Towards Green Cryptography: a Comparison of Lightweight Ciphers from the Energy Viewpoint

Stéphanie Kerckhof, François Durvaux, Cédric Hocquet,
David Bol, François-Xavier Standaert.

Université catholique de Louvain, Institute of Information and Communication Technologies, Electronics and Applied Mathematics, Crypto Group.

Place du Levant, 3, B-1348 Louvain-la-Neuve, Belgium.

Abstract. We provide a comprehensive evaluation of several lightweight block ciphers with respect to various hardware performance metrics, with a particular focus on the energy cost. This case study serves as a background for discussing general issues related to the relative nature of hardware implementations comparisons. We also use it to extract intuitive observations for new algorithm designs. Implementation results show that the most significant differences between lightweight ciphers are observed when considering both encryption and decryption architectures, and the impact of key scheduling algorithms. Yet, these differences are moderated when looking at their amplitude, and comparing them with the impact of physical parameters tuning, e.g. frequency / voltage scaling.

1 Introduction

Lightweight cryptography is an active research direction, as witnessed by the number of algorithms aiming at "low-cost" implementations designed over the last years. Looking at block ciphers, the list includes (but is not limited to) `DESXL` [15], `HIGHT` [13], `ICEBERG` [22], `KATAN` [2], `KLEIN` [10], `LED` [11], `mCrypton` [16], `NOEKEON` [3], `Piccolo` [20], `PRESENT` [1], `SEA` [21] and `TEA` [24]. Although these algorithms are useful and inventive in many ways, determining which one to use in which application with good confidence can be difficult. One first reason for this is that the very definition of low-cost is hard to capture, as it is highly dependent on the target platform. For illustration, operations that are cheap in hardware (e.g. wire crossings) may turn out to be annoyingly expensive in software. In fact, even for a given technology, there are various criteria that could be considered to evaluate the low-cost nature of different algorithms. The implementation size (measured in gates, program memory, ...) generally comes in the first place, but power or energy can be more reflective in certain application scenarios. Besides, lightweight cryptography has mainly been developed through several independent initiatives, over an already long time period. This is in contrast with the design of standard algorithms for which the selection was/will be the result of an open competition. One outcome of the Advanced Encryption Standard (`AES`) and `SHA3` competitions is the publication of well motivated comparative studies. Taking the example of hardware (ASIC and FPGA) implementations, several works can be mentioned both for the `AES`, e.g. [7, 8, 23], and `SHA3` candidates [9,

12, 14]. By contrast, only a few evaluations of lightweight algorithms are available in the literature. For example, the companion paper of the KATAN algorithm includes gate counts and throughput estimations for several ciphers [2], but they consider different technologies. A recent initiative can also be mentioned for software implementations [6]. But to the best of the authors' knowledge, there exist no systematic evaluations for hardware implementations to date.

In this paper, we compare the hardware performances of 6 block ciphers, with different block and key sizes. Namely, we considered the `AES` [4] and `NOEKEON` for 128-bit blocks and keys, `HIGHT` and `ICEBERG` for 64-bit blocks and 128-bit keys, and `KATAN` and `PRESENT` for 64-bit blocks and 80-bit keys. This choice of algorithms was motivated by having different block and key sizes, together with different styles of key scheduling and decryption. After a brief discussion underlying the relative nature of evaluation metrics for hardware implementations, we evaluate different figures of merits for these 6 candidates, with a particular focus on the energy efficiency (which explains the word "green" of our title). For this purpose, we first analyze hardware design choices and describe different architectures for encryption, decryption and encryption/decryption, with and without round unrolling and parallelization. This allows us to quantify the combinatorial cost and delays of the different ciphers, and to analyze their respective implementations. Next, we study the tuning of physical parameters, and evaluate the impact of frequency / voltage scaling on our comparisons. Doing so, we investigate the relevance of the energy per bit as a comparison criteria for lightweight block ciphers, i.e. its independence with respect to hardware design choices and frequency / voltage scaling. In other words, we question the extent to which such a metric reflects algorithmic design choices and discuss its possible biases. We answer positively and argue that it nicely summarizes the "energy efficiency" of an algorithm. We also show that the informativeness of this metric is further improved if correlated with the "performance efficiency" (usually measured with a throughput over area ratio). As a conclusion of our experiments, we finally try to extract useful suggestions for new lightweight cryptographic algorithms.

2 Evaluation metrics for hardware implementations

Evaluating hardware implementations is a challenging task. In this section, and as a background to our following case study, we introduce different metrics that can be used for this purpose, together with possible shortcomings with respect to their relevance for comparing algorithms. Namely, we will consider the area, power consumption, throughput and energy cost, as summarized in Figure 1. This selection was motivated by the fact that these metrics are generally reflective of the application constraints that may be encountered in practice. In general, the most revealing units for these metrics are physical (i.e. μm^2, Watts, bit/sec and Joules). However, as these physical units can only be obtained at the very end of an implementation process, convenient first-order estimates are obtained with the gate count, switching activity (i.e. number of bit transitions per clock cycle), number of cycles per algorithm execution and block size.

application constraints	physical units	relative to	pre-layout units	HW design goals	algorithmic design goals	relevance w.r.t. algorithms
AREA	um^2	time or energy constraints	#gates	share resources	reduce components cost & versatility	weakly discriminant ***
INST. POWER (dynamic)	W (J/sec)	time or energy constraints	switching activity	reduce datapath	reduce components cost & versatility	somewhat arbitrary **
THROUGHPUT	bit/sec	area or power constraints	#cycles (& block size)	unroll, parallelize & pipeline	minimize the total combinatorial cost	very arbitrary *
ENERGY	J/enc, J/bit	area or power constraints	#cycles X POWER	unroll	minimize the total combinatorial cost	somewhat discriminant ***

Fig. 1. Summary of evaluation metrics for hardware implementations.

The first important observation regarding these metrics is that they are always *relative*, meaning that it is usually possible to optimize a single metric quite arbitrarily, if the other ones can be degraded. Hence, in order to make any comparison relevant, it is necessary to agree on some application objectives. For example, the area and power can be relative to time or energy constrains, while the throughput and energy can be relative to area or power constraints. As a result, optimization goals can be roughly separated as "design for low area or power" and "design for high throughput or low energy". In the first case, designers will typically share the resources (to decrease the area cost) and reduce the datapath (to limit the switching activity). Such optimizations are illustrated for the case of a block cipher S-box layer in the left part of Figure 2. Quite naturally, re-using the same component also implies that the relative cost of the S-box compared to the control logic and memory decreases, which implies a lower efficiency. Therefore, in the second case, the designer will rather unroll the S-boxes (i.e. implement them all on chip) and parallelize their computation (i.e. perform them in a single clock cycle), as illustrated in the right part of the figure. Besides, the performances of these implementations will also depend on their

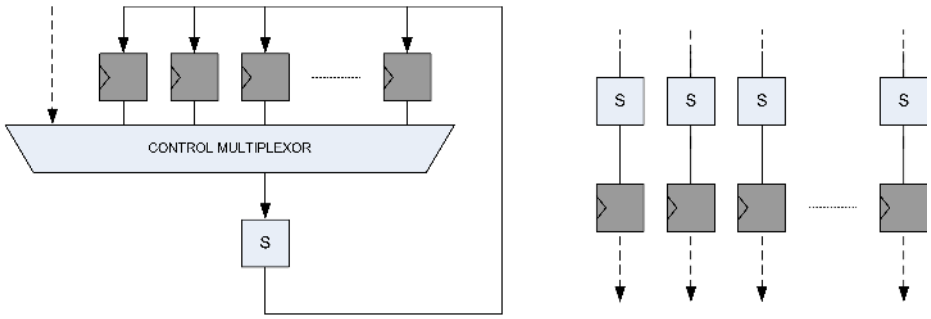

Fig. 2. Left: serial design with resource sharing. Right: unrolled & parallelized design.

clock frequency, for which the maximum value f_{max} is inversely proportional to the longest combinatorial path (aka critical path) between two registers. If the clock frequency is not sufficient, inner pipelining can be used in order to cut the critical path by the addition of registers, as illustrated in Appendix, Figure 6.

Having roughly described these optimization techniques allows us to come back on the relativity of the metrics when comparing different algorithms. For example, the throughput is very arbitrary, as it can be straightforwardly improved by multiplying the circuit size. The same observation holds (to a smaller extent) for the instantaneous power consumption, as a designer could theoretically reduce his datapath to a single bit, independently of the algorithm to implement. In general, a lack of instantaneous power can also be overcome by decreasing the clock frequency and relying on decoupling capacitances. Hence, applications where this metric really matter are quite limited (RFID being the most frequent example). The area cost becomes slightly more discriminant, since increasing the sharing of resources generally implies a cost penalty in the control part. Finally, the energy per encryption is more discriminant, as it corresponds to an integral over time and is not compressible beyond what is allowed by the total combinatorial cost of an algorithm. Quite naturally, many combined metrics can also be derived, e.g. the "throughput over area ratio" is one of the most popular tool to express the performance efficiency of a given hardware implementation.

The main consequence of this relativity is that the fair comparison of hardware implementations is always specialized to a set of constraints. In the following sections, we will define our methodology for this purpose, and investigate the energy cost of different algorithms, for various hardware architectures. Beforehand, a few more comments about this evaluation are worth being mentioned.

(1) Present hardware design flows make intensive use of automated tools, of which the options highly influence the final performance. For example, imposing stronger constraints on the clock frequency can be automated in this way, at the cost of area increases. In such cases, it is useful to agree on the maximum tolerated penalty (compared to the area obtained without frequency constraints).

(2) Once all design choices have been taken, it is always possible to further tune the performances of an implementation, e.g. by taking advantage of frequency / voltage scaling. This issue will be investigated in Section 5.

(3) As technologies are shrinking to the nanometer scale, a part of their power consumption may become static (i.e. happen independent of the switching activity)[1]. As leakage currents are essentially dependent on the circuit size, it implies that the optimization goals for area and power become closer in this case. Significant leakage currents also have an impact on the energy performances.

(4) In general, comparisons are only meaningful for algorithms with the same block and key size. Yet, different block sizes can sometimes be reflected in the metrics (e.g. by computing the energy per bit rather than per block).

[1] Note that this effect can be mitigated by exploiting low-leakage libraries.

3 The case of 6 block ciphers: methodology

In order to make our performance evaluations as relevant as possible, we defined a strict methodology for all our implementations. It defines requirements on the target architectures, their interface and the implementation flow.

Regarding architectures, and for all the investigated ciphers, we considered encryption, decryption and encryption/decryption designs. The reference point of our evaluations is a standard loop implementation of the `AES` Rijndael, performing one encryption in 12 cycles, taking advantage of the efficient S-box representation of Mentens et al. [18]. This choice was mainly motivated by our low-energy consumption goal. Further reduction of the area (e.g. with 8-bit or 32-bit architectures) would lead to less energy-efficient designs. We also thought that the throughputs of these `AES` implementations (of a few Gbps) were large enough for a wide range of applications. Next, for all the investigated lightweight ciphers, we analyzed the generic unrolled architecture depicted in Figure 3, where N_r rounds are executed per clock cycle. Having at least one full round implemented was again motivated by our low-energy consumption objective. We used this generic architecture in order to determine the number of lightweight cipher rounds that are needed to consume the same area, or that require the same delay as an `AES` round. Besides, they are also interesting architectures for very low-latency implementations. As unrolling without adding pipeline is generally a suboptimal choice regarding the critical path, we further considered two implementation scenarios. In the first one, we assumed a clock frequency of 100MHz (determined by the system): it corresponds to a context where such an unrolling is indeed motivated by external constraints. Next, we estimated the maximum clock frequency. In this second case, we further investigated the impact of parallel (or pipelined) architectures. In order to exhaustively analyze our large design space (various N_r values, encryption vs. decryption vs. encryption/decryption, $f = 100$MHz vs. f_{max}), we heavily relied on generic VHDL/Verilog programming. We additionally used a common (generic as well) interface for all the ciphers, with plaintext/ciphertext (resp. key) port width corresponding to the block (resp. key) size, and a simple handshaking mechanism to control the flow.

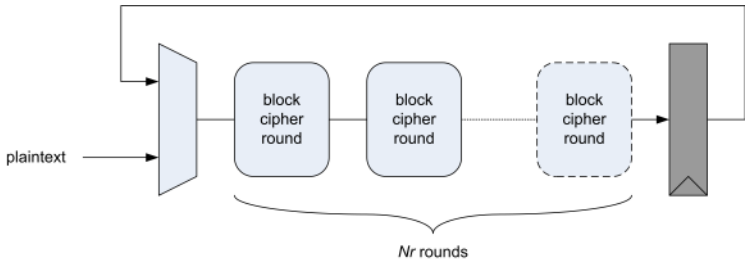

Fig. 3. Unrolled architectures with various number of rounds.

Regarding the synthesis environment, we operated in two steps. In the first place, and in order to study the impact of architectural choices, we investigated the previously defined operating frequencies (i.e. f = 100MHz and f_{max}) at 1.2V, i.e. the nominal supply voltage for the technology used (see Section 4). As previously mentioned, the maximum frequency depends on the synthesis and place-and-route, since the CAD tool can further optimize a design to reach a target timing constraint at the cost of an area increase. Thus, we precisely defined the maximum frequency as the frequency obtained when the area of the design has increased by 10% compared to the unconstrained design. This step allowed us to identify the most efficient architectures for each lightweight cipher. Next, for this reduced set of architectures, we analyzed the possibities of frequency / voltage scaling by carefully tuning the supply voltage (in Section 5).

All the ciphers were implemented following a classical ASIC flow. We used a commercial 65-nanometer CMOS low-power technology. Synthesis was performed using the Synopsys tools suite. We used the switching activity annotation, by means of behavioral simulation, in order to extract realistic power and energy figures. In addition, the supply voltage exploration was performed using the standard cells library that was re-characterized at different Vdd's. Finally, and because of space constraints, we reproduced the most informative metrics provided by our implementations in appendix, and additionally extracted some of them for illustrating our claims in the core of the paper. The remainder of our syntheses data is available online, in the full version of the paper.

4 Implementation results at fixed Vdd=1.2V

Using the previously defined methodology, we first reported the selected performance metrics of our different syntheses at 1.2V supply voltage in Appendix, Figures 8 to 13, where each point in the curves corresponds to a different unrolling parameter N_r. These figures allow us to evaluate the efficiency of the different ciphers implemented. In this section, we report on a number of useful observations regarding both hardware design and algorithmic design issues.

As a starting point, we looked at the area curves (given for f = 100MHz in Figure 8). In general, one would expect the circuit size to increase linearly with the number of rounds unrolled. However, in the case of lightweight ciphers, we observed that a number of rounds may be needed before such a linear dependency appears. This fact is in direct relation with the limited combinatorial cost of the rounds in certain ciphers (most visibly, `KATAN`). That is, if the cost of a round is small in comparison with the state registers and control logic, doubling the number of rounds unrolled will not double the consumed area. A similar behavior is observed for the critical path in Figure 9. If the rounds are simple enough for this critical path to be in the control logic, then doubling the amount of rounds unrolled will not result in cutting the maximum frequency by two. Again, this effect is amplified for `KATAN`, as its round computations only affect a few bits, the other ones being routed from the state register to itself. Overall, these figures recall that the definition of a round is arbitrary: several rounds of a lightweight ciphers are generally needed to reach the cost and delay of an `AES` round.

Looking at the throughput curves first confirms that in general, unrolling an implementation without pipelining it mainly makes sense if the clock frequency if fixed below the maximum one, e.g. because of system constraints. Yet, we also remark that for some ciphers, unrolling a few rounds without pipeline improves the throughput at maximum frequency too (see Figure 10). This is a consequence of "simple rounds" and the previously mentioned non-linear increase of the critical path for low N_r values. Note that even for `KATAN`, the throughput starts to decrease beyond $N_r = 2^6$ (this data is not included in the figures, for visibility reasons). Besides, increasing N_r at maximum frequency does not lead to the expected constant curves. This is explained by a detrimental side-effect related to the overhead cycles required to charge/discharge the plaintext and master key in their registers. Namely, this overhead becomes more significant with the number of implemented rounds (i.e. when the number of clock cycles per encryption becomes small). Note that the impact of this interfacing drawback is stronger for `NOEKEON` and `HIGHT`, as they respectively account for 2 and 3 cycles for these ciphers (one for loading the data, one per initial/final transformation).

The average power implementation results also exhibit different conclusions for $f = 100$MHz and f_{max}. In the first case, they are dominated by the switching activity in the circuit, that increases with N_r. Hence, the power is correlated with the circuit size in this case. By contrast at maximum frequency, unrolling is either neutral or implies a reduction of the average power, when the maximum frequency decreases with N_r faster than the area (e.g. for `HIGHT`).

Interestingly, the energy per bit (given for f_{max} in Figure 11) is remarkably similar in our two frequency contexts, because it is dominated by the switching activity in the selected low-leakage technology. This confirms that it is a reasonably discriminant metric for algorithmic comparisons. It is also quite correlated with the throughput over area metric at 100MHz in Figure 12, i.e. when unrolling the algorithms affects the throughput and area in opposite directions, with close to equivalent impact. Quite naturally, this correlation vanishes at maximum frequency, due to the inefficiency of unrolling without pipeline. Finally, the previously mentioned side-effects (such as overhead cycles and unbalanced use of logic and memory) are naturally reflected in these curves as well.

A summary of our implementation results regarding algorithm efficiency (both in terms of performances and energy) is depicted in Figure 4, where the energy per bit is represented in function of the throughput over area ratio, for our different architectures. Such figures naturally require a few cautionary remarks. First, they have to be interpreted with care, as they only provide a big picture of the implementation efficiency. Practical case studies may focus specifically on different combinations of metrics. Second, even if assuming that efficiency is indeed the design goal, comparing algorithms is difficult as their performances are sometimes close, and depend on the architectures (e.g. encryption-only, decryption-only and encryption/decryption designs lead to different ratings). Yet, we believe that a few interesting conclusions can be extracted that we now detail.

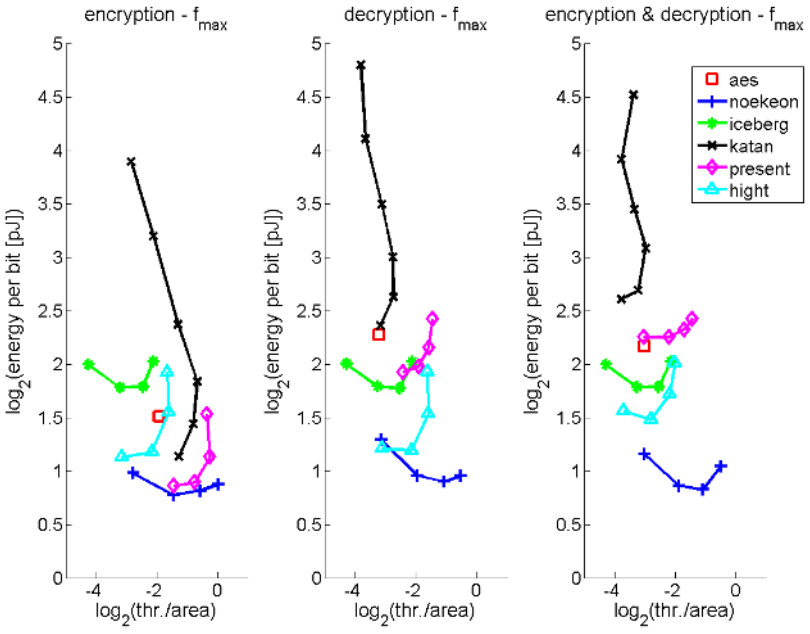

Fig. 4. Vdd = 1.2V: throughput over area ratio vs. energy per bit.

Starting with encryption designs, the comparison roughly suggests NOEKEON $\geq$ PRESENT $\approx$ KATAN $\geq$ HIGHT $\approx$ AES $\geq$ ICEBERG. This ordering is explained by different factors. Maybe the most important one is the significant differences in the key scheduling. At the extremes, NOEKEON does not have any, while for ICEBERG, the key scheduling is as complex as the encryption rounds. Next, the respective block and key sizes strongly matter as well. Having larger key sizes naturally implies lower efficiency in general (but theoretically provides improved security). Less obviously (and less significantly), smaller block sizes are also negative for efficiency. For example, working on 128-bit blocks instead of 64-bit ones doubles the datapath size, but it rarely implies doubling the overall cost (thanks to the strong diffusion layers in modern ciphers). Considering the decryption architectures allows putting forward one more design issue, namely the need to perform the key scheduling "on-the-fly" in forward direction before doing it in backward direction[2] for ciphers such as AES, KATAN and PRESENT. It implies that the comparison is modified into NOEKEON $\geq$ HIGHT $\geq$ ICEBERG $\approx$ PRESENT $\geq$ AES $\approx$ KATAN. Finally, the encryption/decryption designs further indicate the possibility to efficiently share the resources between the cipher and its inverse. Here, involutional ciphers such as ICEBERG gain a particular advantage, leading to a rating: NOEKEON $\geq$ HIGHT $\approx$ ICEBERG $\geq$ AES $\approx$ PRESENT $\geq$ KATAN.

[2] This choice is natural in hardware implementations as storing a fully precomputed expanded key in registers would generally require too large memory oveheads.

An alternative view of the performance and energy efficiency of the different algorithms is given in Appendix, Figure 13 (for $f = 100$MHz), where we plot the ratio between the throughput and the product of the area and the energy per bit. Again, such a figure requires a careful interpretation as they only provide one "global efficiency" metric. Yet, it is interesting to note that for all ciphers, the architecture providing the best such global efficiency has approximately the same latency. Intuitively, this suggest that the computational security of cryptographic algorithms imposes to iterate Boolean functions with a minimum complexity that is somewhat comparable for all ciphers. For illustration, we provide the complete synthesis results for these most efficient architectures for all ciphers in Table 1, where the approximate symbol means that we provide an average for ED figures.

Table 1. Implementation results for most "globally efficient" architectures.

Cipher	Mode E,D,ED	Area [μm^2]	f_{max} [MHz]	Latency [cycles]	Throughput [Mbps]	Power [mW]	Energy [pJ per bit]
`AES`	E	17921	444	12	4740	13,5	2,9
$N_r = 1$	D	20292	377	22	2195	10,6	4,8
	ED	24272	363	≈17	≈2997	≈12,6	≈4,4
`NOEKEON`	E	8011	1149	18	8173	15,0	1,8
$N_r = 1$	D	10431	1075	19	7243	14,1	1,9
	ED	10483	1075	≈18,5	≈7445	≈15,35	≈2,1
`HIGHT`	E	6524	641	19	2159	6,3	2,9
$N_r = 2$	D	6524	645	19	2173	6,3	2,9
	ED	8217	540	19	1820	6,1	3,3
`ICEBERG`	E	11377	699	17	2632	10,7	4,0
$N_r = 1$	D	11359	699	17	2632	10,7	4,0
	ED	11408	689	17	2596	10,6	4,0
`KATAN`	E	6231	952	17	3585	8,1	2,7
$N_r = 16$	D	8616	666	33	1292	9,8	6,1
	ED	12609	473	≈25	≈1347	≈12,7	6,4
`PRESENT`	E	5024	1123	17	4230	9,3	2,2
$N_r = 2$	D	6060	1041	33	2020	8,9	4,4
	ED	8213	884	≈25	≈2523	≈12,6	4,7

Impact of parallelism/pipeline As mentioned in the previous section, unrolling our architectures without parallelizing or pipelining becomes suboptimal at maximum frequency. In cases where the (already high) throughputs obtained in Table 1 are not sufficient for a given application, it is possible to further increase them with parallelization and pipelining. For this purpose, it is natural to start from the efficient architectures with N_r determined as in Table 1. In the first case, we just multiply several circuits as depicted in Figure 7. Since only the control part can be shared between the multiple instances, we essentially double the throughput at the cost of a doubled area (in particular, the control part is small in our implementations and this "doubling rule" was precise up

to a few percents). In the case of outer pipelining, it is additionally possible to spare a few multiplexors. Yet, the trends observed for all metrics and all ciphers are essentially the same as well. In particular for the throughput over area ratio and the energy per encrypted bit (i.e. the two metrics we mainly focus on in this work), we observed similar conclusions for all the investigated ciphers. This behavior is again due to the limited cost of the control part compared to the state registers and the datapath in our block cipher implementations. As doubling the parallelism or pipeline essentially comes at the cost of a doubled area, the throughput over area ratio remains close to constant. Since the same comment applies to the energy per bit (i.e. doubling the throughput doubles the power consumption), we conclude that parallelization and pipeline do not increase the efficiency, nor do they notably affect our comparisons of algorithms.

5 Frequency / voltage scaling

The previous sections investigated the impact of architectural choices on the efficiency of different implementations. We used them to compare different lightweight ciphers. A natural extension of this work is to investigate the impact of physical parameters, e.g. in terms of frequency / voltage scaling. Two main questions can be investigated in this setting. First, do the algorithm comparisons remain unchanged with variable supply voltage? Second, are the efficiency differences between different lightweight ciphers significant in front of the differences when tuning a physical parameter. In order to answer these questions, we re-synthesized the "most efficient" implementations of Table 1 at different Vdd's. The library we used for this purpose is composed of cells that tolerate supply voltages from 1.2V to 0.4V. Note that, beyond the previously listed remarks about the relative nature of hardware performance comparisons, synthesis results at low supply voltage are particularly sensitive to synthesis options. This confirms the general discussion found in Saar Drimer's PhD dissertation in the context of FPGAs [5]. As a consequence, while we would expect the comparison of algorithms to be fully independent of the frequency / voltage scaling, we observed some curve overlaps in our performance evaluations. Our conclusions are as follows.

From the hardware design point of view and as expected, the critical path increases faster, as the supply voltage decreases (in Figure 14). Hence, the maximum frequency and throughput both decrease non-linearly as well, resulting in a reduction of the throughput over area ratio for low Vdd's. More positively, the reason why we lower the supply voltages is to reach lower power consuming points. This is what we observed in our experiments: the power decreases non-linearly with the supply voltage. However, due to the first observation, this power reduction is also moderated by the critical path increase. As a result, the energy per bit (represented in Figure 15) only decreases close to quadratically with the supply voltage, as expected from the energy required to switch the internal capacitances. Note that for all our syntheses, the leakage currents remained negligible (they would become significant below 0.4V in our target technology).

Regarding algorithms, we again plotted a global view of the power and performance efficiency metrics in Figure 5. This final picture allows us to answer the two previously listed questions. First, the comparisons of the different algorithms and architectures is essentially similar as the ones for Vdd = 1.2V. Yet, and as previously mentioned, some curve overlaps are noticed, due to the increased variability of our synthesis results at low supply voltage. Second, the difference between different algorithms in terms of efficiency is quite limited when compared with the impact of frequency / voltage scaling. For example, the energy per bit can be decreased by an order of magnitude when reducing Vdd (at the cost of a throughput decrease). By contrast, the difference between the various block ciphers investigated roughly corresponds to a factor 2 for this metric. Yet, the gains obtained when looking at combined metrics is non-negligible (i.e. at least it is larger than the variability due to synthesis options), in particular when looking at all architectures (i.e. not only the encryption one).

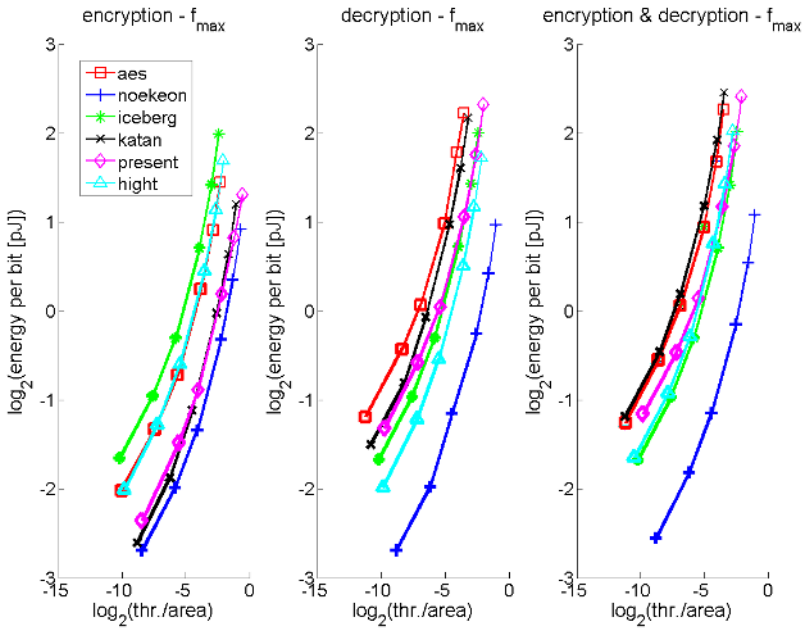

Fig. 5. Voltage scaling: throughput over area ratio vs. energy per bit.

6 Conclusion

This paper provided a first comprehensive comparison of lightweight block ciphers in terms of energy (and performance) efficiency. It confirms that such ciphers do provide interesting figures compared to standard solutions such as the `AES`. However, the gains observed are sometimes limited and may not be sufficient to motivate the use of non standard algorithms in actual applications. Note

that our conclusions are naturally restricted to an energy-oriented case-study. For example, minimizing the area would lead to totally different optimization tweaks (e.g. taking advantage of resource sharing rather than unrolling).

Synthesis results performed for different architectures and supply voltages suggest that using the smallest rounds (e.g. those of `KATAN`) is not the best strategy to reach energy-efficient implementations (because of the too large number of iterations required to complete each encryption). Besides, we noticed that the strong similarity in the block cipher rounds design principles does lead to remarkably comparable implementation figures. In fact, the most meaningful differences between the investigated ciphers relate to key scheduling algorithms and the efficient combination of encryption and decryption designs. Overall, we believe that these results and the general discussion about hardware performance evaluation raise interesting problems for the design of new block ciphers. Namely, finding how to make the algorithmic choices more discriminant with respect to hardware implementations is an interesting research direction.

Acknowledgements. Stéphanie Kerckhof is a PhD student funded by a FRIA grant, Belgium. François Durvaux is a PhD student funded by the Walloon region MIPSs project. Cédric Hocquet is a PhD student funded by the Walloon region MIPSs project. David Bol is a Postdoctoral Researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.). François-Xavier Standaert is an Associate Researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work has been funded in part by the ERC project 280141 (acronym CRASH).

References

1. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. Present: An ultra-lightweight block cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *CHES*, volume 4727 of *LNCS*, pages 450–466. Springer, 2007.
2. Christophe De Cannière, Orr Dunkelman, and Miroslav Knezevic. Katan and ktantan - a family of small and efficient hardware-oriented block ciphers. In Christophe Clavier and Kris Gaj, editors, *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer, 2009.
3. Joan Daemen, Michaël Peeters, Gilles Van Assche, and Vincent Rijmen. Nessie proposal: NOEKEON. Available from `http://gro.noekeon.org/`.
4. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002.
5. Saar Drimer. Security for volatile FPGAs. Technical Report UCAM-CL-TR-763, University of Cambridge, Computer Laboratory, November 2009.
6. Thomas Eisenbarth, Zheng Gong, Tim Gneysu, Stefan Heyse, Stphanie Kerckhof Sebastiaan Indesteege, Franois Koeune, Tomislav Nad, Thomas Plos, Francesco Regazzoni, Francois-Xavier Standaert, and Loic Van Oldeneel. Compact implementation and performance evaluation of block ciphers in ATtiny devices, 2011.
7. Adam J. Elbirt, W. Yip, B. Chetwynd, and Christof Paar. An fpga implementation and performance evaluation of the aes block cipher candidate algorithm finalists. In *AES Candidate Conference*, pages 13–27, 2000.

8. Kris Gaj and Pawel Chodowiec. Comparison of the hardware performance of the aes candidates using reconfigurable hardware. In *AES Candidate Conference*, pages 40–54, 2000.
9. Kris Gaj, Ekawat Homsirikamol, and Marcin Rogawski. Fair and comprehensive methodology for comparing hardware performance of fourteen round two sha-3 candidates using fpgas. In Mangard and Standaert [17], pages 264–278.
10. Zheng Gong, Svetla Nikova, and Yee Wei Law. Klein: A new family of lightweight block ciphers. In Ari Juels and Christof Paar, editors, *RFIDSec*, volume 7055 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011.
11. Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The led block cipher. In Preneel and Takagi [19], pages 326–341.
12. Luca Henzen, Pietro Gendotti, Patrice Guillet, Enrico Pargaetzi, Martin Zoller, and Frank K. Gürkaynak. Developing a hardware evaluation method for sha-3 candidates. In Mangard and Standaert [17], pages 248–263.
13. Deukjo Hong, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bonseok Koo, Changhoon Lee, Donghoon Chang, Jaesang Lee, Kitae Jeong, Hyun Kim, Jongsung Kim, and Seongtaek Chee. Hight: A new block cipher suitable for low-resource device. In Louis Goubin and Mitsuru Matsui, editors, *CHES*, volume 4249 of *Lecture Notes in Computer Science*, pages 46–59. Springer, 2006.
14. Stéphanie Kerckhof, François Durvaux, Nicolas Veyrat-Charvillon, Francesco Regazzoni, Guerric Meurice de Dormale, and François-Xavier Standaert. Compact fpga implementations of the five sha-3 finalists. In Emmanuel Prouff, editor, *CARDIS*, volume 7079 of *LNCS*, pages 217–233. Springer, 2011.
15. Gregor Leander, Christof Paar, Axel Poschmann, and Kai Schramm. New lightweight des variants. In Alex Biryukov, editor, *FSE*, volume 4593 of *Lecture Notes in Computer Science*, pages 196–210. Springer, 2007.
16. Chae Hoon Lim and Tymur Korkishko. mcrypton - a lightweight block cipher for security of low-cost rfid tags and sensors. In JooSeok Song, Taekyoung Kwon, and Moti Yung, editors, *WISA*, volume 3786 of *LNCS*, pages 243–258. Springer, 2005.
17. Stefan Mangard and François-Xavier Standaert, editors. *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*. Springer, 2010.
18. Nele Mentens, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. A systematic evaluation of compact hardware implementations for the rijndael s-box. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 323–333. Springer, 2005.
19. Bart Preneel and Tsuyoshi Takagi, editors. *Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings*, volume 6917 of *LNCS*. Springer, 2011.
20. Kyoji Shibutani, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. Piccolo: An ultra-lightweight blockcipher. In Preneel and Takagi [19], pages 342–357.
21. François-Xavier Standaert, Gilles Piret, Neil Gershenfeld, and Jean-Jacques Quisquater. Sea: A scalable encryption algorithm for small embedded applications. In Josep Domingo-Ferrer, Joachim Posegga, and Daniel Schreckling, editors, *CARDIS*, volume 3928 of *LNCS*, pages 222–236. Springer, 2006.
22. François-Xavier Standaert, Gilles Piret, Gaël Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. Iceberg : An involutional cipher efficient for block encryption in reconfigurable hardware. In Bimal K. Roy and Willi Meier, editors, *FSE*, volume 3017 of *Lecture Notes in Computer Science*, pages 279–299. Springer, 2004.

23. N. Weaver and J. Wawrzynek. A comparison of the aes candidates amenability to fpga implementation. In *AES Candidate Conference*, pages 28–39, 2000.
24. David J. Wheeler and Roger M. Needham. Tea, a tiny encryption algorithm. In Bart Preneel, editor, *FSE*, volume 1008 of *LNCS*, pages 363–366. Springer, 1994.

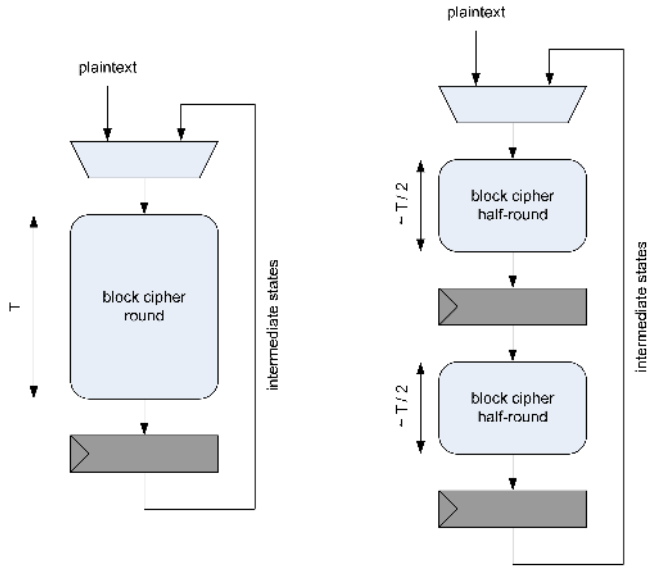

Fig. 6. Inner pipelining of a block cipher round.

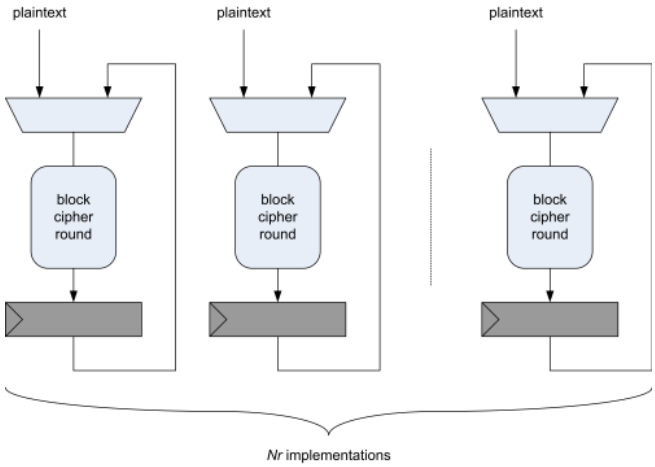

Fig. 7. Parallel architectures with various number of rounds.

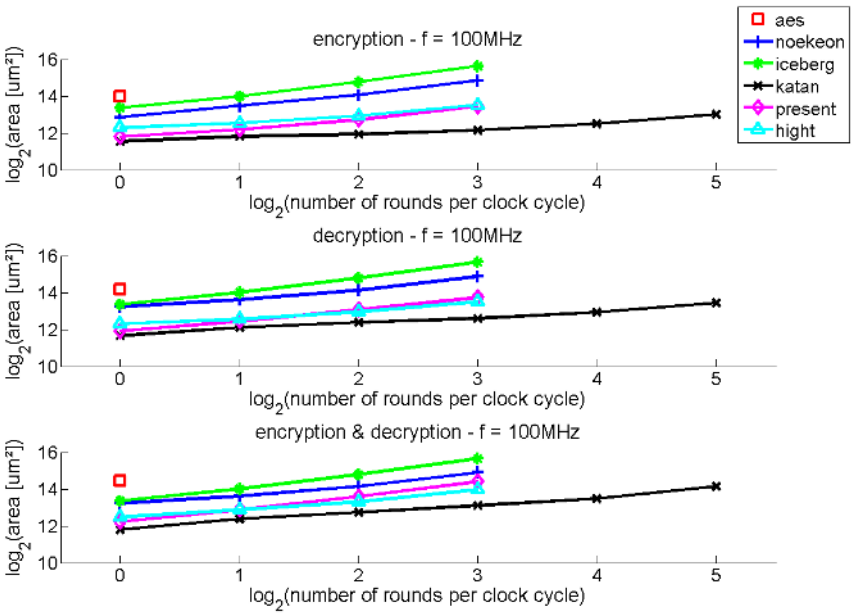

Fig. 8. Vdd = 1.2V, f = 100MHz: area.

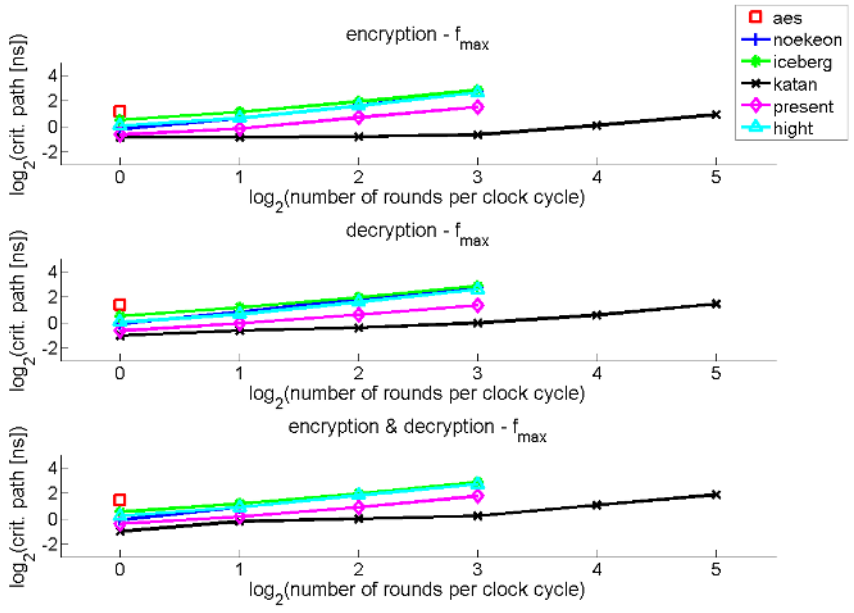

Fig. 9. Vdd = 1.2V, f_{max}: critical path.

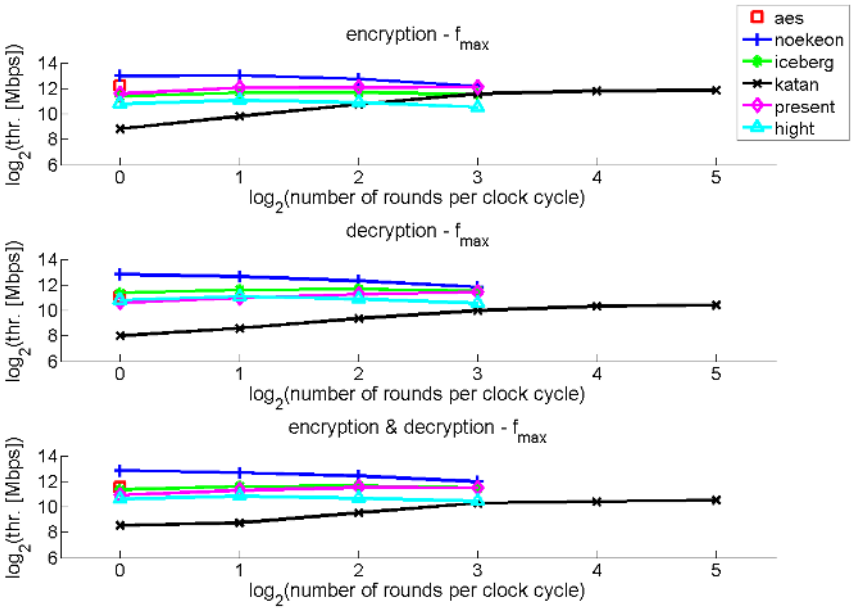

Fig. 10. Vdd = 1.2V, f_{max}: throughput.

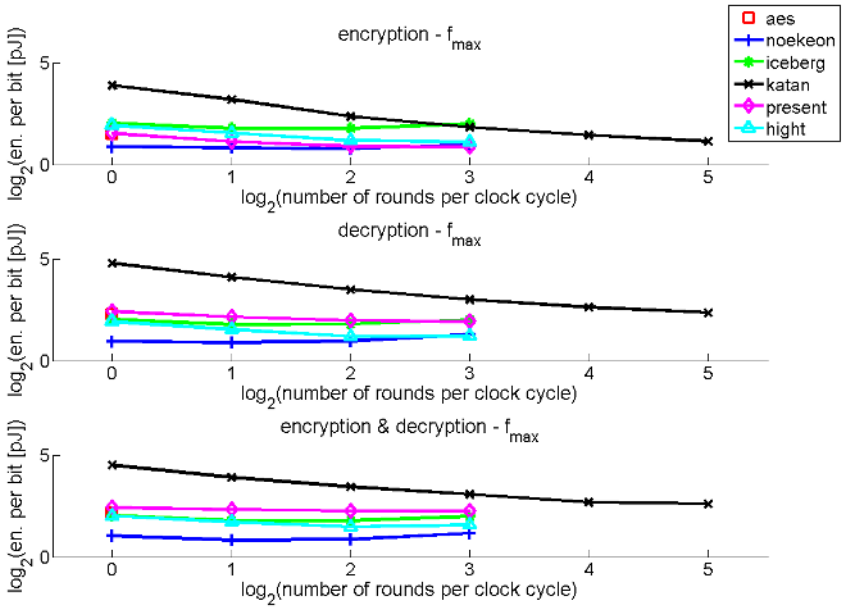

Fig. 11. Vdd = 1.2V, f_{max}: energy per bit.

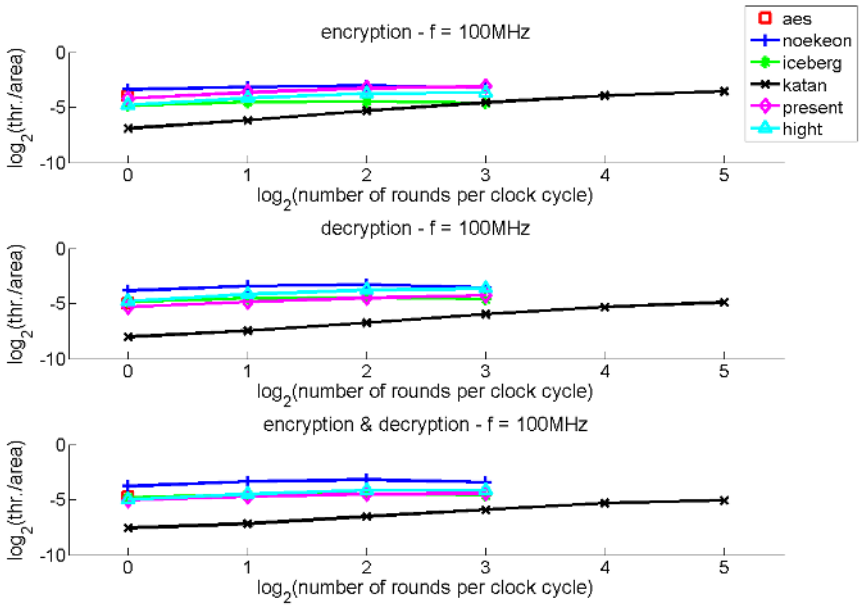

Fig. 12. Vdd = 1.2V, f = 100MHz: throughput over area ratio.

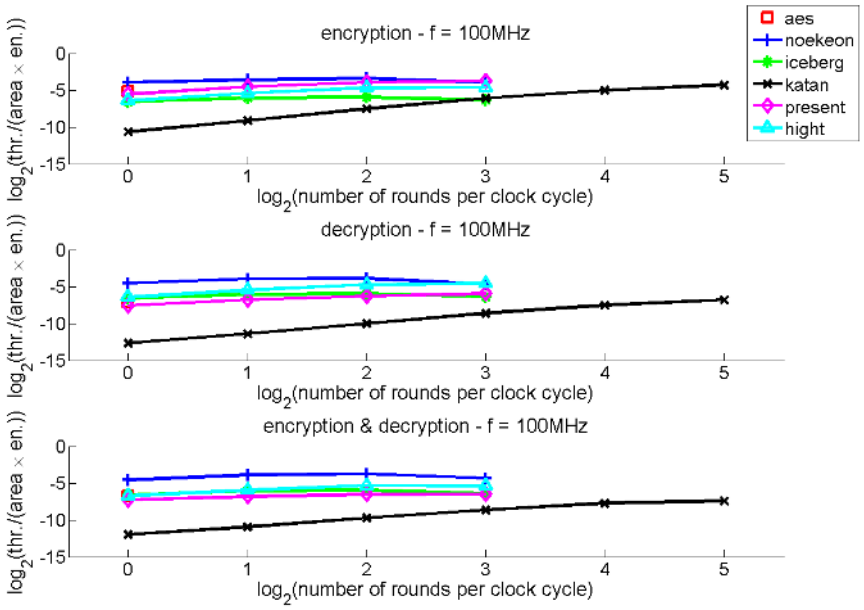

Fig. 13. Vdd = 1.2V, f = 100MHz: throughput over (area × energy per bit) ratio.

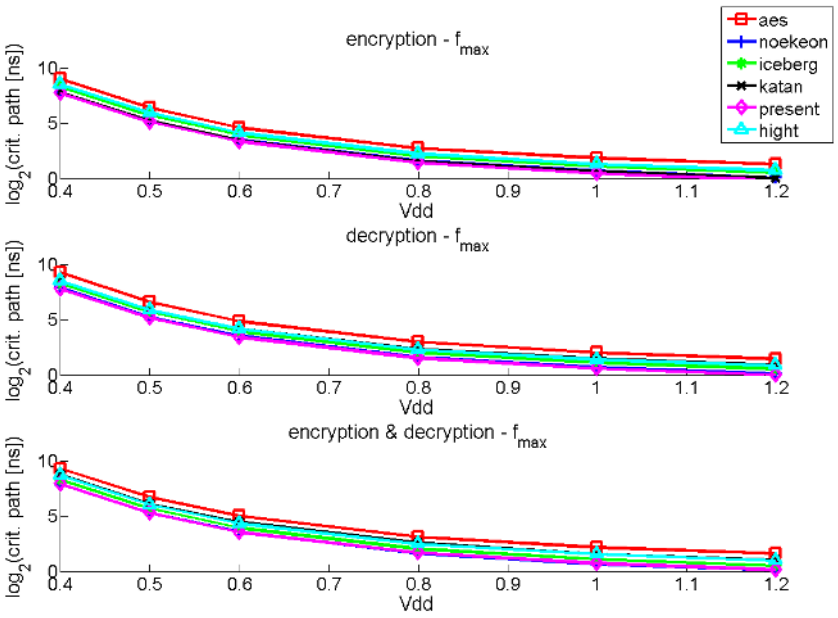

Fig. 14. Voltage scaling: critical path.

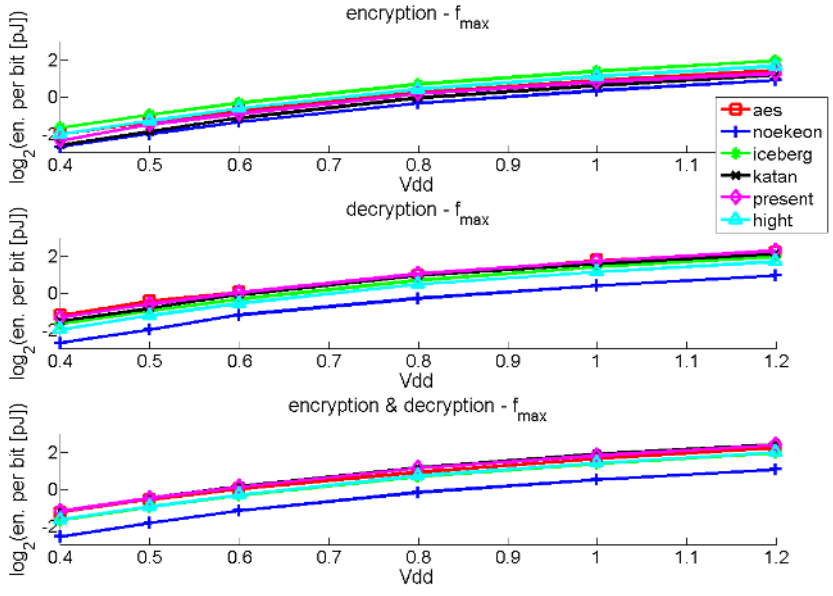

Fig. 15. Voltage scaling: energy per bit.