L5 – Sequential Circuit Design
Sequential Circuit Design

- Mealy and Moore
- Characteristic Equations
- Design Procedure
- Example Sequential Problem – from specification to implementation

- Ref: Unit 14 of text
Types of State Machines

- **Mealy Machine**

  Characterized by – Outputs are a function of both inputs and current state.
Types of State Machines

- **Moore Machine**

Characterized by – Outputs are a function of the current state only.
Notes on Mealy and Moore

- Both Mealy and Moore machine implementation can be implemented with any sequential element.

- Why choose one elements over another?
  - Efficiency – The next state logic may differ significantly when using different F/F types.
  - Efficiency of implementation is also drastically affected by choice of state assignment.
The characteristic equation

- The *Characteristic Equation* formally specifies the flip-flop’s next state as a function of its current state and inputs.
- \( Q^* \) means the next state value for the \( Q \) output of the F/F.
### Characteristic equations for F/Fs

- **Ref: Lect 1**
- **S-R Latch**
  - \( Q^* = S + R' Q \)
- **D Latch**
  - \( Q^* = D \)
- **D F/F**
  - \( Q^* = D \)
- **D F/F with Enable**
  - \( Q^* = EN D + EN' Q \)
- **J-K F/F**
  - \( Q^* = J Q' + K' Q \)
- **T F/F**
  - \( Q^* = Q' \)
  - when \( T = 1 \)
Summary of the Design Procedure

- 1. Given the problem statement, determine the relationship between input and output. Understand the specification and or problem statement. Resolve any questions. Then generate a state graph and/or state table.
- 2. Reduce the state table to the minimum number of states.
- 3. From the number of states determine the number of flip-flops (m states \( \rightarrow \) n flip-flops where m \( \leq 2^n \))
- 4. Generate a transition table (current state – next state)
- 5. Use K-maps to derive flip-flop input equations.
- 6. Derive output functions and implement.
Example – problem statement

- Sequential Code Converter (16.3 example)
- Word description: Design a sequential circuit to convert BCD to excess 3 code. The inputs arrive sequentially, lsb first, i.e. serial input stream. After 4 inputs the circuit resets to the initial state ready for another group of 4 inputs. The excess 3 code is output serially at the same time.
- First question – is it possible to generate the output serially without delay?
## Input – output table

- Input BCD – Output excess 3

<table>
<thead>
<tr>
<th>$X$ Input (BCD)</th>
<th>$Z$ Output (excess-3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_3$</td>
<td>$t_2$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Construct a state Graph

- Walk through the sequences
Build a state table

- From the State Graph can build the state table
- Note the relationship between the two
Then reduce the state table

- And just how is that done – (the coming attraction)

- How many flip-flops are needed?
What next?

- Choose state assignment
- Pick flip-flop of implementation – here D F/Fs
Next state logic

- Logic to generate the next state is generated
- Use K-maps

(a) Assignment map
(b) Transition table
From there: implement the design

- D flip-flop implementation
Implementation with other F/Fs?

- Use the characteristic equation for generation of the transition table.

- Say T flip-flops – example for T1
  - $000 \rightarrow 100 \ (x=0) \ 101 \ (x=1)$

\[
T1 = Q1' Q2' + Q1 Q2
\]
Assignment

- Work through the problem in this lecture on your own and be comfortable with them.