3-bit binary counter with Decoded output

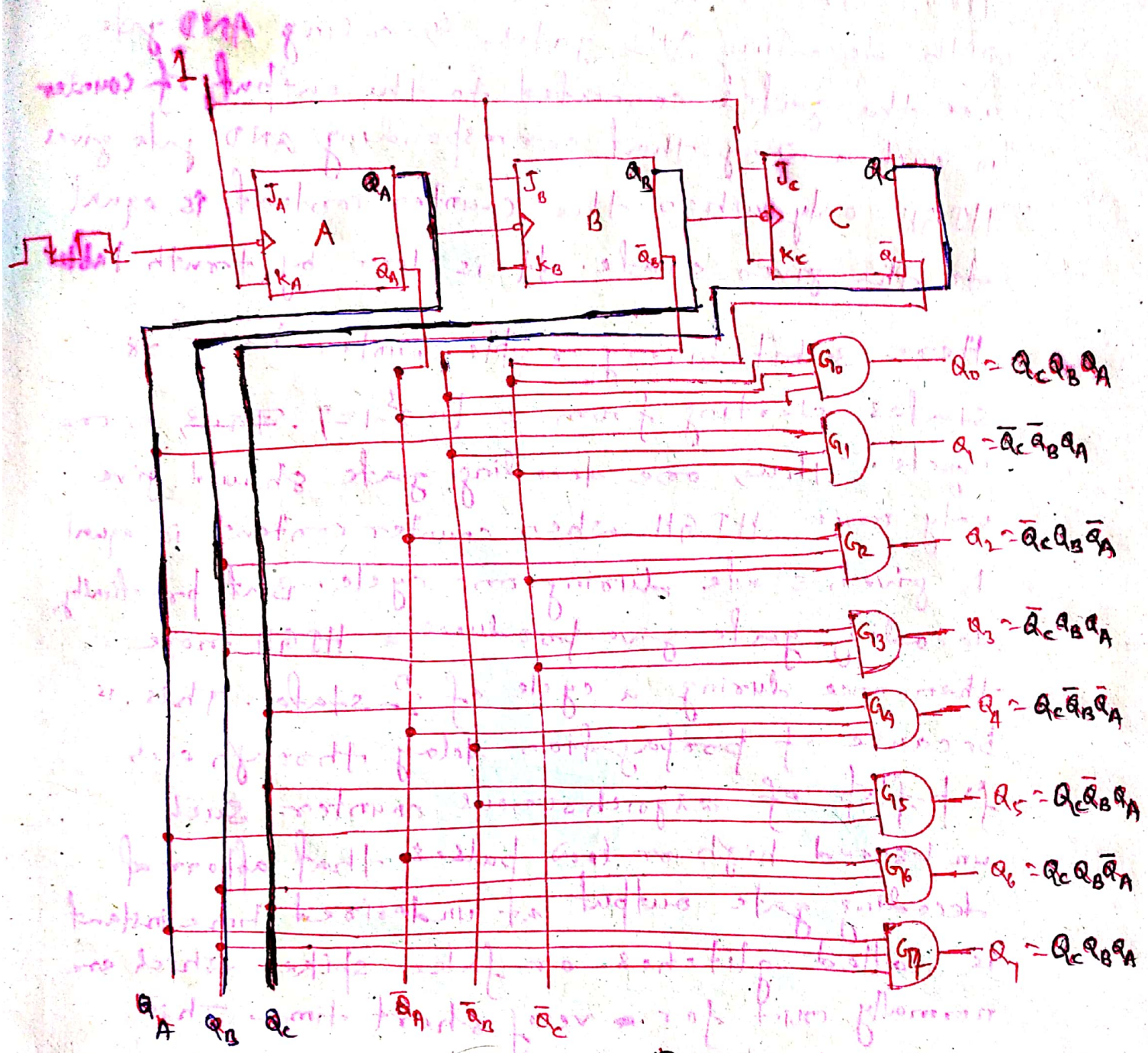


Fig-1.

Truth table

				Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7
state	Q_C	Q_B	Q_A	G_0	G_1	G_2	G_3	G_4	G_5	G_6	G_7
0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	0	0	0	0
2	0	1	0	0	0	1	0	0	0	0	0
3	0	1	1	0	0	0	1	0	0	0	0
4	1	0	0	0	0	0	0	1	0	0	0
5	1	0	1	0	0	0	0	0	1	0	0
6	1	1	0	0	0	0	0	0	0	1	0
7	1	1	1	0	0	0	0	0	0	0	1

Here the counter with decoded output consisting with decoding AND gates. Decoding AND gates are the gates connected to the output of counter in such a way that corresponding AND gate gives HIGH only when the counter content is equal to the given state. It is clear by truth table.

Here 3-bit binary ripple counter has $2^3 = 8$ states starting from 0 to $2^3 - 1 = 7$. ~~Thus~~, in one cycle. Thus, one decoding gate should give only once HIGH when counter content is equal to given state during one cycle. But practically, decoding gate give produces a HIGH more than once during a cycle of 2^n states. This is because of propagation delay through each flip-flop of asynchronous counter. Such undesired high or low pulses that appear at decoding gate output at undesired time instant is called glitches or fake spikes which are normally exist for a very short time. This can be observe by timing diagram as shown in fig-2.

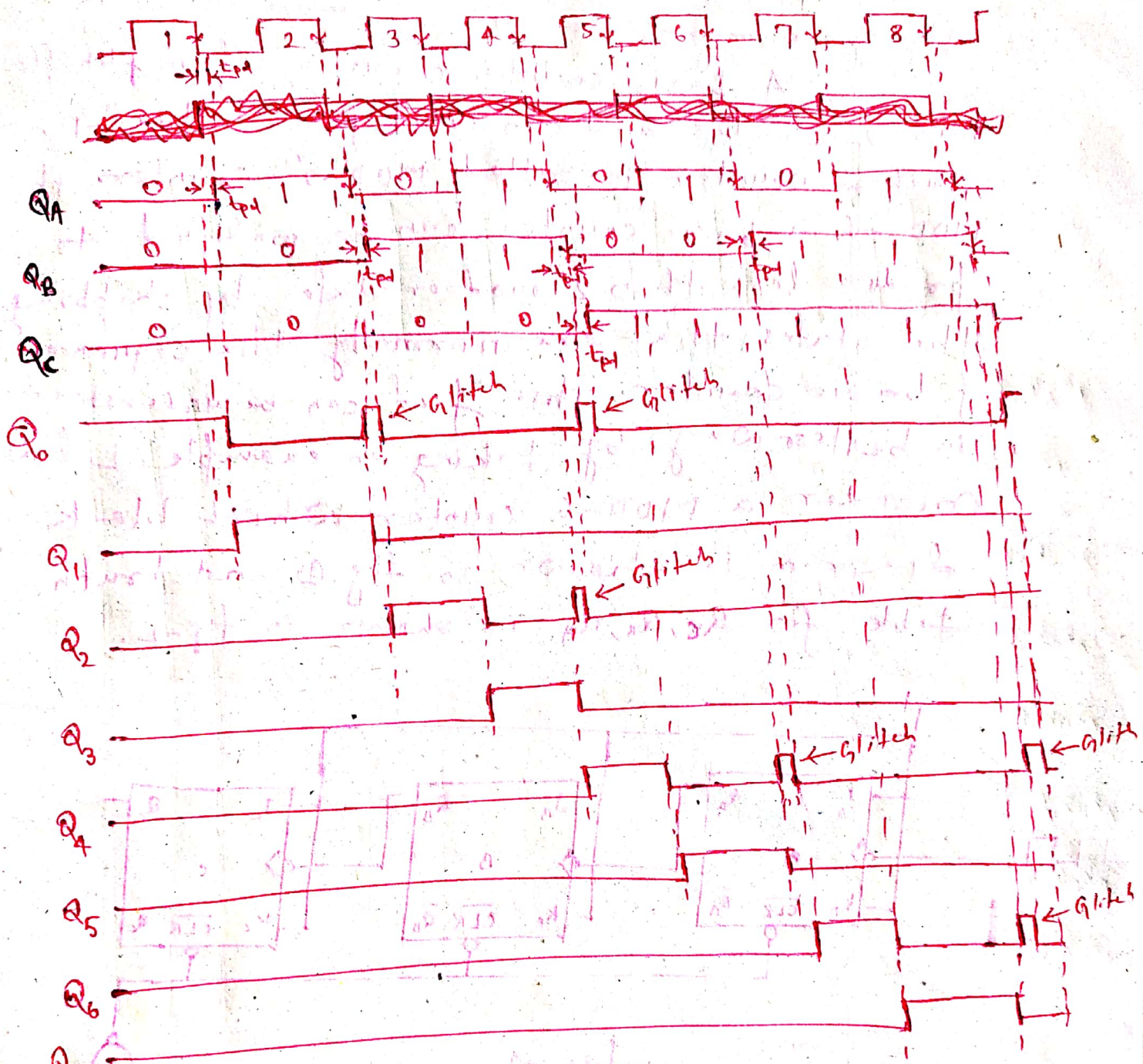


Fig-2

These glitches are avoided by using any one of —

i) Clock input to strobe the decoding gate

ii) Using synchronous counter.

Ripple Counter with modulus $< 2^n$

(Truncated sequence counter)

A basic counter consisting of n-flip-flops has 2^n states and can count from 0 to $(2^n - 1)$. Such a counter is called MOD-2^n counter. The MOD of a basic counter can be modified to produce less than 2^n mod or states by skipping the states that are normally the sequence of basic counter. This fact can be understood in better way by taking an example. Let us consider a MOD-6 counter which block diagram is shown in fig. ① and truth table for Q_C, Q_B, Q_A is shown in fig-2.

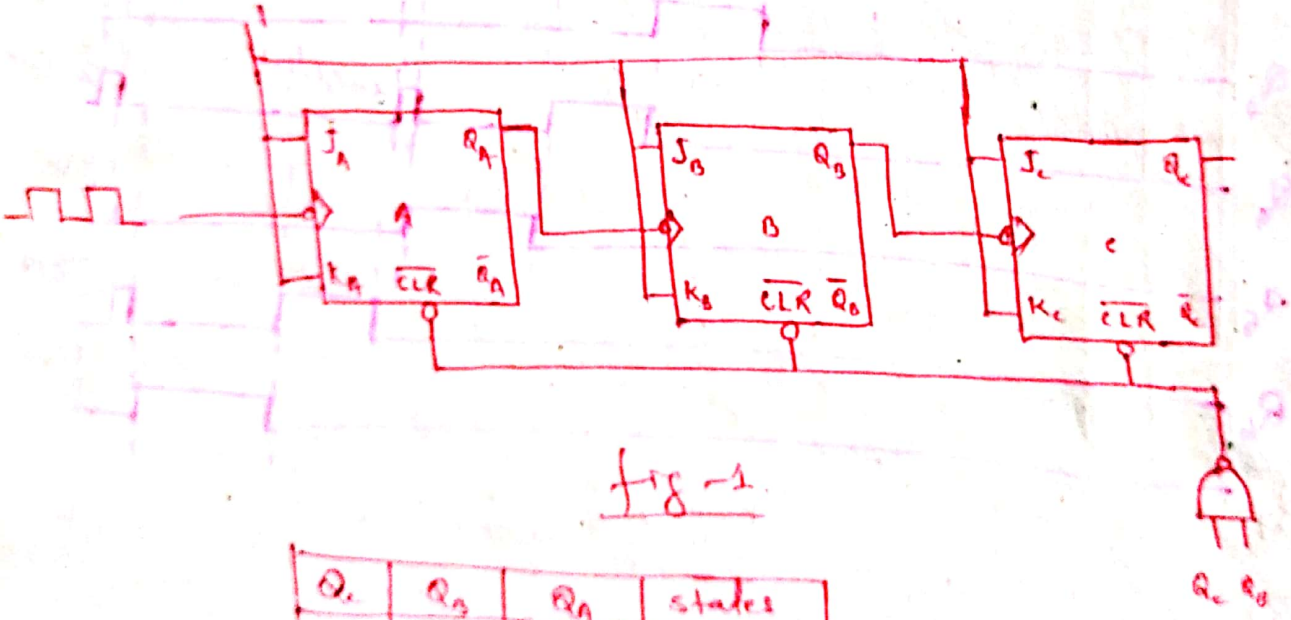


fig-1

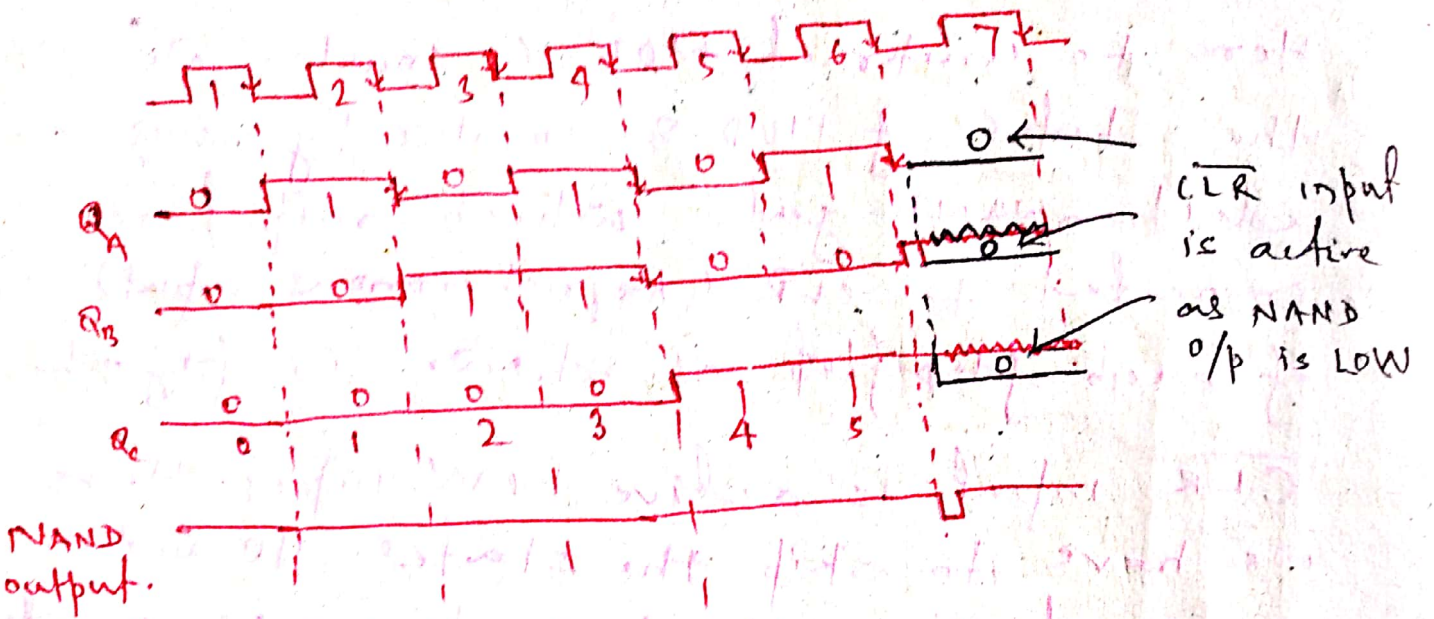
Q_C	Q_B	Q_A	states
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

fig-2

A MOD-6 counter has six states and counts from 0 to 5 (000 to 111). To design MOD-6

counter we need three flip-flops but by using three flip-flops, designed counter has $2^3 = 8$ states and can count from 000 to 111. Here to construct MOD-6 counter, we modify the states of MOD-8 counter by using an addition NAND gate which output is connected to $\overline{CLR}$ (Asynchronous input) input of each flip-flop as shown in fig-1. $\overline{CLR}$ input is active LOW input. Here we have to skip the states 110 and 111 so that the MOD-6 counter may counts from 000 to 101. Hence the output of flip-flop B and C which are Q_B and Q_C are connected to additional NAND gate. As it is clear from truth table that from state 000 to 101, both the Q_B and Q_C are not simultaneously HIGH which causes the NAND gate to give HIGH as its its output which is connected to $\overline{CLR}$ input of each flip-flop and hence $\overline{CLR}$ is inactive and this counter counts from 000 → 001 → 010 → 011 → 100 → 101. But as soon as the counter goes from 101 to 110, the $Q_B = 1$, $Q_C = 1$ and hence output of NAND becomes LOW and $\overline{CLR}$ input of each flip-flop get active and it RESET all the flip-flop to 000. Once the flip-flop reset at 000, the counter state start from 000 again. Here it is to be noted that the state 110 appear there only for a nano-second.

Timing diagram for MOD-6 counter.



Working Rule to construct MOD-N Counter —

i> Find no. of flip-flops required to design MOD-N counter by —

$$2^{n-1} \leq N \leq 2^n$$

ii> Connect all the flip-flops as ripple counter.

iii> Convert N into binary.

iv> Take a NAND gate and out of this NAND gate is connected to $\overline{CLR}$ input of all flip-flops.

v> The output of those flip-flops for which Q=1 corresponding to state N, are connected to NAND inputs.