Digital Logic Design
ENEE 244-010x

Lecture 22

Announcements

- Make sure you are up-to-date with material covered last time (on 11/25).
- Homework 9 up on course webpage, due on Monday, 12/7
  - Final homework assignment
- Please fill out Course Evaluations online.
  - Class time on Monday, 12/7
  - Make sure to bring in laptop, phone, etc.

Agenda

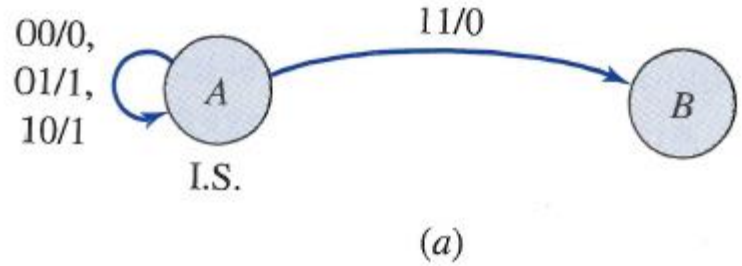
- Last Time:
  - Modeling Clocked Synchronous Sequential Network Behavior (7.3)
  - Started State Table Reduction (7.4)
- This time:
  - Quick review of state diagrams (7.3)
  - State Table Reduction (7.4)
  - The State Assignment (7.5)
  - Completing the Design of Clocked Synchronous Sequential Networks (7.6)

Modeling clocked synchronous sequential network behavior

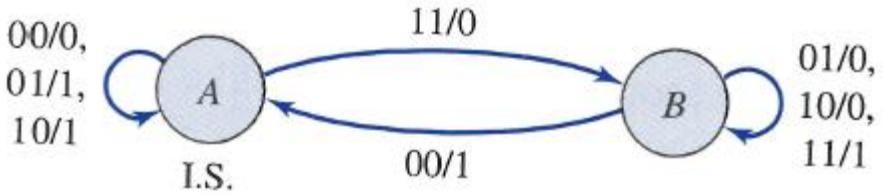
- Approach for the synthesis of clocked synchronous sequential networks:
  - State table/state diagram is constructed from word specifications.
  - State reduction technique to obtain a state table with minimum number of states.
  - Transition table is formed by coding the states of the state table.
  - Excitation table is constructed based on the flip-flop types to be used.
  - From the excitation table, the excitation and output expressions for the network are determined.
  - Finally, the logic diagram is drawn.

Examples of Modeling Step

State Diagram for Mealy serial binary adder



(a)



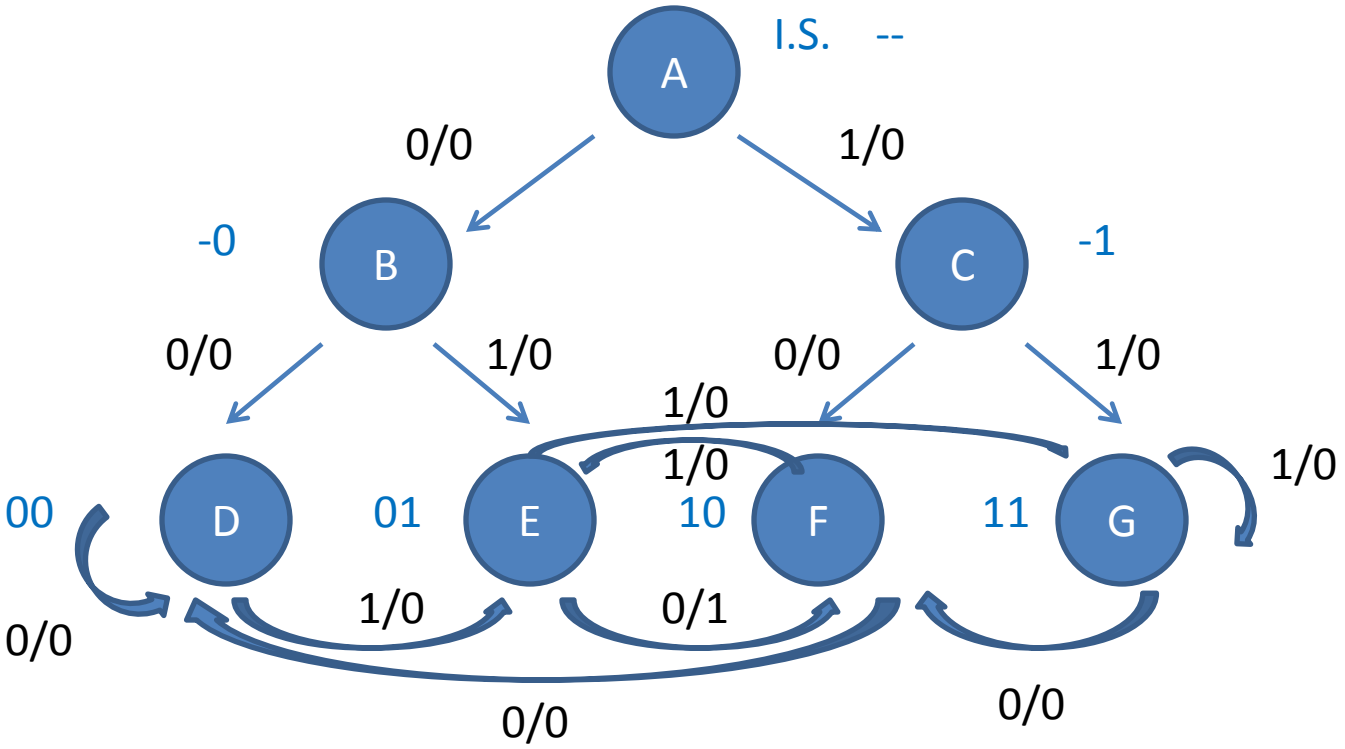
(b)

Figure 7.12 Obtaining the state diagram for a Mealy serial binary adder. (a) Partial state diagram. (b) Completed state diagram.

State Diagram for a Sequence Recognizer

- Network produces a 1 output iff the current input and the previous three inputs correspond to 010.
- 010 Sequence Recognizer
- The 1 output is to occur at the time of the third input of the recognized sequence. Outputs of 0 are to be produced at all other times.
- A Mealy network model is developed since the output is a function of the current input x.
- Network is not required to reset upon the occurrence of the fourth input.
- Sequences may overlap.

State Diagram for a Sequence Recognizer (010)



Can we reduce the number of states?
How do we know?

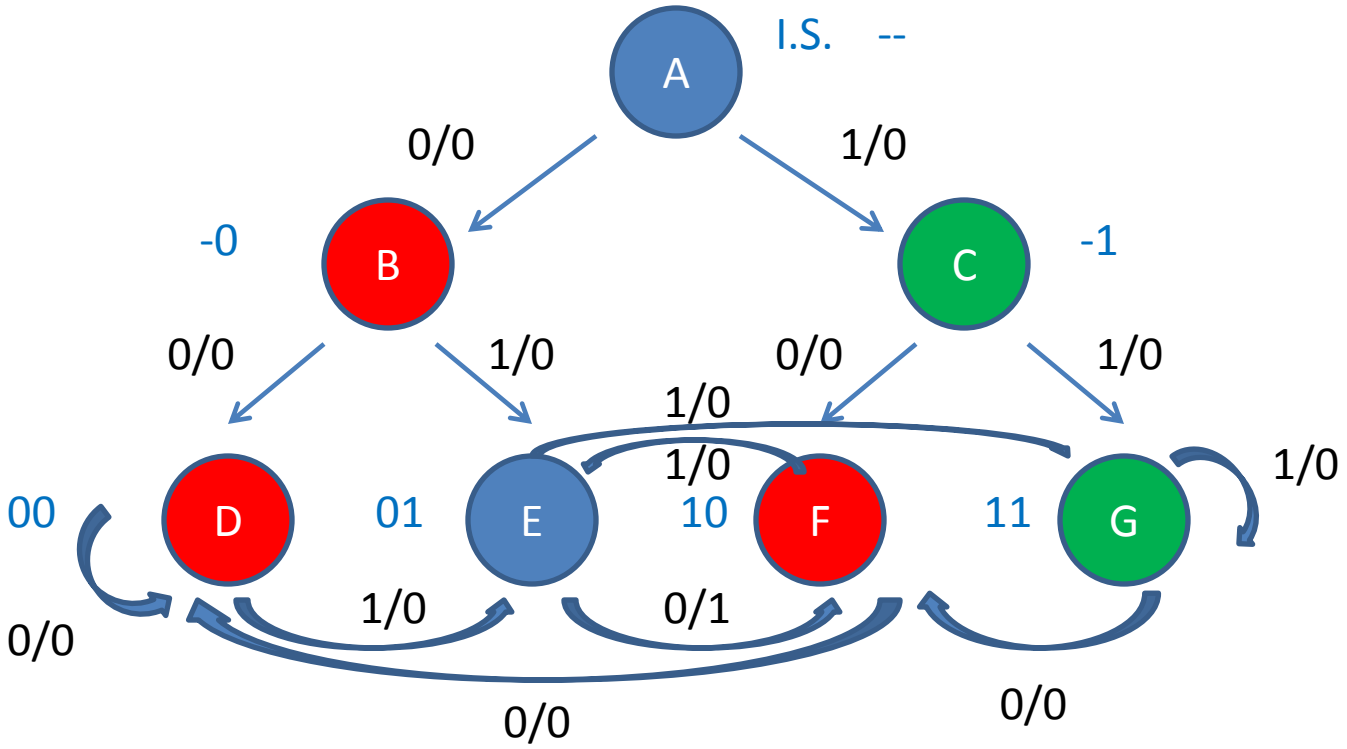
Determining Equivalent Pairs of States

Theorem:

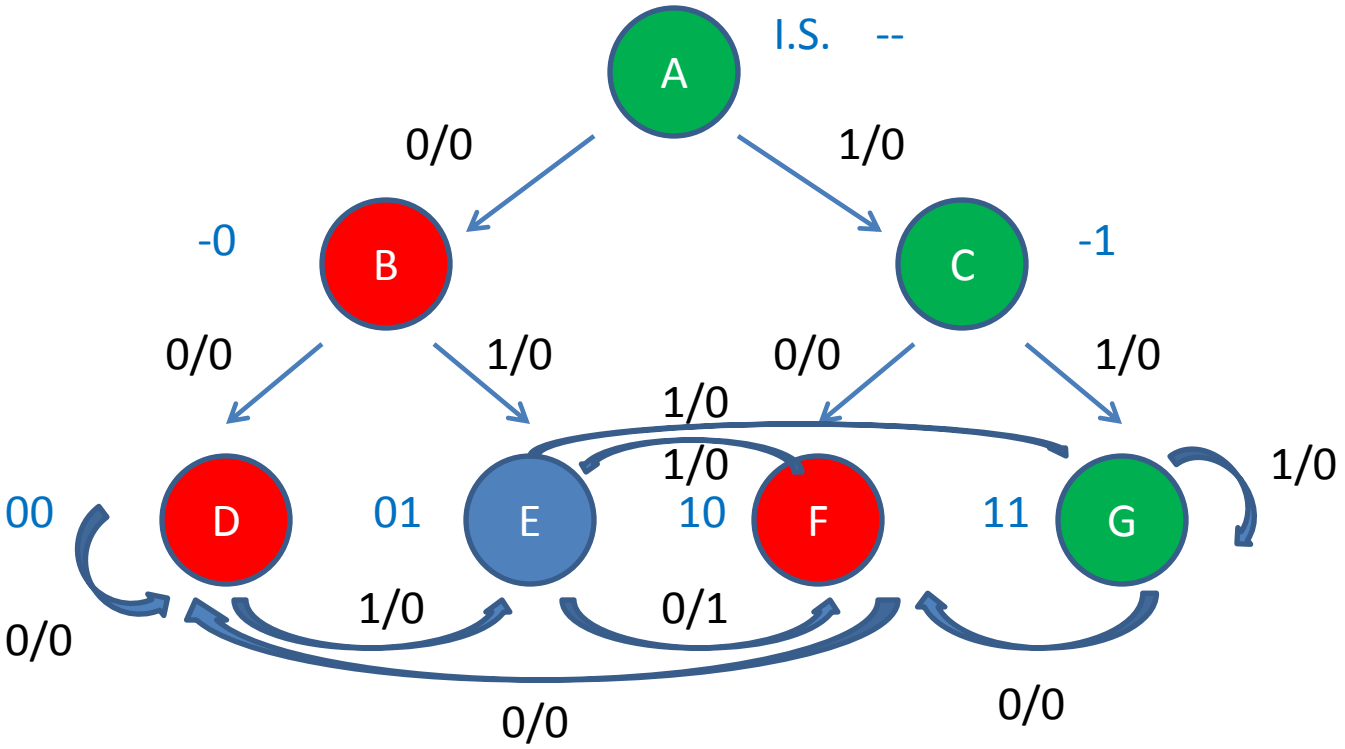
Two states p and q of a clocked synchronous sequential network are equivalent iff for each combination of values of the input variables

1. Their outputs are identical
2. Their next states are equivalent

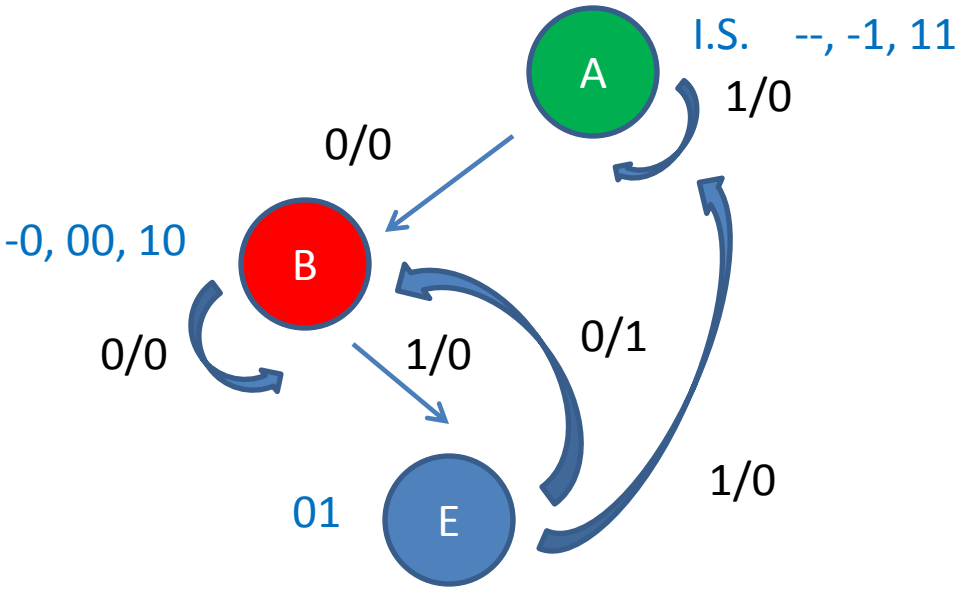
State Diagram for a Sequence Recognizer (010)



State Diagram for a Sequence Recognizer (010)



State Diagram for a Sequence Recognizer (010)



Algorithm for Determining Equivalent Pairs of States

- Uses an implication table

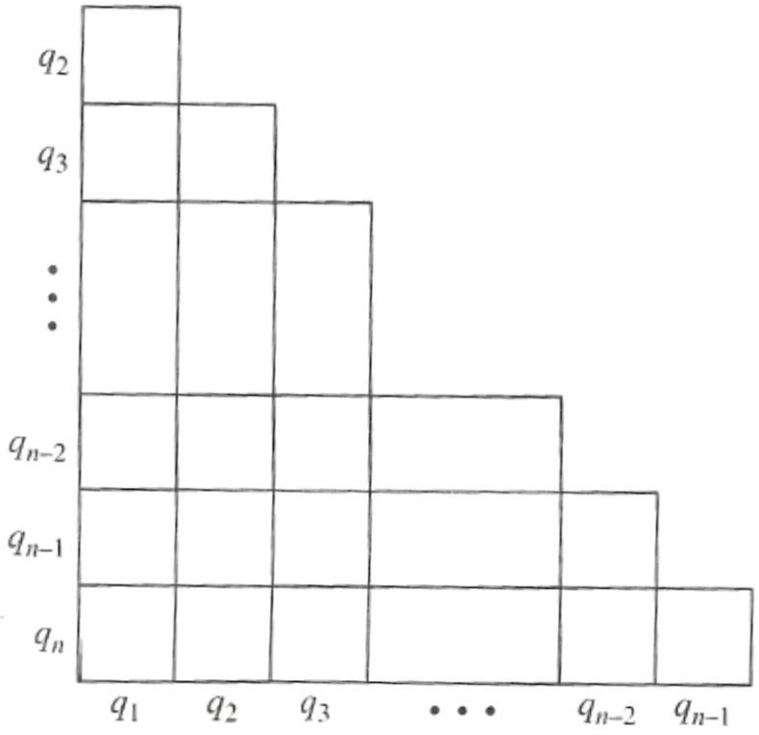


Figure 7.19 The structure of an implication table.

$q_1, \ldots, q_n$ are the states of the state table. There is one cell in the implication table for each pair of distinct cells.

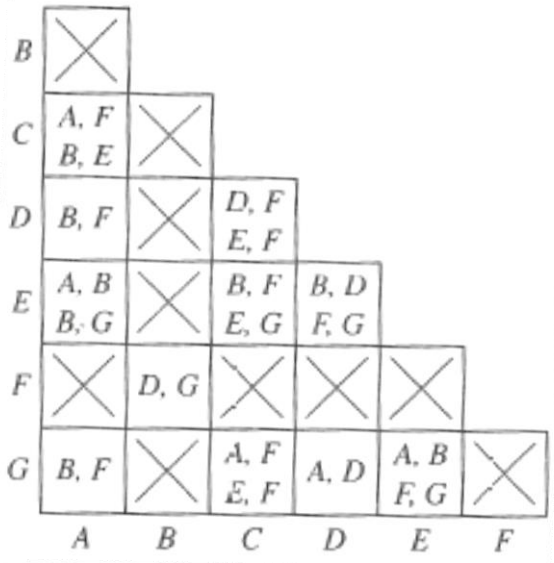
Algorithm for Determining Equivalent Pairs of States

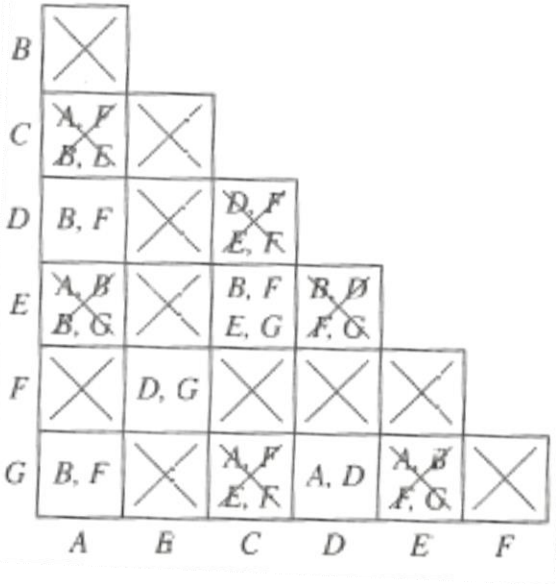
1. Place a × in the (q_i, q_j)-cell if the outputs are contradictory for some input. If there are no contradictory outputs then enter the pair of next states for each input. If neither a × nor pairs of states are entered in the cell, then a check mark is inserted (denoting equivalence of the two states).
2. All state pair entries are inspected by the following process:
   - If (q_a, q_b) is an entry in the (q_i, q_j)-cell and if the (q_a, q_b)-cell contains an × then an × is placed in the the (q_i, q_j)-cell and all other entries are ignored.
   - Otherwise, process is repeated on one of these other state pairs.
   - **Next-state pairs of the form $(q_i, q_j), (q_j, q_i)$ or (q_k, q_k) are not entered.
3. Repeat Step 2 until it is possible to make an entire pass of the implication table without any additional × being entered. If the (q_i, q_j)-cell has no × at this time, then $q_i \equiv q_j$.

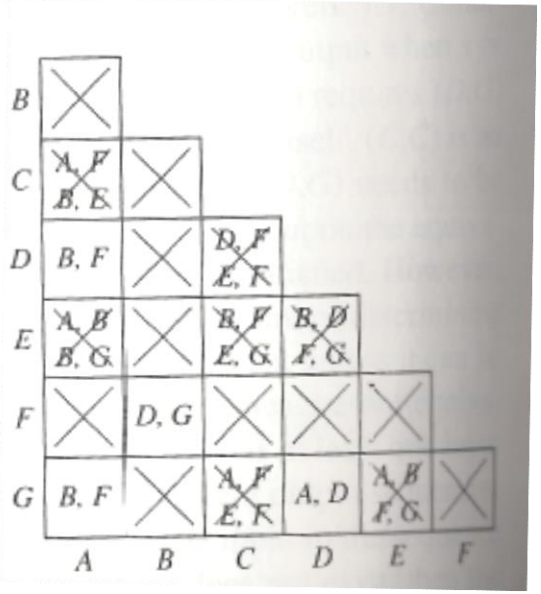
Example of Algorithm

Table 7.13 Example of a state table in which state reduction can be performed

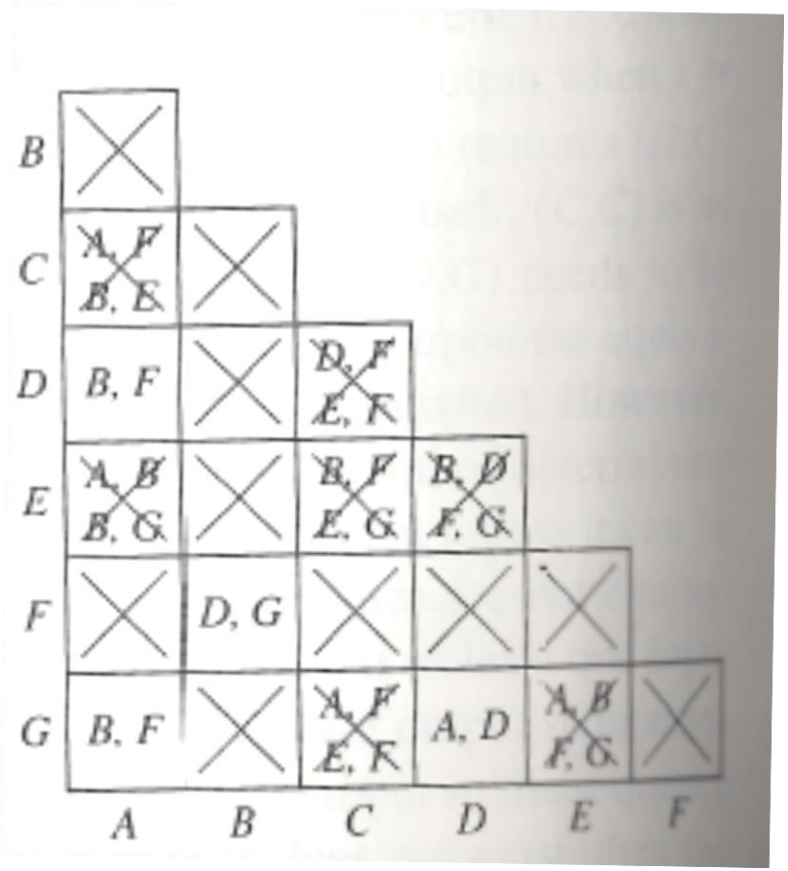
Present state	Next state		Output (z)	
	Input (x)		Input (x)	
	0	1	0	1
*A	A	B	0	0
B	D	C	0	1
C	F	E	0	0
D	D	F	0	0
E	B	G	0	0
F	G	C	0	1
G	A	F	0	0







Example for Implication Table



1. (D, G)
2. $(D, G), (B, F)$
3. $(A, D, G), (B, F)$
4. $(A, D, G), (B, F), (C), (E)$

Constructing the Minimal State Table

Table 7.13 Example of a state table in which state reduction can be performed

Present state	Next state		Output (z)	
	Input (x)		Input (x)	
	0	1	0	1
*A	A	B	0	0
B	D	C	0	1
C	F	E	0	0
D	D	F	0	0
E	B	G	0	0
F	G	C	0	1
G	A	F	0	0

Table 7.14 Minimal state table for Table 7.13

Present state		Next state		Output (z)	
		Input (x)		Input (x)	
		0	1	0	1
(A,D,G):	*α	α	β	0	0
(B,F):	β	α	γ	0	1
(C):	γ	β	δ	0	0
(E):	δ	β	α	0	0