

1. How Hamming(7,4) works

111	110	101	100	011	010	001
7	6	5	4	3	2	1
D3	D2	D1	C2	D0	C1	C0

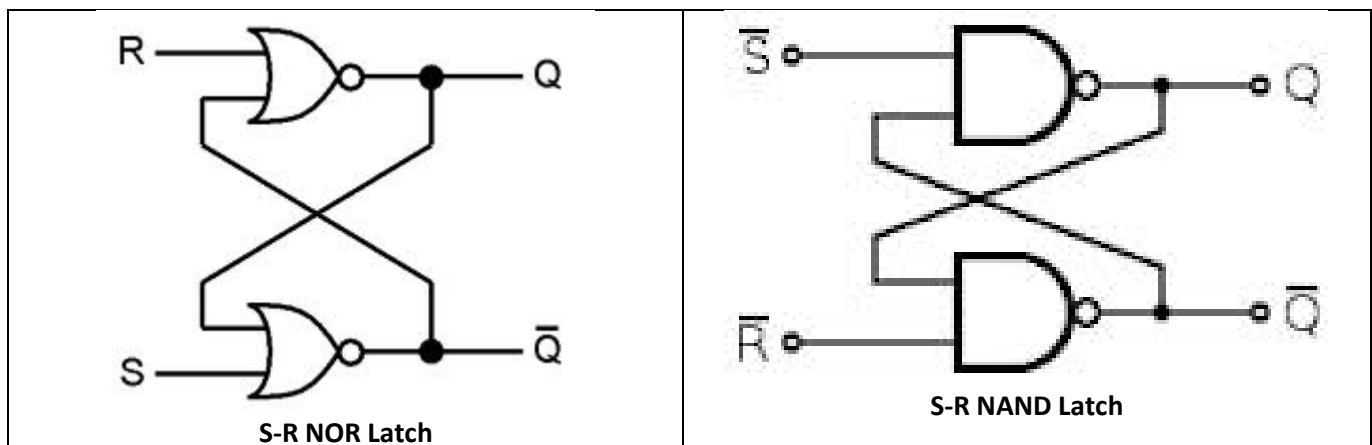
- a. Each data bit must be covered (checked) by at least 2 parity bits
 - i. Doesn't matter which parity bits cover which data bits
 - ii. However, doing it in the way we did allows you to determine the error location easily
- b. Why we start numbering from 1
 - i. Need a bit pattern that indicates there were no errors
 - ii. If the recreated check bits match the original check bits, XOR returns 0
 - iii. We use that position to indicate no errors
- c. Which parity bits check which data bits
 - i. C0 is in location 1
 1. Checks all bits with bit 1 high (except for itself)
 2. 3, 5, 7, 9, 11, so on
 - ii. C1 is in location 2
 1. Checks all bits with bit 2 high (except for itself)
 2. 3, 6, 7, 10, 11, so on
 - iii. C2 is in location 4
 1. Checks all bits with bit 4 high (except for itself)
 2. 5, 6, 7, 12, 13, so on
 - iv. If we had a C3, it would be in location 8
 1. Would check all bits with bit 8 high (except for itself)
 2. 9, 10, 11, 12, so on
 - v. In general
 1. Next check bit lies at bit positions that are powers of 2
- d. Why we can XOR to determine the bit position
 - i. We are essentially creating a binary number based on all the check bits
 - ii. Whether or not a given data position is set or not affects all the check bits that check that location
 1. If a data bit is changed, all the check bits associated with that location will be affected
 2. All of them will differ by 1 from the original calculation for the check bit at that position
 3. Take the example from the previous notes
 - a. Original data 0110011 → 0010011 with bit 6 flipped
 - b. Bit 6 was the one that changed
 - c. The two check bits that were checking it (C2 and C1) changed as well
 4. Combination of affected check bits form the position of which bit was changed
- e. Does it matter which bits get flipped?
 - i. No difference in algorithm if data bit or check bit gets flipped (try it for yourself!)
 - ii. Algorithm only cares about difference in Hamming distance
- f. Expanding it further
 - i. Expanding to cover a larger data size
 1. $2^K - 1 \geq M + K$, where M = data bits, and K = check bits
 2. From above, $2^3 - 1 \geq 4$ data bits + 3 check bits
 - ii. Covering more errors
 1. Given T errors:
 - a. Hamming distance between valid code words must be $T + 1$ to detect
 - b. Hamming distance between valid code words must be $2T + 1$ to correct

2. Sequential circuits

- a. So far, only discussed combinational circuits
 - i. Output strictly dependent on the inputs only
- b. Sequential circuits – output depends on the current inputs as well as the *history* of inputs
 - i. Other way to phrase it – they have memory
 - ii. History of inputs determines “state” of the circuit
- c. Examples
 - i. House alarm that needs to be reset once an intrusion is detected
 - ii. Combination lock

3. Latches

- a. Circuit that has two stable states, can be used to store information
 - i. Change state via signals applied to the control inputs
- b. S-R latches
 - i. Set sets the latch
 - ii. Reset clears the latch
 - iii. Q and !Q give the output and complement
- c. S-R latch implementation



d. Characteristic tables

- i. Expected operation of latch

S	R	Q_{n+1}	S	R	Q_{n+1}
0	0	Q_n	1	1	Q_n
0	1	0	1	0	0
1	0	1	0	1	1
1	1	Undefined	0	0	Undefined

S-R NOR Latch S-R NAND Latch