Lab08
Channel and Interface

Finite state machine (FSM)
Protocol, Interface

林昱維

Purpose

- Be familiar with advanced FSM
- Introduce high speed communication protocol

System

- 一個系統可以分成三個部分
  - Master: 主設備，通常是運算單元等，讀取資料做運算
  - Interconnect: 包含decoder, arbiter等，負責處理不同設備間連接與先後問題
  - Slave: 從設備，通常是memory，負責儲存資料

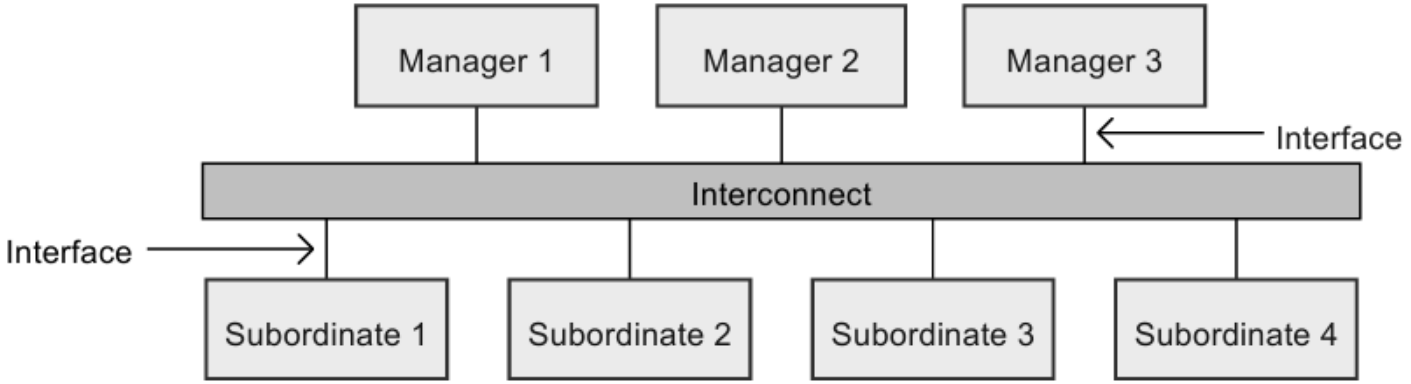
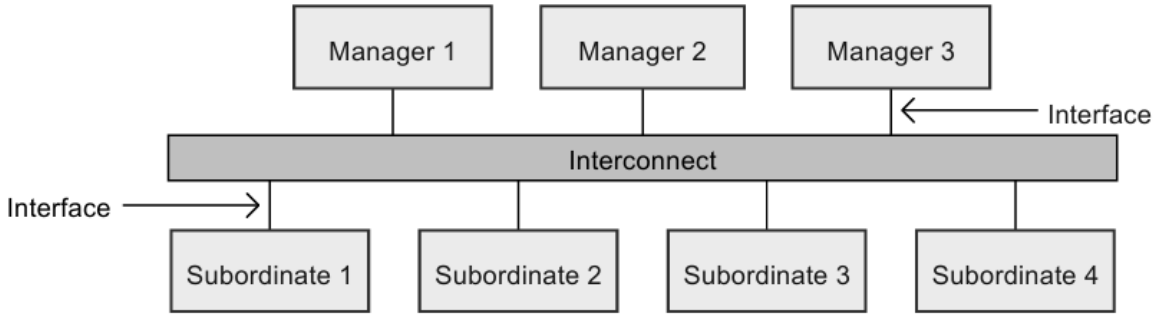


Image Source: AMBA AXI protocol spec https://developer.arm.com/documentation/ihi0022/latest/

Protocol

- 常見名詞
  - Bus: 傳輸通道，像是data bus, control bus, address bus等
  - Interface: 連接標準
  - Protocol: 數據傳輸規則，定義所有設備、interface、bus...等之間的關係
- AXI是一種高速傳輸協議，其中定義了下列之間的interface定義
  - Master – interconnect
  - Interconnect – slave
  - Master - slave



AXI4 bus

- AXI共有5組channel
- 可同時執行read和write

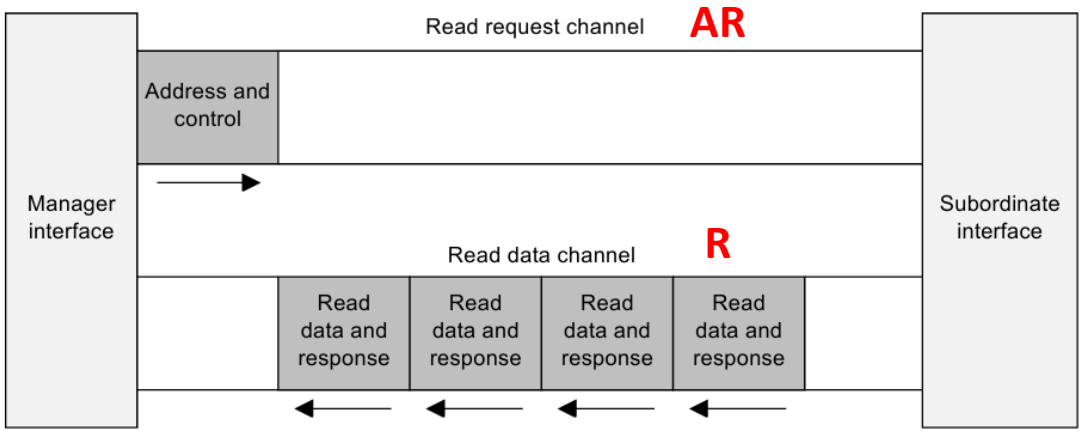


Figure A1.2: Channel architecture of reads

AXI4 read channel architecture

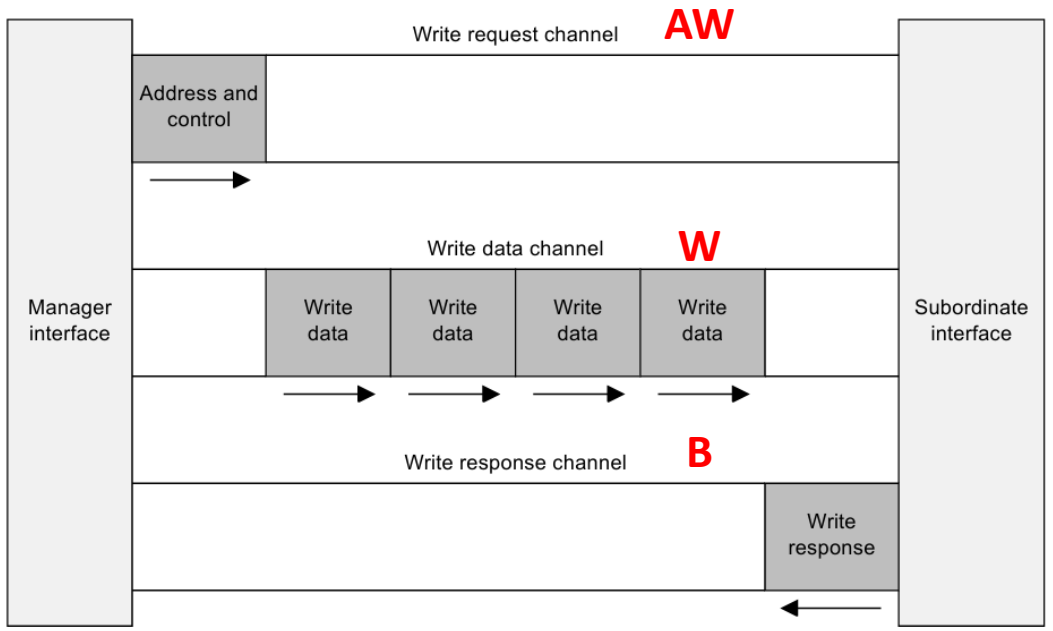


Figure A1.1: Channel architecture of writes

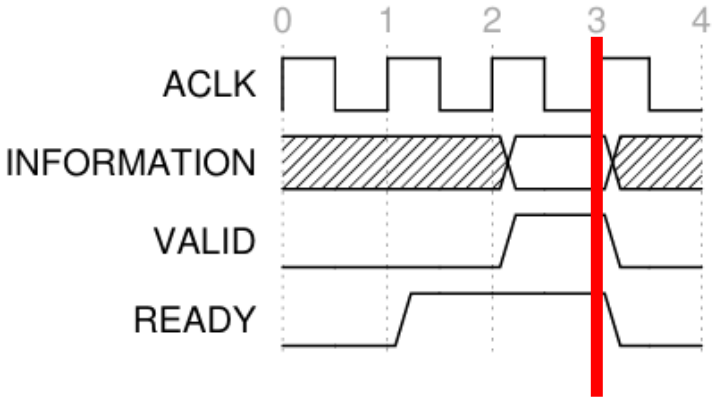
AXI4 write channel architecture

Image Source: AMBA AXI protocol spec https://developer.arm.com/documentation/ihi0022/latest/

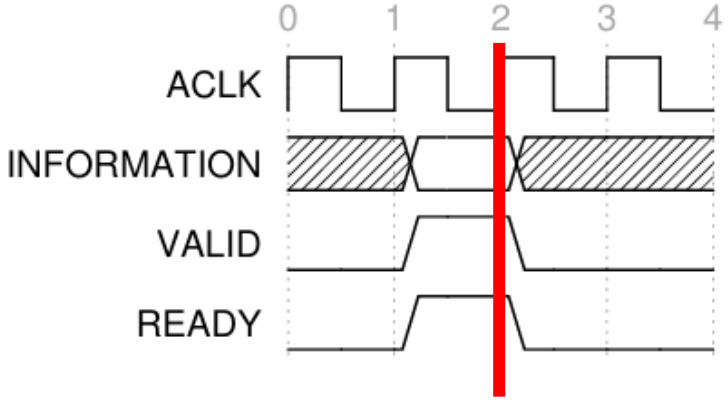
Handshake process

- Valid: 傳輸端，值為1代表資料準備好可以傳輸
- Ready: 接收端，值為1代表準備好接收資料
- data只有在valid和ready同時拉起時才能順利傳遞

ready先拉起
valid後拉起



valid & ready
同時拉起



valid先拉起
ready後拉起

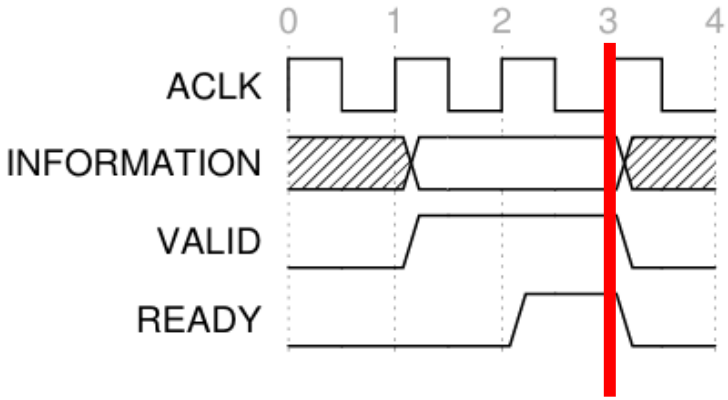
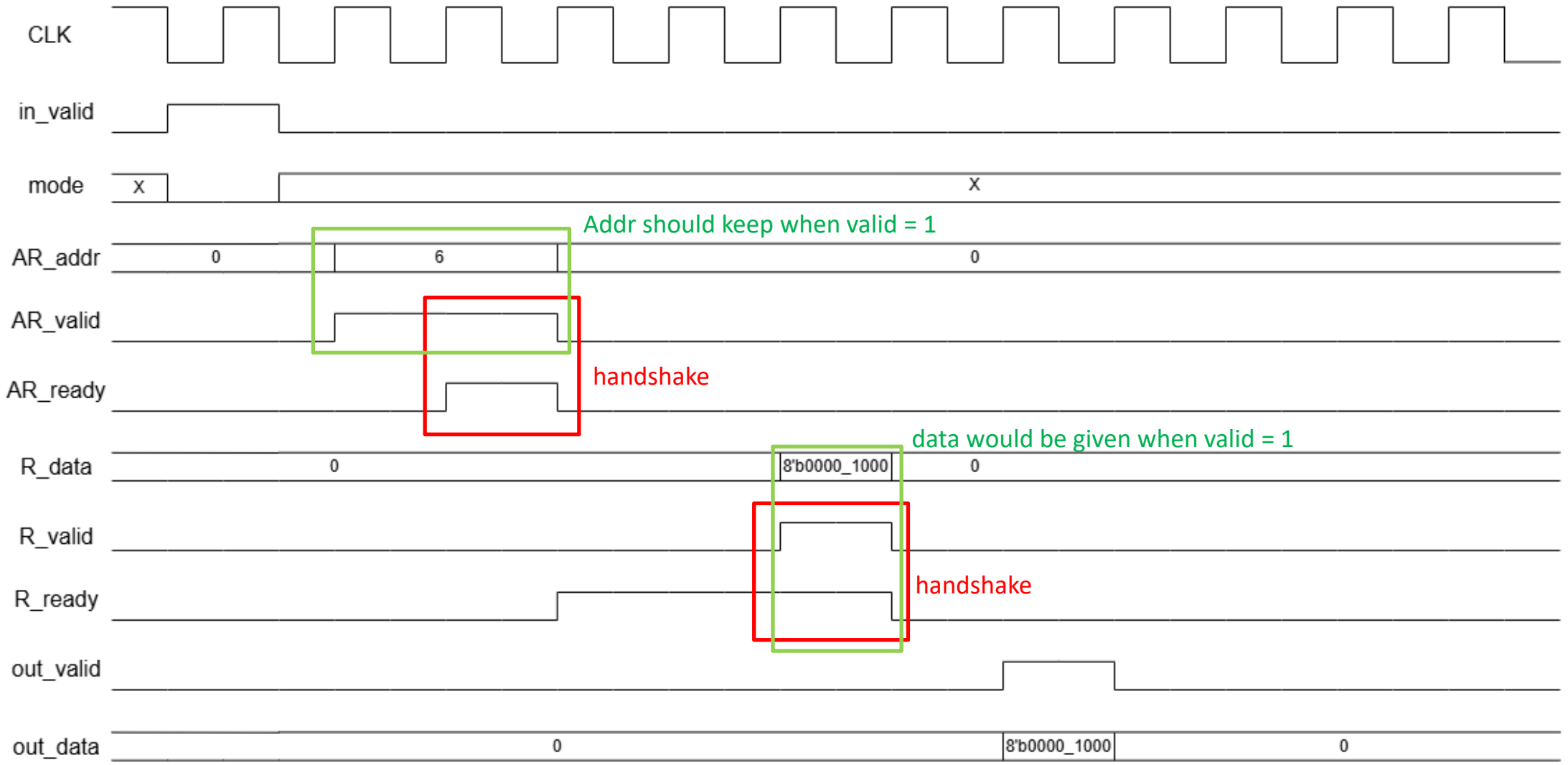
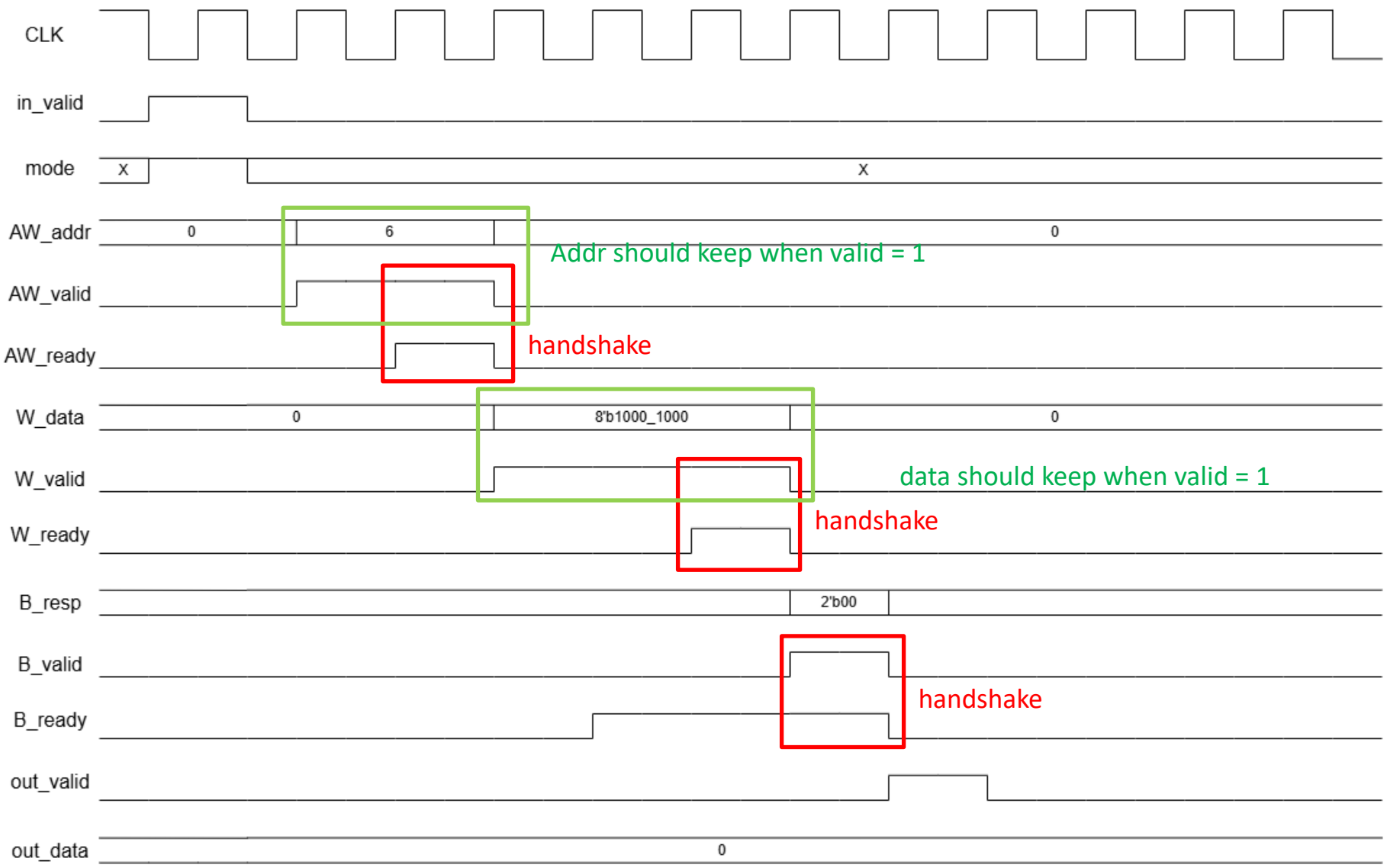


Image Source: AMBA AXI protocol spec https://developer.arm.com/documentation/ihi0022/latest/

Simplified AXI transaction – read (Not this Lab)



Simplified AXI transaction – write (Not this Lab)



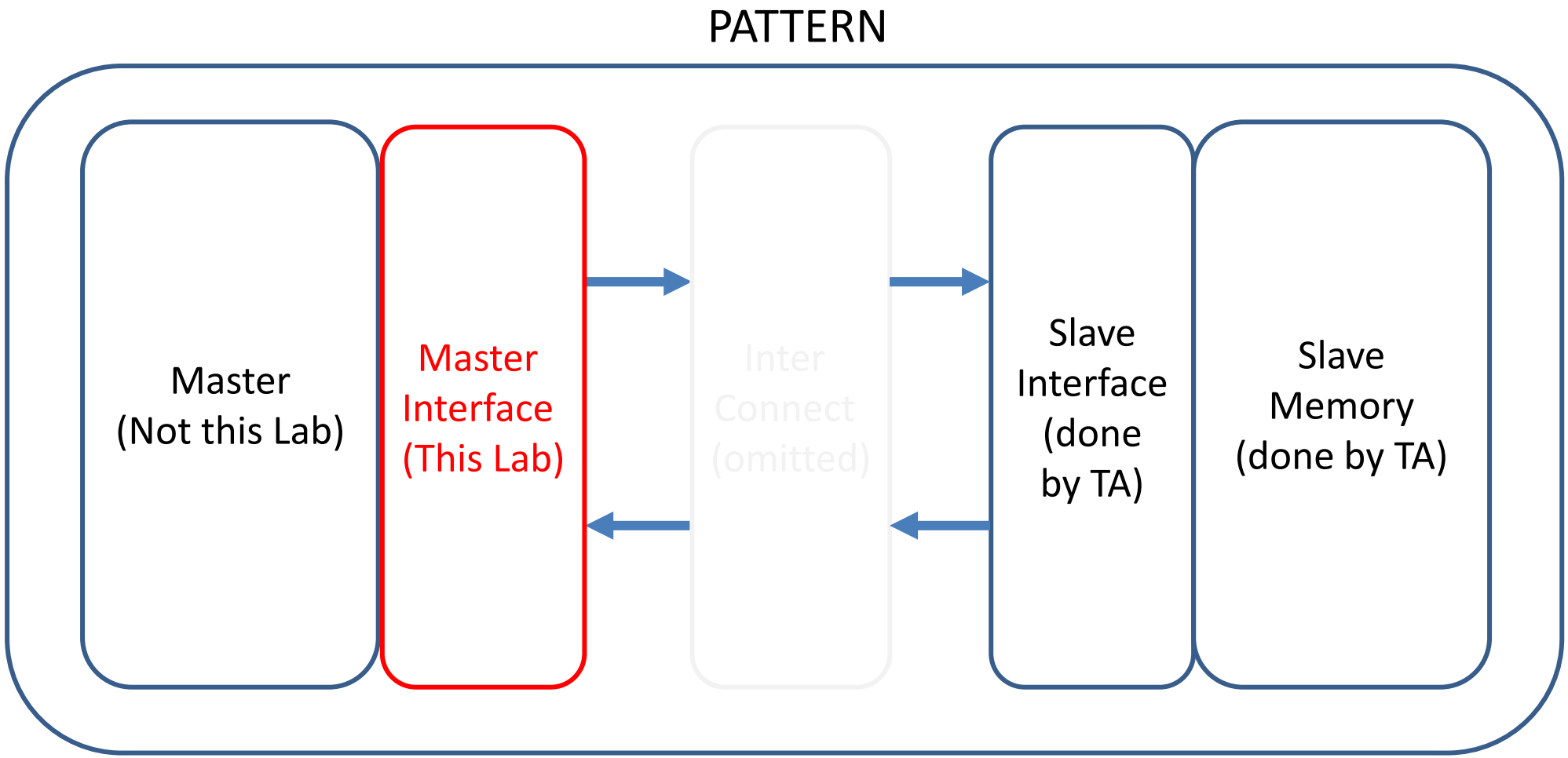
AW phase
傳輸addr

W phase
傳輸data

B phase
確認傳輸

Lab block diagram

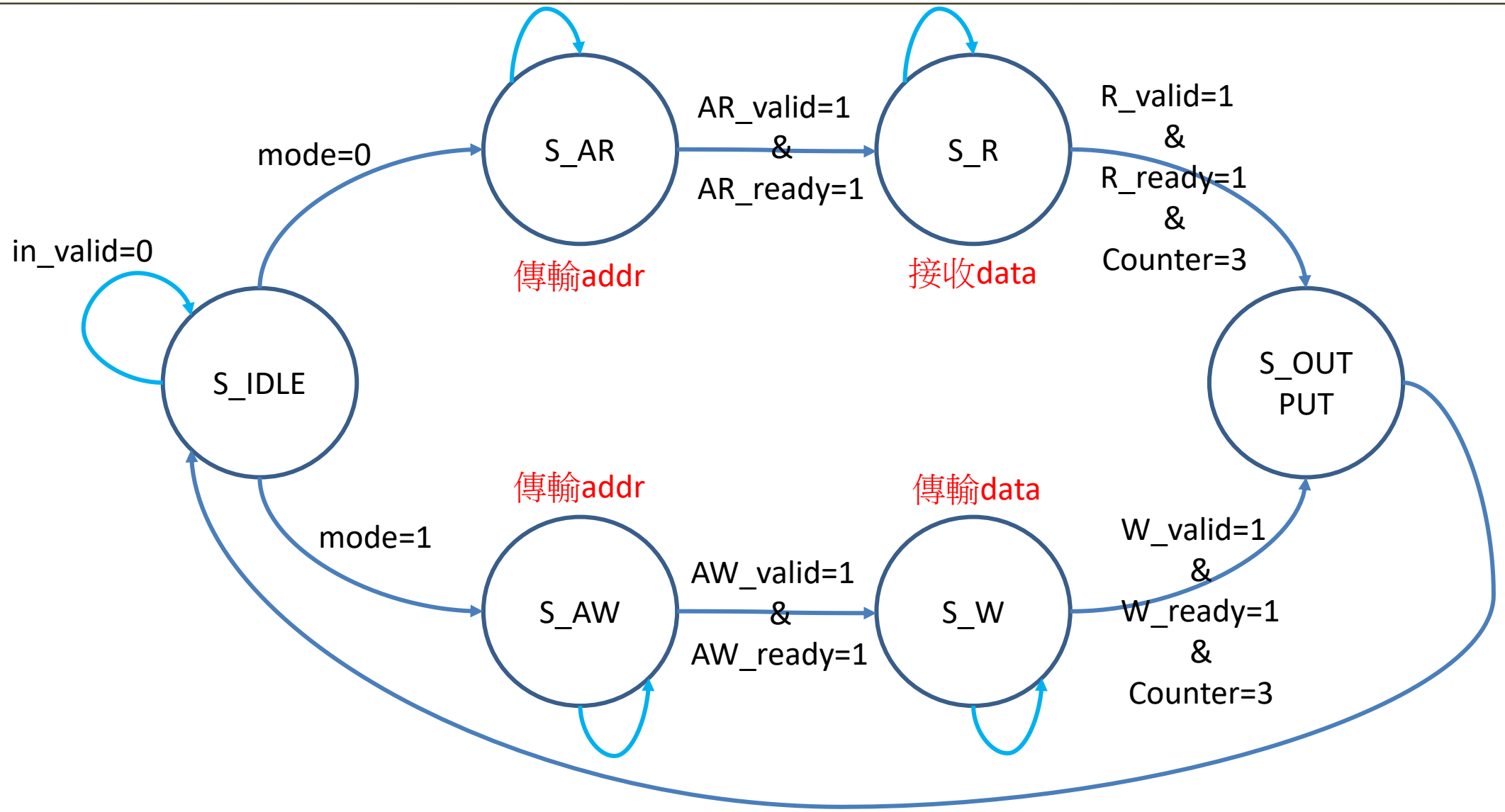
- Interconnect is omitted since there's only 1 master and 1 slave in this Lab



Lab explain

- 這次Lab要實作的是在master interface對slave進行資料傳輸
- Input給予mode, addr與data，design要負責進行memory access
- Design完成memory access後要將結果output回傳回去

Lab FSM


S_IDLE
S_AR
S_R
S_OUT
PUT
S_AW
S_W
in_valid=0
mode=0
mode=1
AR_valid=1
&
AR_ready=1
R_valid=1
&
R_ready=1
&
Counter=3
傳輸addr
接收data
傳輸addr
傳輸data
AW_valid=1
&
AW_ready=1
W_valid=1
&
W_ready=1
&
Counter=3

Ports

Signals	Bit width	in/output	Definition
clk	1	input	Clock
rst_n	1	input	Asynchoronous active-low reset
in_valid	1	input	High when input is valid
in_mode	1	input	Read / write mode (read=0, write=1)
in_addr	4	input	Read / write address
in_data	8	input	Write data, will only be given at write mode

Signals	Bit width	In/output	Definition
out_valid	1	output	Read / write finish
out_data	8	output	Read data from slave memory

AXI read channel ports

- Read port

Signals	Bit width	In/output	Definition
ar_addr	4	output	Read address
ar_valid	1	output	Read address valid
ar_ready	1	input	Read address ready, 1 when memory confirm address
r_data	8	Input	Read data
r_valid	1	input	Read data valid, 1 when memory read successfully
r_ready	1	output	Read data ready, after AR output 1 when read is ready

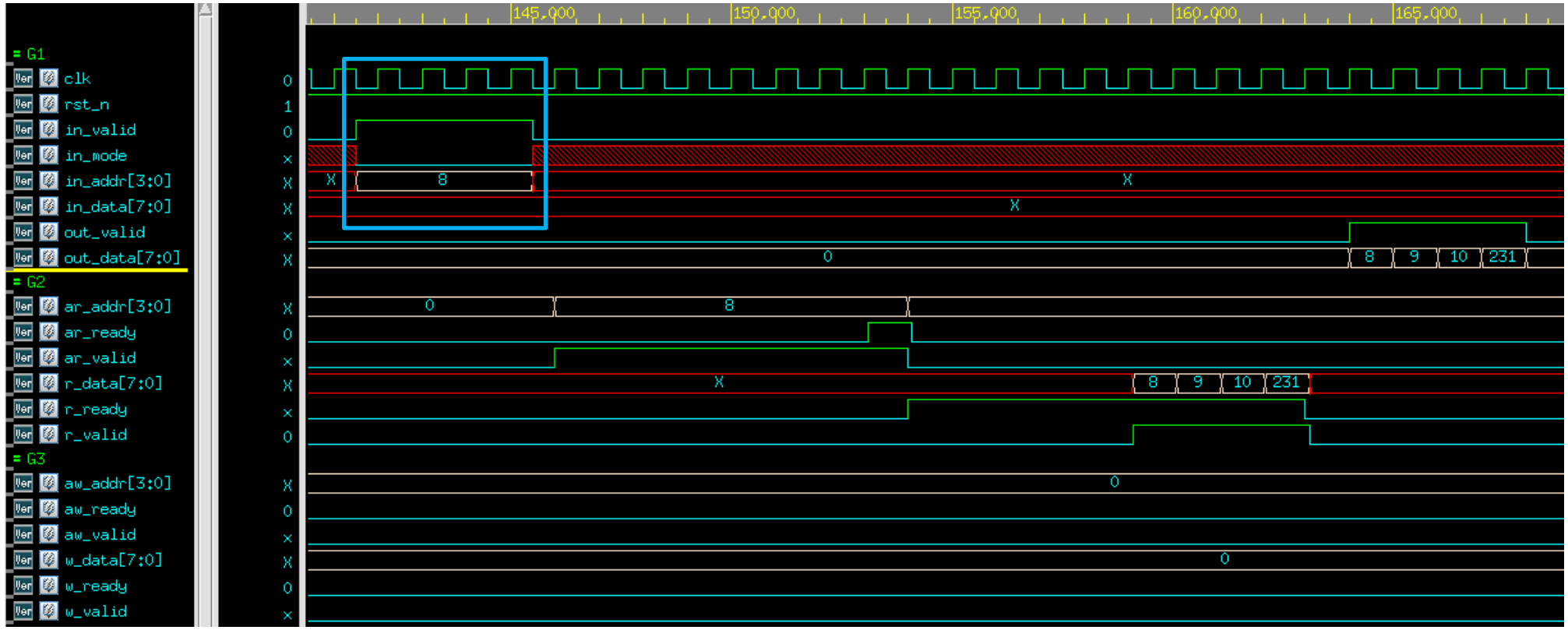
AXI write channel ports

- Write port

Signals	Bit width	In/output	Definition
aw_addr	4	output	Write address
aw_valid	1	output	Write address valid
aw_ready	1	input	Write address ready, 1 when memory confirm address
w_data	8	output	Write data
w_valid	1	output	Write data valid, after AW output 1 when data is valid
w_ready	1	input	Write data ready, 1 when memory write successfully

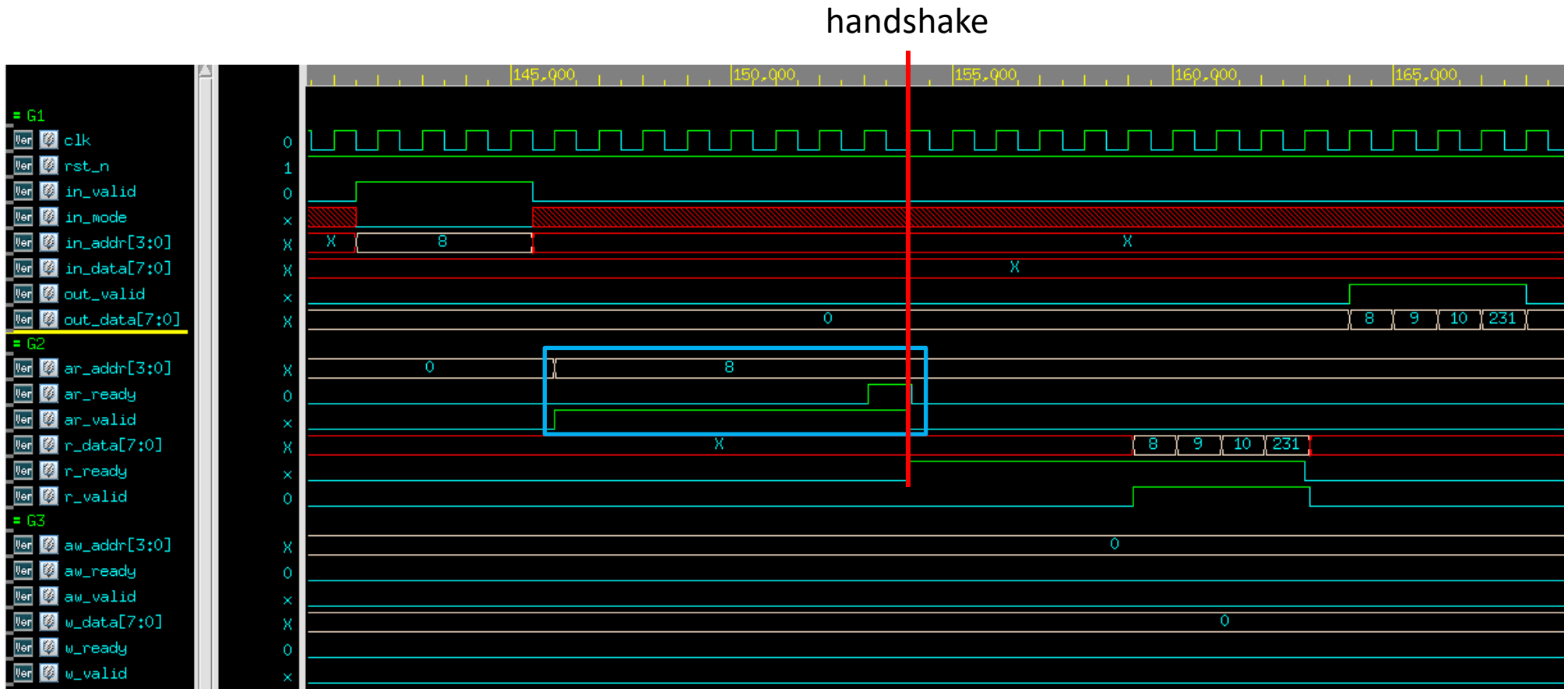
Waveform - read

- in_valid 4 cycles
- in_mode & in_addr會連續給予4 cycles，但值完全相同，所以實際上僅一筆輸入，因指令及位址只會有一個
- in_data會維持unknown，因為不須寫入資料



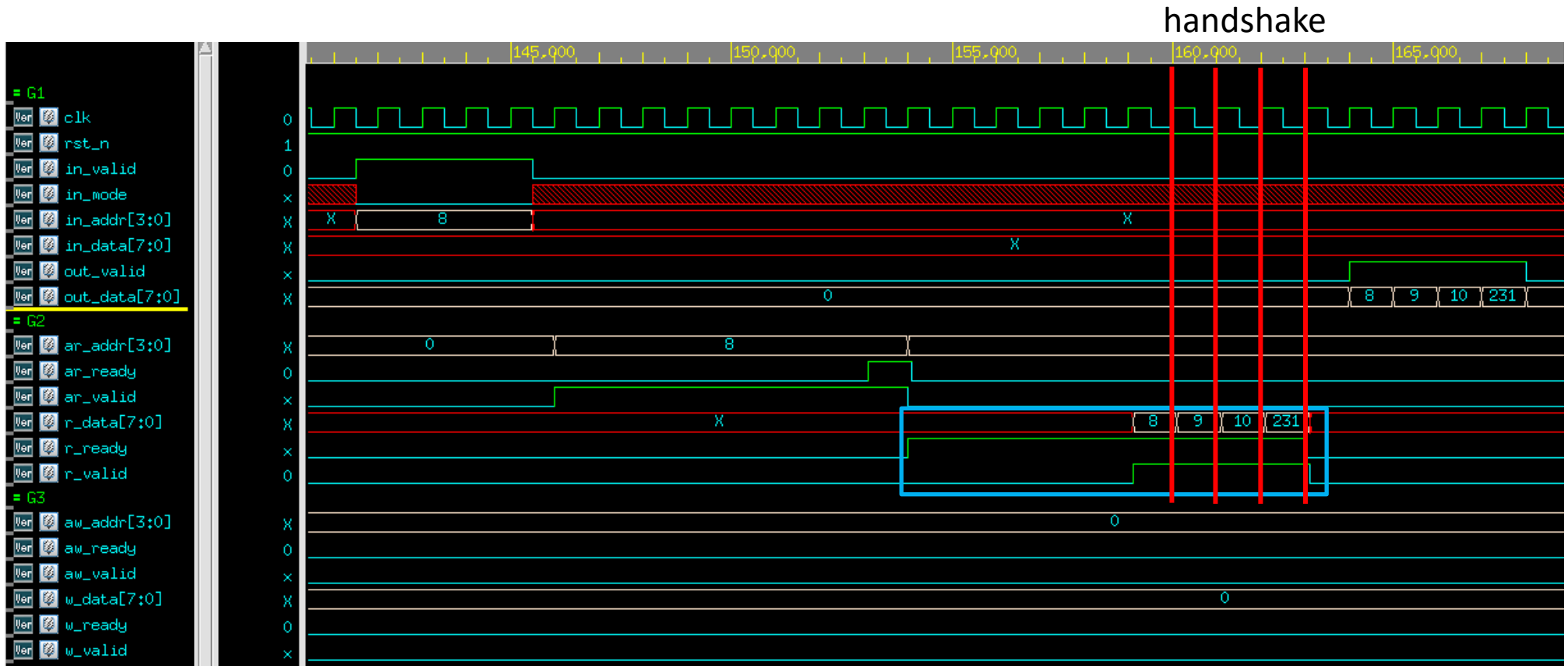
Waveform - read

- input完成後，必須在30 cycle內拉起ar_valid, ar_addr向slave確認addr
- 等待ar_ready，當slave回應ar_ready後完成handshake，進入R phase
- ar_valid, ar_addr在handshake完成後必須馬上reset回0



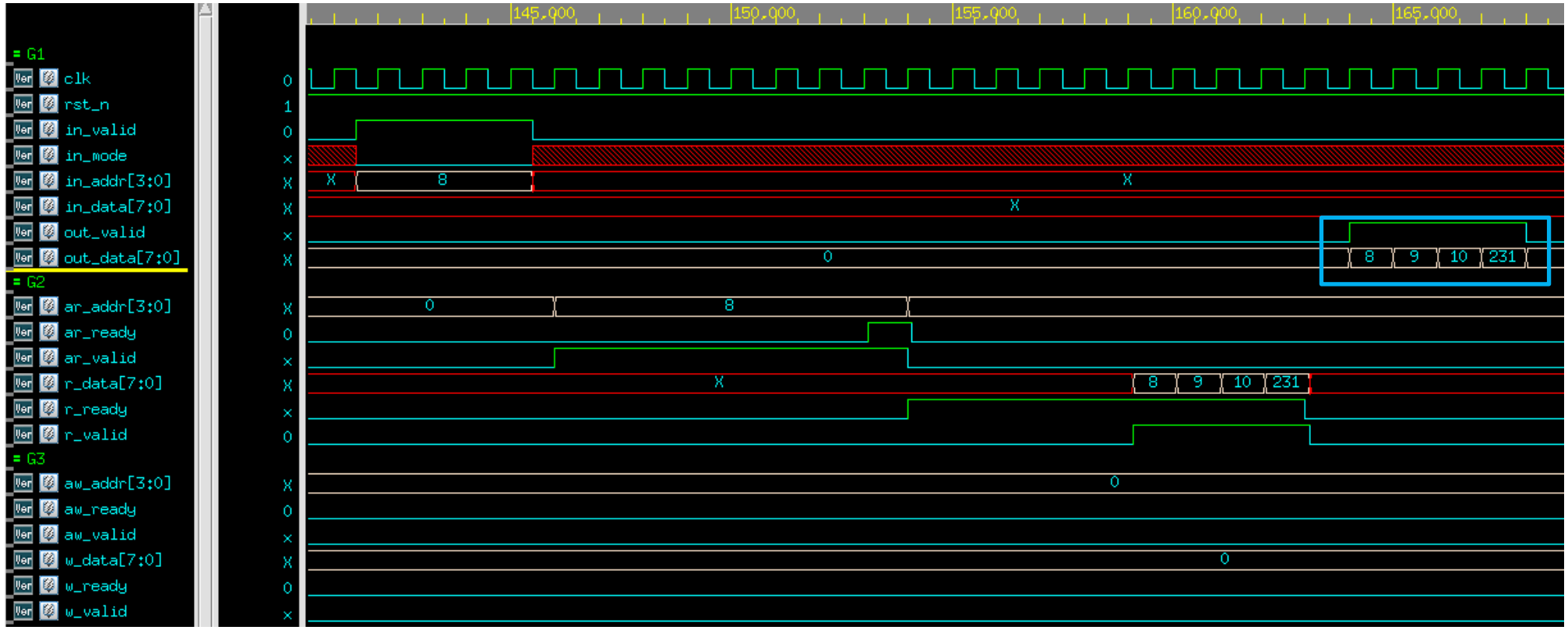
Waveform - read

- AR phase完成後，必須在30 cycle內拉起r_ready，準備接收資料
- 等待r_valid，當slave回應r_valid完成handshake，總共會handshake四次
- r_ready在handshake四次完成後必須馬上reset回0



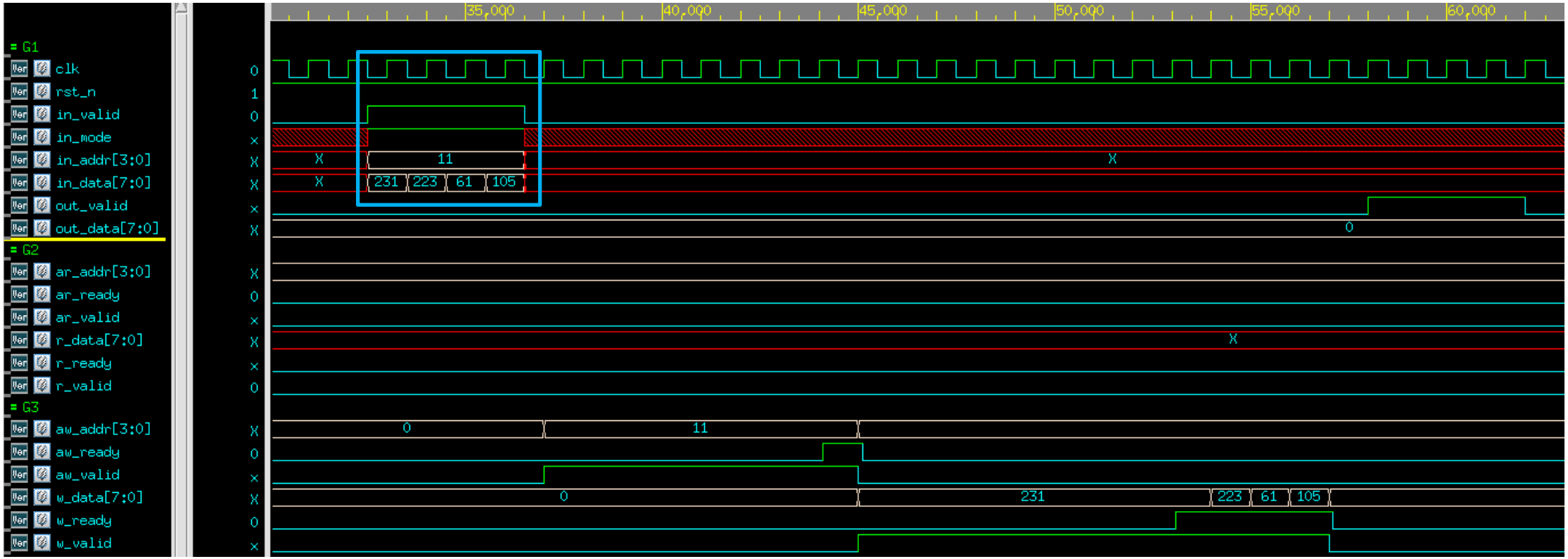
Waveform - read

- R phase完成後，必須在30 cycle內拉起out_valid輸出剛剛接收的資料
- 連續拉起out_valid 4 cycle依序輸出剛剛接收到的四個值
- out_valid, out_data在連續輸出4 cycle後必須馬上reset回0



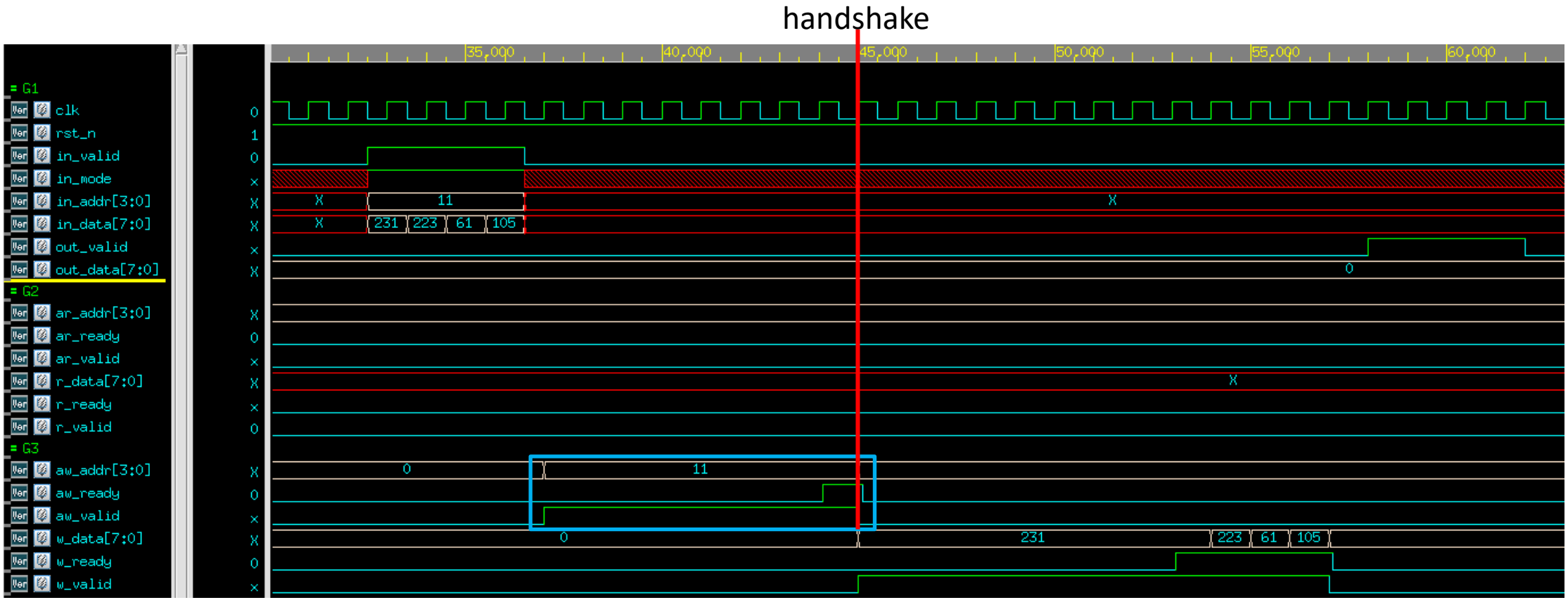
Waveform - write

- in_valid 4 cycles
- in_mode & in_addr會連續給予4 cycles，但值完全相同，所以實際上僅一筆輸入，因指令及位址只會有一個
- in_data會連續給予4個不同的值，必須依序傳送



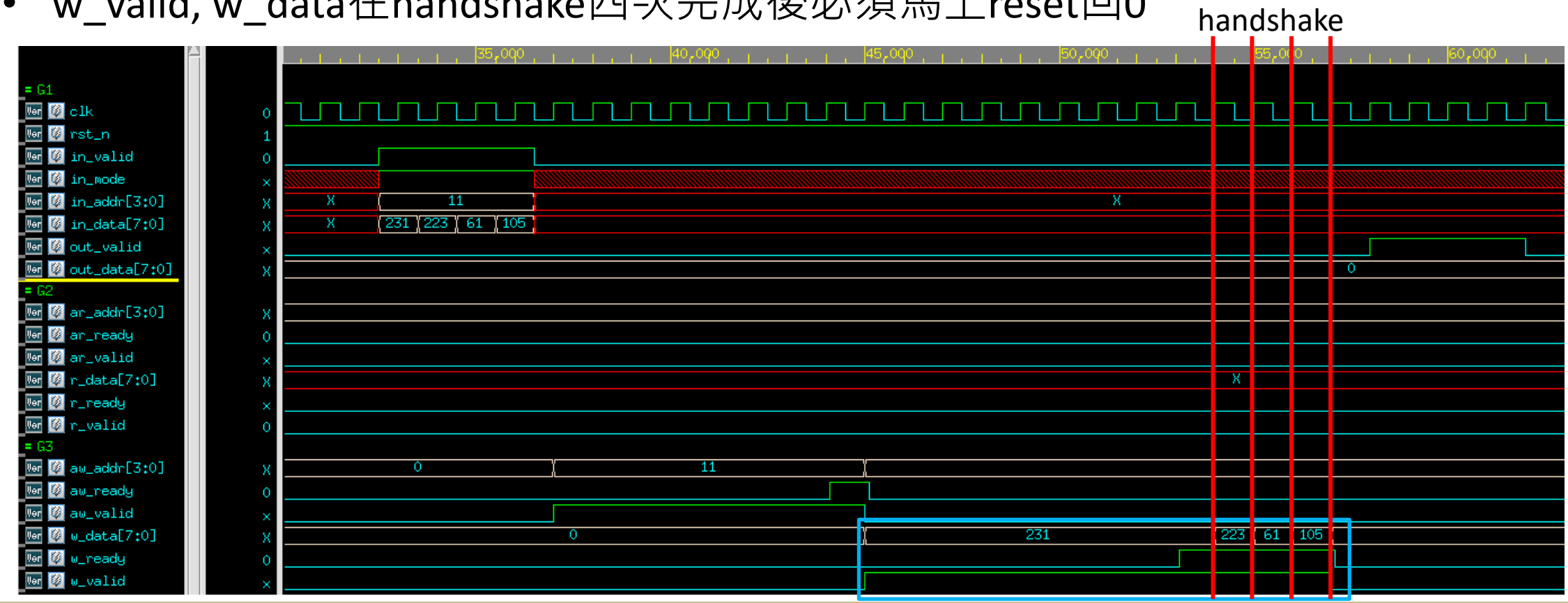
Waveform - write

- input完成後，必須在30 cycle內拉起aw_valid, aw_addr向slave確認addr
- 等待aw_ready，當slave回應aw_ready後完成handshake，進入W phase
- aw_valid, aw_addr在handshake完成後必須馬上reset回0



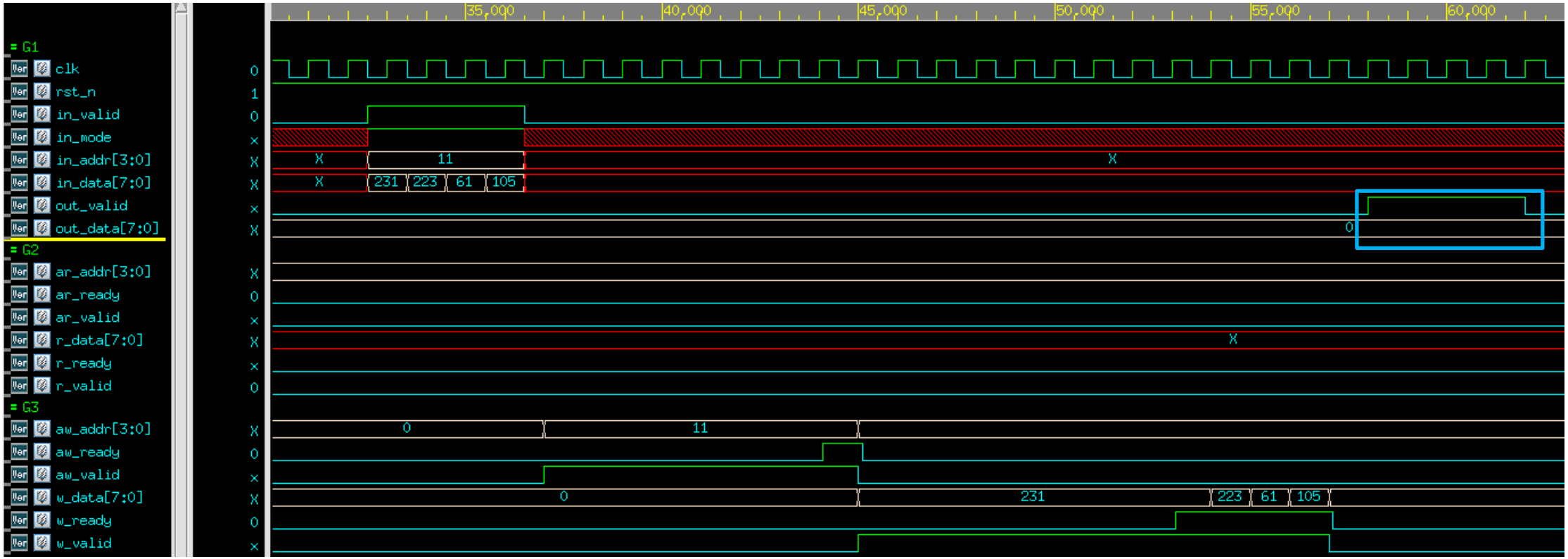
Waveform - write

- AW phase完成後，必須在30 cycle內拉起w_valid，準備傳遞資料
- 等待w_ready，當slave回應w_ready完成handshake，總共會handshake四次，將剛剛in_data輸入的資料依序傳輸
- w_valid, w_data在handshake四次完成後必須馬上reset回0



Waveform - write

- W phase完成後，必須在30 cycle內拉起out_valid確認完成傳輸
- 連續拉起out_valid 4 cycle，out_data在4個cycle都要輸出0
- out_valid, out_data在連續輸出4 cycle後必須馬上reset回0



Spec – pattern check (input, output)

- 所有output必須asynchoronous negedge reset
- in_valid拉下後30 cycles內必須進行handshake
- 在R,W的四次handshake完成前不能拉起out_valid
- R,W的四次handshake完成後30 cycles內必須拉起out_valid
- out_data只有在out_valid=1且mode是read情況才能有值，其他時候必須為0
- Read mode out_valid拉起時 out_data必須為讀出的值，且照順序(r_data)
- Write mode out_valid拉起時 out_data必須為0
- out_valid只能連續拉起4 cycle，完成後下一cycle必須reset回0

Spec – pattern check (Read process)

- In whole read process
  - Read mode時，aw_addr, aw_valid, w_valid, w_data都必須是0
- AR phase
  - ar_valid必須在in_valid拉下後30 cycles內拉起
  - ar_valid拉起時，ar_addr必須維持且正確
  - ar_valid拉起後，handshake前不能reset回0
  - AR handshake後，ar_addr和ar_valid必須reset回0
- R phase
  - r_ready必須在AR handshake後30 cycles拉起
  - r_ready拉起後，一次handshake前不能reset回0
  - r_ready和r_valid的handshake須確實完成4次
  - R handshake四次完成後，r_ready必須reset回0

Spec – pattern check (Write process)

- In whole write process
  - Write mode時，ar_addr, ar_valid, r_ready都必須是0
- AW phase
  - aw_valid必須在in_valid拉下後30 cycles內拉起
  - aw_valid拉起時，aw_addr必須維持且正確
  - aw_valid拉起後，handshake前不能reset回0
  - AW handshake後，aw_addr和aw_valid必須reset回0
- W phase
  - w_valid必須在AW handshake後30 cycles拉起
  - w_valid拉起後，一次handshake前不能reset回0
  - w_valid和w_ready的handshake須確實完成4次
  - 每次handshake時傳輸的data必須正確
  - W handshake四次完成後，w_valid必須reset回0

Spec – input given

- in_valid會有4 cycle，且會連續給予
- Read mode的ar_ready會在in_valid拉下後的1~9個posedge #(1)拉起
  - Ar_ready拉起後在handshak完前不會放下
- Read mode的r_valid會在ar_ready拉起後的1~9個posedge #(1)拉起
  - r_valid拉起後在四次handshake完前不會放下
- Write mode的aw_ready會在in_valid拉下後的1~9個posedge #(1)拉起
  - aw_ready拉起後在handshake完前不會放下
- Write mode的w_ready會在AW handshake後的1~9個posedge #(1)拉起
  - w_ready拉起後在四次handshake完前不會放下
- 下一筆測資的in_valid會在out_valid放下後1~6個negedge clk拉起
- 測資會在out_valid後才會給下一筆，不會有同時多筆或需要同時讀寫的情況
- 所有input和output都有0.5cycle的external delay，注意不要input接output

Fetch the files

- 在工作站輸入以下指令來取得檔案
- tar -xvf ~dcsTA01/Lab08.tar

Grading Policy

- Pass the RTL & Synthesis & Gate-level simulation: 100%
  - 合成結果**不能有Error、Timing report slack 必須是 MET、不能有Latch**
  - Gate-level simulation**不可以發生timing violation**
- Demo 2 打7折

```
Warning!  Timing violation
          $hold( posedge RN:119954 FS, posedge SN:204217 FS,  0.229292 : 229292 FS );
           File: /usr/cad/umc018/Verilog/umc18_neg.v, line = 9379
          Scope: TESTBED.I_Seq.lose_reg
           Time: 204217 FS
```

Upload

- 請將Lab08/01_RTL裡的INF.sv依以下命名規則重新命名後上傳至E3
- 命名規則：INF_dcsxxx.sv，xxx為工作站帳號號碼
- **命名錯誤扣5分!!!**

- **Deadline:**
  - **Demo 1: 5/1 17:25**
  - **Demo 2: 5/1 23:59**

Appendix

- 這次Lab為簡化題目在寫入有省略B部分，正常在AW和W後會有B phase，用於確認寫入是否成功，如果B_resp傳輸回來的是ERROR是需要重新handshake傳輸的
- 一次輸入多少個值實際上是可以調整的，稱為burst length，正式的AXI有ARLEN和AWLEN兩個訊號可以調整，在這次Lab為簡化就沒有要求各位調整，就限定4個值
- AXI是非對齊式的傳輸，雖然輸入的是一個位址，但多筆輸入的值儲存位置是會受到SIZE, BURST訊號影響，例如addr給的是1，有4筆，實際上data可能是儲存在address 1,2,3,4的位置，而不一定是在address 1儲存4筆資料

Reference

- https://developer.arm.com/documentation/ihi0022/latest/
  - AXI 的說明文件，如果對protocol有興趣可以看看 _(:3 」∠)_
  - 但我想285頁的說明文件沒人想看，我也不想
- https://developer.arm.com/documentation/ihi0033/latest/
  - AHB 的說明文件，比AXI簡單一些，同樣有興趣可以看看 _(:3 」∠)_