

# SHA Hashing Notes

Mimanshu Maheshwari

Thursday 28<sup>th</sup> March, 2024, 01:46

# Contents

<b>1</b>	<b>SHA 256</b>	<b>3</b>
1.1	Introduction . . . . .	3
1.2	Implementation . . . . .	3
1.3	Preprocessing . . . . .	4

# List of Tables

1.1	Notation Reference . . . . .	3
-----	------------------------------	---

# Chapter 1

## SHA 256

### 1.1 Introduction

SHA256 is a 256 bits hash. Ment to provide 128 bits of security against collision attack.

### 1.2 Implementation

SHA256 operates in a manner of MD4, MD5 and SHA-1. The message to be hashed is

1. Padded with its length in such a way that the result is multiple of 512 bits long.
2. Parsed into 512 bits message blocks  $M^1, M^1, \dots, M^1$ ,
3. Message blocks are processed one block at a time: Beginning with a fixed initial hash value  $H^{(0)}$ , sequentially compute

$$H^{(i)} = H^{(i-1)} + C_{M^{(i)}}(H^{(i-1)})$$

where  $C$  is the SHA-256 *compression function* and  $+$  means word-wise  $\bmod 2^{32}$  addition.  $H^{(N)}$  is the **hash** of  $M$ .

SHA-256 operates on 512-bits *message block* and a 256-bits *intermediate hash value*. It essentially is a 256-bit cypher algorithm which encrypts intermediate hash value using the message block as key. Hence, their are two main components:

- Compression Function
- message schedule

Notation	Meaning
$\oplus$	Bitwise XOR
$\vee$	Bitwise AND
$\wedge$	Bitwise OR
$\neg$	Bitwise Complement
$+$	$\bmod 2^{32}$ addition
$R^n$	right shift by $n$ bits
$S^n$	right rotate by $n$ bits

Table 1.1: Notation Reference

All of the operators in 1.1 table act on 32-bit words.

The initial value of  $H^{(0)}$  is the following sequence of 32 bit words (which are obtained by taking the fractional parts of the square roots of the first eight primes.)

$$H_1^{(0)} = 6a09e667 \quad (1.1)$$

$$H_2^{(0)} = bb67ae85 \quad (1.2)$$

$$H_3^{(0)} = 3c6ef372 \quad (1.3)$$

$$H_4^{(0)} = a54ff53a \quad (1.4)$$

$$H_5^{(0)} = 510e527f \quad (1.5)$$

$$H_6^{(0)} = 9b05688c \quad (1.6)$$

$$H_7^{(0)} = 1f83d9ab \quad (1.7)$$

$$H_8^{(0)} = 5be0cd19 \quad (1.8)$$

### 1.3 Preprocessing

Computing the hash of message begins by padding the message:

1. Pad the message in usual way: Suppose the length of message  $M$ , in bits, is  $l$ . Append the bit "1" to the end of message, and then the  $k$  zero bits, where  $k$  is the smallest non-negative solution to the equation  $l + 1 + 1 \equiv 448 \pmod{512}$ . To this append the 64-bit block which is equal to the number  $l$  written in binary. For example, the (8-bit ASCII) message "abc" has length  $8 \cdot 3 = 24$  so it is padded with a one, then  $448 - (24 + 1) = 423$  zero bits, and then the length to become the 512-bit padded message:

$$011000010110001001100011 \underbrace{00 \dots 0}_{423-bits} \overbrace{00 \dots 011000}^{64-bits}$$