COE 202: Digital Logic Design
Sequential Circuits
Part 4

Dr. Ahmad Almulhem
Email: ahmadsm AT kfupm
Phone: 860-7554
Office: 22-324
Objectives

- Registers
- Shift Registers
- Ripple Counters
Registers

- A register is a group of flip-flops.
- An n-bit register is made of n flip-flops and can store n bits.
- A register may have additional combinational gates to perform certain operations.
4-Bit Register

- A simple 4-bit register can be made with 4 D-FF
- Common Clock
  - At each positive-edge, 4 bits are loaded in parallel
  - Previous data is overwritten
- Common Clear
  - Asynchronous clear
  - When Clear = 0, all FFs are cleared; i.e. 0 is stored.
Question: How to modify this register to enable/disable loading new data (overwriting previous)?
4-Bit Register (cont.)

**Question:** How to modify this register to enable/disable loading new data (overwriting previous)?

**Answer:** When Load=0, the clock input to the FFs will never take a transition (0 to 1, 1 to 0), no new data will be loaded. When Load=1, normal data loading takes place.

This is called **clock gating**.
Clock Skew Problem:
It results from clock gating.

The flip flop gets its clock signal late thus, its output appears late...
4-Bit Register (cont.)

Better Solution: Register with Parallel Load

Use a 2x1 MUX as shown:

\[ D = \overline{\text{Load}} \cdot Q_i + \text{Load} \cdot D_i \]
4-Bit Register (cont.)

A 4-bit Parallel Load Register

When Load = 0, the data is not changed (no loading)

When Load = 1, the data is loaded in parallel at the rising edge (+ve)
Shift Registers

- A **shift register** is a register which shifts its content (right, left, or both)
- Made of flip-flops with common clock
- Useful to load data serially

```
0 1 ... n-1
```
A simple 4-bit shift register can be made with 4 D-FF

Common Clock
- At each positive-edge, 1 bit is shifted in
- Rightmost bit is discarded

Which direction this register is shifting?
Using Shift Register (Examples)

Serial Addition
Using Shift Register (Examples)

Serial Transfer between two registers

(a) Block diagram

(b) Timing diagram

Ahmad Almulhem, KFUPM 2009
Universal Shift Register

**Question:** Design a Universal Shift Register with the following capabilities:

- A *clear* control to clear the register to 0
- A *clock* to synchronize the operations
- A *shift-right* control (associated with serial in/out)
- A *shift-left* control (associated with serial in/out)
- A *parallel-load* control (to parallel load n bits)
- *n-parallel* output lines
- A *control* signal to leave register unchanged

Ahmad Almulhem, KFUPM 2009
Universal Shift Register (cont.)

Parallel outputs

Clear

CLK

$4 \times 1$ MUX

$s_1$

$s_0$

Serial input for shift-right

$I_3$

$I_2$

$I_1$

$I_0$

Parallel inputs

Mode Control

<table>
<thead>
<tr>
<th>$s_1$</th>
<th>$s_0$</th>
<th>Register Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>No change</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Shift right</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Shift left</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Parallel load</td>
</tr>
</tbody>
</table>

Ahmad Almulhem, KFUPM 2009
Universal Shift Register (cont.)

How does it work?

- 4 D-FF and 4 MUXs with selection $S_0, S_1$
- $S_0S_1=00$, FF output is feedback to its input
- $S_0S_1=01$, A FF input is from left FF or serial-in (shift-right)
- $S_0S_1=10$, A FF input is from right FF or serial-in (shift-left)
- $S_0S_1=11$, parallel data transferred in
Ripple Counters

- Instead of having a common clock signal to all Flip Flops, in a Ripple counter the output of one stage (Flip Flop) is connected to the clock input of the next stage.
- T or JK flip flops are used for this construction because of their capability to flip their stored bits.
- Clock is connected to the least significant bit.
- Flip flops are negative edge-triggered (clock is bubbled) – are active when the clock signal is falling (high to low).
- Flip flops invert their stored bits, when the input clock signal goes from high (1) to low (0).
Ripple Counters (cont.)

<table>
<thead>
<tr>
<th>$Q_3$</th>
<th>$Q_2$</th>
<th>$Q_1$</th>
<th>$Q_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Q1: How to make it count down?
Q2: What if we use positive-edge FF?
Q3: What if we use $Q'$ instead of $Q$?
Ripple Counters

- Advantages:
  - Simple hardware

- Disadvantages:
  - Asynchronous – delay dependent

- Good for low power circuits