Lecture 14
Class Project Introduction

Xuan 'Silvia' Zhang

Washington University in St. Louis

http://classes.engineering.wustl.edu/ese461/

Final Class Project

- Goal: <u>learn by doing</u>
  - account for 35% of grade
  - work in teams of 3
  - choose from two project topics
  - optimize design to meet/exceed performance goals
  - a custom designed IC chip as the end result

- Evaluation
  - completion of the design flow
  - performance achieved
  - techniques applied
  - presentation
  - report

Evaluation

- Completion of the design flow
  - functional simulation with behavior Verilog
  - logic synthesis
  - place and route
  - verification simulation with post-P&R netlist

- Performance achieved
  - meet basic specification of clock frequency and power budget
  - if exceed basic requirement, bonus points will be allocated according to performance ranking

Evaluation

- Techniques applied
  - design optimization techniques
  - e.g. pipelining, parallel units, memory buffer
- Presentation
  - last week of the class
  - every team member has to participate
- Report
  - submit a single report as a team
  - clearly specify individual contribution

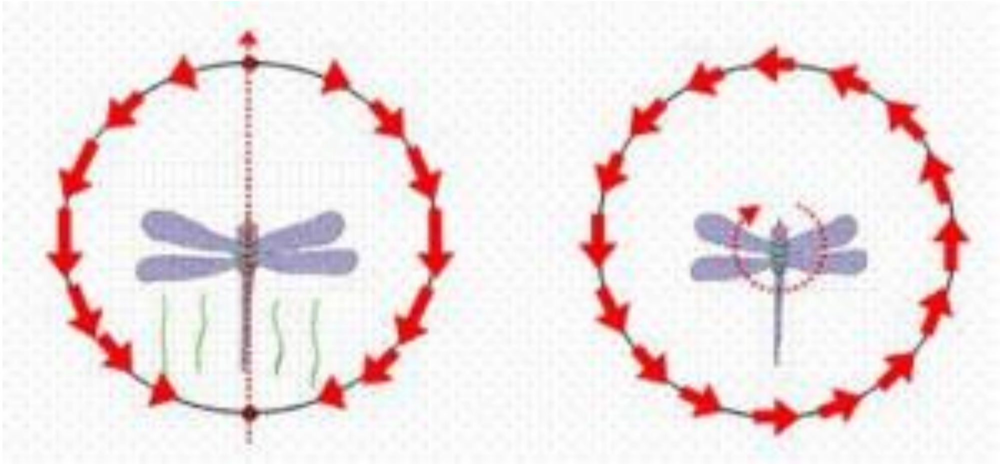
Choose Between Two Topics

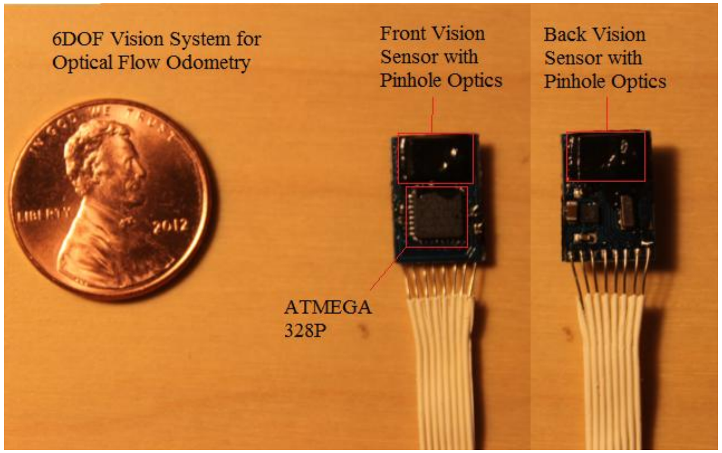
- Optical flow accelerator
  - motion estimation algorithm
  - inspired by biology
  - used in robotic application

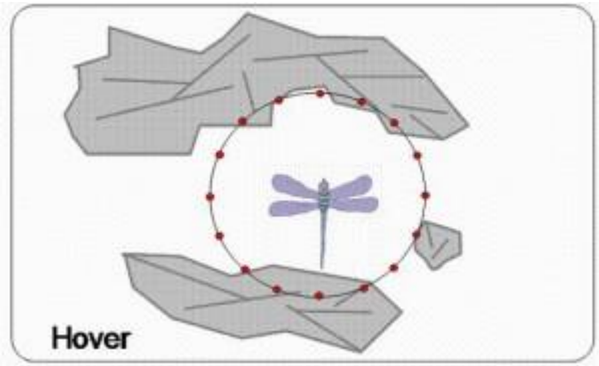
- Bitcoin hashing accelerator
  - a novel cryptocurrency
  - use proof-of-work to verify transactions

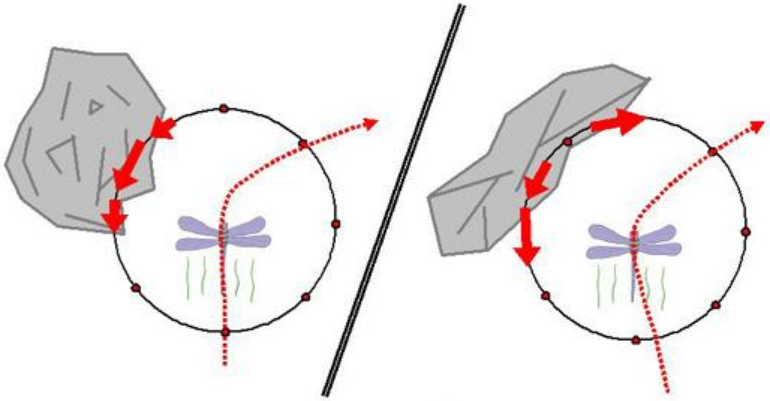
Optical Flow

- Insect use optical flow for navigation





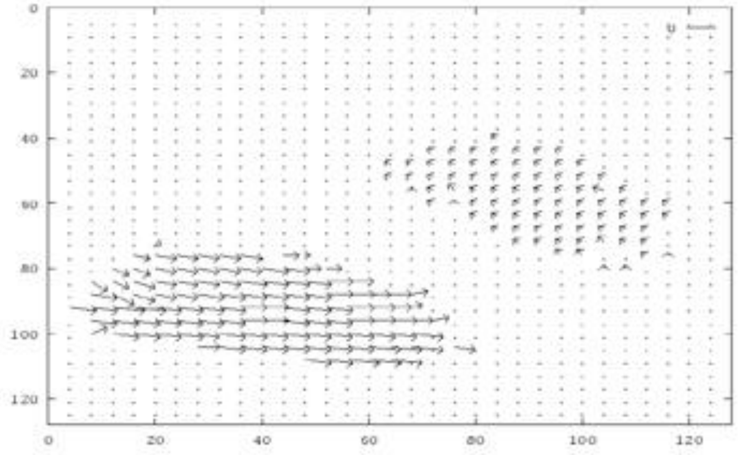




Saccade away from obstacles

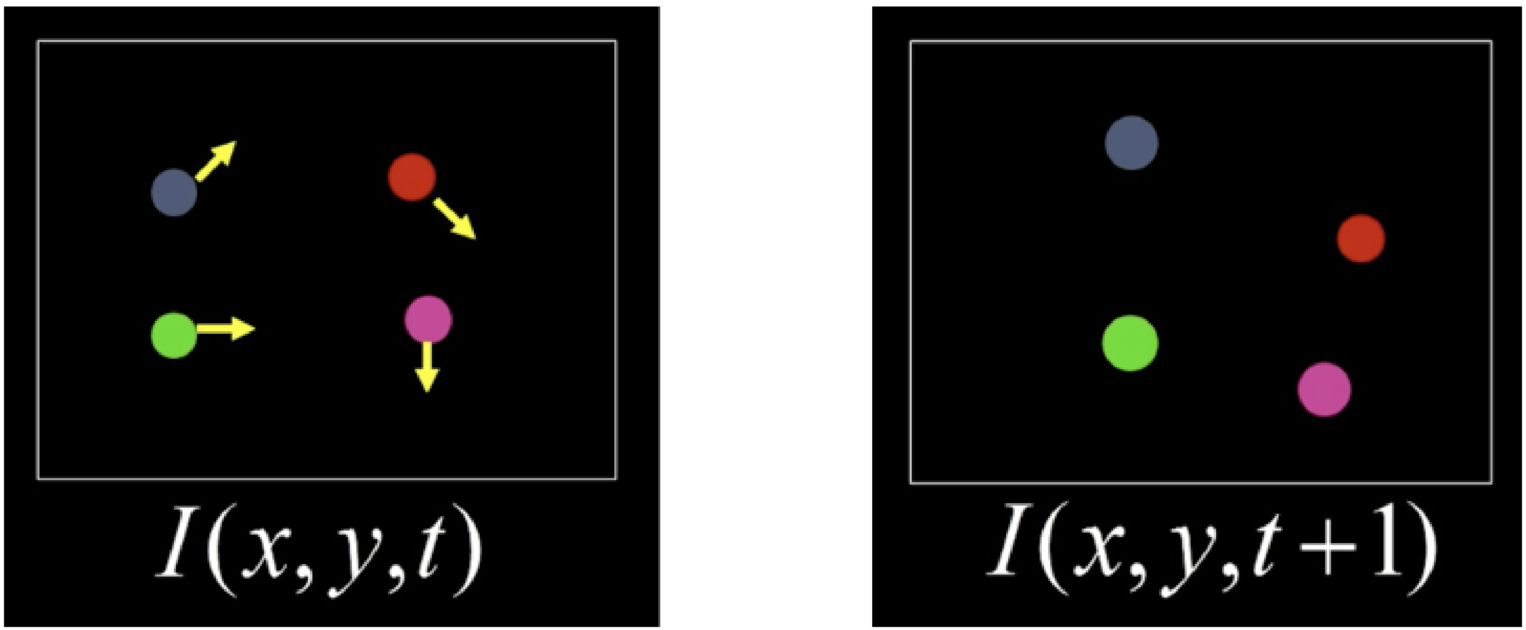
Motion Estimation

- Find the motion vector(u,v) between two frames



Problem Definition

- How to find the pixel motion from I(x,y,t) to I(x,y,t+1)?



Lucas-Kanade Method

- Assumptions
  - brightness constancy: a point in I(x,y,t) is same as I(x',y',t+1)
  - small motion: points do not move too far
  - small region moving together: approximately constant moving within a neighborhood of the point p

Lucas-Kanade Method

- Brightness constancy

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t)$$

- Small motion: (Taylor series expansion)

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t + \text{H.O.T.}$$

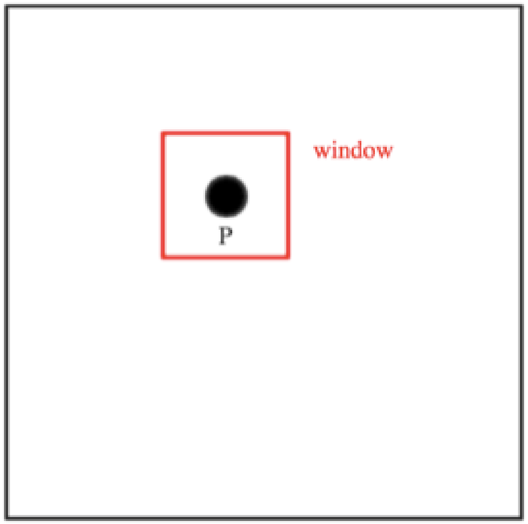
$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0$$

$$\frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t}\frac{\Delta t}{\Delta t} = 0$$

$$\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0 \Rightarrow I_x V_x + I_y V_y = -I_t$$

Windowing in Optical Flow

- Nearby region moving together
  - equation can be assumed to hold for all pixels within a window centered at p

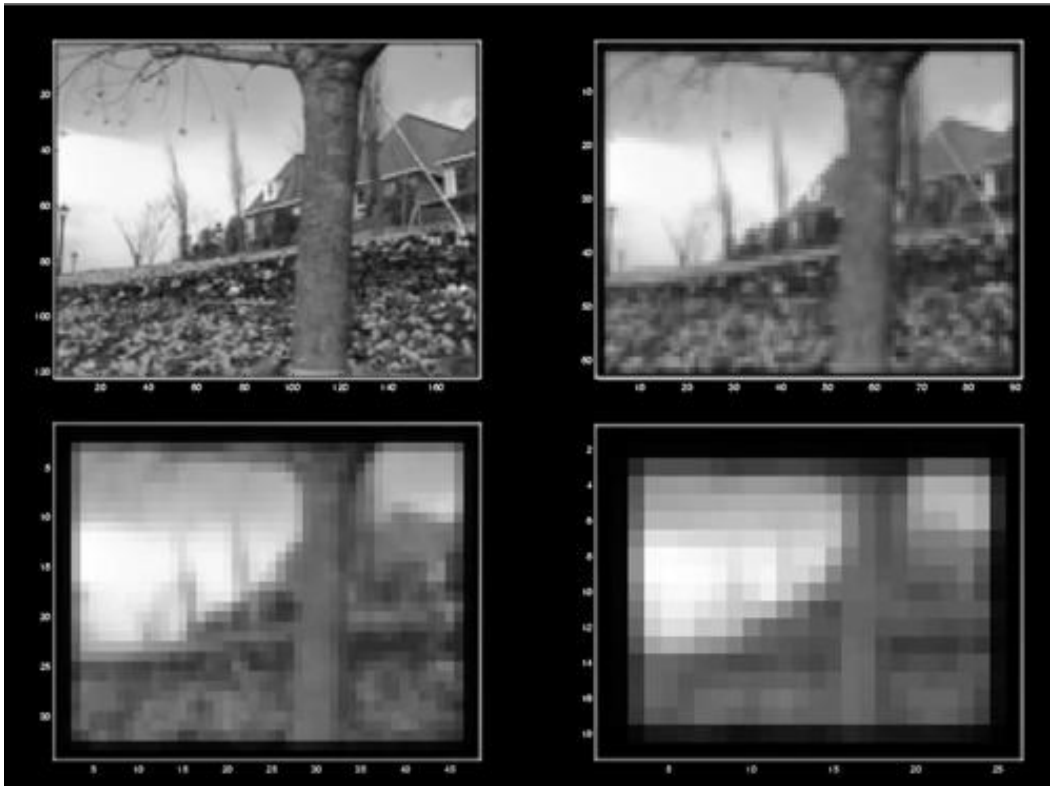


$$I_x(q_1)V_x + I_y(q_1)V_y = -I_t(q_1)$$
$$I_x(q_2)V_x + I_y(q_2)V_y = -I_t(q_2)$$
$$\vdots$$
$$I_x(q_n)V_x + I_y(q_n)V_y = -I_t(q_n)$$

  - q_1, q_2, …., q_n are the pixels inside the window —> moving together

Errors in Lucas-Kanade

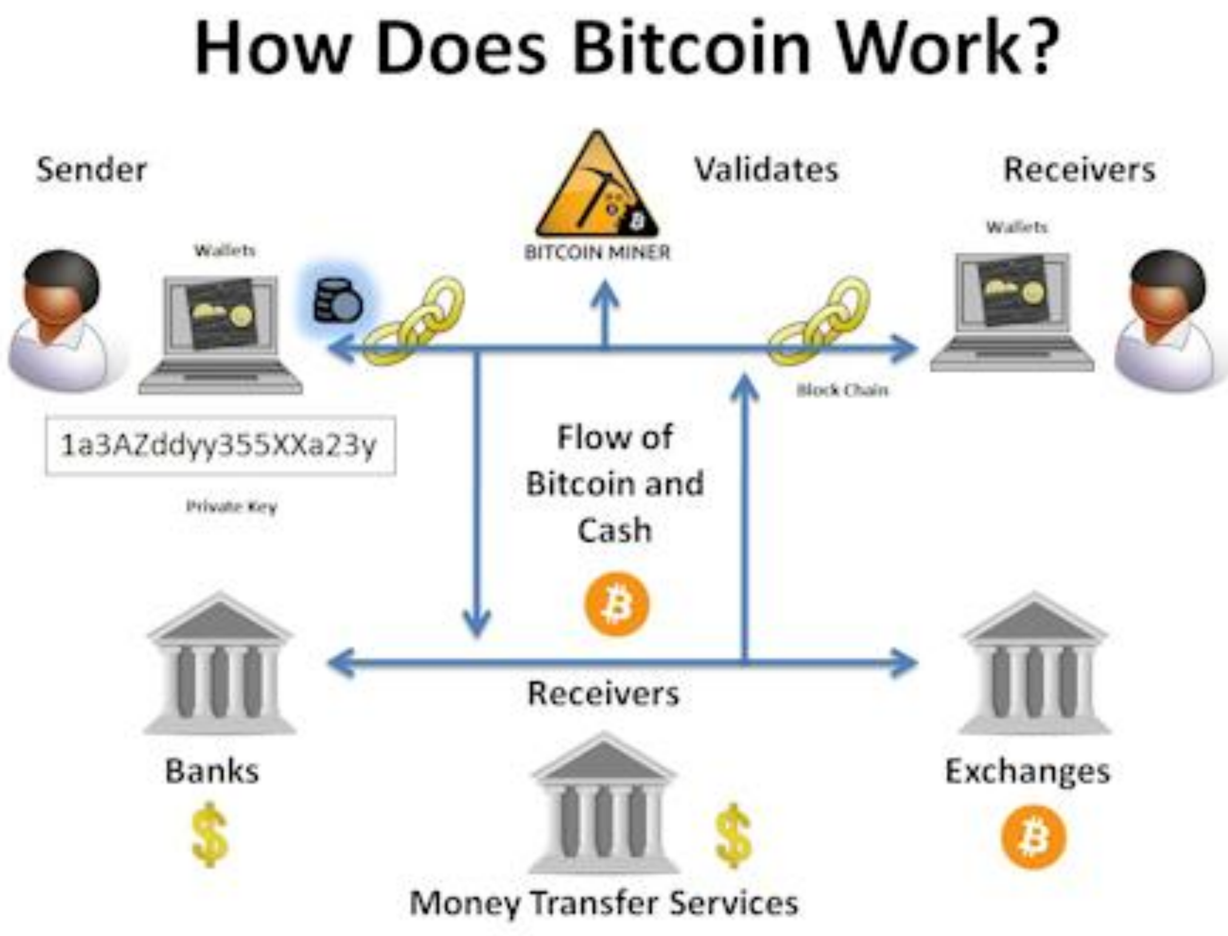
- Large motion violates assumption
  - reduce the resolution



Moving can be limited in one pixel change in the lowest resolution image

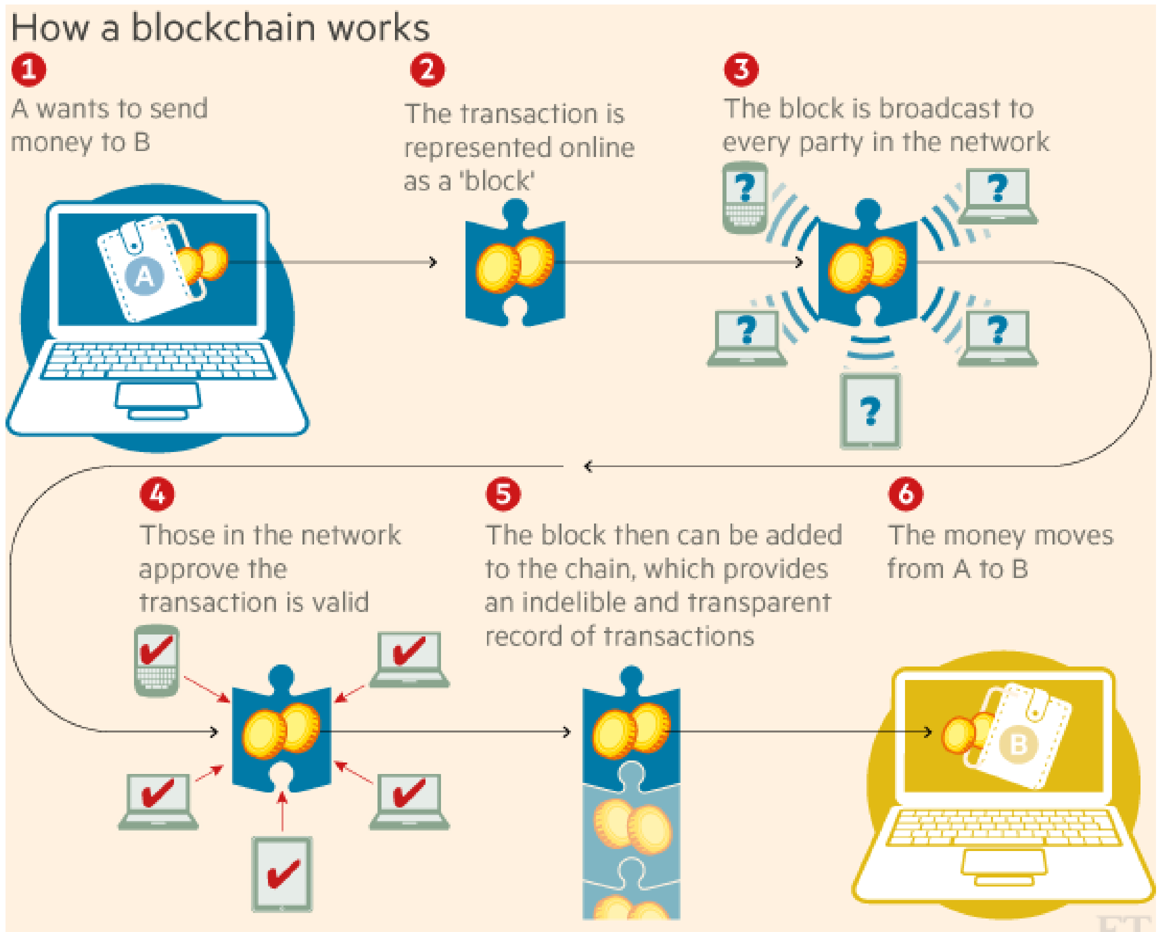
Bitcoin

- A cryptocurrency based on distributed consensus



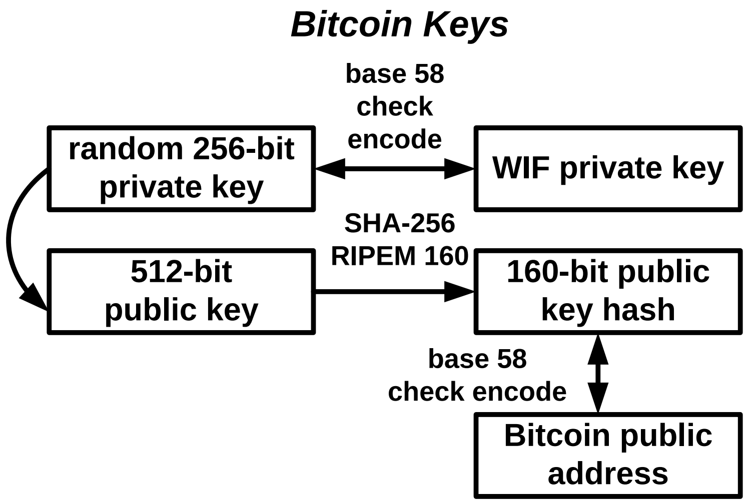
Blockchain

- A distributed public ledger maintained by a peer-to-peer network



Cryptography Used in Transaction

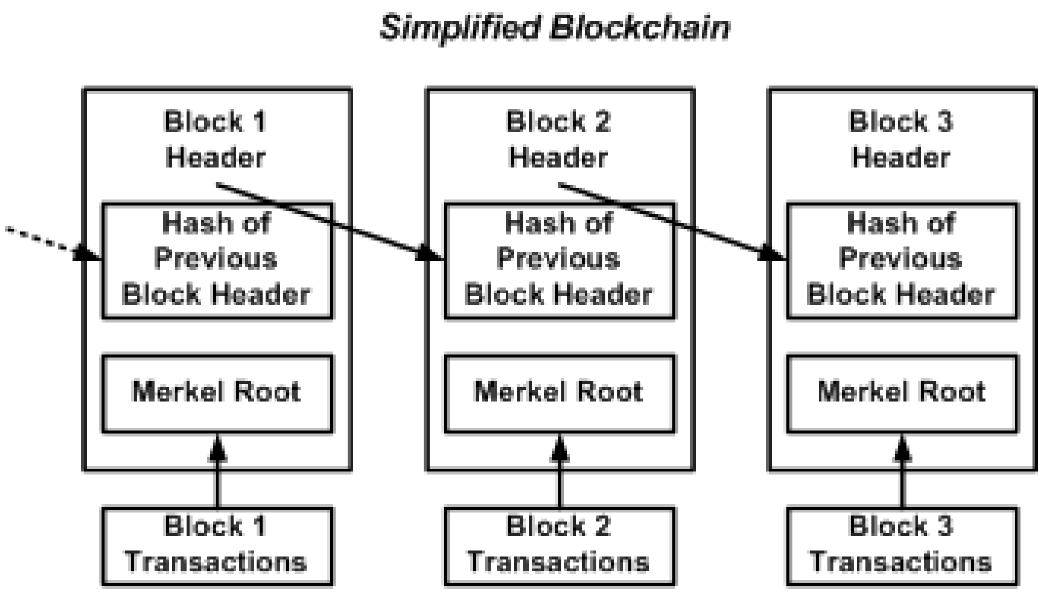
- Address generation

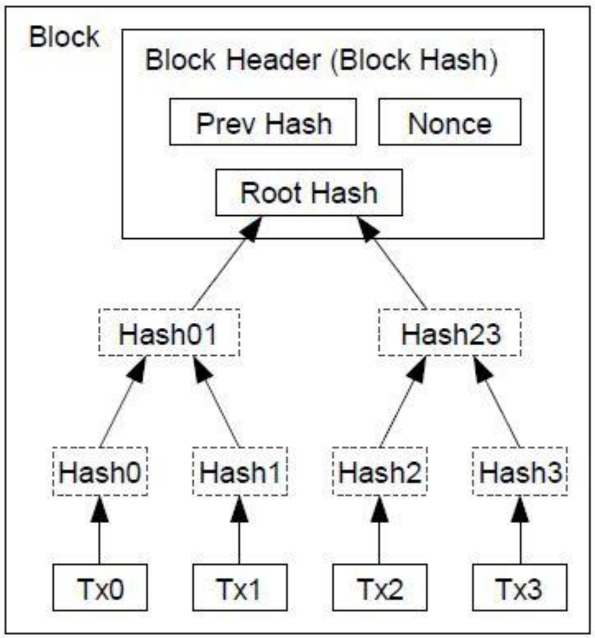


- Digital signature
  - elliptic curve digital signature algorithm

Bitcoin Mining

- Solve a "very hard" crypto puzzle
- Blockchain data structure

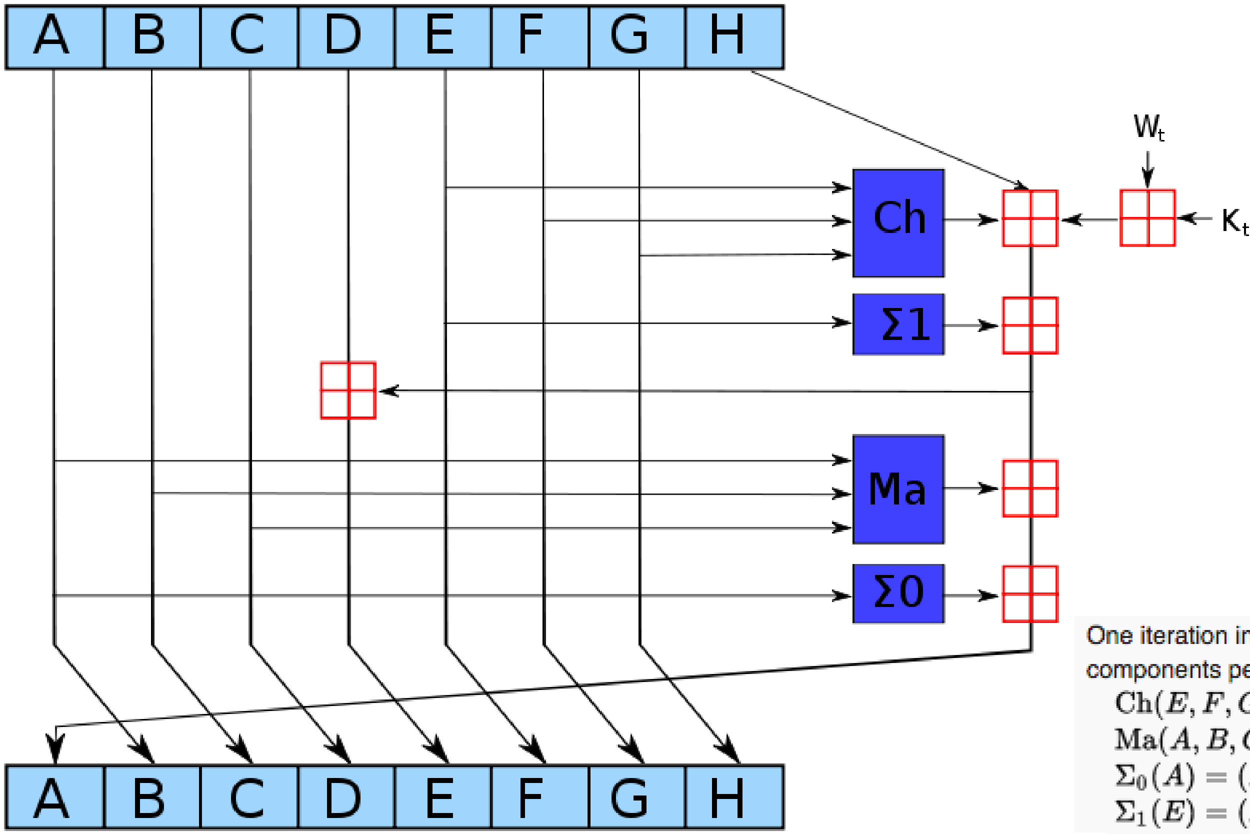




- Double SHA-256 hash
  - find a nounce that results in a hash with certain number of preceding zeros

SHA-256

- Secure Hash Algorithm (SHA)
  - hash function designed by NSA
  - SHA-256: run 64 rounds of iteration

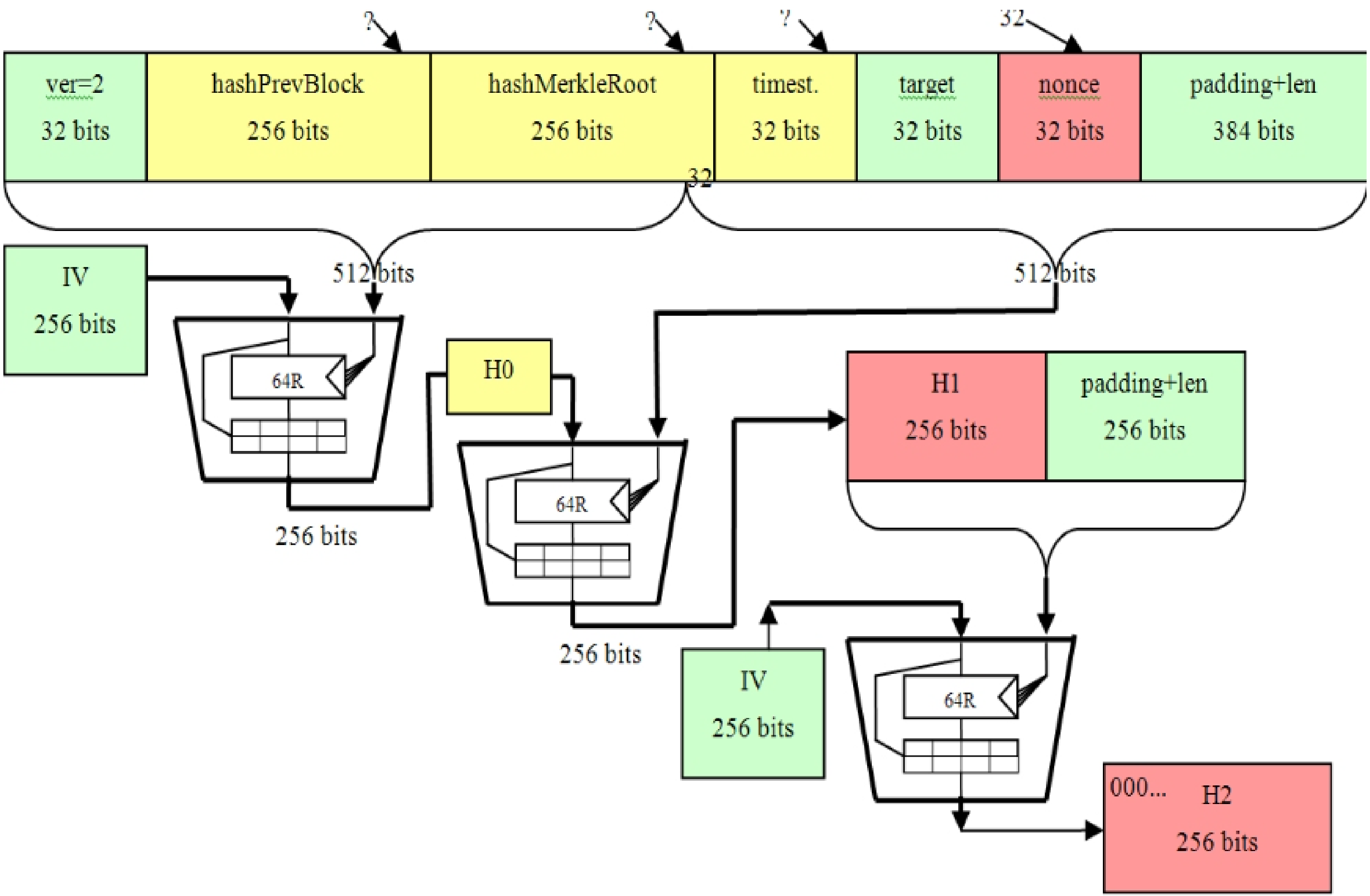


One iteration in a SHA-2 family compression function. The blue components perform the following operations:

$$\mathrm{Ch}(E,F,G) = (E \land F) \oplus (\neg E \land G)$$
$$\mathrm{Ma}(A,B,C) = (A \land B) \oplus (A \land C) \oplus (B \land C)$$
$$\Sigma_0(A) = (A \ggg 2) \oplus (A \ggg 13) \oplus (A \ggg 22)$$
$$\Sigma_1(E) = (E \ggg 6) \oplus (E \ggg 11) \oplus (E \ggg 25)$$

The red ⊞ is addition modulo 2^{32}.

Proof-of-Work Using Double SHA-256



Proof-of-Work Using Double SHA-256

Field	Size	Description
version	32 bits	Version of the Bitcoin software version creating this block
hashPrevBlock	256 bits	Hash of the previous block considered as valid in the Bitcoin network (most of the time there is only one candidate)
hashMerkleRoot	256 bits	Here a set of recent yet unconfirmed Bitcoin transactions are hashed into one single value on 256 bits = the Merkle Root
timestamp	32 bits	Current timestamp in seconds since 1970-01-01 00:00 UTC
target	32 bits	The current Target represented in a compact 32 bit format
nonce	32 bits	Nonce chosen by the miner, typically goes from 0x00000000 to 0xFFFFFFFF until the CISO puzzle is solved
padding + len	384 bits	standard fixed SHA256 padding on 384 bits for Len=640 bits

Timeline

- Week 8
  - introduction
- Week 9
  - release specification
  - form teams
  - select topic
- Week 9-11
  - develop and debug behavioral code
- Week 12-14
  - design flow and optimization
- Week 15
  - presentation
  - final report due on 12/12 (Monday)

Assignment

- Form a 3-member team
- Pick the topic of your choice
  - optical flow vs. bitcoin hashing
- Email me (cc the TA) your decision as a team
  - due 10/24 (Monday) by 5pm

Reference

- https://en.wikipedia.org/wiki/Lucas%E2%80%93Kanade_method
- https://www.mathworks.com/help/vision/ref/opticalflowlk-class.html
- http://docs.opencv.org/trunk/d7/d8b/tutorial_py_lucas_kanade.html
- http://www.righto.com/2014/02/bitcoins-hard-way-using-raw-bitcoin.html
- http://www.righto.com/2014/02/bitcoin-mining-hard-way-algorithms.html
- https://www.researchgate.net/figure/258144528_fig8_Fig-9-Key-in-the-first-16-rounds-out-of-64-in-each-computation-and-their-provenance

Questions?

Comments?

Discussion?