

Flip flops and sequential circuits

Lecture 5 — § 1.6-1.7
Computer Science 218

Mike Feeley

Overview

sequential circuits

- adding state to combinational circuits
- clocks

flip-flops

- a state-holding component
- there are several of them: SR,D,JK and T
- excitation tables

designing a sequential circuit

- state tables and diagrams
- some examples

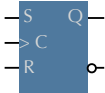
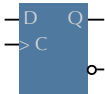
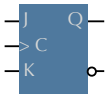
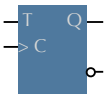
2

Flip flops

one-bit memory element

- latches input for one clock cycle

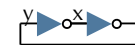
several types

- S 
- D 
- JK 
- T 

Building a flip flop

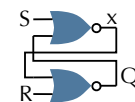
some initial attempts

- consider
 - not good because x is unstable (i.e., $x = x'$)
- a bit better
 - $y = x'$, $x = y'$
 - possible states are $xy=01$ or 10
 - can't set values



how about this?

- if $S=0$ and $R=0$
 - $x=Q'$, $Q=x'$
- if $S=1$ and $R=0$
 - $Q = (R+x)' = (R + (S+Q)')' = 1$
 - $S \rightarrow 0$, value is saved
- if $S=0$ and $R=1$
 - $Q = (R+x)' = (R + (S+Q)')' = 0$
 - $R \rightarrow 0$, value is saved



what if $S=1$ and $R=1$?
 $Q = (1 + (1 + Q)')' = 0$
 $x = (1 + (1 + x)')' = 0$
 don't do this

3

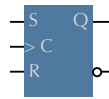
4

SR flip flop

characteristic table

| S | R | Q(t+1) |
|---|---|--------|
| 0 | 0 | Q(t) |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | ? |

S = set
R = reset

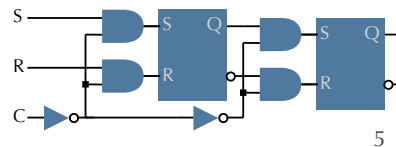
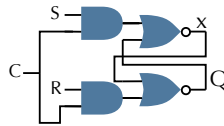


add a clock

- so we can ignore propagation delays
- but when C=1, have the same problem

master-slave SR flip flop

- solves problem
 - when one FF is "pulsed" other is stable
- another approach in the
 - clock-edge triggered



5

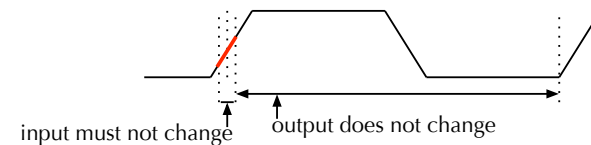
Edge-triggered flip flops

most common way to build synchronized flip-flop

- flip-flop latches on rising (falling) edge of clock
 - rising edge is called positive, falling edge is called negative
- holds value until clock starts next rise (or fall for negative)

how the flip flop latches

- setup time – before transition when input cannot change
- threshold voltage – transition
- hold time – after transition when input cannot change



6

Edge-triggered flip flops

key idea

- ensure stable FF output using narrow window to capture input
- most of the time output is saved FF state
- output=input only for narrow region on clock edge

flip flop latching characteristic values

- threshold voltage
 - voltage at which FF begins to produce new output
- setup time
 - time before threshold voltage is reached when input cannot change
 - allows input voltage to settle in FF
- hold time
 - time after threshold voltage is reached when input cannot change
 - allows input voltage to be reliably captured by FF
- propagational delay
 - time after threshold voltage is reached that it takes FF to output a new value
 - delay ≥ hold time

7

How edge triggering works

exaggerated view of clock cycle

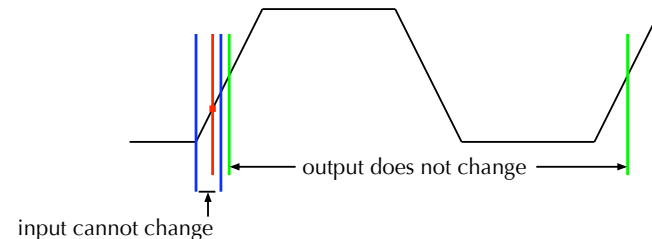


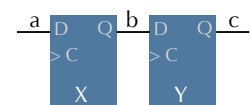
illustration of timing requirements

on each clock "pulse" (rising or falling edge):

- $c_{i+1} = b_i$
- $b_{i+1} = a$

both flip flops are pulsed at once

- X's new output is delayed until after Y's hold time



8

D flip flop

similar to SR

- $D = S + R$
- gets rid of $S=1, R=1$ undefined state

characteristic table

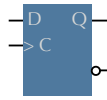
| D | Q(t+1) |
|---|--------|
| 0 | 0 |
| 1 | 1 |

D = data

- $Q(t+1) = D$

to leave state unchanged

- don't pulse the clock



9

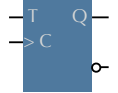
T flip flop

characteristic table

| T | Q(t+1) |
|---|--------|
| 0 | Q(t) |
| 1 | Q'(t) |

T = toggle

- $Q(t+1) = Q(t) \oplus T$



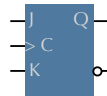
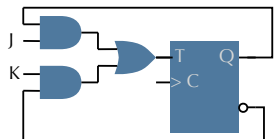
10

JK flip flop

characteristic table

| J | K | Q(t+1) |
|---|---|--------|
| 0 | 0 | Q(t) |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | Q(t)' |

like SR, but with 11 defined



most popular flip flop

11

Flip flop excitation table

describes state transitions of flip-flop

- used when designing circuit using flip flops

excitation tables for the flip flops we know

| Q(t) | Q(t+1) | S | R |
|------|--------|---|---|
| 0 | 0 | 0 | x |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | x | 0 |

| Q(t) | Q(t+1) | J | K |
|------|--------|---|---|
| 0 | 0 | 0 | x |
| 0 | 1 | 1 | x |
| 1 | 0 | x | 1 |
| 1 | 1 | x | 0 |

| Q(t) | Q(t+1) | D |
|------|--------|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

| Q(t) | Q(t+1) | T |
|------|--------|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

12

Sequential circuit behaviour

composed of

- combinational circuit
 - just like before
- flip flops

inputs

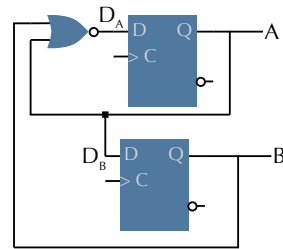
- external inputs to combinational part
- outputs of flip flops
 - e.g., A

outputs

- external outputs
- outputs of flip flops

flip flops

- inputs to flip flops
 - e.g., D_A



13

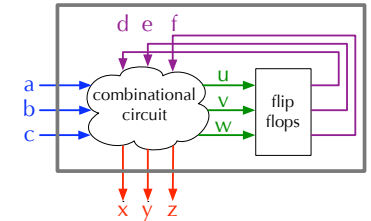
Key idea

sequential circuit is

- combination circuit plus memory (state)
- flip flops are its memory (one per bit)
- output is function of inputs + **current state**

describe it using

- truth table, boolean functions or logic diagram
 - for combinational part (truth table, boolean function, logic diagram)
 - inputs are external inputs plus flop-flop outputs
 - outputs are external outputs plus flip-flop inputs
 - one boolean function for each output in terms of all inputs
- finite state machine
 - for sequential part
 - finite state machine
 - lists all possible states (memory settings)
 - describes how current input and state determine external output and next state



$$\begin{aligned} u &= g(a, b, c, d, e, f) \\ v &= h(a, b, c, d, e, f) \\ w &= i(a, b, c, d, e, f) \\ x &= j(a, b, c, d, e, f) \\ y &= k(a, b, c, d, e, f) \\ z &= l(a, b, c, d, e, f) \end{aligned}$$

14

Understanding them

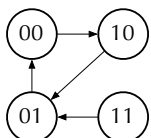
determine flip-flop input equations

- $D_A = (A+B)'$
- $D_B = A$

create state table

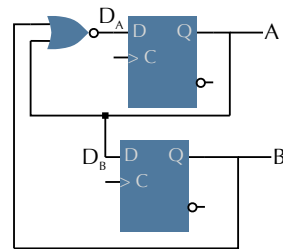
| Q(t) | | Q(t+1) | |
|------|---|--------|---|
| A | B | A | B |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 |

draw state diagram



edges may be labelled a/b

- meaning that input a causes transition
- and output b is the result



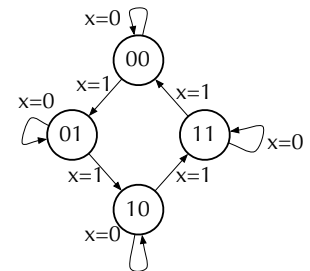
15

Designing a sequential circuit

example: a two-bit counter

- draw a state diagram
- draw excitation table

- left hand columns
 - all combinations of current state + input => next state
- right hand columns
 - one flip flop for each state variable (e.g., A,B)
 - column for each flip-flop input, from excitation table



| Q(t) | | in | Q(t+1) | | flip-flop inputs | | | |
|------|---|----|--------|---|------------------|----------------|----------------|----------------|
| A | B | x | A | B | J _A | K _A | J _B | K _B |
| 0 | 0 | 0 | 0 | 0 | 0 | x | 0 | x |
| 0 | 0 | 1 | 0 | 1 | 0 | x | 1 | x |
| 0 | 1 | 0 | 0 | 1 | 0 | x | x | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | x | x | 1 |
| 1 | 0 | 0 | 1 | 0 | x | 0 | 0 | x |
| 1 | 0 | 1 | 1 | 1 | x | 0 | 1 | x |
| 1 | 1 | 0 | 1 | 1 | x | 0 | x | 0 |
| 1 | 1 | 1 | 0 | 0 | x | 1 | x | 1 |

| Q(t) | Q(t+1) | J | K |
|------|--------|---|---|
| 0 | 0 | 0 | x |
| 0 | 1 | 0 | x |
| 1 | 0 | x | 0 |
| 1 | 1 | x | 0 |

16

Designing a sequential circuit (II)

3. get boolean functions

- for
 - flip-flop inputs and external outputs
- in terms of
 - flip-flop outputs and external inputs
- using k-maps

| Bx | | J _A | | | |
|----|--|----------------|----|----|----|
| A | | 00 | 01 | 11 | 10 |
| 0 | | 0 | 0 | 1 | 0 |
| 1 | | x | x | x | x |

$$J_A = Bx$$

| Bx | | K _A | | | |
|----|--|----------------|----|----|----|
| A | | 00 | 01 | 11 | 10 |
| 0 | | x | x | x | x |
| 1 | | 0 | 0 | 1 | 0 |

$$K_A = Bx$$

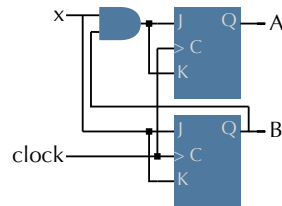
| Bx | | J _B | | | |
|----|--|----------------|----|----|----|
| A | | 00 | 01 | 11 | 10 |
| 0 | | 0 | 1 | x | x |
| 1 | | 0 | 1 | x | x |

$$J_B = x$$

| Bx | | K _B | | | |
|----|--|----------------|----|----|----|
| A | | 00 | 01 | 11 | 10 |
| 0 | | x | x | 1 | 0 |
| 1 | | x | x | 1 | 0 |

$$K_B = x$$

| Q(t) | | in | Q(t+1) | | flip-flop inputs | | | |
|------|---|----|--------|---|------------------|----------------|----------------|----------------|
| A | B | x | A | B | J _A | K _A | J _B | K _B |
| 0 | 0 | 0 | 0 | 0 | 0 | x | 0 | x |
| 0 | 0 | 1 | 0 | 1 | 0 | x | 1 | x |
| 0 | 1 | 0 | 0 | 1 | 0 | x | x | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | x | x | 1 |
| 1 | 0 | 0 | 1 | 0 | x | 0 | 0 | x |
| 1 | 0 | 1 | 1 | 1 | x | 0 | 1 | x |
| 1 | 1 | 0 | 1 | 1 | x | 0 | x | 0 |
| 1 | 1 | 1 | 0 | 0 | x | 1 | x | 1 |



4. draw logic diagram