Performance Analysis of AES candidates on the 6805 CPU core

Geoffrey Keating

February 1, 1999

Abstract

The AES candidate block ciphers Crypton, RC6, and Rijndael were implemented on the Motorola 6805 series 8-bit architecture. Their performance, including ROM and RAM sizes and time to encrypt a single block, was measured in simulation, and the results presented and compared with results for the other NIST cryptography algorithms SHA and DEA and previously published results for AES candidate Twofish. Rijndael was found to be the clear "winner", but the ciphers Crypton and Twofish also performed acceptably.

The NIST is currently evaluating encryption algorithms as part of its Advanced Encryption Standard development effort. Among the requirements for the AES is that it should be efficient on small 8-bit processors as found in smart cards. Unfortunately, although most of the AES submissions presented performance estimates (sometimes even timings of actual implementations) for some kind of 8-bit processor, there were almost as many 8-bit processors used as there were submissions. In this paper, we hope to rectify this by implementing the most likely AES candidates for a single 8-bit platform, the Motorola 6805 series [2] and measuring their performance in simulation.

The candidates we chose were Crypton, RC6, and Rijndael. The authors of the Twofish AES submission [7] have already implemented Twofish on a 6805 CPU, so we simply quote their results below. These make up four of the fastest five algorithms on the reference platform; the remaining algorithm, MARS, is being worked on, although it seems less suited for smartcard implementation.

1 The 6805 processors

The processor family we chose is based around the Motorola HC05 core. There are a large number of variants, all of which use the same instruction set and tim-

Table 1: 68HC05-series processors.

Part	RAM (bytes)	EEPROM (bytes)	ROM (bytes)	typical package	approximate price (USD)
MC68HC705KJ1	64	0	1240	16-pin PDIP	0.84
MC68HC705P6A	176	0	4672	28-pin PDIP	1.89
MC68HC705JJ7	224	0	6160	20-pin PDIP	2.19
MC68HC705C8A	304	0	7774	40-pin PDIP	3.95
MC68HC05SC41	128	3008	6144	sawn wafer	
MC68HC05SC42	384	8192	32040	sawn wafer	

ings but which vary in ROM, RAM, ancillary logic, and packaging.

Table 1 lists some members of the family (on the cheaper side), with their RAM and ROM capacities. The 68HC705 parts have EPROM instead of ROM and are thus suitable for prototyping work—for instance, testing AES candidates. They can often also be programmed by the user's code, which would be useful for (for instance) loading a key schedule in the field, so that the key can be stored in EPROM.

The smartcard variants (the 68HC05SC models) in the family are not available with EPROM, and data books are not available to the casual enquirer. However, they use the same CPU core as the general-purpose microcontrollers so the resource requirements for an encryption algorithm should be the same.

The processors can run at cycle rates of up to 5Mhz, depending on the particular model and operating voltage. 2Mhz is a typical maximum.

The 6805 CPU core has four registers: 8-bit accumulator, index register, and stack pointer, and a 16-bit program counter (of which the high bits may be unimplemented). There are four broad classes of instructions:

- Register/Memory instructions, that operate between the accumulator and another value stored in memory, for instance, ADD;
- Read-modify-write instructions, that operate on a memory location, the accumulator, or the index register, for instance, INCX, which increments the index register;
- Branch instructions.
- Bit manipulation instructions, which can set, clear, or test a single bit in memory.

There are also some instructions that do not fit in these categories.

Some of the features of the core are:

Table 2: Memory requirements of the algorithms.

Cipher	RAM (bytes)			ROM (bytes)	
	Scheduled key	Encrypt	Schedule	Encrypt	Encrypt+ Schedule
Crypton	32	51	50	1101	1349
RC6	176	22	28	639	933
Rijndael	16	37	0	553	553
Twofish	24	36	?	?	2200
DES	96	21	19	680	1036
SHA	20	98	0	379	419

- Single-bit rotates and shifts only;
- An $8 \times 8 \to 16$-bit multiply instruction, which takes its inputs in the accumulator and index register and returns the result in the same places;
- Faster addressing modes that operate on the first 256 bytes of memory;
- The number of cycles required for any instruction is independent of the value of the data operated on.

The author constructed a simulation environment for this processor for the purposes of this evaluation. The simulator keeps track of the number of cycles executed, so it is possible to obtain exact cycle counts for the algorithms. The simulator, and the algorithms implemented, are available from the author's web site [4].

2 Results

The five ciphers chosen, the DES and the SHA were implemented, tested, and their performance measured. The results are shown in table 2 for the memory requirements and table 3 for the execution time.

In the table, 'Encrypt' is the resources required to encrypt a single 128-bit block (64 bits for DES), or to step the hash function once (hashing 512 bits). 'Schedule' is the resources required to schedule a 128-bit key (56 bits for DES), or to set up the initial hash.

Note that the 'Encrypt' resource requirements for Twofish include the capability of performing decryption; for the other ciphers, this is not the case. Also, all RAM requirements assume that the input data is stored in RAM; for some key schedules this is not necessary.

Table 3: Execution time of the algorithms.

Cipher	Time (cycles)	
	Encrypt	Schedule
Crypton	31524	5075
RC6	32731	82167
Rijndael	14945	0
Twofish	26500	1750
DES	17458	12320
SHA	67244	478

In general, the algorithms were implemented to fit within 128 bytes of RAM including the key schedule, and 64 bytes of RAM if the key is scheduled into ROM or EEPROM. The algorithms were implemented to take about 1024 bytes of ROM, but flexibility was allowed where this would cause a large speed penalty.

The algorithms were implemented from the specification in the AES submissions. The author did not spend large amounts of time studying each algorithm, and it is quite possible that a significant optimisation may have been missed if it was not discussed in the AES submission. Certainly, there are minor improvements that can be made to the implementations.

Updated versions of this paper are available from [4].

2.1 Crypton

Crypton is unexpectedly slow. It turns out that the π transformation takes 1140 clock cycles to execute because of the single accumulator, accounting for 43% of the execution time. The implementation above executes π twelve times; it would be possible to reduce this by one by specifying, in the notation of [5], that the final round is performed as $\sigma_{K_e^0} \circ \tau \circ \gamma_e$. Of course, this would not help decryption.

2.2 RC6

RC6 includes a variable-size rotate, which takes 260 clock cycles when implemented to be constant-time. The polynomial $(2x + 1)x$ also takes 260 clock cycles to compute. Since these are executed 40 times each, they account for 31% each of the total encryption time. About half, on average, of the time taken for the variable-size rotate could be eliminated if a constant-time implementation was unnecessary.

2.3 Rijndael

Rijndael was the smallest and fastest cipher of those implemented, being more than twice as fast as DES. Rijndael also deserves special mention for being implemented in 64 bytes of RAM.

Rijndael uses a primitive, `xtime`, which would, if not implemented carefully, allow for a timing attack. Making it timing-independent cost about 576 clock cycles.

2.4 Twofish

The Twofish AES submission [7] also gives a range of alternative Twofish implementations, varying between the one given in the table and one that uses 1760 bytes of ROM and takes about 37100 clock cycles per block.

2.5 DES

The DES implementation performs a partial key schedule to fit in 128 bytes of RAM. The Twofish authors quote a DES implementation on the 6805 that takes about 2k code, 23 bytes of RAM, and about 20000 clocks/block. This fits well with our estimates.

3 Future Work

Future work will concentrate on the ciphers Mars, E2, and CAST-128, and the implementation of decryption for Rijndael. Further work would concentrate on the AES second round candidates, particularly any that have been overlooked in this paper.

4 Conclusions

It is difficult to draw a direct conclusion from the above results, because of the question (which is not addressed here) of the relative security of the algorithms.

However, it does seem than Rijndael performs remarkably well in a quite reasonable amount of ROM (even allowing for the need for separate encryption and decryption algorithms), and, more importantly, in a very restricted amount of RAM. Rijndael also performs quite well on the AES reference platform.

Also doing well is Twofish, although it seems to require a substantial amount of ROM.

By far the most reassuring result is that all the algorithms examined so far are faster per byte encrypted than DES (and much faster than Triple-DES, of course), and most are smaller than DES.

References

[1] Carolynn Burwick, Don Coppersmith, Edward D'Avignon, Rosario Gennaro, Shai Halevi, Charanjit Jutla, Stephen M. Matyas Jr., Luke O'Connor, Mohammad Peyravi, David Stafford, and Nevenko Zunic. *MARS—a candidate cipher for AES*. IBM Corporation, June 1998. AES submission.

[2] CISC System Design Group, Motorola Inc., Austin, Texas. *68HC705P6A General Release Specification*, July 1996. Rev. 1.0.

[3] Joan Daernen and Vincent Rijmen. *AES Proposal: Rijndael*, June 1998. AES submission.

[4] `http://www.ozemail.com.au/%7Egeoffk/aes-6805/`.

[5] Chae Hoon Lim. *CRYPTON: A New 128-bit Block Cipher*, 1998. AES submission.

[6] Ronald L. Rivest, M.J.B. Robshaw, R. Sidney, and V. L. Yin. *The RC6 Block Cipher*, 1998. AES submission.

[7] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson. *Twofish: A 128-Bit Block Cipher*, June 1998. AES submission.