

LCD Display (1)

Hsi-Pin Ma

<http://lms.nthu.edu.tw/course/24953>

Department of Electrical Engineering

National Tsing Hua University

ROM Revisit

ROM Truth Table (Partial)

address (decimal)

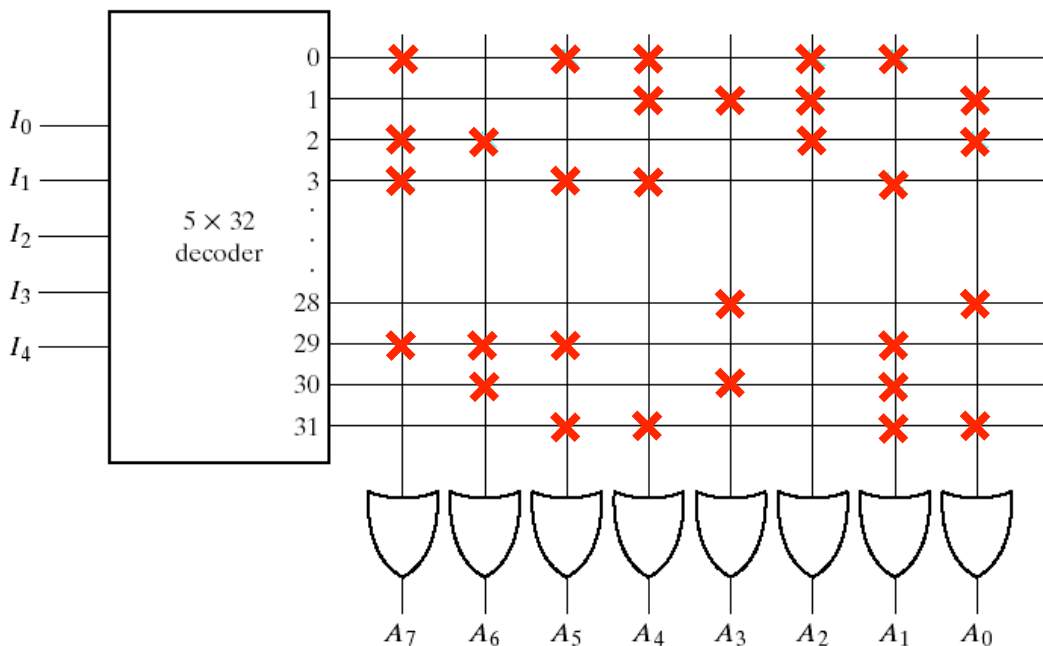
	Inputs					Outputs							
	I4	I3	I2	I1	I0	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	0	1	0	1	1	0	1	1	0
1	0	0	0	0	1	0	0	0	1	1	1	0	1
2	0	0	0	1	0	1	1	0	0	0	1	0	1
3	0	0	0	1	1	1	0	1	1	0	0	1	0
...			⋮					⋮					
28	1	1	1	0	0	0	0	0	0	1	0	0	1
29	1	1	1	0	1	1	1	1	0	0	0	1	0
30	1	1	1	1	0	0	1	0	0	1	0	1	0
31	1	1	1	1	1	0	0	1	1	0	0	1	1

address (k)

$2^k \times n$ ROM

of output bits (n)

$2^5 \times 8$ ROM



COE Format

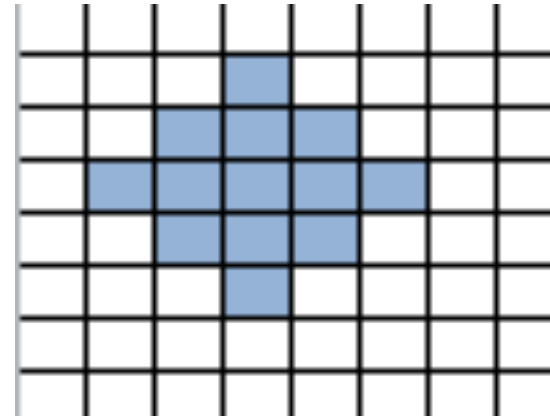
- COE: memory coefficient file
- Two parameter:
 - **memory_initialization_radix**
 - Radix of the values in the *memory_initialization_vector*
 - Ex: 2, 10, or 16
 - **memory_initialization_vector**:
 - Memory content
 - Memory words are separated by **whitespace**
 - You can use comma (,) to help identify the boundary
 - Vector (entire memory) ended by **semicolon**

COE Example



```

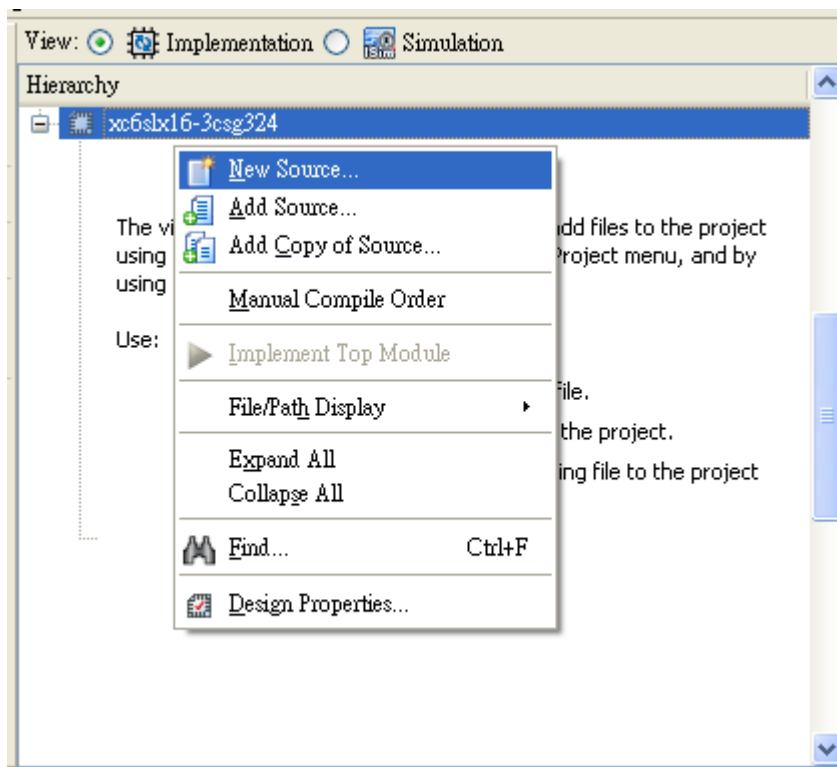
; 8-bitwide by 8-deep RAM
memory_initialization_radix=2;
memory_initialization_vector=
00000000 , ← whitespace
00010000 ,
00111000 ,
01111100 ,
00111000 ,
00010000 ,
00000000 ,
00000000 ;
  
```



You can use ASCII art generator to generate the pictures or use drawing tool to export the figures for you.

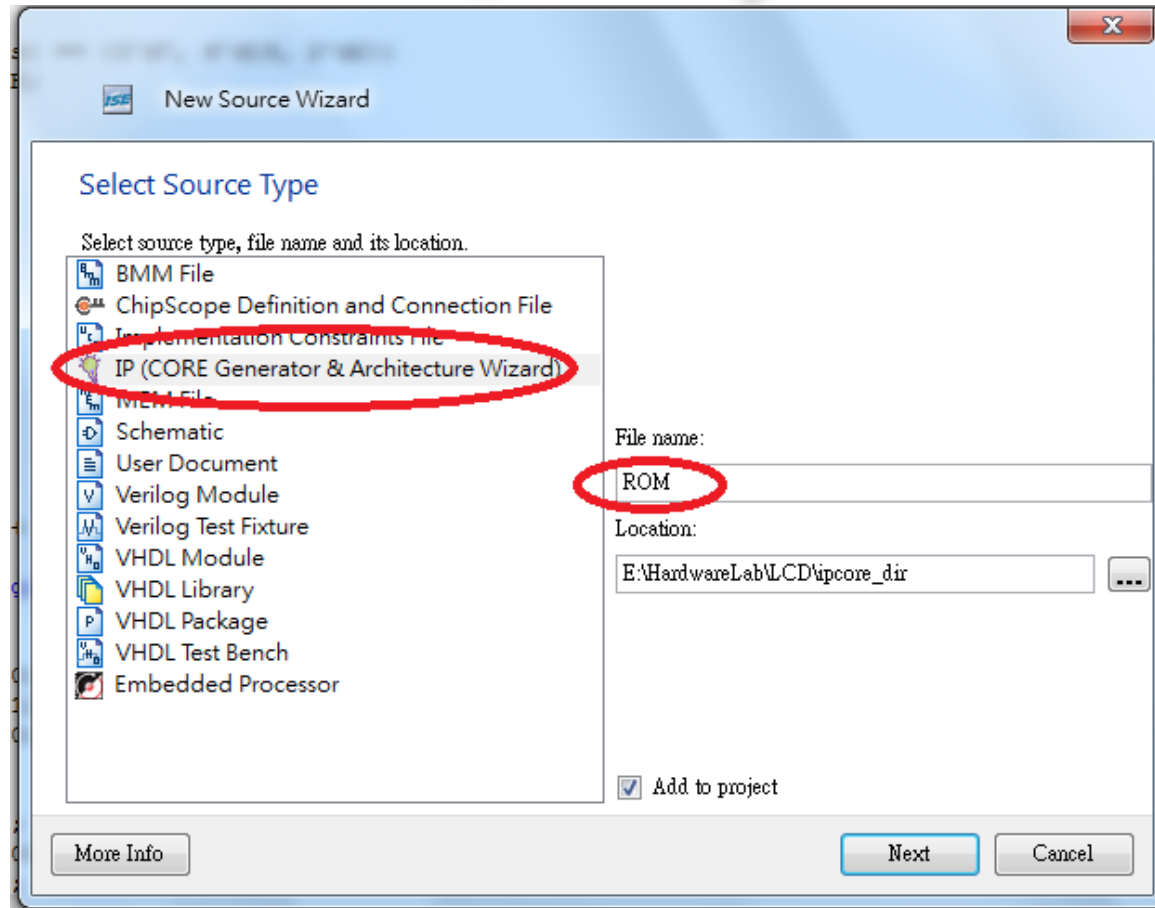
Generate ROM (1/6)

- New Source



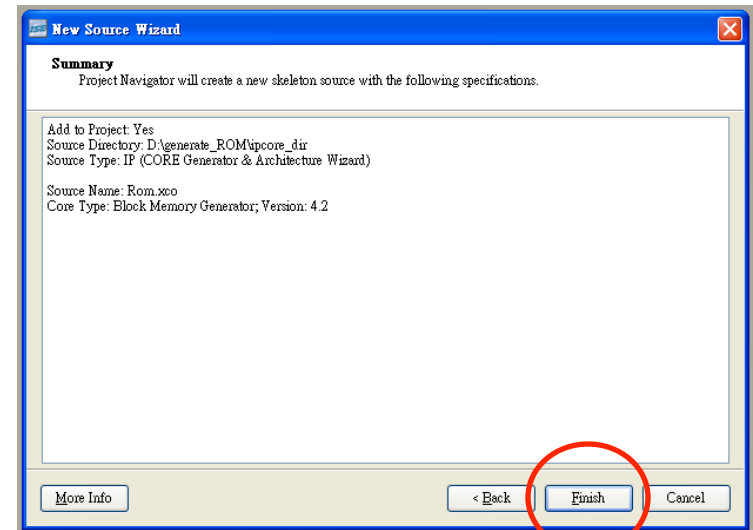
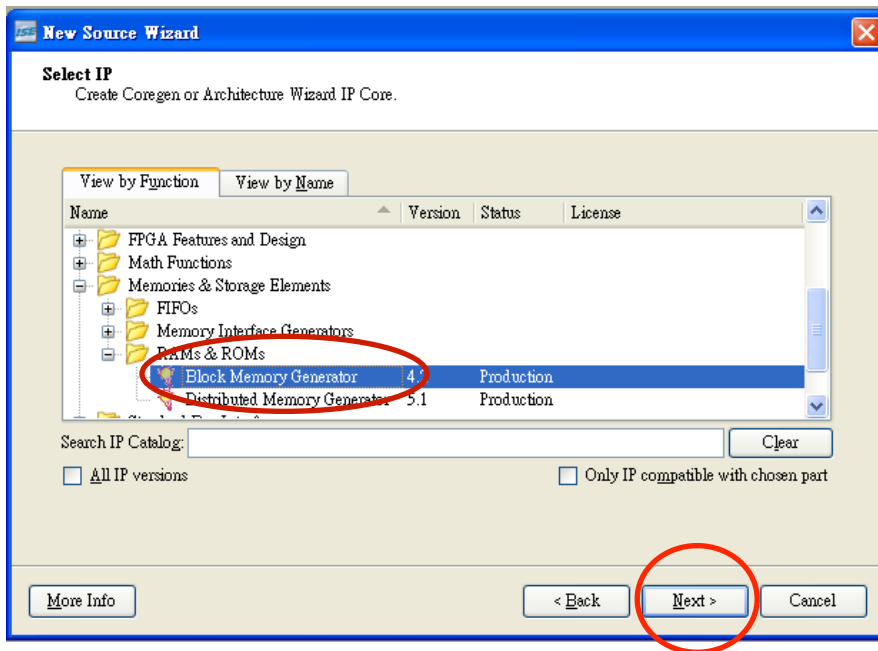
Generate ROM (2/6)

- Choose the source type: IP (CORE Generator & Architecture Wizard) and key in the filename



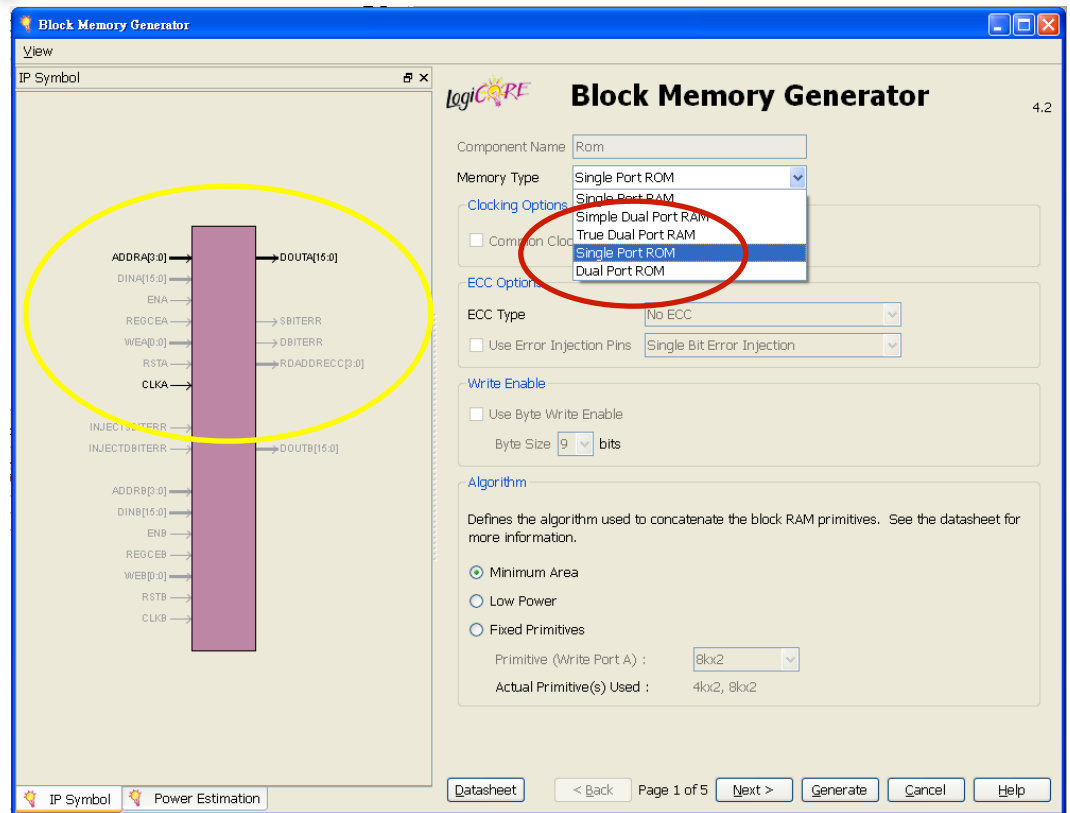
Generate ROM (3/6)

- Select:
 - Memories & Storage Elements -> RAMs/ROMs -> Block Memory Generator



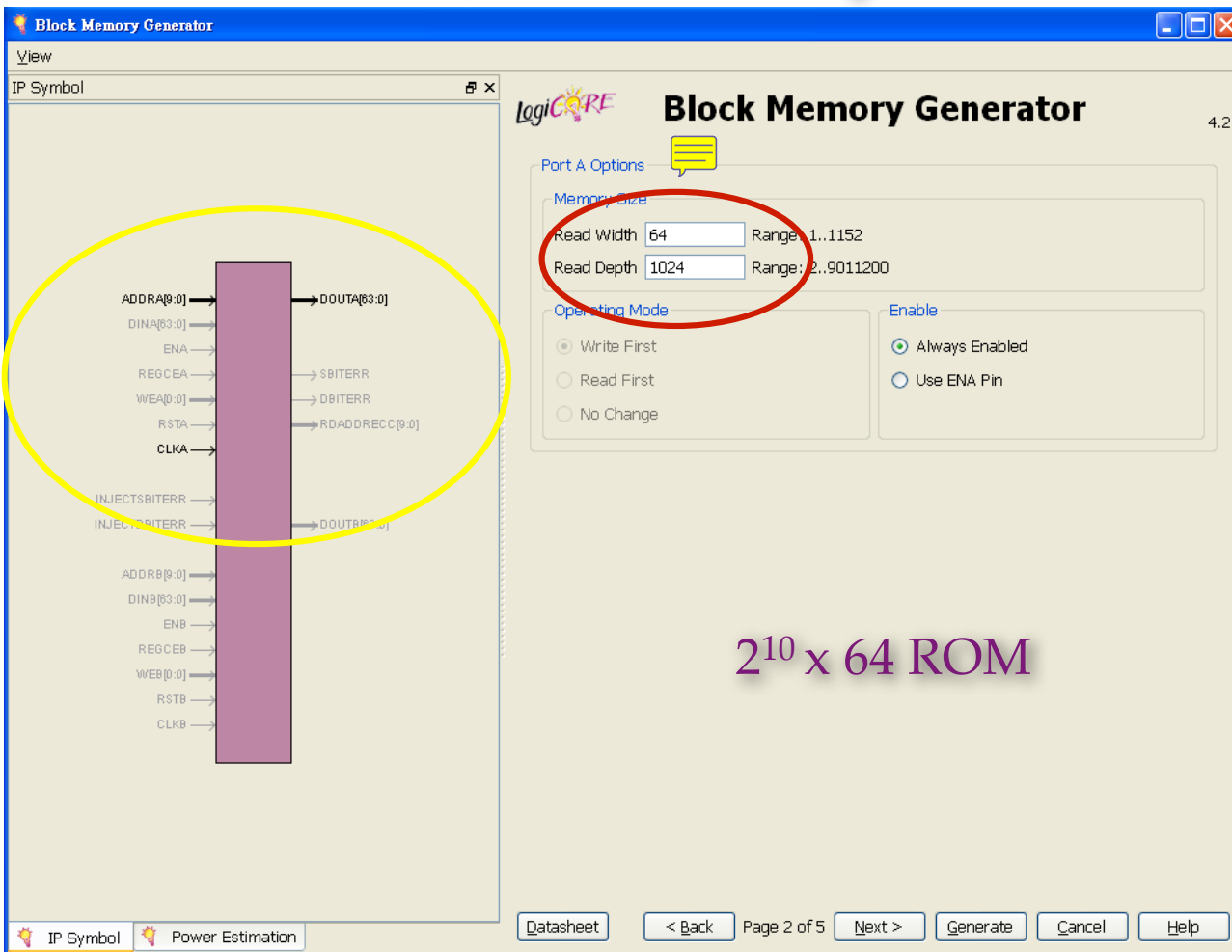
Generate ROM (4/6)

- Wait for a while
- Select Memory Type:
Single Port ROM



Generate ROM (5/6)

- Data width: 64 bits, address depth: 1024



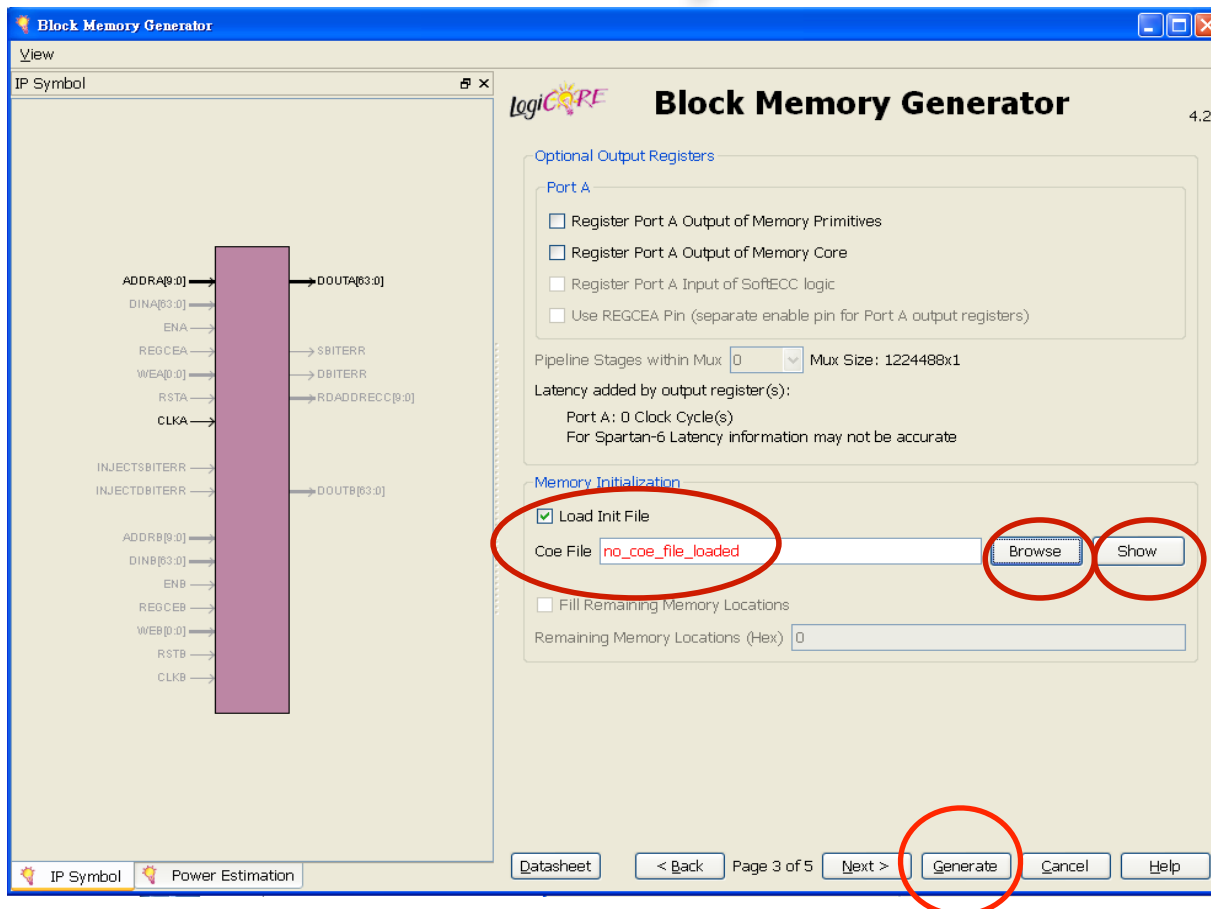
The screenshot displays the Block Memory Generator interface. On the left, a block diagram shows a vertical purple bar representing the memory. A yellow oval highlights the top portion of this bar, which is connected to various control signals: ADDR A[0:9], DINA[63:0], ENA, REG CEA, WE A[0:9], RSTA, CLKA, INJECT SBITERR, and INJECT DBITERR. On the right, the configuration panel shows the following settings:

- Memory Size:**
 - Read Width: 64 (Range: 1..1152)
 - Read Depth: 1024 (Range: 2..9011200)
- Operating Mode:**
 - Write First
 - Read First
 - No Change
- Enable:**
 - Always Enabled
 - Use ENA Pin

At the bottom right of the configuration panel, the text $2^{10} \times 64$ ROM is displayed in purple. The bottom of the window features navigation buttons: Datasheet, < Back, Page 2 of 5, Next >, Generate, Cancel, and Help.

Generate ROM (6/6)

- Check “Load Init File”
- Select “Browse” and load your COE file



Block Memory Generator

View

IP Symbol

LogiCORE **Block Memory Generator** 4.2

Optional Output Registers

Port A

Register Port A Output of Memory Primitives

Register Port A Output of Memory Core

Register Port A Input of SoftECC logic

Use REGCEA Pin (separate enable pin for Port A output registers)

Pipeline Stages within Mux: 0 Mux Size: 1224488x1

Latency added by output register(s):

Port A: 0 Clock Cycle(s)

For Spartan-6 Latency information may not be accurate

Memory Initialization

Load Init File

Coe File: no_coe_file_loaded **Browse** **Show**

Fill Remaining Memory Locations

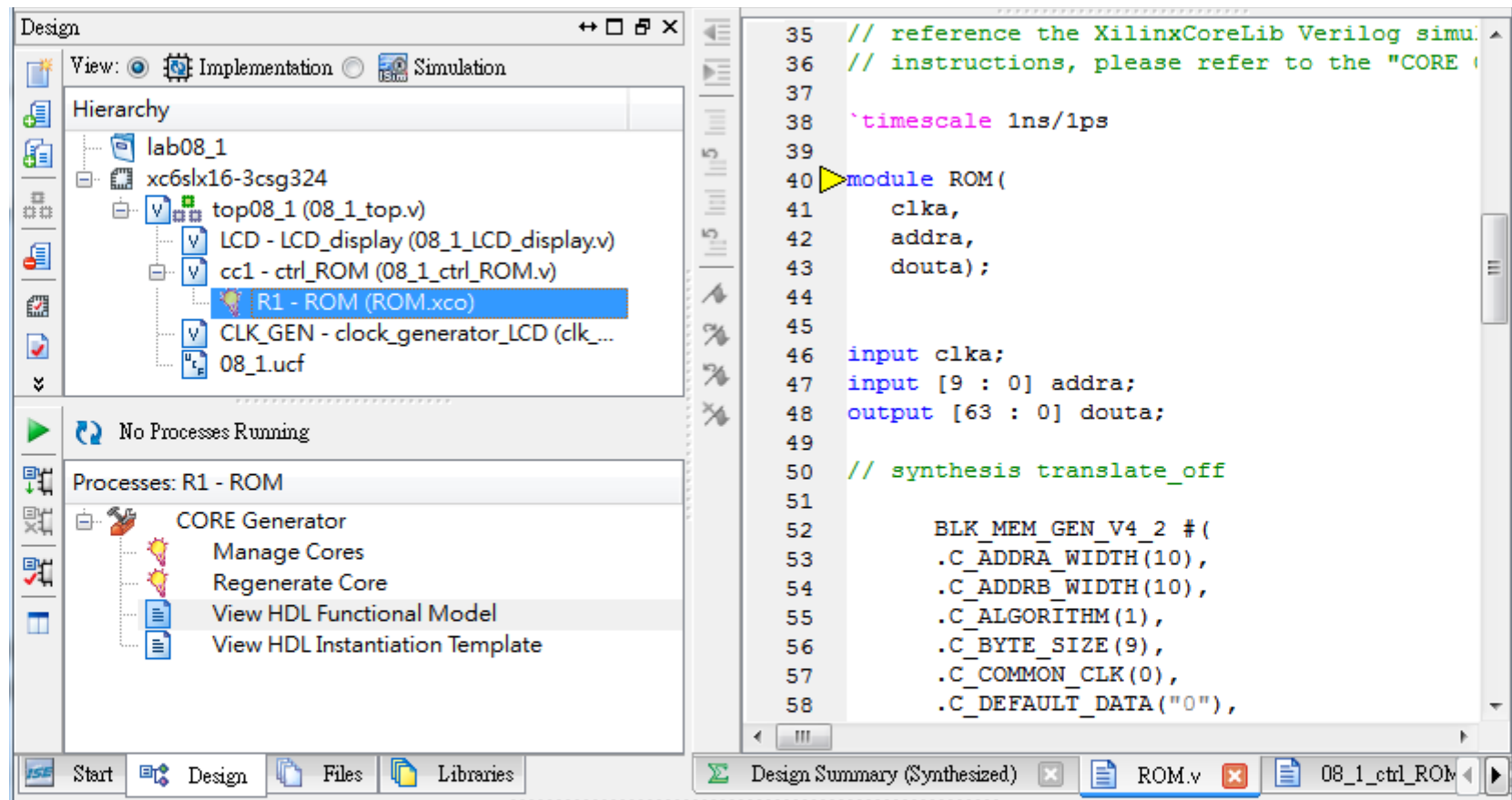
Remaining Memory Locations (Hex): 0

IP Symbol Power Estimation

Datasheet < Back Page 3 of 5 Next > **Generate** Cancel Help

How to Use ROM Module

- You can find the port names through the functional model



The screenshot shows the Xilinx ISE IDE interface. On the left, the 'Design' window displays a hierarchy tree where the 'R1 - ROM (ROM.xco)' component is selected. Below the hierarchy, the 'Processes: R1 - ROM' section shows options like 'View HDL Functional Model' and 'View HDL Instantiation Template'. On the right, the Verilog code for the ROM module is displayed, showing the module definition with ports 'clk_a', 'addr_a', and 'dout_a'.

```

35 // reference the XilinxCoreLib Verilog simul
36 // instructions, please refer to the "CORE (
37
38 `timescale 1ns/1ps
39
40 module ROM(
41     clk_a,
42     addr_a,
43     dout_a);
44
45
46 input clk_a;
47 input [9 : 0] addr_a;
48 output [63 : 0] dout_a;
49
50 // synthesis translate_off
51
52     BLK_MEM_GEN_V4_2 #(
53         .C_ADDR_A_WIDTH(10),
54         .C_ADDR_B_WIDTH(10),
55         .C_ALGORITHM(1),
56         .C_BYTE_SIZE(9),
57         .C_COMMON_CLK(0),
58         .C_DEFAULT_DATA("0"),

```

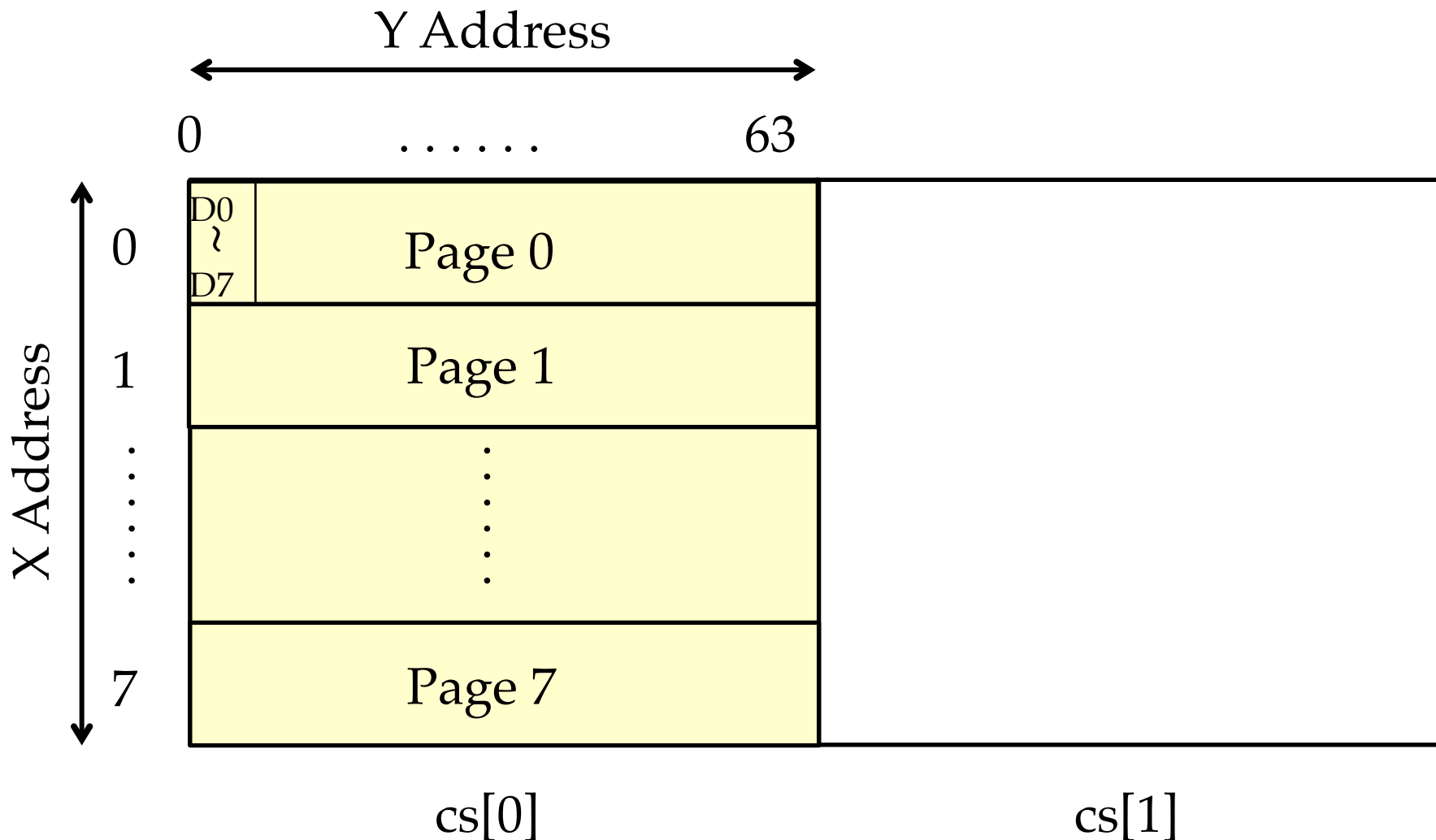
Please note you should **wait for one clock** after your issue an address and then you can get your correct data.

LCD Display

- LCD128x64

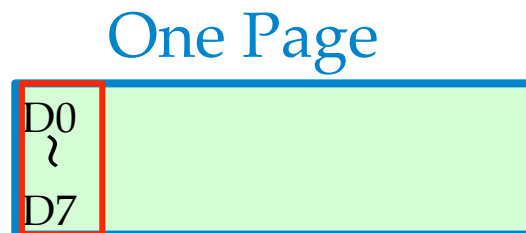
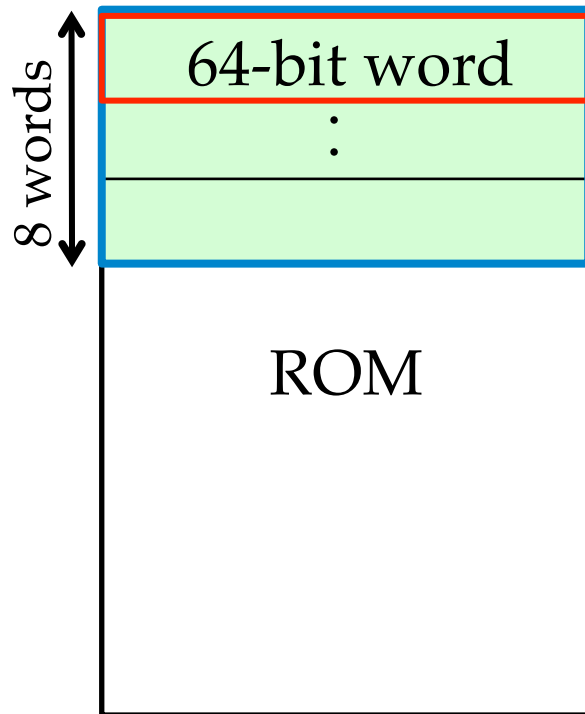
- 128 pixels in row (Y) and 64 pixels in column (X)
- Two interlace frames cs[0] and cs[1] (similar as 14SD control)
- 1 frame has 8 pages, and 1 page has 64 x 8 bits (64 bytes)
- Use instructions to control the internal state
 - Set 'Display Start Line'
 - Set 'Address' (Y)
 - Set 'Page' (X)
 - Write display data
 - Display ON/OFF
- Check the details in EVS6使用手冊 (P27,P28)

LCD Display (128x64)

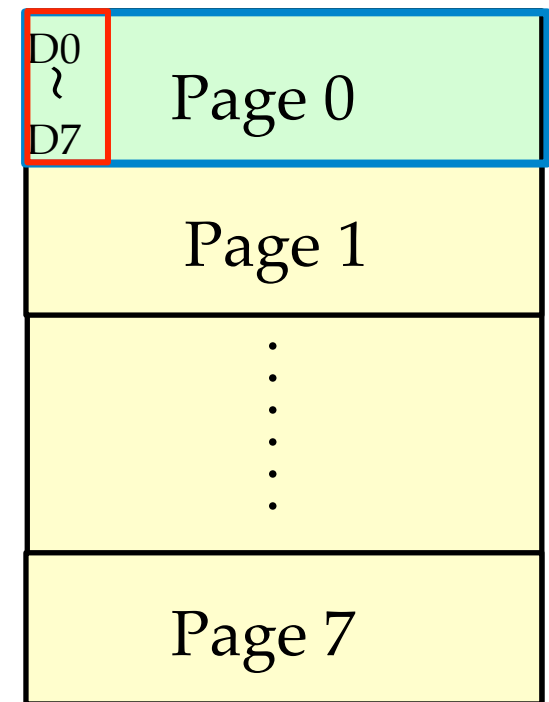


Concept of a ROM Controller

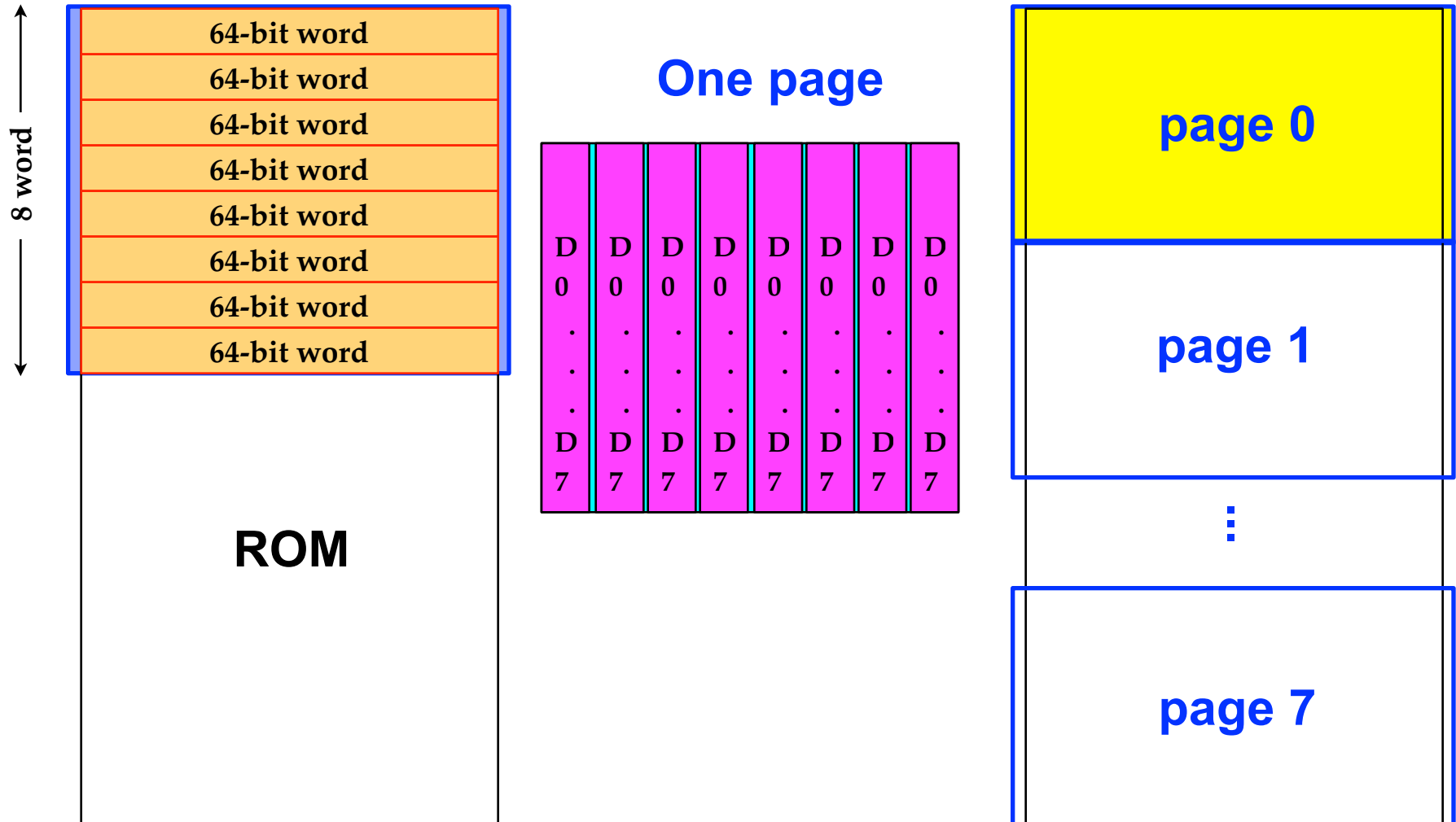
- Fetch a page one time
- Data rearrangement (words to bytes)
 - 8x64-bit (8 *words*) to 64x8-bit (64 *bytes*)



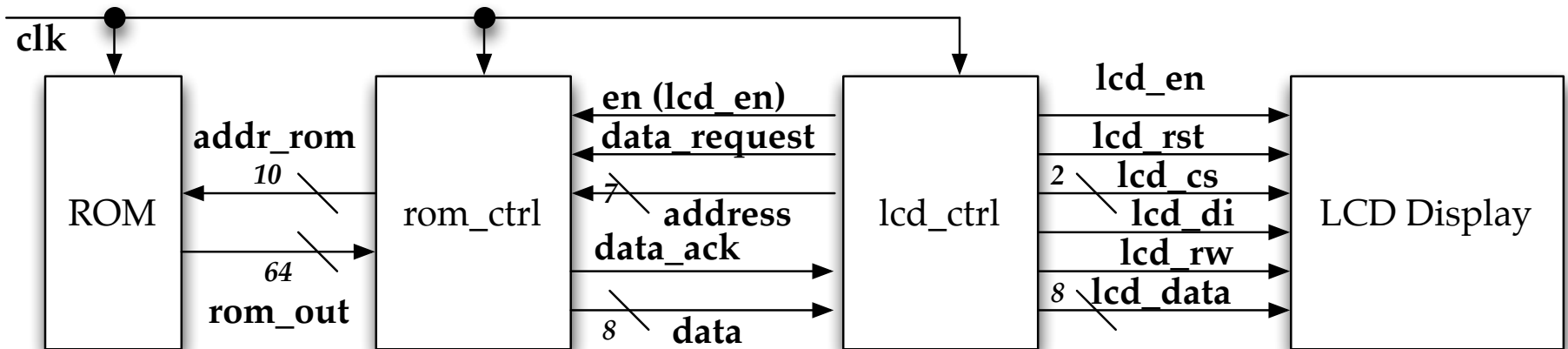
LCD Display



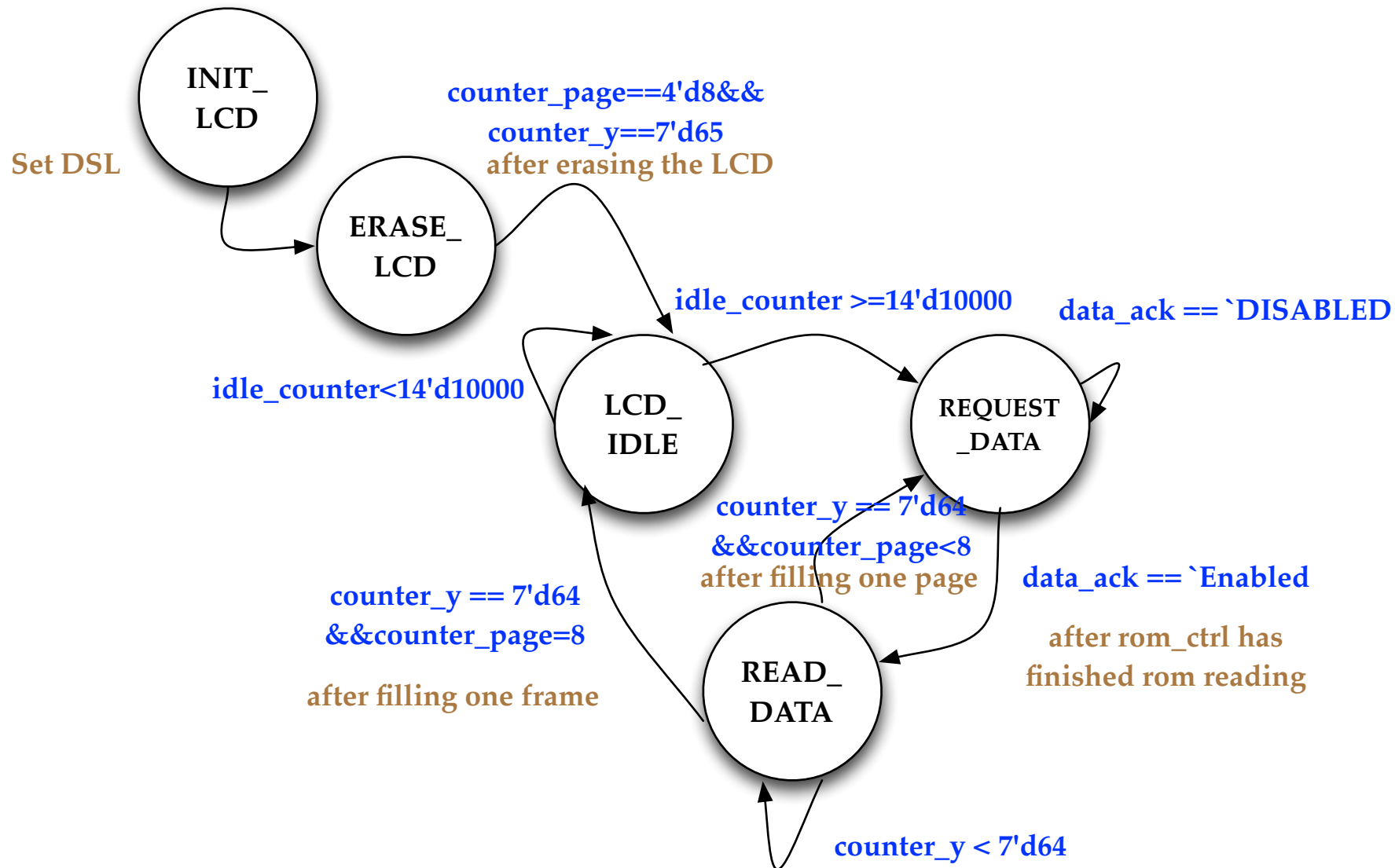
Concept of a ROM Controller



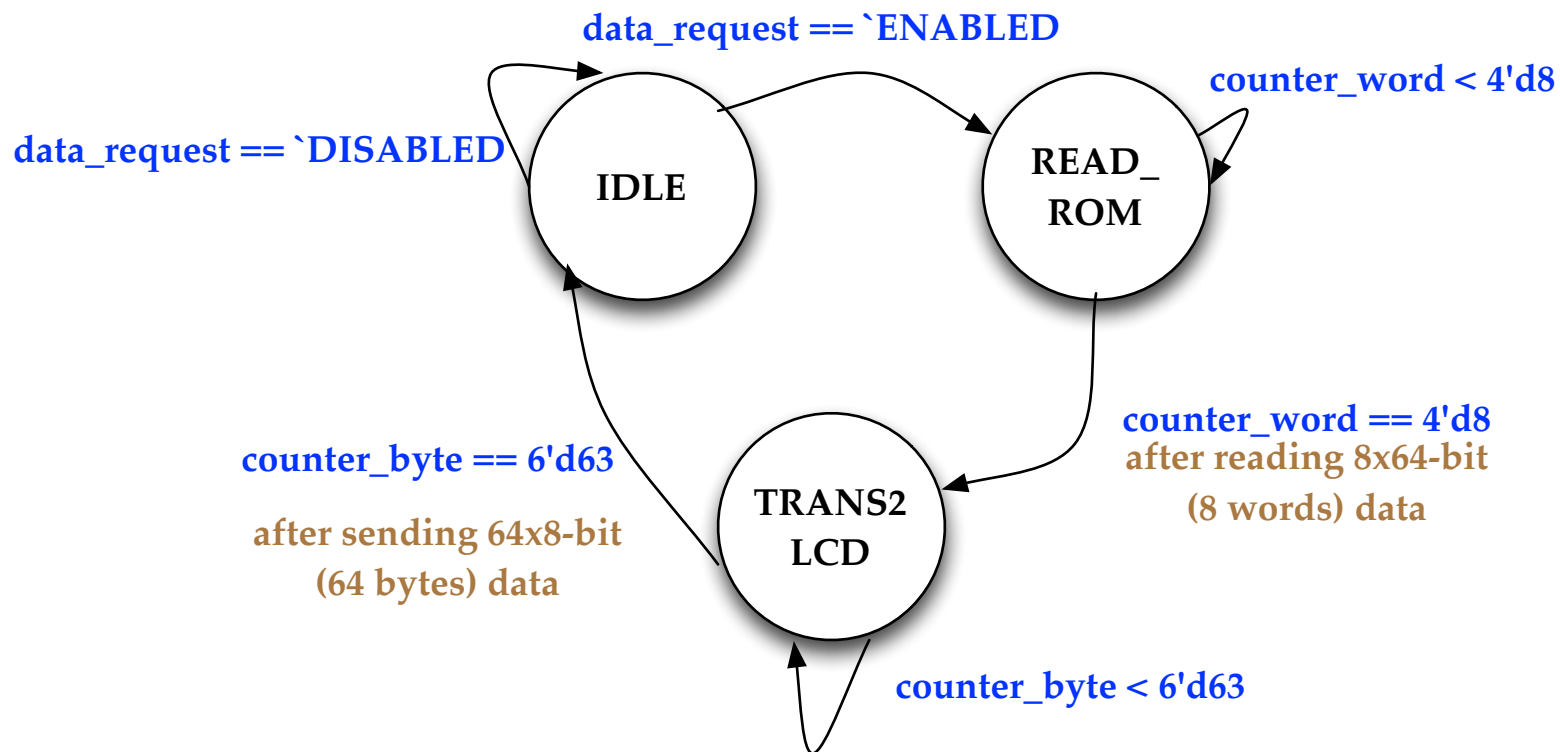
LCD Display



lcd_ctrl



rom_ctrl



Pin Assignment

LCD IO	FPGA Pin Assignment
LCD_RST	E3
LCD_CS[1]	E1
LCD_CS[0]	F4
LCD_E	F5
LCD_RW	C2
LCD_DI	C1

LCD IO	FPGA Pin Assignment
LCD_D[7]	F3
LCD_D[6]	D2
LCD_D[5]	D1
LCD_D[4]	H7
LCD_D[3]	G6
LCD_D[2]	E4
LCD_D[1]	D3
LCD_D[0]	F6

Use RAM

- You can use RAM for changeable LCD display of your project
- Similar as ROM
 - The same in IP generator, except choose 'Single Port RAM'
 - Now have write
- Timing
 - Write control, address, data should be at the same clock cycle
 - Data read out from RAM is one clock cycle late than the address control

Indexed Vector Part Selects

- Verilog-2001 adds the capability to use variables to select a group of bits from a vector
 - The starting point of the part-select can vary
 - The width of the part-select remains constant

```
mem_next[((counter_word-1)*64) +: 64] = rom_out;
```

The starting point of the part-select is variable

The width of the part-select is constant

- +:** Indicates the part-select increases from the starting point
- :** Indicates the part-select decreases from the starting point

Indexed Vector Part Selects

```
mem_next[((counter_word-1)*64)+:64] = rom_out;
```

counter_word	address range
1	0-63
2	64-127
3	128-191
4	192-255
5	256-319
6	320-383
7	384-447
8	448-511

Bad Coding Style:

Inferred Latches in Combinational Circuits

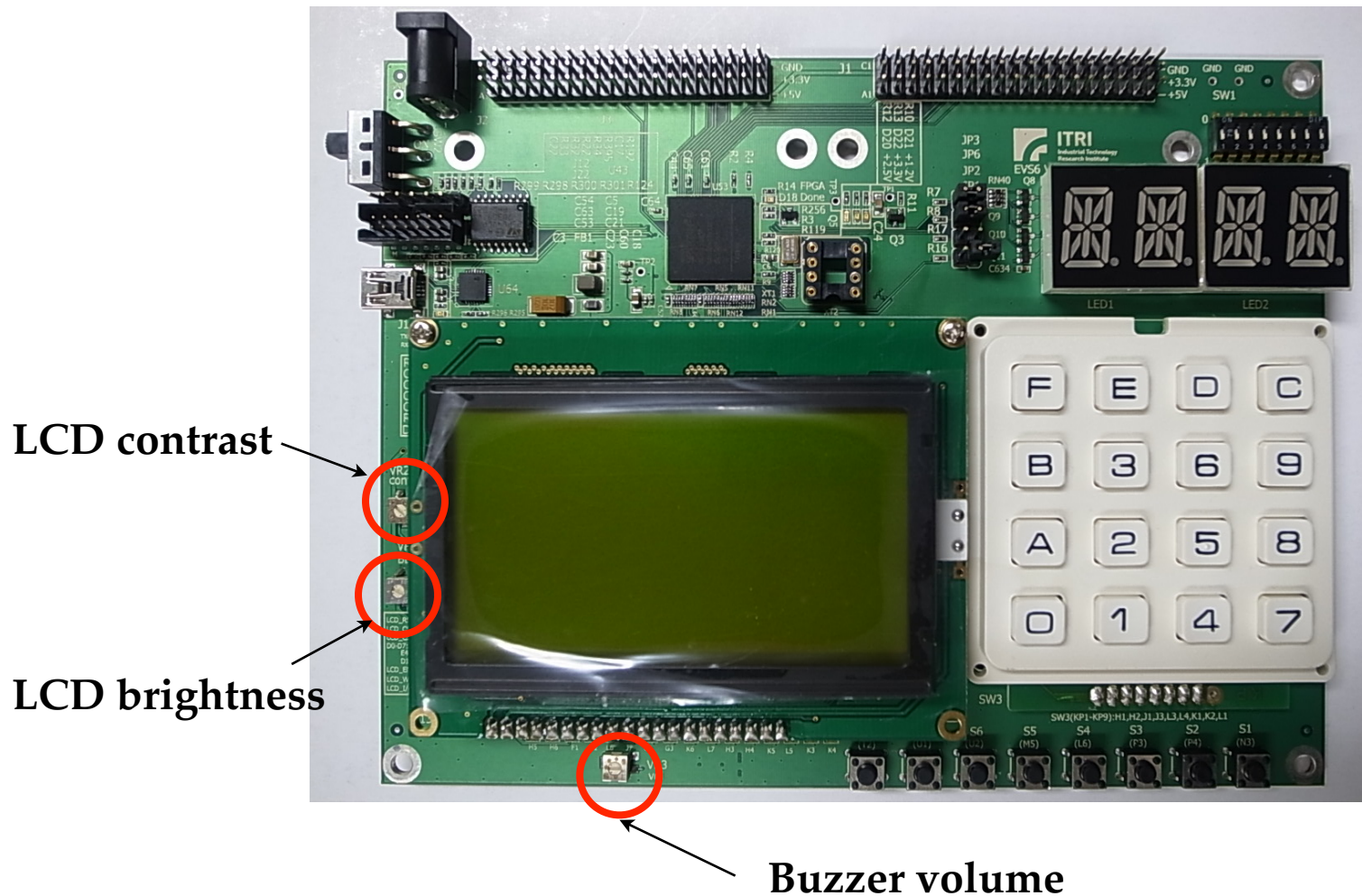
- Incomplete case statement

- Make sure to have *default* case
- Or always specify the default value in the beginning of the always block

```
always @*  
begin  
  y=0;  
  case (alu_control)  
    2'd0: y = x + z;  
    2'd1: y = x - z;  
    2'd2: y = x * z;  
    default: y = 0;  
  endcase  
end
```

```
always @(state or in)  
begin  
  next_state = `INIT_STATE;  
  case (state)  
    S0: if (in) next_state = S1;  
    S1: if ...  
    ...  
  endcase  
end
```

Device Adjustment



EVS6 使用手冊

Display Control

Pin No.	Symbol	Level	Description
1	V _{SS}	0V	Ground
2	V _{DD}	3.0V	Supply voltage for logic
3	V _O	(Variable)	Operating voltage for LCD
4	D/I	H/L	H: Data , L : Instruction
5	R/W	H/L	H: Read (MPU←Module) , L: Write (MPU→Module)
6	E	H	Enable signal
7	DB0	H/L	Data bit 0
8	DB1	H/L	Data bit 1
9	DB2	H/L	Data bit 2
10	DB3	H/L	Data bit 3
11	DB4	H/L	Data bit 4
12	DB5	H/L	Data bit 5
13	DB6	H/L	Data bit 6
14	DB7	H/L	Data bit 7
15	CS1	H	Select Column 1~ Column 64
16	CS2	H	Select Column 65~ Column 128
17	RST	L	Reset signal
18	V _{out}	-10V	Negative Voltage
19	A	—	Power Supply for LED backlight (+)
20	K	—	Power Supply for LED backlight (-)

Display Control Instruction

Instruction	D/I	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Function	
Display ON/OFF	0	0	0	0	1	1	1	1	1	0/1	Controls the display on or off. Internal status and display RAM data are not affected. 0:OFF, 1:ON	
Set Address	0	0	0	1	Y address (0~63)						Sets the Y address in the Y address counter.	
Set Page (X address)	0	0	1	0	1	1	1	Page (0 ~7)			Sets the X address at the X address register.	
Display Start Line	0	0	1	1	Display start line(0~63)						Indicates the display data RAM displayed at the top of the screen.	
Status Read	0	1	BUS Y	0	ON/ OFF	RES ET	0	0	0	0	Read status. BUSY 0:Ready, 1:In operation. ON/OFF 0:Display ON, 1:Display OFF. RESET 0:Normal, 1:Reset.	
Write Display Data	1	0	Display Data									Writes data (DB0:7) into display data RAM. After writing instruction, Y address is increased by 1 automatically.
Read Display Data	1	1	Display Data									Reads data (DB0:7) from display data RAM to the data bus.