Counters

- Special sequential circuits (FSMs) that sequence through a set outputs.
- Examples:
  - binary counter: 000, 001, 010, 011, 100, 101, 110, 111, 000, 001, ...
  - gray code counter: 000, 010, 001, 101, 111, 011, 100, 000, 001, 010, 110, ...
  - one-hot counter: 0001, 0010, 0100, 1000, 0001, 0101, ...
  - BCD counter: 0000, 0001, 0010, ..., 1001, 0000, 0001
  - pseudo-random sequence generators: 10, 01, 00, 11, 10, 01, 00, ...
- Moore machines with "ring" structure to STD:

What are they used?

- Examples from this semester:
  - Clock divider for UART circuit
    \[ \text{16MHz} \rightarrow \text{64} \]
  - UART control (count to 8 between bit capture).
  - Bit-serial multiplier control circuitry (from HW and quiz)
  - In general: counters simplify controller design by
    - providing a specific number of cycles of action,
    - sometimes used in with a decoder to generate a sequence of control signals.

How do we design counters?

- For binary counters (most common case) incremented circuit would work:

```
register
+1
```
- In Verilog, a counter is specified as: `x = x+1;`
  - This does not imply an adder
  - An incrementer is simpler than an adder
  - And a counter is simpler yet.
- In general, the best way to understand counter design is to think of them as FSMS, and follow general procedure. But before that ...

“Ripple” counters

- Each stage is $\div 2$ of previous.
- Look at output waveforms:
- Often called “asynchronous” counters.

Synchronous Counters

- Binary Counter Design:
  Start with 3-bit version and generalize:

\[
\begin{align*}
\text{a} & = a' + b + c \\
\text{b} & = a + b' \\
\text{c} & = a'c + abc + b'c = \text{abc}' + \text{abc} \\
\end{align*}
\]
Synchronous Counters

- How do we extend to n-bits?
- Extrapolate c": d" = d ⊕ abc, e" = e ⊕ abcd
- Has difficulty scaling (AND gate inputs grow with n)
- CE is “count enable”, allows external control of counting,
- TC is “terminal count”, is asserted on highest value, allows
  cascading, external sensing of occurrence of max value.

Binary Counter from JK FFs

Up-Down Counter

With Parallel Load

Odd Counts

- Extra combinational logic can be added to terminate count before
  max value is reached:
- Alternative:
  Example: count to 12
  - Example: count to 12
  - Alternative:

  Example: count to 12
  - Alternative:
Ring Counters

- "one-hot" counters

What are these good for?

Johnson Counter

(a) Two-stage switch-off flip-flop

Sequence of events:

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Input</th>
<th>Action</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 0 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>0 0 1</td>
<td>0 1 0</td>
<td>0 1 0</td>
</tr>
<tr>
<td>3</td>
<td>0 1 0</td>
<td>1 0 0</td>
<td>1 0 0</td>
</tr>
<tr>
<td>4</td>
<td>0 1 1</td>
<td>1 1 0</td>
<td>1 1 0</td>
</tr>
<tr>
<td>5</td>
<td>1 0 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>6</td>
<td>1 0 1</td>
<td>0 1 0</td>
<td>0 1 0</td>
</tr>
<tr>
<td>7</td>
<td>1 1 0</td>
<td>1 0 0</td>
<td>1 0 0</td>
</tr>
<tr>
<td>8</td>
<td>1 1 1</td>
<td>1 1 0</td>
<td>1 1 0</td>
</tr>
</tbody>
</table>

(b) Circuit diagram and truth table