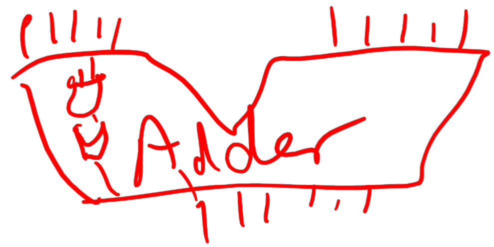
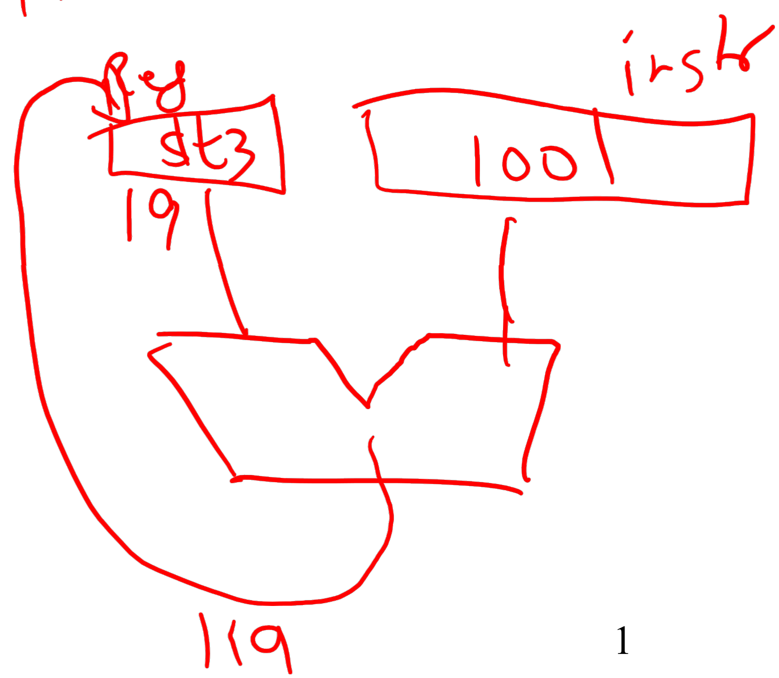
Lecture 14: Sequential Circuits, FSM

- Today's topics:
  - Adder wrap-up
  - Sequential circuits
  - Finite state machines

Combinational Ccts



addi $t3, $t3, 100



Adder Summary

$g = a_i \cdot b_i$
$p = (a_i + b_i)$
[16-bit Addition]

- Using the generate/propagate abstraction to add layers of ccts
- Key: all g/p/G/P signals can be calculated based on a/b inputs

  (they don't need carry-in as inputs, so they can all be done rightaway in parallel)
- First calculate g/p with 1 gate delay: gi = ai.bi ; pi = ai + bi
- Then calculate G/P with up to 2 gate delays (for a block of 4 bits):

  Gi = g3 + g2.p3 + g1.p2.p3 + g0.p1.p2.p3

  Pi = p0.p1.p2.p3
- Then calculate all the carries, including for the 16^{th} bit, with 2 more gate delays:

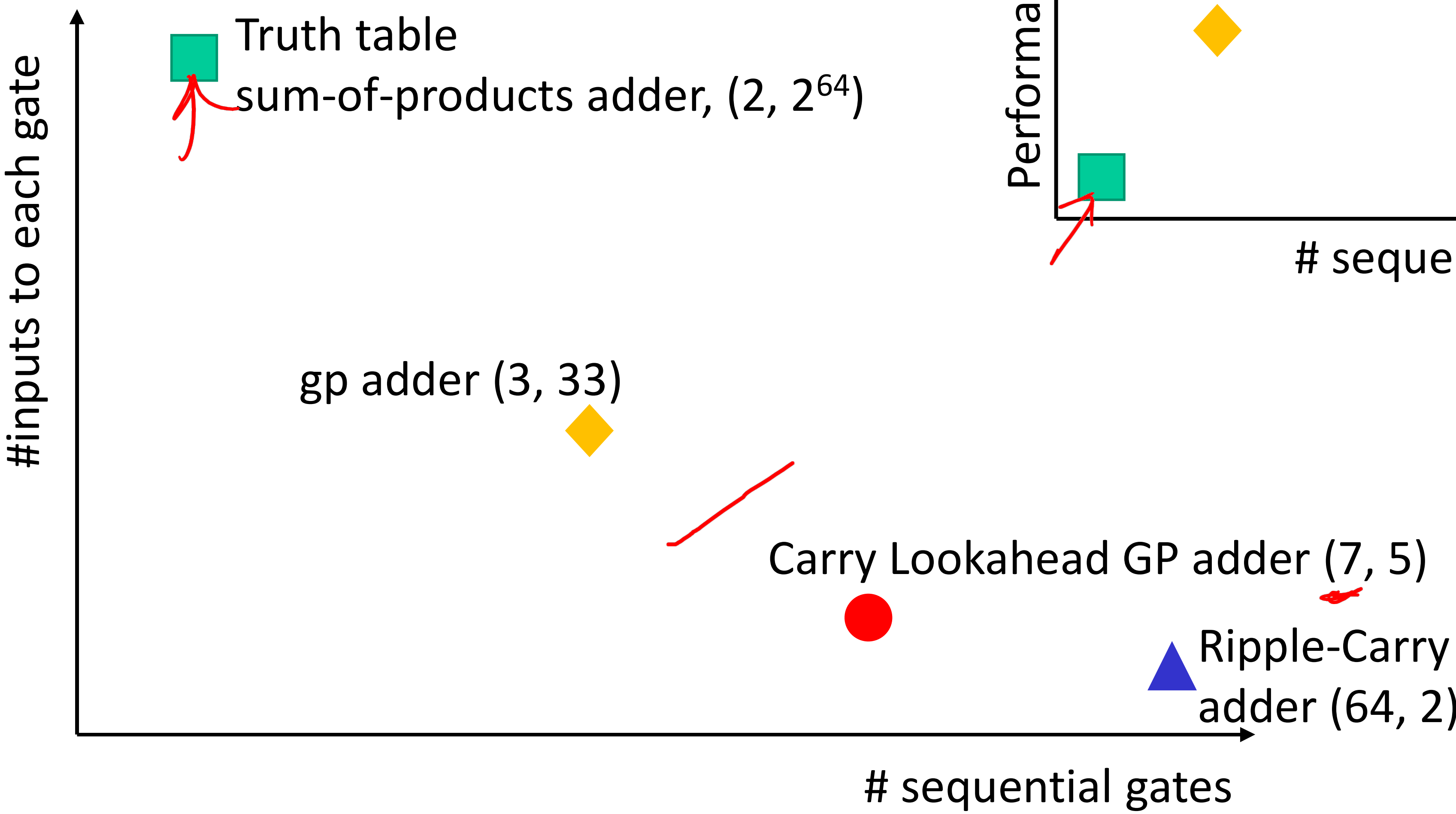
  C4 = G3 + (P3.G2) + (P3.P2.G1) + (P3.P2.P1.G0) + (P3.P2.P1.P0.c0)

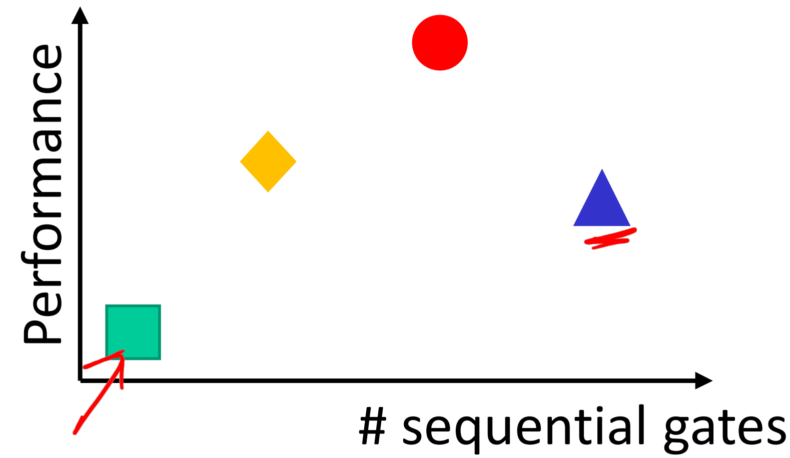
- Thus, this abstraction enables a design with a modest number of total gates, a modest number of delays, and a modest number of inputs per gate.

Trade-Off Curve

mod # seq gates
mod # inputs per gate
mod # total gates

(32-bit addition)

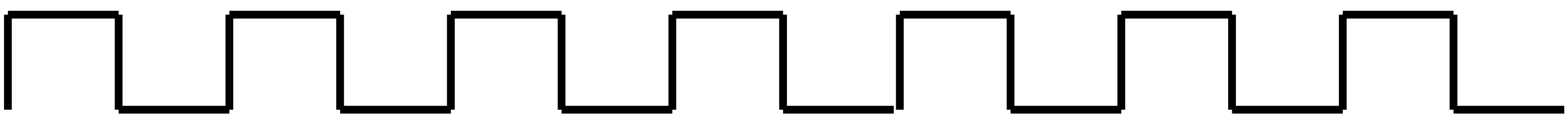




Clocks

- A microprocessor is composed of many different circuits that are operating simultaneously – if each circuit X takes in inputs at time TI_X, takes time TE_X to execute the logic, and produces outputs at time TO_X, imagine the complications in co-ordinating the tasks of every circuit

- A major school of thought (used in most processors built today): all circuits on the chip share a clock signal (a square wave) that tells every circuit when to accept inputs, how much time they have to execute the logic, and when they must produce outputs

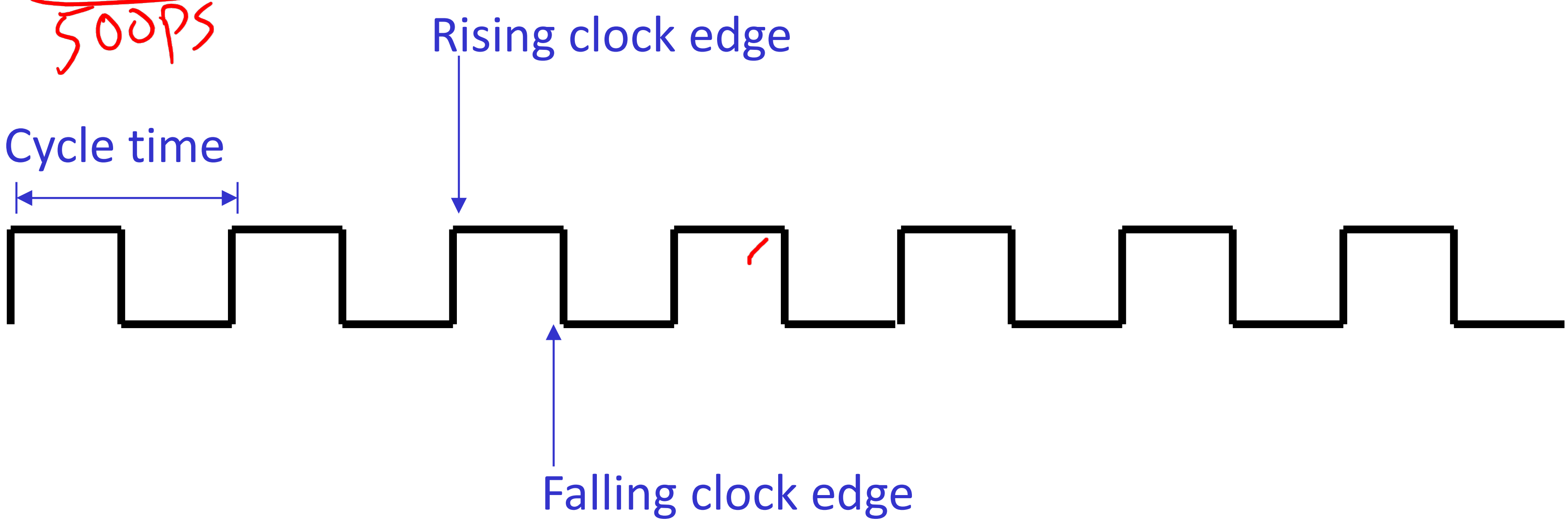


Clock Terminology

1.67 GHz 2 GHz

600 ps

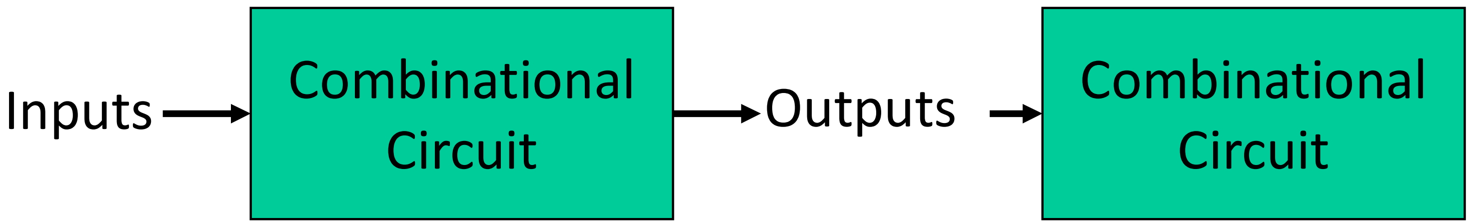
500ps



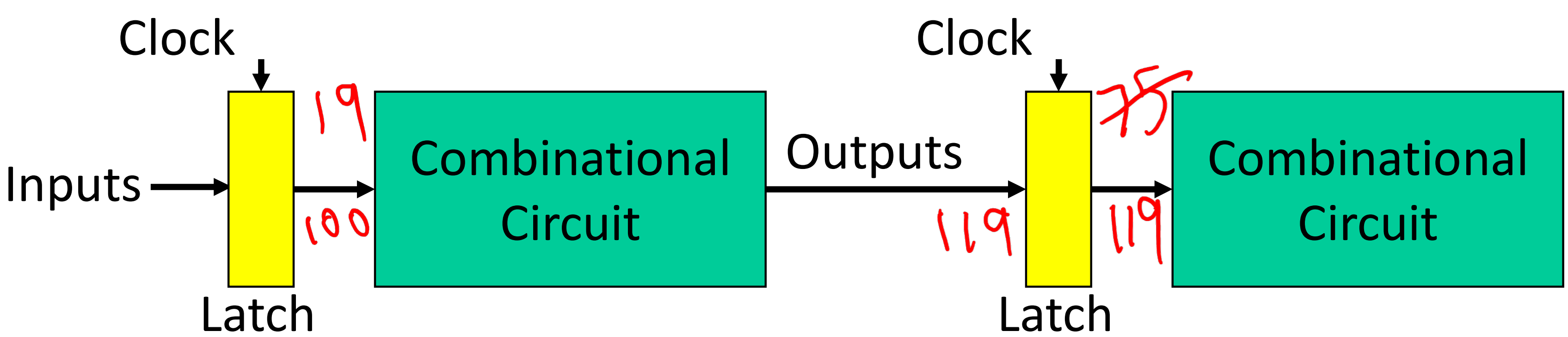
$$4\text{ GHz} = \text{clock speed} = \frac{1}{\text{cycle time}} = \frac{1}{250\text{ ps}}.$$

Sequential Circuits

- Until now, circuits were combinational – when inputs change, the outputs change after a while (time = logic delay thru circuit)

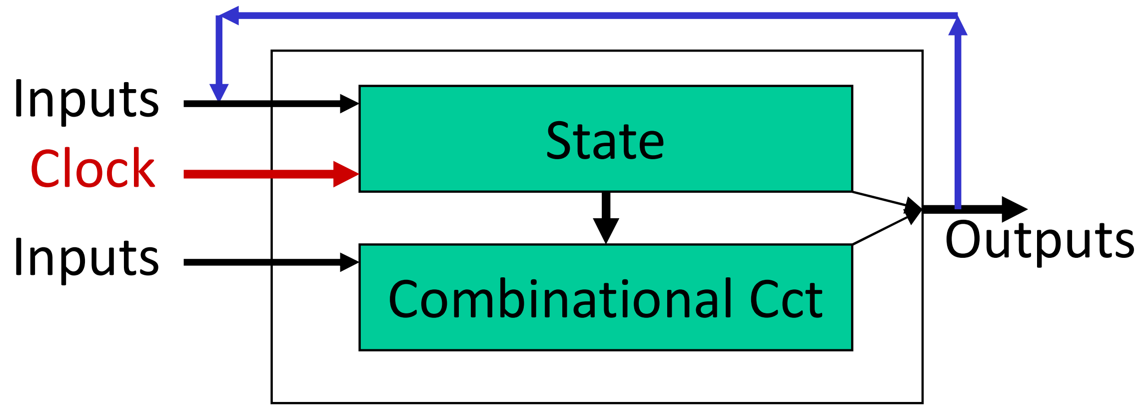


- We want the clock to act like a start and stop signal – a "latch" is a storage device that separates these circuits – it ensures that the inputs to the circuit do not change during a clock cycle



Sequential Circuits

- Sequential circuit: consists of combinational circuit and a storage element
- At the start of the clock cycle, the rising edge causes the “state” storage to store some input values

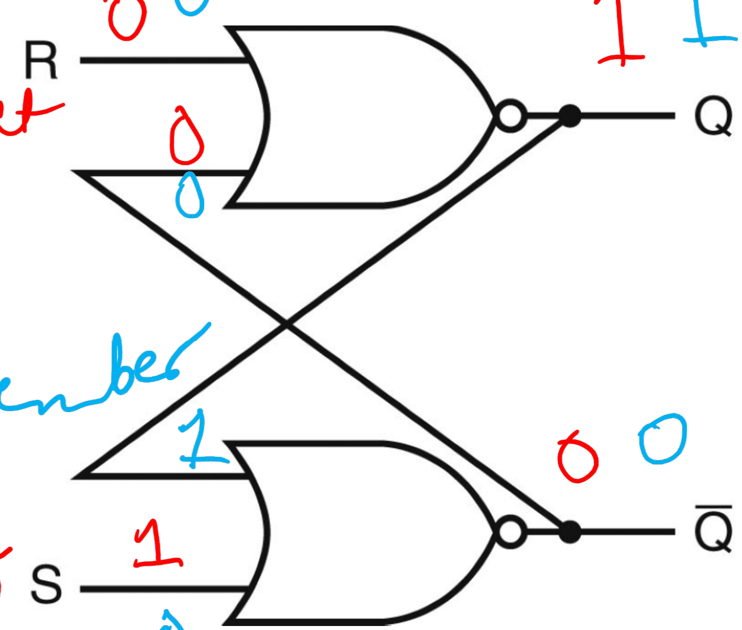


- This state will not change for an entire cycle (until next rising edge)
- The combinational circuit has some time to accept the value of “state” and “inputs” and produce “outputs”
- Some of the outputs (for example, the value of next “state”) may feed back (but through the latch so they’re only seen in the next cycle)

Designing a Latch

- An S-R latch: set-reset latch
  - When Set is high, a 1 is stored
  - When Reset is high, a 0 is stored
  - When both are low, the previous state is preserved (hence, known as a storage or memory element)
  - Both are high – this set of inputs is not allowed

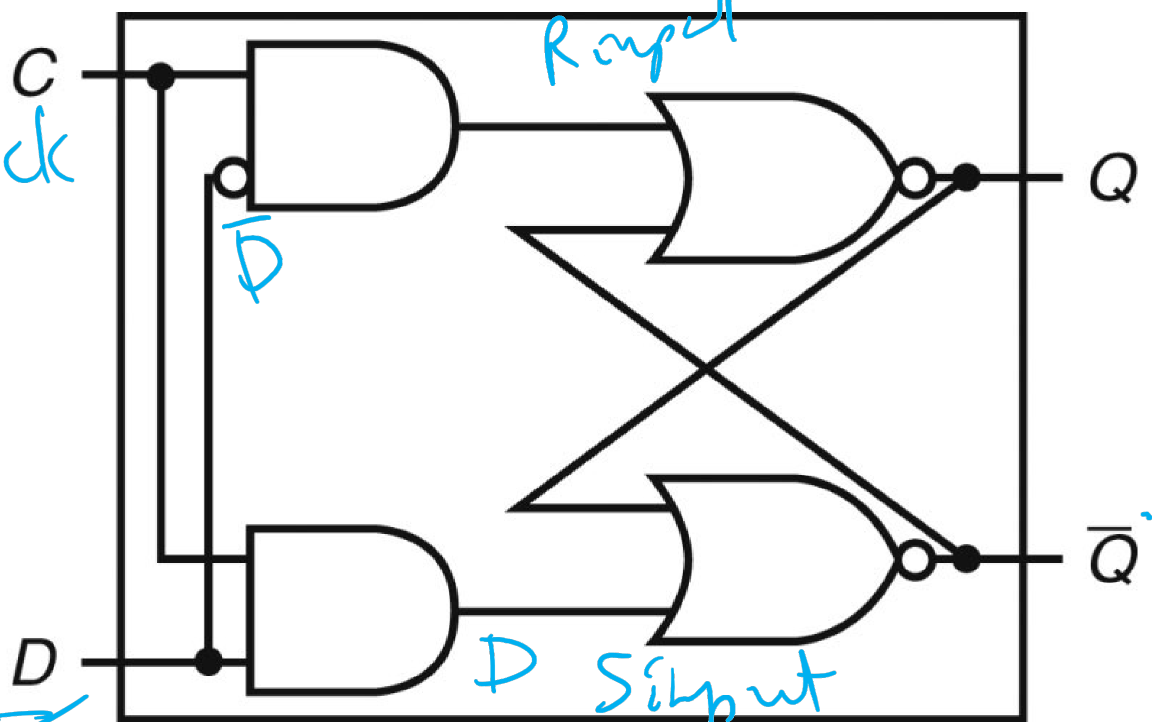
Verify the above behavior!



Source: H&P textbook

D Latch

- Incorporates a clock
- The value of the input D signal (data) is stored only when the clock is high – the previous state is preserved when the clock is low

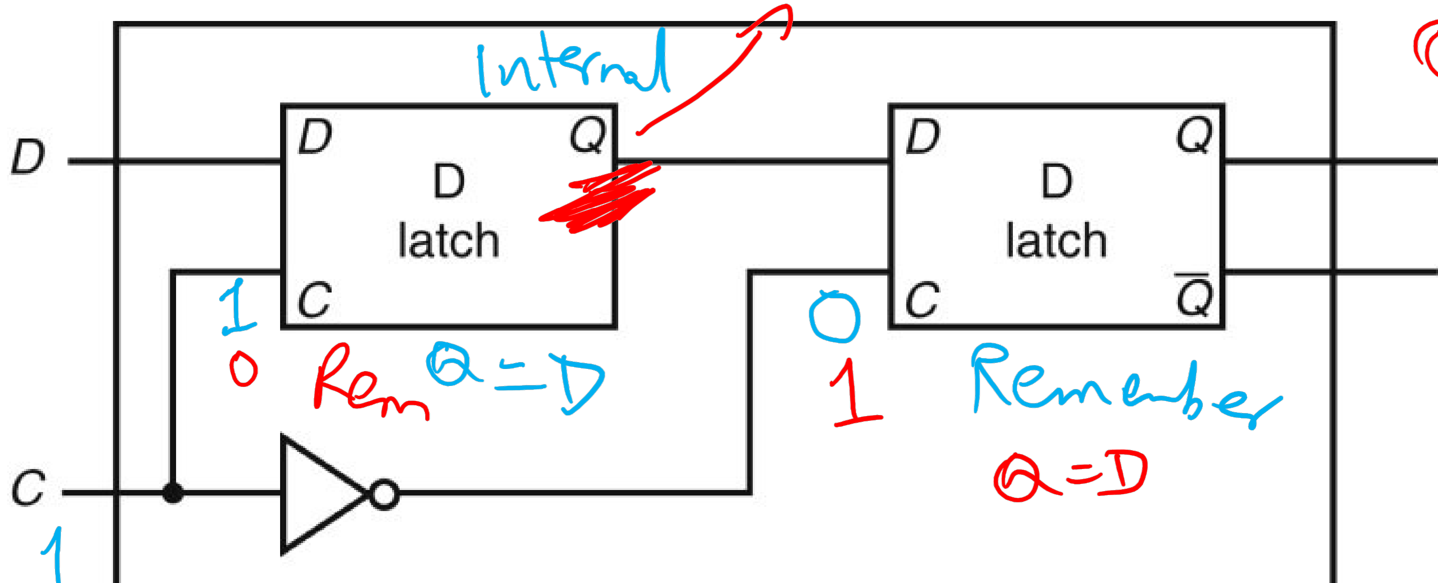


Source: H&P textbook

D Flip Flop

- Terminology:
  Latch: outputs can change any time the clock is high (asserted)
  Flip flop: outputs can change only on a clock edge

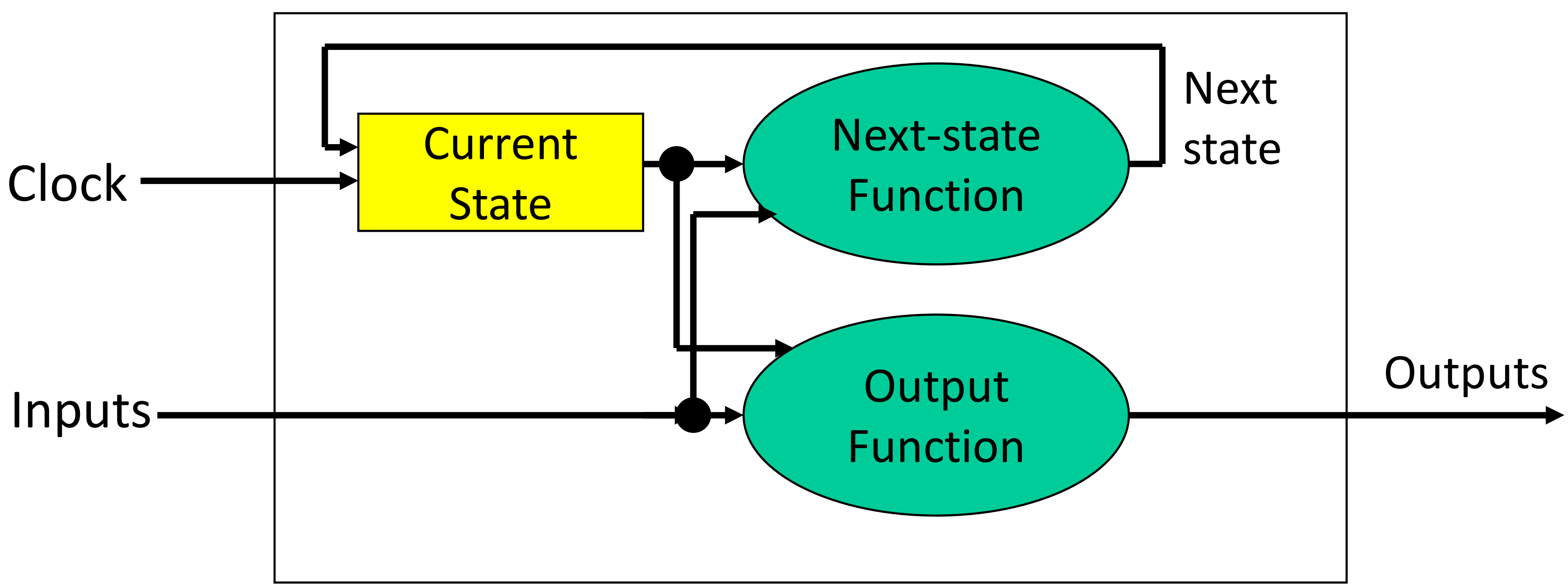
- Two D latches in series – ensures that a value is stored only on the falling edge of the clock



Source: H&P textbook

Finite State Machine

- A sequential circuit is described by a variation of a truth table – a finite state diagram (hence, the circuit is also called a finite state machine)
- Note that state is updated only on a clock edge



State Diagrams

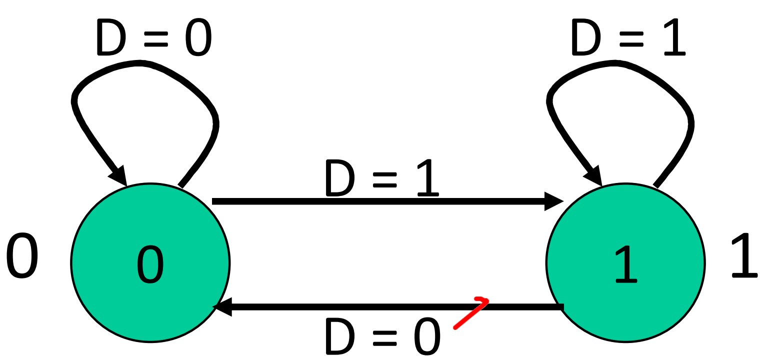
curr state	D	output state
0	0	0
0	1	1
1	0	0
1	1	1

clk exists

- Each state is shown with a circle, labeled with the state value – the contents of the circle are the outputs
- An arc represents a transition to a different state, with the inputs indicated on the label

1-bit saturating counter

2 output states 0 + 1



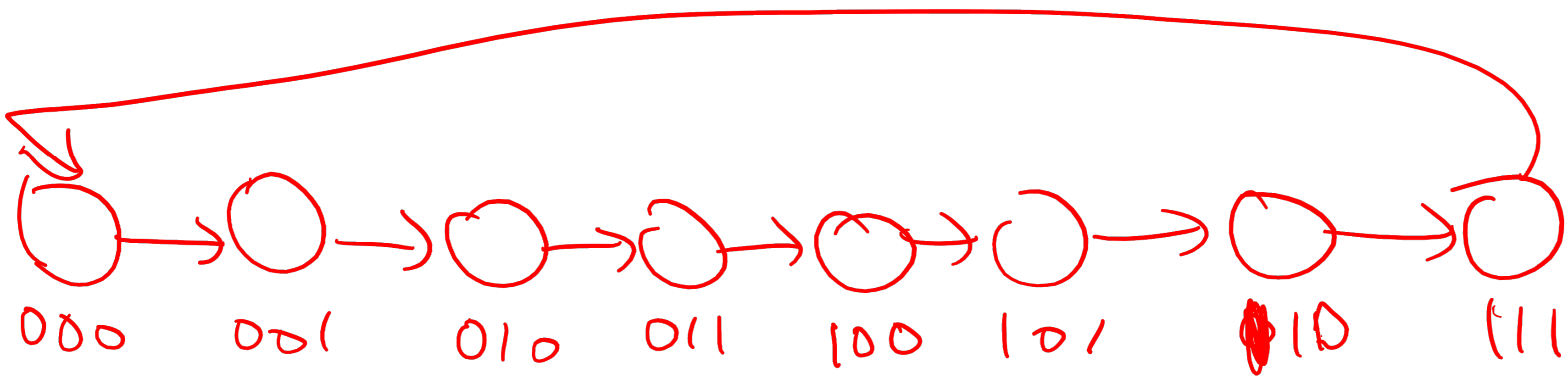
This is a state diagram for ___?

1 input D with 2 values 0 + 1

3-Bit Counter

- Consider a circuit that stores a number and increments the value on every clock edge – on reaching the largest value, it starts again from 0

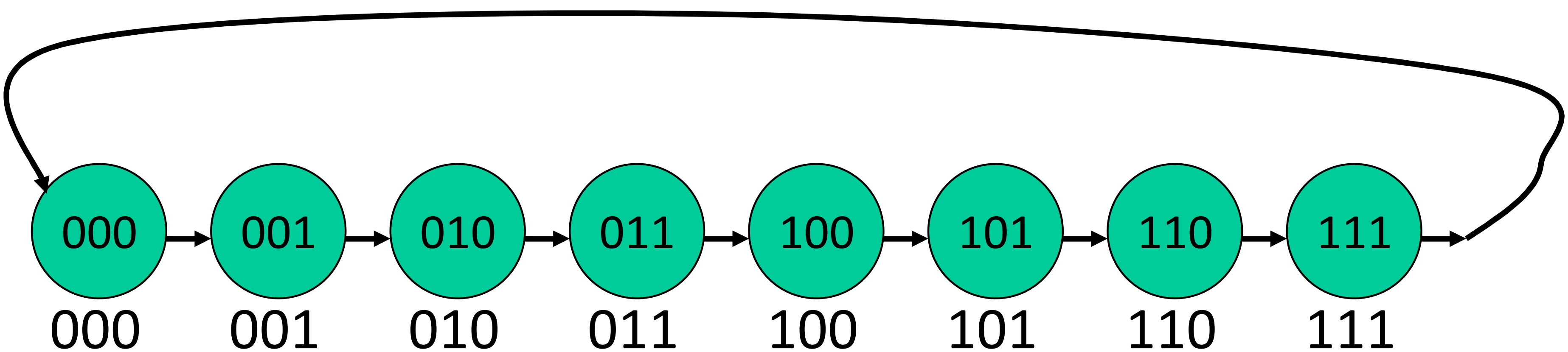
Draw the state diagram: 8 states
  - How many states?
  - How many inputs? 0 (clock)



3-Bit Counter

- Consider a circuit that stores a number and increments the value on every clock edge – on reaching the largest value, it starts again from 0

  Draw the state diagram:
  - How many states?
  - How many inputs?

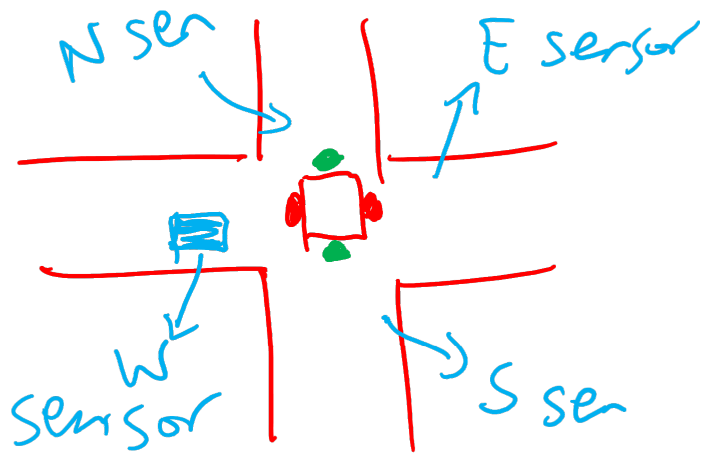


Tackling FSM Problems

- Three questions worth asking:
  - What are the possible output states? Draw a bubble for each.
  - What are inputs? What values can those inputs take?
  - For each state, what do I do for each possible input value? Draw an arc out of every bubble for every input value.

Traffic Light Controller

- Problem description: A traffic light with only green and red; either the North-South road has green or the East-West road has green (both can't be red); there are detectors on the roads to indicate if a car is on the road; the lights are updated every 30 seconds; a light need change only if a car is waiting on the other road



State Transition Table:

How many states? 2 NS or EW

curr state + How many inputs? → 2 EW sen or NS sen

Wait or NWait

2 sensors clubbed together

How many outputs?

State Transition Table

- Problem description: A traffic light with only green and red; either the North-South road has green or the East-West road has green (both can't be red); there are detectors on the roads to indicate if a car is on the road; the lights are updated every 30 seconds; a light must change only if a car is waiting on the other road

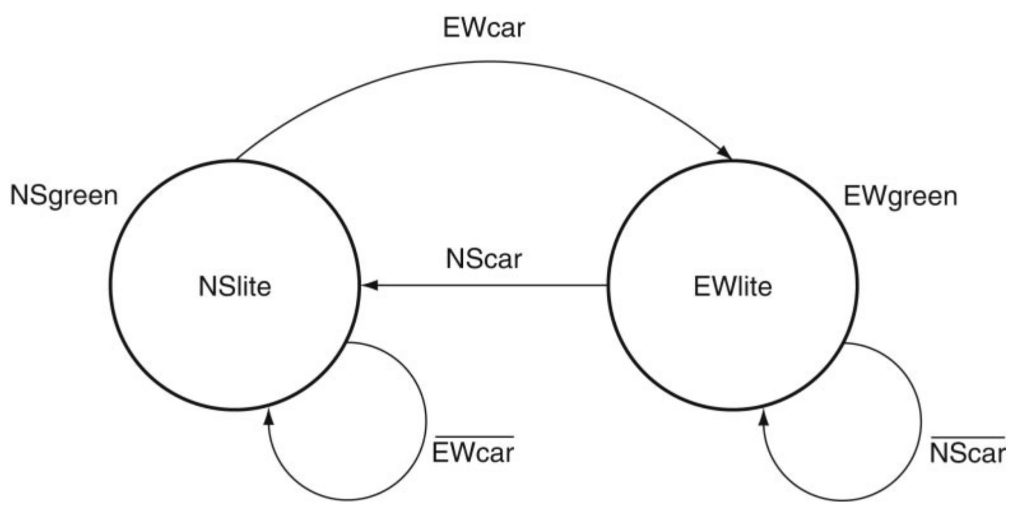
State Transition Table:

CurrState	InputEW	InputNS	NextState=Output
N	0	0	N
N	0	1	N
N	1	0	E
N	1	1	E
E	0	0	E
E	0	1	N
E	1	0	E
E	1	1	N

State Diagram

State Transition Table:

CurrState	InputEW	InputNS	NextState=Output
N	0	0	N
N	0	1	N
N	1	0	E
N	1	1	E
E	0	0	E
E	0	1	N
E	1	0	E
E	1	1	N



Source: H&P textbook