Homework Ⅲ 說明

Instructor：Lih-Yih Chiou

TA：Mick Chang

Date：2022/11/2

EE

LIPLAB VLSI Design LAB

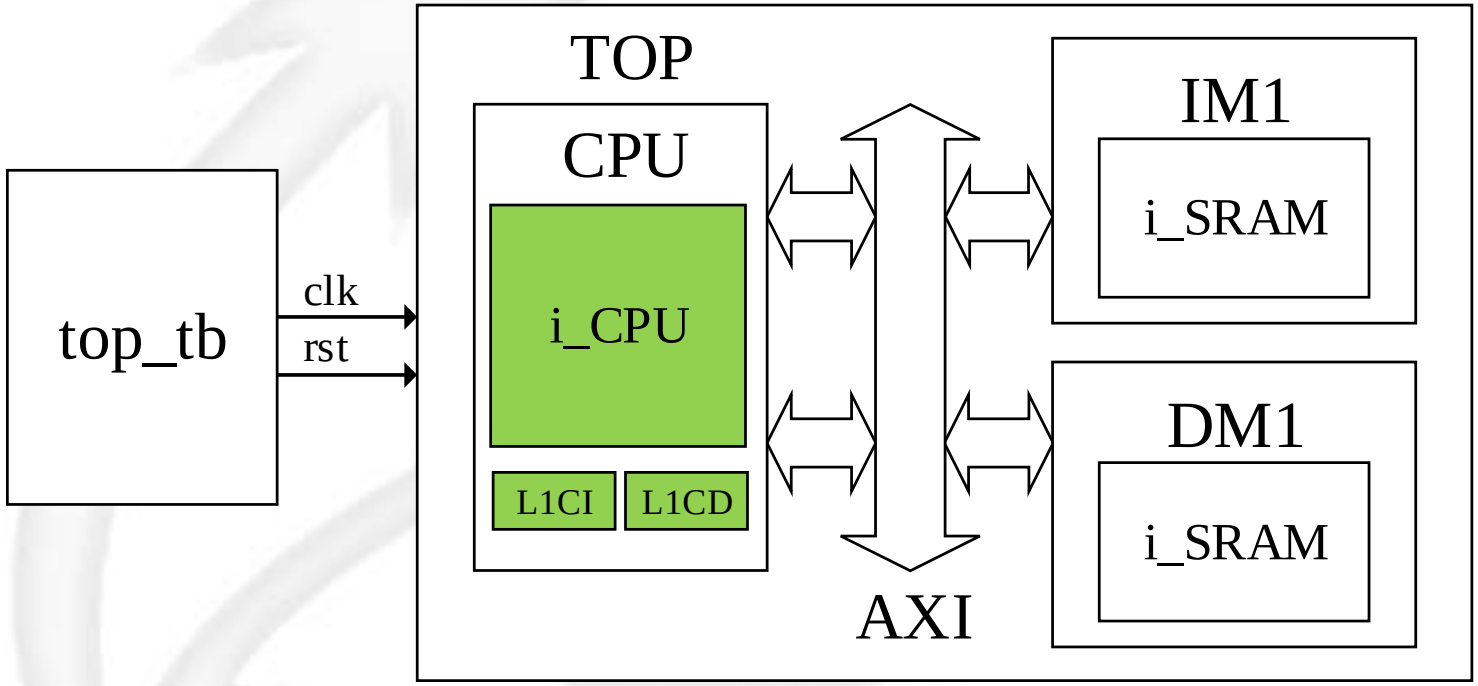
Outline

- Problem 1 - cache
  - Specification
  - Verification
- Problem 2 - booting
  - Specification
  - Verification
- Submission rule

Problem 1 - cache

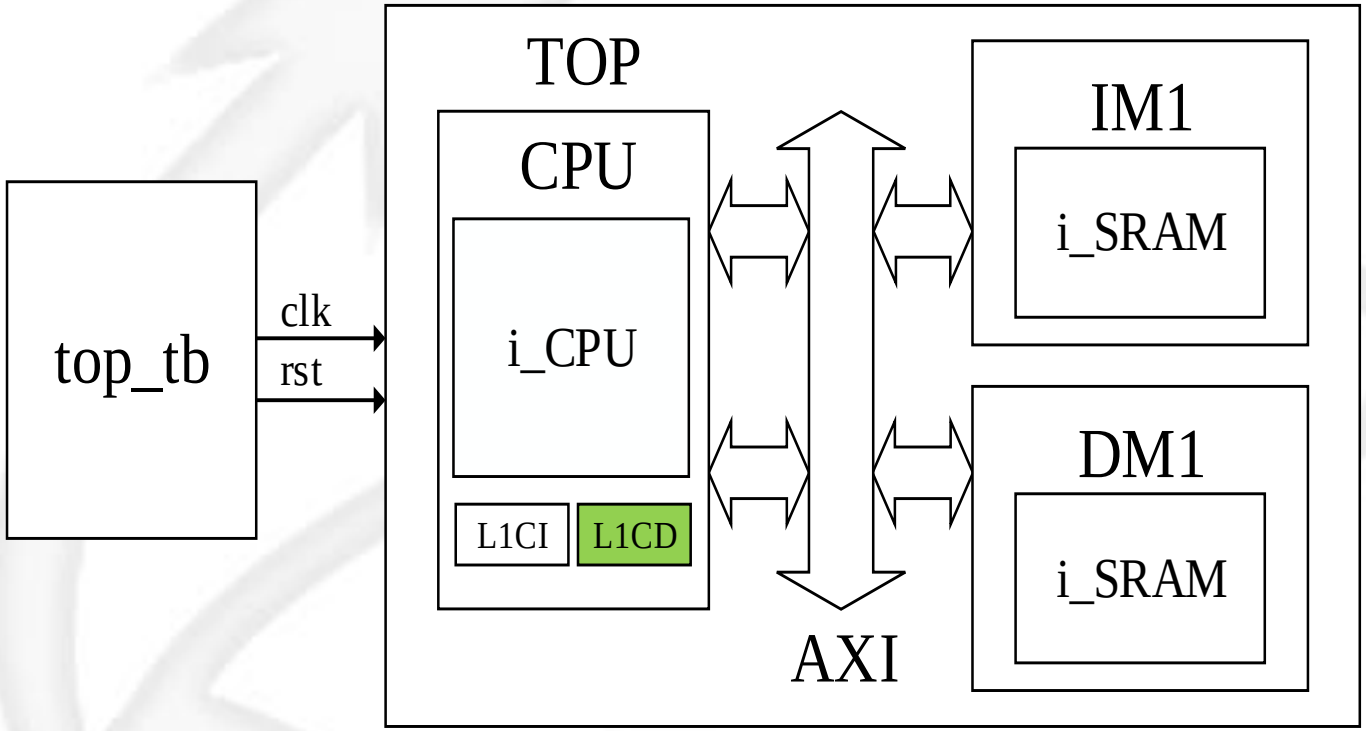
Specification

System Architecture



VLSI Design LAB

Specification(1/7)



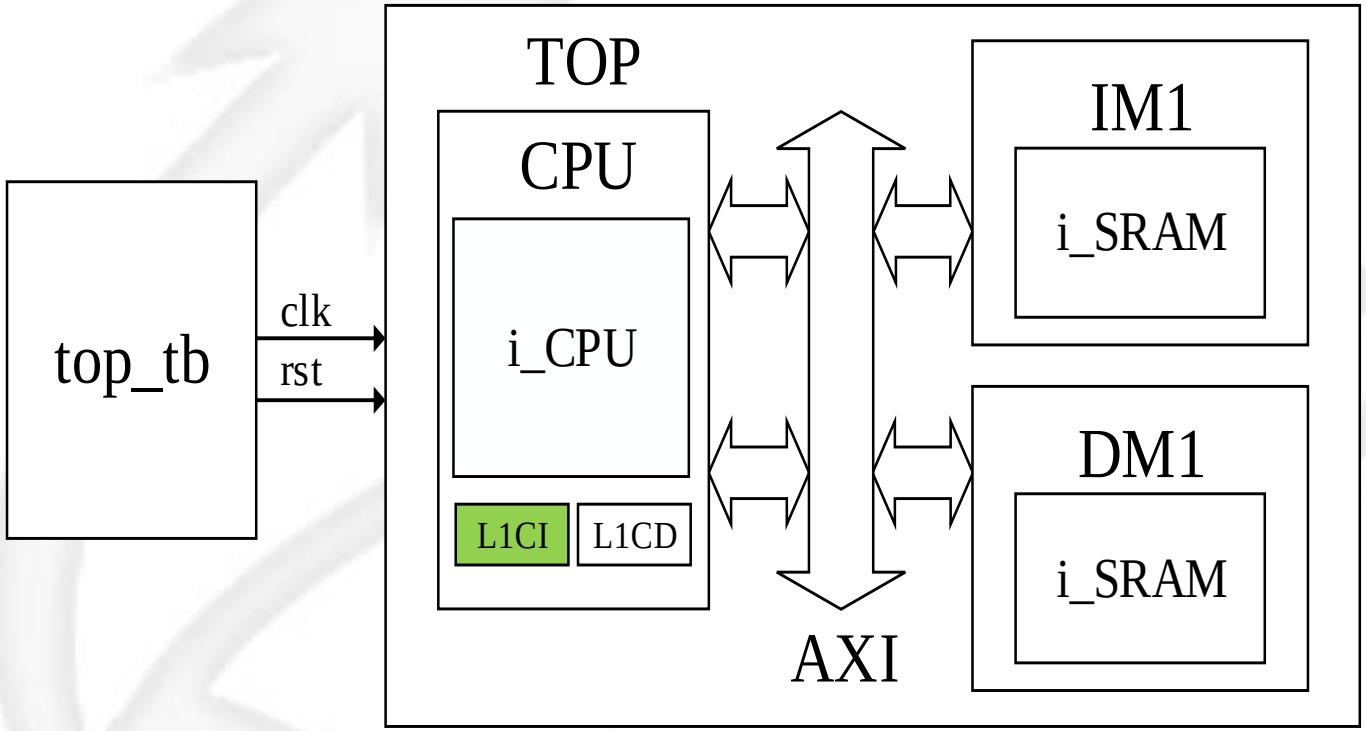
Specification(2/7)

L1C_data	clk	input	1	clock
	rst	input	1	reset (active high)
	core_addr	input	32	address from CPU
	core_req	input	1	memory access request from CPU
	core_write	input	1	write signal from CPU
	core_in	input	32	data from CPU
	core_type	input	3	write/read byte, half word, or word (listed in include/def.svh) from CPU
	D_out	input	32	data from CPU wrapper
	D_wait	input	1	wait signal from CPU wrapper
	core_out	output	32	data to CPU
	core_wait	output	1	wait signal to CPU
	D_req	output	1	request to CPU wrapper
	D_addr	output	32	address to CPU wrapper
	D_write	output	1	write signal to CPU wrapper
	D_in	output	32	write data to CPU wrapper
	D_type	output	3	write/read byte, half word, or word (listed in include/def.svh) to CPU wrapper
	valid	logic	64	valid bits of each cache line
	index	logic	6	address to tag array/data array
	TA_write	logic	1	write signal to tag_array
	TA_read	logic	1	read signal to tag_array
	TA_in	logic	22	write data to tag_array
	TA_out	logic	22	read data from tag_array
	DA_write	logic	16	write signal to data_array
	DA_read	logic	1	read signal to data_array
	DA_in	logic	128	write data to data_array
	DA_out	logic	128	read data from data_array

Inputs and outputs of module L1C_data/L1C_inst has declared in src/L1C_data.sv and src/L1C_inst.sv.
You can modify input and output bit width if needed.

VLSI Design LAB

Specification(3/7)



Specification(4/7)

L1C_inst	clk	input	1	clock
	rst	input	1	reset (active high)
	core_addr	input	32	address from CPU
	core_req	input	1	memory access request from CPU
	core_write	input	1	write signal from CPU
	core_in	input	32	data from CPU
	core_type	input	3	write/read byte, half word, or word (listed in include/def.svh) from CPU
	I_out	input	32	data from CPU wrapper
	I_wait	input	1	wait signal from CPU wrapper
	core_out	output	32	data to CPU
	core_wait	output	1	wait signal to CPU
	I_req	output	1	request to CPU wrapper
	I_addr	output	32	address to CPU wrapper
	I_write	output	1	write signal to CPU wrapper
	I_in	output	32	write data to CPU wrapper
	I_type	output	3	write/read byte, half word, or word (listed in include/def.svh) to CPU wrapper
	valid	logic	64	valid bits of each cache line
	index	logic	6	address to tag array/data array
	TA_write	logic	1	write signal to tag_array
	TA_read	logic	1	read signal to tag_array
	TA_in	logic	22	write data to tag_array
	TA_out	logic	22	read data from tag_array
	DA_write	logic	16	write signal to data_array
	DA_read	logic	1	read signal to data_array
	DA_in	logic	128	write data to data_array
	DA_out	logic	128	read data from data_array

Inputs and outputs of module L1C_data/L1C_inst has declared in src/L1C_data.sv and src/L1C_inst.sv.
You can modify input and output bit width if needed.

VLSI Design LAB

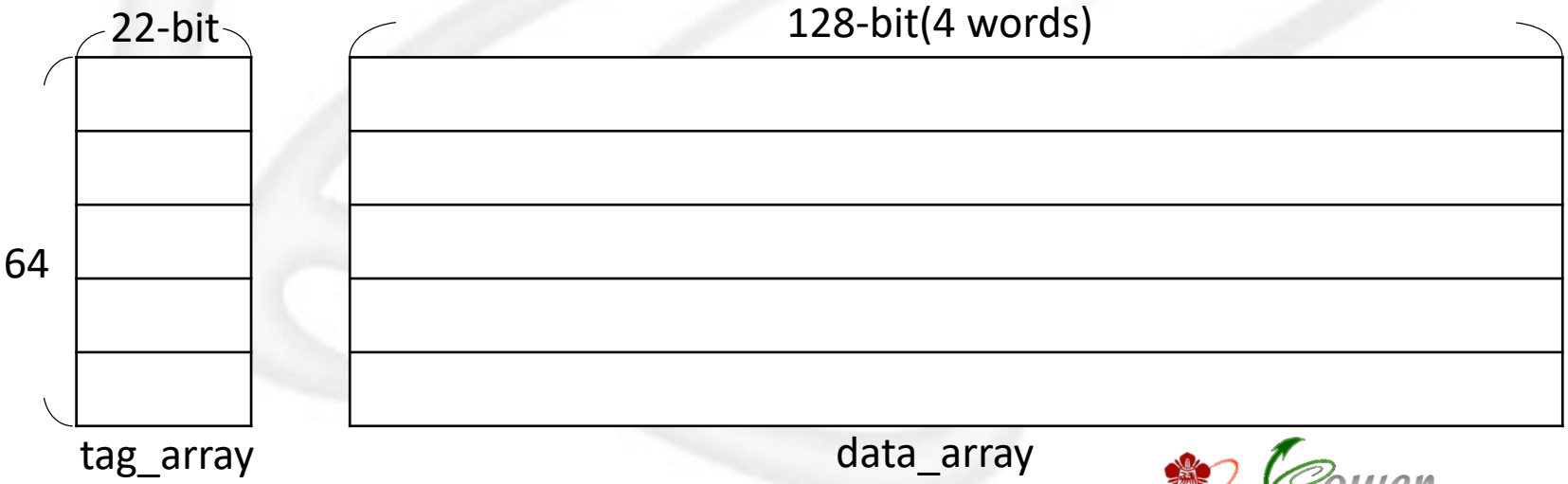
Specification(5/7)

Table 1-1: Cache actions on different condition

condition	core read	core write
hit	transmit data into core	write data into cache and memory
miss	read a line from memory	only write data into memory

Table 1-2: Array characteristics

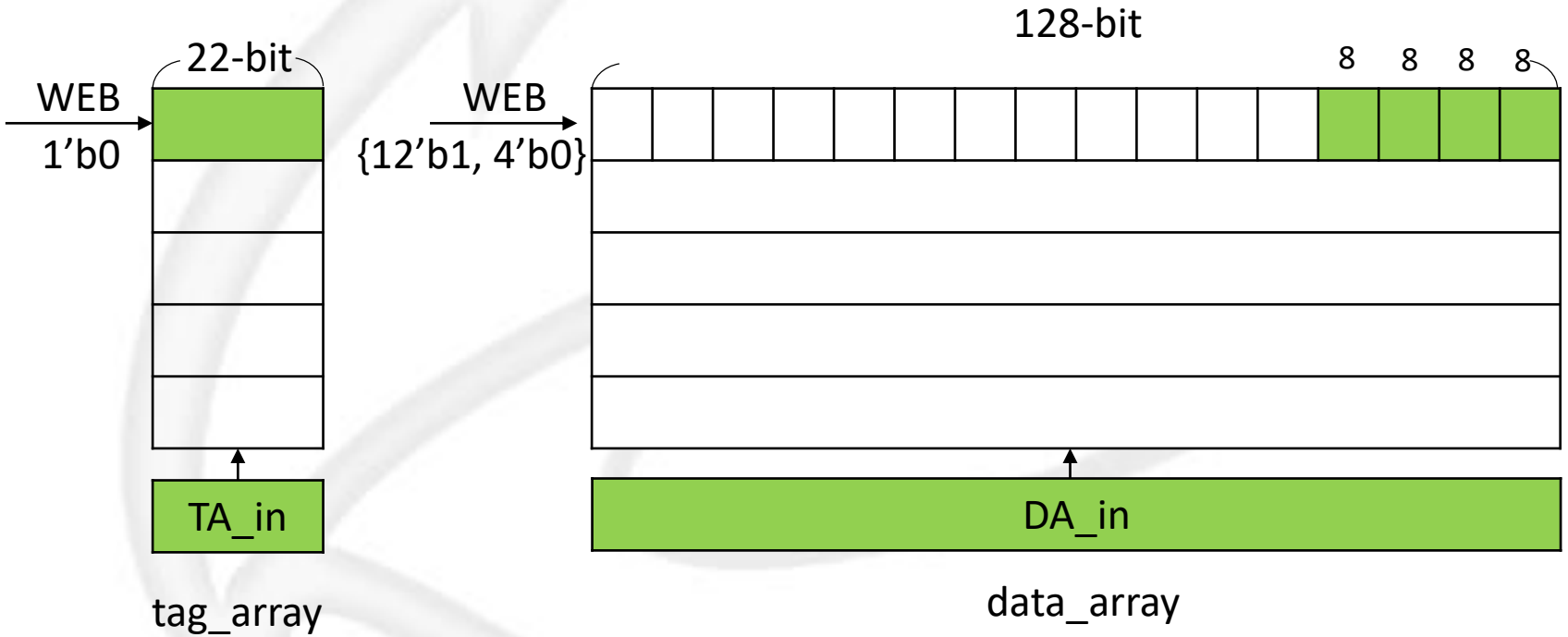
Type	Lines	Words per line	Bytes per word	Bits per line	Writing mode
data_array	64	4	4	128	Byte Write
tag_array		1		22	Word Write



Hit = valid && (TA_out equal core_addr[31:10])

Specification(6/7)

- Tag_array is word write, while data_array is byte write



VLSI Design LAB

Specification(7/7)

Table 1-2: Caches specification

Type	Size	Line Bits	Associativity	Write Policy	
Cache	1 KB	128	Direct Map	Hit	write through
				miss	write around

Table 1-3: Cache actions on different condition

condition	core read	core write
hit	transmit data into core	write data into cache and memory
miss	read a line from memory	only write data into memory

VLSI Design LAB

VLSI Design LAB

Problem 1 - cache

Verification

Report Requirements

- Proper explanation of your design is required for full credits.
- Block diagrams shall be drawn to depict your designs.
- Show the result of JasperGold with full prove, explain the assertion you modified.
- Show the hit rate of your instruction cache and data cache
- Show the IPC (instruction per cycle) of your CPU

$$\text{cache hit rate} = \frac{Number\ of\ cache\ hits}{Number\ of\ cache\ hits\ +\ Number\ of\ cache\ misses}$$

$$\text{IPC} = \frac{Number\ of\ instructions\ CPU\ execute}{Total\ cycles}$$

VLSI Design LAB

Problem 2 - booting

Specification

Slave Configuration

Table 1-5 : Slave configuration

NAME	Number	Start address	End address
ROM	Slave 0	0x0000_0000	0x0000_1FFF
IM	Slave 1	0x0001_0000	0x0001_FFFF
DM	Slave 2	0x0002_0000	0x0002_FFFF
sensor_ctrl	Slave 3	0x1000_0000	0x1000_03FF
DRAM	Slave 4	0x2000_0000	0x201F_FFFF

- You should design all slave wrappers (except slave 3) !!

VLSI Design LAB

ROM

ROM	System signals			
	CK	input	1	System clock
	Memory ports			
	DO	output	32	ROM data output
	OE	input	1	Output enable (active high)
	CS	input	1	Chip select (active high)
	A	input	12	ROM address input
	Memory Space			
	Memory_byte0	reg	8	Size: [0:4095]
	Memory_byte1	reg	8	Size: [0:4095]
	Memory_byte2	reg	8	Size: [0:4095]
	Memory_byte3	reg	8	Size: [0:4095]

VLSI Design LAB

DRAM

DRAM	System signals			
	CK	input	1	System clock
	RST	input	1	System reset (active high)
	Memory ports			
	CSn	input	1	DRAM Chip Select (active low)
	WEn	input	4	DRAM Write Enable (active low)
	RASn	input	1	DRAM Row Access Strobe (active low)
	CASn	input	1	DRAM Column Access Strobe (active low)
	A	input	11	DRAM Address input
	D	input	32	DRAM data input
	Q	output	32	DRAM data output
	VALID	output	1	DRAM data output valid
	Memory space			
	Memory_byte0	reg	8	Size: [0:2097151]
	Memory_byte1	reg	8	Size: [0:2097151]
	Memory_byte2	reg	8	Size: [0:2097151]
	Memory_byte3	reg	8	Size: [0:2097151]

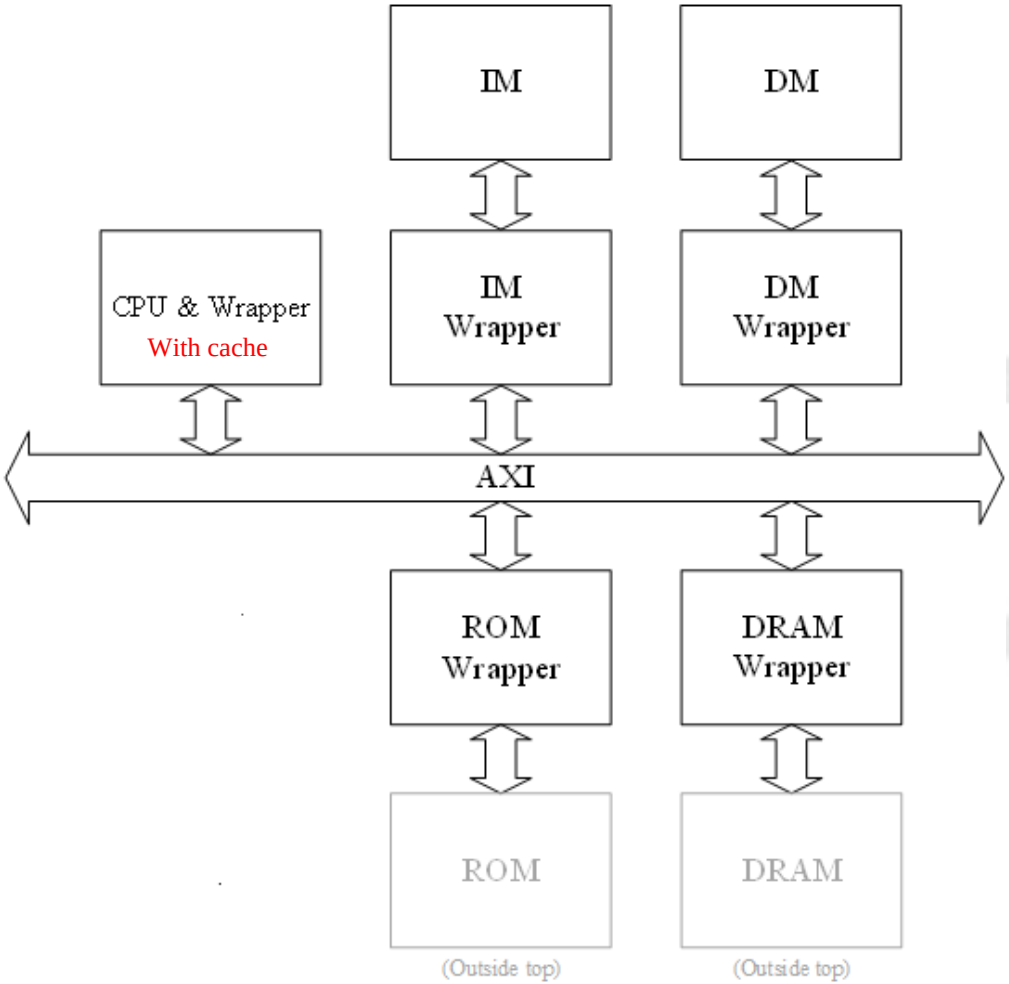
★ Row address is 11-bit and column address is 10-bit

★ Row addr = araddr/awaddr [22:12]

★ Col addr = araddr/awaddr[11:2]

VLSI Design LAB

System Architecture



VLSI Design LAB

Problem 2 - booting

Verification

Verification(1/7)

- prog0
  - 測試45個instruction (助教提供)
- prog1
  - Sort algorithm of half-word
- prog2
  - Gray scale
- prog3
  - Matrix multiplication

Verification (2/7)

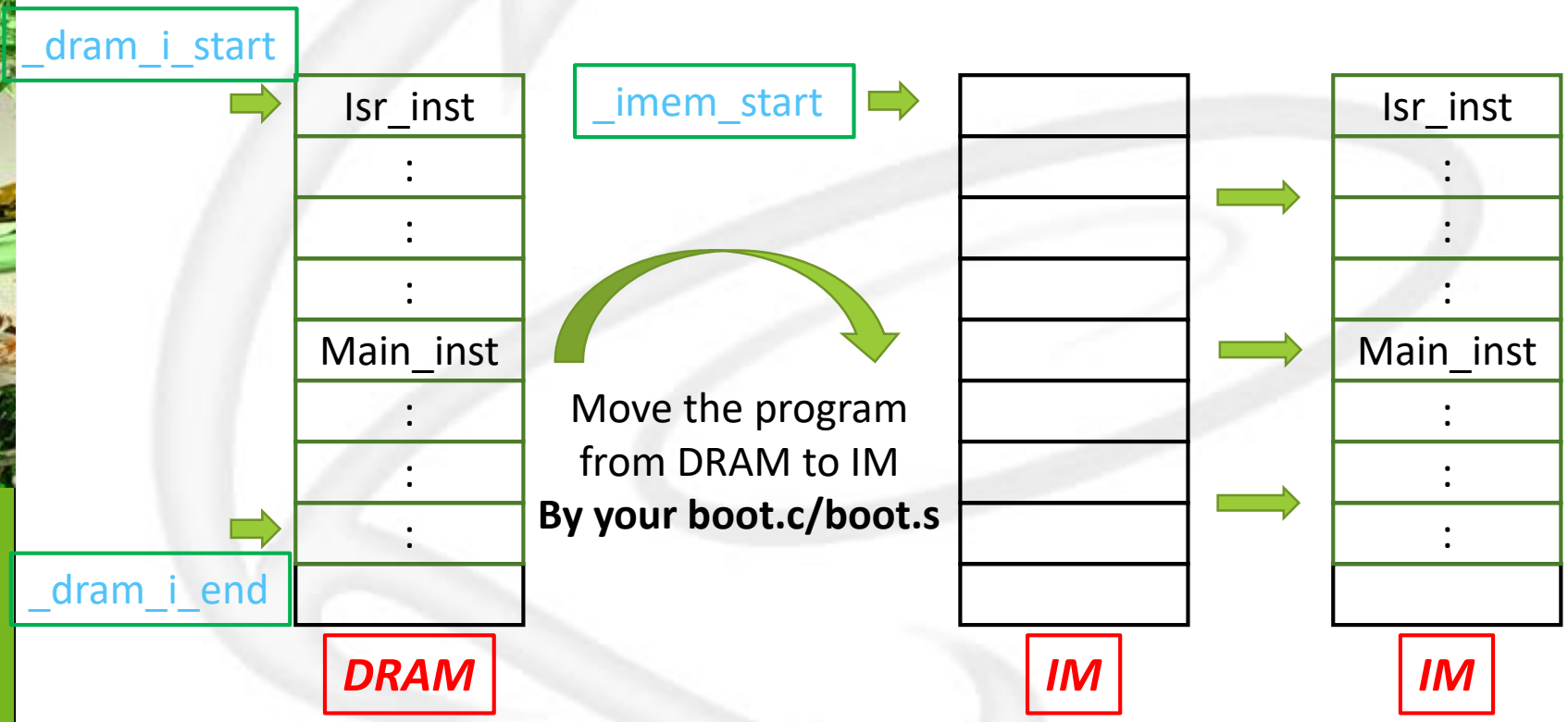
- Booting
  - The booting program is stored in ROM
  - Moves data from DRAM to IM and DM

```
extern unsigned int _dram_i_start;
extern unsigned int _dram_i_end;
extern unsigned int _imem_start;

extern unsigned int __sdata_start;
extern unsigned int __sdata_end;
extern unsigned int __sdata_paddr_start;

extern unsigned int __data_start;
extern unsigned int __data_end;
extern unsigned int __data_paddr_start;
```

VLSI Design LAB

Verification (3/7)

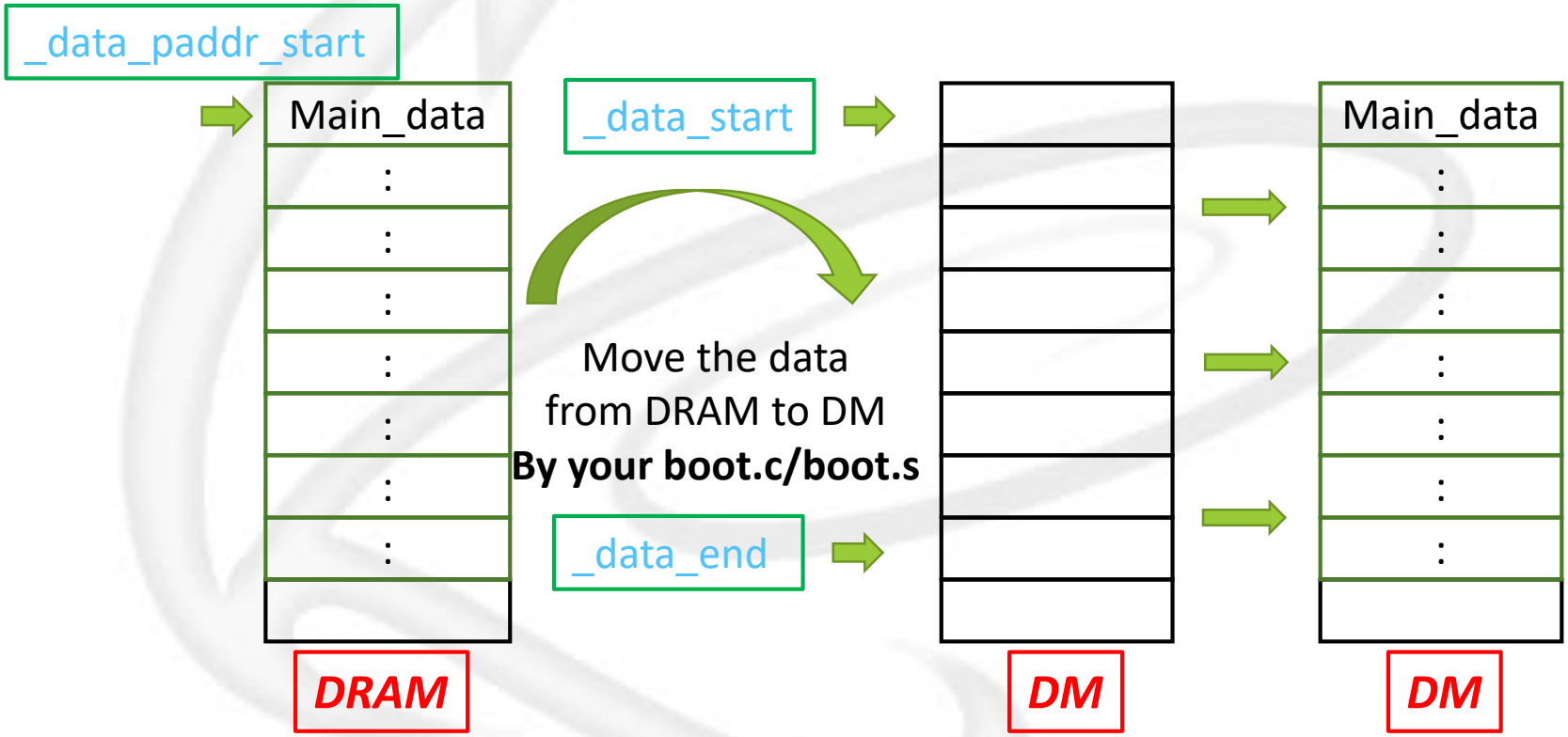
- _dram_i_start = instruction start address in DRAM.
- _dram_i_end = instruction end address in DRAM.
- _imem_start = instruction start address in IM



VLSI Design LAB

Verification (4/7)

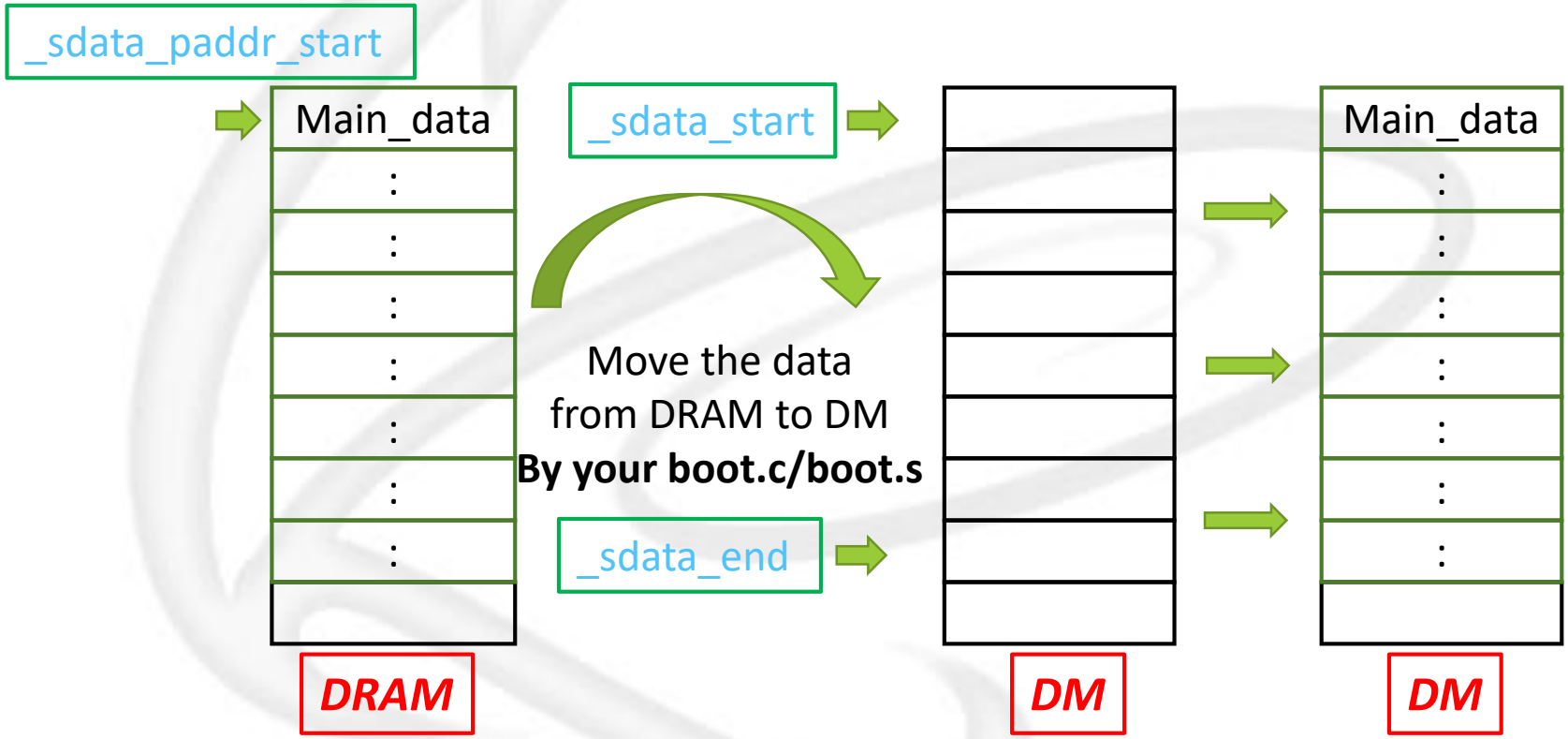
- _data_start = Main_data start address in DM.
- _data_end = Main_data end address in DM.
- _data_paddr_start = Main_data start address in DRAM



VLSI Design LAB

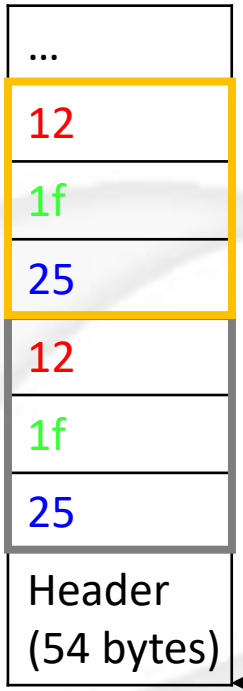
Verification (5/7)

- _sdata_start = Main_data start address in DM.
- _sdata_end = Main_data end address in DM.
- _sdata_paddr_start = Main_data start address in DRAM



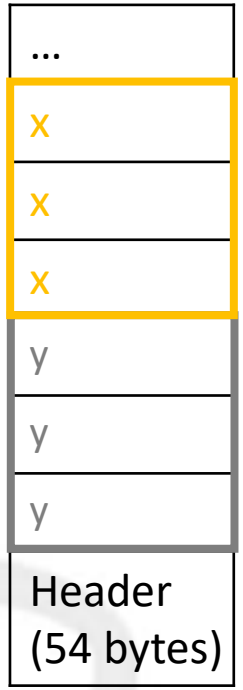
VLSI Design LAB

Verification(6/7)



x = 0.11*b + 0.59*g + 0.3*r
= 0.11*25 + 0.59*1f + 0.3* 12

y = 0.11*b + 0.59*g + 0.3*r
= 0.11*25 + 0.59*1f + 0.3* 12



B M Size of BMP file (byte)

The number of bits per pixel

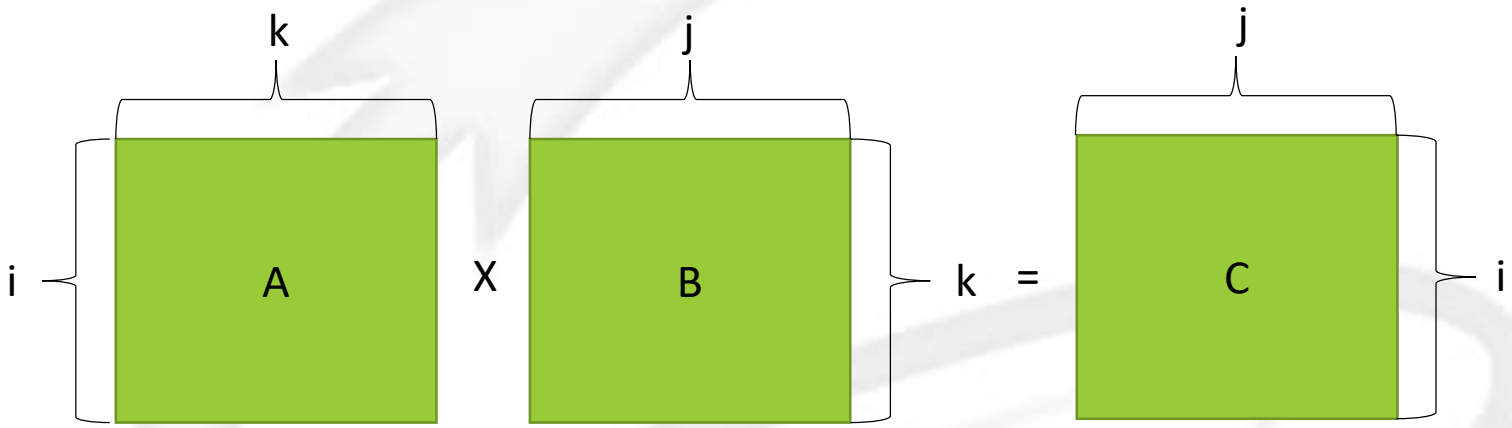
Address	0	1	2	3	4	5	6	7	8	9	a	b	Dump
00000000	42	4d	36	00	24	00	00	00	00	00	36	00	BM6.$.....6.
0000000c	00	00	28	00	00	00	00	04	00	00	00	03	..(.....
00000018	00	00	01	00	18	00	00	00	00	00	00	00	.....
00000024	24	00	c4	0e	00	00	c4	0e	00	00	00	00	$.?..?....
00000030	00	00	00	00	00	00	25	1f	12	25	1f	12	......%..%..
0000003c	25	1f	12	25	1f	12	25	1f	12	25	1f	12	%..%..%..%..
00000048	25	1f	12	25	1f	12	25	1f	12	25	1f	12	%..%..%..%..
00000054	25	1f	12	25	1f	12	25	1f	12	25	1f	12	%..%..%..%..
00000060	25	1f	12	25	1f	12	25	1f	12	25	1f	12	%..%..%..%..
0000006c	25	1f	12	25	1f	12	25	1f	12	25	1f	12	%..%..%..%..
00000078	25	1f	12	25	1f	12	25	1f	12	25	1f	12	%..%..%..%..

從檔案頭到點陣圖資料須偏移的byte數

BMP檔案解析參考資料:
https://www.itread01.com/content/1549504280.html

Verification(7/7)

- Matrix Multiplication



$$\begin{pmatrix} data\cdot 1 & data\cdot 2 & data\cdot 3 \\ data\cdot 4 & data\cdot 5 & data\cdot 6 \end{pmatrix} \cdot \begin{pmatrix} data\cdot 7 & data\cdot 8 & data\cdot 9 & data\cdot 10 \\ data\cdot 11 & data\cdot 12 & data\cdot 13 & data\cdot 14 \\ data\cdot 15 & data\cdot 16 & data\cdot 17 & data\cdot 18 \end{pmatrix} = \begin{pmatrix} result\cdot 1 & result\cdot 2 & result\cdot 3 & result\cdot 4 \\ result\cdot 5 & result\cdot 6 & result\cdot 7 & result\cdot 8 \end{pmatrix}$$

VLSI Design LAB

Report Requirements

- Proper explanation of your design is required for full credits.
- Block diagrams shall be drawn to depict your designs.
- Show your screenshots of the waveforms and the simulation results on the terminal for the different test cases in your report and illustrate the correctness of your results. Explain cache read hit/read miss/write hit/write miss with waveform.
- Explain your codes of program1, program2 and program3.
- Explain your codes of boot.c.
- Report the number of lines of your RTL code, the final results of running Superlint and 3~5 most frequent warning/errors in your code. Describe how you modify your code to comply with Superlint.

Report Requirements

- Report and show screenshots of your prog0 to prog3 simulation time after synthesis and total cell area of your design.

VLSI Design LAB

Submission rule

Report

- 請使用附在檔案內的Submission Cover
- 請勿將code貼在.docx內 (program的程式可截圖說明)
  - 請將.sv包在壓縮檔內，不可截圖於.docx中
- 需要Summary及Lessons learned(Summary table請放在第二頁，清楚列出有完成以及沒完成的部分)
- 若兩人為一組，須寫出貢獻度(貢獻度請放第二頁)
  - Ex: A(N26071234) 55%, B(N26075678) 45%
  - Total 100%
  - 自己一組則不用寫

Specification

- Module name須符合下表要求

Category	Name		
	File	Module	Instance
RTL	top.sv	top	TOP
Gate-Level	top_syn.v	top	TOP
RTL	L1C_inst.sv	L1C_inst	L1CI
RTL	L1C_data.sv	L1C_data	L1CD
RTL	SRAM_wrapper.sv	SRAM_wrapper	IM1
RTL	SRAM_wrapper.sv	SRAM_wrapper	DM1
RTL	SRAM_rtl.sv	SRAM	i_SRAM
RTL	tag_array_wrapper.sv	tag_array_wrapper	TA
RTL	tag_array_rtl.sv	tag_array	i_tag_array
RTL	data_array_wrapper.sv	data_array_wrapper	DA
RTL	data_array_rtl.sv	data_array	i_data_array
Behavior	ROM.v	ROM	i_ROM
Behavior	DRAM.v	DRAM	i_DRAM

- 需按照要求命名，以免testbench抓不到正確的名稱

繳交檔案 (1/2)

- 依照檔案結構壓縮成 “.tar” 格式
  - ➔ 在Homework主資料夾(N260XXXXX)使用make tar產生的tar檔即可符合要求
- 檔案結構請依照作業說明
- 請勿附上檔案結構內未要求繳交的檔案
  - ➔ 在Homework主資料夾(N260XXXXX)使用make clean即可刪除不必要的檔案
- 請務必確認繳交檔案可以在SoC實驗室的工作站下compile，且功能正常
- 無法compile將直接以0分計算
- 請勿使用generator產生code再修改
- 禁止抄襲

繳交檔案 (2/2)

- 一組只需一個人上傳作業到Moodle
  - 兩人以上都上傳會斟酌扣分
- 壓縮檔、主資料夾名稱、Report名稱、StudentID檔案內的學號都要為上傳者的學號，其他人則在Submission Cover內寫上自己的學號。
  - Ex: A(N26101234)負責上傳，組員為B(N26105678)
  - N26101234.tar (壓縮檔)
    N26101234 (主資料夾)
    N26101234.docx (Report，Cover寫上兩者的學號)

繳交期限

- 2022/11/30 (三) 14:00前上傳
  - 不接受遲交，請務必注意時間
  - Moodle只會留存你最後一次上傳的檔案，檔名只要是「*N26XXXXXX.tar*」即可，不需要加上版本號

VLSI Design LAB

Thanks for your participation and attendance !!