Review Problem 46

Swapped speaker cables using only muxes.

For a stereo, design a crossover box to deal with...
Shift register can be used for serial transfer. Serial: one bit transferred at a time. Parallel: all bits transferred at the same time. 2 modes of communication: Parallel vs. Serial.

Transfer of Data
Shift Register W/Parallel Load
### Table: Output and Receiver

<table>
<thead>
<tr>
<th>Cycle</th>
<th>OP - Sender</th>
<th>OP</th>
<th>Receiver</th>
<th>OP - Receiver</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td></td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>13</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Diagram: Conversion between Parallel and Serial

1. **Parallel Input:**
   - DI
   - Shif
   - DI
   - Shif
   - 4-bit
   - 4-bit
   - 4-bit
   - 4-bit

2. **Serial Output:**
   - DO
   - Reg
   - DO
   - Reg
   - 4-bit
   - 4-bit
   - 4-bit
   - 4-bit

---

Conversion between Parallel & Serial
3-bit binary up-down counter state diagram:

- **3-bit Binary Counter**: Value decreases by 1
- **Down Counter**: Value decreases by 1
- **Up Counter**: Value increases by 1
- **n-bit Binary Counter**: Counts from 0 to $2^n - 1$ in binary
- A reg. that goes through a specific state sequence

Counters
Binary UP-Counter

\[ p_{5} \oplus (p_{5} \oplus p_{5} \oplus p_{5} \oplus \ldots) \]

\[ p_{5} \]

\[ \overbrace{\underbrace{\underbrace{p_{5}}_{N_{5}} N_{5}^{2} N_{5}^{3}}_{N_{5}}}^{N_{5}} \]

\[ \frac{1}{p_{5}} \]
Complex Binary Counter

<table>
<thead>
<tr>
<th>Load Parallel</th>
<th>Reset</th>
<th>Up Count</th>
<th>Old = Old</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Action: Load Count
module (end)

end

end

end

begin

if (reset) n5 = 1, t0, 000, 0

begin

always @(posedge C) begin

d' [3:0] = 0;
reset, load, clk;

input [7:0] in;
input [7:0] in2;

output [7:0] out;
output [7:0] out2;

module reg (out, reset, load, d, clk);


Verilog

Implement the controlled register from lecture in

Review Problem 48
Design a 3-bit counter that goes through the sequence

...<000->010->100->101->110->111->001->011->110->000->...

Arbitrary Sequence Counters
module

endmodule

always @ (posedge clk)
begin

end

option

elsif (out > 0)
begin

elsif (fin > 0)
begin

elsif (fout += 1)
begin

elsif (cout += 1)
begin

input [WIDTH-1:0] out;

input [WIDTH-1:0] out;

input [WIDTH-1:0] out;

input [WIDTH-1:0] out;

input [WIDTH-1:0] out;

module upcounter # (parameter WIDTH = 8) (clk, reset, out);

Counters in Verilog
Memory

ROM: Read-only Memory
Random Access Memory: RAM

64x8 RAM

Address

0110111
1010000
1010101
001000
1010100
111000
1111111
011000
0000000
101000
0111100
001000
0110010
000000
0111100
000000
0111110
000000

Computer programs, data, pictures, etc.
Need method for storing large amounts of data
Change data based on input when row is selected.

Store one bit of data.

Requirements:

RAM Cell