Bitcoin calculator on fpga

Bruno Pujos

14/01/2015

Introduction

- Bitcoin is a virtual crypto currency introduced in 2008-2009.
- Based on the bruteforce of a SHA-256.
- For fast computation, use of FPGA/ASIC.
- FPGA: Field-Programmable Gate Array.
- ASIC: Application-Specific Integrated Circuit.

Introduction

- Goal of this project is to implement a Bitcoin calculator on FPGA.
- Don't know how Bitcoin works.
- Don't know how SHA-256 works.
- Never done any verilog/FPGA.
- Hmm. . . Seems fun, let's go!

Plan

1. Bitcoin

Bitcoin

- Crypto-currency (decentralized virtual currency).
- Peer-to-peer.

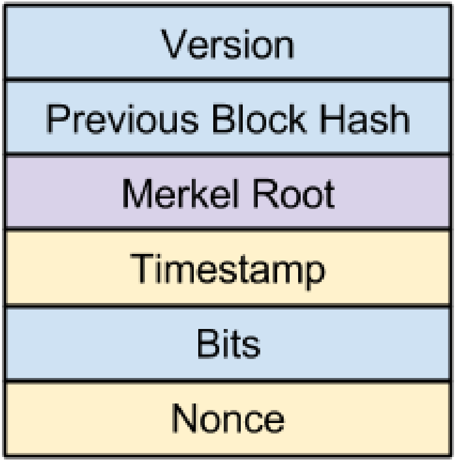
The block chain

- It is a public record of bitcoin transactions.
- Each ~10 minutes a block is added to the chain.
- Each block must be linked to the others.
- To link blocks, bitcoin puts the hash of the previous block into the header of the next.

Mining

- Mining is basically validating the block chain.
- To validate a block, the miners have to find a header that has a hash inferior to a certain value.
- The value which as to be inferior is called the difficulty target.
- hash(header) <= target.
- The hash is a double sha256: sha256(sha256(header)).

Bitcoin header

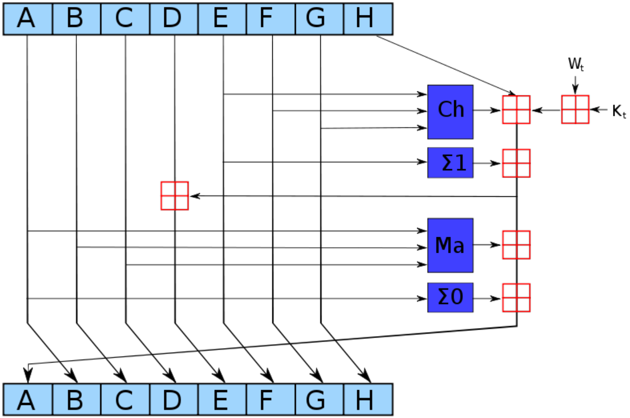


Plan

2 SHA-256

SHA-256

- Secure Hash Algorithm.
- Cryptographic hash functions.
- Part of the SHA-2 set designed by NSA.
- Code really simple.


A B C D E F G H
Ch
Σ1
Ma
Σ0
Wt
Kt
A B C D E F G H

Algorithm

- Preprocessing
- Computation

Preprocessing

- Pad the message to hash to be a multiple of 512 bits in length.
- Add a 1, enough 0 and the size of the payload (on 64 bits).
- Example: "abc" become:
- "616263" + "80" + "00" * 52 + "0000000000000018"
- Then split the result into 512 bits blocks.

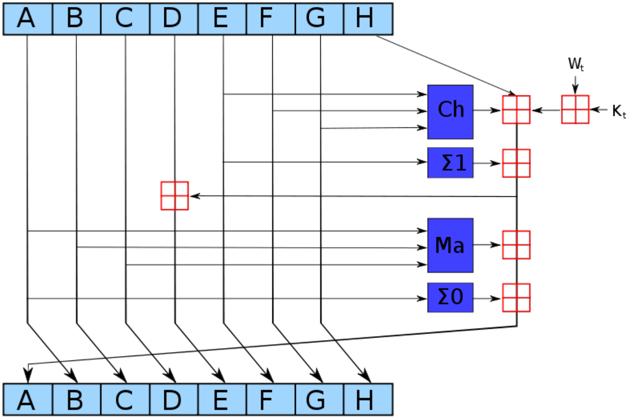
Computation

- We have an initial state defined by the standard composed of 8 32bits value.
- For each block of 512 bits we will do an operation which will change the state.
- At the end of algorithm, our hash is our state.

Computation: per block

- Still really simple.
- 64 iterations for each block.
- Each iteration modifies the state, everything is shifted except two values which are modified (differently).
- Modifications are computed from constants, depending of the state and depending of the value in the block.

Computation: per block



SHA-256: operations

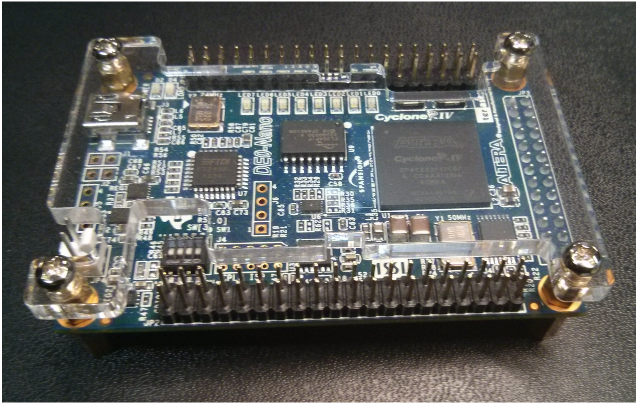
- Really simple operations:
  - Xor
  - Add
  - Shift
  - Ror
- Really simple code, why not do it in hardware?

3 FPGA implementation

FPGA

- ALTERA Cyclone IV: EP4CE22
- 22320 Logic elements
- Code in verilog
- Software: iverilog & Quartus

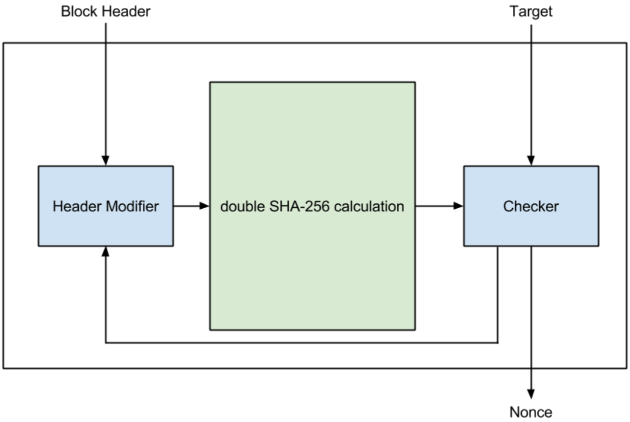
ALTERA Cyclone IV



FPGA programming

- Size & Response time
- An asynchronous circuit is a sequential digital logic circuit which is not governed by a clock circuit or global clock signal. (Wikipedia)
- A synchronous circuit is a digital circuit in which the parts are synchronized by a clock signal. (Wikipedia)

Global design



SHA-256 implementation: Asynchronous

- SHA-256 is really simple, the only part where we could need a clock is because we don't know the size of our input…
- With bitcoin, the size is always the same.
- It can be implemented as asynchronous.
- This means we can be really fast!
- 1 clock cycle = 1 SHA-256

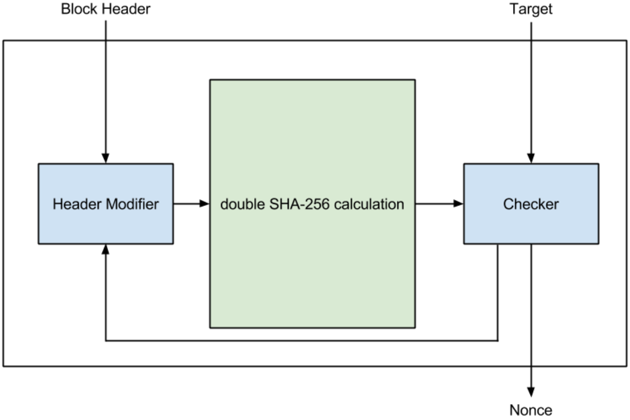
SHA-256: Asynchronous problem

- I wrote a simple round for SHA-256.
- Working in simulation but. . .
- 26000 number of logics elements.
- The Cyclone IV as only 22320 LE.
- Fail :(

SHA-256 implementation: synchronous

- Synchronous implementation is small enough.
- Much much slower.
- As bitcoin uses a double SHA-256 I need three rounds:
  - 2 for the first 1024 bits of the header.
  - 1 for the result of the 2 first.
- Still small enough but 400 clock cycles for 1 double SHA. (28MHz)

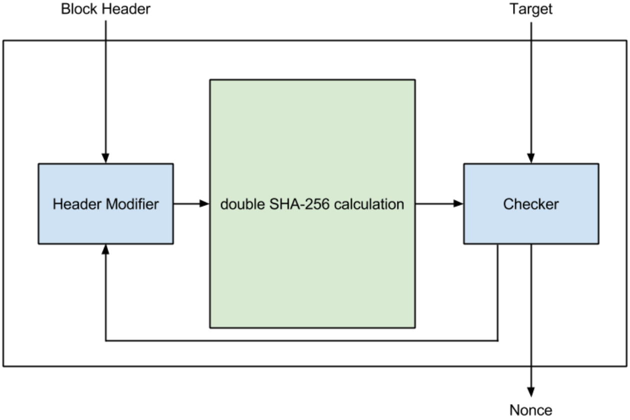
Global design



Header implementation

- In theory three fields can change: Timestamp, Merkel Root & Nonce.
- Timestamp: condition depending of the previous block timestamp and the current timestamp, means a lot of input necessary.
- Merkel Root: ”Complex” calcul for recalculating the whole tree.
- Nonce: Easy, just increment.

Global design



Conclusion: Results

- In python: 59000 per second (on my computer).
- On cyclone IV: 388000 per second.
- ASIC Red Fury Bitcoin Miner: 2.5GHs ($35).

Conclusion: Possible ameliorations

- When we change only the nonce the first of the three stages of the double SHA-256 is always the same. Huge time improvement.
- With the number of logic elements I used, I could put another double SHA-256. Twice faster.
- The code can be optimized. Space improvement.
- Handle more header change. I/O takes a lot of time, we reduce them but costly in space.
- Linear change of the header is maybe not the best idea.
- Handle the bitcoin protocol in hardware. Timing amelioration but cost LE.
- ASIC.

Conclusion

Thanks to @Ptishell

git.lse.epita.fr/users/pujos_b/fpga_bitcoin.git

Links

- http://www.righto.com/2014/02/bitcoin-mining-hard-way-algorithms.html
- http://www.righto.com/2014/09/mining-bitcoin-with-pencil-and-paper.html
- https://bitcoin.org/en/developer-guide
- http://csrc.nist.gov/groups/STM/cavp/documents/shs/sha256-384-512.pdf