Shifter Example: Approximate Celsius to Fahrenheit Converter

- Convert 8-bit Celsius input to 8-bit Fahrenheit output
  - \( F = C \cdot \frac{9}{5} + 32 \)
  - Approximate: \( F = C \cdot 2 + 32 \)
  - Use left shift: \( F = \text{left shift}(C) + 32 \)

<table>
<thead>
<tr>
<th>C</th>
<th>00001100 (12)</th>
<th>00011000 (24)</th>
<th>00111000 (56)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>( \lll 0 ) (shift in 0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>( 00100000 (32) )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>( \lll 2 )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Net result: shift by 5: \( 01100000 \) (by 1)

Shifter Example: Temperature Averager

- Four registers storing a history of temperatures
- Want to output the average of those temperatures
- Add, then divide by four
  - Same as shift right by 2
  - Use three adders, and right shift by two

Shifter

- A shifter that can shift by any amount
  - 4-bit barrel left shift can shift left by 0, 1, 2, or 3 positions
  - 8-bit barrel left shifter can shift left by 0, 1, 2, 3, 4, 5, 6, or 7 positions
  - Shifting an 8-bit number by 8 positions is pointless -- you just lose all the bits
- Could design using 8x1 muxes and lots of wires
  - Too many wires
- More elegant design
  - Chain three shifters: 4, 2, and 1
  - Can achieve any shift of 0..7 by enabling the correct combination of those three shifters, i.e., shifts should sum to desired amount

Comparators

- N-bit equality comparator. Outputs 1 if two N-bit numbers are equal
  - 4-bit equality comparator with inputs A and B
    - \( a3 \) must equal \( b3 \), \( a2 = b2 \), \( a1 = b1 \), \( a0 = b0 \)
    - Two bits are equal if both 1, or both 0
    - \( \text{eq} = (axor + \text{axor}) \cdot (axor + \text{axor}) \cdot (axor + \text{axor}) \cdot (axor + \text{axor}) \)
    - Recall that XOR outputs 1 if its two input bits are the same
      - \( \text{eq} = (a \oplus b) \cdot (a \oplus b) \cdot (a \oplus b) \cdot (a \oplus b) \)

<table>
<thead>
<tr>
<th>C</th>
<th>00001100 (12)</th>
<th>00010010 (22)</th>
<th>00111000 (56)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>( \lll 0 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>( 01100000 ) (by 4)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Net result: shift by 5: \( 11000000 \) (by 1)
Design Challenge
Not Really a Quiz

- Design a comparator that determines if three 4-bit numbers are equal.

Magnitude Comparator

- By-hand example leads to idea for design
  - Start at left, compare each bit pair, pass results to the right
  - Each stage has 3 inputs indicating results of higher stage, passes results to lower stage

Magnitude Comparator

- How design? Consider how compare by hand. First compare a3 and b3. If equal, compare a2 and b2. And so on.
  - Stop if comparison not equal -- whichever’s bit is 1 is greater. If never see unequal bit pair, A=B.

Magnitude Comparator

- Each stage:
  - \( \text{out}_{lt} = \text{in}_{lt} + (\text{in}_{eq} \; a' \; b) \)
  - \( \text{out}_{eq} = \text{in}_{eq} \; a \; b' \)
  - \( \text{out}_{gt} = \text{in}_{gt} + (\text{in}_{eq} \; a' \; b) \)
- A B (so far) if already determined in higher stage, or if higher stages equal but
  - \( \text{out}_{lt} = \text{in}_{lt} + (\text{in}_{eq} \; a' \; b) \)
  - \( \text{out}_{eq} = \text{in}_{eq} \; a \; b' \)
  - \( \text{out}_{gt} = \text{in}_{gt} + (\text{in}_{eq} \; a' \; b) \)
- Simple circuit inside each stage, just a few gates (not shown)

Magnitude Comparator

- Final answer appears on the right
  - Takes time for answer to "ripple" from left to right
  - Thus called "carry-ripple" style after the carry-ripple adder
  - Even though there’s no "carry" involved
Magnitude Comparator Example: Minimum of Two Numbers

- Design a combinational component that computes the minimum of two 8-bit numbers
  - Solution: Use 8-bit magnitude comparator and 8-bit 2x1 mux
    - If \( A < B \), pass \( B \) through mux. Else, pass \( A \).

Design Challenge

- Design a comparator that determines if three 4-bit numbers are equal, by connecting 4-bit magnitude comparators together and using additional logic if necessary.

  - Due:
    - Next Lecture (Monday, October 10)
    - Extra Credit (Homework)
      - 2 points