Review Problem 19

- Extend this Verilog code to also show the letter "A" on input pattern 1010 (ten) and "F" on pattern 1111 (fifteen).

module seg7 (bcd, leds);
  input logic [3:0] bcd;
  output logic [6:0] leds;

  always_comb begin
    case (bcd)
      // BCD[]
      // 3210
      4'b0000: leds = 7'b0111111;
      4'b0001: leds = 7'b0000110;
      4'b0010: leds = 7'b1011011;
      4'b0011: leds = 7'b1001111;
      4'b0100: leds = 7'b1100110;
      4'b0101: leds = 7'b1101101;
      4'b0110: leds = 7'b1111011;
      4'b0111: leds = 7'b0000111;
      4'b1000: leds = 7'b1111111;
      4'b1001: leds = 7'b1101111;
      default: leds = 7'b1X;
    endcase
  end
endmodule
Review: Circuit Implementation Techniques

- Truth Tables - Case-by-case circuit description
- Boolean Algebra - Math form for optimization
- K-Maps - Simplification technique
- Circuit Diagrams - TTL Implementations
- Verilog – Simulation & Mapping to FPGAs
Combinational Logic Design Process

1. Understand the Problem
   what is the circuit supposed to do?
   write down inputs (data, control) and outputs
   draw block diagram or other picture

2. Formulate the Problem in terms of a truth table or other suitable
   design representation
   truth table, Boolean Algebra, Verilog, etc.

3. Choose Implementation Target

4. Follow Implementation Procedure
   K-maps, Boolean algebra, Quartus synthesis
Process Line Control Example

Statement of the Problem

- Rods of varying length (+/-10%) travel on conveyor belt
- Mechanical arm pushes rods within spec (+/-5%) to one side
- Second arm pushes rods too long to other side
- Rods too short stay on belt

- 3 light barriers (light source + photocell) as sensors

Design combinational logic to activate the arms

Understanding the Problem

- Inputs are three sensors, outputs are two arm control signals
- Assume sensor reads "1" when tripped, "0" otherwise
- Call sensors A, B, C

- Draw a picture!
Where to place the light sensors A, B, and C to distinguish among the three cases?

Assume that A detects the leading edge of the rod on the conveyor
Process Line Control Example (cont.)

A to B distance place apart at specification - 5%
A to C distance placed apart at specification +5%
### Process Line Control Example (cont.)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Meaning</th>
<th>Accept</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No Rod/Start</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>entering</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Inspect/Start</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>entering</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>short</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2 rods</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>In Spec</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Long</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Rule:** Rods must be at least Spec +10% apart.

**Accept** = \(A \cdot B \cdot C^c\)

**Long** = \(A \cdot C\)
Logical Function Unit

Create a unit that can compute the AND, OR, or XOR of two inputs A and B, based upon control lines C0 and C1.

\[
\begin{array}{ccc|c}
C_0 & C_1 & A & B & F \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & X \\
1 & 0 & 0 & 1 & X \\
1 & 1 & 0 & 0 & X \\
1 & 1 & 0 & 1 & X \\
\end{array}
\]

\[F = AB\overline{C_1} + AB\overline{C_0} + \overline{A}BC_0\]

Similar to the main computation unit in a Microprocessor
Logical Function Unit (cont.)

- Implementation: