Implementation of the SCREAM Tweakable Block Cipher in MSP430 Assembly Language

William Diehl

George Mason University, Fairfax VA 22033, USA
wdiehl@gmu.edu

Abstract. The encryption mode of the Tweakable Block Cipher (TBC) of the SCREAM Authenticated Cipher is implemented in the MSP430 microcontroller. Assembly language versions of the TBC are prepared using both precomputed tweak keys and tweak keys computed "on-the-fly." Both versions are compared against published results for the assembly language version of SCREAM on the ATMEL AVR microcontroller, and against the C reference implementation in terms of performance and size. The assembly language version using precomputed tweak keys achieves a speedup of 1.7 and memory savings of 9 percent over the reported SCREAM implementation in the ATMEL AVR. The assembly language version using tweak keys computed "on-the-fly" achieves a speedup of 1.6 over the ATMEL AVR version while reducing memory usage by 15 percent.

Keywords: Cryptography, encryption, MSP430, assembly, speed, efficiency

1 Introduction

Authenticated ciphers combine the functionality of confidentiality and integrity into one algorithm. In 2014 the Competition for Authenticated Encryption: Security, Applicability, and Robustness (CAESAR) called for submission of authenticated cipher candidates [1]. CAESAR candidates are evaluated in terms of security, size, robustness, flexibility, and performance. Software reference implementations written in C code are required as part of Rounds One and Two submissions. Many of the Round One submissions contained the authors' evaluations in both hardware and software. Additionally, several Round One submissions investigated the software performance of the authors' algorithms on various types of platforms, including high-end CPU and resource-constrained microcontrollers suitable for embedded applications.

SCREAM (Side-channel resistant authenticated encryption with masking) is one of the 29 CAESAR Round Two candidates. As part of the submission, the SCREAM authors conducted both hardware and software evaluations in several different configurations, and listed comparisons in area and performance of known cryptographic block ciphers, such as AES. As part of the software evaluation, the SCREAM authors conducted evaluations on high-end CPUs (Cortex A15, Atom, Core i7), and on an 8-bit microcontroller, the ATMEL AVR ATtiny 45 [2]. This research builds upon the SCREAM authors' observations by implementing SCREAM-10 (E) (i.e., SCREAM Tweakable Block Cipher (TBC) encryption mode consisting of ten steps) on the Texas Instruments TI MSP430 microcontroller. The MSP430 results are compared to the ATMEL AVR results in terms of size (ROM and RAM bytes) and performance (clock cycles per block and cycles per byte).

2 MSP430 Microcontroller

2.1 MSP430 Background

The TI MSP430F5529 Launch Pad Evaluation Kit was used in this research. This launch pad contains the 16-bit MSP430 CPU. The MSP430 can be clocked at speeds up to 25 MHz given the proper power configurations, however, the default setting of 8 MHz (corresponding to `PMMCoreVx = 0`) was used in this research. This configuration contains 128kB of Flash RAM and 8 kB of RAM. The MSP430, like the ATMEL AVR, uses a RISC instruction set. However, the MSP430 is a 16-bit processor vice the 8-bit ATMEL AVR. The MSP430 can address up to 1MB of memory, which requires a 20-bit address space and use of the MSP430X instruction set. However, only two MSP430X-specific instructions are used in this research, `calla` and `reta`.

The MSP430 uses twelve 16-bit general purpose registers (R4 – R15). Although the MSP430 is advertised as a RISC instruction set, it utilizes seven addressing modes, including register, indexed, symbolic, absolute, indirect register, indirect register auto-increment, and immediate. The latency for operations is variable, with register operations taking only a single cycle, and ALU operations in absolute mode taking up to six cycles. However, five-cycle absolute `mov.w` and six-cycle absolute `xor.w` operations can still be advantageous to overall performance, since the overhead of multiple load-and-store operations is avoided, as well as the overhead of corresponding `push` and `pop` commands [3].

2.2 Related Work

The MSP430 has been used extensively in cryptographic research, in particular for comparison of light-weight cipher implementations in embedded and wireless sensor networks. Law et al. compared the performances of five block ciphers in C implementations on the MSP430 [4]. Cazorla et al. constructed 12 lightweight and five conventional block cipher implementations in C using the MSP430 [5]. Wenzel-Brenner et al. compared 12 SHA-3 candidates on seven different CPU platforms using C implementations [6].

Other research has measured algorithm performance on the MSP430 using a combination of C and assembly language. For example, Düll et al. investigated performance of Elliptic-Curve Cryptography (ECC) using base code written in C but optimizing high-performance field multiplication in assembly language [7]. Burlow et al. extensively studied various implementations of block ciphers in both C and assembly language, and made performance comparisons between implementations [8]. Gouvêa and López produced high-speed implementations of several authenticated encryption modes of AES (such as Counter-with-CBC MAC Mode, Galois Counter Mode, Offset Codebook Mode, etc.) using the MSP430 with the CC430F6137 chip. The authors implemented these modes in C, with critical functions written in assembly. The authors also compared and contrasted performance with and without the MSP430's built-in AES instruction set [9].

Additionally, Schwabe, Yang, and Yang investigated pure assembly language implementations of SHA-3 candidates on the ARM11 processor. They noted that assembly language implementations of some candidates had a speedup of 2 over C implementations, and emphasized the importance of assembly language in performance evaluations of cryptographic algorithms [10].

3 Methodology

In support of this research several versions of SCREAM-10 (E) were implemented on the MSP430. The first version was a C implementation ported directly from the SCREAM Reference C implementation available at [11], and formatted to run on the MSP430. The SCREAM authors, in producing their ATMEL AVR implementation, attempted to follow insofar as possible the benchmarking methodology for hash functions outlined in [12]. This research is informed by the methodology used in [12], but deviates from this methodology as it measures block ciphers on the MSP430, not hash functions on the ATMEL AVR. However, the subsequent implementations of SCREAM-10 (E) in this research are written entirely in assembly language, and should provide a rational basis for comparison between ATMEL AVR and MSP430 performance.

The SCREAM authors, as discussed in [2], noted the degradation in performance in their own implementation resulting from computation of tweak keys "on-the-fly." Therefore, one assembly language implementation uses precomputed tweak keys, and another uses tweak keys computed on-the-fly, thus providing a basis for comparison in terms of size and performance. In contrast to [9], no comparison with the code using the AES instruction set is possible, since the SCREAM block cipher does not employ AES.

The two assembly language versions employ registers R8 – R15 as global variables containing the 128-bit status word, which eliminates the need to save and restore these registers during function calls. However, this reduces the amount of available registers to four (R4-R7), which necessitates some saving and restoring of registers, as well as use of RAM locations for memory-to-memory (i.e., absolute) operations. The MSP430F5529 contains a 32-bit by 32-bit hardware multiplier which was not used in these designs. Additionally, there is no multiplication operation in the instruction set. This code uses a look-up table to conduct the finite possible numbers of multiplications required for computation of the round constants.

This research is conducted in TI CCS Code Composer Studio. Like the TI IAR Embedded workbench environment, CCS allows for clock cycle measurement in the debugging mode, which is used to measure cycle counts in this research.

4 Results

The results of the three implementations in comparison to those reported in [2] are shown in Table 1.

Table 1. Comparison of SCREAM-10 (E) implementations on ATMEL AVR and MSP430

	ROM Bytes			RAM Bytes	Cycles/ block	Cycles/ byte	Cycles/ S Box	Cycles/ L Box
Implementation	Code	Tables	Total					
ATMEL AVR								
C	1398	2048	3446	160	7646	478	-	-
MSP 430								
C	2104	1150	3254	254	17296	1081	131	317
Assembly								
Precomputed Tweak Keys	2184	1048	3232	46	4424	277	70	108
Tweak Keys on-the-fly	2002	1048	3096	46	4752	297	70	108

As expected, the C implementation struggles against all three assembly language implementations in terms of cycle count, but is generally on par in terms of size. However, the MSP430 assembly version using precomputed tweak keys exhibits a speedup of 1.7 over the ATMEL AVR version, and reduces total memory usage by 9 percent. The MSP430 assembly version which computes tweak keys on-the-fly is less efficient, and exhibits a speedup of 1.6 over the ATMEL AVR.

There are several observations to be made. One observation is that the MSP430 outperforms the ATMEL AVR for this particular application using assembly language. This is most likely due to the 16-bit MSP430 versus the 8-bit ATMEL AVR, and possibly due to advantages provided by the variety of addressing modes available in the MSP430.

A second observation is that, as noted by the SCREAM authors in [2], precomputation of tweak keys in SCREAM is more efficient than tweak keys computed “on-the-fly.” While this is not universally true for all tweakable block ciphers, the structure of SCREAM lends itself to easy precomputation of tweak keys, since each tweak key is repeated every three steps. Precomputation of tweak keys in this case is also memory-efficient, since only 32 additional bytes are required to store all three precomputed tweak keys. Overall, the version with precomputed tweak keys uses only 4 percent more memory than the version employing tweak keys on-the-fly, but enjoys a speedup of 1.07. In terms of cycles, 108 cycles are required for tweak key precomputation, plus approximately 10 cycles per step of recurring overhead. In contrast, tweak keys computed on-the-fly forego the 108 cycle initial computation, but require approximately 50 cycles per step.

The third observation supports claims in [2] and [13] that the bitslice construction used in the S Box is efficient. For example, in the relatively inefficient C implementation, each step consumes 1729 cycles (averaged over a 10-step implementation). Since there are two rounds and thus two S Box calls per step, 262 cycles (i.e., 131 cycles per S Box call) are spent in the S Box out of 1729 cycles per step, or only 15 percent of total cycle count. This efficiency is achieved by using only `xor`, `and`, and `not` operations for the bitslice S Box, and no look-up table accesses. The assembly versions reduce the cycle count of each S Box to 70 cycles per call.

5 Conclusion

The SCREAM Tweakable Block Cipher encryption mode with 10 steps was successfully implemented on the MSP430 microcontroller using the Reference C code and two assembly language implementations. The assembly language versions have a significantly reduced cycle count on the MSP430 in comparison to the ATMEL AVR using similar design methodology. For this application, precomputation of tweak keys is faster than tweak keys computed on-the-fly, and uses only slightly more memory. However, this is due to the simple nature of SCREAM tweak keys, and might not be applicable to Tweakable Block Ciphers in general. The bitslice and LS-cipher construction used in SCREAM are efficient in both C and assembly languages.

Future study could involve the implementation of full authenticated cipher candidates in MSP430 assembly language, comparison of precomputed tweak keys versus tweak keys computed on-the-fly in other algorithms which support such a comparison, and implementations of authenticated ciphers on the new TI MSP432 32-bit microcontroller.

References

1. “CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness.” Internet: http://competitions.cr.yp.to/caesar.html, Jun. 16, 2014 [Jul. 19, 2015].
2. V. Grosso, G. Leurent, F. Standaert, K. Varici, F. Durvaux, L. Gaspar, S. Kerckhof, “SCREAM & iSCREAM, Side-Channel Resistant Authenticated Encryption with Masking,” Update to Version 2, Internet: http://perso.uclouvain.be/fstandae/SCREAM/SCREAM_update1.pdf, Mar. 1, 2014 [Sep. 15, 2014].
3. “MSP430x5xx and MSP430x6xx Family User's Guide”, Texas Instruments, Internet: www.ti.com/lit/ug/slau208n/slau208n.pdf, 2008 (Revised 2014) [Jun 26, 2015].

4. Y. Law, J. Doumen, P. Hartel, “Survey and benchmark of block ciphers for wireless sensor networks. *ACM Transactions on Sensor Networks* (TOSN), Volume 2, Issue 1, pp. 65-93. ACM, New York, 2006.
5. M. Cazorla, K. Marquet, M. Minier, “Survey and benchmark of lightweight block ciphers for wireless sensor networks” in SECRYPT 2013 - *Proceedings of the 10th International Conference on Security and Cryptography*, Reykjavik, Iceland, pp. 29-31 July, 2013.
6. C. Wenzel-Brenner, J. Gräf, J. Pham and J.P. Kaps, “XBX Benchmarking Results,” The 3rd SHA-3 Candidate Conference, Washington, D.C., Mar. 22, 2012.
7. M. Düll, B. Haase, G. Hinterwälder, M. Hutter, C. Paar, A. H. Sánchez and P. Schwabe, “High-speed Curve25519 on 8-bit, 16-bit, and 32-bit microcontrollers,” Cryptology ePrint Archive, Report 2015/343, 2015.
8. B. Buhrow, P. Riemer, M. Shea, B. Gilbert, E. Daniel, “Block Cipher Speed and Energy Efficiency Records on the MSP430: System Design Trade-Offs for 16-bit Embedded Applications,” Cryptology ePrint Archive, Report 2015/011, 2015.
9. C. Gouvêa and J. López, “High speed implementation of authenticated encryption for the MSP430X microcontroller,” *Proceedings of the 2nd International Conference on Cryptology and Information Security in Latin America*, 2012.
10. P. Schwabe, B. Y. Yang, S. Y. Yang, “SHA-3 on ARM11 Processors”, The 3rd SHA-3 Candidate Conference, Washington, D.C., Mar. 22, 2012.
11. “SUPERCOP, European Network of Excellence in Cryptology II,” Internet: (http://bench.cr.yp.to/supercop.html), Sep. 11, 2014 [Sep. 15, 2014].
12. T. Eisenbarth, S. Heyse, I. von Maurich, T. Poeppelmann, J. Rave, C. Reuber, and A. Wild, “Evaluation of SHA-3 Candidates for 8-bit Embedded Processors,” presented at the National Institute of Standards and Technology, Second SHA Candidate Conference, Santa Barbara, CA, 2010.
13. V. Grosso, G. Leurent, F.-X. Standaert, and K. Varici, LS-designs: “Bitslice encryption for efficient masked software implementations.” *Proceedings, 21st International Workshop on Fast Software Encryption (FSE 2014)*, London, U.K., Mar. 3 – 5, 2014.