Time

Until now: we have essentially ignored the issue of time

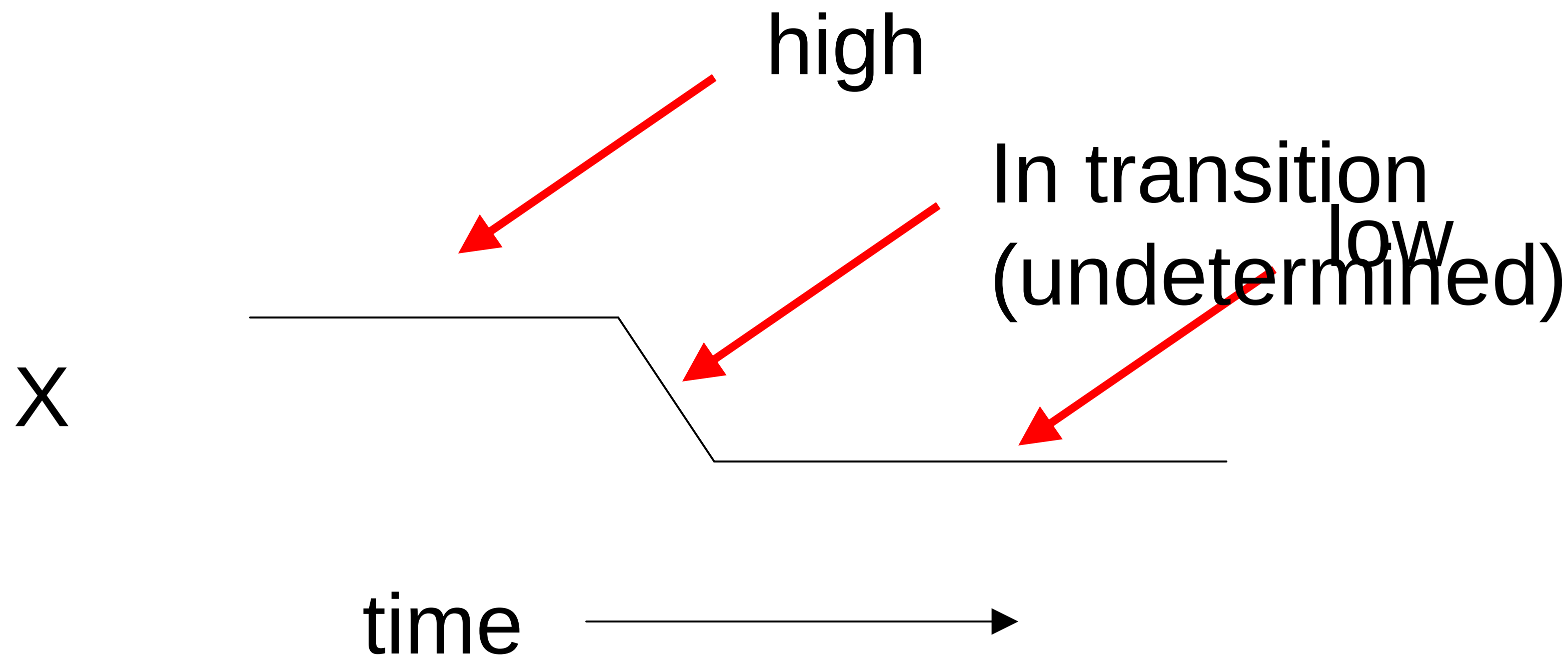
- We have assumed that our digital logic circuits perform their computations instantaneously
- Our digital logic circuits have been "stateless"
  - Once you present a new input, they forget everything about previous inputs
  - We call this type of digital system **combinatorial logic**

Time

In reality, time is an important issue:

- Even our logic gates induce a small amount of delay (on the order of a few nanoseconds)
- For much of what we do – we actually want our circuits to have some form of memory

Timing Notation



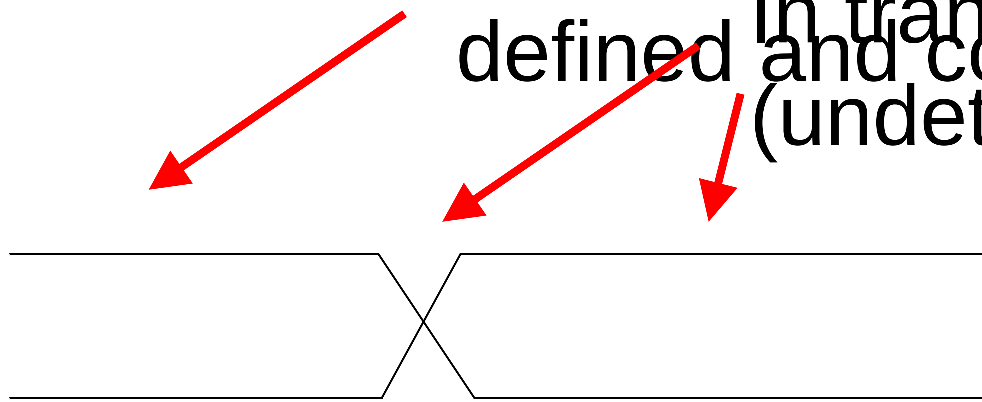
Timing Notation

Either high or low (but well defined and constant)

In transition

(undetermined)

X

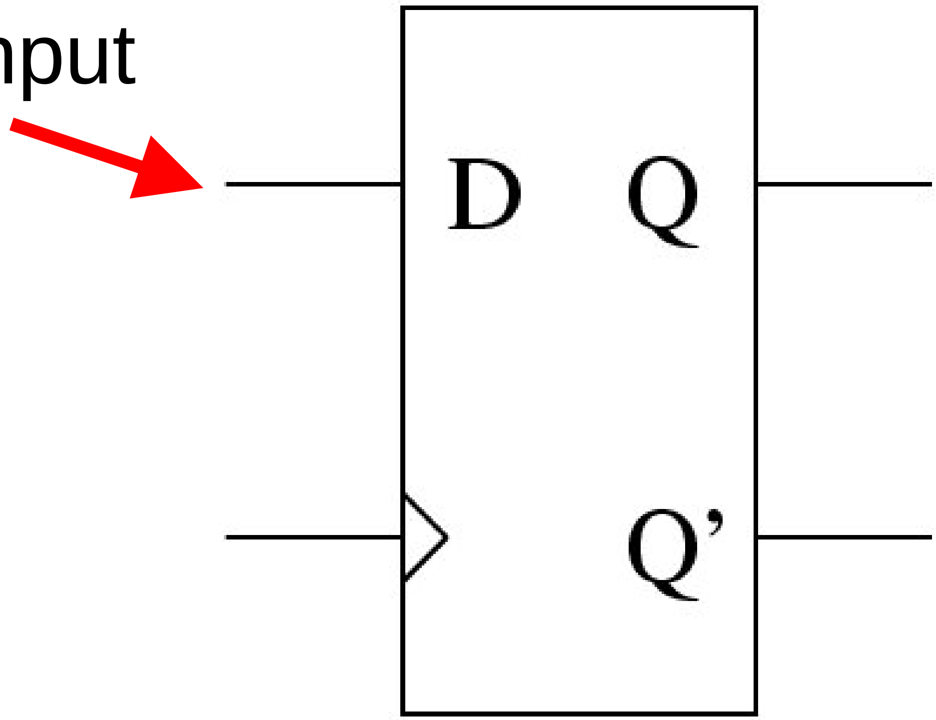


time

low

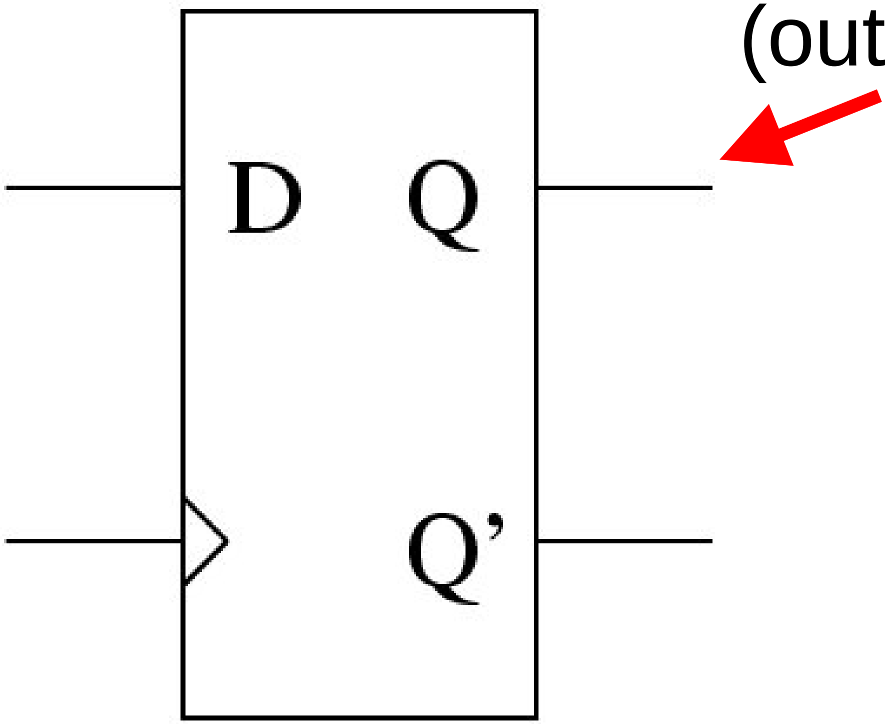
D Flip Flops

Data Input



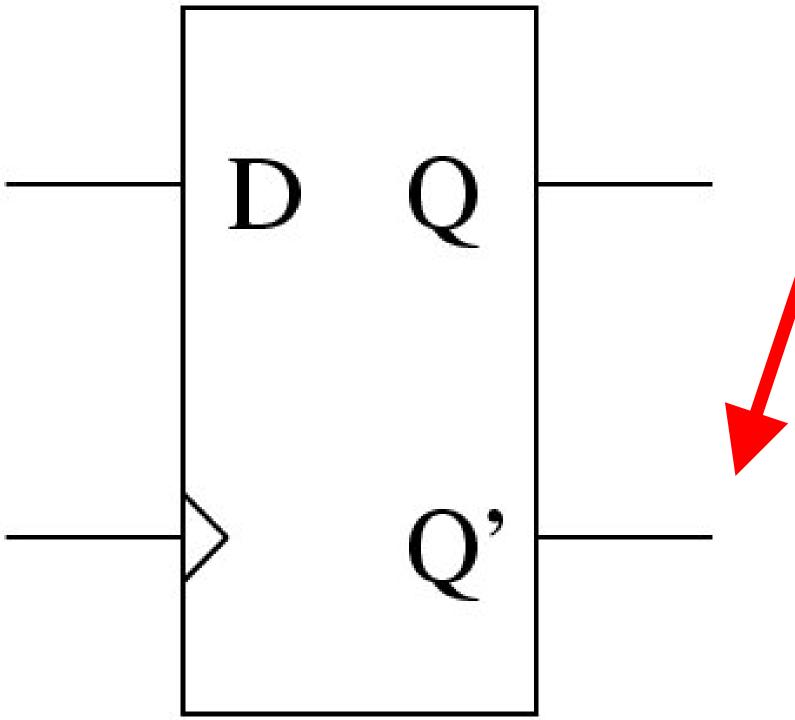
D Flip Flops

Stored value
(output)

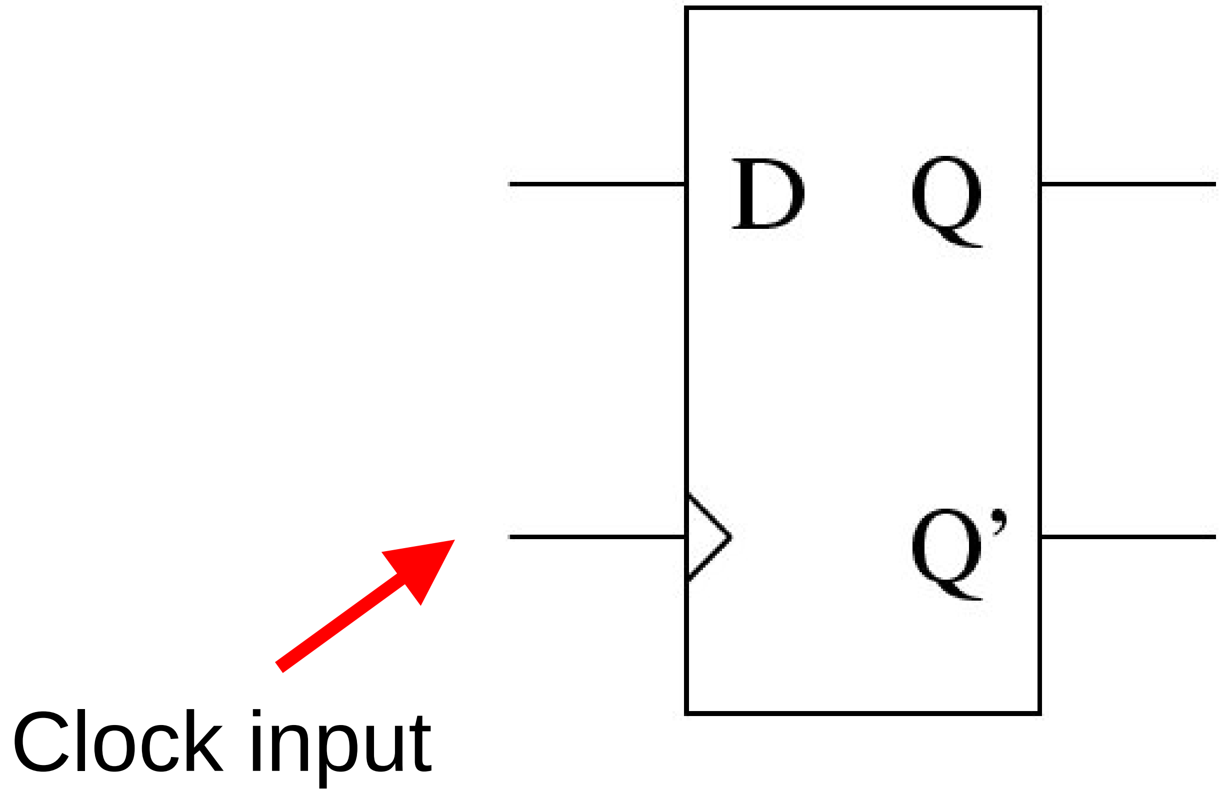


D Flip Flops

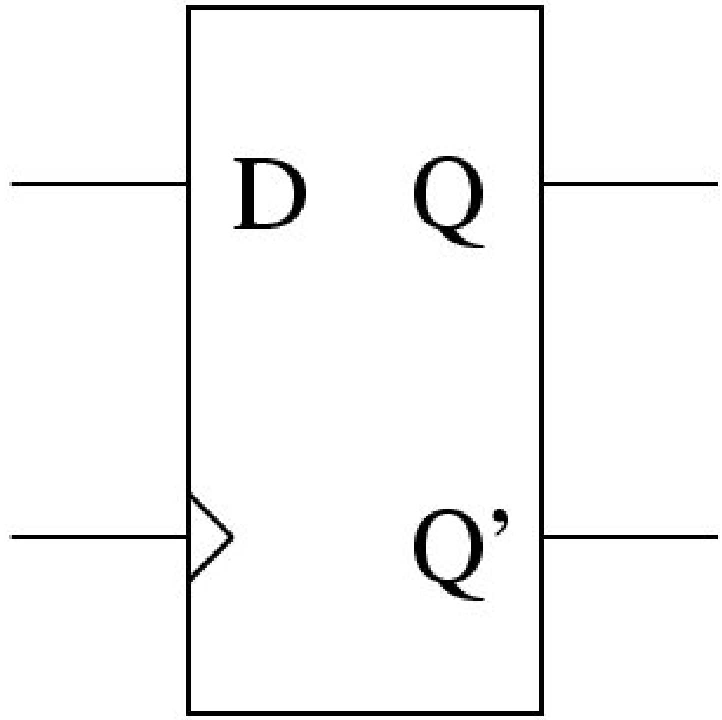
NOT of stored value



D Flip Flops

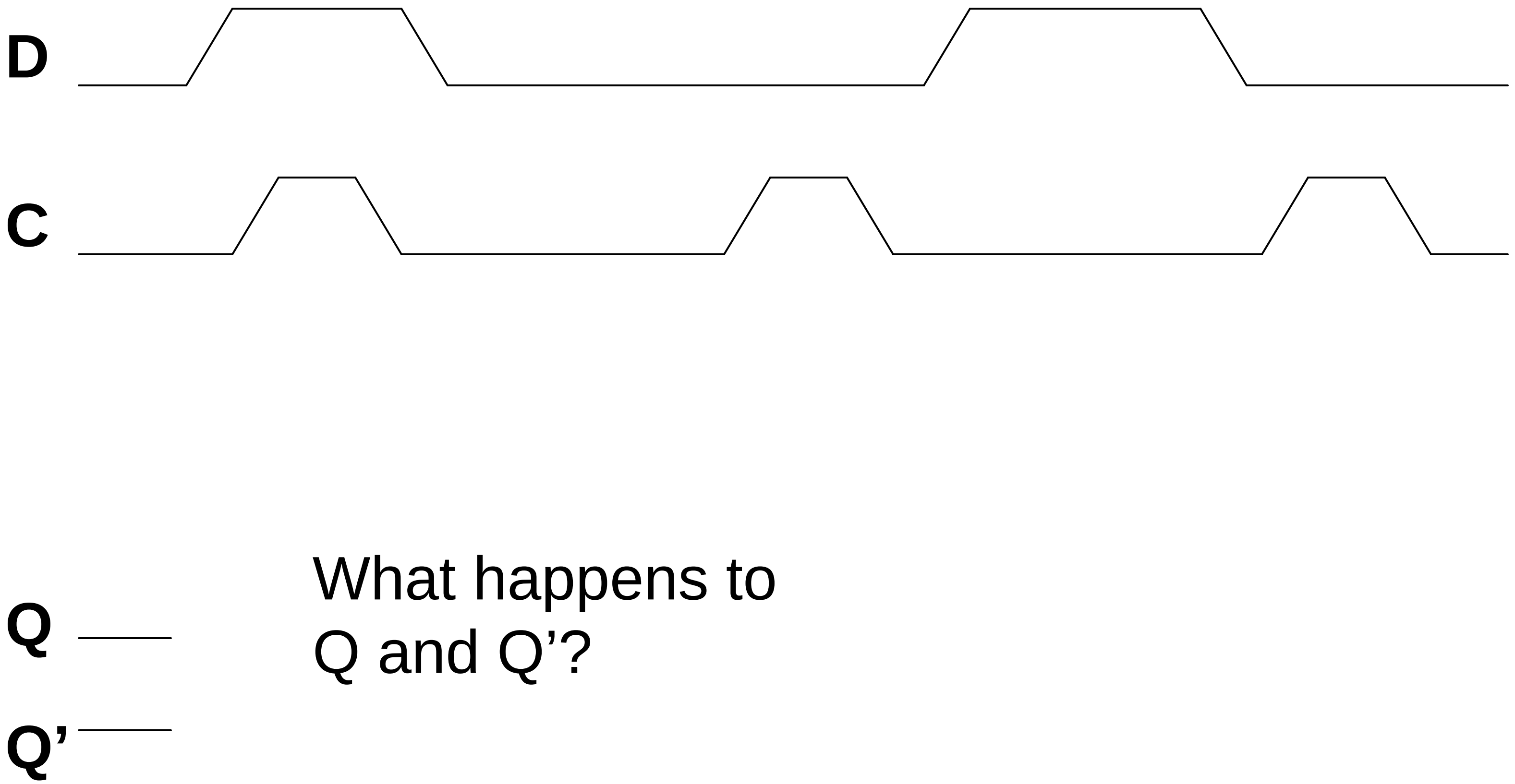


D Flip Flops



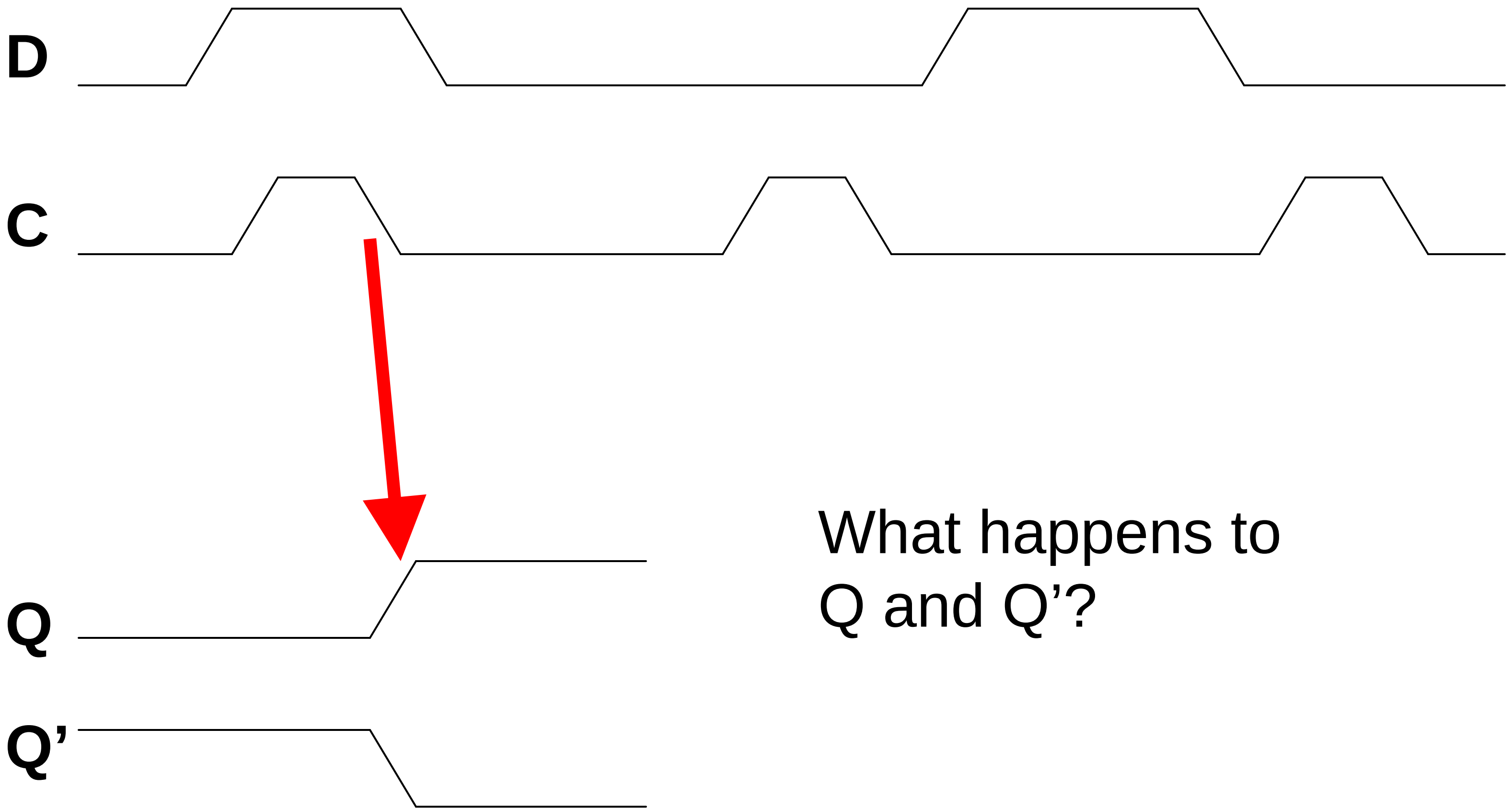
When the clock transitions from high to low:
the value of D is stored

D Flip Flop



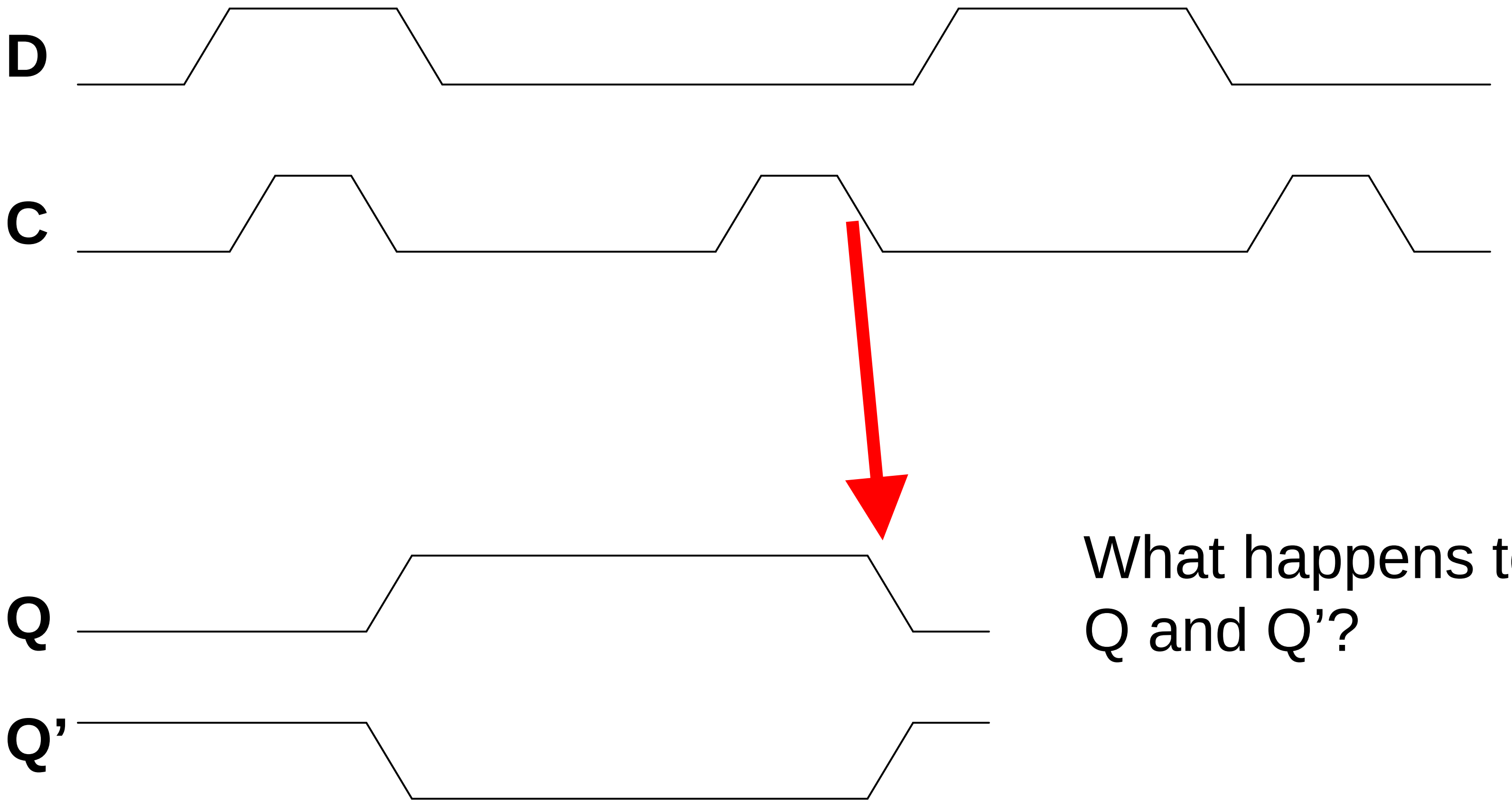
What happens to
Q and Q'?

D Flip Flop



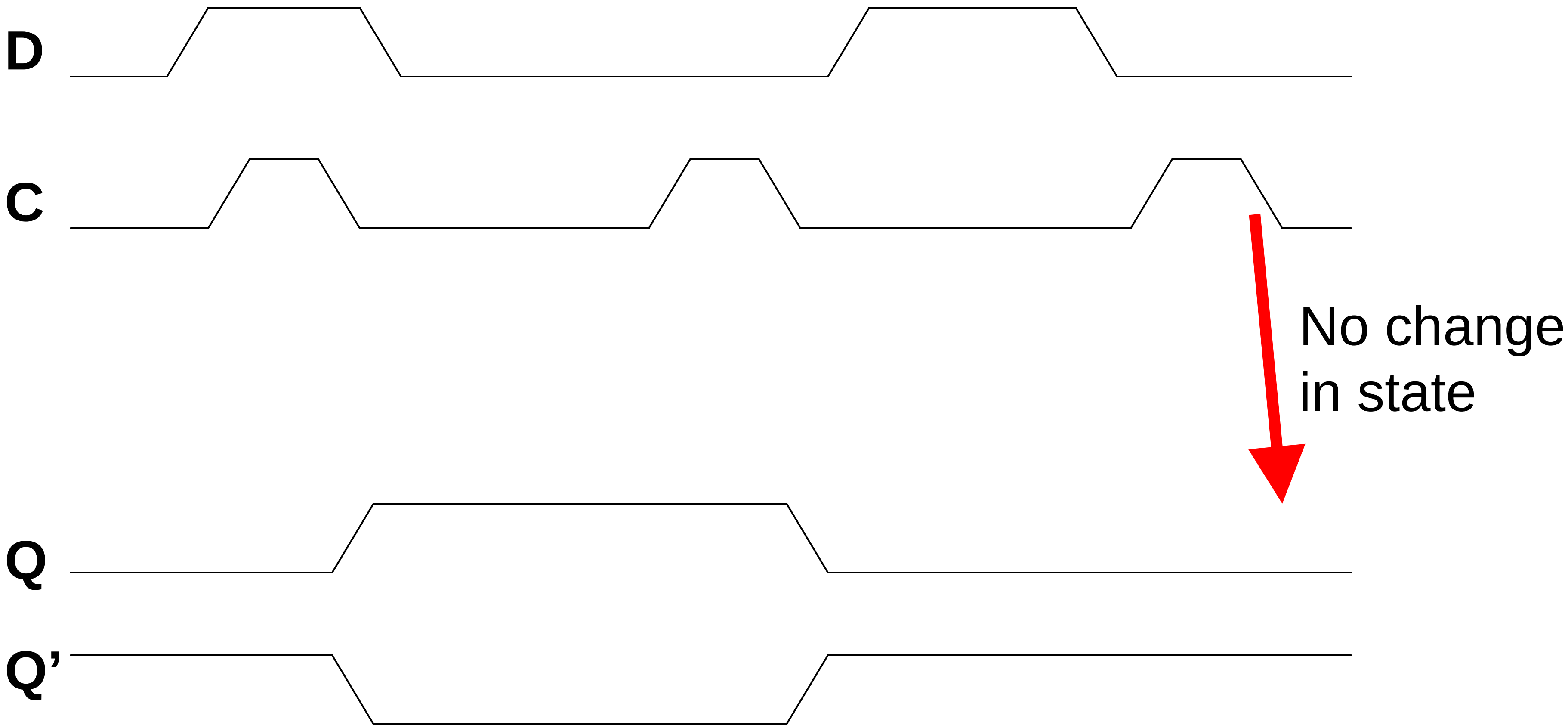
What happens to
Q and Q'?

D Flip Flop



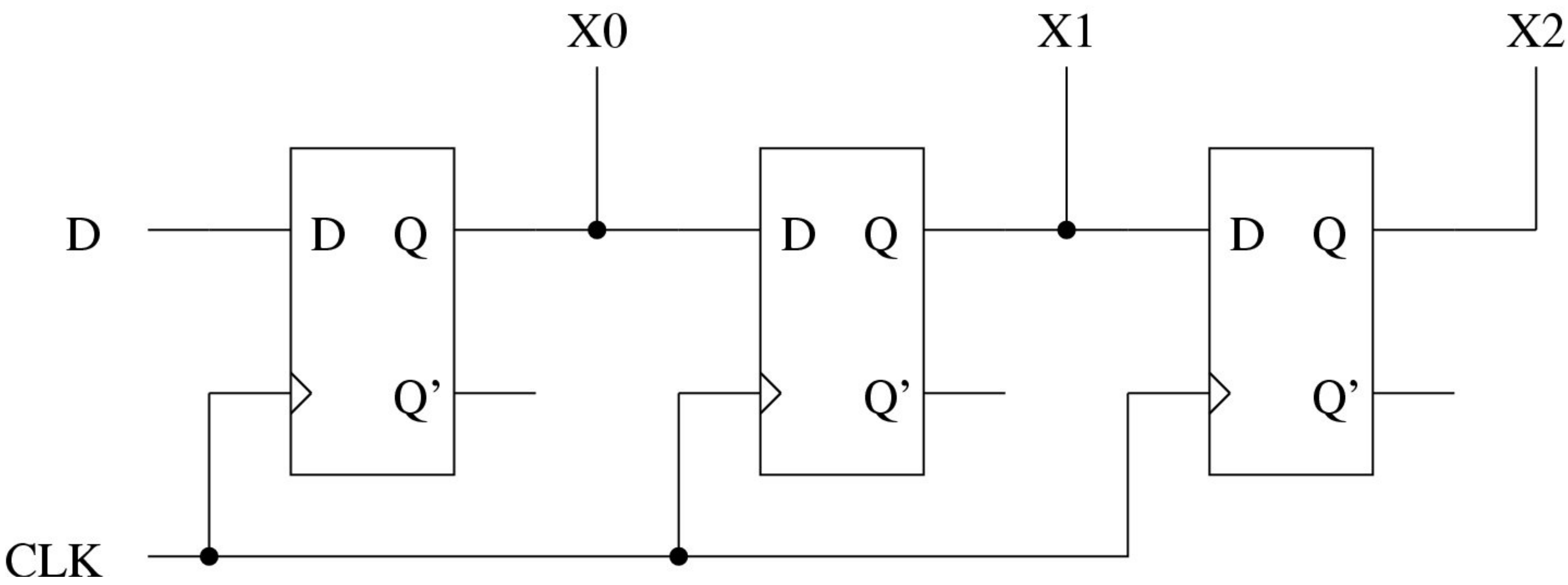
What happens to Q and Q’?

D Flip Flop



An Application of D Flip Flops

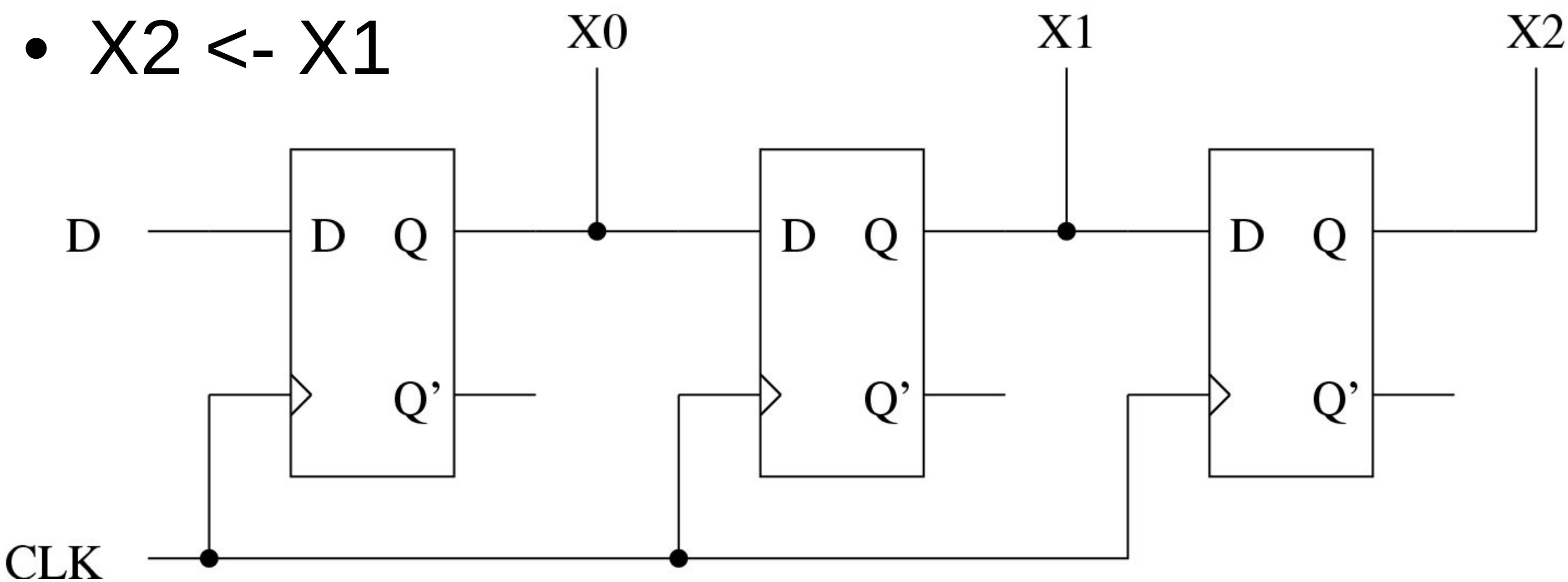
What does this circuit do?



Shift Register

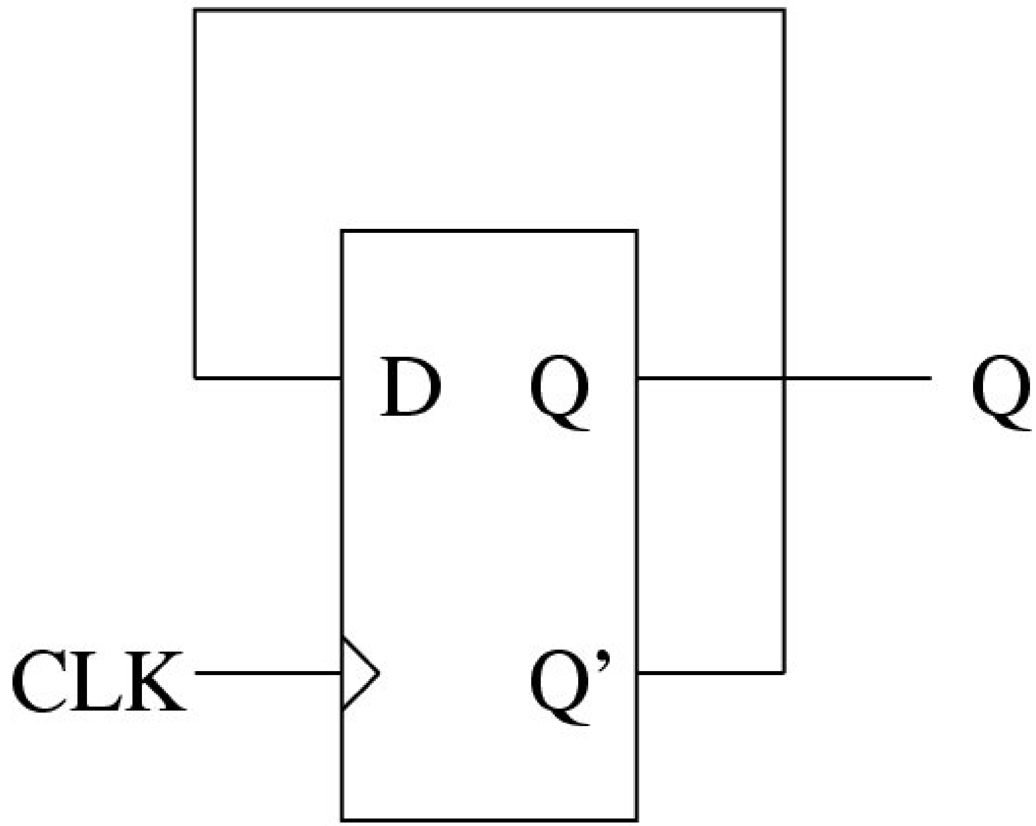
On each clock transition from high to low:

- X0 takes on the current value of D
- X1 <- X0
- X2 <- X1



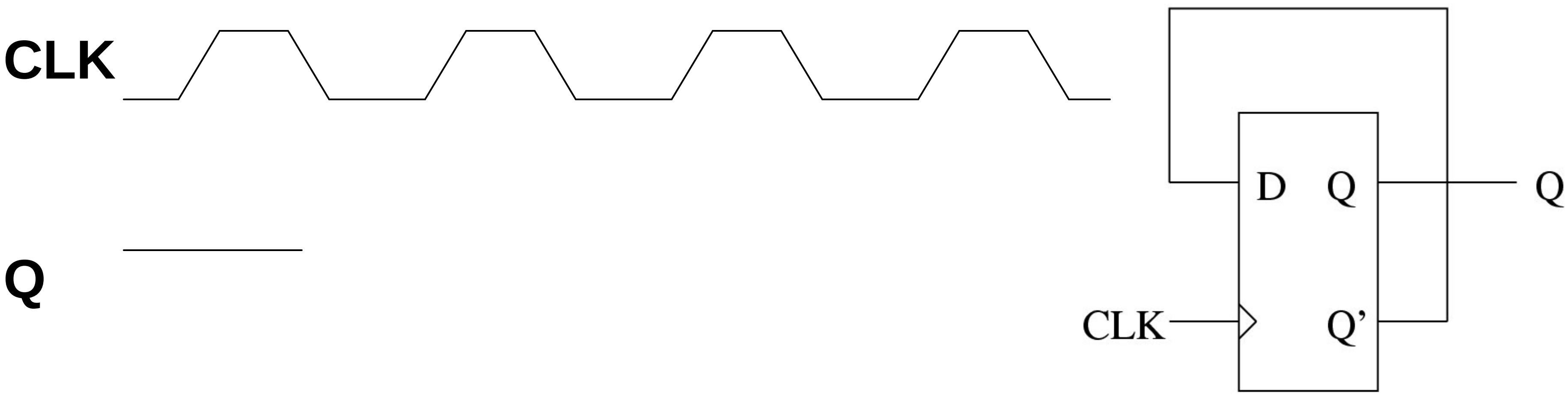
Another D Flip Flop Circuit

How does this circuit behave?



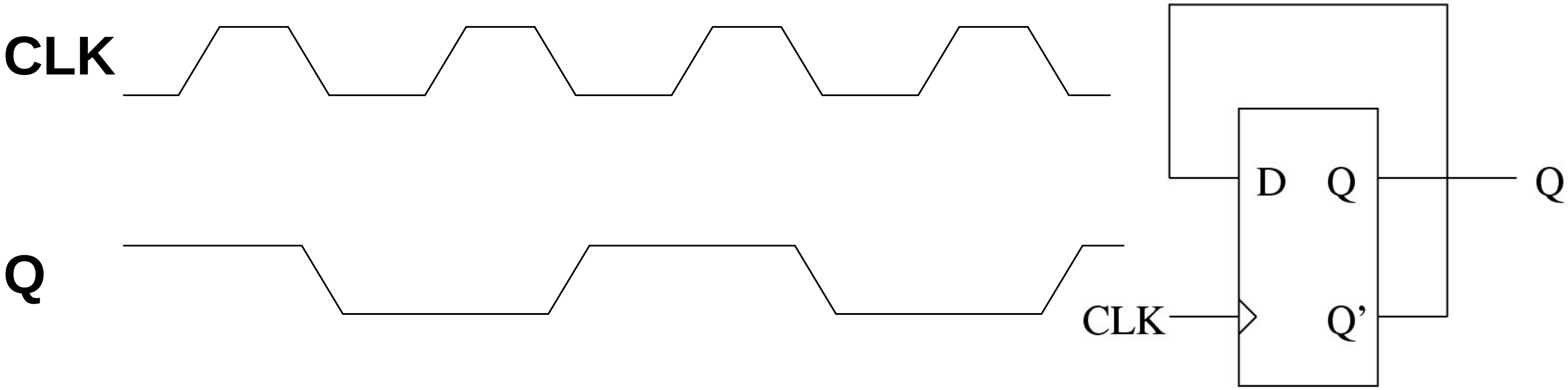
Frequency Divider

How does this circuit behave?



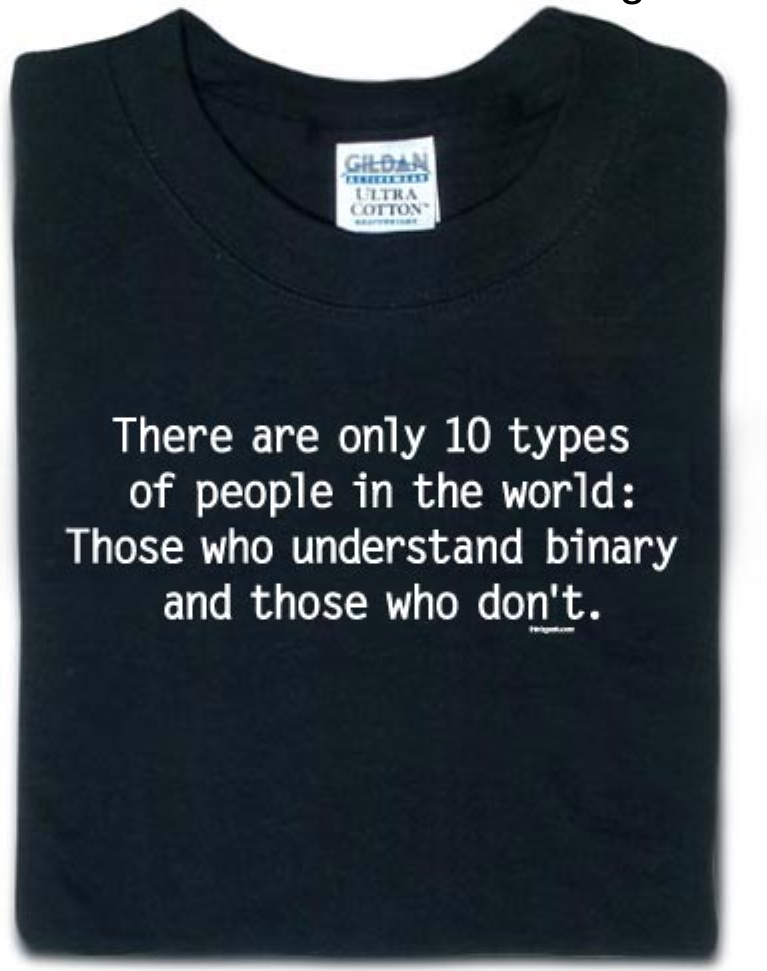
Frequency Divider

Q flips state on every downward edge of the clock



A Bit About Binary Encoding

www.thinkgeek.com



If a boolean variable can only encode two different values, how do we represent a larger number of values?

Binary Encoding

How do we represent a larger number of values?

Binary Encoding

How do we represent a larger number of values?

- As with our decimal number system: we concatenate binary digits (or “bits”) into strings

Binary Encoding

- The first (rightmost) bit is the 1's digit
- The second bit is the 2's digit
- The ith bit is the 2^{i-1} 's digit

Binary Encoding

How do we convert from binary to decimal in general?

B2	B1	B0		decimal
0	0	0		0
0	0	1		1
0	1	0		2
0	1	1		3
1	0	0		4
1	0	1		5
1	1	0		6
1	1	1		7

Last Time

Sequential Logic

- D Flip Flops
- Shift registers
- Binary number system

Today

- A little more on number systems
- Use of flip-flops
- Microprocessor basics
  - Memory
  - Arithmetic Logical Units
  - Instructions and execution

Administrivia

- Homework 1 due today at 5:00
- Homework 2 available tonight

Binary to Decimal Conversion

$$value = B_0 + B_1 * 2^1 + B_2 * 2^2 + B_3 * 2^3 + \ldots$$

$$value = \sum_{i=0}^{N-1} B_i * 2^i$$

How do we convert from decimal to binary?

Decimal to Binary Conversion

$$\forall i : B_i \leftarrow 0$$

$$while(value \neq 0)$$

$$\{$$

$$Find\ i\ suchthat\ 2^{i+1} > value \geq 2^i$$

$$B_i \leftarrow 1$$

$$value \leftarrow value - 2^i$$

$$\}$$

Binary Counter

How would we build a circuit that counts the number of clock ticks that have gone by?

B2	B1	B0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Binary Counter

How would we build a circuit that counts the number of clock ticks that have gone by?

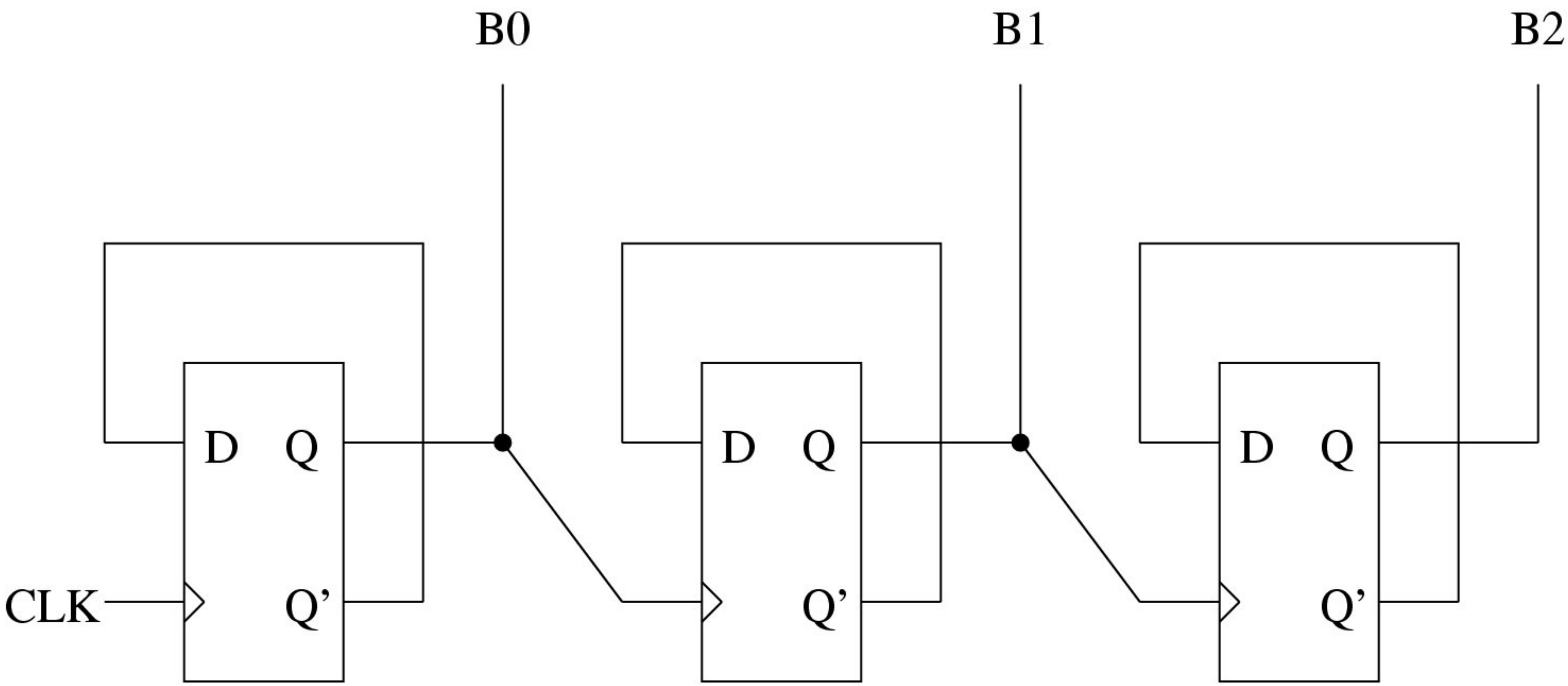
Insight:

- B1 changes state at half the frequency that B0 does
- B2 changes state at half the frequency of B1

B2	B1	B0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Ripple Counter

The carry “ripples” down the chain …



Ripple Counter Notes

- The bits do not change state at the same time
- This can be repaired with a more sophisticated circuit design
  - We will experiment with this in hw2

Flip-Flop Notes

- Means of storing ‘bits’ of data
- Have now seen two circuits that operate on sets of ‘bits’ (or binary numbers)
  - – Counter
  - – Shift register
    - What arithmetic operation does shifting perform?
- These are examples of operations that are performed by the “Arithmetic Logical Unit”