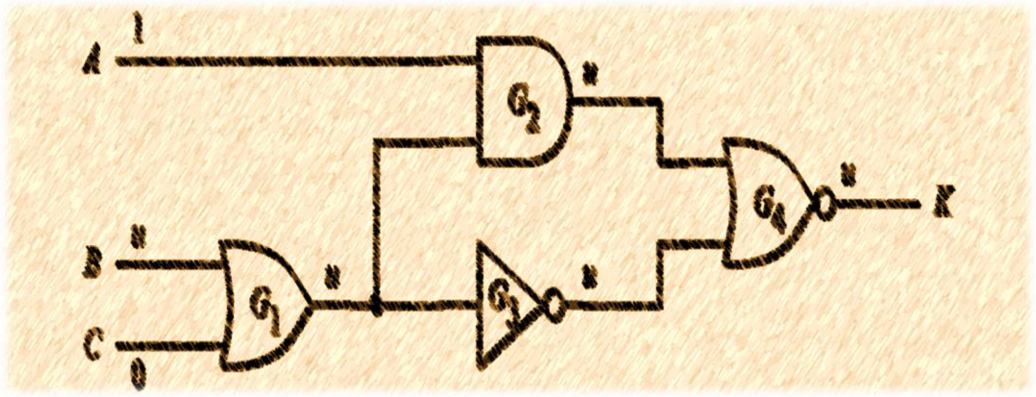
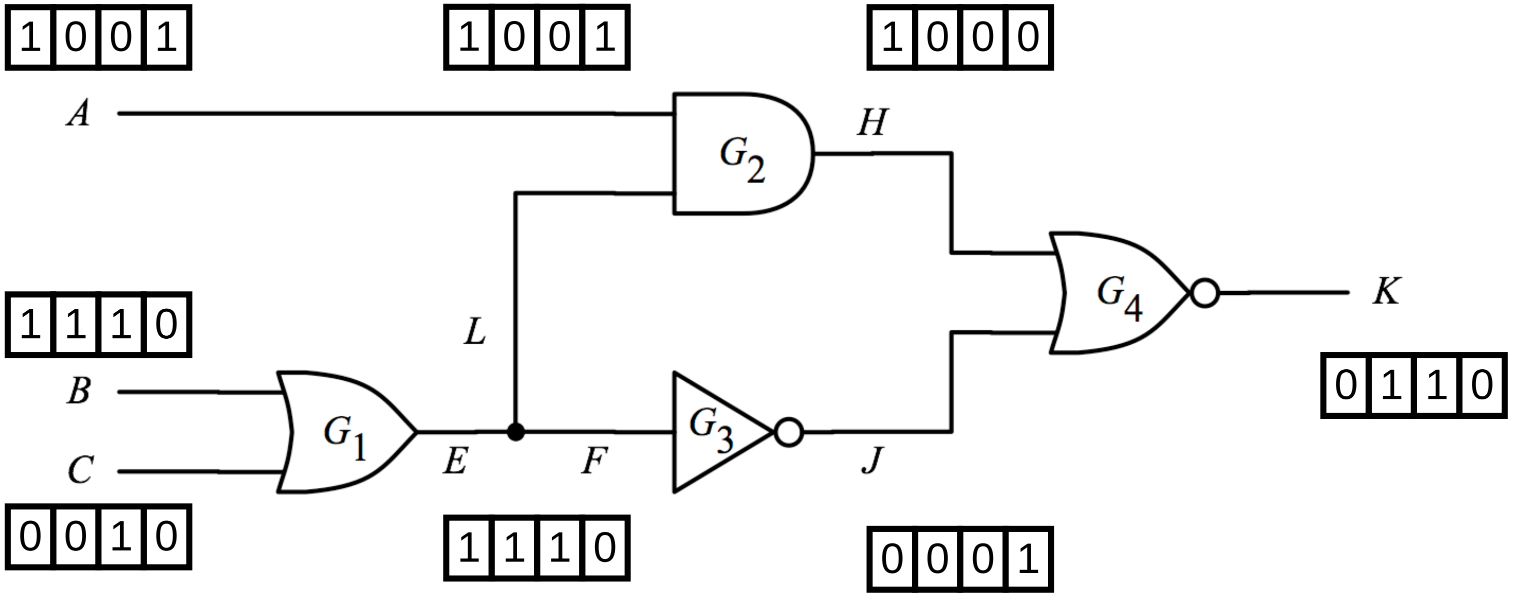
Logic Simulation

- Introduction
- Simulation Models
- **Logic Simulation Techniques**
  - Compiled code simulation
  - Event driven simulation
  - **Parallel Simulation**
- Issues of Logic Simulations
- Conclusions



Parallel Simulation

- Pack *W* patterns into a single word (*W* = word size)
  - Simultaneously evaluate a gate with *W* patterns
- Exploit parallelism of *bit-wise logic* operation
- Example: *W* = 4 (WWW Fig 3.8)
  - Consider only binary logic (no *u* or *z*)



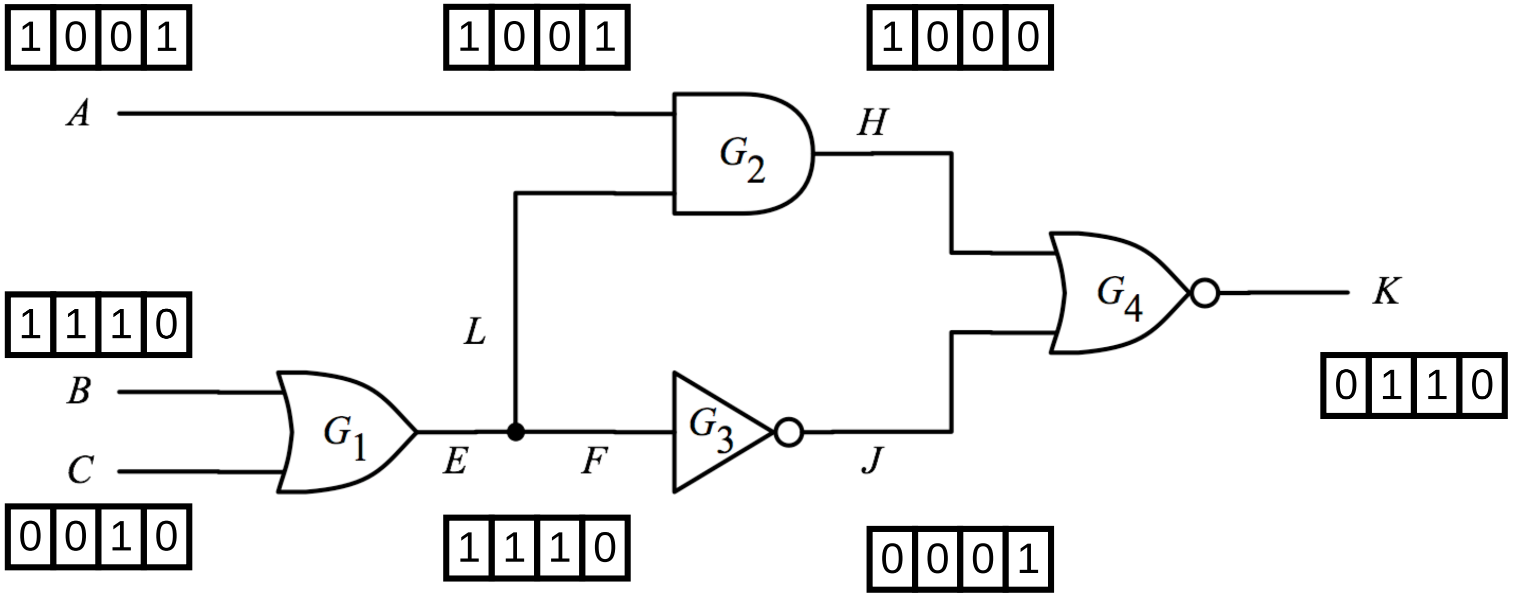
Quiz

Q: what is the speedup of parallel simulation? (*W* = CPU size)

A. *W*

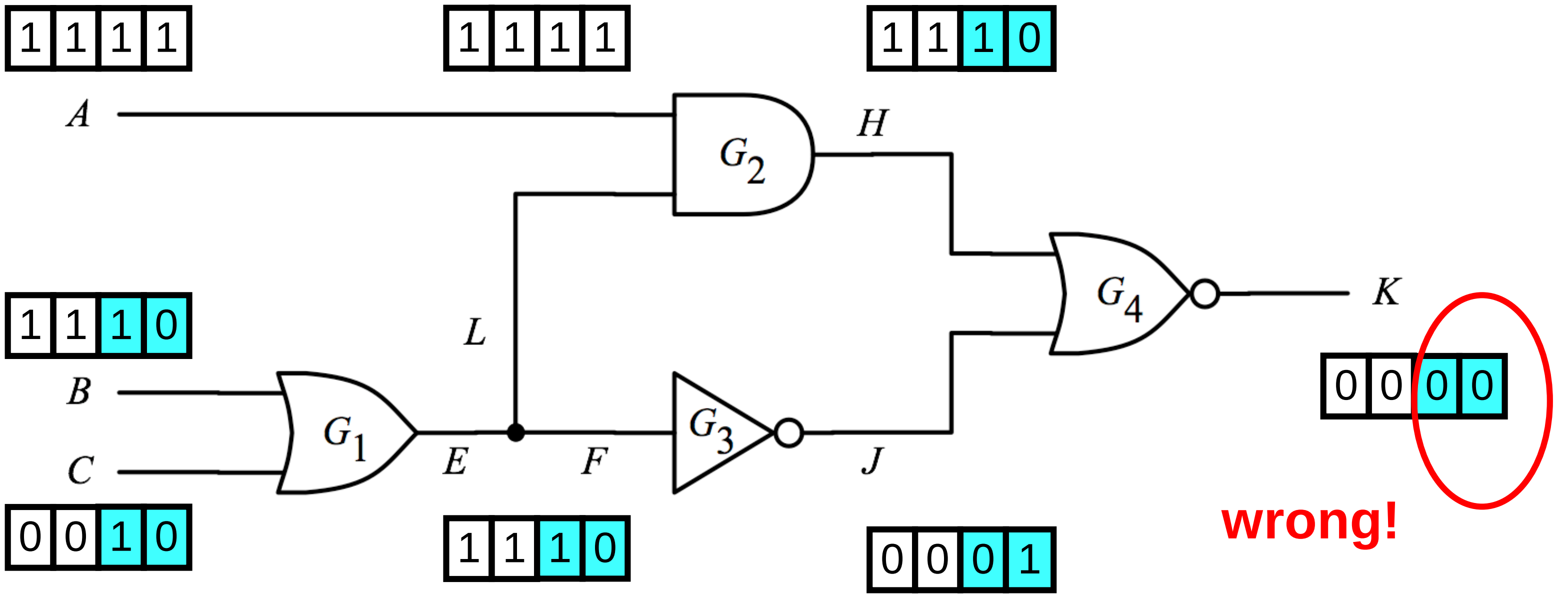
B. W^2

C. *No speedup*



How about Ternary Logic?

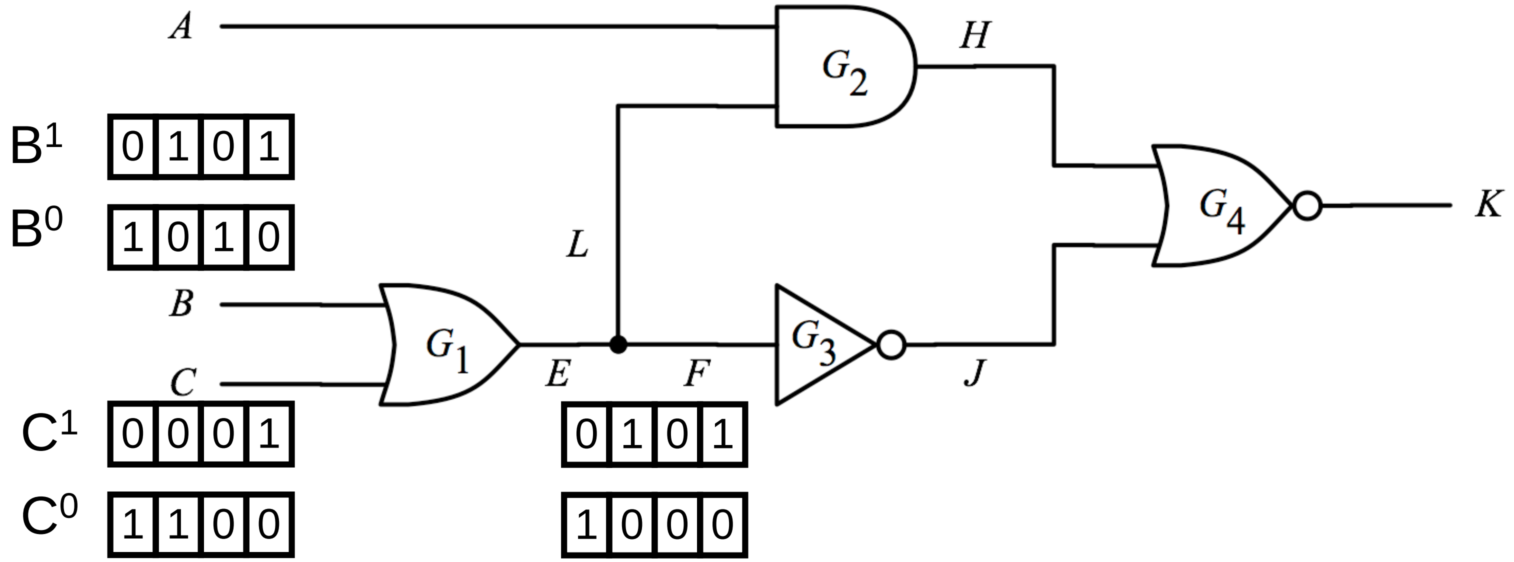
- Simple idea: use 2 bits to represent ternary logic
- Simple encoding method
  - Logic one=11, Logic zero=00, Unknown =10
- Works fine with OR/AND …. but wrong with inversion



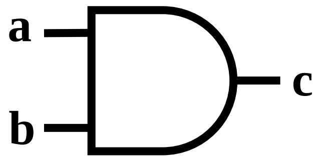
- Can we use both 10 and 01 as *u*? No

Sol: Improved Encoding Method

- Two words to encode single *A*: A^1, A^0
  - A^1=1 means logic one. A^0=1 means logic zero.
  - A^1=0 A^0=0 means unknown.
- Example *W*=4, four patterns
  - *C* = {0, 0, u, 1}
  - C^1 = (0, 0, 0, 1)
  - C^0 = (1, 1, 0, 0)



Parallel Gate Evaluation



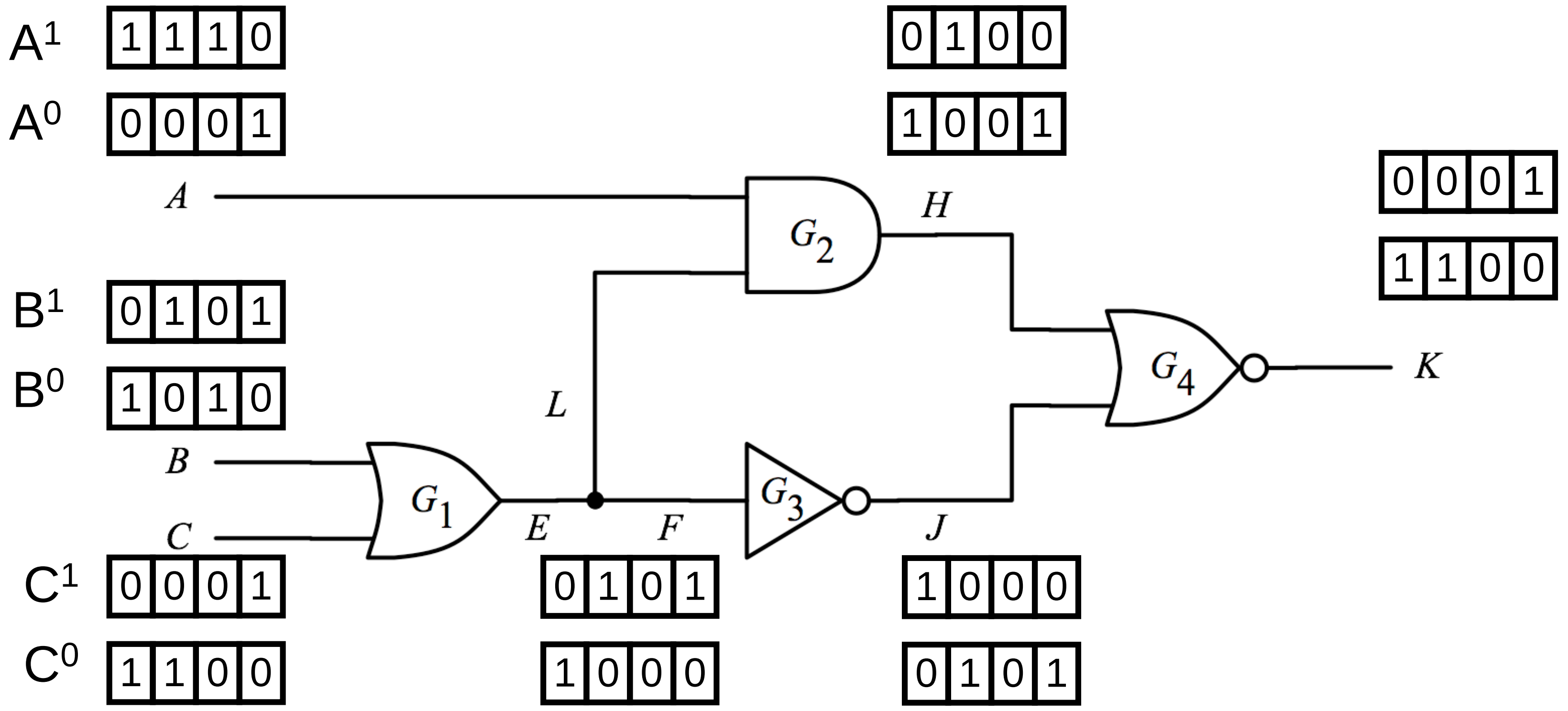
$\cdot$= bitwise AND
+ = bitwise OR

Gate	Bitwise Operations
AND	$c^1 = a^1.b^1$ $c^0 = a^0 + b^0$
NAND	$c^1 = a^0 + b^0$ $c^0 = a^1.b^1$
OR	$c^1 = a^1 + b^1$ $c^0 = a^0.b^0$
NOR	$c^1 = a^0.b^0$ $c^0 = a^1 + b^1$
INV	$c^1 = a^0$ $c^0 = a^1$

Quiz: what are equations for XOR?
A:

Example

- **Apply four patterns**
  - **A={1,1,1,0 }; B={0,1,0,1 }; C={0,0,u,1 }**
  - **K={0,0,u,1 }**



What is Complexity of LogicSim?

- Suppose P patterns, G gates
- Compiled-code, parallel simulation = $\Theta(P \times G)$
- Event-driven simulation = $\Theta(P \times E)$
  - E: number of events in each pattern
  - Assume E = O(G)
    - O($P \times G$)

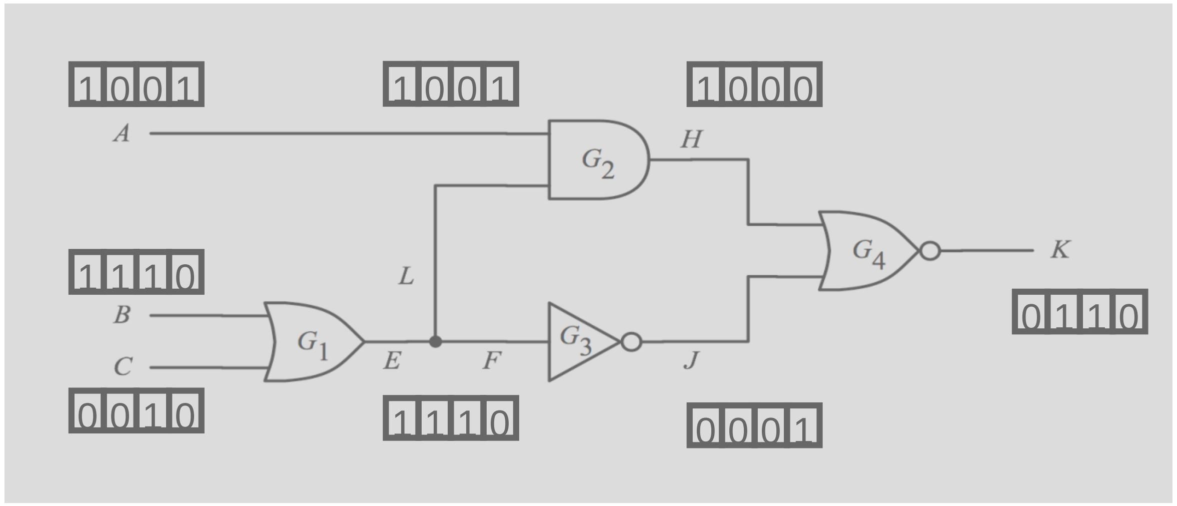
- Logic simulation is polynomial time complexity

Logic Simulation is Polynomial Time

Summary

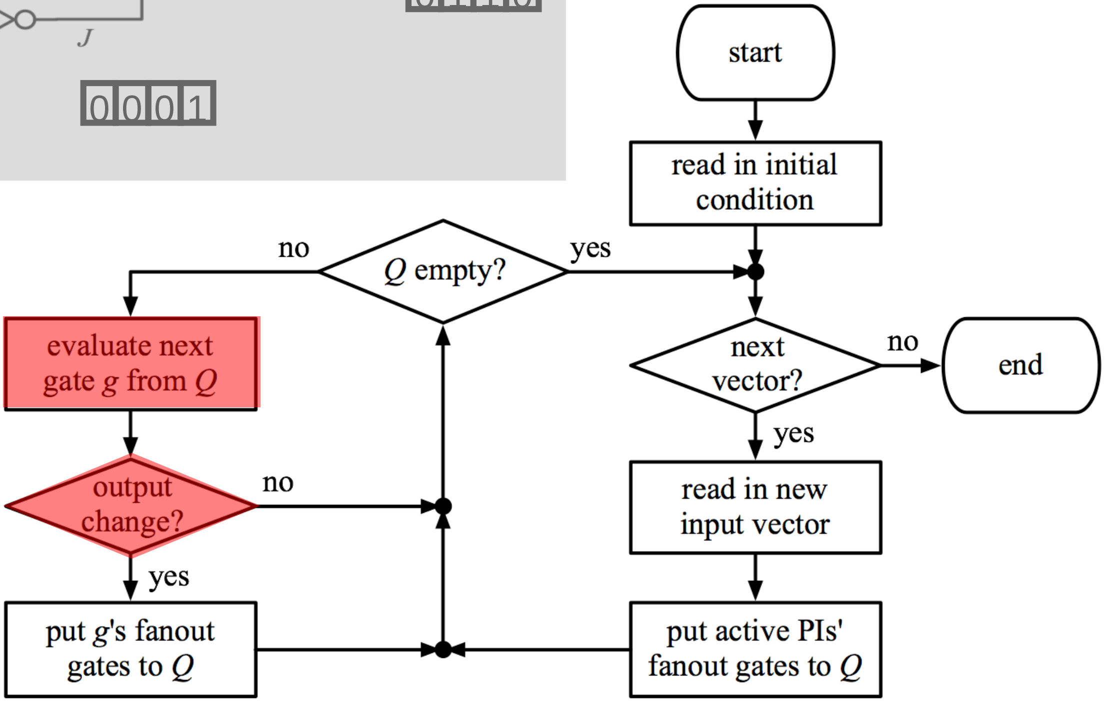
- Introduction
- Simulation Models
- Logic Simulation Techniques
  - Compiled code simulation
  - Event driven simulation
  - Parallel Simulation
    - Exploits bitwise operation to gain linear speed up
    - Improved encoding for unknowns
    - Logic simulation is polynomial time
- Issues of Logic Simulations
- Conclusions

Parallel Version of Compiled-code/Event-driven



```
while{true} do
  read(A,B,C);
  E   OR(B,C);
  H   AND(A,E);
  J   NOT(E);
  K   NOR(H,J);
end
```

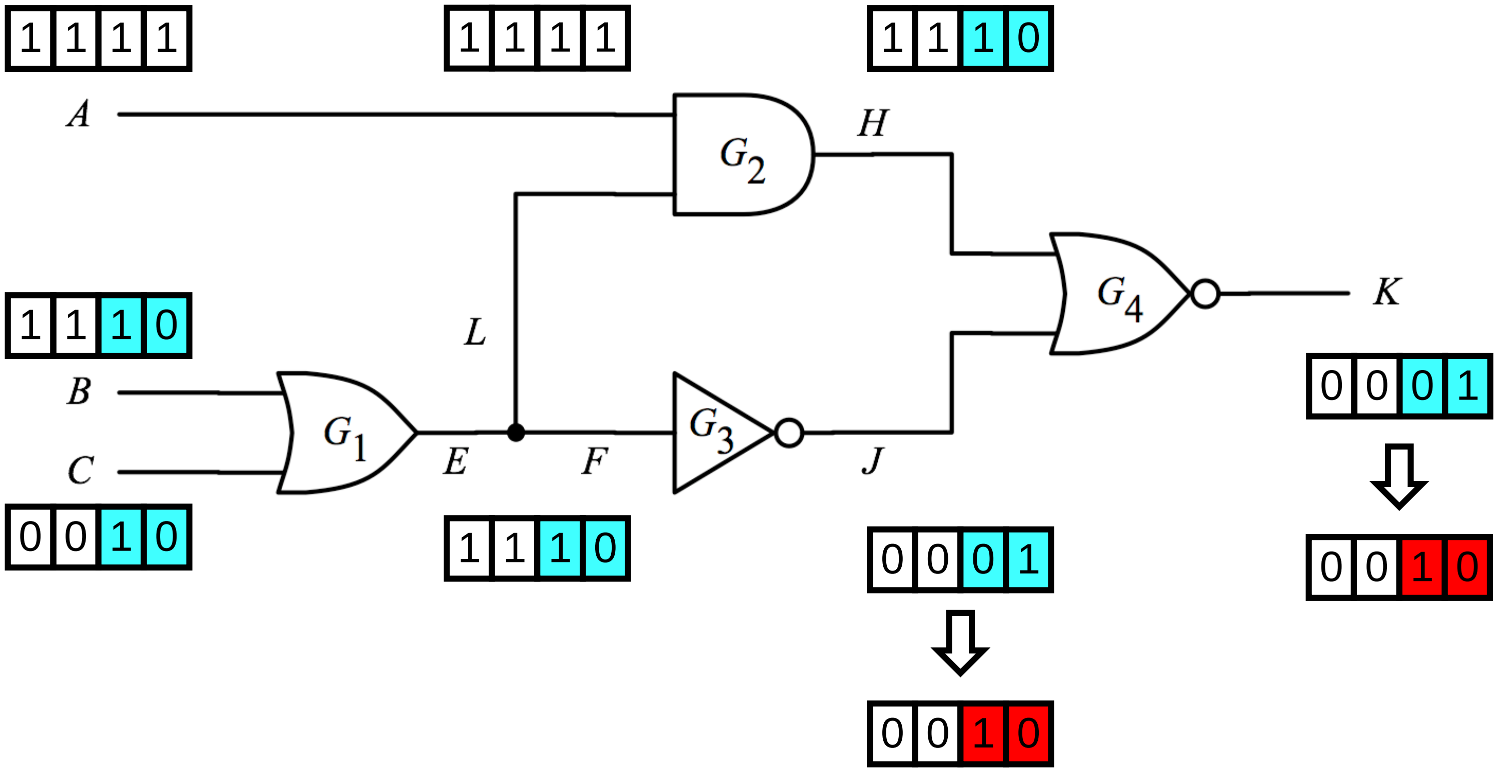
compiled-code


start
read in initial condition
no
Q empty?
yes
evaluate next gate g from Q
next vector?
no
end
yes
output change?
no
read in new input vector
yes
put g's fanout gates to Q
put active PIs' fanout gates to Q

one-pass event-driven (zero-delay)

FFT

- Q1: Can we swap bit pairs after inverter?
- Q2: If we can, what are advantages/disadvantages of 1-word encoding method compared with the 2-word encoding method ?



1 1 1 1
1 1 1 1
1 1 1 0
A
G_2
H
1 1 1 0
L
G_4
K
0 0 0 1
0 0 1 0
B
C
G_1
E
F
G_3
J
0 0 1 0
1 1 1 0
0 0 0 1
0 0 1 0