Check for right locations first

(mark as seen) delete letters that match from guess & answer

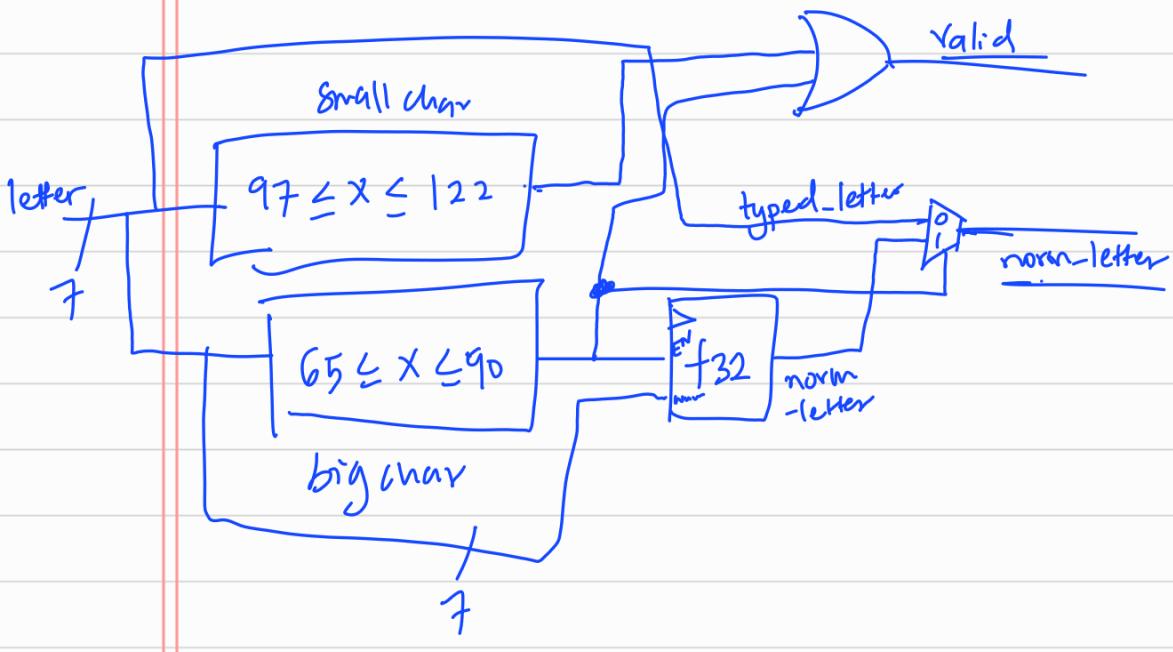
Check the rest of the wrong locations next

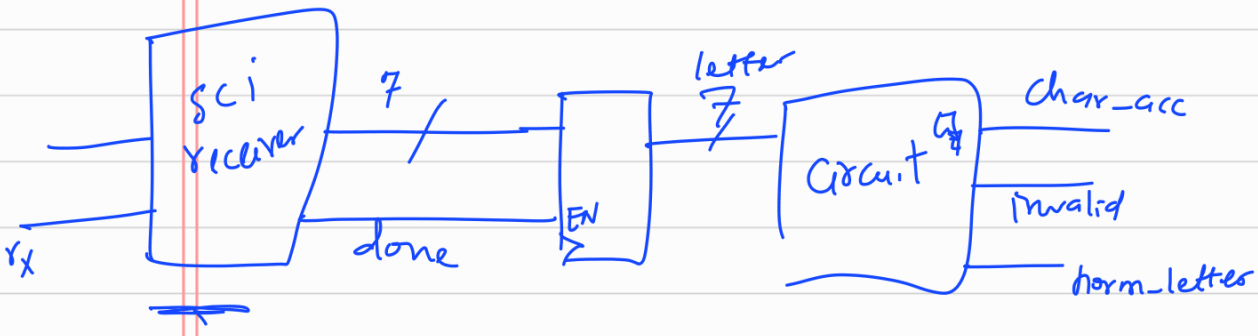
delete letter that exists in both

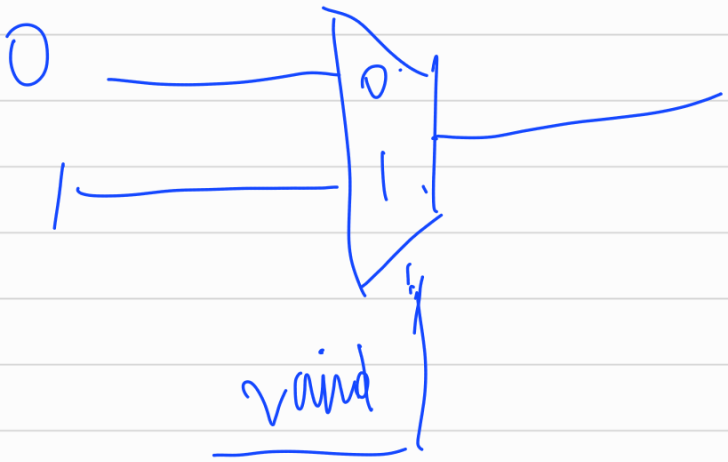
loop till end of either word.

(65 → 90) (

If its small char (97 → 122)
then valid
if within 65 → 90 big char
then add 32
return valid







Steps:

Rx

Tx , wiring

Rx talks to Tx

basic word logic

Rx → word logic → Tx

rom

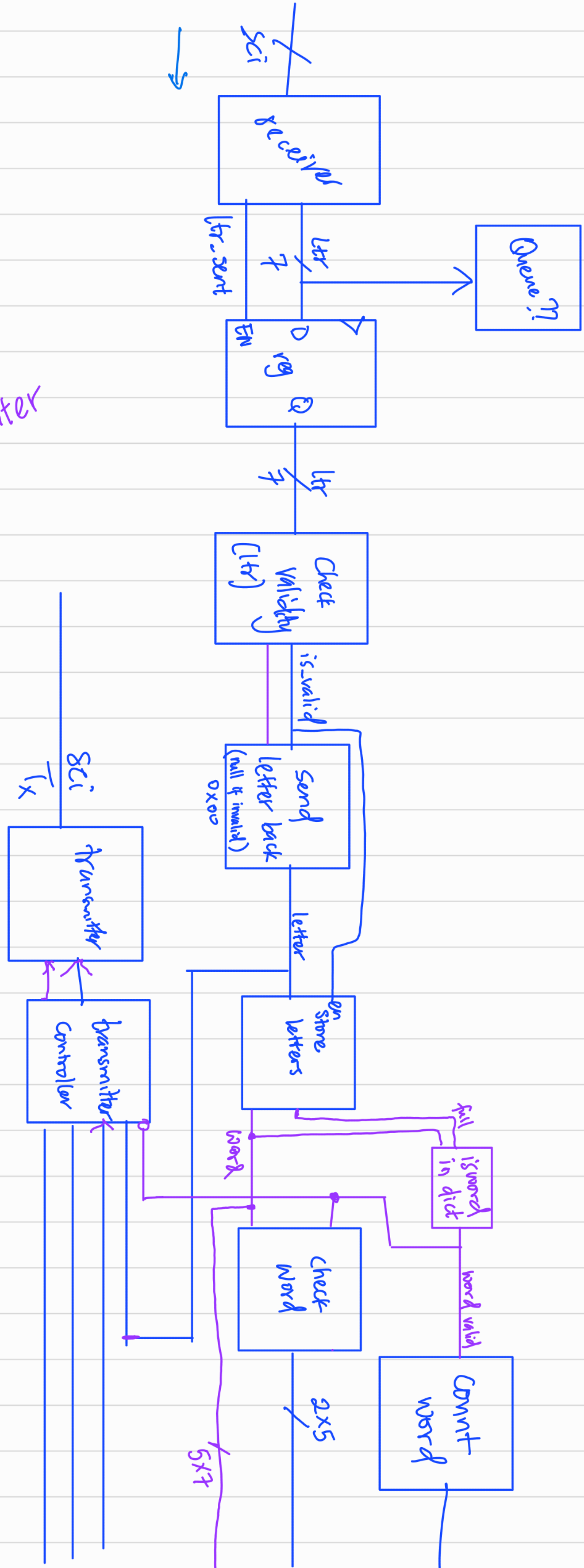
Rx → word logic → rom → Tx

wait 2 cycles before reading

account for backspace

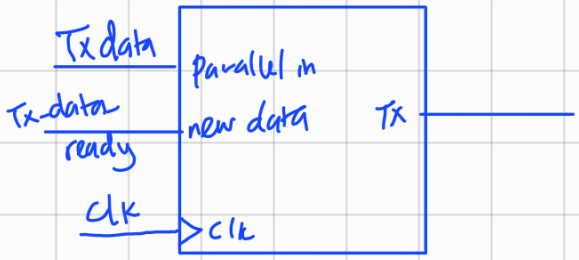
don't validate until enter

rom for win / lose

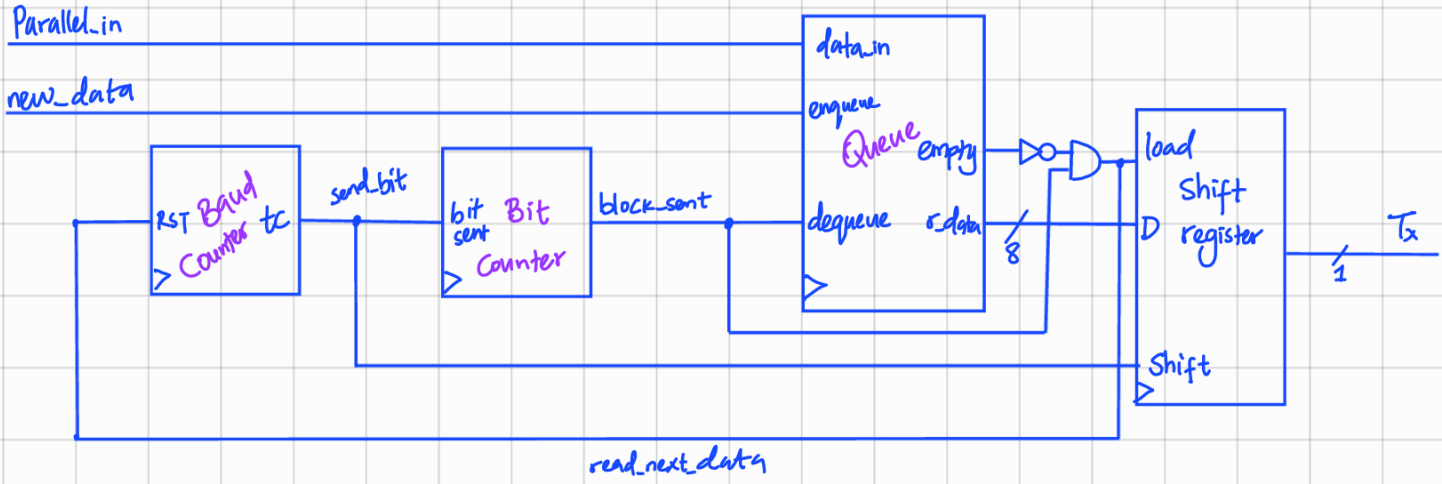


tc
2x5
5x7 mirror
Determine color
wrong
Connect
win msg
lose
lose Message

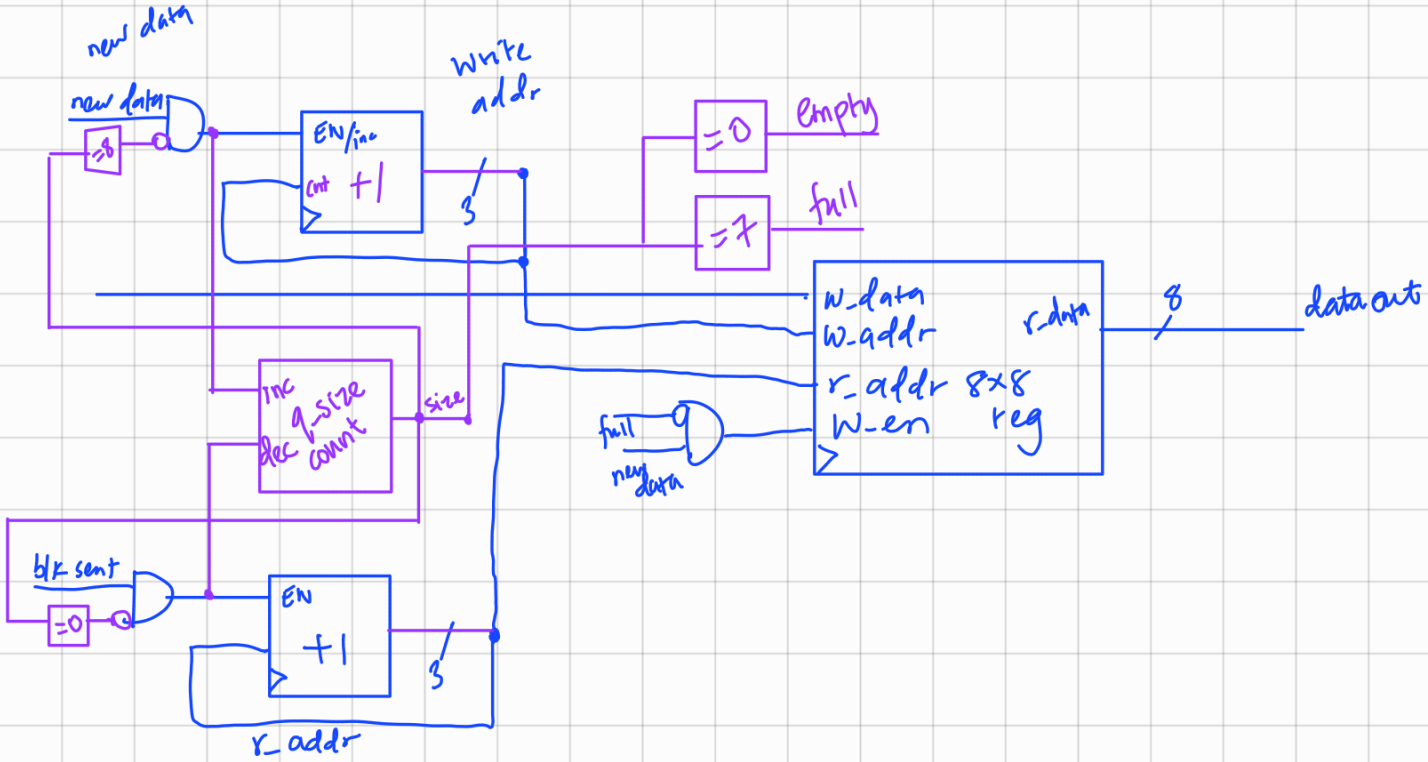
1a.



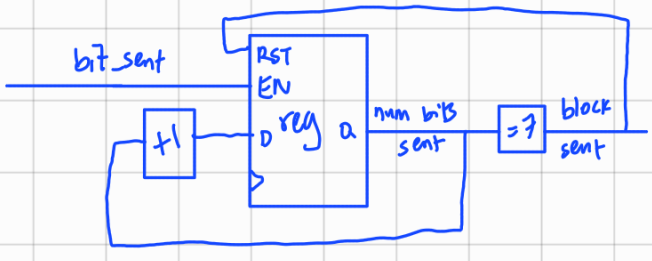
Transmitter



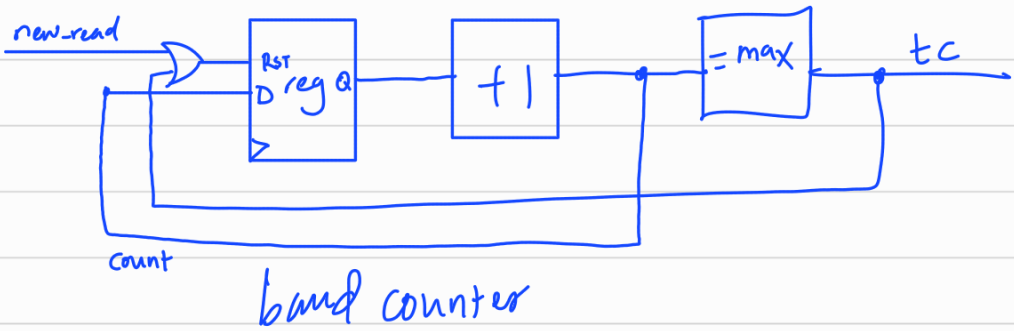
Queue



Bit Counter



2.



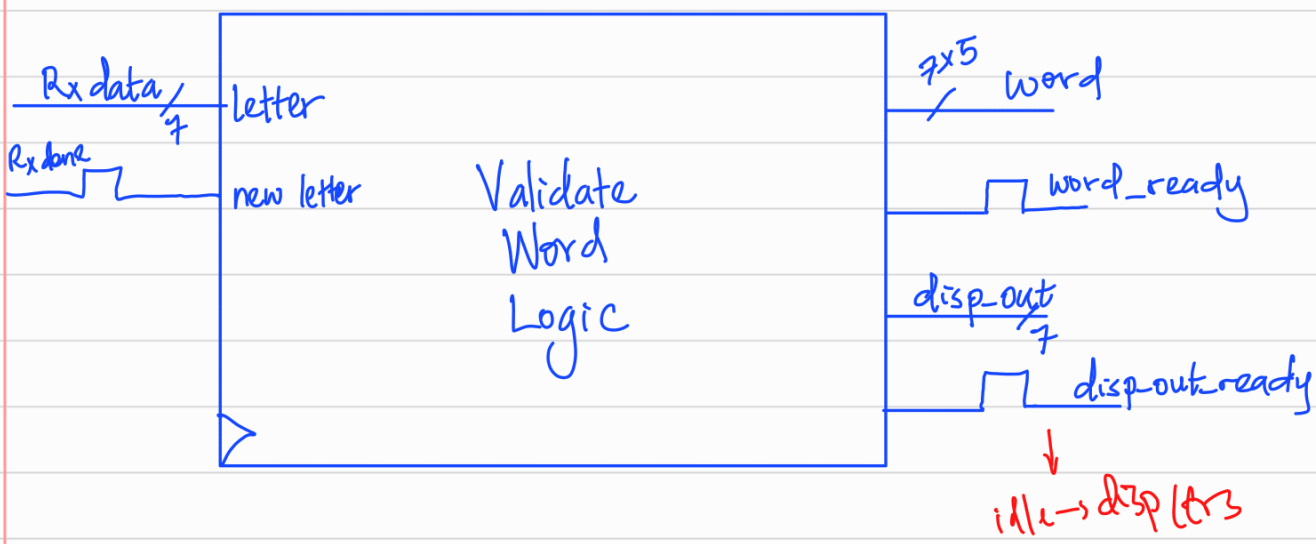
baud counter

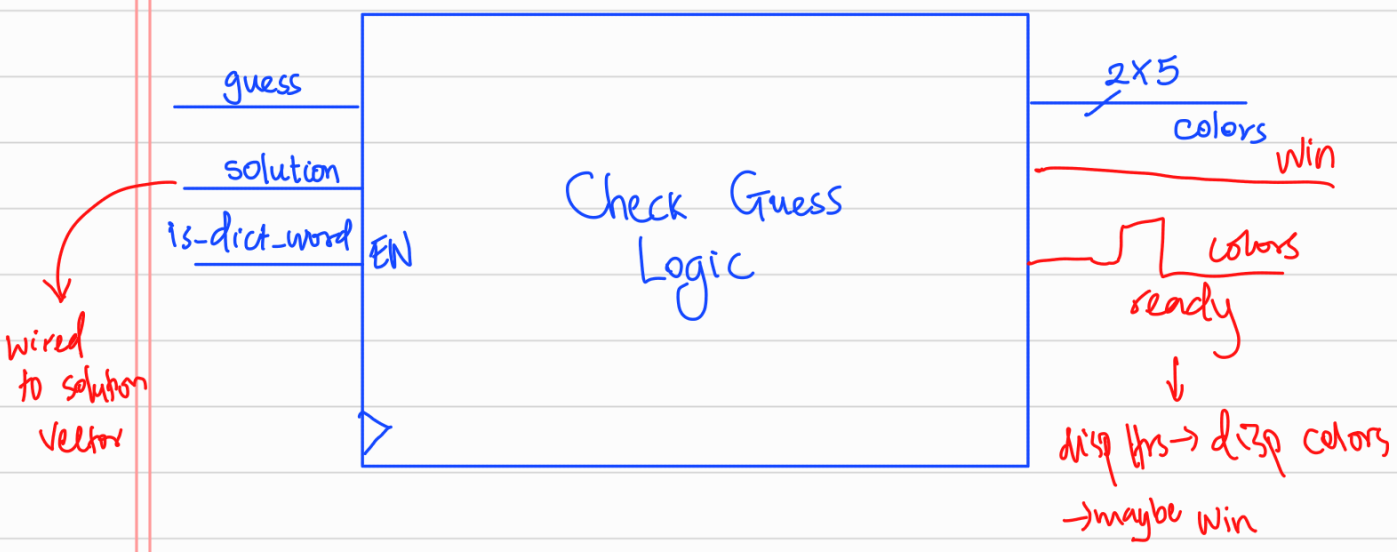
https://www.edaplayground.com/x/9bFb

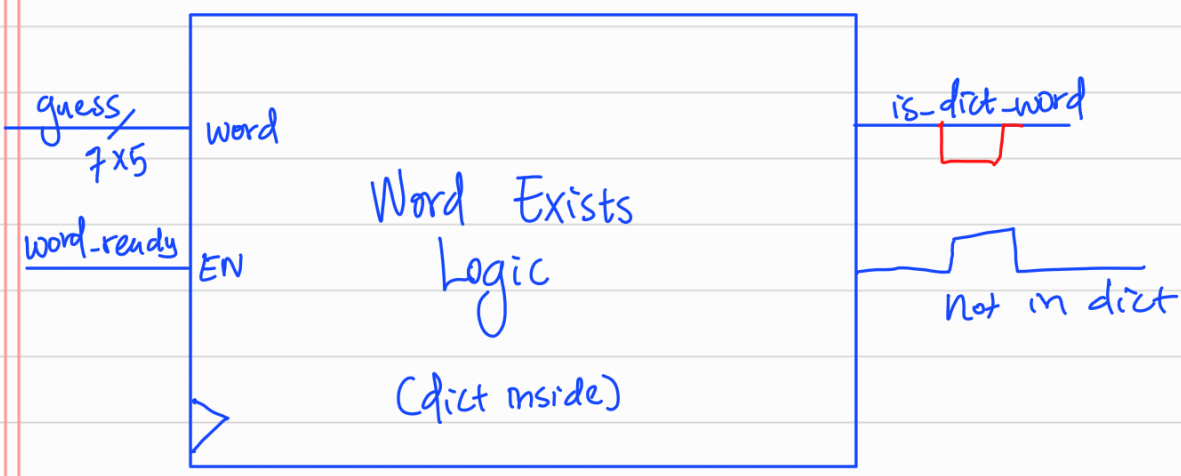
inputs
what is typed
backspace ;
enter

letter is accepted
store the word

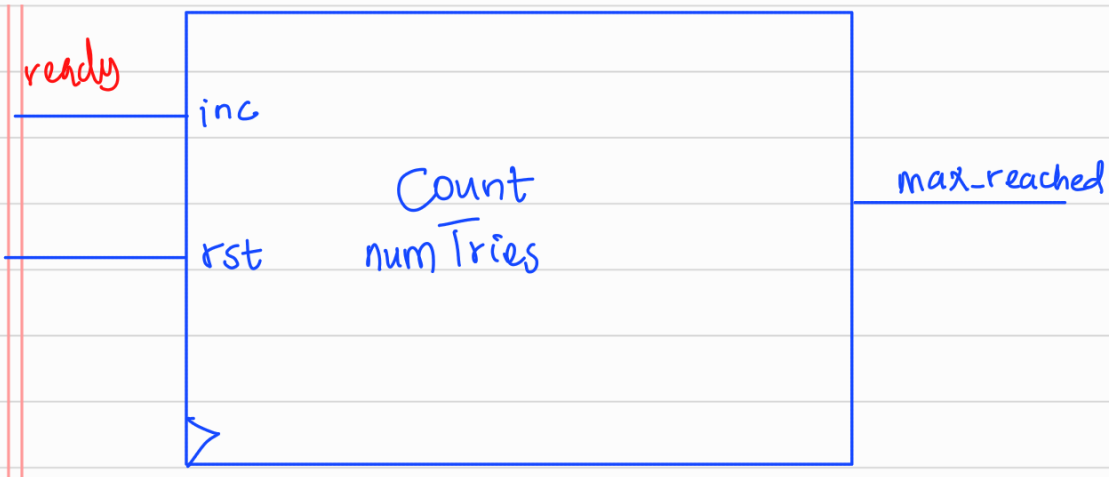
Determines
is a letter, so store

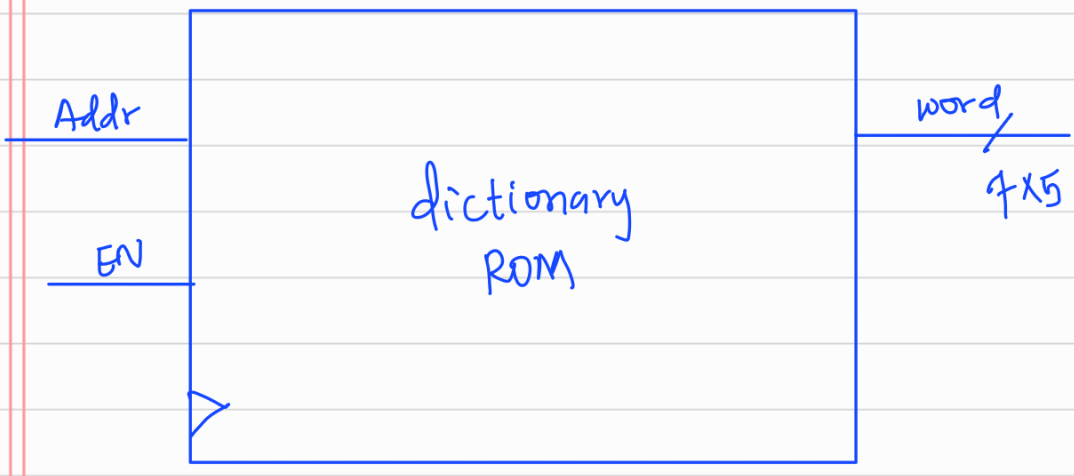

Rx data
7
Letter
Rx done
new letter
Validate Word Logic
7x5
word
word_ready
disp_out
7
disp_out_ready
idle → disp ltrs

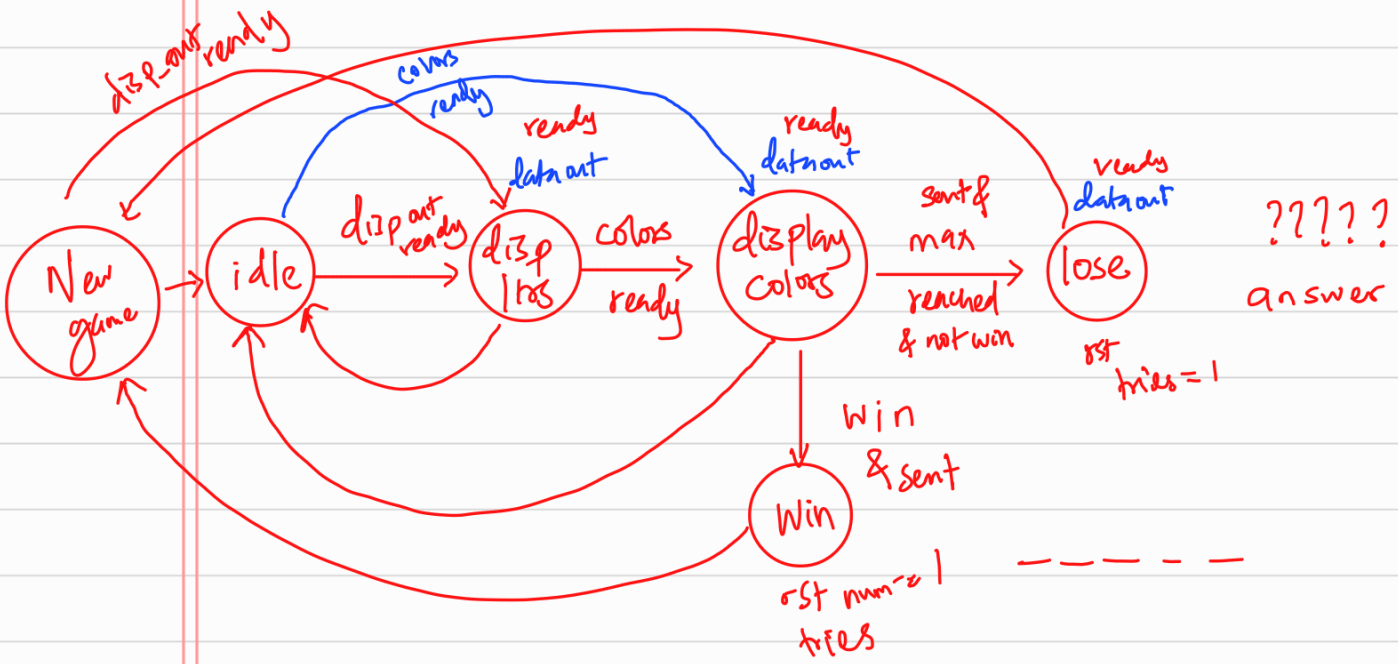

guess
solution
is_dict_word
EN
Check Guess Logic
2x5
colors
win
colors ready
disp ltrs → disp colors
→ maybe win
wired to solution vector


guess
7x5
word
word_ready
EN
Word Exists Logic
(dict inside)
is_dict_word
not in dict


guess
Counter
EN
dict
data
=


ready
inc
rst
Count
numTries
max_reached


Addr
EN
dictionary
ROM
word
7X5

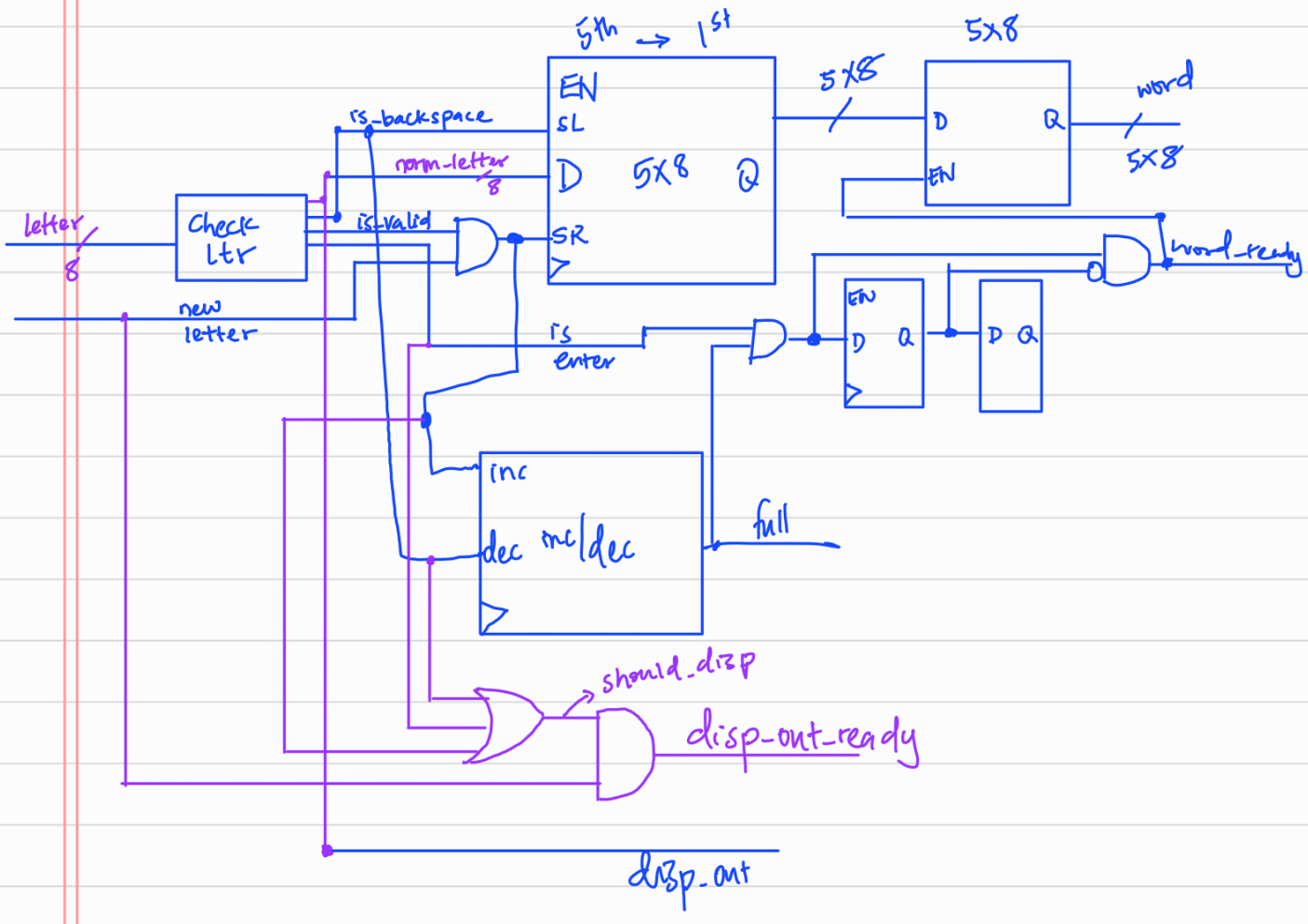

disp_out ready
colors ready
ready
data out
ready
data out
ready
data out
New game
idle
disp out ready
disp logs
colors ready
display colors
sent & max reached & not win
lose
rst tries = 1
?????
answer
win & sent
Win
rst num = 1 tries


12x8
Pin
ready
Tx ready
Tx data
Sent

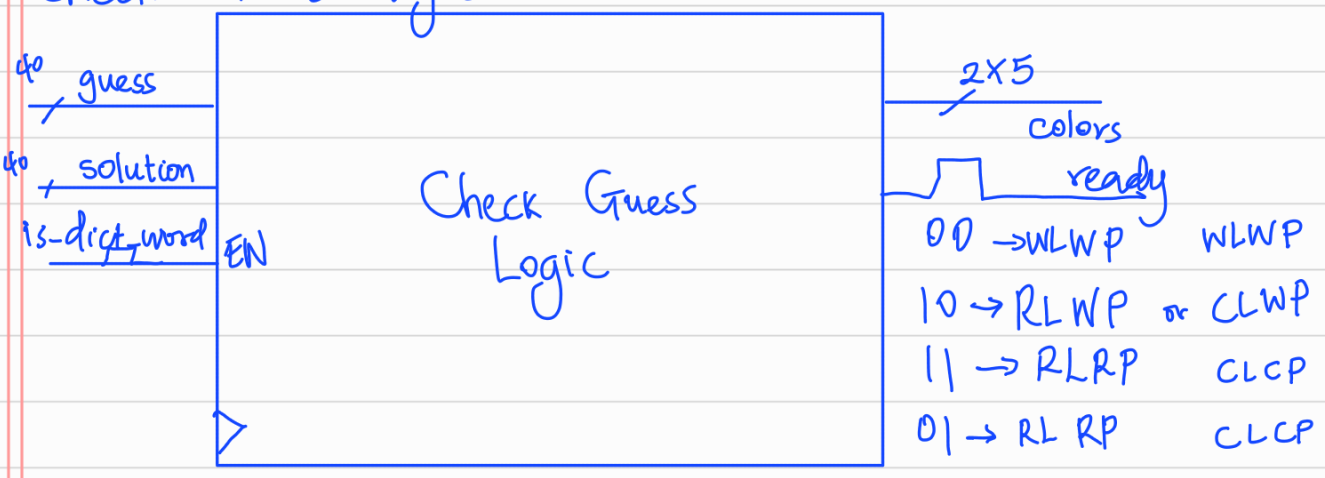
Validate word logic

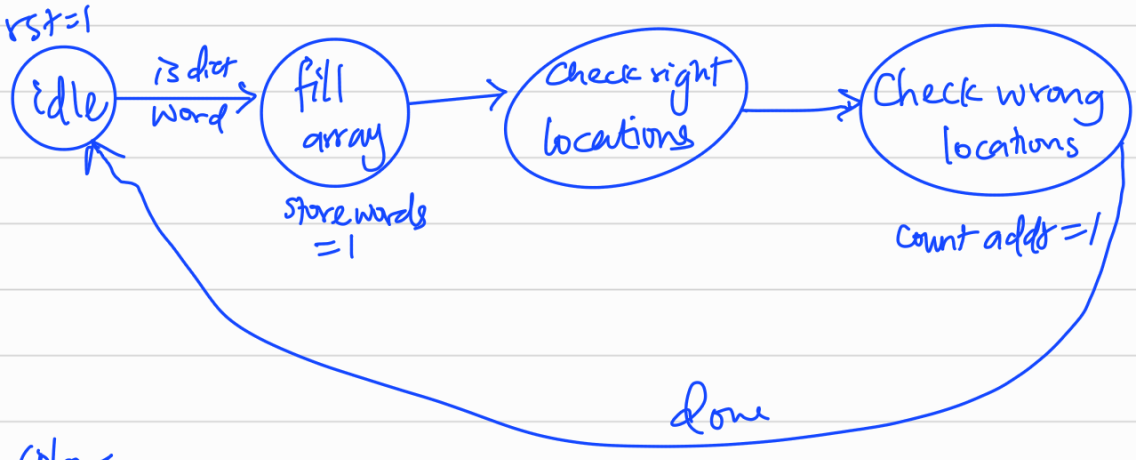
e.g. word = a p p l e

e (p P a



Check Guess logic





colors
L P

CL	CP	blk	yellow	green
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	0	0	1

blk: $\overline{LP}$

yellow: $L\overline{P}$

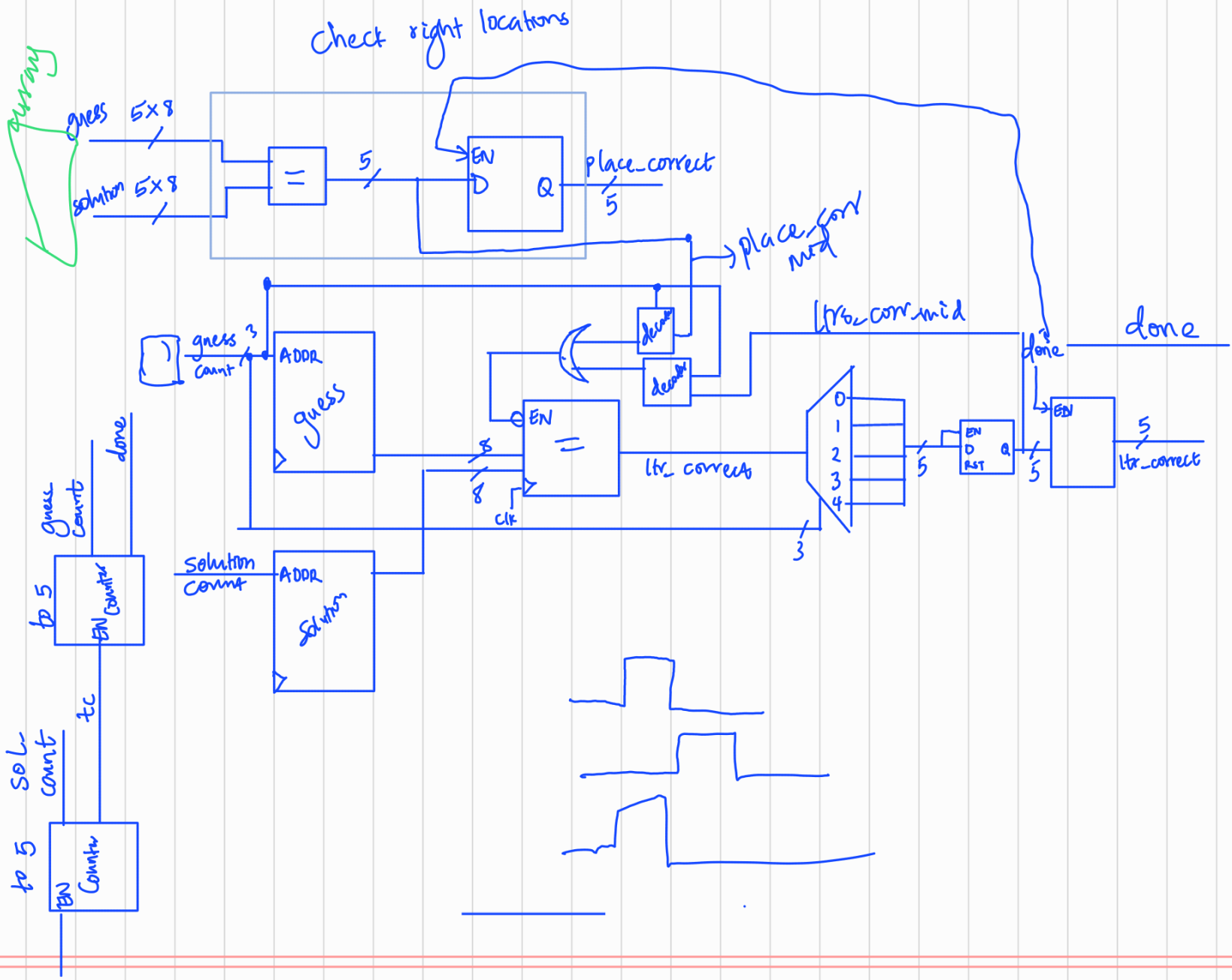
green: P

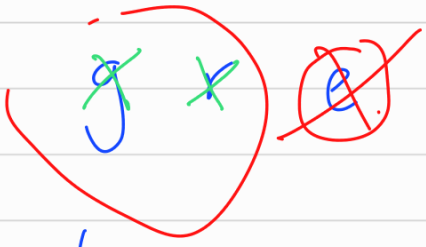
5 bytes
40 bits

0 → 7
8 → 15

Cr_lts: 5 bits
Cr_plc: 5 bits

2^5 (word_loc - x byte size + 7


Check right locations
query
guess 5x8
solution 5x8
=
5
EN
D
Q
place_correct
5
place_correct mid
guess 3
count
ADDR
guess
decode
decode
ltr_corr_mid
done
done
EN
=
8
8
CLK
ltr_ correct
0
1
2
3
4
3
5
EN
D
RST
Q
5
EN
5
ltr_correct
guess count
done
to 5
EN Counter
tc
sol_ count
to 5
EN Counter
Solution count
ADDR
Solution



	a	p	p	l	e
	10	00	00	00	10
CP	0	0	0	0	0
CL	1	0	0	0	1

	h	e	e	d	s
	h	e	e	d	s
	00	00	11/01	00	00
	0	0	1	0	0
	0	0	0/1	0	0

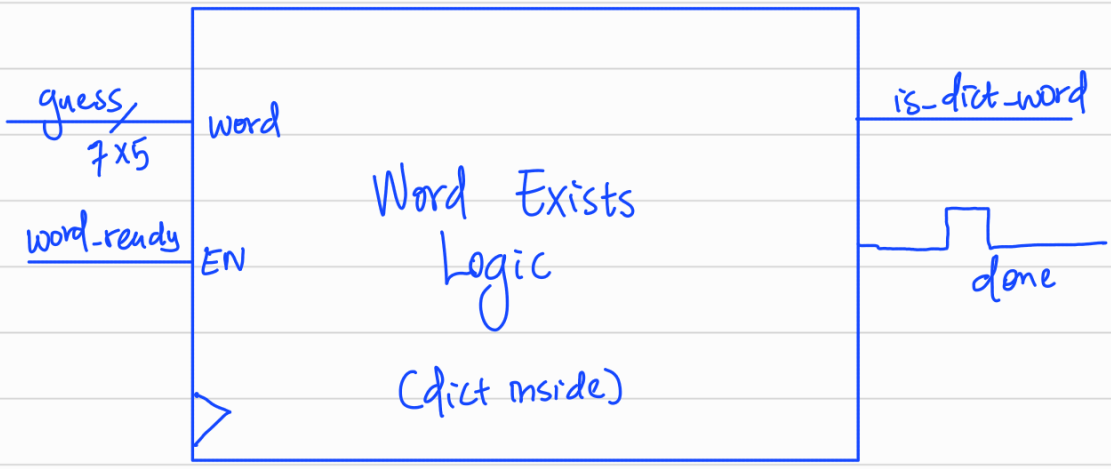
	g	r	a	t	e
	11/01	11/01	10	10	10
CP	1	1	0	0	0
CL	0/1	0/1	1	1	1

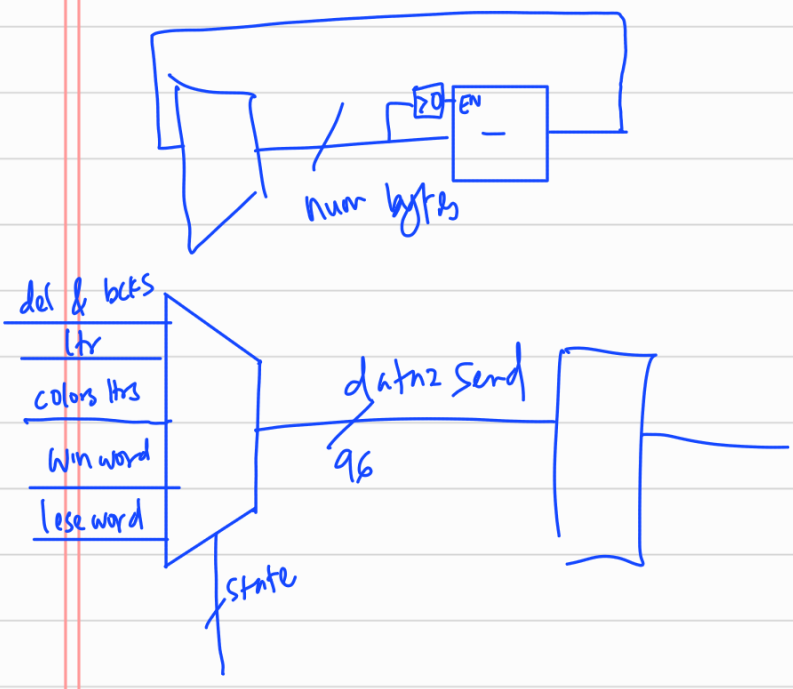
1 → 2

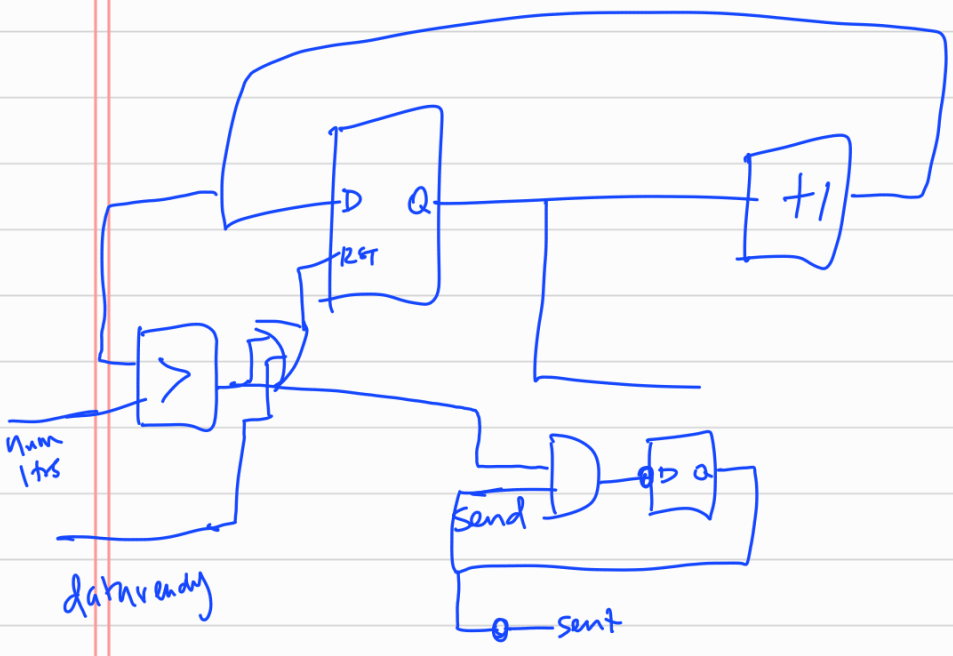
e vs e

guess_ltr correct (g_idx)

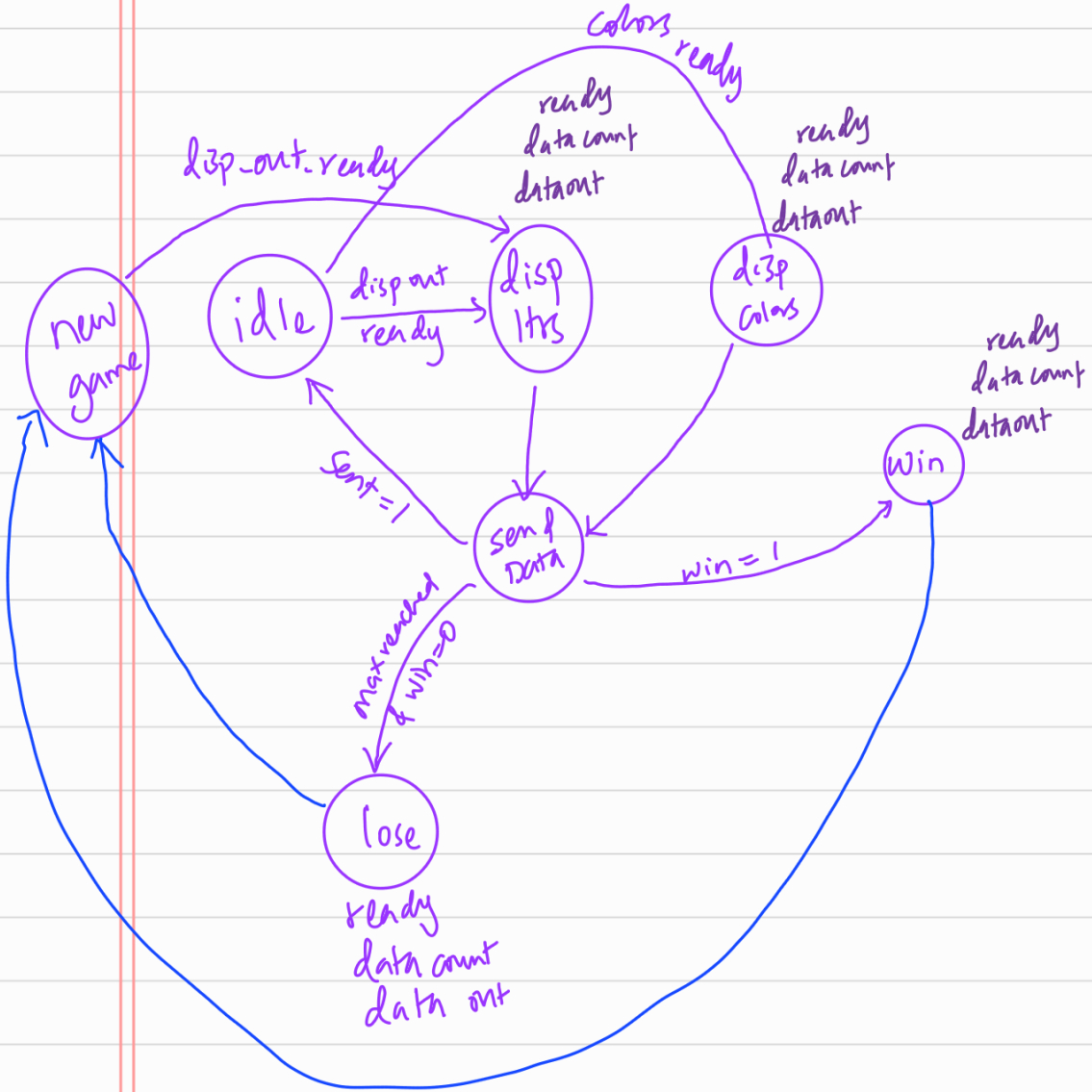
sol_ltr_correct (s_idx)

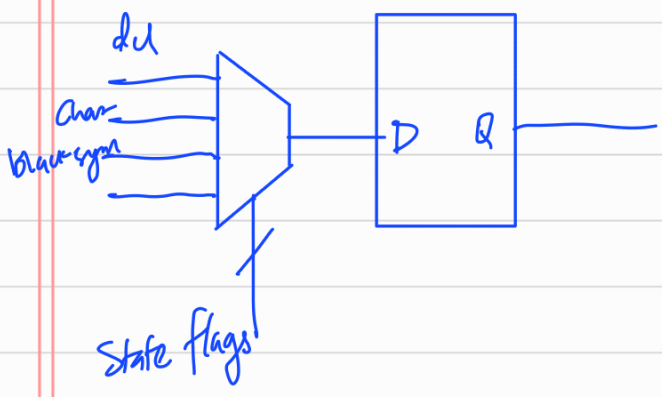

guess
7x5
word
word_ready
EN
Word Exists
Logic
(dict inside)
is_dict_word
done


EN
num bytes
del & bcks
ltr
colors lfts
win word
lose word
data2 Send
96
state


D
Q
REG
+1
num ltrs
Send
sent
data ready


disp_out_ready
colors ready
ready data out
ready data out
ready data out
New game
idle
disp out ready
disp ltrs
colors ready
display colors
sent & max reached & not win
lose
?????
answer
rst tries=1
win & sent
Win
rst num=1 tries


colors ready
ready data count data out
ready data count data out
disp_out_ready
new game
idle
disp out ready
disp ltrs
disp colors
ready data count data out
sent=1
send data
win=1
mat reached & win=0
lose
ready data count data out


dd
char
block sync
D
Q
state flags

new ltr	full	valid letter	enter	backspace	readyout
1	0	0	0	0	0
	0	0	0	1	1
1	0	0	1	0	0
	0	0	1	1	1
1	0	1	0	0	1
	0	1	0	1	1
1	0	1	1	0	1
	0	1	1	1	1
	1	0	0	0	0
1	1	0	0	1	1
	1	0	1	0	1
	1	0	1	1	1
	1	1	0	0	0
1	1	1	0	1	1
	1	1	1	0	1
	1	1	1	1	1

EB

FV \ EB	00	01	11	10
00	0			0
01				
11	0			
10	0			

new ltr & not $(F\overline{EB} + \overline{FVB})$