ECE380 Digital Logic

Flip-Flops, Registers and
Counters:

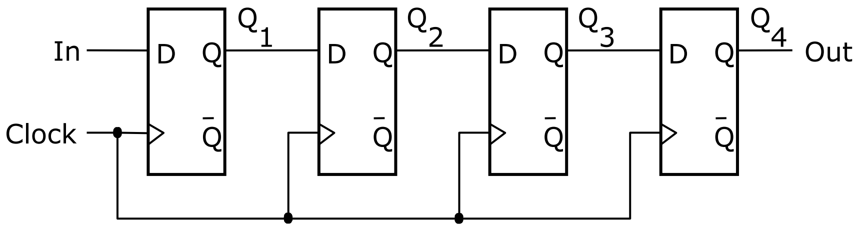
Registers and Counters

Registers

- A flip-flop stores one bit of information
- When a set of *n* flip-flops is used to store *n* bits of data, we refer to these flip-flops as a ***register***
  - Common register usages include
    - Holding a data value output from an arithmetic circuit
    - Holding a count value in a counter circuit
- A common clock signal is typically used for each flip-flop in a register

Shift register

- A register that provides the ability to shift its contents by a single bit
  - May be to the right or left (or possibly both)



Shift right register

Shift right register

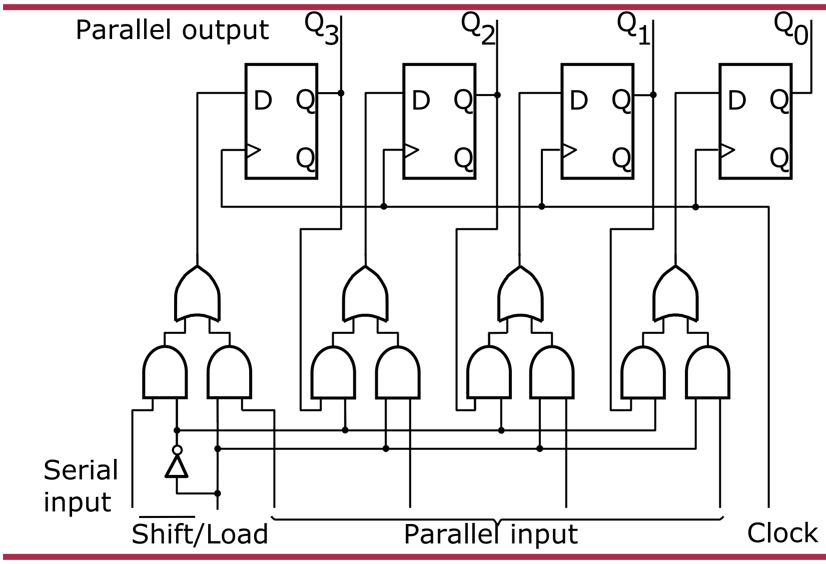
- Data is shifted to the right in a serial fashion using the ***In*** input
- Positive edge triggered
  - Contents of each flip-flop are transferred to the next flip-flop at each positive edge of the clock
- Level sensitive devices would not be appropriate for this circuit

	In	Q_1	Q_2	Q_3	Q_4
t_0	1	0	0	0	0
t_1	0	1	0	0	0
t_2	1	0	1	0	0
t_3	1	1	0	1	0
t_4	1	1	1	0	1
t_5	0	1	1	1	0
t_6	0	0	1	1	1
t_7	0	0	0	1	1

Parallel-access shift register

- Data transfer in computer systems is a common function
  - If the transfer is *n*-bits at a time, the transfer is said to be in ***parallel***
  - If the transfer is *1*-bit at a time, the transfer is said to be ***serial***
- To transfer data serially, data is loaded into a register in parallel (in one clock cycle) and then shifted out one bit at a time
  - Parallel-to-serial data conversion
- If bits are received serially, after *n* clock cycles the contents of a register can be accessed in parallel as an *n*-bit item
  - Serial-to-parallel conversion

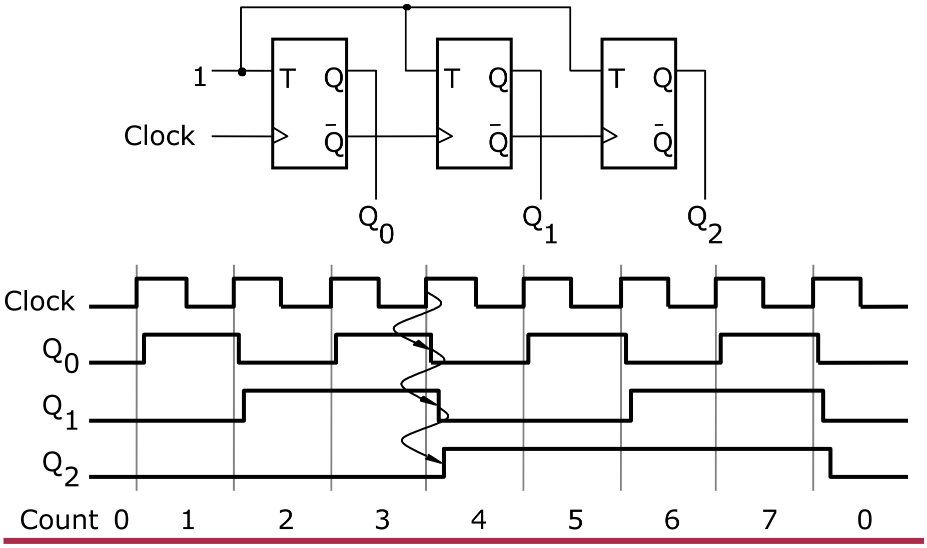
Parallel-access shift register



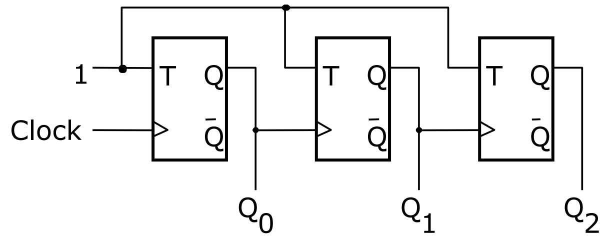
Counters

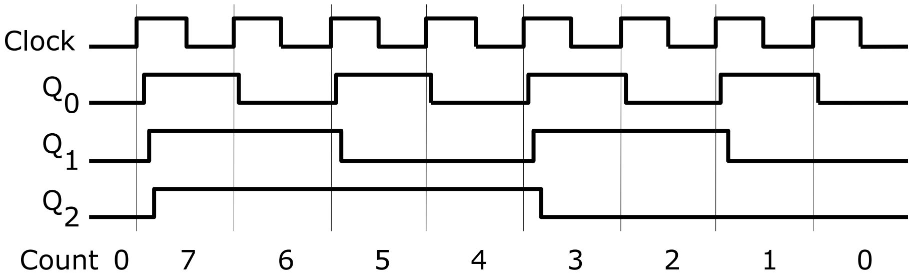
- Special purpose arithmetic circuits used for the purpose of counting
  - Design circuits that can increment or decrement a count by 1
- Counter circuits server many purposes
  - Count occurrences of certain events
  - Generate timing intervals for controlling various tasks in a digital system
  - Track elapsed time between events
- Often (but not always) built with T flip-flops because the toggle feature is naturally suited for implementing the counting operation

Up-counter with T flip-flops



Down counter with T flip-flops





Asynchronous counters

- The previous counters are examples of ***asynchronous*** counters. Also called ***ripple*** counters.
  - Input clock is only connected to one flip-flop
  - Clocks for other flip-flops are (or are derived from) the outputs of the previous flip-flops
- This form of counter is slow because the cascaded clocking scheme
  - The clock source ripples from stage-to-stage
  - The ripple effect is similar to that of a ripple carry adder circuit

Synchronous counters

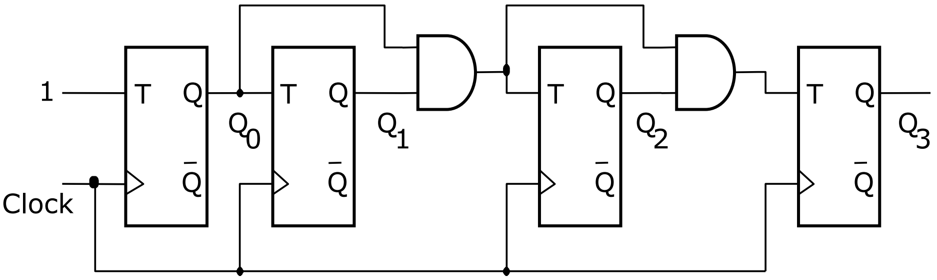
- Synchronous counters are built by clocking all the flip-flops at the same time (with a single clocking source)
  - Faster response than asynchronous counters
- Synchronous counters with T flip-flops
  - Least significant bit, Q_0, changes every clock cycle
  - Bit one, Q_1, only changes when Q_0=1
  - Bit two, Q_2, only changes when Q_0=Q_1=1

	Q_2	Q_1	Q_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

Q_1 changes

Q_2 changes

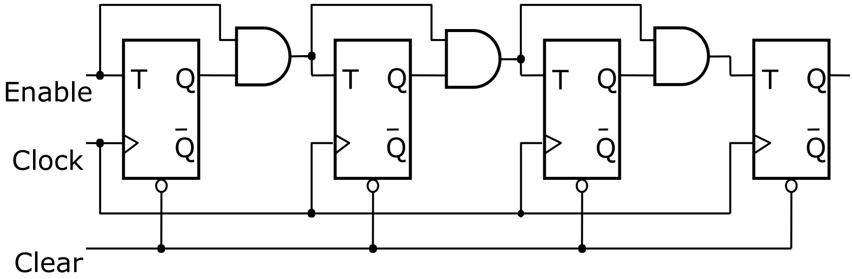
T flip-flop synchronous counter



4-bit synchronous up counter

Enable and clear capability

- It may be desirable to disable counting or clear the counter
  - Include an ***enable*** control signal
  - Use a flip-flop with asynchronous clear capability

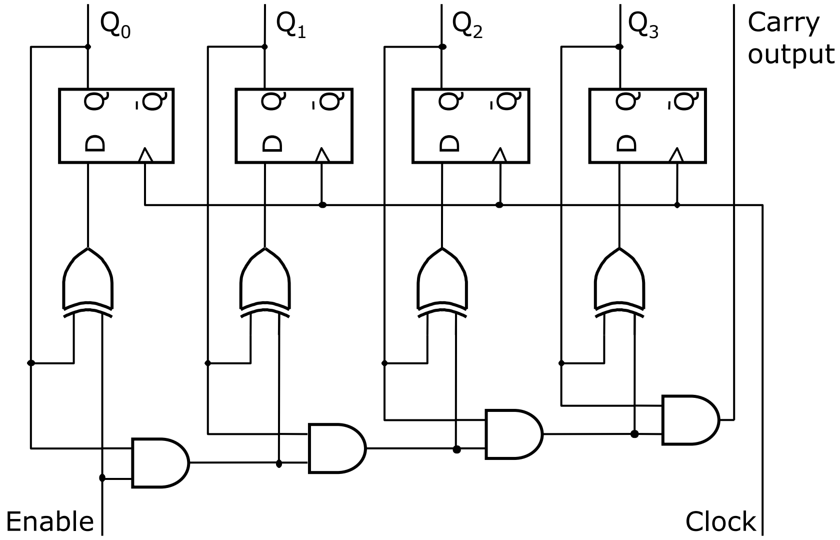


D flip-flop synchronous counter

- A 4-bit up counter counts in the sequence 0,1,2,…,15,0,1…
- The count is given by the flip-flop outputs $Q_3Q_2Q_1Q_0$
- The D inputs are given by:

$$D_0=Q_0\oplus \text{Enable}$$
$$D_1=Q_1\oplus Q_0\cdot \text{Enable}$$
$$D_2= Q_2\oplus Q_1\cdot Q_0\cdot \text{Enable}$$
$$D_3= Q_3\oplus Q_2\cdot Q_1\cdot Q_0\cdot \text{Enable}$$

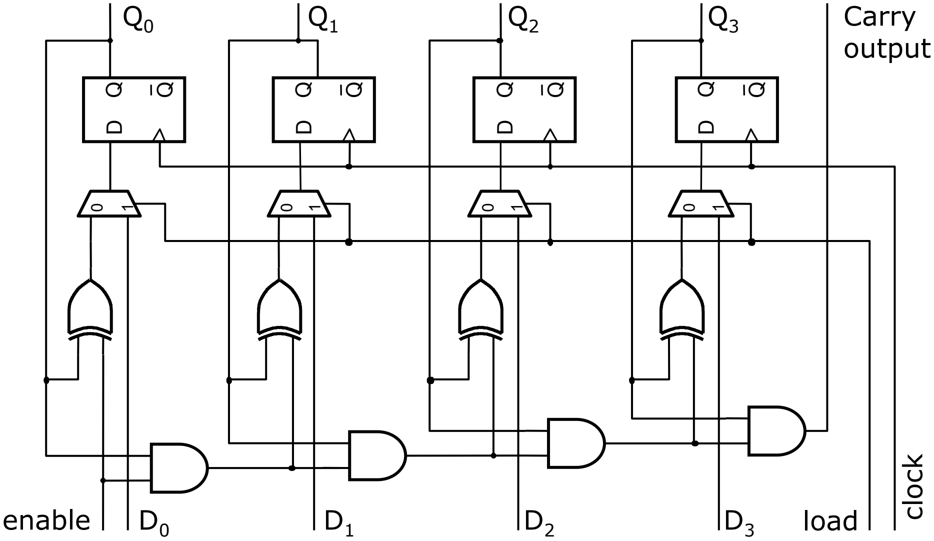
Four-bit counter (D flip-flops)



Counters with parallel load

- It is common for counters to begin a count with a zero value
  - An asynchronous ***clear*** input can be used for his purpose
- It may be desirable for a counter to begin with a non-zero value
- Adding circuitry to provide ***parallel load*** capability is necessary
- A control input, ***load***, is used to select a mode of operation
  - Load=0, circuit counts
  - Load=1, parallel load a new value into the counter

Parallel load counter


Q0
Q1
Q2
Q3
Carry output
Q
Q̄
D
0
1
enable
D0
D1
D2
D3
load
clock