Updated version of paper published in Proc. SCIS2001, pp. 727-730.

SCIS 2001 The 2001 Symposium on Cryptography and Information Security
Oiso, Japan, January 23-26, 2001
The Institute of Electronics, Information and Communication Engineers

Performance Evaluation of AES/DES/Camellia On the 6805 and H8/300 CPUs*

Chung-Huang Yang **楊 中皇**
National Kaohsiung First University of Science and Technology **國立高雄第一科技大學**
1 University Road, Yenchao, Kaohsiung 824, Taiwan **台灣 高雄**
chyang@ccms.nkfust.edu.tw
http://www.nkfust.edu.tw/~chyang/

January 26, 2001

Abstract

Here we describe our implementation results of several cryptographic algorithms on the Motorola 6805 and Hitachi H8/300 CPUs. The implemented algorithms include Data Encryption Standard (DES), Advanced Encryption Standard (AES), and Camellia. Assembly codes were written and emulated using Hitachi's E6000 Emulator and Motorola's In-Circuit Simulator Kits.

Performance of implemented AES algorithm is 2,000 clock cycles for key schedule and 9,000 clock cycles for encryption on the 6805 CPU, this gives us approximately 30 Kbits/s encryption throughput for the MC68HC705B16 device running at 4.2 MHz external clock. Camellia achieves about same throughput while DES only got 6.4 Kbits/s, assuming all three algorithms use 2.5 KBytes ROM.

Keywords: cryptography, private-key, symmetric encryption, DES, AES, Rijndael, Camellia, microcontroller, smart card.

1. Introduction

On October 2000, NIST announced that Rijndael [1] has been selected as the proposed AES (Advanced Encryption Standard) [2] and it is expected to become an official standard within one year to replace the aging DES [3].

DES is the most well-known cryptographic algorithm since mid 1970s. DES was designed with hardware implementation in mind. It has 56-bit key and several bit permutations are performed in DES encryption/decryption. Although such permutations are easy to handle in hardware but it is costly in software or firmware implementation.

The key length of AES can be independently specified to 128, 192 or 256 bits with input block length of 128, 192 or 256 bits. This gives approximately 3.4×10^{38} possible 128-bit AES keys compared with 7.2×10^{16} possible 56-bit DES keys, therefore AES could withstand "brute-force" exhaustive attacks.

We also implemented the Camallia encryption algorithm [4]. Camellia was jointly developed by NTT and Mitsubishi and had been submitted to European NESSIE [5] and Japan's CRYPTOREC [6]. It has fixed block length of 128 bits while key size is 128,192, or 256 bits.

The Motorola 6805 and Hitachi H8/300 are two CPUs with quite different structure. The former has only one 8-bit accumulator with an 8-bit index register, in addition to a 12-bit stack pointer. On the other hand, H8/300 CPU has 14 general-purpose 8-bit registers, in addition to a 16-bit stack pointer. Target platforms for our implementation are the Motorola MC68HC705B16 [7] single-chip microcontroller and the Hitachi H8/3113 smart card [8]. Both devices have built-in EEPROM to store encryption/decryption keys and are suitable for cryptographic applications.

We wrote assembly codes to evaluate the performance of AES, DES, and Camellia in the two 8-bit microcomputers. Implementation of NIST's SHA-256 [9] is also in progress.

* Further implementation results of this paper will be put on http://www.nkfust.edu.tw/~chyang/SCIS2001/.

2. The 6805 Devices

In order to make low-cost cryptographic applications, we select Motorola's 6805 embedded devices with built-in EEPROM. MC68HC705B16 [7] microcontroller offers an 8-bit 6805 CPU, 15K-byte ROM, 256-byte EEPROM, 352-byte RAM, a timer, a serial communication interface, and four general-purpose input/output ports. The EEPROM area is where we store cryptographic keys. Figure 1 shows internal block diagram of this chip.

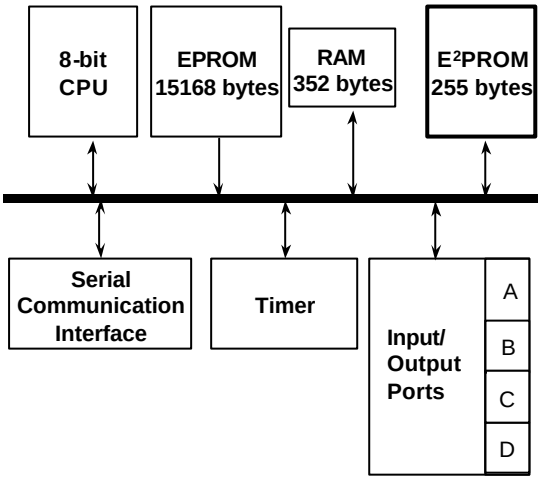

Figure 1. Block diagram of MC68HC705B16

The 6805 CPU's core has an accumulator, an index register, a 6-bit stack pointer, and a condition code register. The instruction set features 10 uncomplicated addressing modes including 8 and 16-bit indexing from the 16-bit program counter. Bit manipulation instructions are provided to set, clear, test, or jump based on a bit value anywhere in the memory map. It only has simple Math functions of addition, subtraction, and multiplication.

MC68HC705B16 has 352 bytes of RAM in two separate areas, RAM1 and RAM2, each with 176 bytes. Access time of these two areas is different with one clock cycle, RAM1 could be access with direct addressing mode while RAM2 could only be access using extended addressing mode. We will use low-speed RAM2 to store subkeys or round-keys during key schedule for AES and Camellia algorithms. Size of RAM in this device is suitable for the implementation of AES and Camellia with 128-bit key and 128-bit input block but insufficient to handle AES with 256-bit key and 256-bit block.

The MC68HC705B16 is Motorola's most popular one-time programmable (OTP) microcontroller that includes both EPROM and byte erasable EEPROM memories. It can be operated at maximum clock rate of 4.2 MHz and this gives us cycle time of approximately 476 ns. The vendor provides an in-circuit simulator kit, which is an extremely economical tool with suggested retail price of US$99, for developing and debugging target systems incorporating the 68HC05 microcontroller under Windows environment. Figure 2 shows the M68ICS05B in-circuit simulator kit. A one-time programmable device, with suggested list price of US$6.45, is also available for fast prototype development.

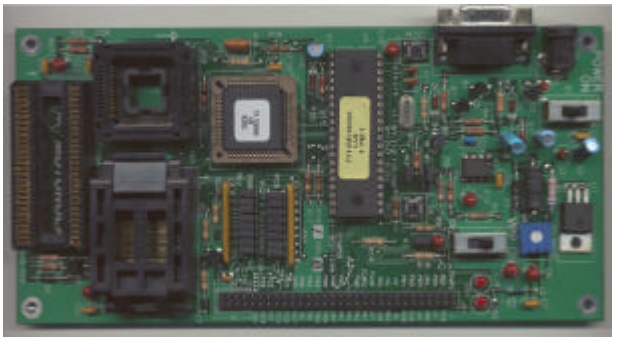

Figure 2. Motorola's in-circuit simulator kit

3. The H8/300 Devices

The H8/3113 microcomputer contains an 8-bit RISC-like H8/300 microprocessor [8], 32-Kbyte ROM, 16-Kbyte EEPROM, 2.5K-byte RAM, an arithmetic coprocessor capable of performing 1024-bit ABR^{-1} *mod* N Montgomery modular multiplication, a random number generator, and a watchdog timer. The on-chip coprocessor could be used to calculate A^B *mod* N modular exponentiation at a high speed. Figure 3 shows the block diagram of the Hitachi H8/3113 chip.

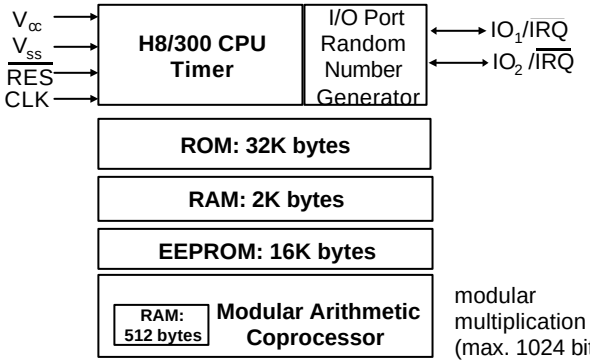

Figure 3. Block diagram of H8/3113

The H8/3113 is a powerful 8-bit microcomputer suitable for the implementation

of both symmetric and asymmetric cryptographic algorithms. With built-in random number generator and coprocessor, one could do high-speed key generation for most public-key cryptographic algorithms. Its functional drawback is in low input/output rate of smart card since only two I/O pins are available. The device can operate at a maximum of 10 MHz external clock rate or cycle time of 200 ns.

The H8/300 CPU contains 8 general-purpose registers of 16-bits each. It has 8 addressing modes and 55 basic instructions with most instructions are executed in 2 to 4 states. Basic instruction set contains multiplication and division and some instructions, such as read/write from RAM, could handle 16-bit operands as fast as 8-bit operands.

The vendor provides a real-time in-circuit emulator under Windows environment that can support the Hitachi H8/3113 smart card devices. It may be used totally self-contained, for software development and debug, or connected to a smart card reader for real time I/O debugging. Figure 4 shows the emulator.

Figure 4. H8/3113 development tools

	68HC705B16	H8/3113
CPU	8-bit	8-bit
ROM	15K	32K
EEPROM	256 bytes	16K bytes
RAM	352 bytes	2.5K bytes
Random number generator	No	Yes
Built-in coprocessor	No	Yes, 1024-bit Montgomery multiplier
Export Control	No	Yes

Table 1. Functional comparison of MC68HC705B16 with H8/3113

In the Table 1 we give some comparison information between MC68HC705B16 and H8/3113. Although the capacities of H8/3113 is obviously much more powerful than the capacities of MC68HC705B16, the H8/3113 with ROM codes of strong cryptography, such as AES, is under export control.

4. Implementation of the cryptographic algorithms

Assembly codes for DES, AES, and Camellia algorithms were written and emulated using Hitachi's E6000 Emulator and Motorola's In-Circuit Simulator Kits. Since 128-bit input with 128-bit key will be the most likely usage, our implemented assembly codes only support this key size for AES and Camellia.

The DES algorithm [3] enciphers a 64-bit input block into a 64-bit output block under the control of 56-bit cryptographic key. The 56-bit key is first transformed and expanded into 16 subkeys of 48-bits each. The encryption process first applied an initial permutation to the plaintext, passed through 16 rounds of computation with 16 subkeys, and then a final permutation is applied to give the ciphertext.

Performance of implemented DES algorithm on 6805 CPU is less than 14 ms for key schedule with encryption (or decryption) at 4.2 MHz external clock. Program size is about 2.5K bytes, including tables for 8 revised S-boxes.

Next we turn to AES. AES was designed with speed and code compactness in mind [1]. The 128-bit key is expanded and transformed into 11 subkeys of 128-bits each. Figure 5 shows the operations involved in AES encryption. Entire 11 round keys of the 128-bit AES occupy 176 bytes and could be put in RAM area to improve encryption throughput. Two tables of 256-bytes each are used to implement *ByteSubstitution* operation and its inverse. The most complicated operations involved in AES encryption is the 32-bit substitution, *MixColumn*, that involves a multiplication by the polynomial $03\ X^3 + 01\ X^2 + 01\ X + 02$ modulo $X^4 + 1$ where coefficients are given in $GF(2^8)$.

While keeping the ROM size for our AES implementation on 6805 CPU to about 2.5 Kbytes, about same size of our implementation for DES, we spent extra efforts to speedup AES performance. Performance of our preliminary implementation of AES algorithm is less than 2,000 clock cycles for key schedule and less than 9,000 cycles for encryption operation on the MC68HC705B16 device, this gives us approximately 30 Kbits/s encryption throughput

for the device running at 4.2 MHz external clock.

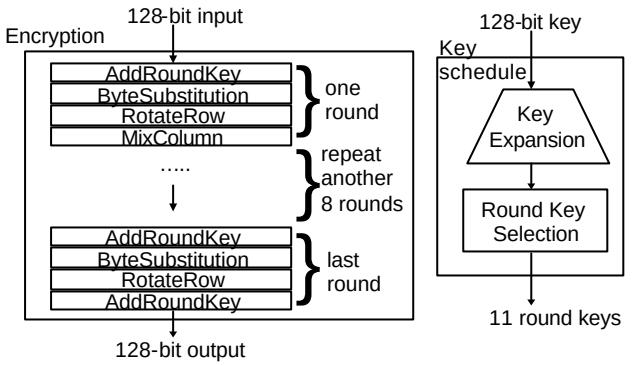

Figure 5. AES encryption for a 128-bit key and a 128-bit block

For Camellia algorithm [4], the input 128-bit key, K_L, is first used to compute another 128-bit parameter K_A. Then 26 64-bit subkeys are generated from K_L and K_A by some bit rotation operations. The encryption processes are consisted of a 18-round Feistel structure with two FL/FL^{-1}-function layers after the 6th and 12th rounds, and a 128-bit XOR operations before the first round and after the final round.

The Feistel structure is executed 18 times during encryption and 4 times during key schedule, therefore we paid especial attention on its performance and achieve 400 cycles for its operation on 6805 CPU. Performance of he implemented Camellia algorithm is about 9900 cycles for encryption on the MC68HC705B16 device and implementation results are summarized in Table 2. The implementation results using H8/3113 are given in Table 3.

	Key Schedule	Encryption (per block)	Throughput
DES 56-bit key 64-bit block	4.0 ms 8400 cycles	10.0 ms 21000 cycles	6.4 Kbits/s
AES 128-bit key 128-bit block	0.95 ms 2000 cycles	4.3 ms 9000 cycles	30 Kbits/s
Camellia 128-bit key 128-bit block	3.5 ms 7500 cycles	4.7 ms 9900 cycles	27 Kbits/s
Feal-32X 128-bit key 64-bit block	2.2 ms 4700 cycles	3.1 ms 6500 cycles	20.7 Kbits/s
RC6 128-bit key 128-bit block	36 ms 76200 cycles	?	?

Table 2. Performance of DES/AES/Camellia on MC68HC705B16 running at internal clock speed of 2.1 MHz

	Key Schedule	Encryption (per block)	Throughput
AES 128-bit key 128-bit block	0.43 ms 1080 cycles	1.67 ms 4180 cycles	76 Kbits/s
Camellia 128-bit key 128-bit block	0.95 ms 2380 cycles	1.64 ms 4100 cycles	78 Kbits/s

Table 3. Performance of AES and Camellia on H8/3113 at internal clock speed of 5 MHz

5. Conclusions

The NIST had finally proposed new Advanced Encryption Standard (AES). Here we describe our implementation efforts for symmetric cryptographic algorithms on the 6805 and H8/300 based devices. We found that both AES and Camellia perform at least twice faster than DES with same code size while key size is more than double that of DES. Export control is an issue that we need to consider.

References

1. J. Demen and V. Rijmen, "The Rijndael Block Cipher," Document version 2, March 1999, http://www.esat.kuleuven.ac.be/~rijmen/rijndael.
2. NIST, "Commerce Department Announces Winner of Global Information Security Competition," October 2, 2000, http://www.nist.gov/public_affairs/releases/g00-176.htm or http://www.nist.gov/aes/.
3. National Institute of Standards and Technology, Federal Information Processing Standard (FIPS) 46-3, *Data Encryption Standard*, October 25, 1999, http://csrc.nist.gov/fips/fips46-3.pdf
4. Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima, and Toshio Tokita, "Camellia - A 128-Bit Block Cipher Suitable for Multiple Platforms," *7th Annual Workshop on Selected Areas in Cryptography (SAC2000)*, August 2000.

http://info.isl.ntt.co.jp/camellia/Publications/sac_camellia.pdf.
5. "New European Schemes for Signatures, Integrity, and Encryption (NESSIE), " http://www.cosic.esat.kuleuven.ac.be/nessie/.
6. "Call for Cryptographic Techniques," IPA, June 2000, http://www.ipa.go.jp/security/enc/CRYPTREC/index-e.html.
7. *MC68HC05B4/705B5/05B6/05B8/(7)05B16/705B16N/(7)05B32 Technical Data,* Rev. 4, Motorola Ltd., January 1999. See http://www.mcu.motsps.com/ for more information.
8. Hitachi Single-Chip Microcomputer H8/3113 Hardware Manual, Hitachi Ltd., 1998. See http://www.hitachi.co.jp/Sicd/English/Products/micom/300micome.htm for more information.
9. New hash algorithms (SHA-256, SHA-384, and SHA-512), NIST, October 2000, http://csrc.nist.gov/cryptval/shs/sha256-384-512.pdf.