The 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications
15-17 September 2011, Prague, Czech Republic

Performance Evaluation of Symmetric Cryptography in Embedded Systems

Ondrej Hyncica, Pavel Kucera, Petr Honzik, Petr Fiedler
Department of Control and Instrumentation, Faculty of Electrical Engineering and Communication
Brno University of Technology, Kolejni 4, 612 00, Brno, hyncica@feec.vutbr.cz

Abstract—**We consider the problem of implementing security algorithms into embedded systems deployed in automation applications. Such systems are typically built on embedded microcontrollers with limited resources and as hardware changes may not be possible or convenient, the software based cryptography is a suitable solution. In this paper we present results of performance benchmarks of different software-implemented symmetric cryptography algorithms on 8 and 16-bit microcontroller platforms. The contribution of the work is in comparing performance of different algorithms, embedded microcontroller platforms, effects of optimizations and different implementations.** [1]

Keywords—symmetric cryptography; block ciphers; embedded; microcontroller; benchmark

I. INTRODUCTION

There are many automation applications deployed that lack security. This is common for many industrial embedded systems (smart instrumentation, embedded controllers, communication devices). As the security requirements are becoming a more and more important parameter, many deployments are being revised and cryptographic functions are being worked into these systems. Adding security to such systems is possible by additional or specialized hardware or by implementing the cryptographic algorithms in software. Therefore there is a need for selection of a suitable cryptographic algorithm that can be implemented with sufficient performance in given embedded microcontrollers.

In this work we evaluate the suitability of several 8 and 16-bit microcontrollers for software implementation of symmetric cryptography algorithms. Three popular microcontrollers Freescale HCS08, Atmel ATmega128 and Texas Instruments MSP430 are evaluated in terms of memory requirements and encryption/decryption speed of symmetric block ciphers. The experimental results presented here show the performance of the selected algorithms on different platforms and identify the parameters important for selection of a suitable algorithm for the given embedded platforms.

Previous works investigating performance of symmetric block cipher algorithms [1]-[3] focus on optimizations and specialized implementations for embedded platforms [2]. Large number of the presented benchmarks is concerned with selecting a suitable algorithm for wireless sensor networks [1], [3]. The authors in [1] present a very detailed benchmark of six different symmetric block cipher algorithms on a single microcontroller platform (TI MSP430 microcontroller). In [2] the authors compare implementation of several ciphers on the Atmel AVR ATmega128L microcontroller and present their modifications to the implementation of the Rijndael (AES) algorithm resulting in significantly improved performance for the selected platform. More embedded platforms (three small microcontrollers and two 32-bit processors) are compared for five different cryptographic algorithms, two of them block ciphers in [3]. Comparison of asymmetric algorithms (RSA and ECC) running on two different 8-bit microcontrollers is presented in [4]. Surveys and benchmarks on more powerful embedded platforms such as DSPs are more common, as example can be taken [5].

The presented work compares performance and resource requirements of fifteen symmetric block ciphers on three different embedded microcontroller platforms. The implementation of the algorithms is taken from freely available open source libraries of cryptographic primitives. The evaluated parameters are the generated code-size and the speed of encryption/decryption for the algorithms and microcontroller platforms.

The chapter II describes the problem of software-implemented cryptography of embedded systems and presents the evaluated algorithms and embedded microcontroller platforms. Chapter III introduces the method of evaluation, chapter IV shows the experimental results and chapter V concludes the paper.

II. SOFTWARE BASED CRYPTOGRAPHY IN EMBEDDED SYSTEMS

Systems with no available hardware acceleration of cryptographic functions have to rely on software-implemented cryptography. Software based cryptography is very resource consuming, particularly hash functions

[1] This work has been supported by the research project 2C08002 (Research of Universal and Complex Authentication and Authorization for Permanent and Mobile Computer Networks – KAAPS), by the Ministry of Education of the Czech Republic under Project 1M0567 (Center for Applied Cybernetics – CAK) and by Brno University of Technology.

978-1-4577-1425-2/11/$26.00 ©2011 IEEE

and asymmetric cryptography need significantly more resources than symmetric block ciphers. The symmetric block cipher algorithms are considered here as they are not so resource consuming and are available in open source libraries that can be ported for the embedded microcontroller platforms. There are open source libraries of cryptographic primitives specialized for 8-bit microcontrollers, but are not comprehensive and usually are written for a specific platform in assembly language. The implementation of the cipher algorithm may have a great impact on performance and security of the system.

Proprietary, specific and custom solutions to cryptography are in general not a good way to go and therefore open source libraries seem to be a reasonable option. There are several such libraries available, mostly targeted to 32-bit architectures. Another option can be specialized implementations for specific platforms that are available commercially. As a suitable solution for embedded implementations seem libraries OpenSSL, TomCrypt or GCrypt. These libraries are extensive and support many cryptographic functions. Other implementations such as Crypto++ or Botan turned out to be problematic to port. For the detailed evaluation on all of the target platforms the TomCrypt (LTC) in version 1.16 was selected, because of easy porting to embedded platforms, as it is written in plain C language and has low memory requirements.

A. Cipher Algorithms

The cipher algorithms included in the tests are 15 different symmetric block ciphers in the ECB mode. The algorithms range from general purpose ciphers to ciphers ready for embedded implementation. The list of selected algorithms is: AES, XTEA, RC5, Skipjack, SAFER, DES, Anubis, Twofish, CAST5, Kasumi, Kseed, Noekeon, RC2, RC6, Blowfish.

TABLE I. SYMMETRIC BLOCK CIPHERS IN THE EVALUATION

Algorithm	Key length	Block size
AES	128, 192, 256 bit	128 bit
XTEA	128 bit	64 bit
RC5	128 bit	64 bit
Skipjack	80 bit	64 bit
SAFER	64, 128 bit	64 bit
DES	64 bit	64 bit
Anubis	128 – 320 bit	128 bit
Twofish	128, 192, 256 bit	128 bit
CAST5	40, 80, 128 bit	64 bit
Kasumi	128 bit	64 bit
Kseed	128 bit	128 bit
Noekeon	128 bit	128 bit
RC2	64, 128 bit	64 bit
RC6	128, 192, 256 bit	128 bit
Blowfish	64 bit	64 bit

The key-length for all the ciphers was chosen to be 128 bits when applicable, otherwise the default key length was used. The ciphers along with their parameters are summarized in TABLE I.

B. Microcontroller Platforms

The performance evaluation in the survey is executed for the 8-bit Freescale HCS08, 8-bit Atmel ATmega128 and 16-bit Texas Instruments MSP430 microcontrollers. These platforms are commonly used in many application areas where security can be an issue.

1) 68HCS08 The 68HCS08 is a general purpose 8-bit CISC microcontroller by Freescale with multiply and division instructions. The type used in this evaluation is MC9S08GT60 [6], capable of running up to 20 MHz bus clock (40 MHz CPU clock). The chip has 64 kB of internal flash and 4 kB of internal RAM available. The results presented here were obtained with CodeWarrior 5.0 development environment and compiler.

2) ATmega128 The ATmega128 is an 8-bit RISC microcontroller by Atmel with a two-cycle multiplication instruction. The device used in this evaluation is ATmega1281 [7]. It is capable of running at frequencies up to 16 MHz CPU clock and has 128 kB of internal flash and 8 kB of internal RAM. For compilation the AVRStudio 4.16 with compiler avr-gcc 4.1.2 was used.

3) MSP430 MSP430 is a 16-bit RISC microcontroller developed by Texas Instruments, with hardware multiplication. The type of device used in this evaluation is MSP430F1612 [8] capable of running at frequencies up to 8 MHz system clock. It has 55 kB of internal flash, 5 kB of internal RAM. As compiler the IAR Embedded Workbench 4.0 and IAR C/C++ V3.42A/Win32 was used.

III. EXPERIMENTAL METHOD

Several criteria for evaluating cryptographic algorithms are presented in [9], in this test the compactness and performance criteria from [9] are evaluated. The compactness criteria describe the memory requirements of the implementation, the performance criteria is expressed as the throughput in encrypted bits per second. The generated code size, the key-setup time and the encryption/decryption throughput were evaluated. The compiler optimizations were considered as a test parameter.

For the encryption/decryption only the ECB (electronic code book) mode was used, as it does not bring additional overhead to the algorithm itself. Comparison of performance and properties of different block modes can be found in other works [1]. For all of the platforms the maximal available system clock frequency was used. This may not be optimal in terms of energy consumption, but gives the upper bound for performance of the individual platforms.

A. Generated Code Size

The code size is taken as the size of the cipher implementation reported by a linker. Because the library code includes necessary auxiliary functions that differ for various ciphers, the comparison is not fully fair due to varying auxiliary code overhead. Also the runtime data and stack RAM requirements are not considered here.

B. Performance

The speed of the ciphers is obtained as the time it takes to complete the given operation. There are several options how to measure the execution time. The method chosen here is based on measuring the time directly using an internal timer and subtracting the additional interrupt overhead. This way is easier than counting cycles, as one basic test code can be used for different platforms. The accuracy of the method was compared to cycle counting and external time measurement. The results are shown and compared for maximal frequency the given platform can operate at.

IV. EXPERIMENTAL RESULTS

The experiments measuring the encryption throughput, key-setup time and size of the generated code were carried out for the three platforms and 15 symmetric block cipher algorithms. The results presented here were obtained with the LTC implementation.

A. Encryption Throughput

The experimental results of the performance evaluation are shown in TABLE II. The results are ranked from fastest algorithm to the slowest. The table also shows the time needed for key setup. The results are listed for the highest system clock available for each of the platforms, when no compiler optimization is used.

TABLE II. CIPHERS RANKED BY ENCRYPTION SPEED

Algorithm	Platform	Encryption [kbit/s]	Key-setup [ms]
AES	MSP430	290.24	0,6
Twofish	ATmega	235.13	271,9
SAFER	MSP430	232.81	1,7
AES	ATmega	223.32	49,0
Twofish	MSP430	212.15	15,0
RC2	MSP430	161.12	1,2
SAFER	ATmega	157.91	1,8
Noekeon	MSP430	138.78	0,1
RC2	ATmega	138.40	1,4
DES	MSP430	123.03	16,6
RC5	MSP430	121.44	3,3
Kasumi	MSP430	119.93	0,3
XTEA	MSP430	112.14	0,4
XTEA	ATmega	108.01	0,6
CAST5	MSP430	106.95	0,8
Kasumi	ATmega	93.99	0,3
Skipjack	MSP430	92.04	0,0
KSEED	MSP430	91.90	0,8
RC6	MSP430	87.51	5,8
DES	ATmega	87.06	26,2
SAFER	HCS08	85.48	4,1
Skipjack	ATmega	82.39	0,0
Noekeon	ATmega	71.33	0,1
KSEED	ATmega	63.04	0,9
Blowfish	MSP430	56.65	45,0
Skipjack	HCS08	35.28	0,1
AES	HCS08	26.22	6,7
RC2	HCS08	22.56	4,9
Kasumi	HCS08	14.76	1,9
KSEED	HCS08	8.95	10,4
DES	HCS08	8.27	222,9
XTEA	HCS08	7.76	5,8
Noekeon	HCS08	4.32	2,5

B. Memory Requirements

In the TABLE III. are listed the generated code sizes for the algorithms and platforms. The algorithms were ported and compiled for all the platforms, however the ciphers AES, DES, CAST5, SAFER were able to compile only with an option to generate smaller code (not using memory tables). All results are displayed without compiler optimizations. The utilization in percent listed in the table is defined as the ratio of the code size and flash memory size of the individual platform.

TABLE III. CIPHERS RANKED BY CODE SIZE

Algorithm	Platform	Code size [kB]	Utilization [%]
Skipjack	MSP430	1,9	3,5
RC2	MSP430	2,0	3,6
Skipjack	HCS08	2,1	3,3
XTEA	MSP430	2,5	4,5
XTEA	HCS08	2,8	4,4
RC2	HCS08	2,8	4,4
RC5	MSP430	3,1	5,6
SAFER	MSP430	3,1	5,6
Skipjack	ATmega	3,2	2,5
Kasumi	MSP430	3,4	6,2
SAFER	HCS08	3,8	5,9
RC2	ATmega	3,8	3,0
Kasumi	HCS08	4,3	6,7
Noekeon	MSP430	4,5	8,2
Noekeon	HCS08	4,6	7,2
XTEA	ATmega	5,1	4,0
RC6	MSP430	5,3	9,6
Kasumi	ATmega	5,4	4,2
SAFER	ATmega	5,7	4,5
DES	HCS08	7,2	11,3
Blowfish	MSP430	7,3	13,3
Noekeon	ATmega	7,5	5,9
KSEED	MSP430	8,4	15,3
KSEED	HCS08	8,7	13,6
DES	ATmega	10,8	8,4
Twofish	MSP430	11,3	20,5
AES	MSP430	12,4	22,5

Algorithm	Platform	Code size [kB]	Utilization [%]
KSEED	ATmega	13,7	10,7
CAST5	MSP430	15,1	27,5
Twofish	ATmega	16,4	12,8
AES	ATmega	31,4	24,5
DES	MSP430	39,1	71,1
AES	HCS08	44,9	70,2

To capture the effect of optimizations the results are presented for none/speed/size compiler optimizations in the TABLE IV. For all of the platforms the maximal available system clock frequency was used. This is not the best option in terms of energy consumption, but gives the upper bound for performance of the individual platforms.

TABLE IV. COMPILER SPEED OPTIMIZATION EFFECT ON KEY SETUP TIME AND ENCRYPTION THROUGHPUT OF THE CIPHERS.

Platform	Algorithm	Key Setup Time [ms]		Encryption Speed [kbps]	
		Opt. none	Opt. speed	Opt. none	Opt. speed
ATmega	AES	49,031	0,591	223,32	279,02
ATmega	SAFERSK	1,783	0,749	157,91	653,06
ATmega	RC2	1,406	0,455	138,40	365,32
ATmega	XTEA	0,594	0,330	108,01	179,81
ATmega	Kasumi	0,341	0,191	93,99	306,13
ATmega	DES	26,188	7,219	87,06	116,56
ATmega	Skipjack	0,036	0,007	82,39	277,13
ATmega	Noekeon	0,071	0,048	71,33	102,55
MSP430	AES	0,611	0,491	290,24	371,49
MSP430	SAFER	1,655	1,208	232,81	446,83
MSP430	Twofish	14,995	10,108	212,15	291,96
MSP430	RC2	1,217	0,647	161,12	312,03
MSP430	Noekeon	0,086	0,048	138,78	182,87
MSP430	DES	16,578	13,360	123,03	188,44
MSP430	RC5	3,266	2,815	121,44	126,70
MSP430	Kasumi	0,327	0,200	119,93	190,82
MSP430	XTEA	0,427	0,247	112,14	170,79
MSP430	CAST5	0,775	0,618	106,95	128,66
MSP430	Skipjack	0,032	0,019	92,04	125,14
MSP430	KSEED	0,770	0,526	91,90	101,32
MSP430	RC6	5,811	4,643	87,51	95,56
MSP430	Blowfish	45,006	15,410	56,65	58,59
LPC2378	RC5	0,107	0,045	91718,06	125727,68
LPC2378	RC6	0,186	0,080	74558,32	105464,77
LPC2378	Noekeon	0,004	0,003	53507,36	99973,81
LPC2378	XTEA	0,022	0,012	47516,09	69433,31
LPC2378	Blowfish	8,554	4,524	43272,00	81051,89
LPC2378	Twofish	1,572	0,799	36819,32	68356,09
LPC2378	RC2	0,142	0,083	26847,24	41774,81
LPC2378	AES	0,070	0,039	24381,06	43805,69
LPC2378	CAST5	0,111	0,052	22838,45	47813,04
LPC2378	Anubis	0,544	0,256	16583,41	31430,94
LPC2378	SAFER	0,190	0,117	15440,41	21425,37
LPC2378	DES	1,694	1,064	13614,35	23025,88
LPC2378	Kasumi	0,031	0,012	11796,85	26402,27
LPC2378	KSEED	0,084	0,029	7217,42	25098,85
LPC2378	Skipjack	0,003	0,002	6172,97	10268,80

To show the effects of compiler optimizations the benchmarks were performed for code compiled without optimizations and with optimizations for execution speed. The results of key-setup time and encryption throughput benchmarks are summarized in the TABLE IV. The results are grouped for each of the platform and ranked by encryption speed (without optimizations). Results for code compiled without optimizations are in columns titled “Opt. none”, results for speed optimized code are titled “Opt. speed”.

C. *Time to Encrypt Data*

Next parameter, which can be derived from the test described above, is the time it takes to encrypt a given amount of data, including cipher initialization and key setup time. From this parameter can be seen if the implementation will satisfy given timing requirements. The results are shown in graph in Figure 1 where the time it takes to setup and encrypt a block of data using AES for the individual microcontroller platforms is plotted in log scale. The results are given for speed optimized code. The difference between the ATmega and MSP430 platforms is relatively small for the speed optimized code. The 32-bit LPC2378 is one order of magnitude faster.

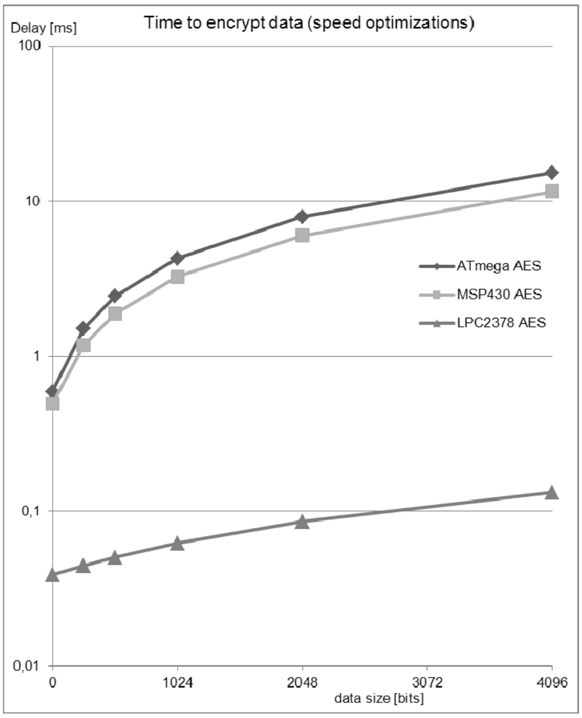

Figure 1. Time for key-setup and encryption of a block of data

V. RESULTS DISCUSSION

The results of performance evaluation show very different results depending on individual algorithm and the platform. The highest throughput ciphers are the AES, Twofish and SAFER, with a significant margin compared to the other evaluated ciphers. The encryption throughput is shown in Figure 2.

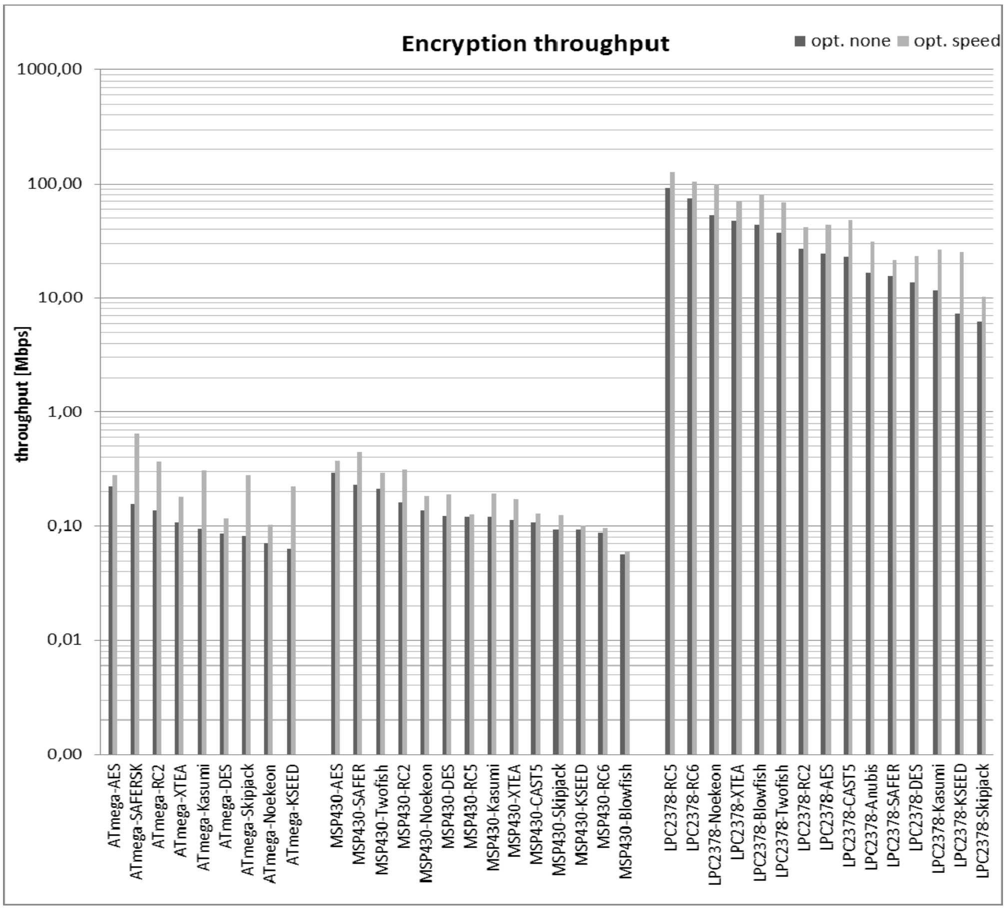

Figure 2. Encryption throughput for the evaluated ciphers and platforms

Most of the ciphers require less than 15 % of the available flash memory of the individual platforms. But the implementations utilizing memory tables require usually more than 25 % of the available flash memory and in several cases more than 70%. The generated code size is comparable for all of the evaluated platforms. There were porting problems and some algorithms did not compile correctly or did not work properly, these were excluded from the results. When comparing the performance results, it should be understood, that the LTC is a general purpose library and is much slower than specialized implementations. For example the throughput of AES on the ATmega microcontroller as reported in [2] is more than 20 times higher than the LTC implementation shown here.

VI. CONCLUSION

Following from the experimental results it can be concluded that although the evaluated open source libraries of cryptographic functions are suitable mostly for 32-bit platforms, it is possible to use these on the 8/16-bit platforms as well. There were porting issues and in some cases the code was generated, but was not executed correctly or the runtime requirements on data memory and stack size were too high. These problems arose particularly for the 8-bit microcontroller platforms (HCS08, ATmega). The results also show the limitations of general purpose implementations, as these produce larger and significantly slower code than special implementations. The selection of general purpose cryptographic libraries is a simple way of delivering security into embedded systems, however it is not very efficient and specialized implementations should be considered when performance requirements or resource limitations are tight.

The results in this paper are presented for maximal CPU frequency for each platform, but this is usually not the best energy efficient option. Therefore an evaluation of performance versus the energy consumption is being prepared as a future work.

REFERENCES

[1] Y. W. Law, J. Doumen and P. Hartel, “Survey and benchmark of block ciphers for wireless sensor networks,” *ACM Trans. on Sensor Networks, vol.* 2, pp 65-93, February 2006.

[2] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, L. Uhsadel, “A Survey of Lightweight-Cryptography Implementations,” *IEEE Design and Test of Computers*, vol. 24, no. 6, pp. 522-533, November 2007, ISSN 0740-7475.

[3] P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller and M. Sichitiu, “Analyzing and modeling encryption overhead for sensor network nodes,” *Proc. of 2nd ACM international Conference on Wireless Sensor Networks and Applications*, pp. 151-159, September 2003.

[4] N. Gura, A. Patel, A. Sander, H. Berle and S.C. Shantz, “Comparing elliptic curve cryptography and rsa on 8-bit cpus,” *Cryptographic Hardware and Embedded Systems*, pp. 119–132, 2004, ISBN: 978-3-540-22666-6.

[5] T. Wollinger, J. Guajardo and C. Paar, “Cryptography in Embedded Systems: An Overview,” *Proceedings of the Embedded World 2003 Exhibition and Conference*, pp. 735-744, Feb. 2003.

[6] Freescale: HCS08 Reference manual. Available online: http://www.freescale.com/files/microcontrollers/doc/ref_manual/HCS08RMV1.pdf

[7] Atmel: AVR ATmega128 datasheet. Available online: http://www.atmel.com/dyn/resources/prod_documents/doc2467.pdf

[8] Texas Instruments: MSP430F1612 Datasheet. Available online: http://www.ti.com/lit/gpn/msp430f1612

[9] L.Batina, S. Kumar, J. Lano, K. Lemke, N. Mentens, C. Paar, B. Preneel, K. Sakiyama and I. Verbauwhede, “Testing Framework for eSTREAM Profile II Candidates,” *ECRYPT Workshop on the State of the Art of Stream Ciphers*, February 2006.