Implement a Load-Store Unit (LSU) to manage memory-mapped in Table 1.

Boundary address	Mapping
0x7820 -- 0xFFFF	(Reserved)
0x7810 -- 0x781F	Buttons
0x7800 -- 0x780F	Switches *(required)*
0x7040 -- 0x70FF	(Reserved)
0x7030 -- 0x703F	LCD Control Registers
0x7020 -- 0x7027	Seven-segment LEDs
0x7010 -- 0x701F	Green LEDs *(required)*
0x7000 -- 0x700F	Red LEDs *(required)*
0x4000 -- 0x6FFF	(Reserved)
0x2000 -- 0x3FFF	Data Memory (8KiB using SDRAM) *(required)*
0x0000 -- 0x1FFF	Instruction Memory (8KiB) *(required)*

Table 1: LSU memory mapping

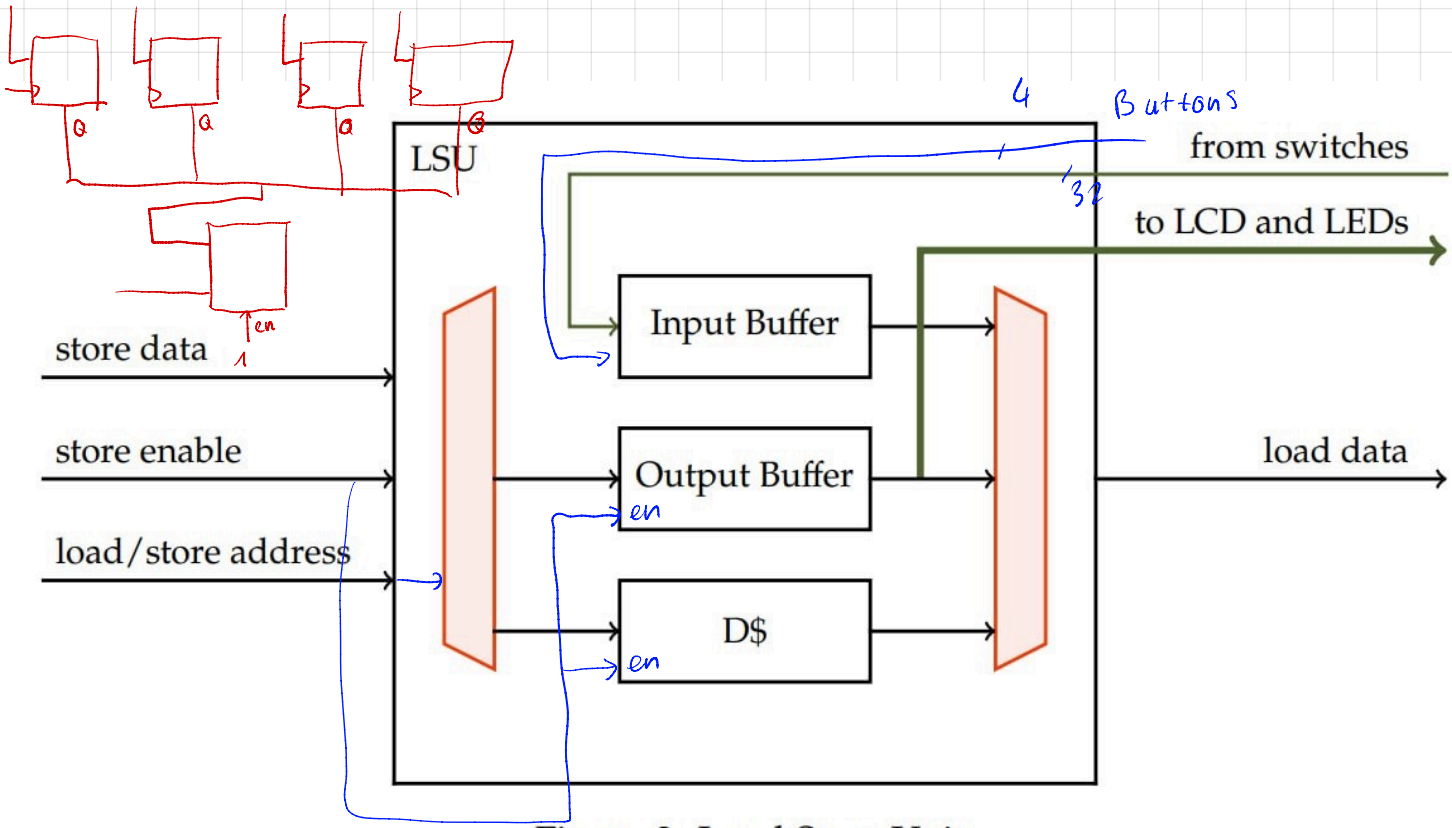
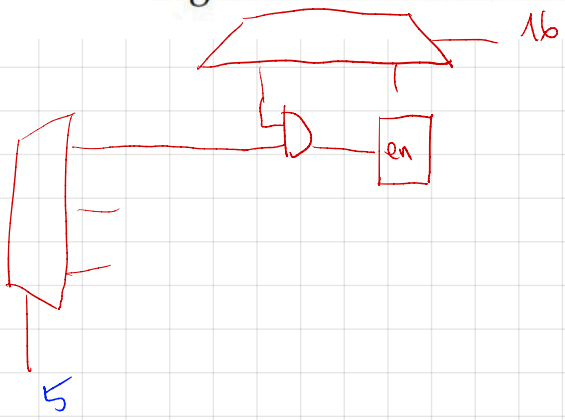


Figure 3: Load Store Unit



Bank				
input	78 - FF	0111	1000 → X111	
output	70	0111	0000	0111 0000
sdram	20 - 3F	0010 0011	0000 1111	001X XXXX [15:8]

Bank_Sel		input	output	sdram
X111	1XXX	0	0	0
0111	0000	1	0	0
001X	XXXX	0	1	0
		0	0	1

vẽ bảng xác tính

Decoder_buf

i_lsu_wren

output_buf_en

sdram_en

addr[15:8]

addr[15:8]

	15	14	13	12	11	10	9	8	output	sd
32 → 63	0	0	1	X	X	X	X	X	0	1
112	0	1	1	1	0	0	0	0	1	0

output_buf_en = $\overline{15}.(14.13.12).(11+10+9+8).$ i_lsu_wren

sdram_en = $\overline{15}.\overline{14}.13.$ i_lsu_wren

Decoder_buf

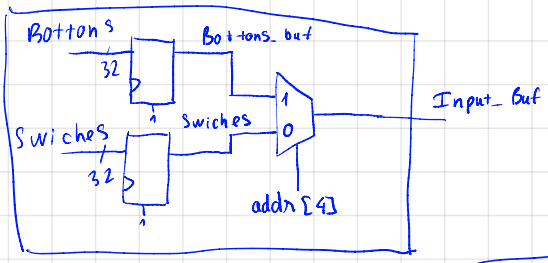
Input_ buf

[7:0] reg [3:0] [8]

4 (3 2 1 0)

7810 - 781F	Bottons
7800 - 780F	Swiches

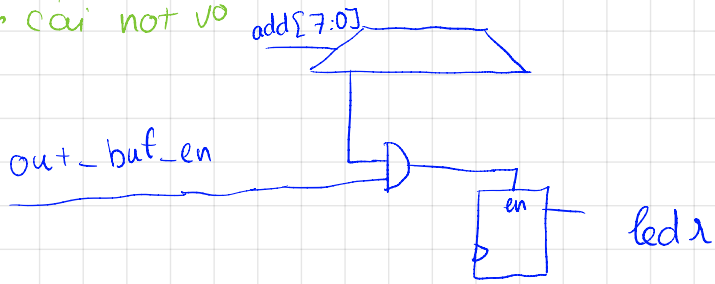
① xxxx



Output_ buf

thêm cái not vô

wb_ sel



led r

ledg

	addr [7:4]	
0	0 0 0 0	Red_ led
1	0 0 0 1	Green_ led
2	0 0 1 0	7 seg
3	0 0 1 0	LCD

0x 7000 → 0x 700F

0	4	8	C
0000	0100	1000	1010

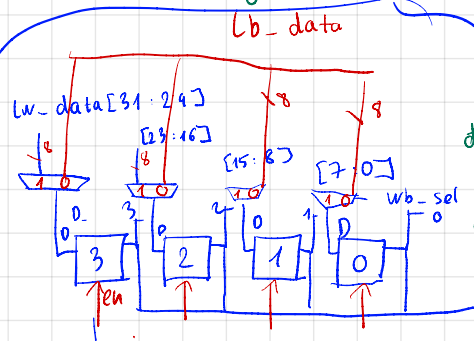
chọn thanh → chọn byte

0	00	00
		01
		10
		11
4	01	00
		01
		10
		11
8	10	00
1		

thanh 1 led đỏ 32 bit

thanh 2 32 bit

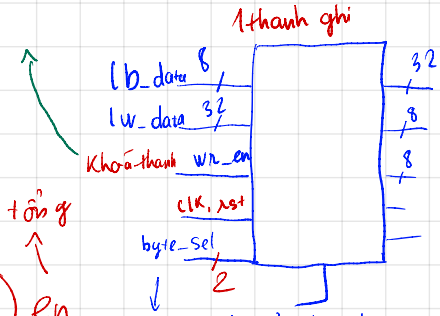
reg có khả năng nạp byte hoặc nạp word



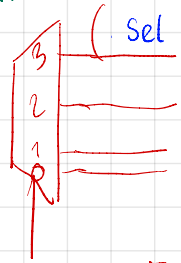
nạp word wb_sel=1 để ghi thì phải cho wr_en = 1

chọn byte để đọc ghi byte_sel [1:0]

(sel + wb_sel) wr_en



Khóa tổng



Với reg này load thì đọc được đúng giá trị hiện tại luôn

Nhưng store phải đợi clk tiếp theo byte_sel [1:0]

reg_ out 0
1 → nối ra 32
2
3
4

D_ store ra ngoài vi

D_ load

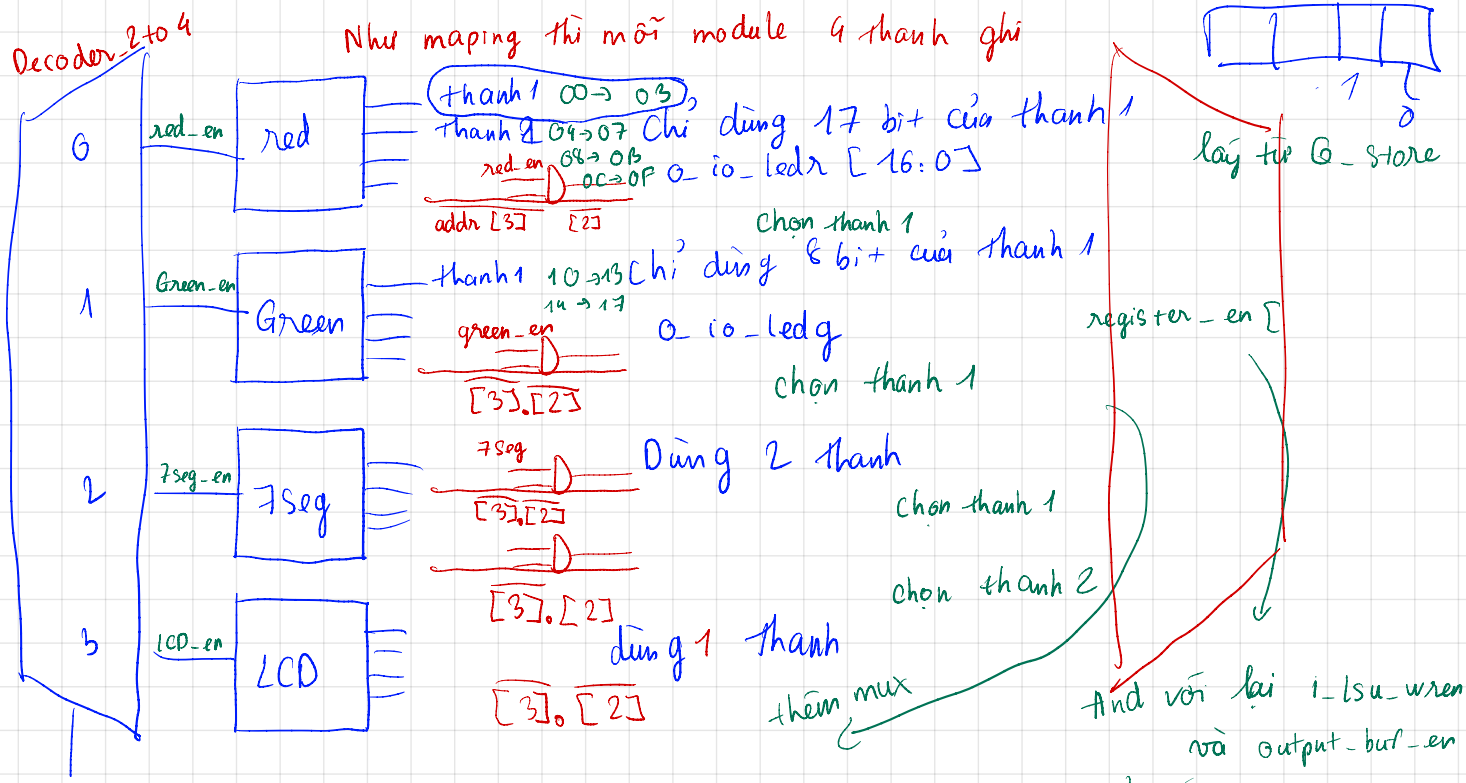
wb byte_sel [1:0]

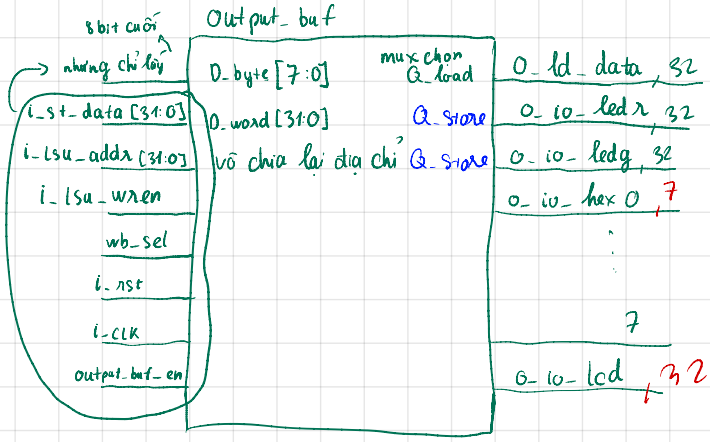
2 bit cuối addr

lb làm sau

byte_sel [1:0] | Sel

00	1000
01	0100
10	0010
11	0001





cắt đi bit cao

Chú ý Array

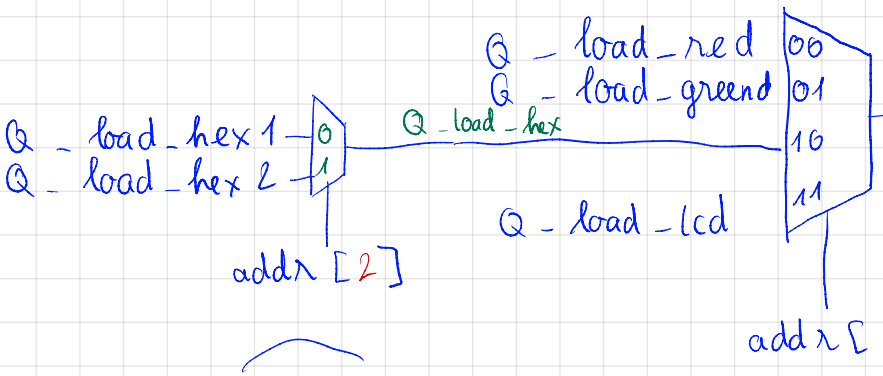
[6:0] o_io_hex [7:0]

mỗi con led 8 con led

có 7 bit đủ 7 đoạn

làm bộ mux chọn thanh 4 to 1 ở mỗi Led, 7reg, LCD

o_ld_data sel



4 F 100 1111

6 C 110 1100

ở ngõ ra sau khi làm xong sdram thì mới dùng mux

0x7000 : 8 con led đỏ cuối

7001 : 8 con led đỏ thứ 2

7810 - 781F | Buttons → lw 0x7810 → 32 bit

7800 - 780F | Switches → lw 0x7800 → 32 sw

① xxxx

hex 0 = 0x7020

hex 1 = 0x7021

hex 2 = 7022

hex 3 = 7023

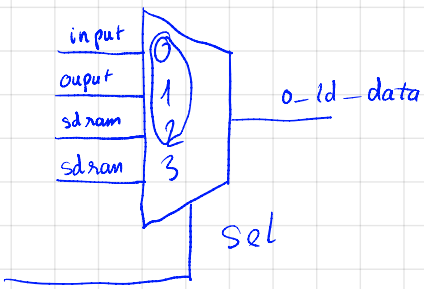
hex 4 = 0x7024

hex 5 25

hex 6 26

hex 7 27

Out_mux



Bank

input	78 - FF	0111 1000 → X111		
output	70	0111 0000	0111 0000	
sdram	20 - 3F	0010 0000 0011 1111	001X XXXX	[15:8]

Bank_sel		input	output	sdram	sel
6	3				
X111	1XXX	1	0	0	0
0111	0000	0	1	0	1
001X	XXXX	0	0	1	2

ip_en = [6] & [5]

op_en = [6] & $\overline{[5]}$

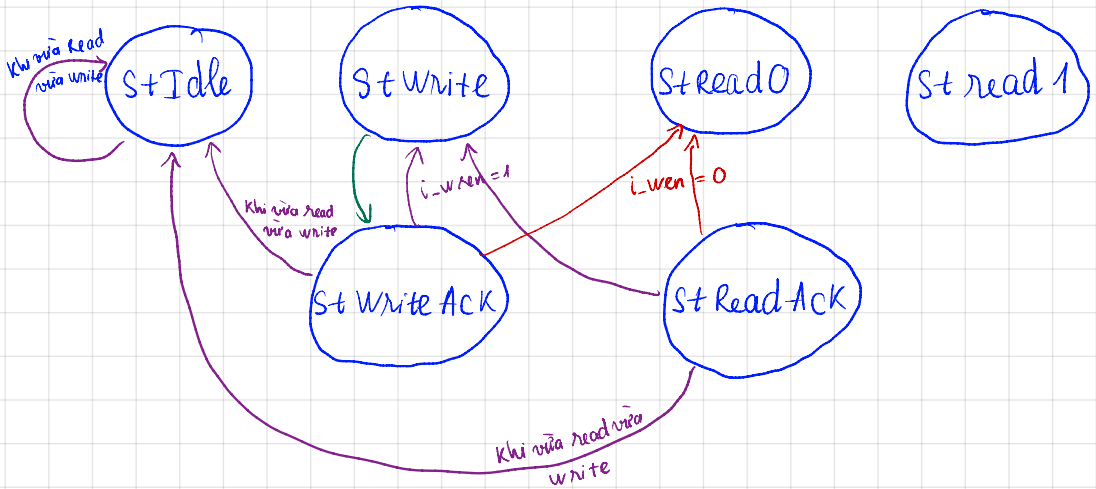
sram_en = $\overline{[6]}$ & [5]

	ip	op	sram	sel [1:0]	
1	0	0	1	10	2
2	0	1	0	01	1
3	1	0	0	00	0

sram \ ip op	00	01	11	10
0				
1				

sel[0] = $\overline{ip}$ & $\overline{op}$ & sram

sel[1] = $\overline{ip}$ & op & $\overline{sram}$

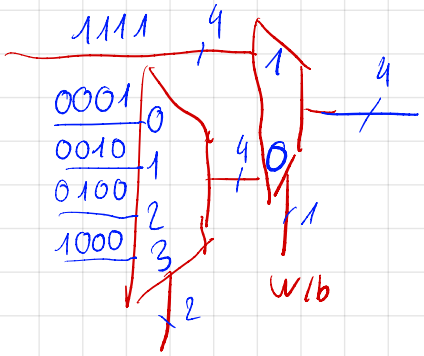


Ghi và đọc đều đọc 2 byte thấp trước nên mới and với 18'h 3FFFE trước rồi khi qua state ACK thì mới tiếp tục đọc ghi với 16 bit cao

d là tiếp theo q là hiện tại

SRAM_DQ thả nổi để k° ảnh hưởng đến data trong sram

wb_sel = 1 → i_BMASK = 1111
0 →



byte_sel [1:0]	i_BMask
00	0001
01	0010
10	0100
11	1000

byte_sel = addr[1:0]