FIPS PUB 180

FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION

# SECURE HASH STANDARD

CATEGORY: COMPUTER SECURITY

Computer Systems Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899

Issued May 11, 1993

**U.S. Department of Commerce**
Ronald H. Brown, Secretary

National Institute of Standards
and Technology
Raymond G. Kammer, Acting Director

## Foreword

The Federal Information Processing Standards Publication Series of the National Institute of Standards and Technology (NIST) is the official publication relating to standards and guidelines adopted and promulgated under the provisions of Section 111(d) of the Federal Property and Administrative Services Act of 1949 as amended by the Computer Security Act of 1987, Public Law 100-235. These mandates have given the Secretary of Commerce and NIST important responsibilities for improving the utilization and management of computer and related telecommunications systems in the Federal Government. The NIST through its Computer Systems Laboratory provides leadership, technical guidance, and coordination of Government efforts in the development of standards and guidelines in these areas.

Comments concerning Federal Information Processing Standards Publications are welcomed and should be addressed to the Director, Computer Systems Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899.

James H. Burrows, Director
Computer Systems
Laboratory

---

## Abstract

This standard specifies a Secure Hash Algorithm (SHA) which can be used to generate a condensed representation of a message called a message digest. The SHA is required for use with the Digital Signature Algorithm (DSA) as specified in the Digital Signature Standard (DSS) and whenever a secure hash algorithm is required for Federal applications. The SHA is used by both the transmitter and intended receiver of a message in computing and verifying a digital signature.

Key words: computer security, digital signatures, Federal Information Processing Standard, hash algorithm.

---

National Institute of Standards and Technology
FIPS PUB 180
22 pages (May 11, 1993)
CODEN: FIPPAT

U.S. Government Printing Office
Washington: 1993

For sale by the National Technical Information Service
U.S. Department of Commerce
Springfield, VA 22161

**Federal Information**
**Processing Standards Publication 180**

**1993 May 11**

**Announcing the**

# SECURE HASH STANDARD

Federal Information Processing Standards Publications (FIPS PUBS) are issued by the National Institute of Standards and Technology (NIST) after approval by the Secretary of Commerce pursuant to Section 111(d) of the Federal Property and Administrative Services Act of 1949 as amended by the Computer Security Act of 1987, Public Law 100-235.

**Name of Standard**: Secure Hash Standard.

**Category of Standard**: Computer Security.

**Explanation**: This Standard specifies a Secure Hash Algorithm (SHA) for computing a condensed representation of a message or a data file. When a message of any length $< 2^{64}$ bits is input, the SHA produces a 160-bit output called a message digest. The message digest can then be input to the Digital Signature Algorithm (DSA) which generates or verifies the signature for the message (see Figure 1). Signing the message digest rather than the message often improves the efficiency of the process because the message digest is usually much smaller in size than the message. The same hash algorithm must be used by the verifier of a digital signature as was used by the creator of the digital signature. The SHA is called secure because it is computationally infeasible to find a message which corresponds to a given message digest, or to find two different messages which produce the same message digest. Any change to a message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify. The SHA is based on principles similar to those used by Professor Ronald L. Rivest of MIT when designing the MD4 message digest algorithm[1], and is closely modelled after that algorithm.

**Approving Authority**: Secretary of Commerce.

---

[1]"The MD4 Message Digest Algorithm," Advances in Cryptology - CRYPTO '90 Proceedings, Springer-Verlag, 1991, pp. 303-311.

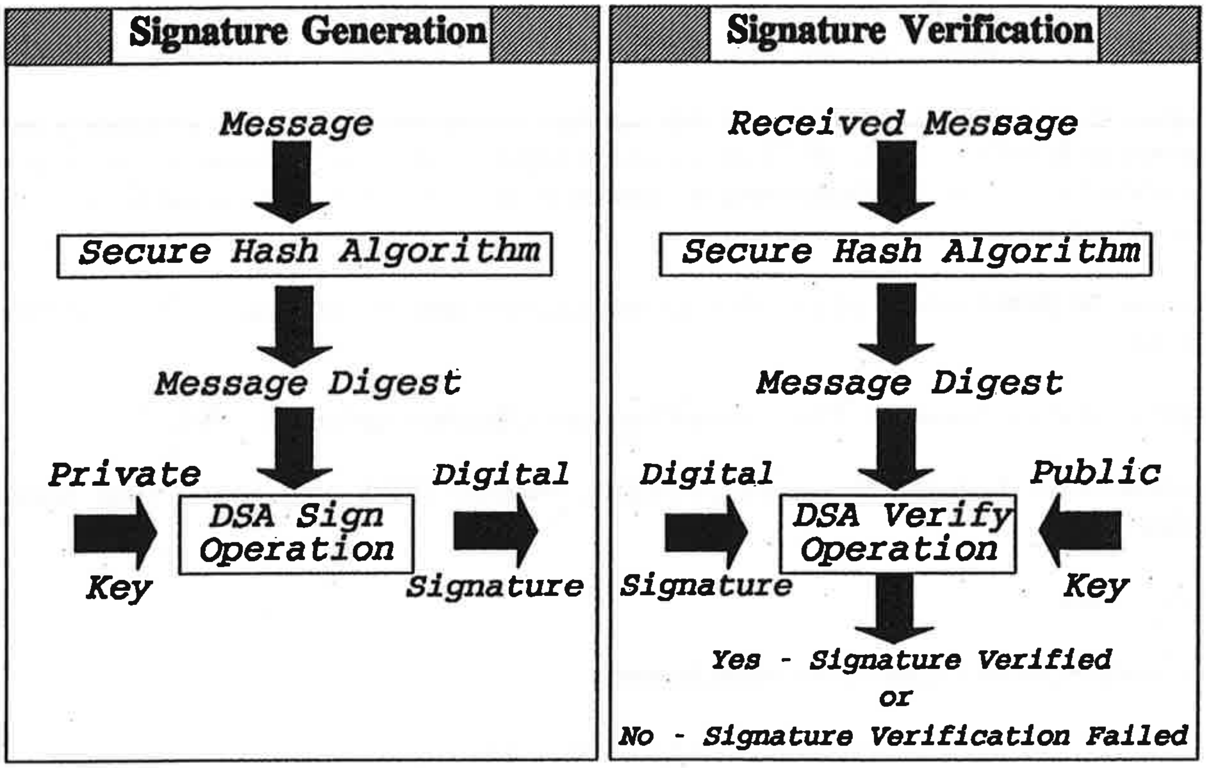


**Figure 1: Using the SHA with the DSA**

**Maintenance Agency**: U.S. Department of Commerce, National Institute of Standards and Technology, Computer Systems Laboratory.

**Applicability**: This standard is applicable to all Federal departments and agencies for the protection of unclassified information that is not subject to section 2315 of Title 10, United States Code, or section 3502(2) of Title 44, United States Code. This standard is required for use with the Digital Signature Algorithm (DSA) as specified in the Digital Signature Standard (DSS) and whenever a secure hash algorithm is required for federal applications. Private and commercial organizations are encouraged to adopt and use this standard.

**Applications**: The SHA may be used with the DSA in electronic mail, electronic funds transfer, software distribution, data storage, and other applications which require data integrity assurance and data origin authentication. The SHA may also be used whenever it is necessary to generate a condensed version of a message.

**Implementations**: The SHA may be implemented in software, firmware, hardware, or any combination thereof. Only implementations of the SHA that are validated by NIST will be considered as complying with this standard. Information about the requirements for validating implementations of this standard can be obtained from the National Institute of Standards and Technology, Computer Systems Laboratory, Attn: SHS Validation, Gaithersburg, MD 20899.

**Export Control:** Implementations of this standard are subject to Federal Government export controls as specified in Title 15, Code of Federal Regulations, Parts 768 through 799. Exporters are advised to contact the Department of Commerce, Bureau of Export Administration for more information.

**Patents:** Implementations of the SHA in this standard may be covered by U.S. and foreign patents.

**Implementation Schedule:** This standard becomes effective October 15, 1993.

**Specifications:** Federal Information Processing Standard (FIPS 180) Secure Hash Standard (affixed).

**Cross Index:**

a. FIPS PUB 46-1, Data Encryption Standard.

b. FIPS PUB 73, Guidelines for Security of Computer Applications.

c. Draft FIPS PUB 140-1, Security Requirements for Cryptographic Modules.

d. FIPS PUB XX, Digital Signature Standard.

**Qualifications:** While it is the intent of this standard to specify a secure hash algorithm, conformance to this standard does not assure that a particular implementation is secure. The responsible authority in each agency or department shall assure that an overall implementation provides an acceptable level of security. This standard will be reviewed every five years in order to assess its adequacy.

**Waiver Procedure:** Under certain exceptional circumstances, the heads of Federal departments and agencies may approve waivers to Federal Information Processing Standards (FIPS). The head of such agency may redelegate such authority only to a senior official designated pursuant to section 3506(b) of Title 44, United States Code. Waiver shall be granted only when:

a. Compliance with a standard would adversely affect the accomplishment of the mission of an operator of a Federal computer system; or

b. Compliance with a standard would cause a major adverse financial impact on the operator which is not offset by Government-wide savings.

Agency heads may act upon a written waiver request containing the information detailed above. Agency heads may also act without a written waiver request when they determine that conditions for meeting the standard cannot be met. Agency heads may approve waivers only by a written decision which explains the basis on which the agency head made the required finding(s). A copy of each decision, with procurement sensitive or classified portions clearly identified, shall be sent to: National Institute of Standards and Technology; ATTN: FIPS Waiver Decisions, Technology Building, Room B-154, Gaithersburg, MD 20899.

In addition, notice of each waiver granted and each delegation of authority to approve waivers shall be sent promptly to the Committee on Government Operations of the House of Representatives and the Committee on Government Affairs of the Senate and shall be published promptly in the Federal Register.

When the determination on a waiver applies to the procurement of equipment and/or services, a notice of the waiver determination must be published in the Commerce Business Daily as a part of the notice of solicitation for offers of an acquisition or, if the waiver determination is made after that notice is published, by amendment to such notice.

A copy of the waiver, any supporting documents, the document approving the waiver and any accompanying documents, with such deletions as the agency is authorized and decides to make under 5 United States Code Section 552(b), shall be part of the procurement documentation and retained by the agency.

**Where to Obtain Copies of the Standard**: Copies of this publication are for sale by the National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161. When ordering, refer to Federal Information Processing Standards Publication 180 (FIPSPUB180), and identify the title. When microfiche is desired, this should be specified. Prices are published by NTIS in current catalogs and other issuances. Payment may be made by check, money order, deposit account or charged to a credit card accepted by NTIS.

**Federal Information**
**Processing Standards Publication 180**

**1993 May 11**

**Specifications for the**

# SECURE HASH STANDARD

## 1. INTRODUCTION

The Secure Hash Algorithm (SHA) is required for use with the Digital Signature Algorithm (DSA) as specified in the Digital Signature Standard (DSS) and whenever a secure hash algorithm is required for federal applications. For a message of length < $2^{64}$ bits, the SHA produces a 160-bit condensed representation of the message called a message digest. The message digest is used during generation of a signature for the message. The SHA is also used to compute a message digest for the received version of the message during the process of verifying the signature. Any change to the message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify.

The SHA is designed to have the following properties: it is computationally infeasible to find a message which corresponds to a given message digest, or to find two different messages which produce the same message digest.

## 2. BIT STRINGS AND INTEGERS

The following terminology related to bit strings and integers will be used:

a. A *hex digit* is an element of the set `{0, 1, ... , 9, A, ... , F}`. A hex digit is the representation of a 4-bit string. **Examples:** `7 = 0111, A = 1010.`

b. A *word* equals a 32-bit string which may be represented as a sequence of 8 hex digits. To convert a word to 8 hex digits each 4-bit string is converted to its hex equivalent as described in (a) above. **Example:**

`1010 0001 0000 0011 1111 1110 0010 0011.` = A103FE23

c. An *integer between 0 and* $2^{32}$ *- 1 inclusive* may be represented as a word. The least significant four bits of the integer are represented by the right-most hex digit of the word representation. **Example:** the integer 291 = $2^8+2^5+2^1+2^0$ = 256+32+2+1 is represented by the hex word, 00000123.

If z is an integer, $0 \leq z < 2^{64}$, then $z = 2^{32}x + y$ where $0 \leq x < 2^{32}$ and $0 \leq y < 2^{32}$. Since x and y can be represented as words X and Y, respectively, z can be represented as the pair of words (X,Y).

d. *block* = 512-bit string. A block (e.g., B) may be represented as a sequence of 16 words.

## 3. OPERATIONS ON WORDS

The following logical operators will be applied to words:

a. ***Bitwise logical word operations***

| | |
|---|---|
| $X \wedge Y$ | = bitwise logical "and" of X and Y. |
| $X \vee Y$ | = bitwise logical "inclusive-or" of X and Y. |
| X XOR Y | = bitwise logical "exclusive-or" of X and Y. |
| ~X | = bitwise logical "complement" of X. |

**Example:**

```
      01101100101110011101001001111011
XOR   01100101110000010110100110110111
      --------------------------------
 =    00001001011110001011101111001100
```

b. The *operation X + Y* is defined as follows: words X and Y represent integers x and y, where $0 \leq x < 2^{32}$ and $0 \leq y < 2^{32}$. For positive integers n and m, let n mod m be the remainder upon dividing n by m. Compute

$$z = (x + y) \bmod 2^{32}.$$

Then $0 \leq z < 2^{32}$. Convert z to a word, Z, and define Z = X + Y.

c. The *circular left shift operation* $S^n(X)$, where X is a word and n is an integer with $0 \leq n < 32$, is defined by

$$S^n(X) = (X << n) \vee (X >> 32\text{-}n).$$

In the above, X << n is obtained as follows: discard the left-most n bits of X and then pad the result with n zeroes on the right (the result will still be 32 bits). X >> n is obtained by discarding the right-most n bits of X and then padding the result with n zeroes on the left. Thus $S^n(X)$ is equivalent to a circular shift of X by n positions to the left.

## 4. MESSAGE PADDING

The SHA is used to compute a message digest for a message or data file that is provided as input. The message or data file should be considered to be a bit string. The length of the message is the number of bits in the message (the empty message has length 0). If the number of bits in a message is a multiple of 8, for compactness we can represent the message in hex. The purpose of message padding is to make the total length of a padded message a multiple of 512. The SHA sequentially processes blocks of 512 bits when computing the message digest. The following specifies how this padding shall be performed. As a summary, a "1" followed by m "0"s followed by a 64-bit integer are appended to the end of the message to produce a padded message of length $512 \times n$. The 64-bit integer is $l$, the length of the original message. The padded message is then processed by the SHA as n 512-bit blocks.

Suppose a message has length $l < 2^{64}$. Before it is input to the SHA, the message is padded on the right as follows:

a. "1" is appended. **Example:** if the original message is "01010000", this is padded to "010100001".

b. "0"s are appended. The number of "0"s will depend on the original length of the message. The last 64 bits of the last 512-bit block are reserved for the length $l$ of the original message.

**Example:** Suppose the original message is the bit string

```
01100001 01100010 01100011 01100100 01100101.
```

After step (a) this gives

```
01100001 01100010 01100011 01100100 01100101 1.
```

Since $l = 40$, the number of bits in the above is 41 and 407 "0"s are appended, making the total now 448. This gives (in hex)

```
61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000.
```

c. Obtain the 2-word representation of $l$, the number of bits in the original message. If $l < 2^{32}$ then the first word is all zeroes. Append these two words to the padded message.

**Example:** Suppose the original message is as in (b). Then $l = 40$ (note that $l$ is computed before any padding). The two-word representation of 40 is hex `00000000 00000028`. Hence the final padded message is hex

```
61626364 65800000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000028.
```

The padded message will contain 16n words for some $n > 0$. The padded message is regarded as a sequence of n blocks $M_1$ , $M_2$, ... , $M_n$, where each $M_i$ contains 16 words and $M_1$ contains the first characters (or bits) of the message.

## 5. FUNCTIONS USED

A sequence of logical functions $f_0, f_1, \ldots, f_{79}$ is used in the SHA. Each $f_t$, $0 \le t \le 79$, operates on three 32-bit words and produces a 32-bit word as output. $f_t$ is defined as follows: for words, B, C, D,

$$f_t(B,C,D) = (B \wedge C) \vee (\sim B \wedge D) \qquad (0 \le t \le 19)$$

$$f_t(B,C,D) = B \text{ XOR } C \text{ XOR } D \qquad (20 \le t \le 39)$$

$$f_t(B,C,D) = (B \wedge C) \vee (B \wedge D) \vee (C \wedge D) \qquad (40 \le t \le 59)$$

$$f_t(B,C,D) = B \text{ XOR } C \text{ XOR } D \qquad (60 \le t \le 79).$$

## 6. CONSTANTS USED

A sequence of constant words $K_0, K_1, \ldots, K_{79}$ is used in the SHA. In hex these are given by

$K_t$ = `5A827999` $(0 \le t \le 19)$

$K_t$ = `6ED9EBA1` $(20 \le t \le 39)$

$K_t$ = `8F1BBCDC` $(40 \le t \le 59)$

$K_t$ = `CA62C1D6` $(60 \le t \le 79)$.

## 7. COMPUTING THE MESSAGE DIGEST

The message digest is computed using the final padded message. The computation uses two buffers, each consisting of five 32-bit words, and a sequence of eighty 32-bit words. The words of the first 5-word buffer are labeled A,B,C,D,E. The words of the second 5-word buffer are labeled $H_0$, $H_1$, $H_2$, $H_3$, $H_4$. The words of the 80-word sequence are labeled $W_0$, $W_1$, ... , $W_{79}$. A single word buffer TEMP is also employed.

To generate the message digest, the 16-word blocks $M_1$, $M_2$, ... , $M_n$ defined in Section 4 are processed in order. The processing of each $M_i$ involves 80 steps.

Before processing any blocks, the $\{H_j\}$ are initialized as follows: in hex,

$H_0$ = 67452301

$H_1$ = EFCDAB89

$H_2$ = 98BADCFE

$H_3$ = 10325476

$H_4$ = C3D2E1F0.

Now $M_1$, $M_2$, ... , $M_n$ are processed. To process $M_i$, we proceed as follows:

a. Divide $M_i$ into 16 words $W_0$, $W_1$, ... , $W_{15}$, where $W_0$ is the left-most word.

b. For t = 16 to 79 let $W_t = W_{t-3}$ XOR $W_{t-8}$ XOR $W_{t-14}$ XOR $W_{t-16}$.

c. Let $A = H_0$, $B = H_1$, $C = H_2$, $D = H_3$, $E = H_4$.

d. For t = 0 to 79 do

$$TEMP = S^5(A) + f_t(B,C,D) + E + W_t + K_t;$$

$$E = D; \quad D = C; \quad C = S^{30}(B); \quad B = A; \; A = TEMP;$$

e. Let $H_0 = H_0 + A$, $H_1 = H_1 + B$, $H_2 = H_2 + C$, $H_3 = H_3 + D$, $H_4 = H_4 + E$.

After processing $M_n$, the message digest is the 160-bit string represented by the 5 words

$H_0 \, H_1 \, H_2 \, H_3 \, H_4$.

## 8. ALTERNATE METHOD OF COMPUTATION

The above assumes that the sequence $W_0, \ldots, W_{79}$ is implemented as an array of eighty 32-bit words. This is efficient from the standpoint of minimization of execution time, since the addresses of $W_{t-3}, \ldots, W_{t-16}$ in step (b) are easily computed. If space is at a premium, an alternative is to regard $\{ W_t \}$ as a circular queue, which may be implemented using an array of sixteen 32-bit words W[0], ... W[15]. In this case, in hex let MASK = 0000000F. Then processing of $M_i$ is as follows:

a. Divide $M_i$ into 16 words W[0], ... , W[15], where W[0] is the left-most word.

b. Let $A = H_0$, $B = H_1$, $C = H_2$, $D = H_3$, $E = H_4$.

c. For t = 0 to 79 do

$s = t \wedge \text{MASK};$

if $(t \geq 16)$ $W[s] = W[(s + 13) \wedge \text{MASK}]$ XOR $W[(s + 8) \wedge \text{MASK}]$ XOR $W[(s + 2) \wedge \text{MASK}]$ XOR $W[s]$;

$\text{TEMP} = S^5(A) + f_t(B,C,D) + E + W[s] + K_t;$

$E = D;\ D = C;\ C = S^{30}(B);\ B = A;\ A = \text{TEMP};$

d. Let $H_0 = H_0 + A$, $H_1 = H_1 + B$, $H_2 = H_2 + C$, $H_3 = H_3 + D$, $H_4 = H_4 + E$.

## 9. COMPARISON OF METHODS

The methods of Sections 7 and 8 yield the same message digest. Although using the method of Section 8 saves sixty-four 32-bit words of storage, it is likely to lengthen execution time due to the increased complexity of the address computations for the {W[t]} in step (c). Other computation methods which give identical results may be implemented in conformance with the standard.

## APPENDIX A. A SAMPLE MESSAGE AND ITS MESSAGE DIGEST

This appendix is for informational purposes only and is not required to meet the standard.

Let the message be the ASCII binary-coded form of "abc", i.e.,

```
01100001  01100010  01100011.
```

This message has length $l = 24$. In step (a) of Section 4, we append "1". In step (b) we append 423 "0"s. In step (c) we append hex `00000000 00000018`, the 2-word representation of 24. Thus the final padded message consists of one block, so that n = 1 in the notation of Section 4. The single block has hex words

```
W[0]  = 61626380
W[1]  = 00000000
W[2]  = 00000000
W[3]  = 00000000
W[4]  = 00000000
W[5]  = 00000000
W[6]  = 00000000
W[7]  = 00000000
W[8]  = 00000000
W[9]  = 00000000
W[10] = 00000000
W[11] = 00000000
W[12] = 00000000
W[13] = 00000000
W[14] = 00000000
W[15] = 00000018.
```

The initial hex values of $\{H_i\}$ are

$H_0$ = `67452301`

$H_1$ = `EFCDAB89`

$H_2$ = `98BADCFE`

$H_3$ = `10325476`

$H_4$ = `C3D2E1F0`.

The hex values of A,B,C,D,E after pass t of the "for t = 0 to 79" loop (step (d) of Section 7 or (c) of Section 8) are

| | A | B | C | D | E |
|---|---|---|---|---|---|
| t = 0: | 0116FC33 | 67452301 | 7BF36AE2 | 98BADCFE | 10325476 |
| t = 1: | 8990536D | 0116FC33 | 59D148C0 | 7BF36AE2 | 98BADCFE |
| t = 2: | A1390F08 | 8990536D | C045BF0C | 59D148C0 | 7BF36AE2 |
| t = 3: | CDD8E11B | A1390F08 | 626414DB | C045BF0C | 59D148C0 |
| t = 4: | CFD499DE | CDD8E11B | 284E43C2 | 626414DB | C045BF0C |
| t = 5: | 3FC7CA40 | CFD499DE | F3763846 | 284E43C2 | 626414DB |
| t = 6: | 993E30C1 | 3FC7CA40 | B3F52677 | F3763846 | 284E43C2 |
| t = 7: | 9E8C07D4 | 993E30C1 | 0FF1F290 | B3F52677 | F3763846 |
| t = 8: | 4B6AE328 | 9E8C07D4 | 664F8C30 | 0FF1F290 | B3F52677 |
| t = 9: | 8351F929 | 4B6AE328 | 27A301F5 | 664F8C30 | 0FF1F290 |
| t = 10: | FBDA9E89 | 8351F929 | 12DAB8CA | 27A301F5 | 664F8C30 |
| t = 11: | 63188FE4 | FBDA9E89 | 60D47E4A | 12DAB8CA | 27A301F5 |
| t = 12: | 4607B664 | 63188FE4 | 7EF6A7A2 | 60D47E4A | 12DAB8CA |
| t = 13: | 9128F695 | 4607B664 | 18C623F9 | 7EF6A7A2 | 60D47E4A |
| t = 14: | 196BEE77 | 9128F695 | 1181ED99 | 18C623F9 | 7EF6A7A2 |
| t = 15: | 20BDD62F | 196BEE77 | 644A3DA5 | 1181ED99 | 18C623F9 |
| t = 16: | ED2FF4A3 | 20BDD62F | C65AFB9D | 644A3DA5 | 1181ED99 |
| t = 17: | 565DF73C | ED2FF4A3 | C82F758B | C65AFB9D | 644A3DA5 |
| t = 18: | 550B1E7F | 565DF73C | FB4BFD28 | C82F758B | C65AFB9D |
| t = 19: | FE0F9E4B | 550B1E7F | 15977DCF | FB4BFD28 | C82F758B |
| t = 20: | B4D4C943 | FE0F9E4B | D542C79F | 15977DCF | FB4BFD28 |
| t = 21: | 43993572 | B4D4C943 | FF83E792 | D542C79F | 15977DCF |
| t = 22: | F7106486 | 43993572 | ED353250 | FF83E792 | D542C79F |
| t = 23: | 775924E6 | F7106486 | 90E64D5C | ED353250 | FF83E792 |
| t = 24: | 45A7EF23 | 775924E6 | BDC41921 | 90E64D5C | ED353250 |
| t = 25: | CCEAD674 | 45A7EF23 | 9DD64939 | BDC41921 | 90E64D5C |
| t = 26: | 02D0C6D1 | CCEAD674 | D169FBC8 | 9DD64939 | BDC41921 |
| t = 27: | 070C437F | 02D0C6D1 | 333AB59D | D169FBC8 | 9DD64939 |
| t = 28: | 301E90BE | 070C437F | 40B431B4 | 333AB59D | D169FBC8 |
| t = 29: | B898C685 | 301E90BE | C1C310DF | 40B431B4 | 333AB59D |
| t = 30: | 669723E2 | B898C685 | 8C07A42F | C1C310DF | 40B431B4 |
| t = 31: | D9316F96 | 669723E2 | 6E2631A1 | 8C07A42F | C1C310DF |
| t = 32: | DB81A5C7 | D9316F96 | 99A5C8F8 | 6E2631A1 | 8C07A42F |
| t = 33: | 99C8DFB2 | DB81A5C7 | B64C5BE5 | 99A5C8F8 | 6E2631A1 |
| t = 34: | 6BE6AE07 | 99C8DFB2 | F6E06971 | B64C5BE5 | 99A5C8F8 |
| t = 35: | C01CC62C | 6BE6AE07 | A67237EC | F6E06971 | B64C5BE5 |
| t = 36: | 6433FDD0 | C01CC62C | DAF9AB81 | A67237EC | F6E06971 |
| t = 37: | 0A33CCF7 | 6433FDD0 | 3007318B | DAF9AB81 | A67237EC |
| t = 38: | 4BF58DC8 | 0A33CCF7 | 190CFF74 | 3007318B | DAF9AB81 |
| t = 39: | EBBD5233 | 4BF58DC8 | C28CF33D | 190CFF74 | 3007318B |
| t = 40: | 825A3460 | EBBD5233 | 12FD6372 | C28CF33D | 190CFF74 |
| t = 41: | B62CBB93 | 825A3460 | FAEF548C | 12FD6372 | C28CF33D |
| t = 42: | AA3F9707 | B62CBB93 | 20968D18 | FAEF548C | 12FD6372 |
| t = 43: | FE1D0273 | AA3F9707 | ED8B2EE4 | 20968D18 | FAEF548C |
| t = 44: | 57AD526B | FE1D0273 | EA8FE5C1 | ED8B2EE4 | 20968D18 |
| t = 45: | 93EBBE3F | 57AD526B | FF87409C | EA8FE5C1 | ED8B2EE4 |
| t = 46: | F9ADF47B | 93EBBE3F | D5EB549A | FF87409C | EA8FE5C1 |

| | | | | | |
|---|---|---|---|---|---|
| t = 47: | 875586D2 | F9ADF47B | E4FAEF8F | D5EB549A | FF87409C |
| t = 48: | D0A22FFB | 875586D2 | FE6B7D1E | E4FAEF8F | D5EB549A |
| t = 49: | C12B6426 | D0A22FFB | A1D561B4 | FE6B7D1E | E4FAEF8F |
| t = 50: | EBC90281 | C12B6426 | F4288BFE | A1D561B4 | FE6B7D1E |
| t = 51: | E7D0EC05 | EBC90281 | B04AD909 | F4288BFE | A1D561B4 |
| t = 52: | 7CB98E55 | E7D0EC05 | 7AF240A0 | B04AD909 | F4288BFE |
| t = 53: | 0D48DBA2 | 7CB98E55 | 79F43B01 | 7AF240A0 | B04AD909 |
| t = 54: | C2D477BF | 0D48DBA2 | 5F2E6395 | 79F43B01 | 7AF240A0 |
| t = 55: | 236BD48D | C2D477BF | 835236E8 | 5F2E6395 | 79F43B01 |
| t = 56: | 9B4364D6 | 236BD48D | F0B51DEF | 835236E8 | 5F2E6395 |
| t = 57: | 5B8C33C9 | 9B4364D6 | 48DAF523 | F0B51DEF | 835236E8 |
| t = 58: | BE2A4656 | 5B8C33C9 | A6D0D935 | 48DAF523 | F0B51DEF |
| t = 59: | 8FF296DB | BE2A4656 | 56E30CF2 | A6D0D935 | 48DAF523 |
| t = 60: | C10C8993 | 8FF296DB | AF8A9195 | 56E30CF2 | A6D0D935 |
| t = 61: | 6AC23CBF | C10C8993 | E3FCA5B6 | AF8A9195 | 56E30CF2 |
| t = 62: | 0708247D | 6AC23CBF | F0432264 | E3FCA5B6 | AF8A9195 |
| t = 63: | 35D201F8 | 0708247D | DAB08F2F | F0432264 | E3FCA5B6 |
| t = 64: | 969B2FC8 | 35D201F8 | 41C2091F | DAB08F2F | F0432264 |
| t = 65: | 3CAC6514 | 969B2FC8 | 0D74807E | 41C2091F | DAB08F2F |
| t = 66: | 14CD9A35 | 3CAC6514 | 25A6CBF2 | 0D74807E | 41C2091F |
| t = 67: | BA564047 | 14CD9A35 | 0F2B1945 | 25A6CBF2 | 0D74807E |
| t = 68: | C241F74D | BA564047 | 4533668D | 0F2B1945 | 25A6CBF2 |
| t = 69: | 2896B70F | C241F74D | EE959011 | 4533668D | 0F2B1945 |
| t = 70: | 564BBED1 | 2896B70F | 70907DD3 | EE959011 | 4533668D |
| t = 71: | 8FA15D5A | 564BBED1 | CA25ADC3 | 70907DD3 | EE959011 |
| t = 72: | 9A226C11 | 8FA15D5A | 5592EFB4 | CA25ADC3 | 70907DD3 |
| t = 73: | F0B94489 | 9A226C11 | A3E85756 | 5592EFB4 | CA25ADC3 |
| t = 74: | 1809D5E2 | F0B94489 | 66889B04 | A3E85756 | 5592EFB4 |
| t = 75: | B86C5A40 | 1809D5E2 | 7C2E5122 | 66889B04 | A3E85756 |
| t = 76: | DFE7E487 | B86C5A40 | 86027578 | 7C2E5122 | 66889B04 |
| t = 77: | 70286C07 | DFE7E487 | 2E1B1690 | 86027578 | 7C2E5122 |
| t = 78: | 24FF7ED5 | 70286C07 | F7F9F921 | 2E1B1690 | 86027578 |
| t = 79: | 9A1F95A8 | 24FF7ED5 | DC0A1B01 | F7F9F921 | 2E1B1690. |

After processing, the hex values of $\{H_i\}$ are

$H_0$ = 67452301 + 9A1F95A8 = 0164B8A9

$H_1$ = EFCDAB89 + 24FF7ED5 = 14CD2A5E

$H_2$ = 98BADCFE + DC0A1B01 = 74C4F7FF

$H_3$ = 10325476 + F7F9F921 = 082C4D97

$H_4$ = C3D2E1F0 + 2E1B1690 = F1EDF880.

Message digest = 0164B8A9 14CD2A5E 74C4F7FF 082C4D97 F1EDF880

## APPENDIX B. A SECOND SAMPLE MESSAGE AND ITS MESSAGE DIGEST

This appendix is for informational purposes only and is not required to meet the standard.

Let the message be the binary-coded form (cf. Appendix A) of the ASCII string

"abcdbcdecdefdefgefghfghighijhijkijkljklmklmnlmnomnopnopq".

Since each of the 56 characters is converted to 8 bits, the length of the message is $l = 448$. In step (a) of Section 4, we append "1". In step (b) we append 511 "0"s. In step (c) we append the 2-word representation of 448, i.e., hex `00000000 000001C0`. This gives n = 2.

The initial hex values of $\{H_i\}$are

$H_0$ = 67452301

$H_1$ = EFCDAB89

$H_2$ = 98BADCFE

$H_3$ = 10325476

$H_4$ = C3D2E1F0.

Start processing block 1. The words of block 1are

```
W[0]  = 61626364
W[1]  = 62636465
W[2]  = 63646566
W[3]  = 64656667
W[4]  = 65666768
W[5]  = 66676869
W[6]  = 6768696A
W[7]  = 68696A6B
W[8]  = 696A6B6C
W[9]  = 6A6B6C6D
W[10] = 6B6C6D6E
W[11] = 6C6D6E6F
W[12] = 6D6E6F70
W[13] = 6E6F7071
W[14] = 80000000
W[15] = 00000000.
```

The hex values of A,B,C,D,E after pass t of the "for t = 0 to 79" loop (step (d) of Section 7 or (c) of Section 8) are

| | A | B | C | D | E |
|---|---|---|---|---|---|
| t = 0: | 0116FC17 | 67452301 | 7BF36AE2 | 98BADCFE | 10325476 |
| t = 1: | EBF3B452 | 0116FC17 | 59D148C0 | 7BF36AE2 | 98BADCFE |
| t = 2: | 5109913A | EBF3B452 | C045BF05 | 59D148C0 | 7BF36AE2 |
| t = 3: | 2C4F6EAC | 5109913A | BAFCED14 | C045BF05 | 59D148C0 |
| t = 4: | 33F4AE5B | 2C4F6EAC | 9442644E | BAFCED14 | C045BF05 |
| t = 5: | 96B85189 | 33F4AE5B | 0B13DBAB | 9442644E | BAFCED14 |
| t = 6: | DB04CB58 | 96B85189 | CCFD2B96 | 0B13DBAB | 9442644E |
| t = 7: | 45833F0F | DB04CB58 | 65AE1462 | CCFD2B96 | 0B13DBAB |
| t = 8: | C565C35E | 45833F0F | 36C132D6 | 65AE1462 | CCFD2B96 |
| t = 9: | 6350AFDA | C565C35E | D160CFC3 | 36C132D6 | 65AE1462 |
| t = 10: | 8993EA77 | 6350AFDA | B15970D7 | D160CFC3 | 36C132D6 |
| t = 11: | E19ECAA2 | 8993EA77 | 98D42BF6 | B15970D7 | D160CFC3 |
| t = 12: | 8603481E | E19ECAA2 | E264FA9D | 98D42BF6 | B15970D7 |
| t = 13: | 32F94A85 | 8603481E | B867B2A8 | E264FA9D | 98D42BF6 |
| t = 14: | B2E7A8BE | 32F94A85 | A180D207 | B867B2A8 | E264FA9D |
| t = 15: | 42637E39 | B2E7A8BE | 4CBE52A1 | A180D207 | B867B2A8 |
| t = 16: | 66036329 | 42637E39 | ACB9EA2F | 4CBE52A1 | A180D207 |
| t = 17: | B59A89E4 | 66036329 | 5098DF8E | ACB9EA2F | 4CBE52A1 |
| t = 18: | 90B9433E | B59A89E4 | 5980D8CA | 5098DF8E | ACB9EA2F |
| t = 19: | DB5227E2 | 90B9433E | 2D66A279 | 5980D8CA | 5098DF8E |
| t = 20: | 91241034 | DB5227E2 | A42E50CF | 2D66A279 | 5980D8CA |
| t = 21: | 4C06BD64 | 91241034 | B6D489F8 | A42E50CF | 2D66A279 |
| t = 22: | 8665831E | 4C06BD64 | 2449040D | B6D489F8 | A42E50CF |
| t = 23: | 3F62D9EC | 8665831E | 1301AF59 | 2449040D | B6D489F8 |
| t = 24: | CD40E178 | 3F62D9EC | A19960C7 | 1301AF59 | 2449040D |
| t = 25: | D83E484E | CD40E178 | 0FD8B67B | A19960C7 | 1301AF59 |
| t = 26: | D70940FE | D83E484E | 3350385E | 0FD8B67B | A19960C7 |
| t = 27: | 39B6981B | D70940FE | B60F9213 | 3350385E | 0FD8B67B |
| t = 28: | 694303AE | 39B6981B | B5C2503F | B60F9213 | 3350385E |
| t = 29: | 8E08FD0A | 694303AE | CE6DA606 | B5C2503F | B60F9213 |
| t = 30: | FBFF1BA5 | 8E08FD0A | 9A50C0EB | CE6DA606 | B5C2503F |
| t = 31: | 8AB96092 | FBFF1BA5 | A3823F42 | 9A50C0EB | CE6DA606 |
| t = 32: | 4206057A | 8AB96092 | 7EFFC6E9 | A3823F42 | 9A50C0EB |
| t = 33: | 2CBCFC1A | 4206057A | A2AE5824 | 7EFFC6E9 | A3823F42 |
| t = 34: | 505759F3 | 2CBCFC1A | 9081815E | A2AE5824 | 7EFFC6E9 |
| t = 35: | 05BB8EC9 | 505759F3 | 8B2F3F06 | 9081815E | A2AE5824 |
| t = 36: | A0FC08A0 | 05BB8EC9 | D415D67C | 8B2F3F06 | 9081815E |
| t = 37: | 8664F5E1 | A0FC08A0 | 416EE3B2 | D415D67C | 8B2F3F06 |
| t = 38: | FE3D2A4F | 8664F5E1 | 283F0228 | 416EE3B2 | D415D67C |
| t = 39: | 07D02AA9 | FE3D2A4F | 61993D78 | 283F0228 | 416EE3B2 |
| t = 40: | 38D7321C | 07D02AA9 | FF8F4A93 | 61993D78 | 283F0228 |
| t = 41: | 1F3CA4C0 | 38D7321C | 41F40AAA | FF8F4A93 | 61993D78 |
| t = 42: | DF27AA0C | 1F3CA4C0 | 0E35CC87 | 41F40AAA | FF8F4A93 |
| t = 43: | 84E2DBA6 | DF27AA0C | 07CF2930 | 0E35CC87 | 41F40AAA |
| t = 44: | 8797EB77 | 84E2DBA6 | 37C9EA83 | 07CF2930 | 0E35CC87 |
| t = 45: | 9D220100 | 8797EB77 | A138B6E9 | 37C9EA83 | 07CF2930 |
| t = 46: | CB326B71 | 9D220100 | E1E5FADD | A138B6E9 | 37C9EA83 |
| t = 47: | 505DE66F | CB326B71 | 27488040 | E1E5FADD | A138B6E9 |
| t = 48: | FFDF8E6F | 505DE66F | 72CC9ADC | 27488040 | E1E5FADD |

| | | | | | |
|---|---|---|---|---|---|
| t = 49: | 47A17A6F | FFDF8E6F | D417799B | 72CC9ADC | 27488040 |
| t = 50: | 2C742CF4 | 47A17A6F | FFF7E39B | D417799B | 72CC9ADC |
| t = 51: | 692C82F3 | 2C742CF4 | D1E85E9B | FFF7E39B | D417799B |
| t = 52: | 741A7AEB | 692C82F3 | 0B1D0B3D | D1E85E9B | FFF7E39B |
| t = 53: | E89625B3 | 741A7AEB | DA4B20BC | 0B1D0B3D | D1E85E9B |
| t = 54: | BB527C29 | E89625B3 | DD069EBA | DA4B20BC | 0B1D0B3D |
| t = 55: | 609A8616 | BB527C29 | FA25896C | DD069EBA | DA4B20BC |
| t = 56: | 5E259CED | 609A8616 | 6ED49F0A | FA25896C | DD069EBA |
| t = 57: | FDCE04C4 | 5E259CED | 9826A185 | 6ED49F0A | FA25896C |
| t = 58: | 2A35958F | FDCE04C4 | 5789673B | 9826A185 | 6ED49F0A |
| t = 59: | 029A9DBB | 2A35958F | 3F738131 | 5789673B | 9826A185 |
| t = 60: | 651604AB | 029A9DBB | CA8D6563 | 3F738131 | 5789673B |
| t = 61: | 3F163F73 | 651604AB | C0A6A76E | CA8D6563 | 3F738131 |
| t = 62: | 60E30527 | 3F163F73 | D945812A | C0A6A76E | CA8D6563 |
| t = 63: | DA53F35E | 60E30527 | CFC58FDC | D945812A | C0A6A76E |
| t = 64: | 59F8E302 | DA53F35E | D838C149 | CFC58FDC | D945812A |
| t = 65: | BE75732C | 59F8E302 | B694FCD7 | D838C149 | CFC58FDC |
| t = 66: | 8D8DFD49 | BE75732C | 967E38C0 | B694FCD7 | D838C149 |
| t = 67: | 556247FC | 8D8DFD49 | 2F9D5CCB | 967E38C0 | B694FCD7 |
| t = 68: | C416C3E2 | 556247FC | 63637F52 | 2F9D5CCB | 967E38C0 |
| t = 69: | 64C244C9 | C416C3E2 | 155891FF | 63637F52 | 2F9D5CCB |
| t = 70: | B0DF5B97 | 64C244C9 | B105B0F8 | 155891FF | 63637F52 |
| t = 71: | 905723FE | B0DF5B97 | 59309132 | B105B0F8 | 155891FF |
| t = 72: | 49946022 | 905723FE | EC37D6E5 | 59309132 | B105B0F8 |
| t = 73: | B3A64DB3 | 49946022 | A415C8FF | EC37D6E5 | 59309132 |
| t = 74: | 281589BC | B3A64DB3 | 92651808 | A415C8FF | EC37D6E5 |
| t = 75: | 4623888D | 281589BC | ECE9936C | 92651808 | A415C8FF |
| t = 76: | 74EB04B7 | 4623888D | 0A05626F | ECE9936C | 92651808 |
| t = 77: | 035D4CD9 | 74EB04B7 | 5188E223 | 0A05626F | ECE9936C |
| t = 78: | B2BDD7D0 | 035D4CD9 | DD3AC12D | 5188E223 | 0A05626F |
| t = 79: | 1D750196 | B2BDD7D0 | 40D75336 | DD3AC12D | 5188E223. |

Block 1 has been processed. The values of $\{H_i\}$ are

$H_0$ = 67452301 + 1D750196 = 84BA2497

$H_1$ = EFCDAB89 + B2BDD7D0 = A28B8359

$H_2$ = 98BADCFE + 40D75336 = D9923034

$H_3$ = 10325476 + DD3AC12D = ED6D15A3

$H_4$ = C3D2E1F0 + 5188E223 = 155BC413.

Start processing block 2. The words of block 2 are

W[0] = 00000000
W[1] = 00000000
W[2] = 00000000

```
W[3]  = 00000000
W[4]  = 00000000
W[5]  = 00000000
W[6]  = 00000000
W[7]  = 00000000
W[8]  = 00000000
W[9]  = 00000000
W[10] = 00000000
W[11] = 00000000
W[12] = 00000000
W[13] = 00000000
W[14] = 00000000
W[15] = 000001C0.
```

The hex values of A,B,C,D,E after pass t of the "for t = 0 to 79" loop (step (d) of Section 7 or (c) of Section 8) are

| | A | B | C | D | E |
|---|---|---|---|---|---|
| t = 0: | D508E54E | 84BA2497 | 68A2E0D6 | D9923034 | ED6D15A3 |
| t = 1: | 42AE69CC | D508E54E | E12E8925 | 68A2E0D6 | D9923034 |
| t = 2: | 738C64E9 | 42AE69CC | B5423953 | E12E8925 | 68A2E0D6 |
| t = 3: | D5B4A0FE | 738C64E9 | 10AB9A73 | B5423953 | E12E8925 |
| t = 4: | 870F3C0B | D5B4A0FE | 5CE3193A | 10AB9A73 | B5423953 |
| t = 5: | 46574E97 | 870F3C0B | B56D283F | 5CE3193A | 10AB9A73 |
| t = 6: | 1405102F | 46574E97 | E1C3CF02 | B56D283F | 5CE3193A |
| t = 7: | 297306DF | 1405102F | D195D3A5 | E1C3CF02 | B56D283F |
| t = 8: | 30185CE2 | 297306DF | C501440B | D195D3A5 | E1C3CF02 |
| t = 9: | 10D7BA0C | 30185CE2 | CA5CC1B7 | C501440B | D195D3A5 |
| t = 10: | 0C28CF6B | 10D7BA0C | 8C061738 | CA5CC1B7 | C501440B |
| t = 11: | 6EABFEC0 | 0C28CF6B | 0435EE83 | 8C061738 | CA5CC1B7 |
| t = 12: | 7E85F170 | 6EABFEC0 | C30A33DA | 0435EE83 | 8C061738 |
| t = 13: | F964F1A3 | 7E85F170 | 1BAAFFB0 | C30A33DA | 0435EE83 |
| t = 14: | 26E19055 | F964F1A3 | 1FA17C5C | 1BAAFFB0 | C30A33DA |
| t = 15: | 156937E7 | 26E19055 | FE593C68 | 1FA17C5C | 1BAAFFB0 |
| t = 16: | 6295F273 | 156937E7 | 49B86415 | FE593C68 | 1FA17C5C |
| t = 17: | B81A706E | 6295F273 | C55A4DF9 | 49B86415 | FE593C68 |
| t = 18: | A5620A0D | B81A706E | D8A57C9C | C55A4DF9 | 49B86415 |
| t = 19: | 2DBC9CFF | A5620A0D | AE069C1B | D8A57C9C | C55A4DF9 |
| t = 20: | BF89C409 | 2DBC9CFF | 69588283 | AE069C1B | D8A57C9C |
| t = 21: | 239A6D9B | BF89C409 | CB6F273F | 69588283 | AE069C1B |
| t = 22: | ADEC9CD5 | 239A6D9B | 6FE27102 | CB6F273F | 69588283 |
| t = 23: | 1CDD463F | ADEC9CD5 | C8E69B66 | 6FE27102 | CB6F273F |
| t = 24: | E0DA5334 | 1CDD463F | 6B7B2735 | C8E69B66 | 6FE27102 |
| t = 25: | B947BDAB | E0DA5334 | C737518F | 6B7B2735 | C8E69B66 |
| t = 26: | AD4E620C | B947BDAB | 383694CD | C737518F | 6B7B2735 |
| t = 27: | CA67CF14 | AD4E620C | EE51EF6A | 383694CD | C737518F |
| t = 28: | FE343974 | CA67CF14 | 2B539883 | EE51EF6A | 383694CD |
| t = 29: | 7CFD680A | FE343974 | 3299F3C5 | 2B539883 | EE51EF6A |
| t = 30: | E4D7304C | 7CFD680A | 3F8D0E5D | 3299F3C5 | 2B539883 |

| | | | | | |
|---|---|---|---|---|---|
| t = 31: | A6FD2352 | E4D7304C | 9F3F5A02 | 3F8D0E5D | 3299F3C5 |
| t = 32: | C57DADCD | A6FD2352 | 3935CC13 | 9F3F5A02 | 3F8D0E5D |
| t = 33: | 5F146AB9 | C57DADCD | A9BF48D4 | 3935CC13 | 9F3F5A02 |
| t = 34: | 469DC798 | 5F146AB9 | 715F6B73 | A9BF48D4 | 3935CC13 |
| t = 35: | 03BCF3DA | 469DC798 | 57C51AAE | 715F6B73 | A9BF48D4 |
| t = 36: | F03F67BA | 03BCF3DA | 11A771E6 | 57C51AAE | 715F6B73 |
| t = 37: | 2E04E8C4 | F03F67BA | 80EF3CF6 | 11A771E6 | 57C51AAE |
| t = 38: | E8B3497E | 2E04E8C4 | BC0FD9EE | 80EF3CF6 | 11A771E6 |
| t = 39: | A9CE9B40 | E8B3497E | 0B813A31 | BC0FD9EE | 80EF3CF6 |
| t = 40: | F261BB65 | A9CE9B40 | BA2CD25F | 0B813A31 | BC0FD9EE |
| t = 41: | 42EF9DD9 | F261BB65 | 2A73A6D0 | BA2CD25F | 0B813A31 |
| t = 42: | B2F2664A | 42EF9DD9 | 7C986ED9 | 2A73A6D0 | BA2CD25F |
| t = 43: | 1291092A | B2F2664A | 50BBE776 | 7C986ED9 | 2A73A6D0 |
| t = 44: | 7C6AEF48 | 1291092A | ACBC9992 | 50BBE776 | 7C986ED9 |
| t = 45: | A9CB9DF6 | 7C6AEF48 | 84A4424A | ACBC9992 | 50BBE776 |
| t = 46: | C5F82E71 | A9CB9DF6 | 1F1ABBD2 | 84A4424A | ACBC9992 |
| t = 47: | 8868C238 | C5F82E71 | AA72E77D | 1F1ABBD2 | 84A4424A |
| t = 48: | B052F768 | 8868C238 | 717E0B9C | AA72E77D | 1F1ABBD2 |
| t = 49: | 61102AC0 | B052F768 | 221A308E | 717E0B9C | AA72E77D |
| t = 50: | 8BEE2FF1 | 61102AC0 | 2C14BDDA | 221A308E | 717E0B9C |
| t = 51: | 9E700133 | 8BEE2FF1 | 18440AB0 | 2C14BDDA | 221A308E |
| t = 52: | 877A43CD | 9E700133 | 62FB8BFC | 18440AB0 | 2C14BDDA |
| t = 53: | C4E901D6 | 877A43CD | E79C004C | 62FB8BFC | 18440AB0 |
| t = 54: | 2C7A07F0 | C4E901D6 | 61DE90F3 | E79C004C | 62FB8BFC |
| t = 55: | 67344973 | 2C7A07F0 | B13A4075 | 61DE90F3 | E79C004C |
| t = 56: | 7EBAEE45 | 67344973 | 0B1E81FC | B13A4075 | 61DE90F3 |
| t = 57: | EB9659B3 | 7EBAEE45 | D9CD125C | 0B1E81FC | B13A4075 |
| t = 58: | 0EBFB62A | EB9659B3 | 5FAEBB91 | D9CD125C | 0B1E81FC |
| t = 59: | 4DBF216A | 0EBFB62A | FAE5966C | 5FAEBB91 | D9CD125C |
| t = 60: | 08089F12 | 4DBF216A | 83AFED8A | FAE5966C | 5FAEBB91 |
| t = 61: | 601ABA34 | 08089F12 | 936FC85A | 83AFED8A | FAE5966C |
| t = 62: | E1685B50 | 601ABA34 | 820227C4 | 936FC85A | 83AFED8A |
| t = 63: | EC956F26 | E1685B50 | 1806AE8D | 820227C4 | 936FC85A |
| t = 64: | 6BED4126 | EC956F26 | 385A16D4 | 1806AE8D | 820227C4 |
| t = 65: | 96D6E5E6 | 6BED4126 | BB255BC9 | 385A16D4 | 1806AE8D |
| t = 66: | A5D83970 | 96D6E5E6 | 9AFB5049 | BB255BC9 | 385A16D4 |
| t = 67: | 74CCF6E4 | A5D83970 | A5B5B979 | 9AFB5049 | BB255BC9 |
| t = 68: | B9BDCA6D | 74CCF6E4 | 29760E5C | A5B5B979 | 9AFB5049 |
| t = 69: | 9526A197 | B9BDCA6D | 1D333DB9 | 29760E5C | A5B5B979 |
| t = 70: | A2E5A7C9 | 9526A197 | 6E6F729B | 1D333DB9 | 29760E5C |
| t = 71: | 3708B81B | A2E5A7C9 | E549A865 | 6E6F729B | 1D333DB9 |
| t = 72: | F27081EC | 3708B81B | 68B969F2 | E549A865 | 6E6F729B |
| t = 73: | 41DAEB9B | F27081EC | CDC22E06 | 68B969F2 | E549A865 |
| t = 74: | 4215A57B | 41DAEB9B | 3C9C207B | CDC22E06 | 68B969F2 |
| t = 75: | 2655C2D6 | 4215A57B | D076BAE6 | 3C9C207B | CDC22E06 |
| t = 76: | 11DC8A86 | 2655C2D6 | D085695E | D076BAE6 | 3C9C207B |
| t = 77: | 69364641 | 11DC8A86 | 899570B5 | D085695E | D076BAE6 |
| t = 78: | 0A6ED856 | 69364641 | 847722A1 | 899570B5 | D085695E |
| t = 79: | 4D974A4A | 0A6ED856 | 5A4D9190 | 847722A1 | 899570B5 |

Block 2 has been processed. The values of $\{H_i\}$ are

$H_0$ = 84BA2497 + 4D974A4A = D2516EE1

$H_1$ = A28B8359 + 0A6ED856 = ACFA5BAF

$H_2$ = D9923034 + 5A4D9190 = 33DFC1C4

$H_3$ = ED6D15A3 + 847722A1 = 71E43844

$H_4$ = 155BC413 + 899570B5 = 9EF134C8.

Message digest = D2516EE1 ACFA5BAF 33DFC1C4 71E43844 9EF134C8

## APPENDIX C. A THIRD SAMPLE MESSAGE AND ITS MESSAGE DIGEST

This appendix is for informational purposes only and is not required to meet the standard.

Let the message be the binary-coded form of the ASCII string which consists of 1,000,000 repetitions of "a".

Message digest = 3232AFFA 48628A26 653B5AAA 44541FD9 0D690603



* U.S. G.P.O.:1993-341-832:82039