Homework II 說明

Instructor：Lih-Yih Chiou

TA : Tsung-Chi Chen

Date : 2021/10/13

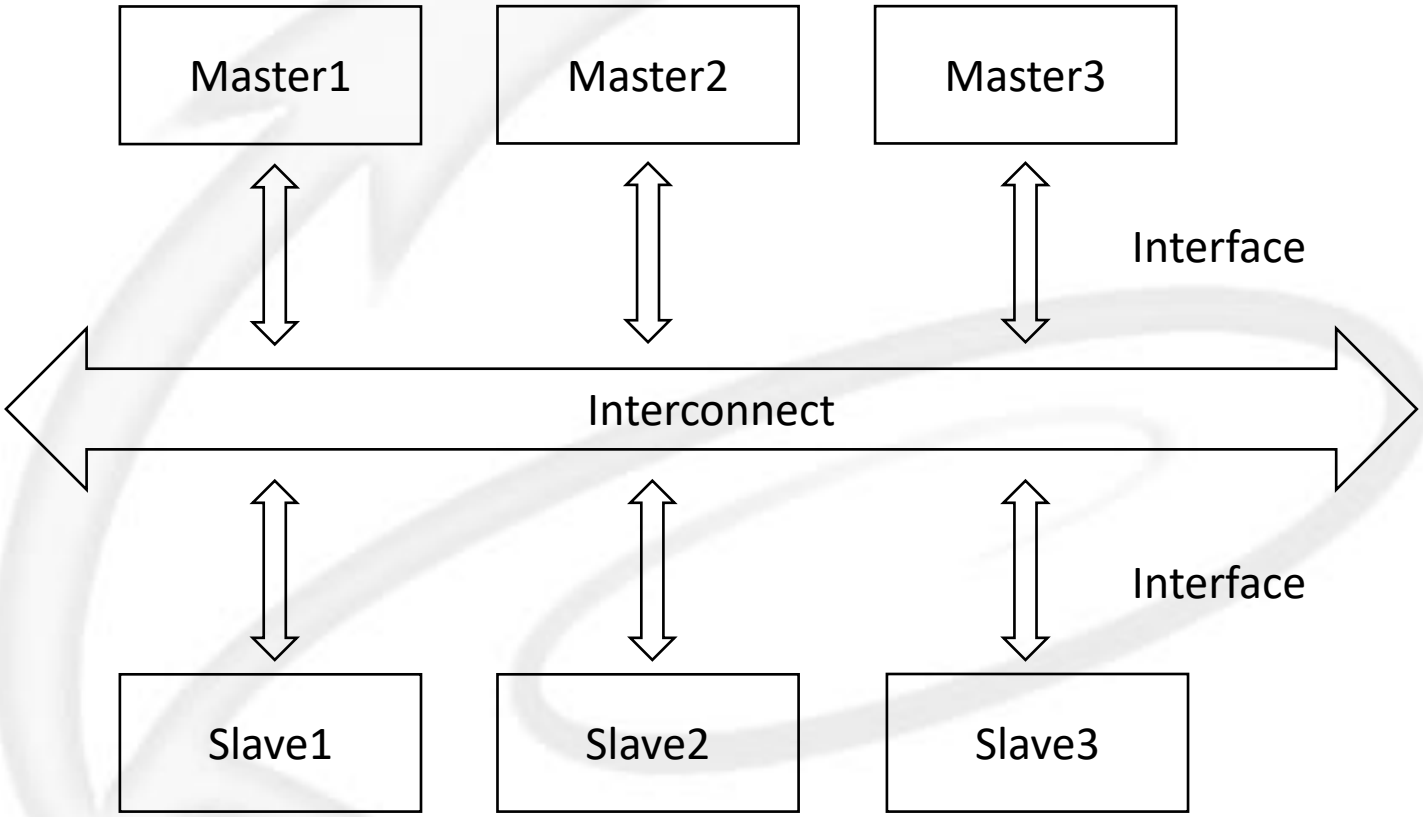
Outline

- 作業內容說明
- Problem 1
  - ➔ 作業驗證說明
- Problem 2
  - ➔ 作業驗證說明
- 作業繳交注意事項

VLSI Design LAB

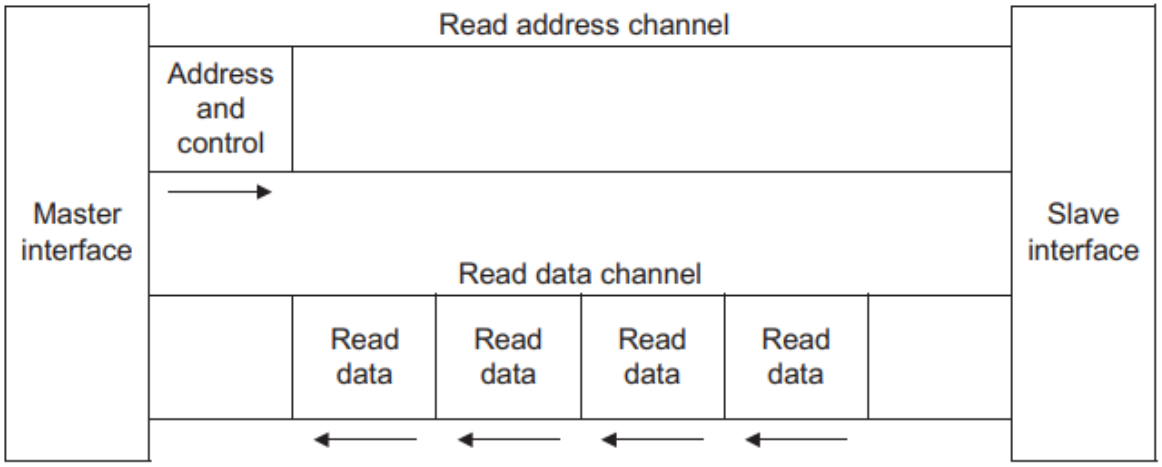
作業內容說明

BUS Architecture



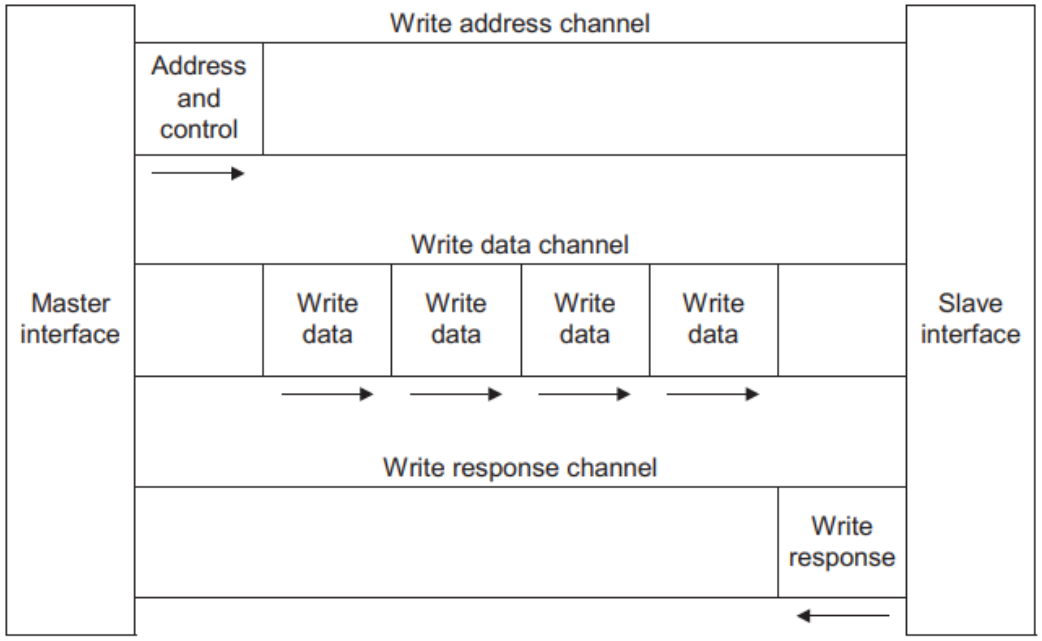
Block Diagram

- 根據AMBA AXI4 Protocol Specification 2.0完成AXI之設計
- Read Channels



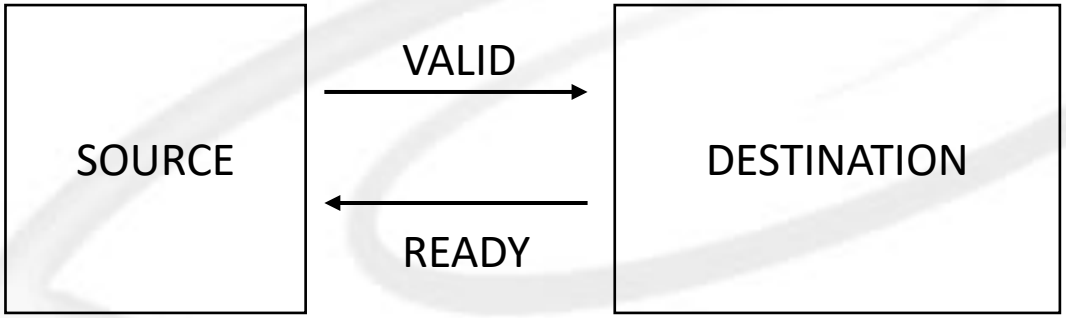
Block Diagram

- Write Channels

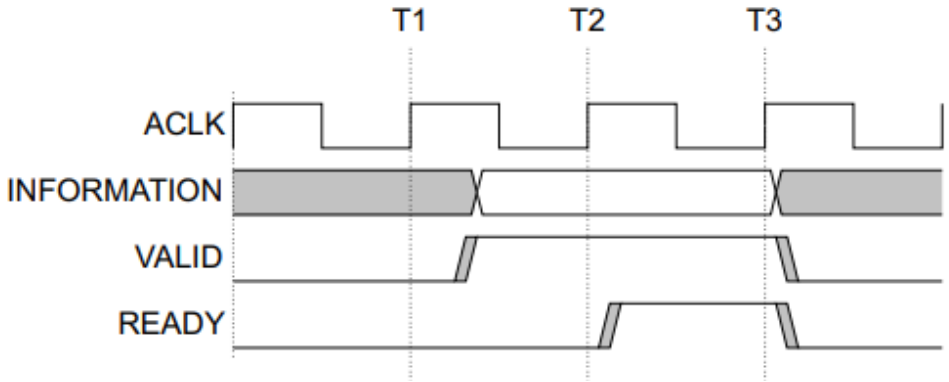


Handshake process

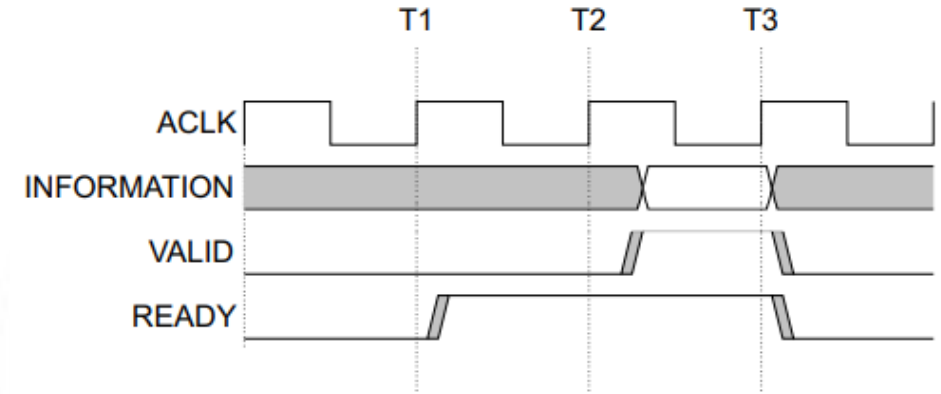
- Each channel uses VALID/READY handshake process to transfer address, data, and control information.
- Transfer occurs only when both the VALID and READY signals are HIGH.



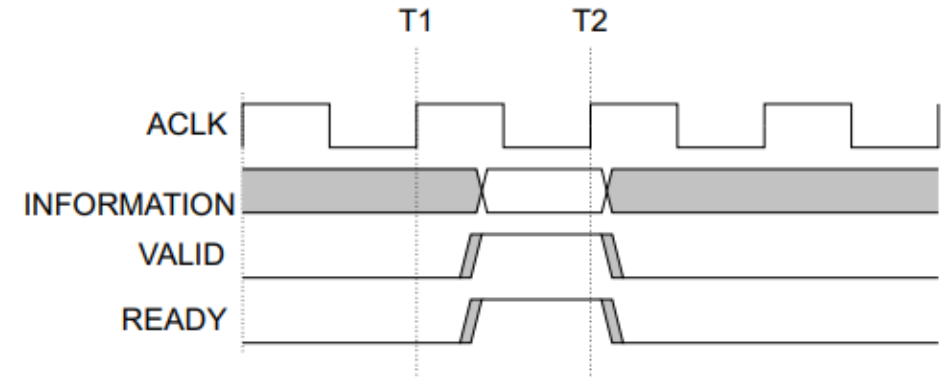
Handshake process



Valid before Ready



Ready before Valid



Valid with Ready

AXI ordering

- AXI protocol enables out-of-order transaction with multiple outstanding addresses.

In order and outstanding=1

address	a1		a2		a3	
data		d1		d2		d3

In order and outstanding>1

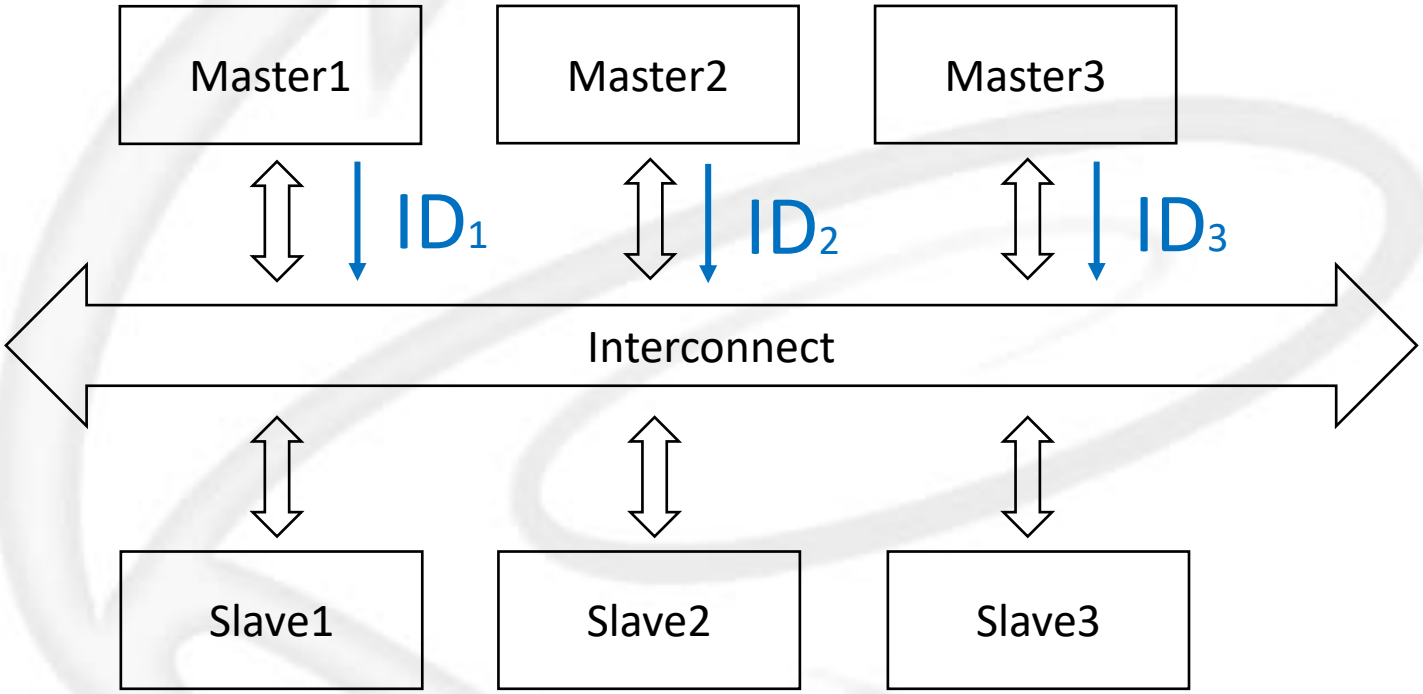
address	a1	a2	a3			
data		d1		d2		d3

Out of order and outstanding>1

address	a1	a2	a3			
data		d1		d3		d2

AXI ordering

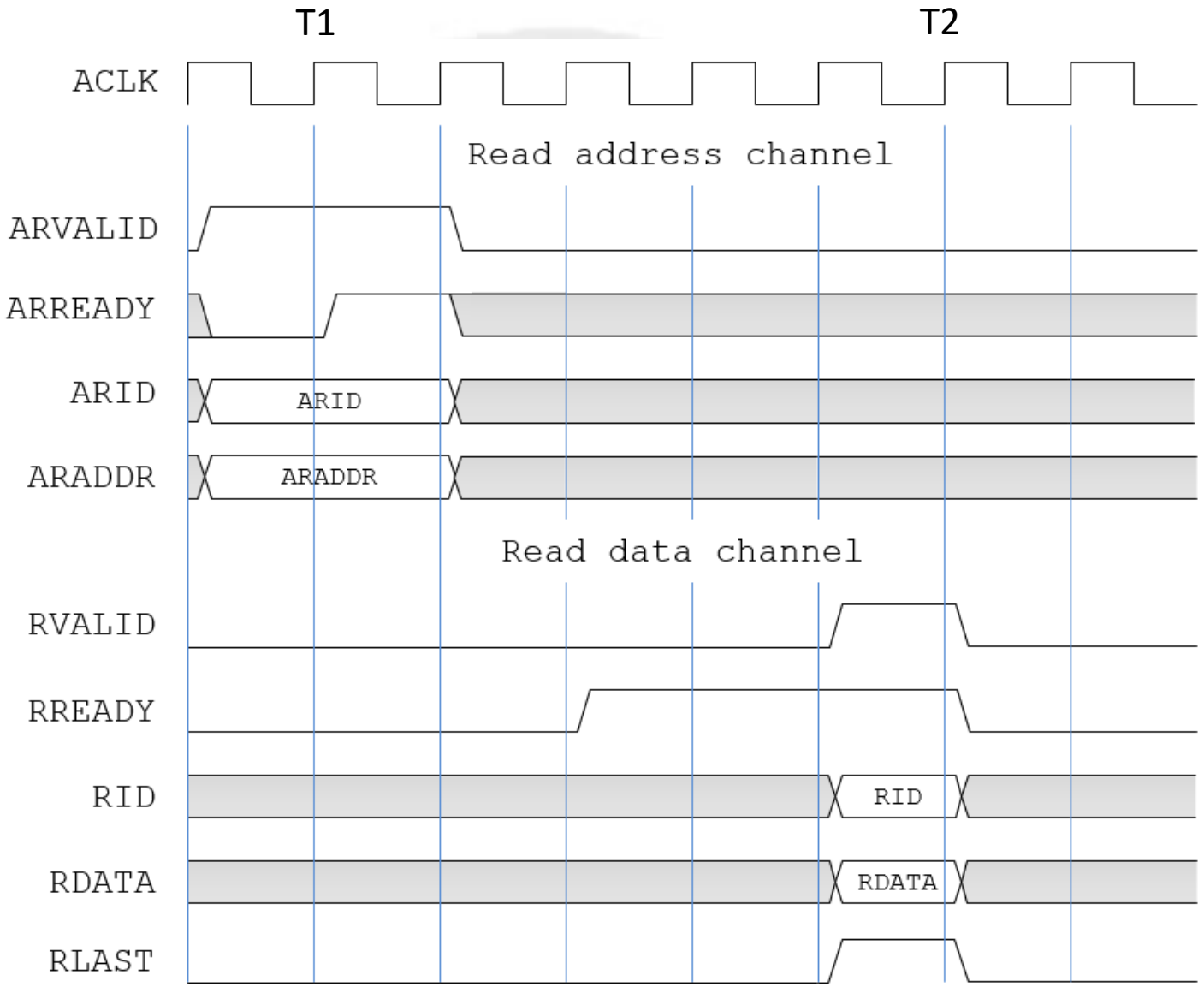
- AXI has ID signals to support out-of-order and outstanding transactions.



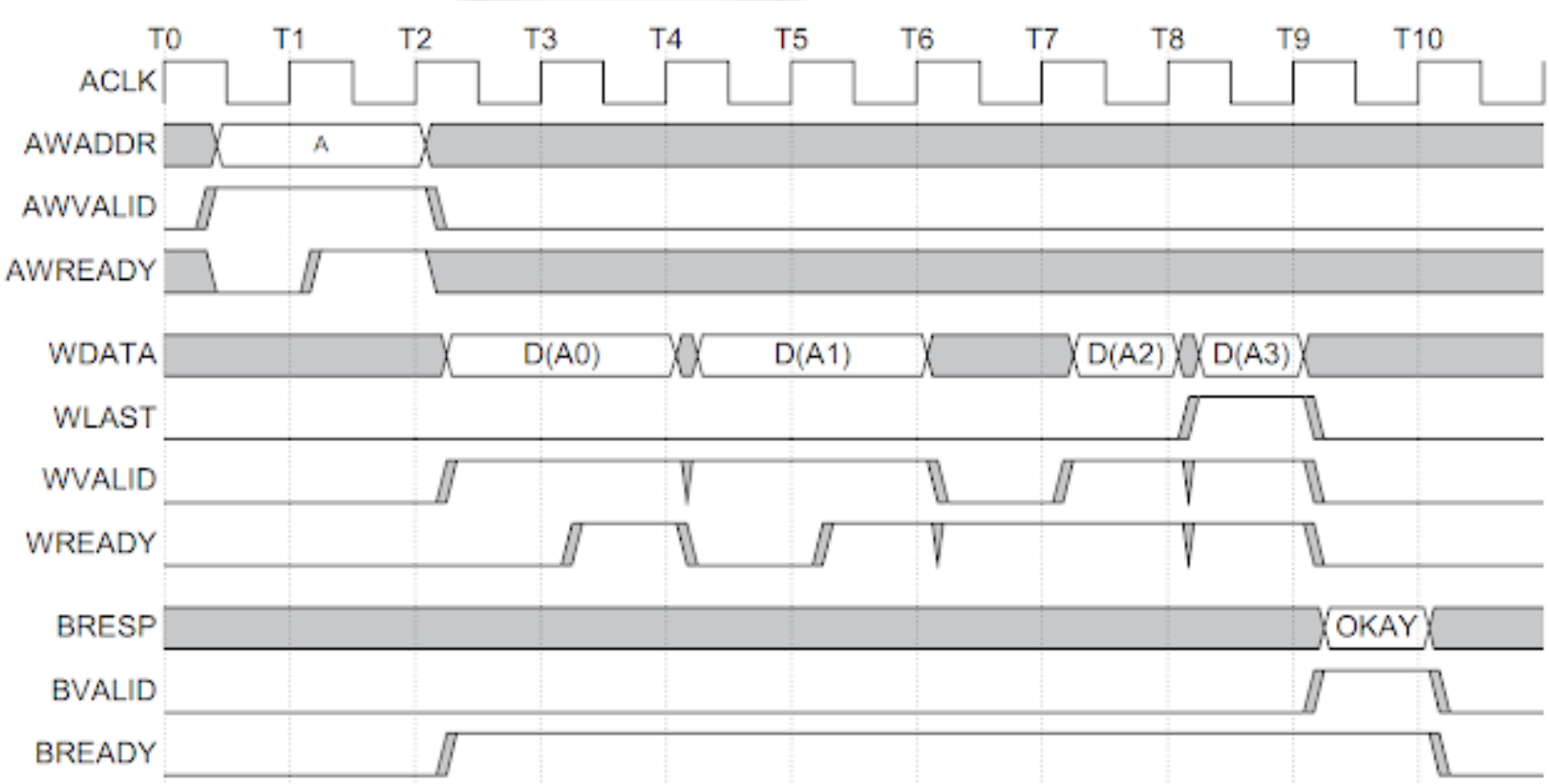
AXI ordering

- Transactions from different masters have no ordering restrictions.
- Transactions from same master, but different ID, have no ordering restrictions.
- The data for a sequence with same AWID or same ARID must complete in same order even if they are aiming at different slaves.
- There are no ordering restrictions with same AWID and ARID.
- For masters that only support single ordered interface, we can tie the ID to a constant number.

AXI Read Transfer (Burst length = 1)

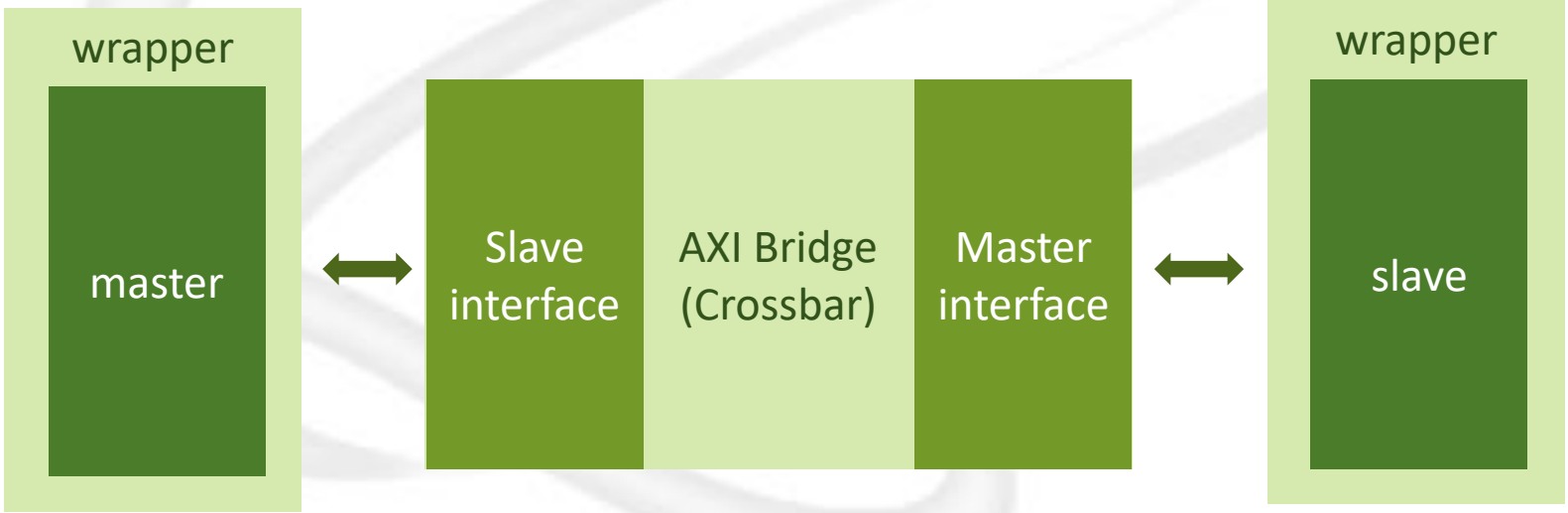


AXI Write Transfer (Burst length = 4)



AXI Interconnect

- AXI Bridge
  - Crossbar
  - Slave Interface
  - Master Interface
- Master(s)
- Slave(s)



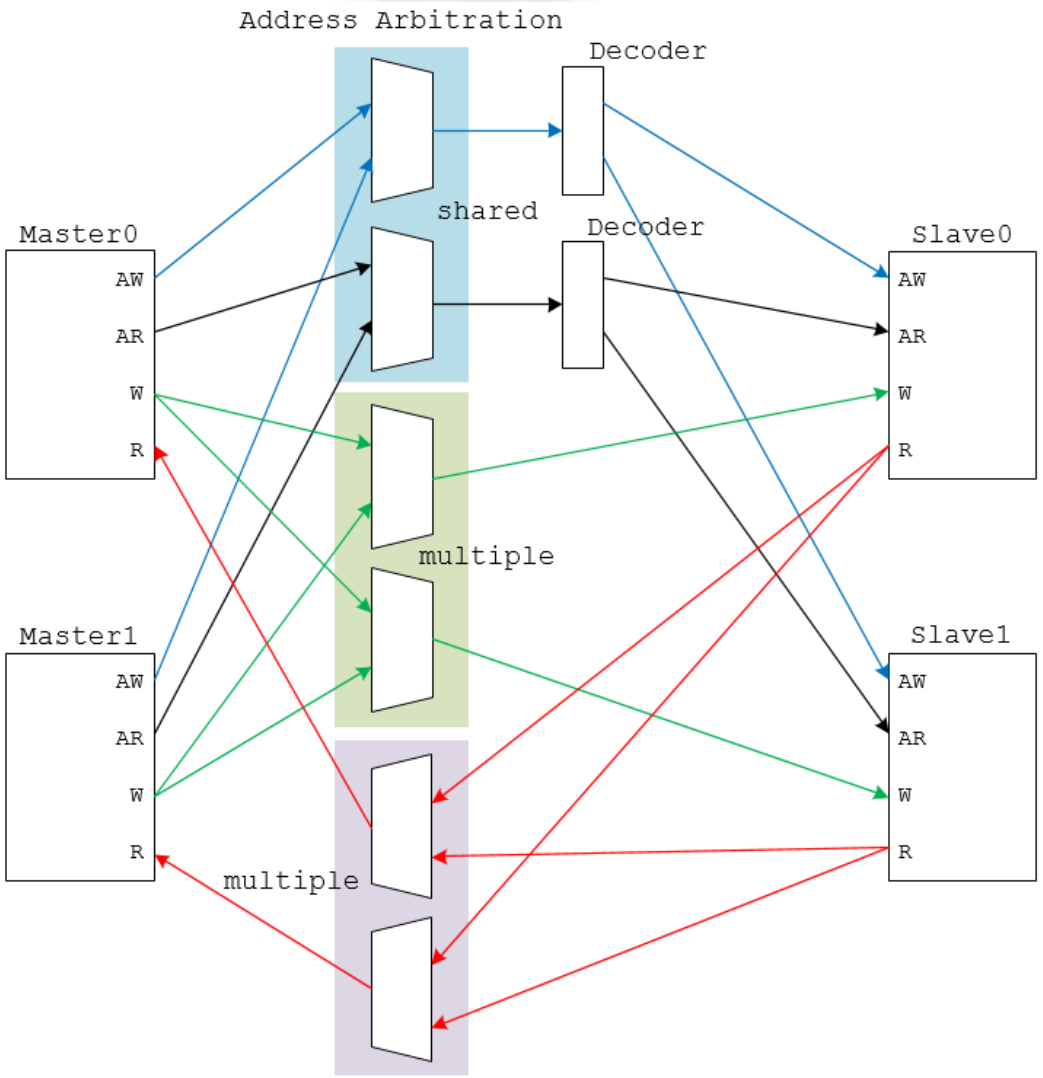
AXI Crossbar

- Selectable Crossbar Architecture
  - Shared Address Multiple Data(SAMD)
  - Shared Address Shared Data(SASD)

AXI Crossbar

- Shared Address Multiple Data(SAMD)
  - High performance.
  - One shared Write address, one shared Read address, and one shared Response buses.
  - Parallel crossbar pathways for data channels.
  - Allow outstanding transaction.

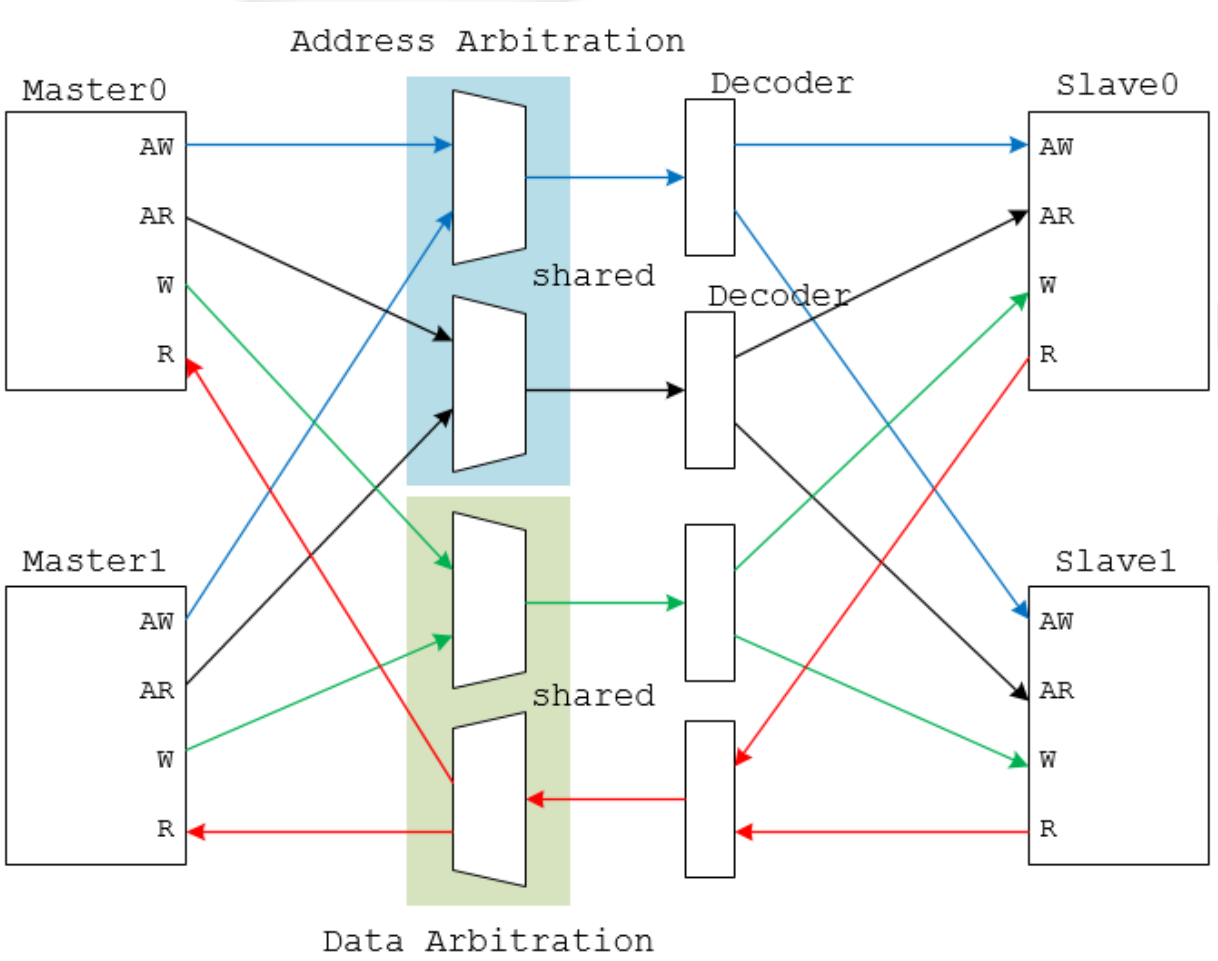
SAMD Architecture



AXI Crossbar

- Shared Address Shared Data(SASD)
  - Low cost.
  - One shared Write data, one shared Read data, one shared Write address, one shared Read address, and one shared Response buses.

SASD Architecture



Port List in this Lab – Global Signals

Signal	Bits	Source	notes
ACLK	1	Clock source	Global clock signal
ARESETn	1	Reset source	Global reset signal, active LOW

--- 詳細的訊號說明請參照spec

Port List in this Lab – Read Address Signals

Signal	Bits	Source	notes
ARID	4/8	Master	Bits of master is 4 and for slave is 8, see A5-80 in spec
ARADDR	32	Master	
ARLEN	4	Master	Burst length.
ARSIZE	3	Master	Burst size.
ARBURST	2	Master	Burst type. Only need to implement INCR type.
ARVALID	1	Master	
ARREADY	1	Slave	

Port List in this Lab – Read Data Signals

Signal	Bits	Source	notes
RID	4/8	Slave	Bits of master is 4 and for slave is 8, see A5-80 in spec
RDATA	32	Slave	
RRESP	2	Slave	Read response. This signal indicates the status of the read transfer.
RLAST	1	Slave	Read last. This signal indicates the last transfer in a read burst.
RVALID	1	Slave	
RREADY	1	Master	

Port List in this Lab – Write Address Signals

Signal	Bits	Source	notes
AWID	4/8	Master	Bits of master is 4 and for slave is 8, see A5-80 in spec
AWADDR	32	Master	
AWLEN	4	Master	Burst length.
AWSIZE	3	Master	Burst size.
AWBURST	2	Master	Burst type. Only need to implement INCR type.
AWVALID	1	Master	
AWREADY	1	Slave	

Port List in this Lab – Write Data Signals

Signal	Bits	Source	notes
WDATA	32	Master	
WSTRB	4	Master	Write strobes. 4 bits because 32/8 = 4
WLAST	1	Master	Write last. This signal indicates the last transfer in a write burst.
WVALID	1	Master	
WREADY	1	Slave	

Port List in this Lab – Write Response Signals

Signal	Bits	Source	notes
BID	4/8	Slave	Bits of master is 4 and for slave is 8, see A5-80 in spec
BRESP	4	Slave	Write response. This signal indicates the status of the write transaction.
BVALID	1	Slave	
BREADY	1	Slave	

Specification (1/2)

- Master : Single Transfer.
  - ➔ Burst length = 1
- Bridge and Slave : Burst transfer.
  - ➔ Burst length up to 2
- 2 masters and 2 slaves

Specification (2/2)

- Slave
  - Slave 1: 0x0000_0000 – 0x0000_ffff
  - Slave 2: 0x0001_0000 – 0x0001_ffff
  - Default slave: 0x0002_0000 – 0xffff_ffff
- Default slave
  - Response ERROR when masters access (RESP == DECERR) it
- Default master
  - Always execute valid = 0 in the transfer
  - You don't need to create a default master module

Module

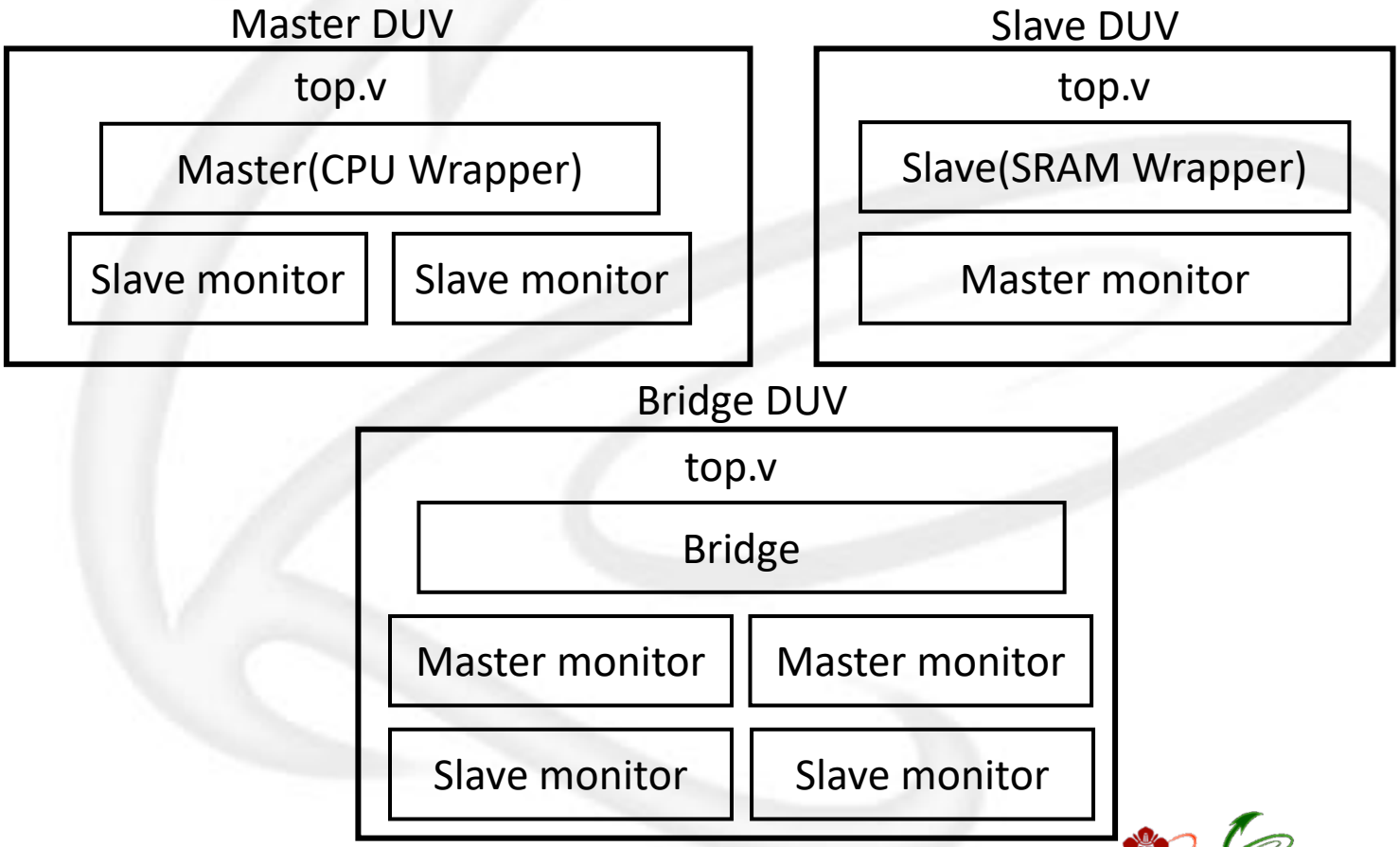
- Module names and module ports of AXI are defined. DO NOT modify them. The definition of Module ports can be checked in AXI spec.
- Ignore the signals that are not in this homework requirement.

Problem 1

作業驗證說明

Verification(1/3)

- 利用JasperGold的Verification IP分別去驗證CPU Wrapper, AXI, SRAM Wrapper



Verification(2/3)

- 請不要更改.tcl
- 在驗證MASTER DUV的時候，可能會出現前提不成立的狀況 

(如：CPU 永遠READY)，需在報告中解釋原因，若為不合理原因則應修改設計
- Assertion應全數通過才算完整
- 請勿更動top.v的parameter

Verification(3/3)

Table B-1: Simulation commands

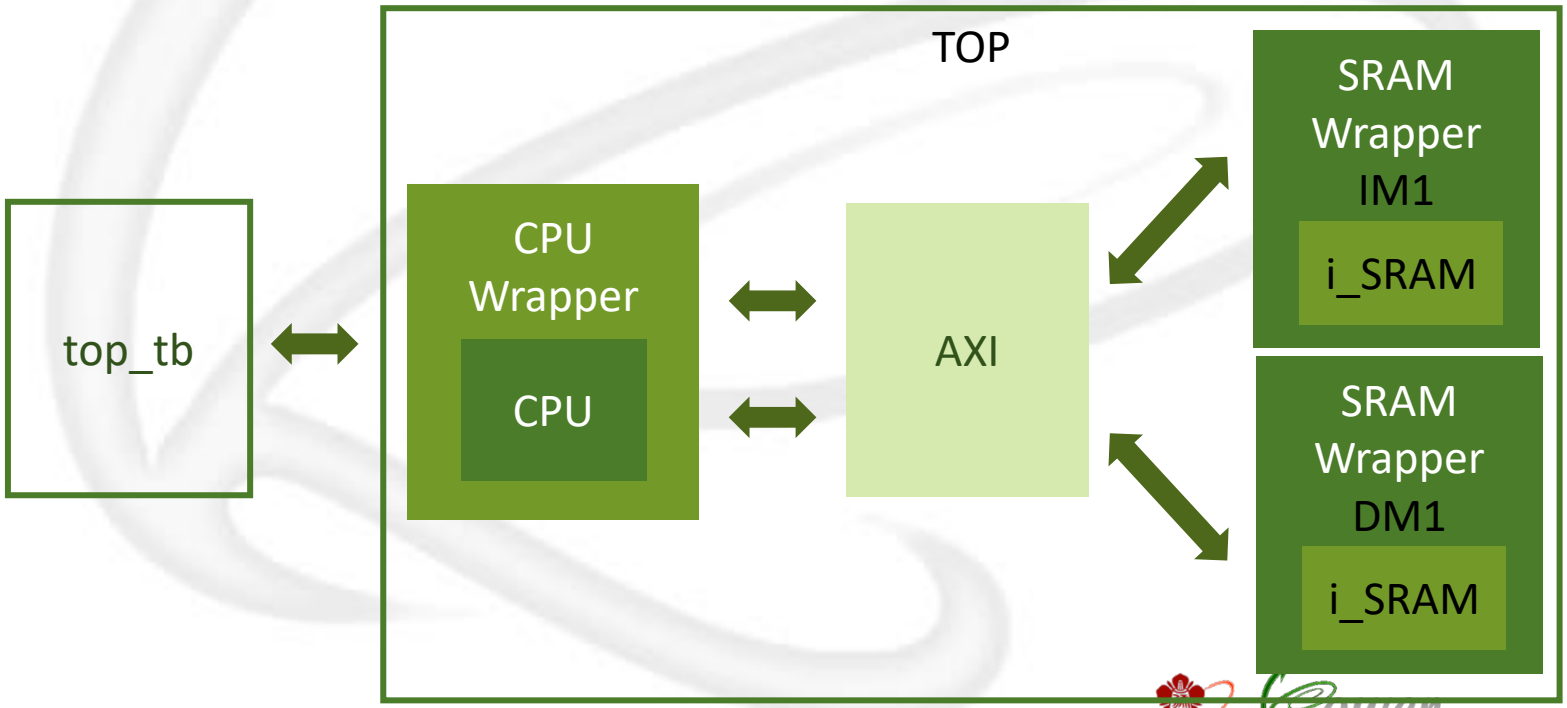
Situation	Command
Run JasperGold VIP on AXI bridge without file pollution (RTL only)	make vip_b
Run JasperGold VIP on AXI master without file pollution (RTL only)	make vip_m
Run JasperGold VIP on AXI slave without file pollution (RTL only)	make vip_s
Delete built files for simulation, synthesis or verification	make clean
Check correctness of your file structure	make check
Compress your homework to *tar* format	make tar

Problem 2

作業驗證說明

Architecture

- CPU Wrapper
  - Instruction fetch (read only interface)
  - Load or store
- AXI
- SRAM Wrapper



Program

- prog0
  - 測試33個instruction (助教提供)
- prog1
  - Sort Algorithm
- prog2
  - Multiplication
- prog3
  - Greatest common divisor

Specification

- Don’t modify any timing constraint except clock period in *DC.sdc*. Maximum clock period is 20 ns.
- Design the master wrapper between CPU and AXI.
  - Transfer Signals between CPU and AXI
- Modify SRAM_wrapper to be compatible with AXI.
- CPU has two masters
  - Instruction
  - Data
- IM (Slave 1)
  - 0x0000_0000 – 0x0000_ffff
- DM (Slave 2)
  - 0x0001_0000 – 0x0001_ffff

Module (1/2)

- Module name須符合下表要求

Category	Name			
	File	Module	Instance	SDF
RTL	top.sv	top	TOP	
Gate-Level	top_syn.v	top	TOP	top_syn.sdf
RTL	SRAM_wrapper.sv	SRAM_wrapper	IM1	
RTL	SRAM_wrapper.sv	SRAM_wrapper	DM1	
RTL	SRAM_rtl.sv	SRAM	i_SRAM	

- 紫色部分為助教已提供或已定義好，請勿任意更改
- 其餘部分需按照要求命名，以免testbench抓不到正確的名稱

Module (2/2)

- Module port須符合下表要求(同HW1)

Module	Specifications			
top	**Name**	**Signal**	**Bits**	**Function explanation**
	clk	input	1	System clock
	rst	input	1	System reset (active high)
SRAM	Memory Space			
	Memory_byte0	logic	8	Size: [16384]
	Memory_byte1	logic	8	Size: [16384]
	Memory_byte2	logic	8	Size: [16384]
	Memory_byte3	logic	8	Size: [16384]

- 紫色部分為助教已提供或已定義好，請勿任意更改
- 其餘部分需按照要求命名，以免testbench抓不到正確的名稱

Simulation

Table B-1: Simulation commands (Partial)

Simulation Level	Command
Problem1	
RTL	make rtl_all
Post-synthesis (optional)	make syn_all

Table B-2: Makefile macros (Partial)

Situation	Command	Example
RTL simulation for progX	make rtlX	make rtl0
Post-synthesis simulation for progX	make synX	make syn1
Dump waveform (no array)	make {rtlX,synX} FSDB=1	make rtl2 FSDB=1
Dump waveform (with array)	make {rtlX,synX} FSDB=2	make syn3 FSDB=2
Open nWave without file pollution	make nWave	
Open Superlint without file pollution	make superlint	
Open DesignVision without file pollution	make dv	
Synthesize your RTL code (You need write *synthesis.tcl* in *script* folder by yourself)	make synthesize	
Delete built files for simulation, synthesis or verification	make clean	
Check correctness of your file structure	make check	
Compress your homework to *tar* format	make tar	

作業繳交注意事項

Report

- 請使用附在檔案內的Submission Cover
- 請勿將code貼在.docx內
  - 請將.sv包在壓縮檔內，不可截圖於.docx中
- 需要Summary及Lessons learned
- 若兩人為一組，須寫出貢獻度
  - Ex: A(N26081234) 55%, B(N26085678) 45%
  - Total 100%
  - 自己一組則不用寫

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)

- 若兩人為一組，只需一個人上傳作業到Moodle
  - 兩人都上傳會斟酌扣分
- 若兩人為一組，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，另一位只需在StudentID2及Submission Cover內寫上自己的學號。
  - Ex: A(N26071234)負責上傳，組員為B(N26075678)
  - N26081234.tar (壓縮檔)
    N26081234 (主資料夾)
    N26081234.docx (Report，Cover寫上兩者的學號)
    StudentID (裡面填上N26081234)
    StudentID2 (裡面填上N26085678)
- 自己一組請直接刪除StudentID2檔案

檔案結構

- 參考 Appendix A
- sim/CYCLE
  - → Specify your clock cycle time
- sim/MAX
  - → Specify max clock cycle number
- sim/prog0
  - → Don’t modify contents
- sim/progX (X ≠ 0)
  - → main.S
  - → main.c
    - Submit one of these

繳交期限

- 2021/11/10 (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

注意事項

- 本次作業合成的部分有計入PA(Performance & Area)，表現較為優異者將有較高的評分
- 作業部分有任何問題請在moodle上的作業討論區發問並可參考其他人是否有類似的問題，助教信箱恕不回覆。
- 在Script/DC.sdc中，可以更改clk period範圍，請注意最大的接受值為20.0，超過此範圍者恕不納入計分。此外，若是有更改預設clock period者，請務必更改set_input_delay –max至1/2 clk period，以利後續作業的合成及模擬。
- 本次作業需在SoC環境下模擬，請同學務必在SoC教室執行模擬，若是助教在評分作業時於SoC無法成功模擬則以0分計算。

VLSI Design LAB

Thanks for your participation and attendance !!