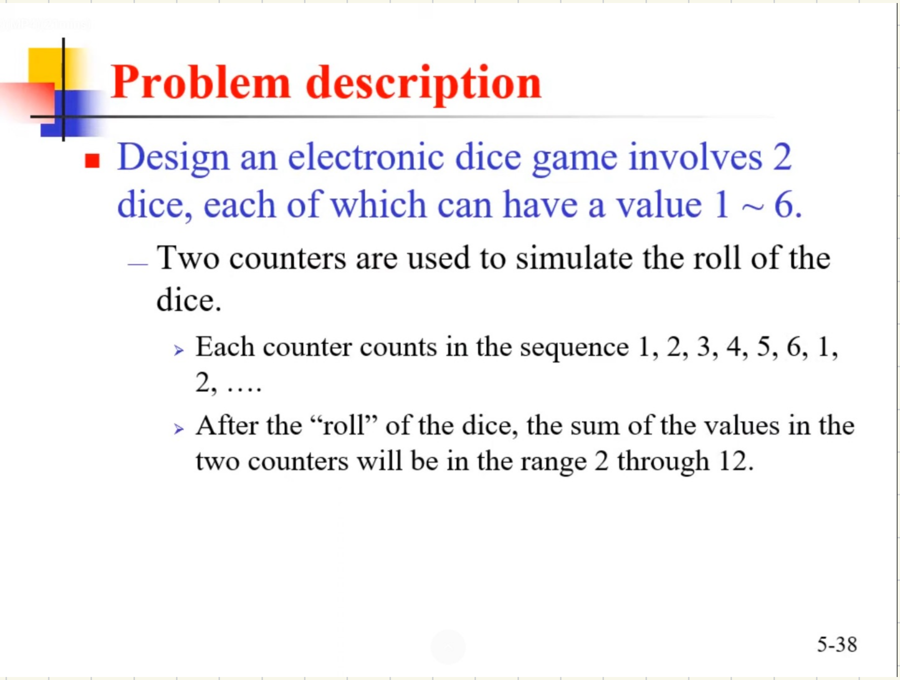
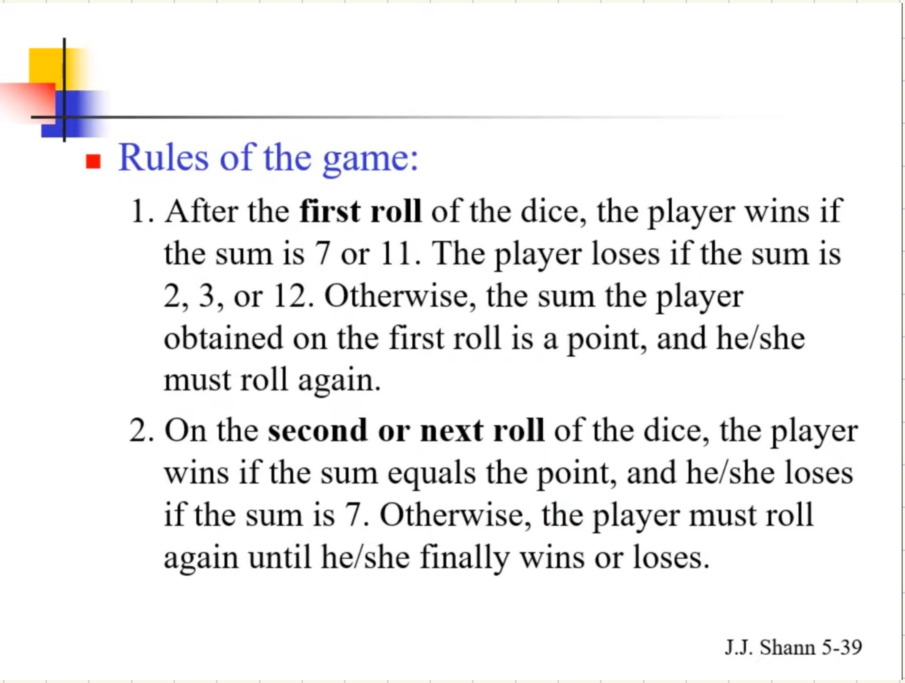
5.2.2 Dice Game

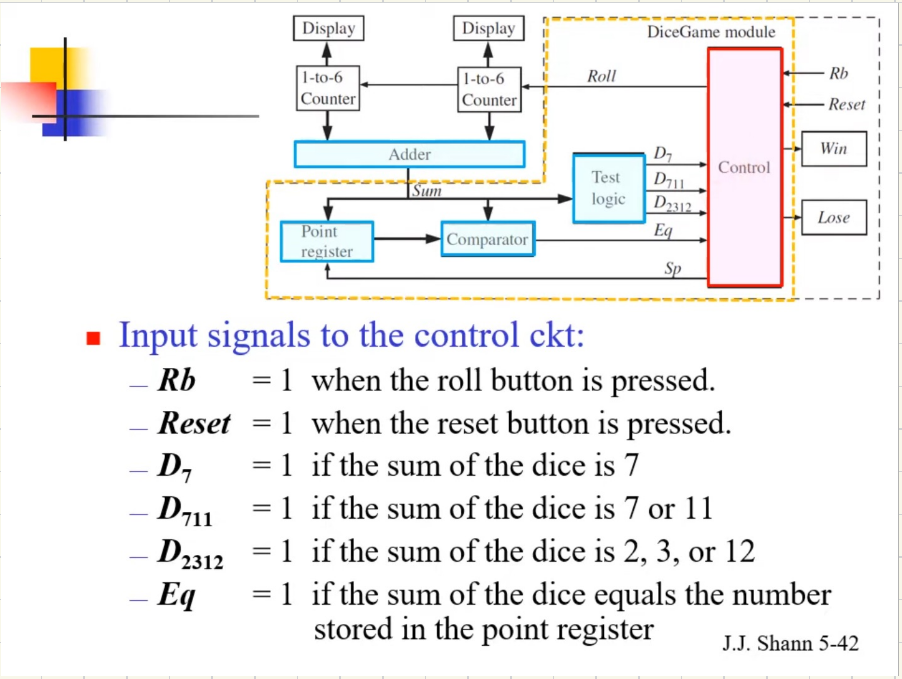


Problem description

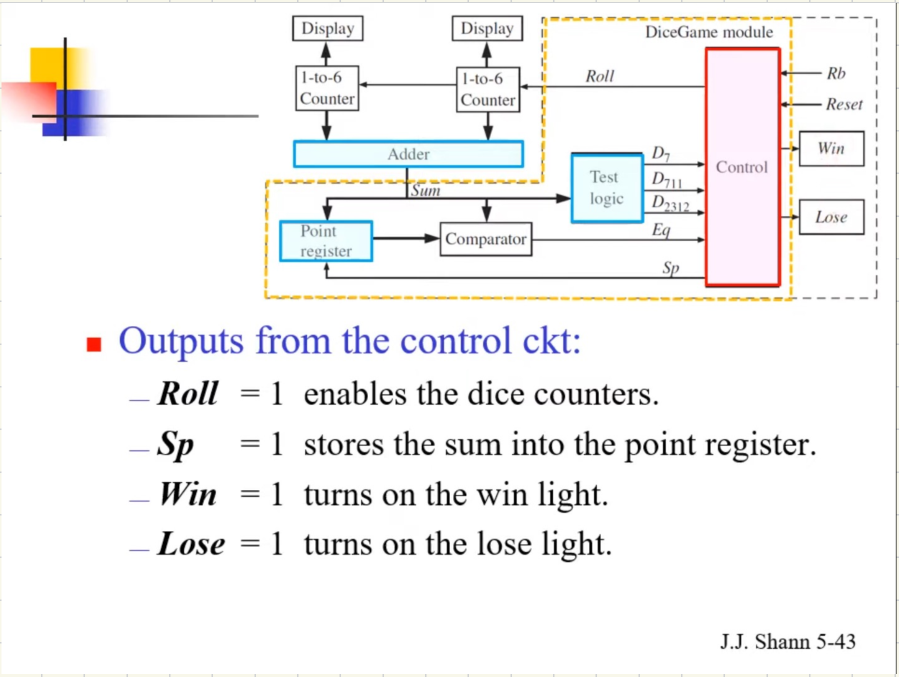
- Design an electronic dice game involves 2 dice, each of which can have a value 1 ~ 6.
  - Two counters are used to simulate the roll of the dice.
    - Each counter counts in the sequence 1, 2, 3, 4, 5, 6, 1, 2, ….
    - After the "roll" of the dice, the sum of the values in the two counters will be in the range 2 through 12.



- Rules of the game:
  1. After the **first roll** of the dice, the player wins if the sum is 7 or 11. The player loses if the sum is 2, 3, or 12. Otherwise, the sum the player obtained on the first roll is a point, and he/she must roll again.
  2. On the **second or next roll** of the dice, the player wins if the sum equals the point, and he/she loses if the sum is 7. Otherwise, the player must roll again until he/she finally wins or loses.



- Input signals to the control ckt:
  - ***Rb*** = 1 when the roll button is pressed.
  - ***Reset*** = 1 when the reset button is pressed.
  - ***D_7*** = 1 if the sum of the dice is 7
  - ***D_{711}*** = 1 if the sum of the dice is 7 or 11
  - ***D_{2312}*** = 1 if the sum of the dice is 2, 3, or 12
  - ***Eq*** = 1 if the sum of the dice equals the number stored in the point register



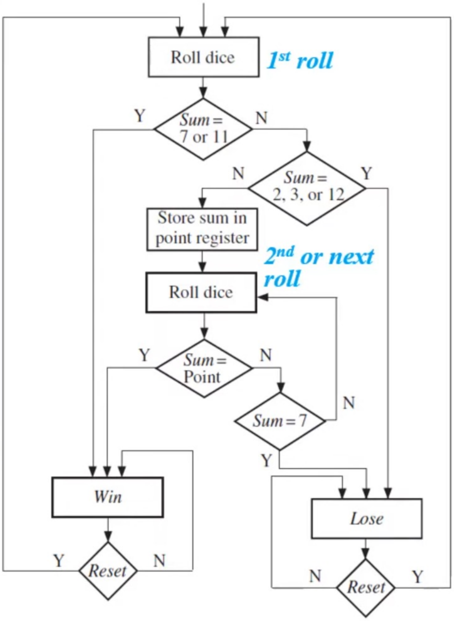
- Outputs from the control ckt:
  - ***Roll*** = 1 enables the dice counters.
  - ***Sp*** = 1 stores the sum into the point register.
  - ***Win*** = 1 turns on the win light.
  - ***Lose*** = 1 turns on the lose light.

Flowchart

- Input signals to the control ckt:
  - ***Rb*** = 1 when the roll button is pressed.
  - ***Reset*** = 1 when the reset button is pressed.
  - D_7 = 1 if the sum of the dice is 7
  - D_{711} = 1 if the sum of the dice is 7 or 11
  - D_{2312} = 1 if the sum of the dice is 2, 3, 12
  - ***Eq*** = 1 if the sum of the dice equals the # stored in the point register
- Outputs from the control ckt:
  - ***Roll*** = 1 enables the dice counters.
  - ***Sp*** = 1 stores the into the point reg.
  - ***Win*** = 1 turns on the win light.
  - ***Lose*** = 1 turns on the lose light.

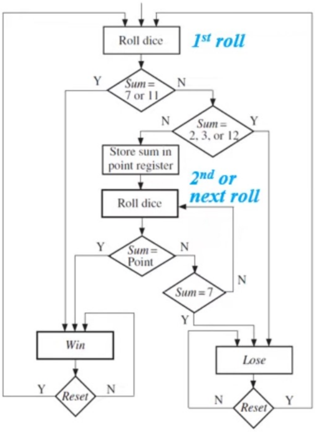
* Assumption:

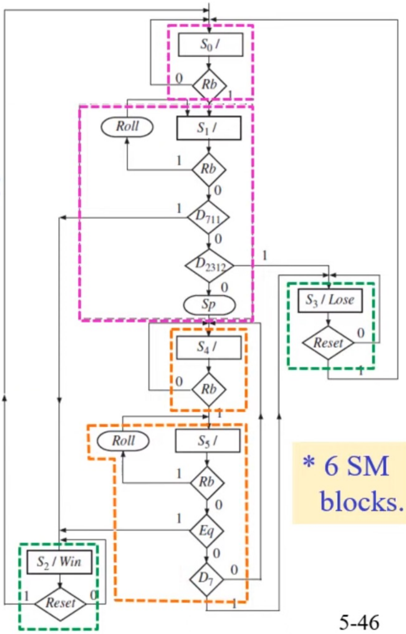
The push buttons are properly debounced and the changes in ***Rb*** are properly synchronized w/ the clock.



SM Chart

- Conversion of flowchart to SM chart:

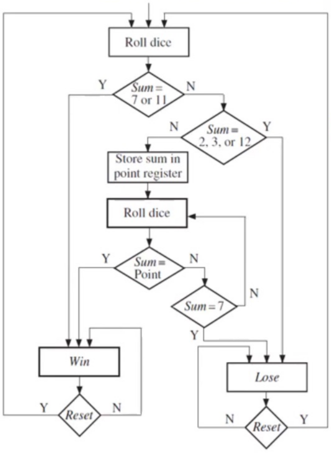


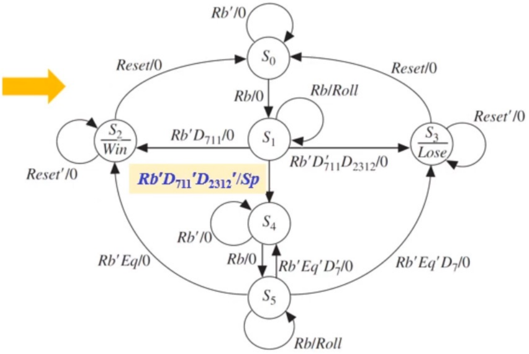


* 6 SM blocks.

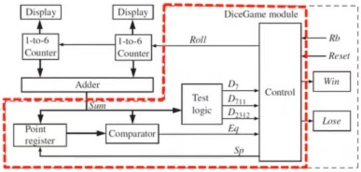
State Graph

- Constructing an equivalent ***state graph*** from the flowchart: an alternative of using ***SM chart***

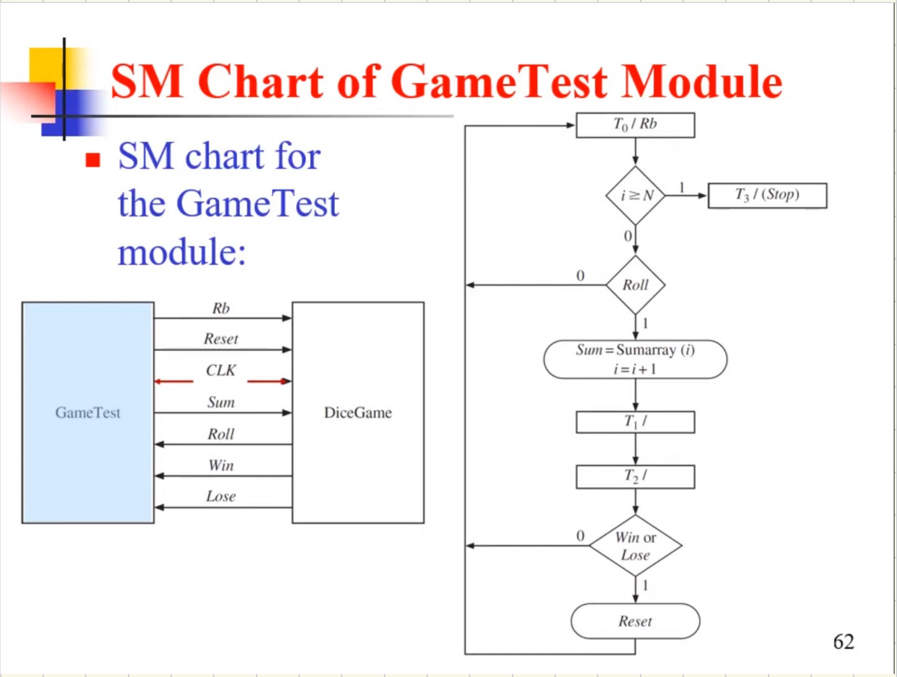




Testing



- Test the behavioral model:
  - Design a Verilog test bench module (GameTest) to ***monitor*** the ***output signals*** from the dice-game module and ***supply*** a sequence of ***inputs*** in response.
  - Functions of the GameTest module :
    1. Initially supply the ***Rb*** signal.
    2. When the DiceGame responds with a ***Roll*** signal, supply a ***Sum*** signal, which represents the sum of the two dice.
    3. If no ***Win*** or ***Lose*** signal is generated by the DiceGame, repeat steps 1 and 2 to roll again.
    4. When a ***Win*** or ***Lose*** signal is detected, generate a ***Reset*** signal and start again.



SM Chart of GameTest Module

- SM chart for the GameTest module: