ВЫСОКОПРОИЗВОДИТЕЛЬНАЯ РЕАЛИЗАЦИЯ HMAC SHA-256 НА БАЗЕ FPGA

Корбут А.А.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Станкевич А. В. – канд. технич. наук

Хеш-функции предназначены для создания «отпечатков» или «дайджестов» для сообщений произвольной длины. Применяются в различных приложениях или компонентах, связанных с защитой информации. HMAC – код проверки подлинности сообщений, используется для проверки целостности информации.

Алгоритм SHA-256

Входное сообщение дополняется и делится на 512-битные блоки. В свою очередь каждый блок разделяется на шестнадцать 32-битных слов, после чего по формуле (1) генерируются 48 дополнительных слов[1].

$$s0 = (w[i-15] \gg 7) \oplus (w[i-15] \gg 18) \oplus (w[i-15]\ shr\ 3)$$
$$s1 = (w[i-2] \gg 17) \oplus (w[i-2] \gg 19) \oplus (w[i-2]\ shr\ 10)$$
$$w[i] = w[i-16] + s0 + w[i-7] + s1$$
(1)

где shr – логический сдвиг вправо.

Обработка одного блока алгоритма выполняется за 64 итерации алгоритма. Схема одной итерации приведена на рисунке 1.

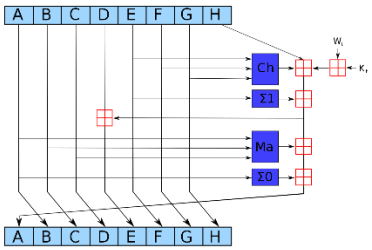

Рисунок 1 – Схема одной итерации алгоритма

A-H первоначально задаются константами. Впоследствии при обработке каждого последующего блока загружаются значениями, вычисленными на предыдущем этапе. W – слово, k – константа, зависящая от номера итерации. Формулы для вычисления Ch, ∑1, Ma, ∑0 вычисляются по формулам (2), (3), (4), (5) соответственно.

$$Ch = (E\ \&\ F) \oplus ((not\ E)\&G) \quad (2),$$
$$\sum 1 = (E \gg 6) \oplus (E \gg 11) \oplus (E \gg 25) \quad (3),$$
$$Ma = (A\&B) \oplus (A\&C) \oplus (B\&C) \quad (4),$$
$$\sum 0 = (\mathrm{A} \gg 2) \oplus (\mathrm{A} \gg 13) \oplus (\mathrm{A} \gg 22) \quad (5),$$

После обработки 64-итерациями алгоритма выходные значения A-H складываются с первоначальными значениями A-H, загруженными на первом раунде цикла. Хеш-значение получается после обработки всех блоков сообщения, путем склеивания последних сумм A-H (6).

$$\mathrm{hash} = \mathrm{h0} \parallel \mathrm{h1} \parallel \mathrm{h2} \parallel \mathrm{h3} \parallel \mathrm{h4} \parallel \mathrm{h5} \parallel \mathrm{h6} \parallel \mathrm{h7} \quad (6),$$

Алгоритм HMAC

Реализуется путем суммирования по модулю 2 ключа с константой i_pad, после чего аналогично складывается с входным сообщением, затем вычисляется первая hash-сумма. После этого ключ суммируется (сумма по модулю 2) с константой o_pad, результат суммируется с hash-суммой 1 и вычисляется путем обработки алгоритмом sha-256 hash-sum2[1]. Пример показан на рисунке 2.

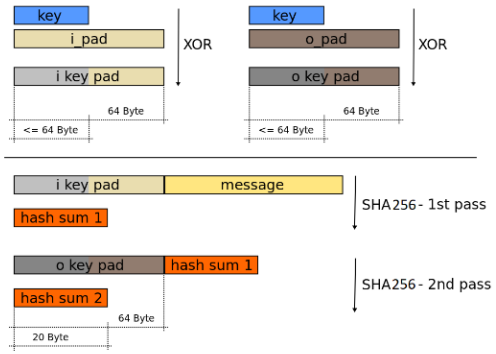

Рисунок 2 – Реализация HMAC-SHA256

Высокая производительность будет достигаться путем параллельно-итеративно-конвейерной реализации двух алгоритмов. Ключ для HMAC-алгоритма ограничим 512 битами, что позволит обеспечить достаточную надежность и уменьшит затраты ресурсов ПЛИС.

Значения ключа и сообщения будем накапливать в блочной памяти (BRAM). Для их приема задействуем общий 64-битных вход. Подачей на вход BRAM управляет демультиплексор. Внутри блока «Block divider» после накопления ключа в последующих тактах произойдет сложение с константой i_pad и выдача 512-битного блока на обработку в экспандер. Одновременно с этим память будет продолжать накопление сообщения. После окончании приема сообщения, оно также будет разделено на 512- битные блоки и подано на обработку алгоритмом SHA-256. Первое вычисленная хеш-сумма вернется в блок HMAC для повторного суммирования с константой o_pad, после чего будет произведено вычисление нового хеш-значения, являющимся окончательным. Описанная схема представлена на рисунке 3.

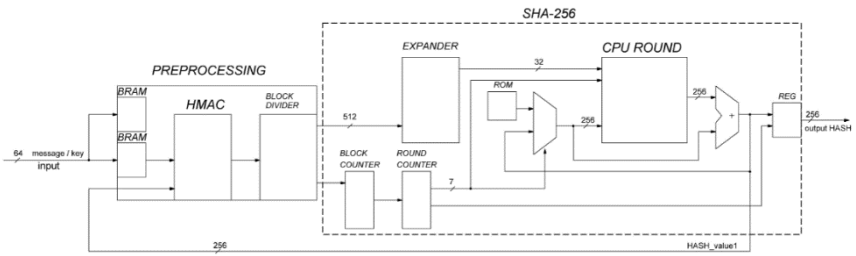

Рисунок 3 – Схема IP-ядра

Также была разработана ускоренная схема для расчета 64 раундов алгоритма SHA-256 с предварительным расчетом переменных[2]. Данная схема использует трехвходовой сумматор с ускоренным переносом, основанный на базовых элементах LUT6_2 и CARRY_4[3].

Список использованных источников:

1. D. Eastlake, T. Hansen, RFC 4634 US Secure Hash Algorithms (SHA and HMAC-SHA): Federal Information Processing Standard / AT&T Labs – USA, 2006.
2. An ASIC Design for a High Speed Implementation of the Hash Function SHA-256 / GLSVLSI'04// Boston, Massachusetts, USA., April 26–28, 2004. – P. 421-425.
3. Ternary Adder IP Cores / Martin Kumm, Jens Willkomm // 2013. – P. 1-4.