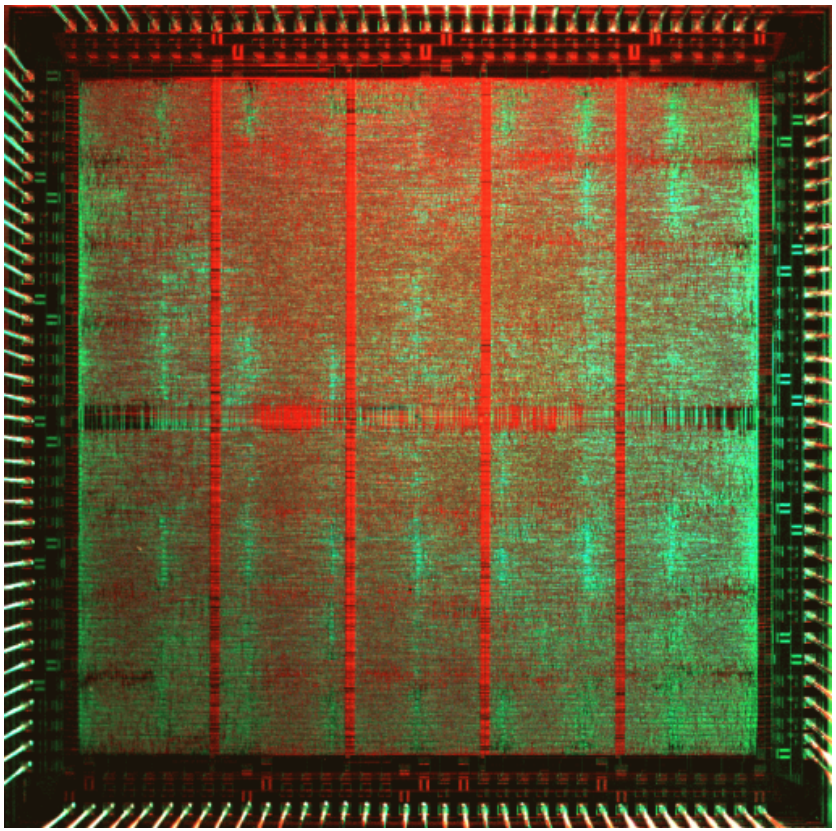
My Goal

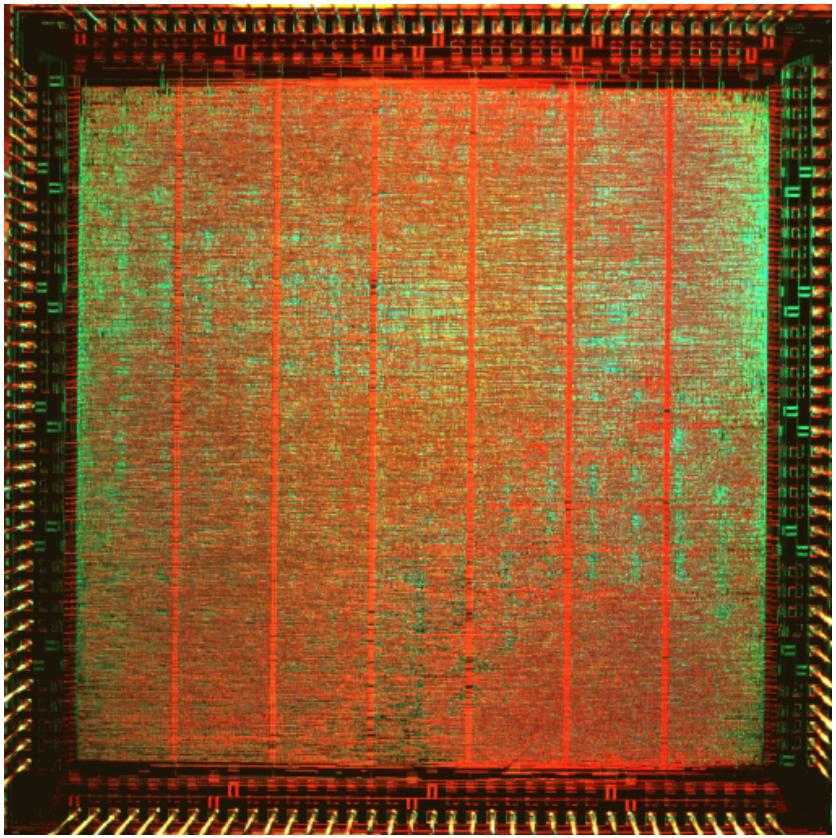
To improve the quality of hardware evaluations for crypto algorithms.

Problems

- Many degrees of freedom in Hardware Design
- Is costly (time/money)
  not many are performed
- Different goals, technologies, metrics
- **Studies are** therefore **not comparable**

My Experience: AES

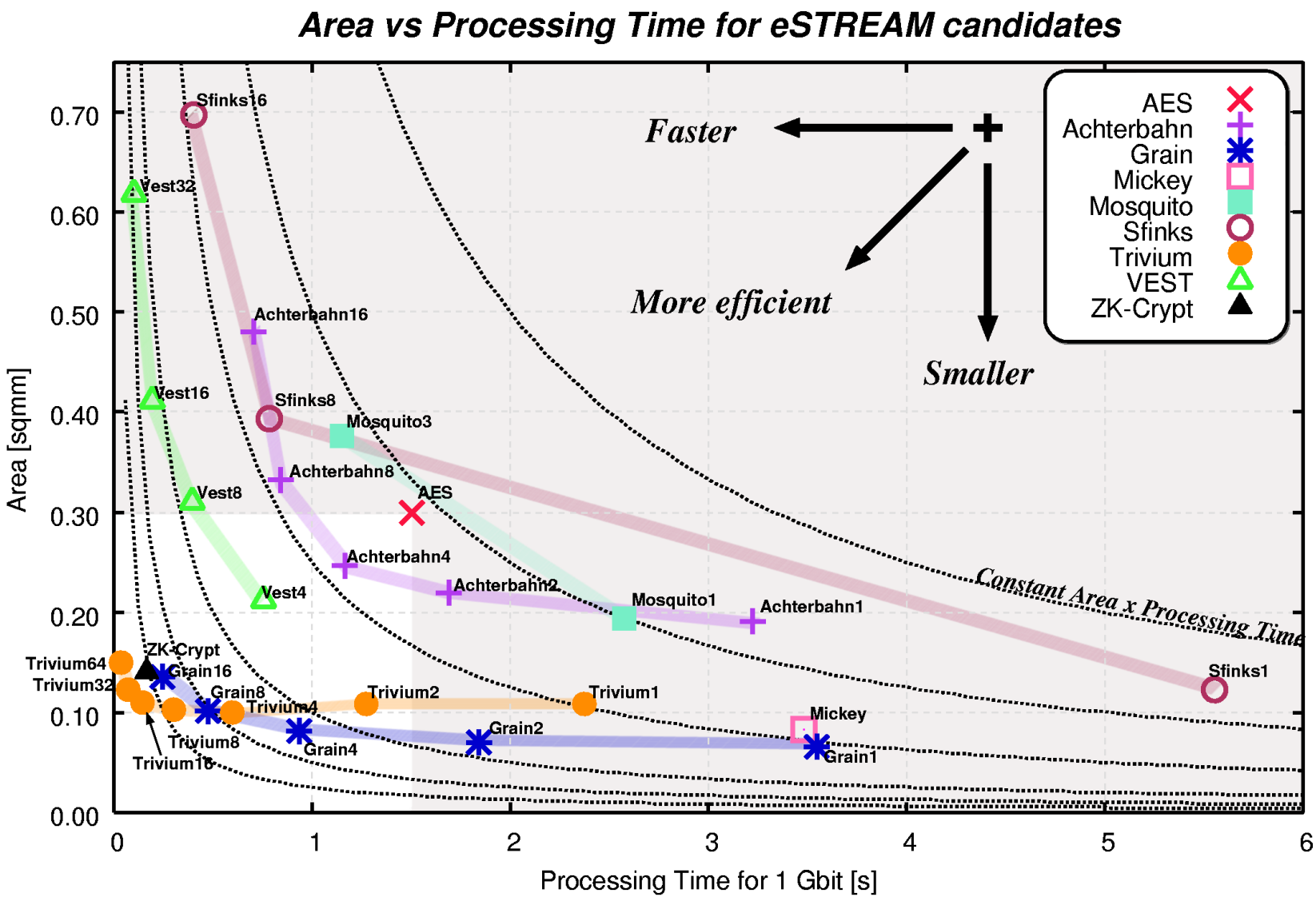




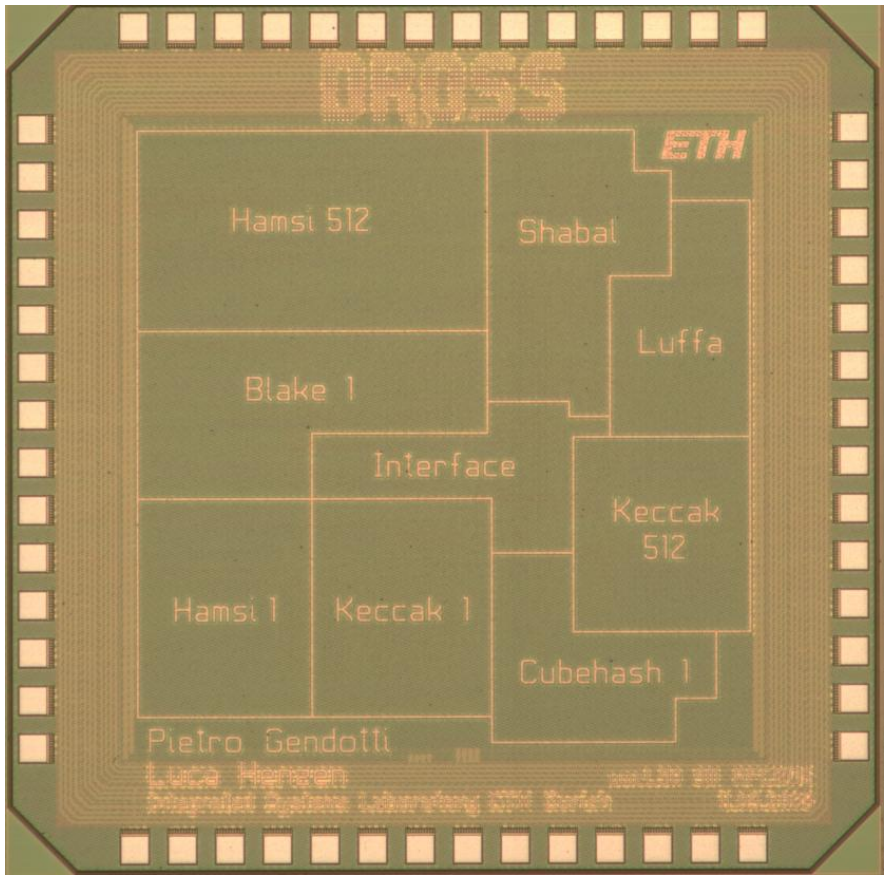
Rijndael and Serpent

http://dx.doi.org/10.1007/3-540-36400-5_12

My Experience: E-stream



My Experience: SHA-3



Collaboration with K. Gaj

http://dx.doi.org/10.1007/978-3-642-15031-9_17

So What is Wrong?

Some observations

- **Hardware design needs constraints!**
  Hardware **design** is an exercise in finding the best circuit that **satisfies** given **constraints**.

So What is Wrong?

Some observations

- **Hardware design needs constraints!**
  Hardware **design** is an exercise in finding the best circuit that **satisfies** given **constraints**.
- **Until now**
  - Implementers choose **some** constraints arbitrarily
  - Results are not comparable between studies
  - **Can not make conlusions**

So What is Wrong?

Some observations

- **Hardware design needs constraints!**
  Hardware **design** is an exercise in finding the best circuit that **satisfies** given **constraints**.
- **Until now**
  - Implementers choose **some** constraints arbitrarily
  - Results are not comparable between studies
  - **Can not make conlusions**
- **Any constraint is better than none**
  The exact constraint is not very important.

What do we suggest ?

To improve the quality of
hardware evaluations for crypto algorithms.

What do we suggest ?

To improve the quality of hardware evaluations for crypto algorithms.

Four suggestions

- Define a **standard testbench**.
- Specify a target **application** scenario.
- State **reference** IC **technology** and FPGA device for comparisons.
- Submitters should **include HDL** code.

#1: Standard Testbench

The call should include a standard testbench for hardware implementations which should read the same KAT files as software models.

Why a Standard Testbench ?

Testbench determines the interface

- I/O bit-widths
- Processing order
- Additional functionality

Why a Standard Testbench ?

Testbench determines the interface

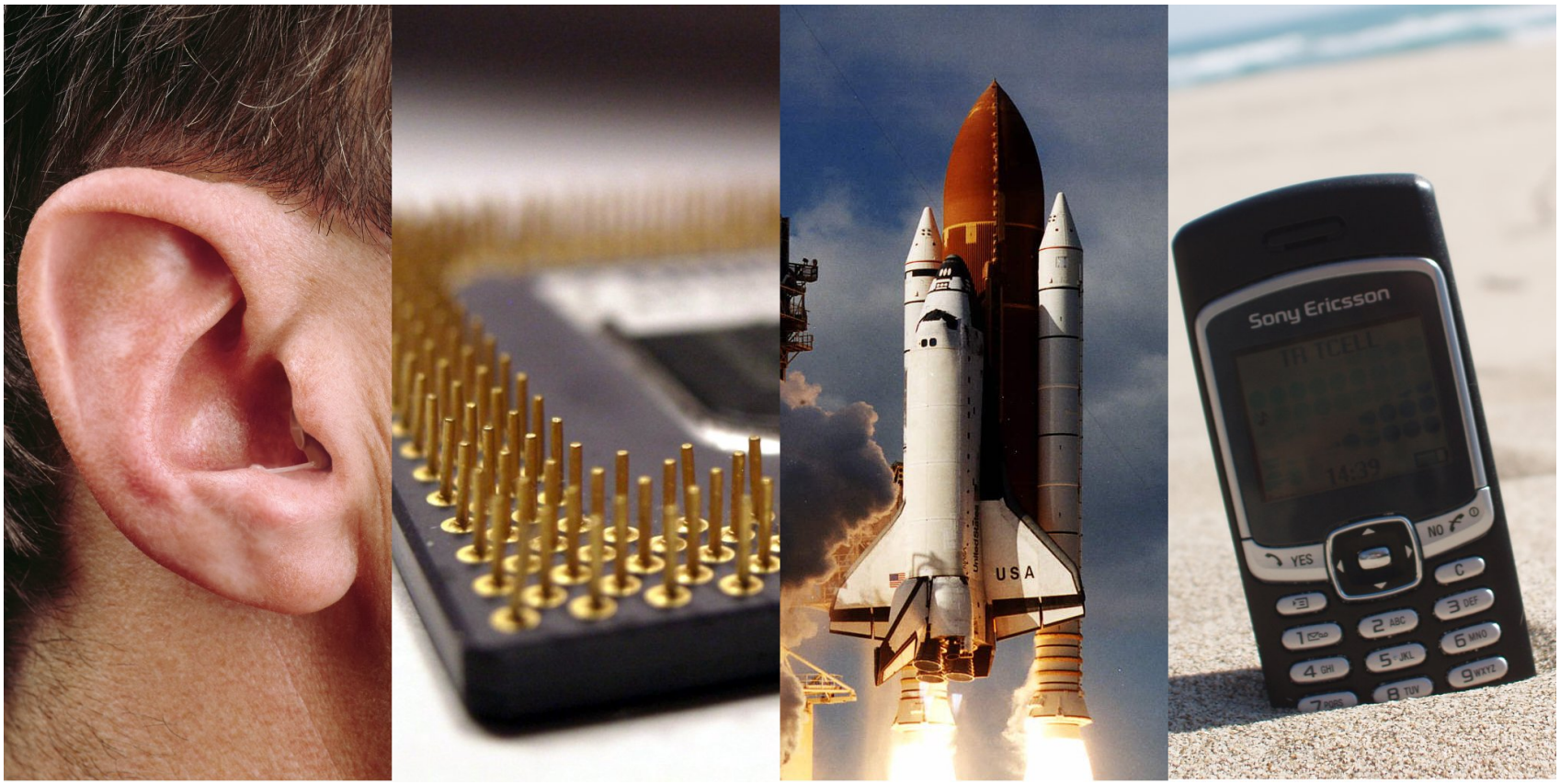
- I/O bit-widths
- Processing order
- Additional functionality

Removes ambiguity about auxillary (non-vital) functions, which may have non-negligable impact on results

#2: Define Application Scenario

The call should include define an application scenario for hardware designs that sets clear constraints for the circuits.

Hardware Design Targets Differ



Scenario Examples

Some suggestions

- **Sensor Node**
  - Messages of 32 to 512 bits
  - Throughput 100 kbits/s
  - Key change every 3 months
  - **Priority** energy consumption
  - Second priority small area

Scenario Examples

Some suggestions

- **Sensor Node**
- **Embedded core**
  - Messages of 32 bits to 4 Mbits
  - Throughput 100 Mbits/s
  - Key change every 10ms
  - **Priority** energy/bit
  - Second priority small area

Scenario Examples

Some suggestions

- **Sensor Node**
- **Embedded core**
- **Data Center**
  - Messages of 2^{10} bits to 2^{64} bits
  - Throughput 10 Gbits/s
  - Key change every 2^{14} bits
  - **Priority** throughput per area
  - Second priority power

Why One Scenario Should Suffice

Defining more than one scenario

- **Is more work**
  Preparing *good* scenarios is difficult.

Why One Scenario Should Suffice

Defining more than one scenario

- **Is more work**
  Preparing *good* scenarios is difficult.
- **Will result in fewer data to compare**
  Groups will choose one of the scenarios, reduce data points

Why One Scenario Should Suffice

Defining more than one scenario

- **Is more work**
  Preparing *good* scenarios is difficult.
- **Will result in fewer data to compare**
  Groups will choose one of the scenarios, reduce data points
- **2 or 3 will also not be enough**
  There will always be an argument for **more**.

Why One Scenario Should Suffice

Defining more than one scenario

- **Is more work**
  Preparing *good* scenarios is difficult.
- **Will result in fewer data to compare**
  Groups will choose one of the scenarios, reduce data points
- **2 or 3 will also not be enough**
  There will always be an argument for **more**.

Suggestion: **One** rather conservative scenario

Standard does not Mean Universal

Actual applications will vary

- **The goal is to achieve uniform results**
  Not necessarily optimal for all applications.
  Consistency is more important.

Standard does not Mean Universal

Actual applications will vary

- **The goal is to achieve uniform results**
  Not necessarily optimal for all applications.
  Consistency is more important.
- **Not possible to account for everything!**
  It should not be the goal of the call to do so.

Standard does not Mean Universal

Actual applications will vary

- **The goal is to achieve uniform results**
  Not necessarily optimal for all applications.
  Consistency is more important.
- **Not possible to account for everything!**
  It should not be the goal of the call to do so.
- **Free to evaluate different options**
  Every evaluation should use standard, but free to explore other options.

#3: Choose a Reference Technology

The call should determine a reference technology and standard cell library for ASIC implementations and a reference FPGA device to be used for comparisons.

Even if Same Technology is Used

Significant differences depending on:

- **Manufacturer**
  Performance of UMC != ST != TSMC

Even if Same Technology is Used

Significant differences depending on:

- **Manufacturer**
  Performance of UMC != ST != TSMC
- **Process Options**
  Low-Leakage, High VT, etc..

Even if Same Technology is Used

Significant differences depending on:

- **Manufacturer**
  Performance of UMC != ST != TSMC
- **Process Options**
  Low-Leakage, High VT, etc..
- **Different Standard Cell Libraries**
  Multiple suppliers, specializations (low-area)

Even if Same Technology is Used

Significant differences depending on:

- **Manufacturer**
  Performance of UMC != ST != TSMC
- **Process Options**
  Low-Leakage, High VT, etc..
- **Different Standard Cell Libraries**
  Multiple suppliers, specializations (low-area)
- **Operating Conditions**
  How to compare 65nm 0.9V, with 65nm 1.2V?

Even if Same Technology is Used

Significant differences depending on:

- **Manufacturer**
  Performance of UMC != ST != TSMC
- **Process Options**
  Low-Leakage, High VT, etc..
- **Different Standard Cell Libraries**
  Multiple suppliers, specializations (low-area)
- **Operating Conditions**
  How to compare 65nm 0.9V, with 65nm 1.2V?

Fix all these parameters !

Some common gates in GE

Foundry	Tech [nm]	Library	Scan FF [GE]	Full Adder [GE]	OR4 [GE]	AND3 [GE]
man A	28	lib A	7.67	9.00	5.33	1.67
man A	45	lib A	**7.33**	**6.33**	2.67	2.00
man A	45	lib A2	5.67	5.33	2.33	1.67
man A	65	lib A	4.75	5.50	2.50	1.75
man B	65	lib B	5.75	5.25	2.00	1.75
man B	90	lib C	**5.75**	**8.25**	2.25	1.75
man B	130	lib C	6.25	5.50	n.a.	1.75
man C	130	lib D	7.00	8.00	2.33	2.00
man A	130	lib A	6.00	5.33	2.33	2.00
man D	150	lib E	7.67	5.67	2.00	1.67
man B	180	lib C	6.33	6.33	n.a.	2.00
man B	180	lib F	6.33	5.33	2.00	1.67
man B	250	lib F	9.00	5.33	2.00	1.67
man E	250	lib H	6.00	5.33	2.33	1.67
man F	350	lib G	6.67	5.00	2.00	2.00
man F	800	lib G	5.00	n.a.	2.00	1.67

Synthesis Results Comparison

Foundry	Tech [nm]	Lib	Example design Com. Chip [mm^2]	[kGE]	[%]
man A	28	lib A	0.175	358	94%
man A	45	lib A	**0.240**	**352**	93%
man A	45	lib A2	**0.318**	**300**	79%
man A	65	lib A	0.577	278	**73%**
man B	65	lib B	0.434	302	80%
man B	90	lib C	1.015	324	85%
man B	130	lib C	1.585	310	82%
man C	130	lib D	1.722	379	**100%**
man A	130	lib A	1.929	319	84%
man D	150	lib E	2.849	315	83%
man B	180	lib C	**3.335**	**356**	94%
man B	180	lib F	**2.972**	**307**	81%
man B	250	lib F	8.103	341	90%
man E	250	lib H	8.207	304	80%
man F	350	lib G	15.917	292	77%
man F	800	lib G	146.544	284	75%

Comparison of Multiple Designs

Foundry	Tech [nm]	Lib	Math [%]	AES [%]	Skein [%]	Comm [%]
man A	28	lib A	99%	100%	94%	94%
man A	45	lib A	93%	97%	92%	93%
man A	45	lib A2	79%	80%	75%	79%
man A	65	lib A	**73%**	**76%**	**71%**	**73%**
man B	65	lib B	81%	85%	78%	80%
man B	90	lib C	84%	79%	84%	85%
man B	130	lib C	81%	91%	81%	82%
man C	130	lib D	100%	100%	100%	100%
man A	130	lib A	**84%**	**101%**	**87%**	**84%**
man D	150	lib E	85%	86%	84%	83%
man B	180	lib C	93%	98%	90%	94%
man B	180	lib F	81%	91%	81%	81%
man B	250	lib F	92%	91%	88%	90%
man E	250	lib H	84%	82%	78%	80%
man F	350	lib G	77%	77%	73%	77%
man F	800	lib G	73%	75%	80%	75%

Which Technology to Choose

Points to consider

- **Should be widely available**
  MOSIS, CMP or Europractice

Which Technology to Choose

Points to consider

- **Should be widely available**
  MOSIS, CMP or Europractice
- **Affordable**
  Will allow actual integrations

Which Technology to Choose

Points to consider

- **Should be widely available**
  MOSIS, CMP or Europractice
- **Affordable**
  Will allow actual integrations
- **Not necessarily the latest technology**
  - No inherent advantage for comparisons
  - Models usually not so reliable
  - Suffer from more second and third effects
  - Expensive

Also Define FPGA Device

No Longer Platform Independent

- **Many, specific resources available**
  Constantly increasing
- **Compilers get better everyday**
  More resources can be utilized automatically
- **Should just select one for comparison**
  Allow all resources to be utilized.

#4: HDL code

Every submission should include a functional HDL implementation (Verilog/VHDL).

Why Provide HDL Code

If Hardware performance is important, designers should also invest time/effort to know how their algorithm performs in hardware.

Why Provide HDL Code (II)

Advantages

- **Basis for comparisons**
- **More time for detailed evaluation**
- **Hardware will be available earlier**
  Could benefit side-channel analysis

Why Provide HDL Code (II)

Advantages

- **Basis for comparisons**
- **More time for detailed evaluation**
- **Hardware will be available earlier**
  Could benefit side-channel analysis

Problems

- **More work**
- **Not everyone is a hardware designer**
  Look for collaborations, i.e. Blake

Other Issues

- **Side Channel Analysis**
  Early availability of hardware could help

Other Issues

- **Side Channel Analysis**
  Early availability of hardware could help
- **Automation (like eBASH)**
  Reference technology could make this feasible

Other Issues

- **Side Channel Analysis**
  Early availability of hardware could help
- **Automation (like eBASH)**
  Reference technology could make this feasible
- **Limitations on Comparisons**
  Take care not to inadvertently publish EDA performance tables.

Summary

To improve the quality of
hardware evaluations for crypto algorithms.

Summary

To improve the quality of hardware evaluations for crypto algorithms.

We suggest, in future calls:

- Define a **standard testbench**.
- Specify a target **application** scenario.
- State **reference** IC **technology** and FPGA device for comparisons.
- Submitters should **include HDL** code.