Registers & Counters

M. Sachdev
Dept. of Electrical & Computer Engineering
University of Waterloo

Registers

- Register is a group of flip-flops
  - $n$-bit register has $n$ flip-flops
  - Can hold $n$ bits of binary data
  - Register may also contain combinational logic

Fig. 6-1: 4-Bit Register
Register with Parallel Load

- Specific control signal to load n-bit data
  - Load = 0, register retains the data
  - Load = 1, register accepts new data

Shift Register

- Capable of shifting data in one or both directions
  - Clock controls the shift operation
- Figure shows a simple shift register with left to right data shifting capability
Serial Data Transfer

- Serial mode → Data is transferred one bit at a time

Serial Addition

- Parallel adders
  - Faster
  - Cost more logic

- Serial adders
  - Slower
  - n-bit addition → n clock cycles
  - Less hardware
Universal Shift Register

<table>
<thead>
<tr>
<th>Mode Control</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>S0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Ripple (Asynchronous) Counter

- Counts the binary sequence
  - Negative edge triggered
  - Output of one flip-flop → Clock to the next
  - Clock skew adds up
### BCD Ripple Counter

- Counter must reset itself after counting the terminal count

![State Diagram of a Decimal BCD-Counter](image)

### Synchronous Counters

- A common clock is applied to all flip-flops
  - Clock skew does not add up
  - Faster than ripple counters
- Synchronous counters can be designed using **sequential circuit procedure**
- Synchronous binary counter

![4-Bit Synchronous Binary Counter](image)
Up-Down Binary Counter

- Can count up (0000 → 1111) or down (1111 → 0000) binary sequence

![4-Bit Up-Down Binary Counter](image)

Synchronous BCD Counter

- Design a synchronous BCD counter with T flip-flops

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Out</th>
<th>Flip-flop inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q8</td>
<td>Q4</td>
<td>Q2</td>
<td>Q1</td>
</tr>
<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>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Home work - Suppose we want to design a counter with 1, 2, 3, 4, 5, 6, 7, 8, 9 sequence (mod-9 counter)?
Counter with Unused States

- A circuit with \( n \) flip-flops has \( 2^n \) states
  - We may have to design a counter with a given sequence (unused states)
  - Unused states may be treated as don’t care or assigned specific next state
  - Outside noise may cause the counter to enter unused state
  - Must ensure counter eventually goes to the valid state

<table>
<thead>
<tr>
<th>Present State</th>
<th>Next State</th>
<th>Flip-flop Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

(a) Logic diagram

(b) State diagram

Fig. 6-16. Counter with Unused States
Ring Counter

Capable of generating different timing signals

Johnson Counter

- Number of states of a ring counter can be doubled
Book Sections – Registers & Counters

- Material is covered in Sections 6.1 – 6.5