

CpE358/CS381

**Switching Theory and
Logical Design**

Class 17

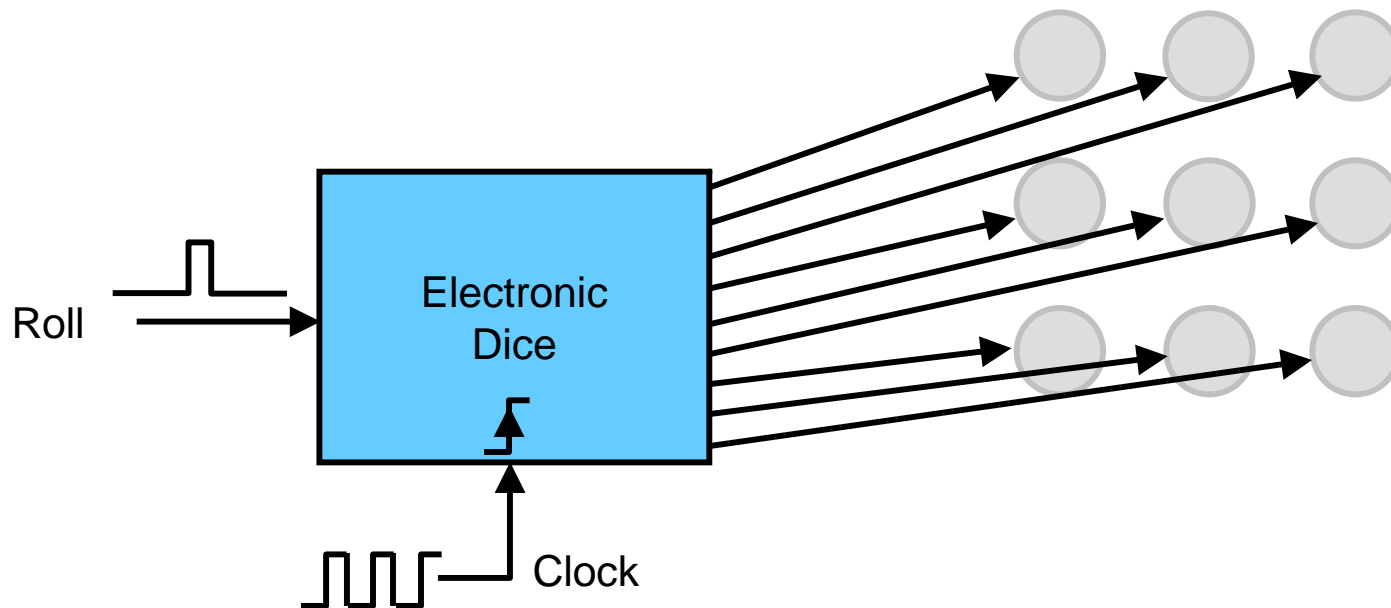
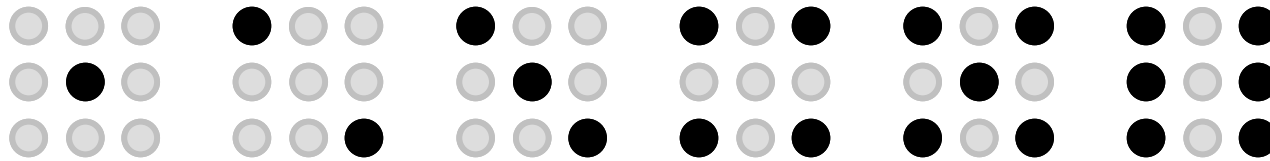
Today

- Fundamental concepts of digital systems (Mano Chapter 1)
- Binary codes, number systems, and arithmetic (Ch 1)
- Boolean algebra (Ch 2)
- Simplification of switching equations (Ch 3)
- Digital device characteristics (e.g., TTL, CMOS)/design considerations (Ch 10)
- Combinatoric logical design including LSI implementation (Chapter 4)
- Flip-flops and state memory elements (Ch 5)
- Sequential logic analysis and design (Ch 5)
- Counters, shift register circuits (Ch 6)
- Hazards, Races, and time related issues in digital design (Ch 9)
- Synchronous vs. asynchronous design (Ch 9)
- Memory and Programmable logic (Ch 7)
- Minimization of sequential systems
- Introduction to Finite Automata
- **Class Wrap-up: Electronic Dice Design Problem**

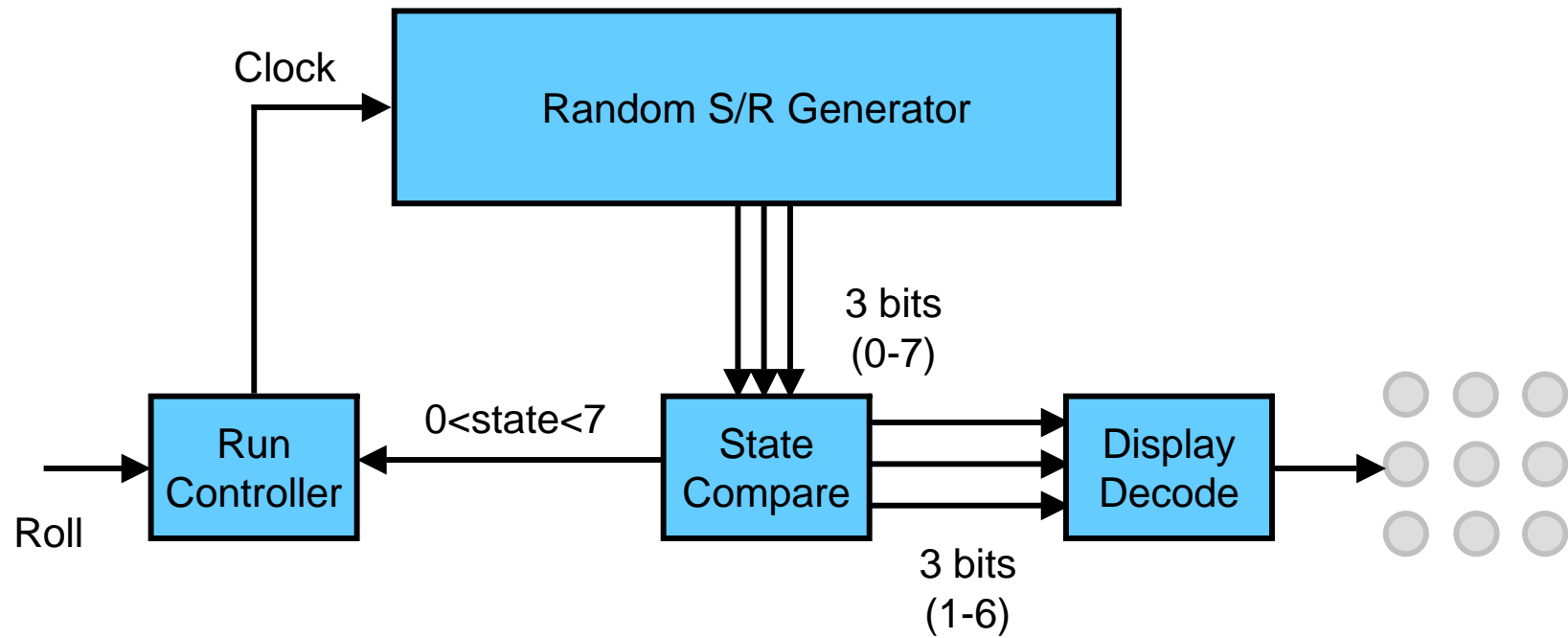
Initial Problem Statement

Electronic Dice (Difficulty=7)

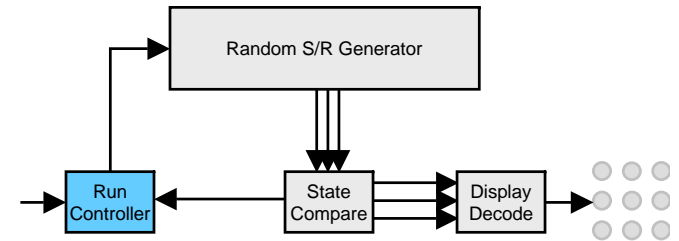
- 7 LEDs flash in a seemingly random fashion (20/second) until user presses a button. Flashing stops within 1 second and LEDs display a pattern representing the 6 possible die faces. Probability of any outcome is equally likely.



Block Diagram of System Implementation



Run Controller



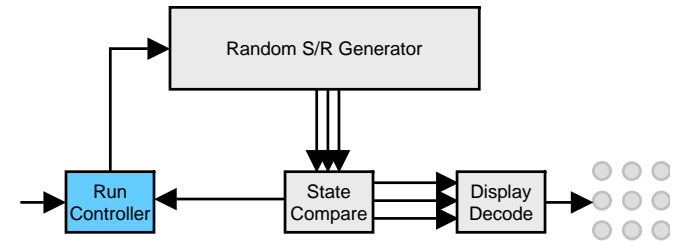
- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.

— $0 < \text{state} < 7$

Roll —

— SR
Clock

Run Controller



- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.

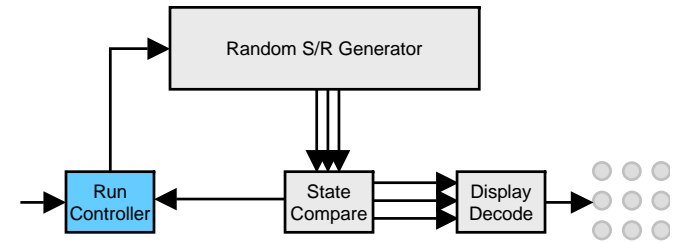
Start —

— $0 < \text{state} < 7$

Roll —

— SR
Clock

Run Controller

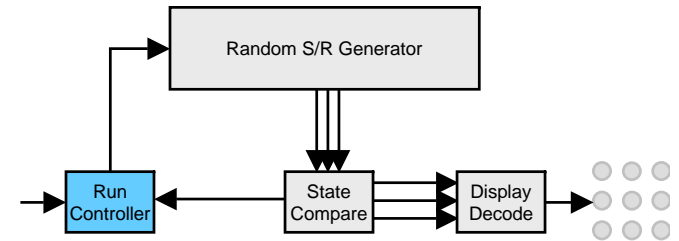


- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.

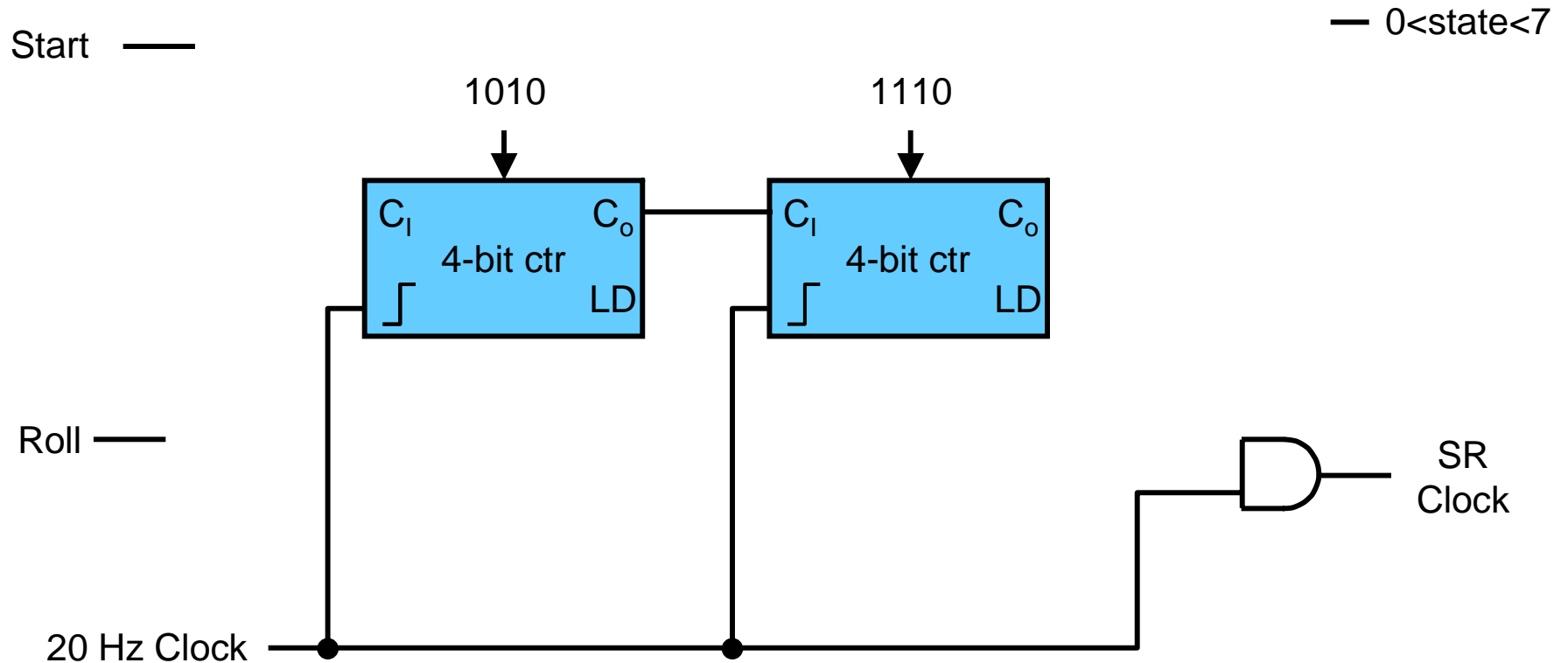
Start — — $0 < \text{state} < 7$



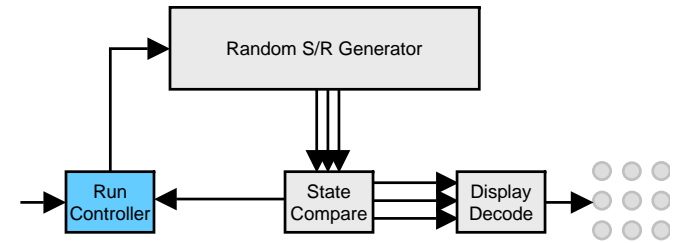
Run Controller



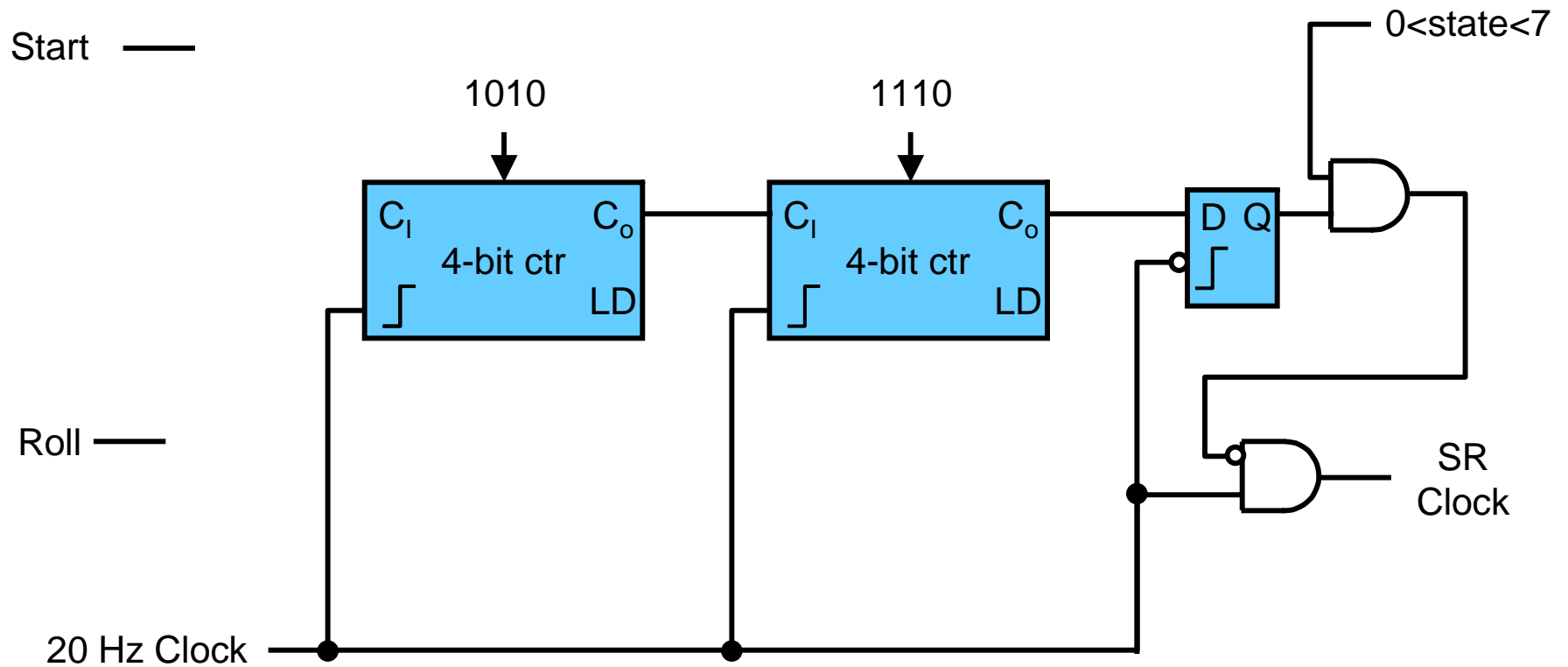
- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.



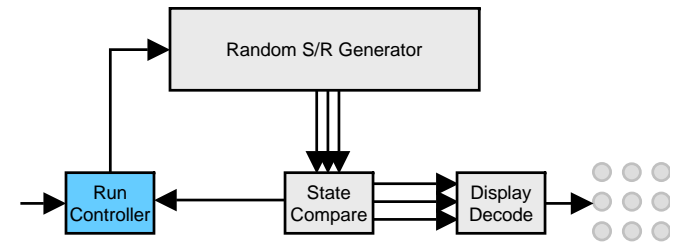
Run Controller



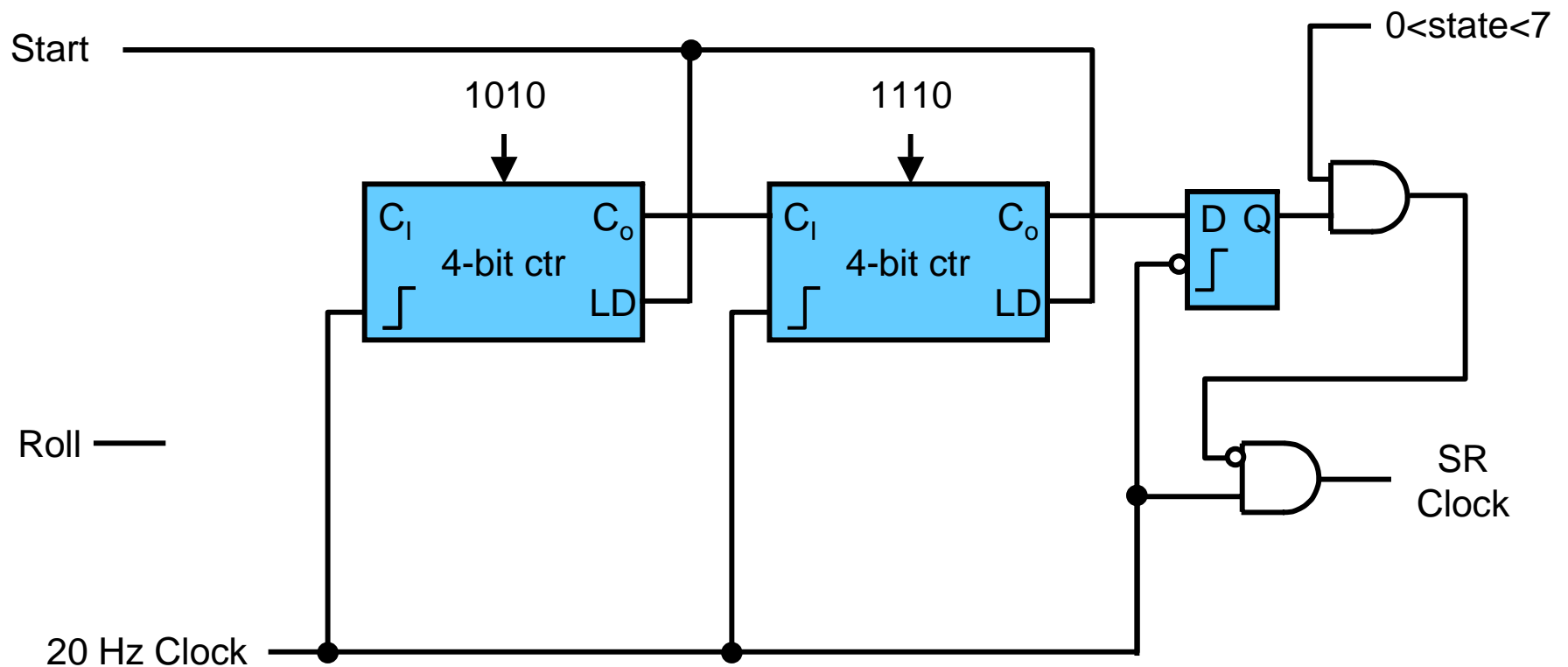
- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.



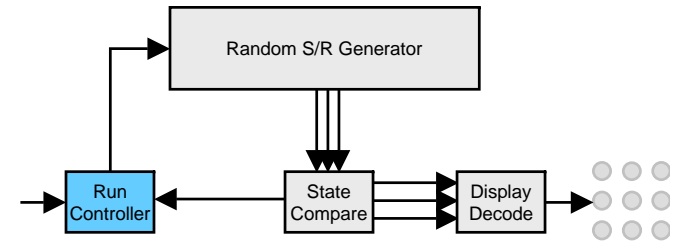
Run Controller



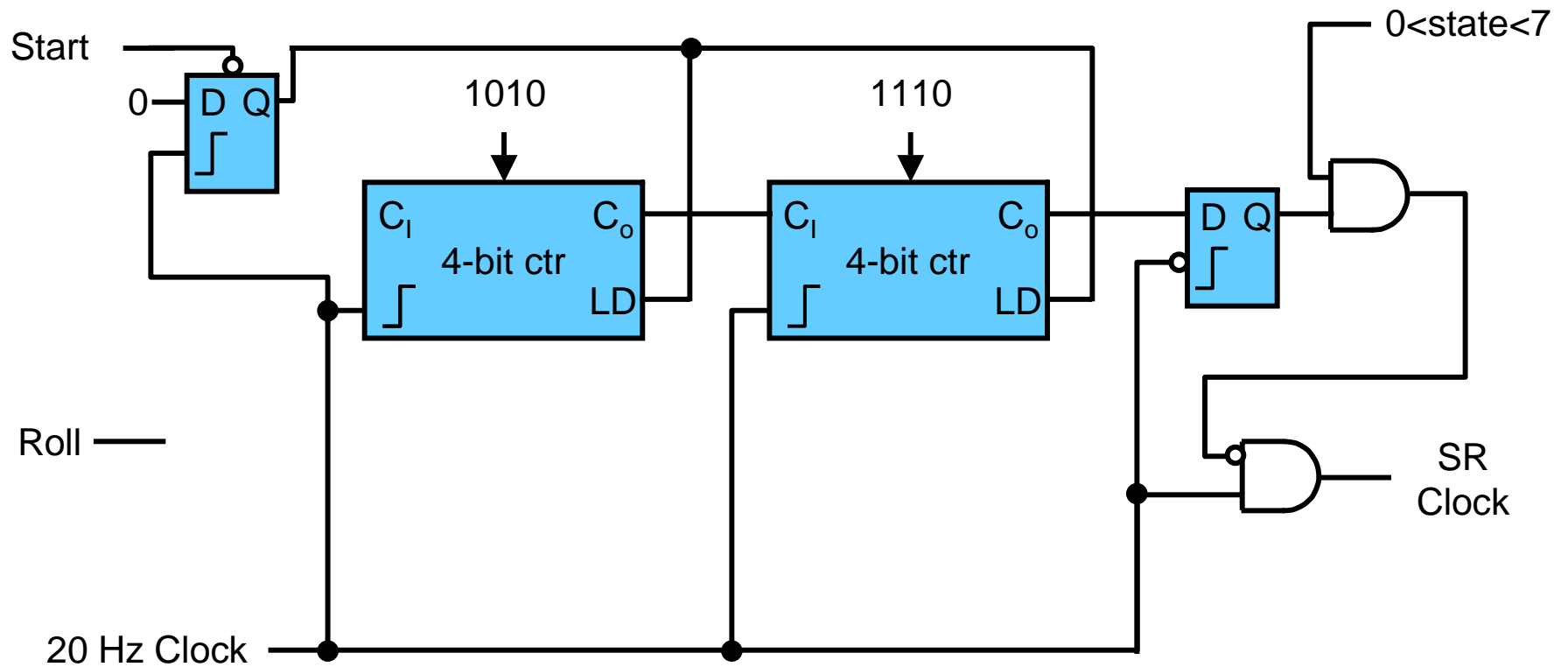
- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.



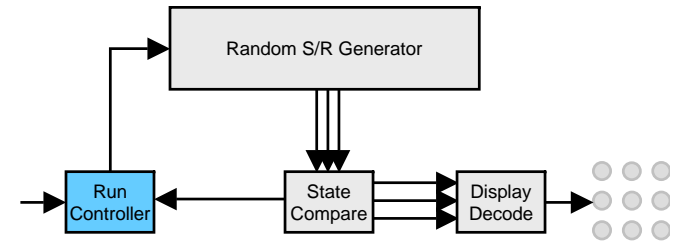
Run Controller



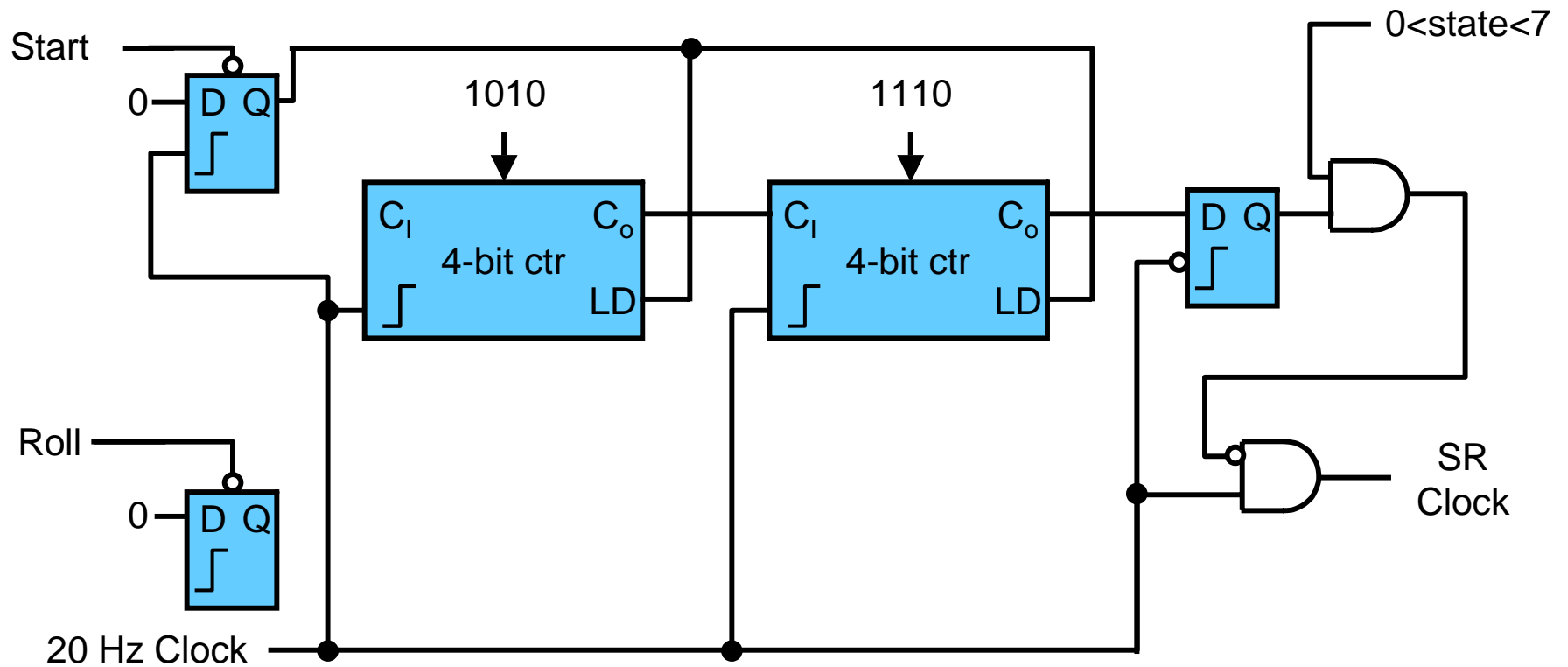
- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.



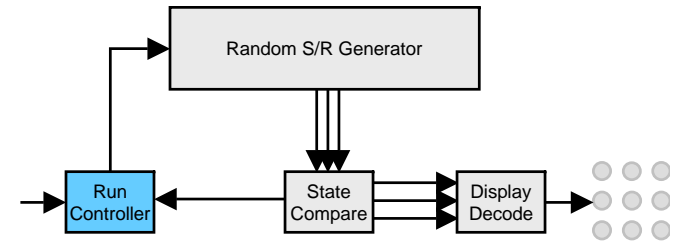
Run Controller



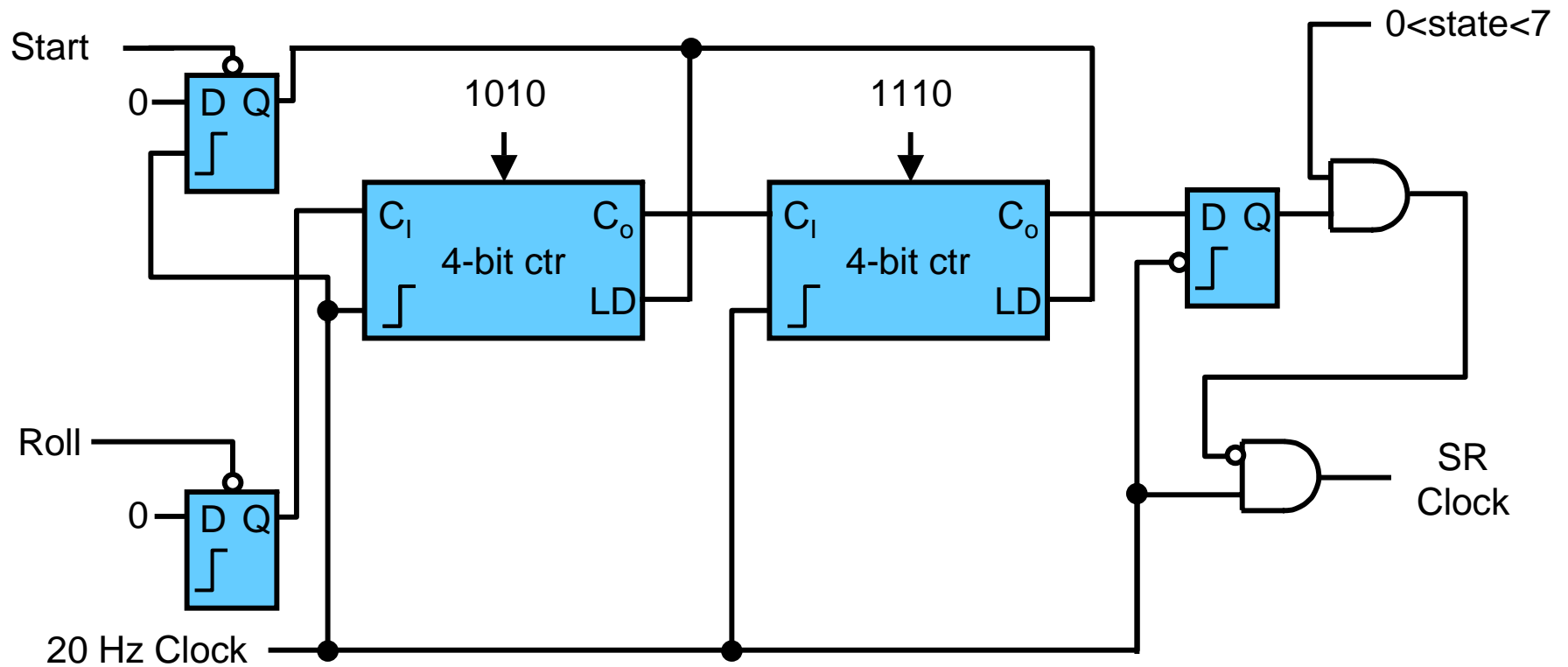
- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.



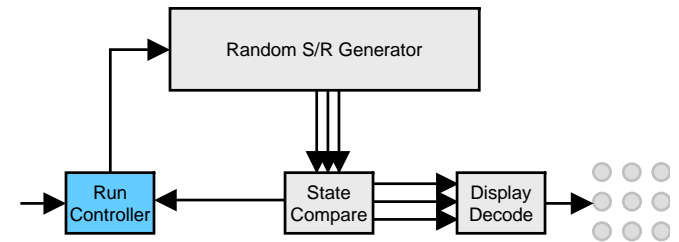
Run Controller



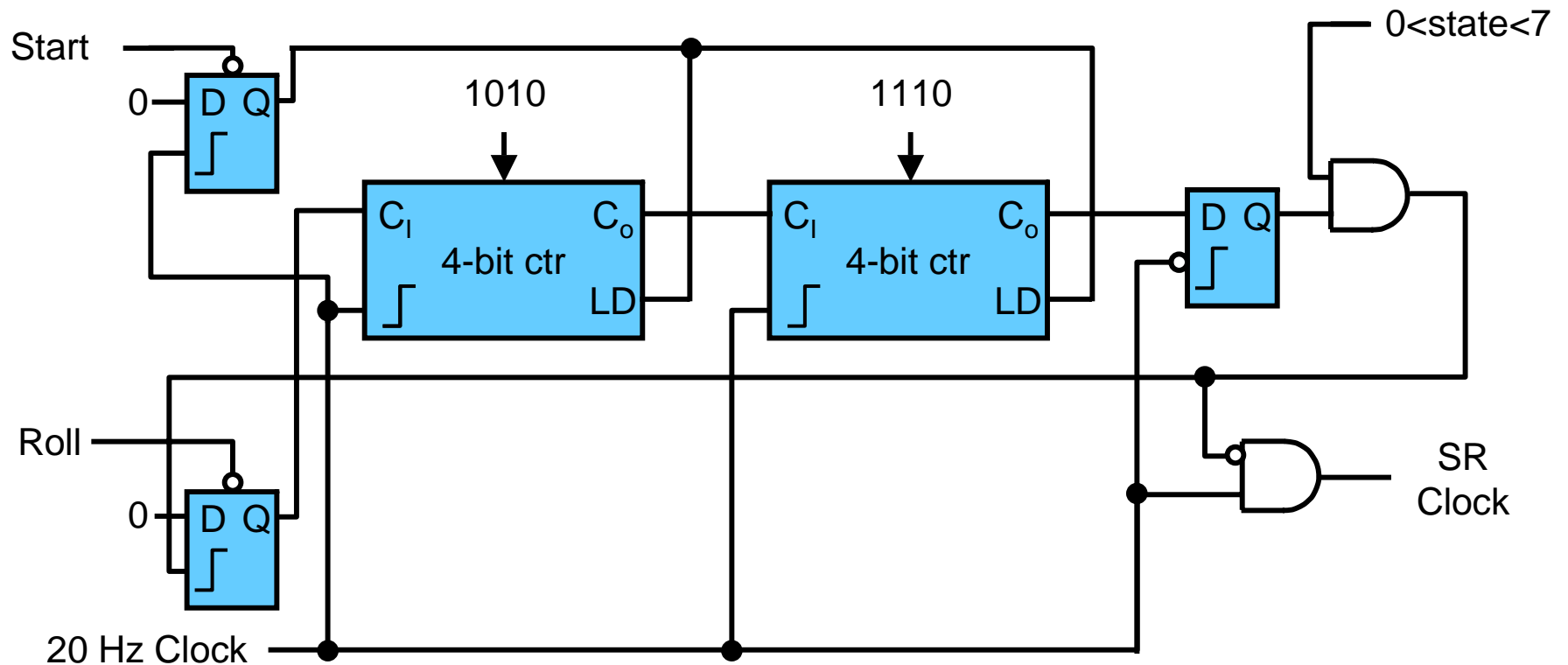
- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.



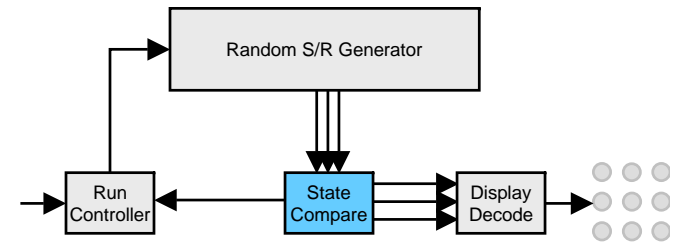
Run Controller



- If “Roll” has been pressed, output clock for no more than one more second. Stop the clock only if the state output of the shift register is 1, 2, 3, 4, 5, or 6.



State Compare

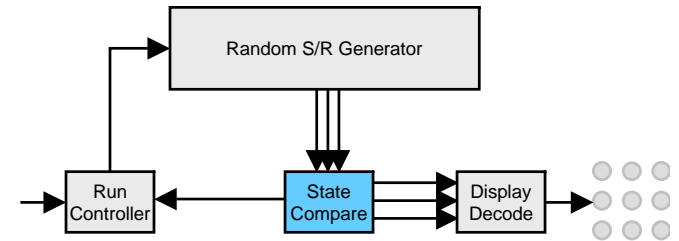


- State Compare prevents the shift register from stopping if the current state is 0 or 7

State	Output
000	0
001	1
010	1
011	1
100	1
101	1
110	1
111	0

$b_2b_1b_0$

State Compare

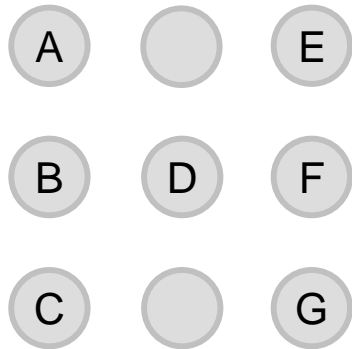


- State Compare prevents the shift register from stopping if the current state is 0 or 7

State	Output
000	0
001	1
010	1
011	1
100	1
101	1
110	1
111	0

$b_2b_1b_0$

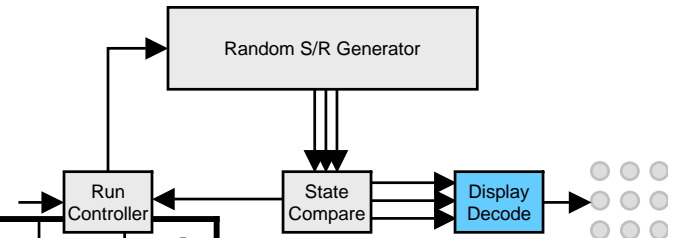
$$\text{Output}' = (b_0b_1b_2) + (b_0+b_1+b_2)$$



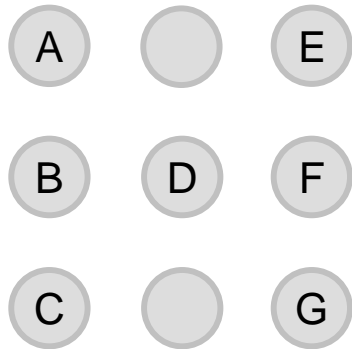
Display Decode

$b_2b_1b_0$

	A	B	C	D	E	F	G
0	X	X	X	X	X	X	X
1	0	0	0	1	0	0	0
2	1	0	0	0	0	0	1
3	1	0	0	1	0	0	1
4	1	0	1	0	1	0	1
5	1	0	1	1	1	0	1
6	1	1	1	0	1	1	1
7	X	X	X	X	X	X	X

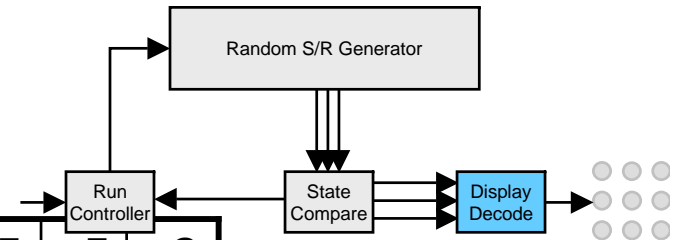


Display Decode



$b_2b_1b_0$

	A	B	C	D	E	F	G
0	X	X	X	X	X	X	X
1	0	0	0	1	0	0	0
2	1	0	0	0	0	0	1
3	1	0	0	1	0	0	1
4	1	0	1	0	1	0	1
5	1	0	1	1	1	0	1
6	1	1	1	0	1	1	1
7	X	X	X	X	X	X	X



b_1b_0

b_2

A	00	01	11	10
0	X	0	1	1
1	1	1	X	1

B	00	01	11	10
0	X	0	0	0
1	0	0	X	1

C	00	01	11	10
0	X	0	0	0
1	1	1	X	1

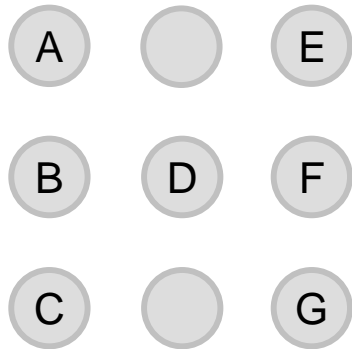
D	00	01	11	10
0	X	1	1	0
1	0	1	X	0

E	00	01	11	10
0	X	0	0	0
1	1	1	X	1

F	00	01	11	10
0	X	0	0	0
1	0	0	X	1

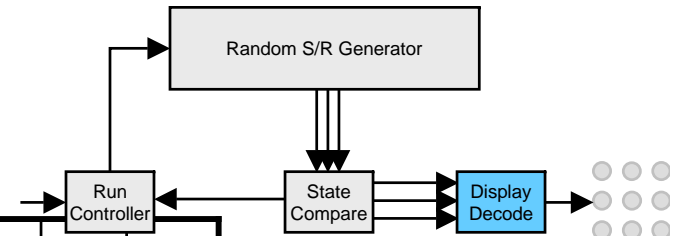
G	00	01	11	10
0	X	0	1	1
1	1	1	X	1

Display Decode



$b_2b_1b_0$

	A	B	C	D	E	F	G
0	X	X	X	X	X	X	X
1	0	0	0	1	0	0	0
2	1	0	0	0	0	0	1
3	1	0	0	1	0	0	1
4	1	0	1	0	1	0	1
5	1	0	1	1	1	0	1
6	1	1	1	0	1	1	1
7	X	X	X	X	X	X	X



b_1b_0

b_2

A	00	01	11	10
0	X	0	1	1
1	1	1	X	1

B	00	01	11	10
0	X	0	0	0
1	0	0	X	1

C	00	01	11	10
0	X	0	0	0
1	1	1	X	1

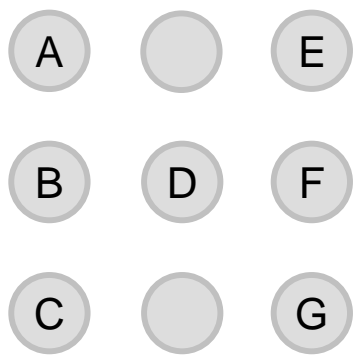
D	00	01	11	10
0	X	1	1	0
1	0	1	X	0

E	00	01	11	10
0	X	0	0	0
1	1	1	X	1

F	00	01	11	10
0	X	0	0	0
1	0	0	X	1

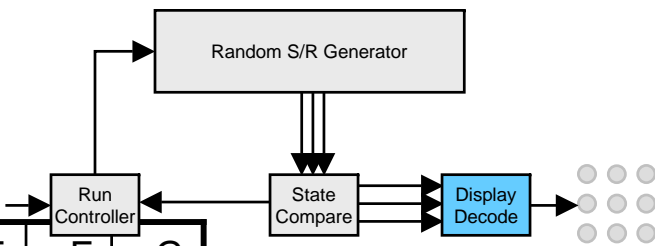
G	00	01	11	10
0	X	0	1	1
1	1	1	X	1

Display Decode



$b_2b_1b_0$

	A	B	C	D	E	F	G
0	X	X	X	X	X	X	X
1	0	0	0	1	0	0	0
2	1	0	0	0	0	0	1
3	1	0	0	1	0	0	1
4	1	0	1	0	1	0	1
5	1	0	1	1	1	0	1
6	1	1	1	0	1	1	1
7	X	X	X	X	X	X	X



b_1b_0

	A	00	01	11	10
0	X	0	1	1	
1	1	1	X	1	

A $A=b_1+b_2$

	B	00	01	11	10
0	X	0	0	0	
1	0	0	X	1	

B $B=b_1b_2$

	C	00	01	11	10
0	X	0	0	0	
1	1	1	X	1	

C $C=b_2$

	D	00	01	11	10
0	X	1	1	0	
1	0	1	X	0	

D $D=b_0$

	E	00	01	11	10
0	X	0	0	0	
1	1	1	X	1	

E $E=b_2$

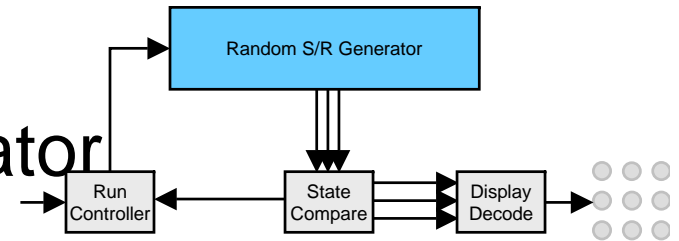
	F	00	01	11	10
0	X	0	0	0	
1	0	0	X	1	

F $F=b_1b_2$

	G	00	01	11	10
0	X	0	1	1	
1	1	1	X	1	

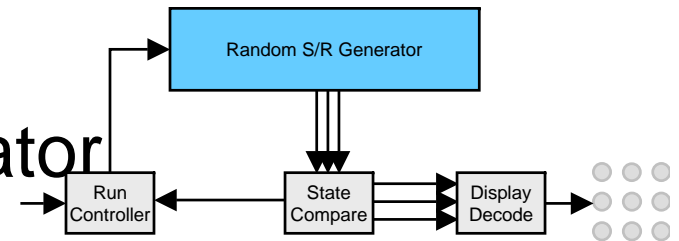
G $G=b_1+b_2$

Random S/R Generator



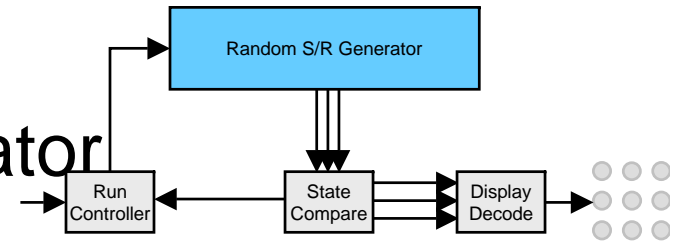
- Use a maximal length S/R generator to create a pseudorandom string of 1's and 0's.

Random S/R Generator



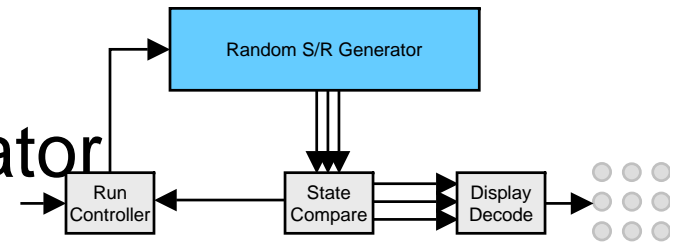
- Use a maximal length S/R generator to create a pseudorandom string of 1's and 0's.
- 3 bits are taken out of the S/R to represent the die state. Since there is an equal probability of any three bit combination, there are 8 states which can be represented. 2 of them are eliminated by other circuitry, so the probability of 1, 2, 3, 4, 5, or 6 are the same.

Random S/R Generator

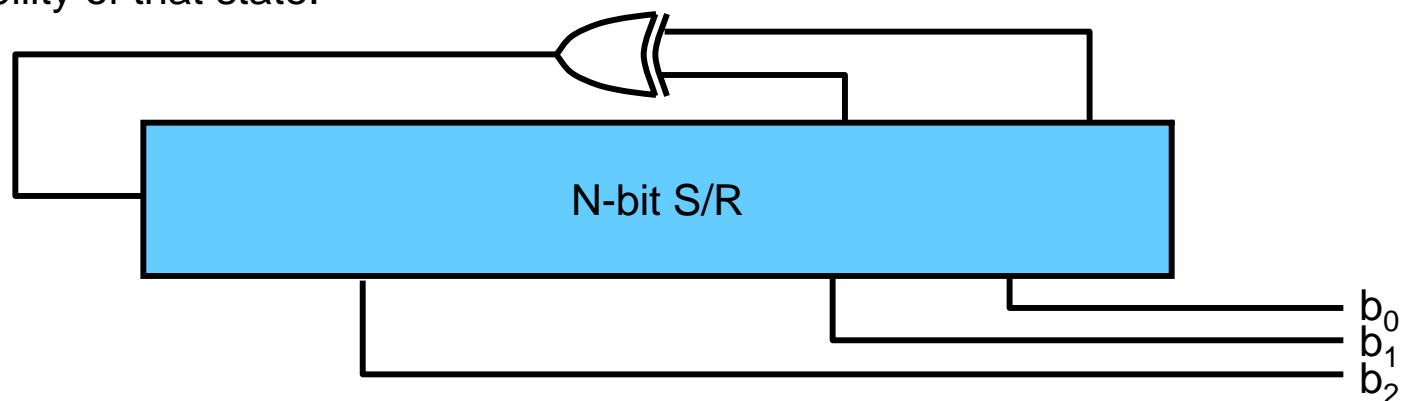


- Use a maximal length S/R generator to create a pseudorandom string of 1's and 0's.
- 3 bits are taken out of the S/R to represent the die state. Since there is an equal probability of any three bit combination, there are 8 states which can be represented. 2 of them are eliminated by other circuitry, so the probability of 1, 2, 3, 4, 5, or 6 are the same.
- The three bits that are selected for use must not be adjacent, otherwise if 000 is rejected, the next state will be 000 or 100, depending on the next bit. 000 will be rejected again, but 100 will be accepted, increasing its probability compared to the other states. Likewise, if 111 is rejected, 011 will be accepted, increasing the probability of that state.

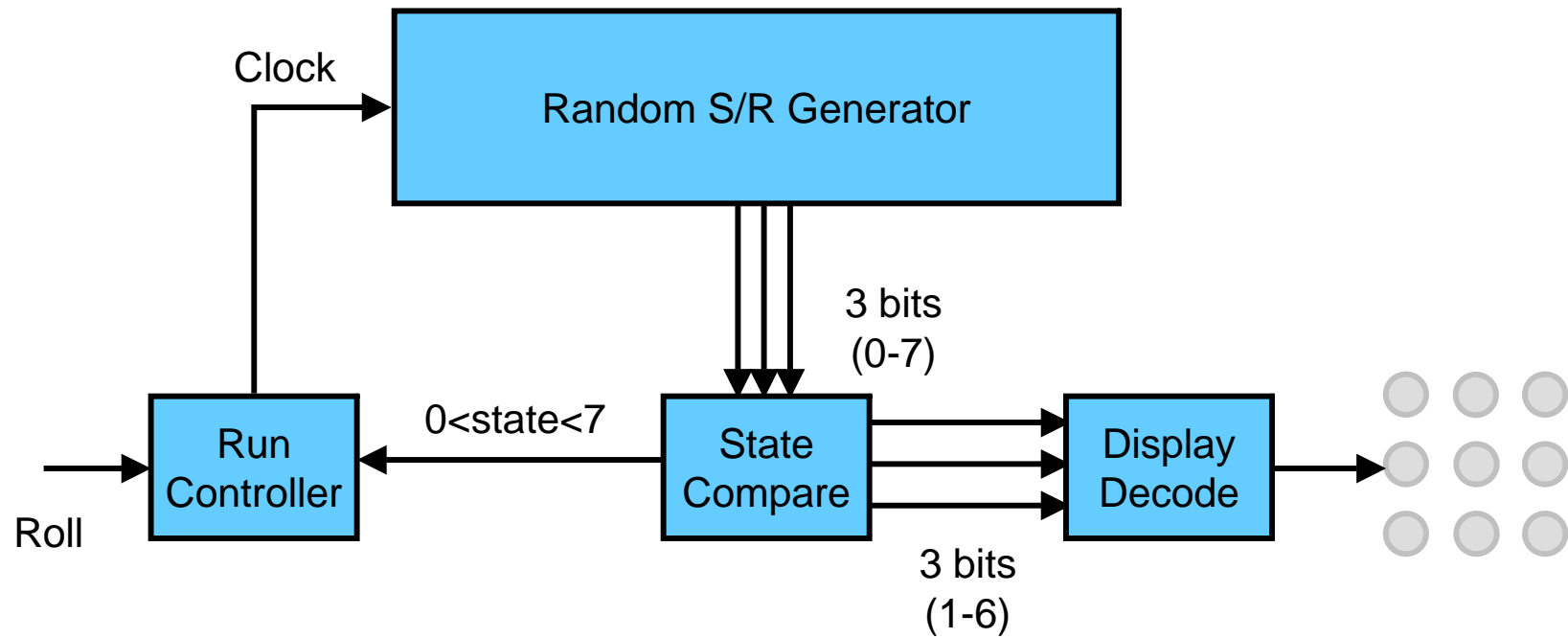
Random S/R Generator



- Use a maximal length S/R generator to create a pseudorandom string of 1's and 0's.
- 3 bits are taken out of the S/R to represent the die state. Since there is an equal probability of any three bit combination, there are 8 states which can be represented. 2 of them are eliminated by other circuitry, so the probability of 1, 2, 3, 4, 5, or 6 are the same.
- The three bits that are selected for use must not be adjacent, otherwise if 000 is rejected, the next state will be 000 or 100, depending on the next bit. 000 will be rejected again, but 100 will be accepted, increasing its probability compared to the other states. Likewise, if 111 is rejected, 011 will be accepted, increasing the probability of that state.



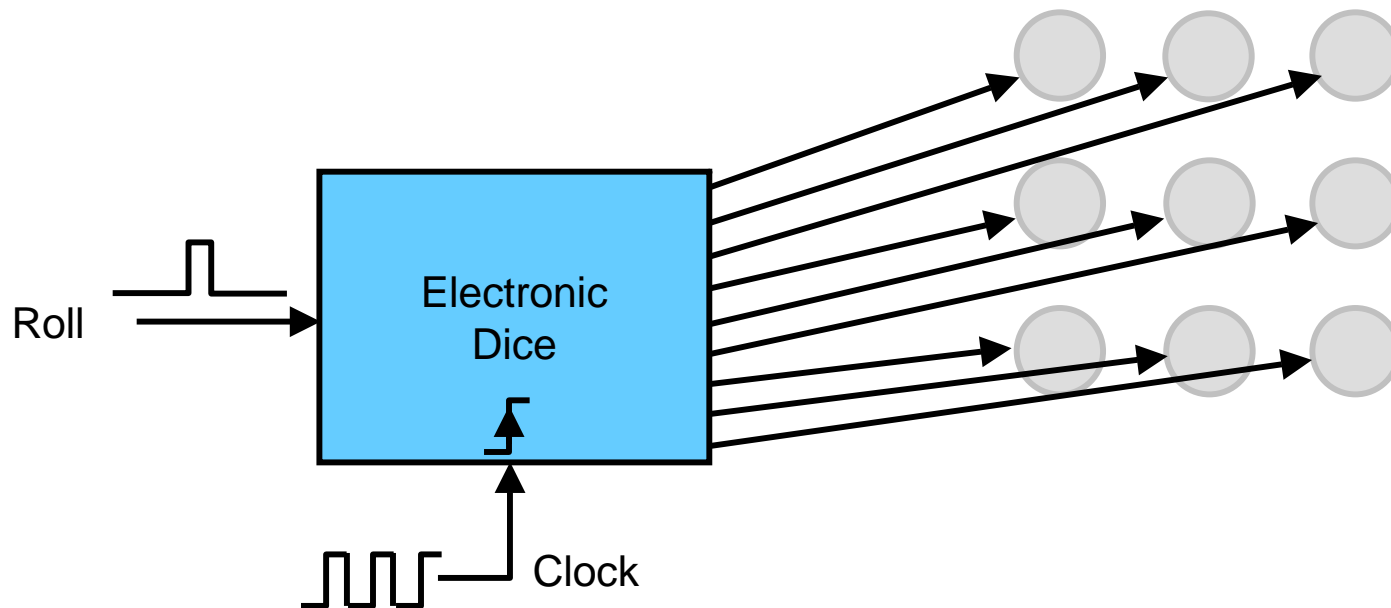
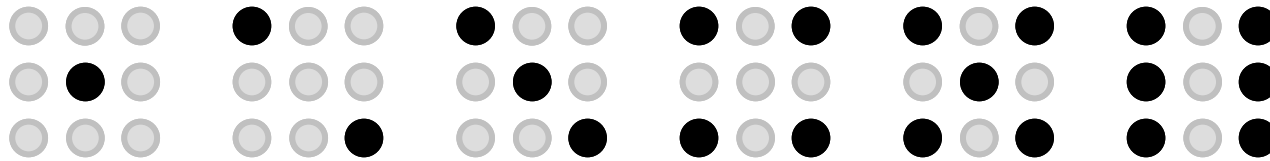
Block Diagram of System Implementation



Initial Problem Statement

Electronic Dice (Difficulty=7)

- 7 LEDs flash in a seemingly random fashion (20/second) until user presses a button. Flashing stops within 1 second and LEDs display a pattern representing the 6 possible die faces. Probability of any outcome is equally likely.



Summary

- Fundamental concepts of digital systems (Mano Chapter 1)
- Binary codes, number systems, and arithmetic (Ch 1)
- Boolean algebra (Ch 2)
- Simplification of switching equations (Ch 3)
- Digital device characteristics (e.g., TTL, CMOS)/design considerations (Ch 10)
- Combinatoric logical design including LSI implementation (Chapter 4)
- Flip-flops and state memory elements (Ch 5)
- Sequential logic analysis and design (Ch 5)
- Counters, shift register circuits (Ch 6)
- Hazards, Races, and time related issues in digital design (Ch 9)
- Synchronous vs. asynchronous design (Ch 9)
- Memory and Programmable logic (Ch 7)
- Minimization of sequential systems
- Introduction to Finite Automata
- **Class Wrap-up: Electronic Dice Design Problem**