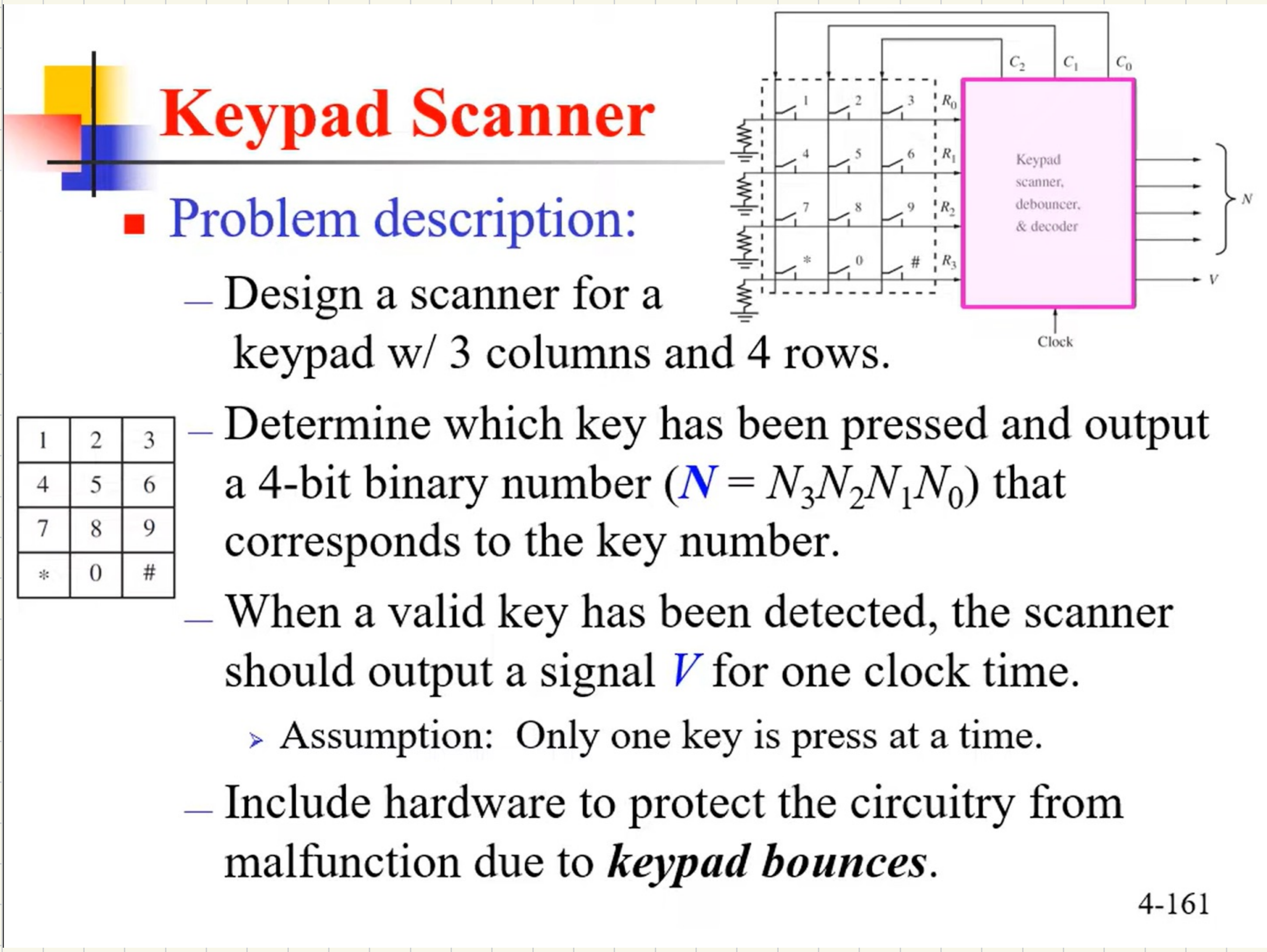
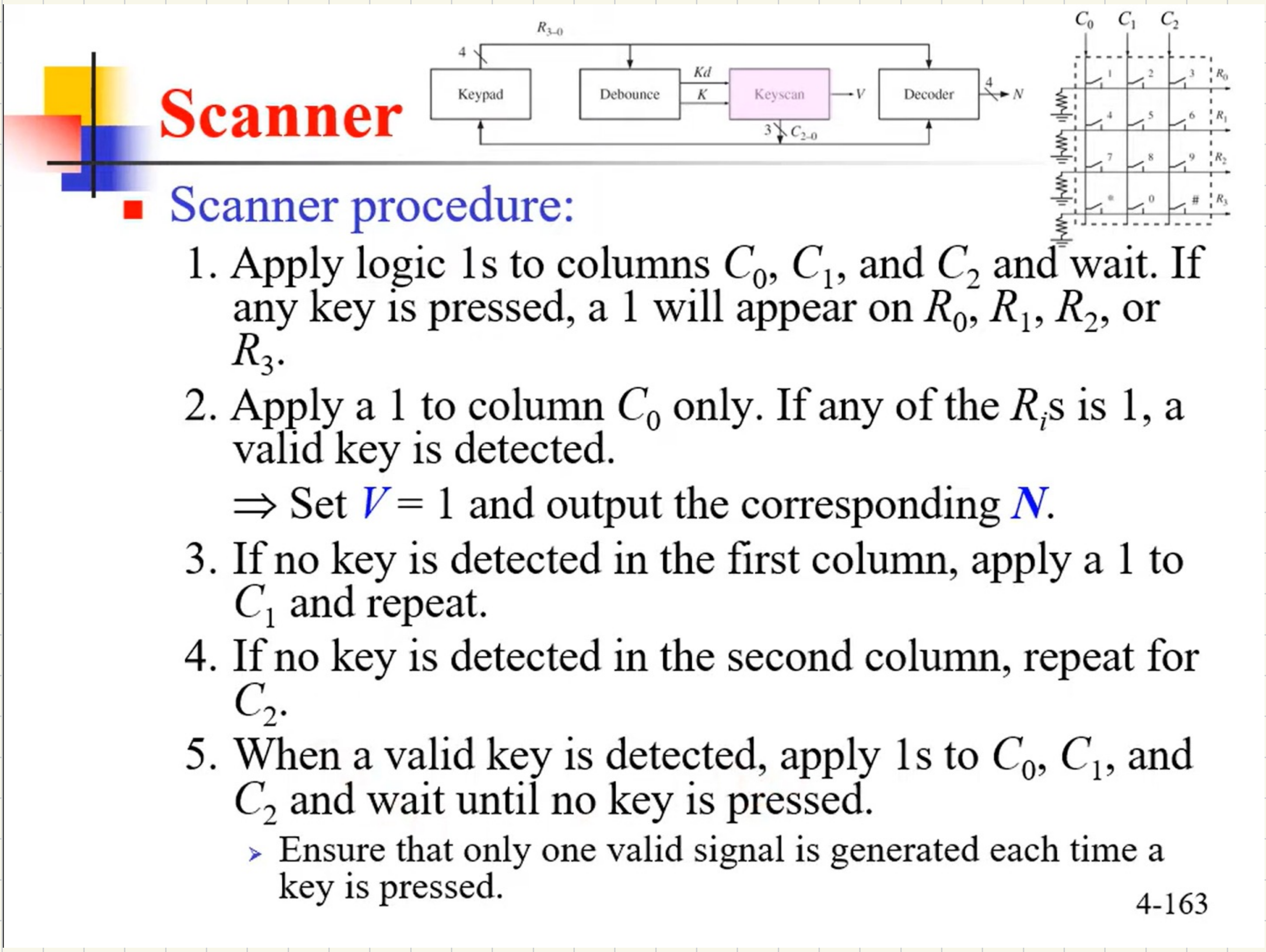
4.11 Keypad Scanner



Keypad Scanner

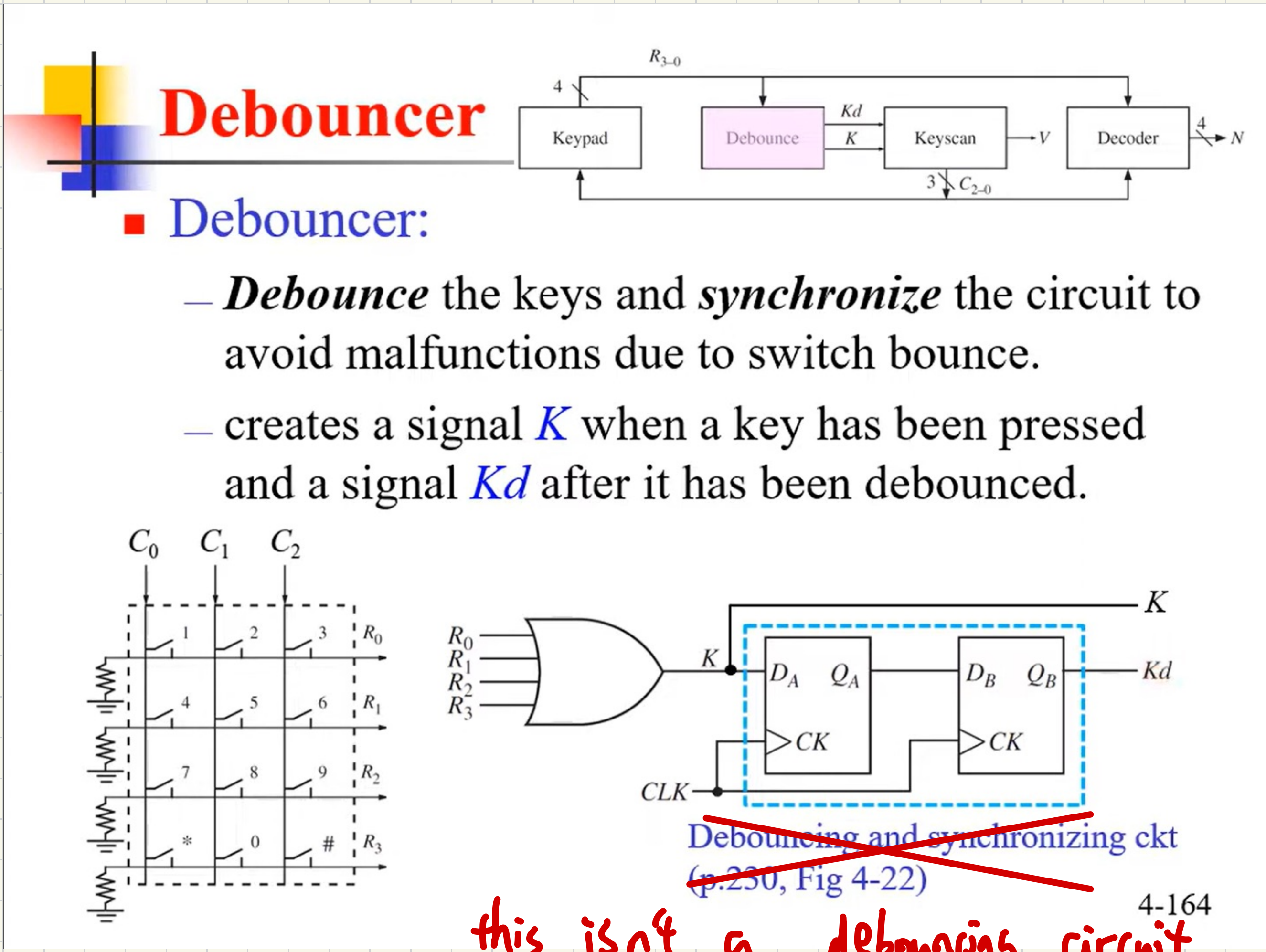
- Problem description:
  - Design a scanner for a keypad w/ 3 columns and 4 rows.
  - Determine which key has been pressed and output a 4-bit binary number (***N*** = $N_3N_2N_1N_0$) that corresponds to the key number.
  - When a valid key has been detected, the scanner should output a signal V for one clock time.
    - Assumption: Only one key is press at a time.
  - Include hardware to protect the circuitry from malfunction due to ***keypad bounces***.

1	2	3
4	5	6
7	8	9
*	0	#



Scanner

- Scanner procedure:
  1. Apply logic 1s to columns C_0, C_1, and C_2 and wait. If any key is pressed, a 1 will appear on R_0, R_1, R_2, or R_3.
  2. Apply a 1 to column C_0 only. If any of the R_is is 1, a valid key is detected.
     ⇒ Set V = 1 and output the corresponding ***N***.
  3. If no key is detected in the first column, apply a 1 to C_1 and repeat.
  4. If no key is detected in the second column, repeat for C_2.
  5. When a valid key is detected, apply 1s to C_0, C_1, and C_2 and wait until no key is pressed.
     - Ensure that only one valid signal is generated each time a key is pressed.



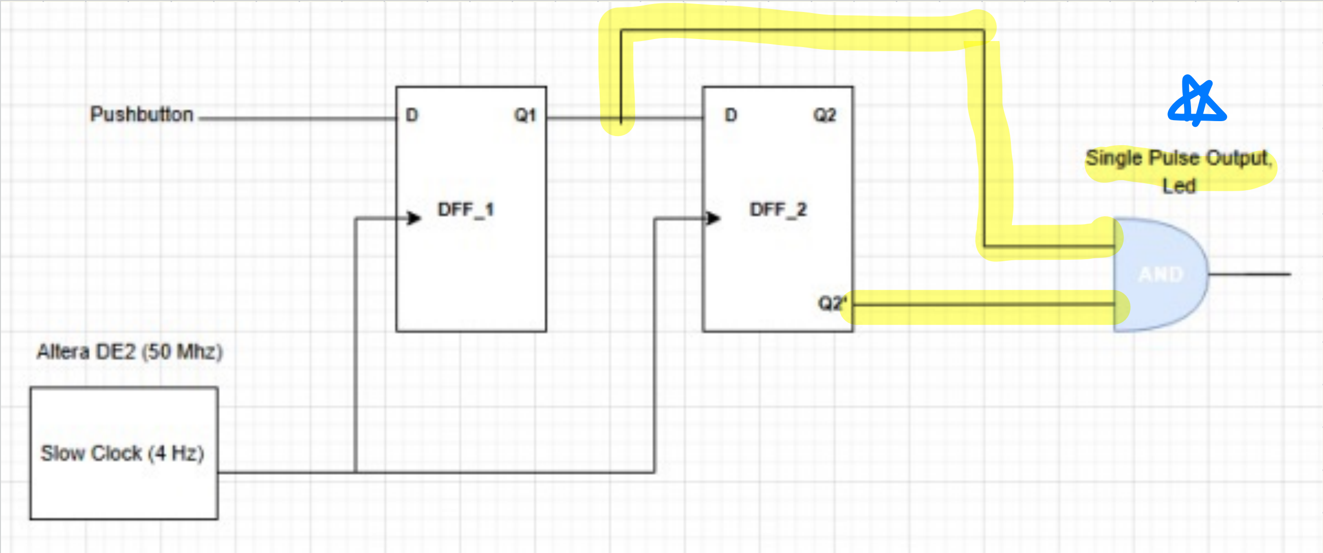
Debouncer

- Debouncer:
  - ***Debounce*** the keys and ***synchronize*** the circuit to avoid malfunctions due to switch bounce.
  - creates a signal K when a key has been pressed and a signal Kd after it has been debounced.

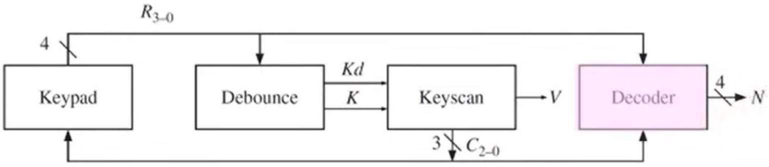
~~Debouncing and synchronizing ckt (p.230, Fig 4-22)~~

this isn't a debouncing circuit, this is a synchronization circuit

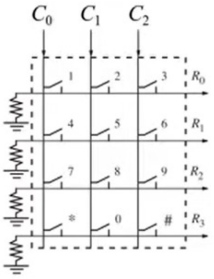
this is a debouncing circuit



Decoder



- Decoder: a ***comb*** circuit
  - determines the key# from the row# and column# using a ***truth table*** that has one row for each of the 12 keys.
    - The remaining rows in the table have ***don't care*** outputs (Assumption: only 1 key is pressed at a time)
  - Since it is a comb ckt, its output will changes as the keypad is scanned.
  - At the time a valid key is detected ($K = 1$ and $V = 1$), its output will have the correct value and this value can be saved in a register .



	C_0	C_1	C_2
R_0	1	2	3
R_1	4	5	6
R_2	7	8	9
R_3	*	0	#

- Truth table and logic equations:

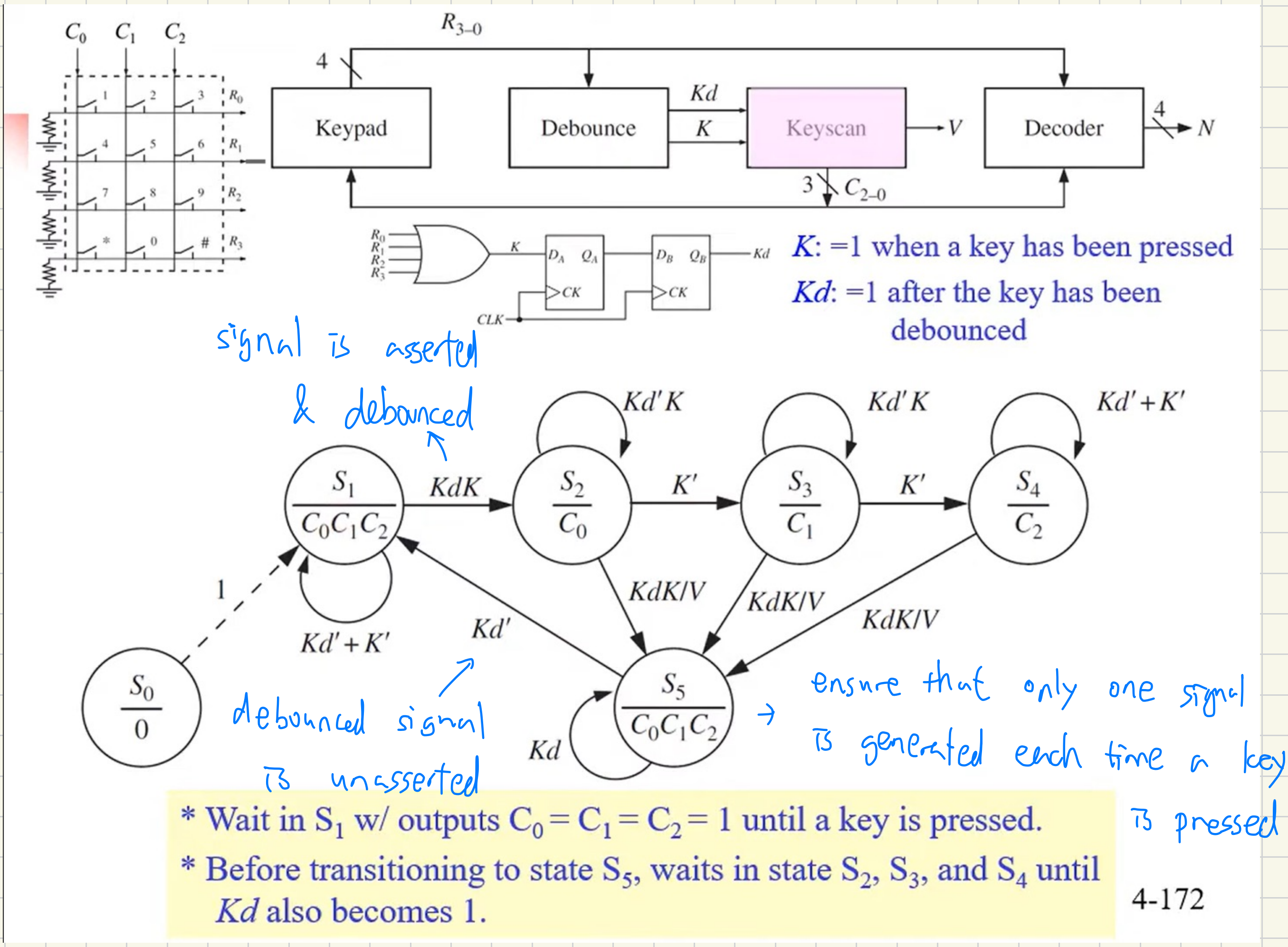
R_3	R_2	R_1	R_0	C_0	C_1	C_2	N_3	N_2	N_1	N_0	
0	0	0	1	1	0	0	0	0	0	1	(1)
0	0	0	1	0	1	0	0	0	1	0	(2)
0	0	0	1	0	0	1	0	0	1	1	(3)
0	0	1	0	1	0	0	0	1	0	0	(4)
0	0	1	0	0	1	0	0	1	0	1	(5)
0	0	1	0	0	0	1	0	1	1	0	(6)
0	1	0	0	1	0	0	0	1	1	1	(7)
0	1	0	0	0	1	0	1	0	0	0	(8)
0	1	0	0	0	0	1	1	0	0	1	(9)
1	0	0	0	1	0	0	1	0	1	0	(*)
1	0	0	0	0	1	0	0	0	0	0	(0)
1	0	0	0	0	0	1	1	0	1	1	(#)

$$N_3 = R_2C_0' + R_3C_1'$$

$$N_2 = R_1 + R_2C_0$$

$$N_1 = R_0C_0' + R_2'C_2 + R_1'R_0'C_0$$

$$N_0 = R_1C_1 + R_1'C_2 + R_3'R_1'C_1'$$



K: =1 when a key has been pressed

Kd: =1 after the key has been debounced

* Wait in S_1 w/ outputs $C_0 = C_1 = C_2 = 1$ until a key is pressed.

* Before transitioning to state S_5, waits in state S_2, S_3, and S_4 until *Kd* also becomes 1.

4-172

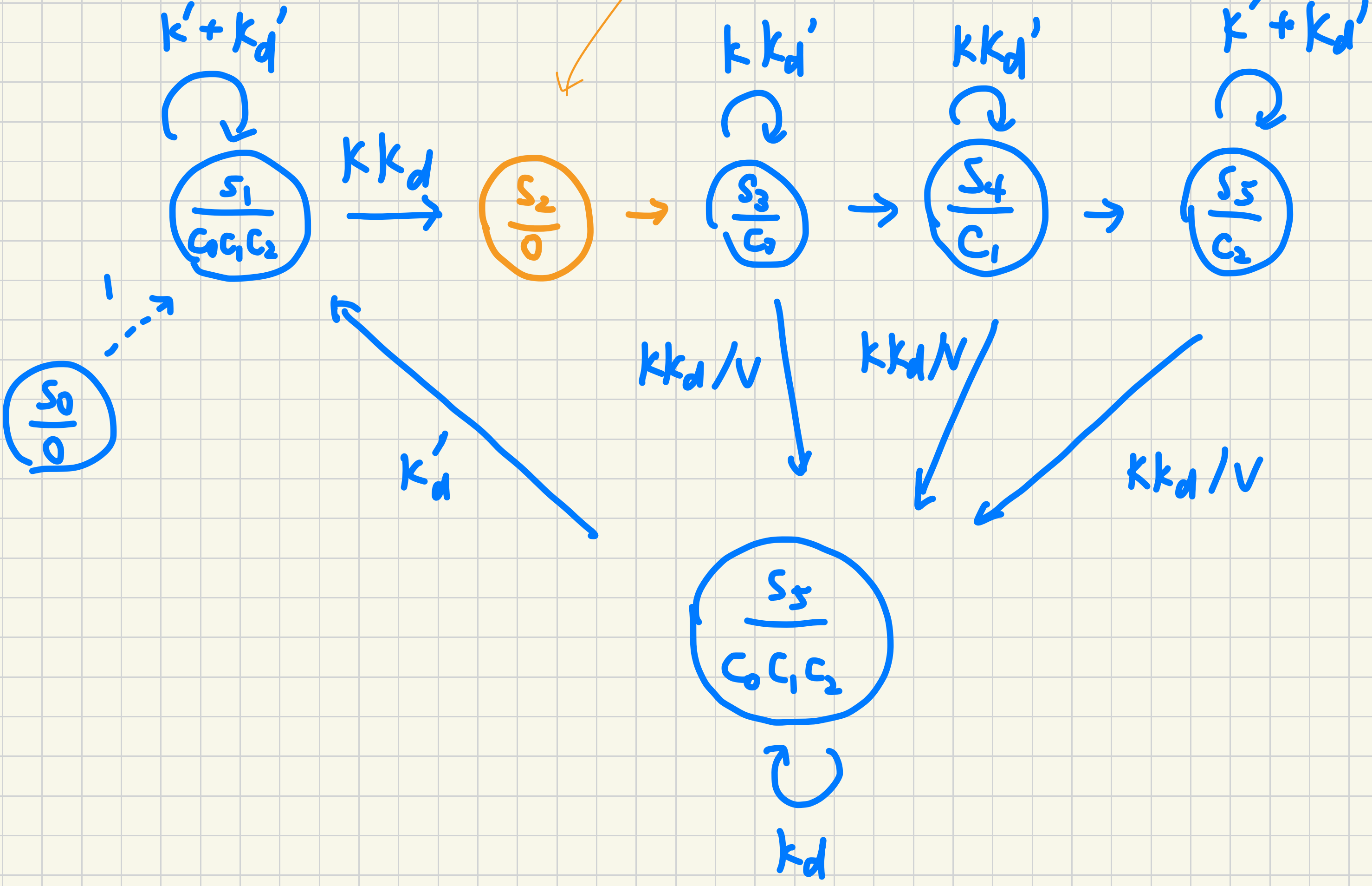
signal is asserted & debounced

debounced signal is unasserted

→ ensure that only one signal is generated each time a key is pressed

my state diagram

I add this state considering my debouncing circuit design


$K'+K_d'$
S_1 / $C_0C_1C_2$
KK_d
S_2 / 0
KK_d'
S_3 / C_0
KK_d'
S_4 / C_1
$K'+K_d'$
S_5 / C_2
1
S_0 / 0
K_d'
KK_d/V
KK_d/V
KK_d/V
S_5 / $C_0C_1C_2$
K_d