Synthesis

- Synchronous Sequential Circuits synthesis procedure
  - Word description of problem /hardest; art, not science /
  - Derive state diagram & state table
  - Minimize /moderately hard /
  - Assign states /very hard, NP-complete problem /
  - Produce state & output transition tables
  - Determine what FFs to use and find their excitation maps
  - Derive output equations/K-maps
  - Obtain the logic diagram
Synthesis

- Types of Sequential Circuits
  - Completely/incompletely specified
Synthesis

- Example:
  - Find D FF realization of circuit defined in table (a)
  - (b): state assignment
  - (c): transition table
  - (d): output K-map
  - (e): excitation K-map
Synthesis

• Example solution:
  – Logic diagram

\[ D_1 = y_1 \bar{y}_2 + xy_2 \]
\[ D_2 = \bar{x}y_1 + x\bar{y}_1 = x \oplus y_1 \]
\[ z = x\bar{y}_1 y_2 + \bar{x}y_1 \bar{y}_2 \]
### Synthesis

- **Q(t)**: state of FF when clock signal is activated
- **Q(t+ε)**: state of FF after clock signal has been activated

#### Table:

<table>
<thead>
<tr>
<th>State transitions</th>
<th>Required inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q(t)</td>
<td>Q(t+ε)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(a) D flip-flop table.

<table>
<thead>
<tr>
<th>State transitions</th>
<th>Required inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q(t)</td>
<td>Q(t+ε)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(b) Clocked SR flip-flop table.

<table>
<thead>
<tr>
<th>State transitions</th>
<th>Required inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q(t)</td>
<td>Q(t+ε)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(c) Clocked T flip-flop table.

<table>
<thead>
<tr>
<th>State transitions</th>
<th>Required inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q(t)</td>
<td>Q(t+ε)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(d) Clocked JK flip-flop table.

Flip-flop input tables. (a) D flip-flop. (b) Clocked SR flip-flop. (c) Clocked T flip-flop. (d) Clocked JK flip-flop.
Synthesis

- Example is same as before, but use JK FFs
- (a): transition table; (b): Excitation tables; (c): Excitation maps
Synthesis

- Example JK FF solution:
  - Logic diagram
Design a synchronous sequential circuit with one input $x$ and one output $z$ that recognizes the input sequence 01.

This circuit can be used to recognize a 0 to 1 transition on the input $x$. In other words, the circuit should produce an output sequence $z = 01$ whenever the input sequence $x = 01$ occurs. For example, if the input sequence is

$$x = 010100000111101$$

then the output sequence will be

$$z = 010100000100001$$
Synthesis Example

(a)  

(b)  

(c)  

(d)  

Sequence Recognizers

Design a synchronous sequential circuit with one input $x$ and one output $z$ that recognizes an input sequence of output values $y^k$.
Synthesis Example

• 01 recognizer timing diagram

\[ S = \bar{x} \]
\[ R = x \]
\[ z = xy^k \]
Let us design a synchronous sequential circuit with one input line and one output line that recognizes the input string $x = 1111$. The circuit is also required to recognize overlapping sequences, as can be seen in the output string $z$ that results from the following input string $x$:

$$x = 1101111111010$$

$$z = 0000001111000$$
Synthesis Example

- State table, transition table and output map
Synthesis Example

- K-maps for a clocked SR realization
Here we want to design a clocked sequential circuit that recognizes the input sequence consisting of exactly two zeros followed by a 10. In other words, the following output sequence should result from the given input sequence.

\[
x = 001001000010010
\]

\[
z = 000100100001001
\]
Synthesis Example

- Feedback for overlapping sequences
- Complete state diagram
- State table
- Reduced state table
- State diagram
Let us design a control unit for a simple coin-operated candy machine. Candy costs 20 cents, and the machine accepts nickels and dimes. Change should be returned if more than 20 cents is deposited. No more than 25 cents can be deposited on a single purchase; therefore, the maximum change is one nickel.
Synthesis Example

- Candy machine state diagram