Comparison of the Power Consumption of the 2nd Round SHA-3 Candidates

Benedikt Westermann[1], Danilo Gligoroski[2], and Svein Knapskog[1]

[1] NTNU/Q2S*, 7491 Trondheim, Norway
[2] NTNU/ITEM, 7491 Trondheim, Norway

Abstract. In the paper we show that the second round candidates of the NIST hash competition differ up to 22 % in their power consumption. We perform a detailed analysis of the candidates with respect to different performance parameters. Finally, we discuss the *time*, the *power consumption*, and the *energy per byte* as criteria to distinguish the candidates with respect to the performance.

1 Introduction

In the last decades the energy consumption of hardware and software have become a serious research topic. While various researchers focus on embedded systems and its energy constraints, other researchers for example analyze the economical challenges of the energy consumption of ICT systems.

In the area of embedded systems one challenge is to implement algorithms in hardware devices as efficient as possible such that the algorithms can run with the actually available power, which may be limited. For example, RFID tags only provide $30 - 50\mu W$[1] to the circuits and therefore this amount of power represents the upper limit for the power available for a computation.

On the global scale the energy consumption is also a serious challenge. In [2] the authors estimate that server farms consume roughly 180 billion kWh each year with a doubling every 4 - 5 years. Thus, the increasing energy consumption is not only a challenge for the electricity industry, but also an economical challenge. An example of Lucas Arts shows that there is a tremendous potential to save costs. The company managed to reduce their energy costs by $ 343,000 a year by selecting hard- and software that consume less energy [3].

With respect to the current NIST hash competition[4] the global influence is rather of minor importance due to the nature of the hash algorithms. However, the power consumption is a meaningful criterion with respect to embedded systems. Nowadays embedded systems have to fulfill more and more security requirements and thus also cryptographic primitives gain more importance and thus also their power consumption. This fact is our motivation to research the energy consumption of the candidates and to evaluate whether it is a well suited

* *"Center for Quantifiable Quality of Service in Communication Systems, Center of Excellence" appointed by The Research Council of Norway, funded by the Research Council, NTNU and UNINETT.* `http://www.q2s.ntnu.no`

criterion for the selection process. Due to the number of second round candidates we chose to evaluate the energy consumption with respect to conventional computers. This has the advantage that we can evaluate all candidates with reasonable costs. The comparison of the power consumption represents the main contribution of the paper. Moreover, we discuss the measured parameters as possible criteria for the selection process of the SHA-3 candidate.

The paper is structured as followed. In Section 2 we explain our methodology and describe the setup of our experiment. The results are presented in Section 3 followed by the related works in Section 4. In Section 5 we discuss our results. The paper concludes with Section 6.

2 Methodology

We have performed our measurements for all candidates of the second round[3] of the NIST Hash competition[4], and measured the power consumption of the 256 bit versions as well as the 512 bit versions. In the following part we describe how we generated the binaries of the candidates. In the last part of this section, we explain how we measured the power consumption of the different candidates.

2.1 Creating the Binary with Supercop

For our measurement we utilized the sources of the second round candidates that are available due to the Supercop project[5]. We used the version 20100818 of *Supercop*. Supercop is a toolkit to measure the performance of cryptographic algorithms amongst others cryptographic hash functions. It tries to find the best suited compiler together with the best suited parameters for the compiler for every algorithm that is included in Supercop. Therefore, Supercop represents for several reasons a well suited tool to compile and create binaries for our measurements. One reason is that the selection process is more objective than just determine some more or less random compiler options for the candidates. The latter could introduce a distortion of the results by selecting a parameter that is non-optimal for one candidate, but optimal for another one.

We used in our measurements the sources of the second round candidates that are included in the Supercop package. This has two advantages. Firstly, the versions of the candidates are mostly more optimized than the versions that are available on the official hash competitions website. Secondly, only small modifications on Supercop are necessary to adopt it to our needs. Thereby, we minimize the risk of introducing bugs which may interfere with the measurements.

The mentioned modifications are necessary, since the binary that is produced by Supercop does not accept any parameters. Thus, it is not possible to determine the amount of data the binary hashes. To this end, we had to modify some files in Supercop to fit the produced binaries to our needs. We modified the `measure.c` in the `crypto_hash` directory to suppress the output and to hash more data in

[3] `http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/index.html`

a single run of the binary. In a single run we hashed 16 MB 1024 times. Thus, in total we hashed 16 GB of data with every execution of a binary and thus with each candidate. Thereby, 16 GB represent a trade-off between the time the measurement takes and the number of samples of the power consumption that could be collected. To avoid unnecessary delay during the measurement, we also fixed the hashed message to a constant value. For both, the 256 bit versions and the 512 bit versions we used the same `measure.c`.

In addition to the modifications of the `measure.c`, we also modified the `do` script of Supercop in order to extract the best performing binary of a candidate. The extracted version was optimized with respect to used CPU cycles and therewith not necessarily optimized to minimize the power consumption.

During the search of the best parameters we have disabled the dynamic frequency selection of the CPU. The CPU was set to a frequency of 2.27 GHz.

On our system there were two different compilers installed that are compatible with Supercop. Supercop used these two compilers and picked the best compiler together with its best suited parameters for each candidate. A `gcc` in version 4.3 was the first compiler which is shipped with the used Ubuntu 9.04 operating system. The second compiler was the `icc` compiler of Intel. We used the version 11.1 20100414 for the `icc` compiler. The extracted binaries were used to measure the consumed power of the candidates.

2.2 Power Measurement

In order to perform the measurement of the power consumption, we utilized the ACPI functions of our test machine, a Dell Studio 1537 notebook. Various parameters are given in the file `/proc/acpi/battery/BAT0/state` on our Ubuntu system. The content of the file is depicted in Figure 1. Some required values, e.g., *present rate* are only available when the machine runs on battery. Thus, we ran all tests on battery and had to recharge the battery after every round of the measurements. We sampled the values of the file every other second. This sample rate roughly correspond to the refresh rate of the parameters in the file.

The parameter *present rate* provides either the current or the consumed energy that is drained from the battery each second. The displayed value depends on the system. Note that the unit of power is energy per second ($\frac{J}{s} = W$).

```
present:                 yes
capacity state:          ok
charging state:          discharging
present rate:            1620 mW
remaining capacity:      2299 mWh
present voltage:         11391 mV
```

Fig. 1. Content of the file /proc/acpi/battery/BAT0/state

Due to a bug in the system, the unit for the current is not correctly displayed. Instead of the correct unit mA the unit is stated as mW. However, this is at least for our used system not correct. In order to compute the consumed power P we need to multiply the voltage U (*present voltage*) with the current I (*present rate*). Thus, the power is calculated by $P = U \cdot I$ and represents the power consumption of the whole system.

For our measurements we are not interested in the power consumption of the whole system, but in the power consumption of the hash algorithms. In order to retrieve the power consumption the hashing of data requires, we first recorded the consumed power in the idle mode. After that, we recorded the power consumption when the system was hashing data. The difference of both values is, in the optimal case, the power consumed by the hash algorithm.

We measured the idle power prior to each run of a candidate. To this end, we sampled the current and the voltage for one minute. In total, we collected 30 samples prior to each run. The samples were stored in a SQLite3 database. By measuring the power consumption prior to every run, we avoid that a slight increase of the power consumption have a significant influence on the long run, e.g. if we consider that the accuracy of measurement depends on the remaining energy in the battery.

After we have collected the 30 samples we started the compiled binary of a candidate and measured the time it took to execute the binary and thereby to hash 16 GB of data. Additionally, four seconds after the binary was started, we started to measure the power consumption. We sampled again the current and the voltage. With help of the values we calculated the power consumption of the system. We chose the delay of four seconds to avoid that values from the idle mode are counted for the hashing mode and therefore distort the measurement.

During the measurement every unnecessary service was stopped. Thereby, we limit the power consumptions of other processes and reduce possible side effects and sources of interference.

As soon as the binary had been terminated, we stopped the power measurement. The power consumption of a SHA-3 candidate was computed by the difference of the two collected median values of the power consumption. We chose the median to lower the impact of short distortions, for example short irregular spikes caused by other processes. The difference of both values represents the power the systems requires to hash the data. Naturally, this includes also the additional power that is necessary to cool the system.

We repeated this measurement 10 times for every candidate and every version. With each run we randomized the order of the candidates to avoid positioning effects. Additionally, we waited 15 seconds between two measurements.

For the different versions we measured the time and the power needed by every candidate to hash the data.

3 Results

In this section we present our measurement results. In 3.1 we show the measurement results of the 256 bit versions. Afterwards, we introduce the measurement results with respect to the 512 bit versions.

For all measurements we used a Dell Studio 1537 notebook equipped with a P8400 Core 2 Duo CPU from Intel. 4 GB of RAM were installed at the notebook. The operating system was a 64-bit version of Ubuntu 9.04. The installed kernel

had the identifier `2.6.28-18-generic` and was the kernel shipped the in kernel package of the distribution.

During the whole measurement the CPU was set to a frequency of 2.27 GHz.

3.1 Results of the 256 Bit Versions

In addition to the official round 2 candidates, the Supercop versions of MD5, SHA-1 and SHA-256 were included in the measurements of the 256 bit versions. As some candidates in Supercop produce only a single executable for 256 bit and 512 bit, we used the same version for both measurements. To this end, we used the same binary for 256 bit and 512 bit versions for each of Hamsi, CubeHash, Keccak, Fugue, and Shabal.

The results of the time measurements of the 256 bit versions are depicted in Figure 2(a). The figure shows a clear difference among the candidates regarding the time that is needed to hash the 16 GB of data. The solid horizontal line represents the time SHA-256 needs to hash the data. While Blake, and Blue Midnight Wish are even faster than SHA-1, the candidates Echo, Fugue, Groestl, Hamsi, JH and Shavite are slower than SHA-1 and also slower than SHA-256.

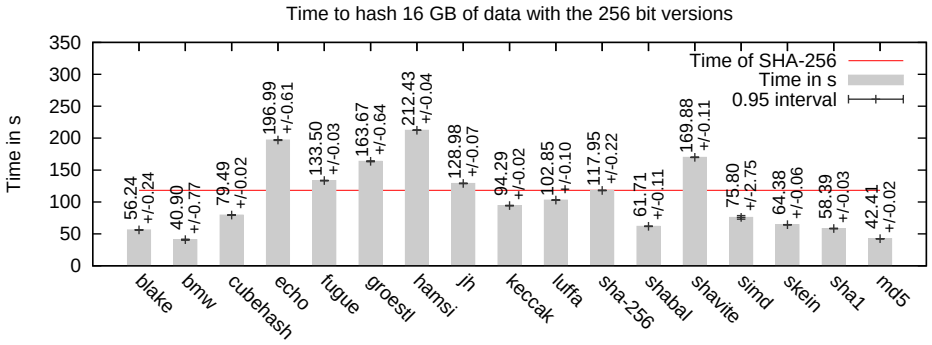
Figure 2(b) depicts the power required by each candidate to hash data. Please note, that the x-axis starts at 10 W. Even though the differences are not as significant as the time differences, there are clear differences among the candidates. MD5 is the hash function with the lowest power consumption. The SHA-3 candidate with the lowest power consumption is most probably Skein, at least regarding our system and the 256 bit versions. It is interesting that the used power does not correlate with the used time. In fact the rank correlation coefficient of time and power is 0.37. A value near to -1 means that a higher power consumption correlates with a shorter runtime of the binary, while a correlation of 1 means that the power consumption comes hand in hand with a longer time to hash the data. However, in our case the value indicates that there is no linear correlation between the power consumption and the execution time.

With the help of the power and the time used to hash 16 GB of data, we can compute the energy that is necessary to hash a single byte. The value can be computed by:

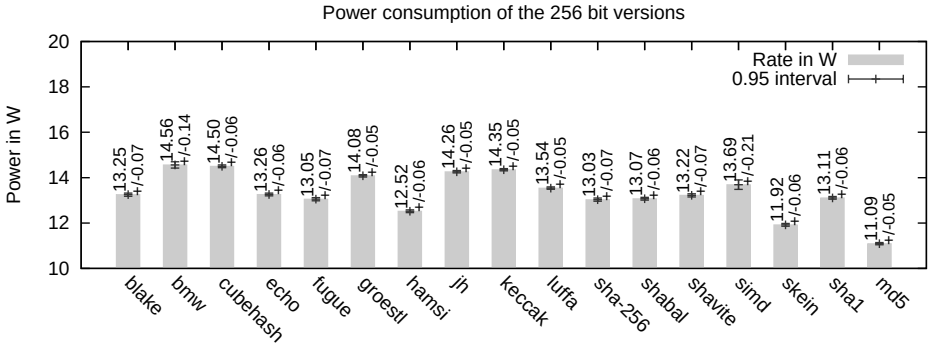
$$energy\ per\ byte[J/Byte] = \frac{power\ [J/s] \cdot time\ [s]}{data\ volume\ [Byte]}$$

The resulting energy for hashing a single byte (*energy per byte*) is depicted in Figure 2(c). Due to the calculation of the values and the huge differences in time compared to the power consumption, there is a strong similarity to the time plot shown in Figure 2(a).

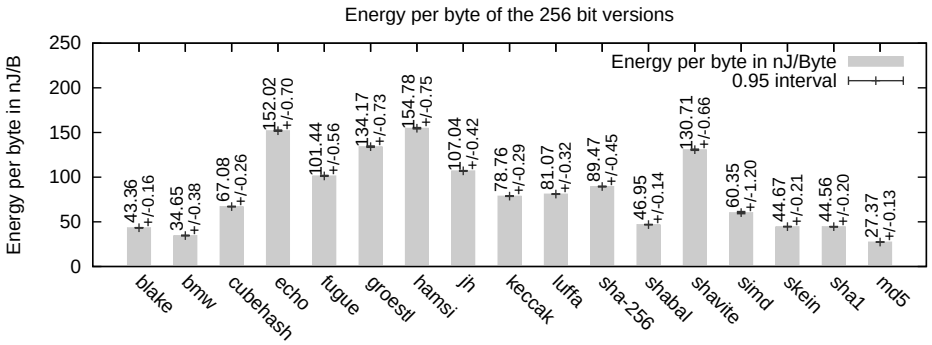
However, it is worth to note that the power consumption of the different candidates is not totally negligible. For example, while Cubehash is on rank eight regarding time, it is on the 16th rank with respect to the power consumption. At least if we only consider the expected values of the candidates (see Table 1). This indicates that also the power consumption of a candidate is a meaningful criterion regarding the performance of the hash algorithms, especially if the



(a) Time in s to hash 16 GB of data



(b) Consumed power in W while hashing data



(c) Energy per Byte in nJ/B

Fig. 2. The results of the measurements of the 256 bit versions

Power

Rank	Name	Rate [W]
1	md5	11.09
2	skein	11.92
3	hamsi	12.52
4	sha256	13.03
5	fugue	13.05
6	shabal	13.07
7	sha1	13.11
8	shavite	13.22
9	blake	13.25
10	echo	13.26
11	luffa	13.54
12	simd	13.69
13	groestl	14.08
14	jh	14.26
15	keccakc	14.35
16	cubehash	14.50
17	bmw	14.56

Time

Rank	Name	Time [s]
1	bmw	40.90
2	md5	42.41
3	blake	56.24
4	sha1	58.39
5	shabal	61.71
6	skein	64.38
7	simd	75.80
8	cubehash	79.49
9	keccakc	94.29
10	luffa	102.85
11	sha256	117.95
12	jh	128.98
13	fugue	133.50
14	groestl	163.67
15	shavite	169.88
16	echo	196.99
17	hamsi	212.43

Energy

Rank	Name	Energy [nJ/B]
1	md5	27.37
2	bmw	34.65
3	blake	43.36
4	sha1	44.56
5	skein	44.67
6	shabal	46.95
7	simd	60.35
8	cubehash	67.08
9	keccakc	78.76
10	luffa	81.07
11	sha256	89.47
12	fugue	101.44
13	jh	107.04
14	shavite	130.71
15	groestl	134.17
16	echo	152.02
17	hamsi	154.78

Table 1. The measured parameters of the 256 bit candidates.

needed time to hash the data is similar. Since the criterion *energy per byte* depends on both, it could be a suited parameter to distinguish among the different candidates.

By splitting the candidates into two groups, those which perform better than SHA-256 and those which perform worse than SHA-256, we end up with the same groups, regardless if we split with respect to the time or to the energy per byte.

3.2 Results of the 512 Bit Versions

In this part we present the results of the 512 bit versions. Figure 3(a) shows the time that is needed by each candidate to hash 16 GB of data. As in Figure 2(a) the solid line marks the time SHA-512 needs to hash the same amount of data. We used the upper confidence interval of SHA-512 as reference value. The candidates Echo, Fugue, Groestl, Hamsi, JH, Luffa, and Shavite need more time as SHA-512. Especially, Echo, Groestl, Hamsi and Shavite take a lot more time than all other candidates in the compiled version of Supercop. For example, Shavite needs around 10 times longer to hash the same amount of data than the fastest 512 bit candidate. It's also worth to mentioned that the 512 bit version of Blue Midnight Wish is more than twice as fast as Shabal which is the third fastest candidate regarding the 512 bit candidates. Even with respect to Skein, the second fastest candidate in this measurement, it is more than a factor of 1.8.

Figure 3(b) presents the power consumption of the different 512 bit candidates. Blue Midnight Wish is the candidates with the highest power consumption, but it is the fastest regarding the time. It requires $1.04W$ (7%) more power

for its computation than the average candidate. This might be an indicator that Blue Midnight Wish is already highly optimized.

However, in general there is no linear correlation between the power consumption and the time a candidate needs to hash the data, as the rank correlation coefficient is only 0.14 in the case of the 512 bit versions.

Figure 3(c) shows the energy that is necessary to hash a single byte for the 512 bit candidates. The differences between the different candidates are significant, mostly due to the significant differences in the time that is needed to hash the data. Most notable is the fact that the 512 bit version of Blue Midnight Wish is likely to consume less energy to hash a byte than MD5. While MD5 requires $27.37 \pm 0.13 \frac{nJ}{B}$, Blue Midnight Wish in the 512 bit version needs $24.31 \pm 0.12 \frac{nJ}{B}$.

Power

Rank	Name	Rate [W]
1	hamsi	12.51
2	shavite	12.93
3	fugue	13.06
4	shabal	13.06
5	skein	13.55
6	echo	13.61
7	blake	13.90
8	luffa	13.93
9	sha512	14.08
10	jh	14.25
11	keccakc	14.33
12	groestl	14.35
13	simd	14.38
14	cubehash	14.47
15	bmw	14.86

Time

Rank	Name	Time [s]
1	bmw	28.10
2	skein	50.00
3	shabal	61.77
4	blake	76.95
5	cubehash	79.48
6	simd	80.61
7	keccakc	94.30
8	sha512	96.73
9	jh	128.63
10	fugue	133.45
11	luffa	151.16
12	hamsi	212.37
13	groestl	229.84
14	echo	261.12
15	shavite	278.71

Energy

Rank	Name	Energy [nJ/B]
1	bmw	24.31
2	skein	39.43
3	shabal	46.96
4	blake	62.27
5	cubehash	66.96
6	simd	67.46
7	keccakc	78.64
8	sha512	79.29
9	fugue	101.44
10	jh	106.68
11	luffa	122.55
12	hamsi	154.65
13	groestl	192.03
14	echo	206.88
15	shavite	209.63

Table 2. The measured parameters of the 512 bit candidates.

3.3 Comparison Between the 256 Bit and the 512 Bit Versions.

In the Figure 4 we compare the 256 bit versions with their 512 bit pendants. As mentioned above, we used for some candidates the same version for both measurements, namely CubeHash, Fugue, Hamsi, Keccak and Shabal.

In the comparison we can see that all these five candidates have almost the same expected energy consumption: there is no significant difference according to a two-sided t-test with a 0.95 confidence interval. This provides some evidence that our methodology result in reproducible and valid results. The candidate JH does not differ in the expected value either, even though different binaries were used. All other candidates ended up in different expected values according to a t-test. The most significant difference can be observed with respect to Skein.

4 Related Works

There are only a few publications that have reported the energy consumption of hash algorithms. In [6] the authors analyzed among others the energy consumption of MD2-MD4,SHA, and SHA-1, and determined the energy that is consumed to hash a single byte. Contrary to our measurements, they used a PDA with an ARM processor as test system. The energy per byte in their setup resulted in $760nJ/B$ for SHA-1 and $590nJ/B$ for MD5. Thus, their results differ by a factor of roughly 17 with respect to SHA-1 and a factor 21 with respect to MD5. The reasons for the huge difference are probably the CPU architecture as well as the size of the circuits of the used CPU.

In [1] the authors present a ultra-low power SHA-1 hardware design. The analysis of their implementation showed that their design consumes $1.49nJ/B$ for SHA-1. This is a factor of 30 lower than our results for SHA-1. Obviously, a specific hardware design could be much more energy efficient than the versions we measured.

5 Discussion

There are some caveats to be aware of when interpreting the results. The most important restriction is that the results are only valid with respect to the used CPU. A test with different CPUs, especially with a complete different architecture are likely to result in significantly different values as it is shown in Section 4. While the energy per byte on the ARM processor used in [6] for SHA-1 is 17 times higher as on our system, the hardware implementation in [1] consumes 30 times less energy per byte than our used SHA-1 version.

With respect to our measurements, a source of interference might be the sensor that was used to measure the current and the voltage of our system. The sensor is integrated in the system and therefore somehow dependent on the system state. This could influence the accuracy of the chip itself. During our measurements we could observe a slight increase of the power consumption over time: the lower the battery energy the higher was the power consumption of the system. With respect to the overall consumption the decrease is negligible, especially since we determined the idle power consumption prior to each run of a candidate. Additionally, we tried to minimize the effect by the randomizing the order of the candidates. Another problem might be that the sensor is not very precise and thus provides a constantly to high or to low value.

As mentioned above, the power consumption includes the power the fan of the notebook consumes during the measurements. Naturally, the fan was more frequently used during the hashing operations. Therefore, the measured results are most likely a upper bound for our test system.

6 Conclusion

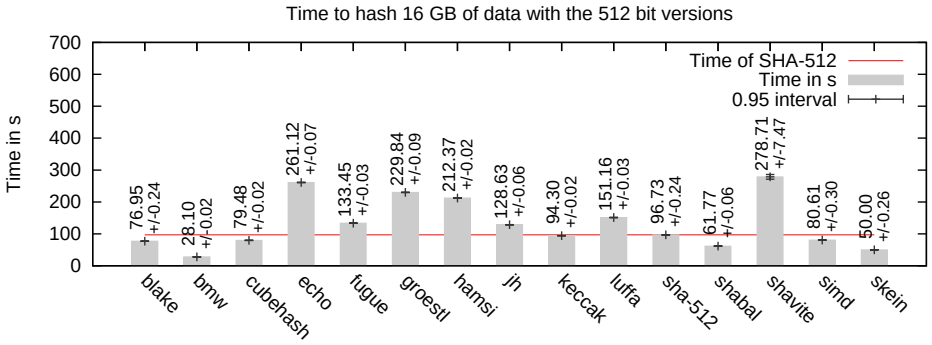
In this paper we provided an evaluation of the power consumption for the SHA-3 candidates, and we have shown that the different hash candidates differ not only

with respect to the needed time to hash data, but also with respect to the power consumption. Both the power consumption and the time are therefore important criteria for the performance of the candidates. The *energy per byte* considers both criteria. Up to now, the time differences among the candidates dominate the overall picture. However, due to future optimization and the selection process it is likely that the time differences between the candidates become less and therefore the criterion *energy per byte* may become an important parameter for the selection process.

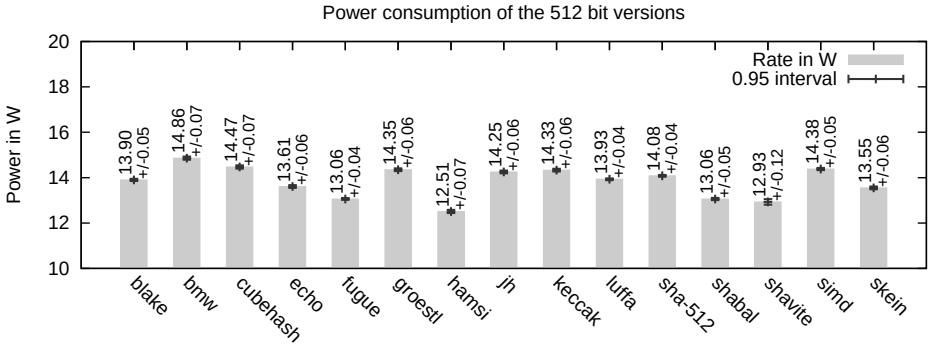
References

1. Kaps, J.P., Sunar, B.: Energy comparison of aes and sha-1 for ubiquitous computing. In Zhou, X., Sokolsky, O., Yan, L., Jung, E.S., Shao, Z., Mu, Y., Lee, D.C., Kim, D., Jeong, Y.S., Xu, C.Z., eds.: EUC Workshops. Volume 4097 of Lecture Notes in Computer Science., Springer (2006) 372–381
2. Fettweis, G., Zimmermann, E.: Ict energy consumption - trends and challenges. In: Proceedings of the 11th International Symposium on Wireless Personal Multimedia Communications. (2008)
3. Ruth, S.: Green it more than a three percent solution? IEEE Internet Computing **13**(4) (2009) 74–78
4. National Institute of Standards and Technology: Cryptographic hash algorithm competition. http://csrc.nist.gov/groups/ST/hash/sha-3/index.html visited 09.05.2010.
5. VAMPIRE lab: Supercop: System for unified performance evaluation related to cryptographic operations and primitives. http://bench.cr.yp.to/supercop.html visited 09.05.2010.
6. Potlapally, N.R., Ravi, S., Raghunathan, A., Jha, N.K.: Analyzing the energy consumption of security protocols. In: ISLPED '03: Proceedings of the 2003 international symposium on Low power electronics and design, New York, NY, USA, ACM (2003) 30–35

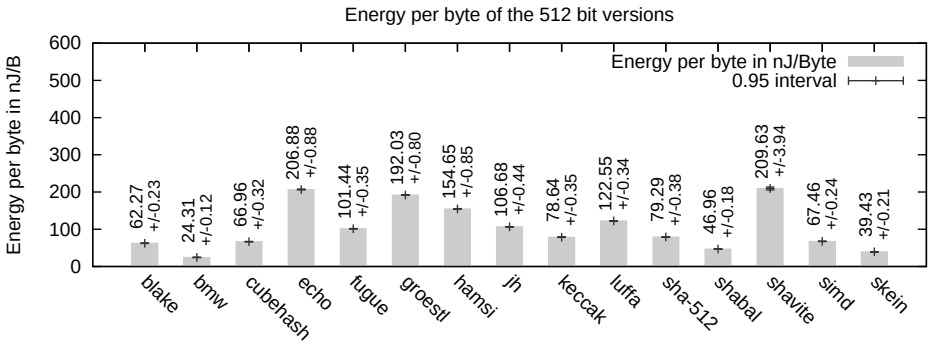
A Appendix: Additional Diagrams and Tables



(a) Time in s to hash 16 GB of data

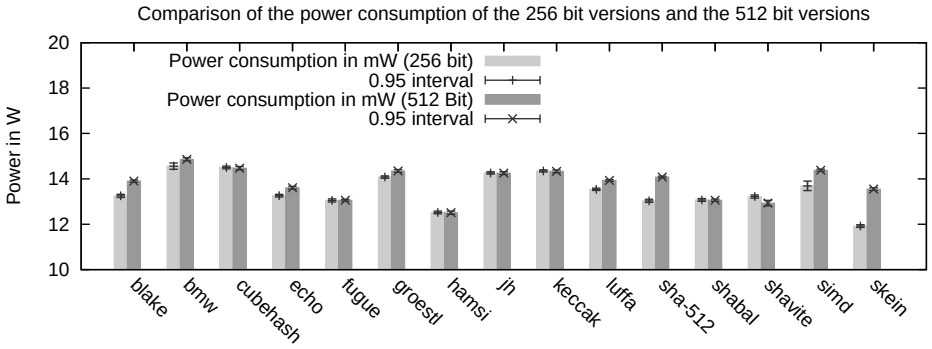


(b) Consumed power in W while hashing data

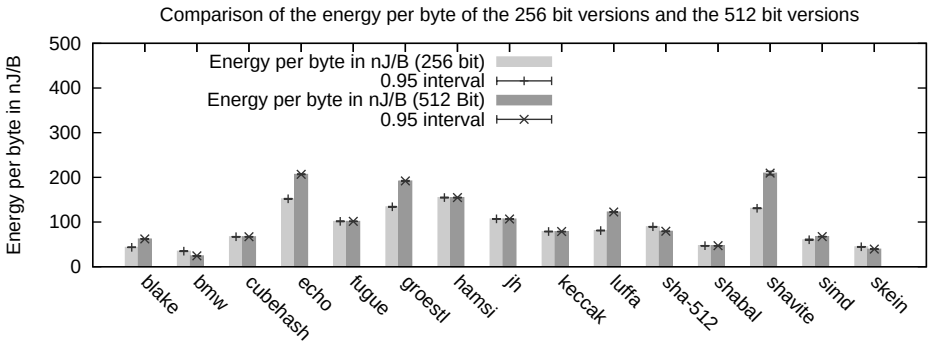


(c) Energy per Byte in nJ/B

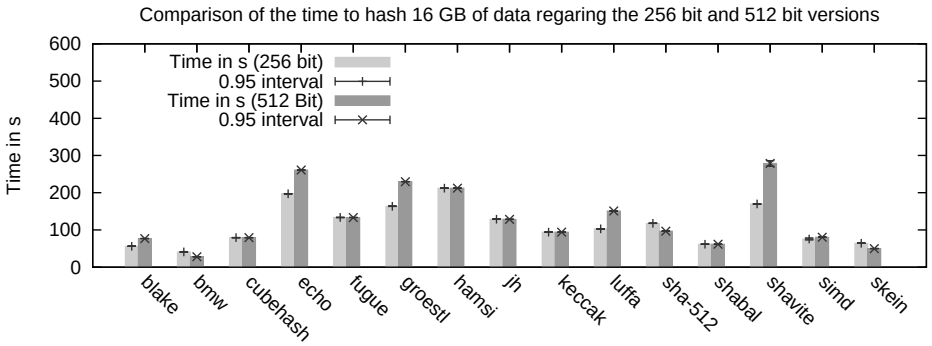
Fig. 3. The results of the measurements of the 512 bit versions



(a) Comparison of the power consumption.



(b) A comparison of the energy per byte.



(c) A comparison of the time to hash 16 GB of data

Fig. 4. Comparison of the 256 bit versions and the 512 bit version