**Integrated Circuit Design**  
HW3 Image Display Control

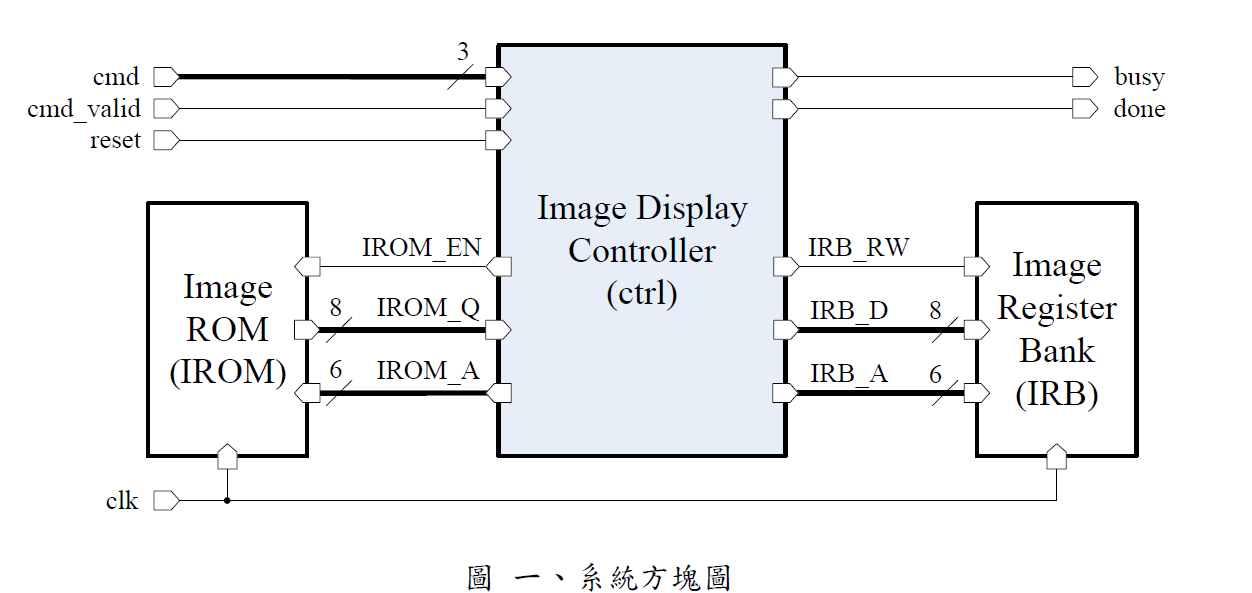
繳交期限4/24 13:00

**1.** 問題描述

請完成一影像顯示控制(Image Display Control)電路設計。此控制電路，可依指定之操控指令，使顯示端的影像進行影像平均(Average)、X軸及Y軸鏡像(Mirror )與水平及垂直方向的平移(Shift)功能。本控制電路有 5 只信號輸入(cmd，cmd\_valid，IROM\_Q，clk，reset)及 7 只信號輸出(IROM\_EN，IROM\_A， IRB\_RW，IRB\_D，IRB\_A，busy，done) ，關於各輸入輸出信號的功能說明，請參考表一。

**2.** 設計規格

**2.1** 系統方塊圖



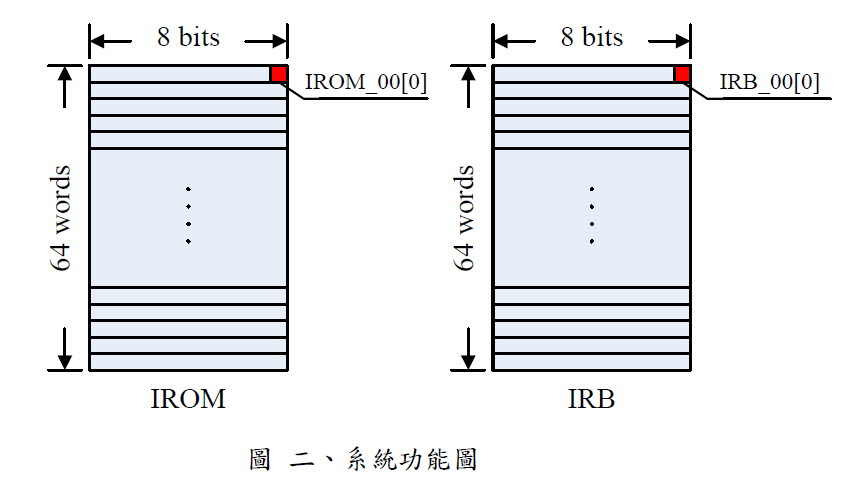
**2.2** 輸出**/**輸入介面

|  |  |  |  |
| --- | --- | --- | --- |
| 信號名稱 | 輸出/入 | 位元寬度 | 說明 |
| *reset* | input | 1 | 高位準非同步(active high asynchronous)之系統重置信號。  說明:本信號應於系統啟動時送出。 |
| *clk* | input | 1 | 時脈信號。  說明:此系統為同步於時脈正緣(posedge)之同步設計。 |
| *cmd* | input | 3 | 指令輸入信號。  說明:本控制器共有八種指令輸入，相關指令說明請參考表二。指令輸入只有在cmd\_valid為high及busy為low時為有效指令 |
| *cmd\_valid* | input | 1 | 讀寫控制訊號。  說明:當本信號為high時表示cmd指令為有效指令輸入。 |
| *IROM\_Q* | input | 8 | Image ROM八位元資料輸出埠。 |
| *IROM\_A* | output | 6 | IROM六位元位址信號。 |
| *IROM\_EN* | output | 1 | IROM致能控制訊號。  說明:當本信號為low時，表示是將啟動IROM進行讀取；信號為high時，表示將關閉IROM。 |
| *busy* | output | 1 | 系統忙碌訊號。  說明:當本信號為high時，表示此控制器正在執行現行指令，而無法接受其他新的指令輸入；當本信號為low時，系統會開始輸入指令。reset時，default設定為high。 |
| *done* | output | 1 | 當控制器完成寫入IRB時，將done設為high表示完成。 |
| *IRB\_A* | output | 6 | IRB六位元位址信號。 |
| *IRB\_D* | output | 8 | IRB八位元資料輸入埠。 |
| *IRB\_RW* | output | 1 | IRB讀寫控制訊號。(本次作業只有使用寫入功能) 說明:當本信號為low時，表示是將啟動IRB進行寫入。 |

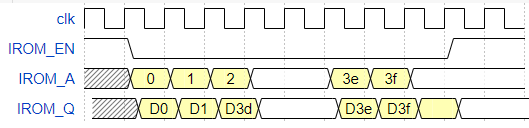
表一、輸入輸出信號功能說明

**2.2.1 Image ROM(IROM)**與**Image Register Bank(IRB)**規格描述

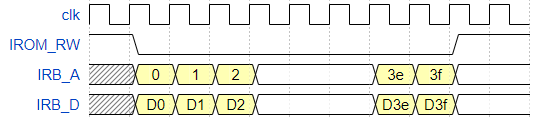
本題使用到兩個記憶體模組，一為唯讀記憶體(ROM)格式，另一為單埠Register file格式。Image ROM(IROM)模組的記憶體寬度為8位元，而記憶體深度為64個word。

Image Register Bank(IRB)模組的記憶體寬度為8位元，而記憶體深度也為64個word。如圖二所示。

圖二、IROM、IRB規格圖

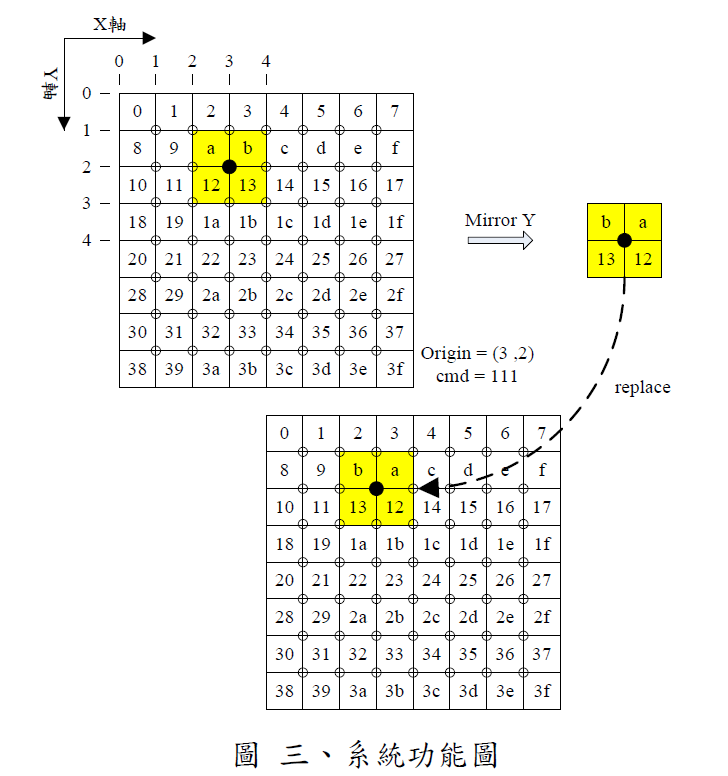
IROM操作方式如圖三所示，當IROM\_EN為low時，表示啟動IROM，即可輸入位址信號IROM\_A，經過一小段delay後，便可取得指定位址的資料IROM\_Q。

圖三、IROM功能圖

IRB操作方式如圖四所示，當IRB\_RW為low時，表示對IRB寫入，即可輸入位址信號RB\_A，並將資料IRB\_D寫入IRB的指定位址。

圖四、IRB功能圖

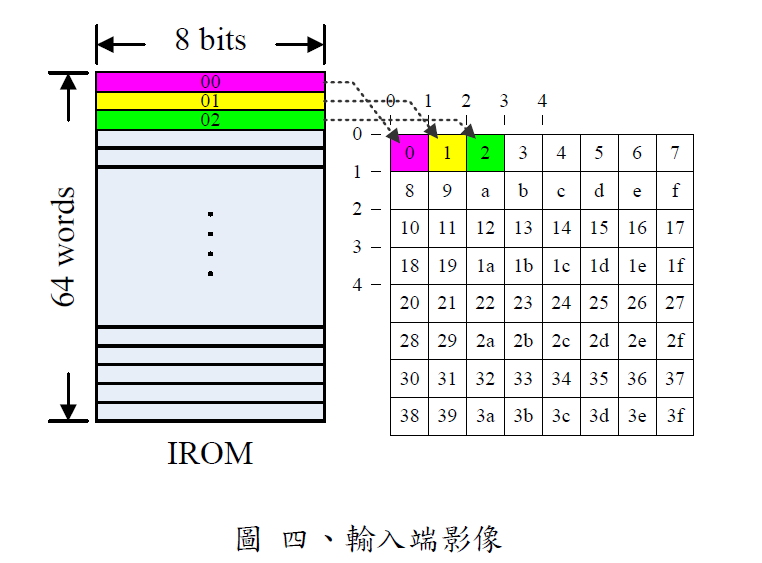
**2.3** 系統功能描述

當reset結束後，影像顯示控制器之輸入端從IROM讀取一張8x8大小的影像。資料影像顯示控制器必須處理使用者輸入之指令，取得顯示相關之座標(origin)及資料參數，使得顯示端達到平均、平移以及鏡像功能，並將經過指令處理完的影像資料寫入IRB，如圖五所示。

圖五、系統功能圖

**2.3.1** 輸入與輸出端之影像及參數規範

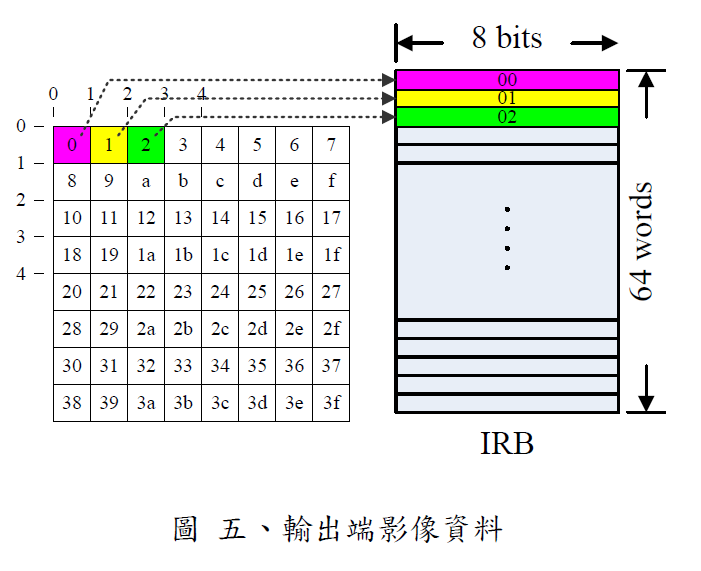
[影像輸入]

輸入端影像資料存至IROM。此影像為8x8共64筆測試樣本，每筆樣本為8位元資料，並且依左而右；由上而下，同學必須由IROM讀取影像資料，並且依照左而右、由上而下存至影像控制電路中。(如圖六所示，輸入的順序為 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, 10, … , 3d, 3e, 3f)

圖六、輸入端影像

[影像輸出]

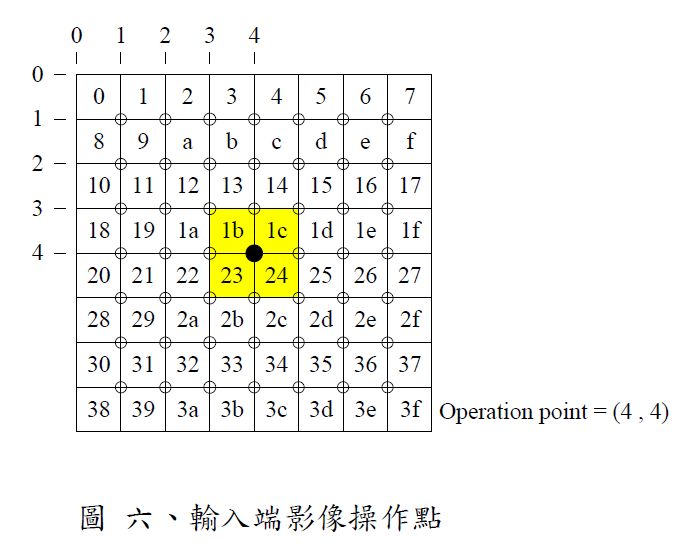
輸出端影像為8x8，共64筆樣本輸出，每筆樣本為8位元資料。並且依照由左而右、由上而下、序列(Serial)的方式循序寫入IRB內。(如圖七所示)。

註: 以下僅為圖例示範，詳細輸入影像值未必如下圖七所示。

圖七、輸出端影像資料

[操作點]

操作點(operation point)指的是在影像資料的座標點，在操作點的上下左右四格為操作影像資料，控制器將使用操作影像資料來運算。本題已定義輸入端影像之座標軸。輸入端影像之水平方向為X軸，垂直方向為Y軸。此外，X軸與Y軸座標範圍為0~+8。(如圖六所示，為確保操作影像資料不超過對應輸入影像邊界，因此限制原點之X軸與Y軸範圍最大為+1~+7)。同學須根據此座標軸，進行顯示端的畫面，進行顯示端的畫面平移(Shift)功能設計。

註: 本題規定讀入控制器後影像資料初始操作點座標為**(4, 4)**，座標圖方向如下圖八所示。

圖八、輸入端影像操作點

**2.3.2** 影像顯示控制器功能規範

[指令定義]

影像控制器電路控制指令。輸入指令(cmd)所對應之功能如表二所示。

表二、控制指令定義

●寫入(Write)

》當執行寫入(Write)指令時，控制器會依照由左而右、由上而下的順序將影像資料寫入IRB。

●畫面上移(Shift Up)

》上移顯示區塊。執行此Shift Up指令，將使操作點的Y減少1，但Y軸座標最小不可低於1。

》當Y座標等於1時，倘若再收到上移指令，則Y軸座標將仍維持為1，操作點維持不變。

●畫面下移(Shift Down)

》下移顯示區塊。執行此Shift Down指令，將使操作點的Y軸增加1，但Y軸座標最大不可大於7。

》當Y座標等於7時，倘若再收到下移指令，則Y軸座標將仍維持為7，操作點維持不變。

●畫面左移(Shift Left)

》左移顯示區塊。執行此Shift Left指令，將使操作點的X軸刪減1，但X軸座標最小不可低於1。

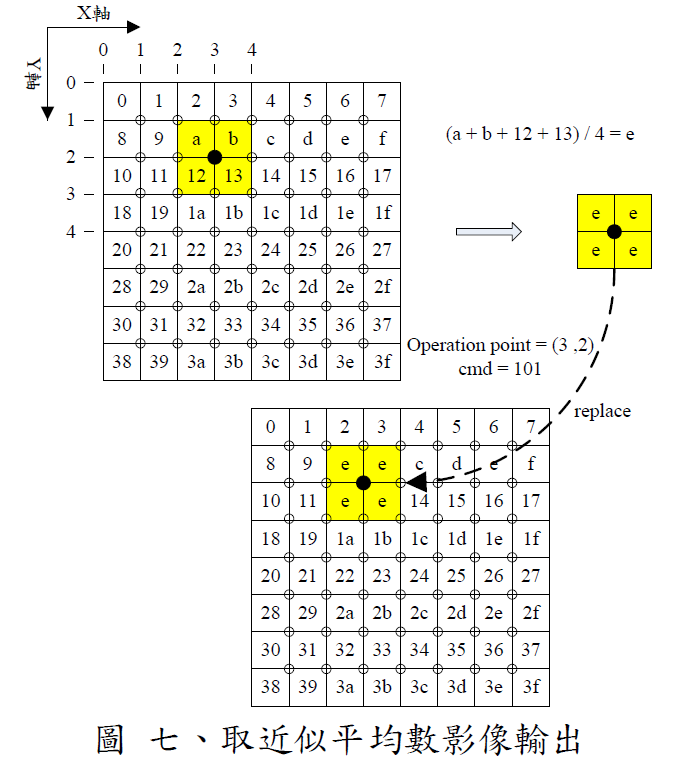
》當X座標等於1時，倘若再收到左移指令，則X軸座標將仍維持為1，操作點維持不變。

●畫面右移(Shift Right)

》右移顯示區塊。執行此Shift Right指令，將使操作點的X軸增加1，但X軸座標最小不可大於7。

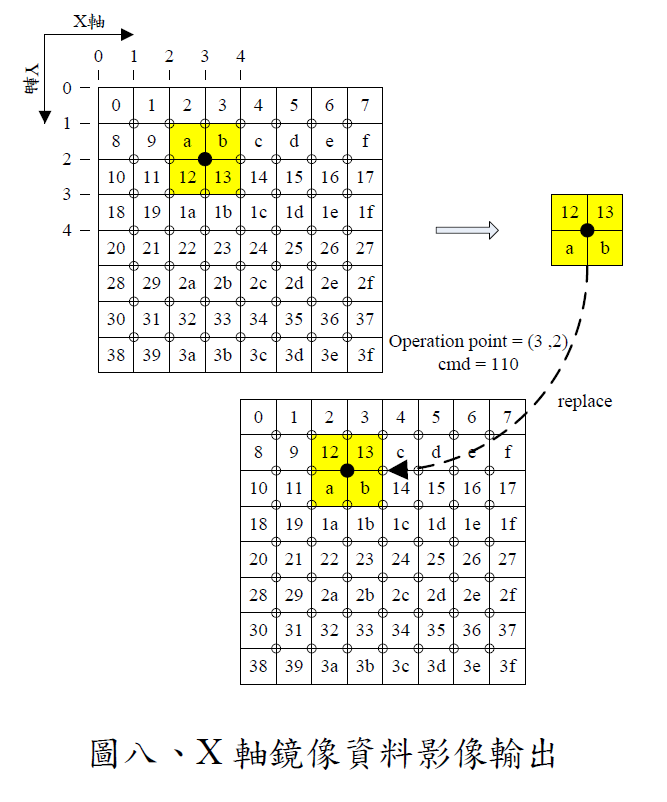
》當X座標等於7時，倘若再收到右移指令，則X軸座標將仍維持為7，操作點維持不變。

●影像資料平均(Average)

》當執行平均(Average)指令時，將執行目前操作點座標之影像資料取近似平均數之計算，即將目前座標所對應的4筆影像資料相加之後再除以4，當有小數點時則以無條件捨去法處理(例如(a+b+12+13)/4=14.5，即輸出14)，輸出影像資料4筆皆輸出計算後之近似平均數，並改變原始影像資料。如圖九所示。

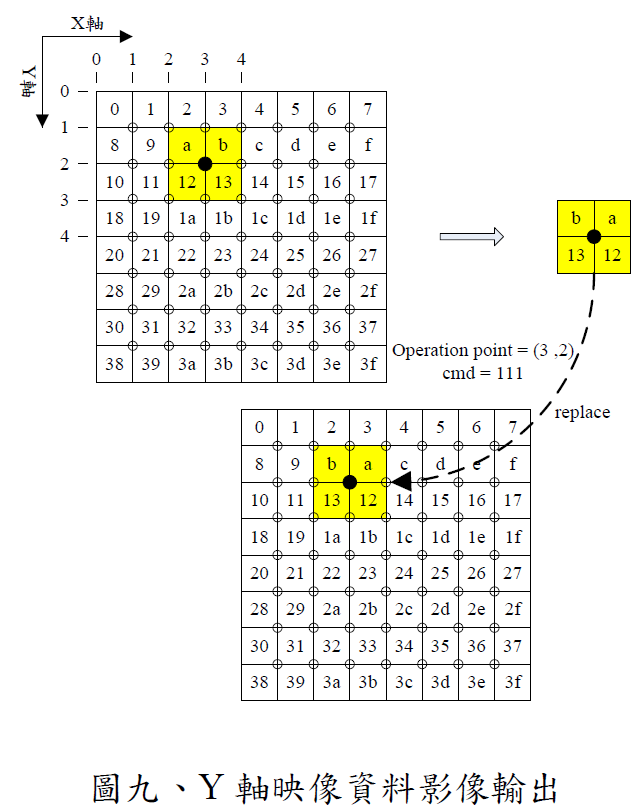
圖九、取近似平均數影像輸出

●影像資料X軸鏡像(Mirror X)

》當執行X軸鏡像(Mirror X)指令時，將輸出目前操作點座標之影像資料皆以目前座標對X軸翻轉，並改變原始影像資料。如圖十所示。

圖十、X軸鏡像資料影像輸出

●影像資料Y軸鏡像(Mirror Y)

》當執行Y軸鏡像(Mirror Y)指令時，將輸出目前操作點座標之影像資料皆以目前座標對Y軸翻轉，並改變原始影像資料。如圖十一所示。

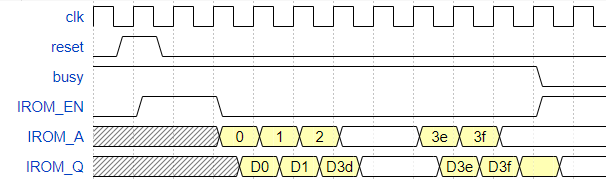
圖十一、Y軸鏡像資料影像輸出

**2.4** 時序規格圖

●重置(Reset)後之時序規格圖，如圖十二所示。

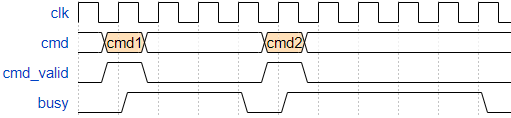
》在電路重置(Reset)之後，控制器將會由IROM讀取64筆影像資料。

》當IROM\_EN為low時，表示啟動IROM，即可輸入位址信號讀取IROM內的影像資料。

》在整個處理過程中，busy皆維持為high。並在讀取完成後，將busy設回low以接受新指令輸入。

圖十二、資料讀取之時序規格圖

●其他控制指令(average、shift up、shift down、shift left、shift right、mirror x、mirror y)之時序規格圖，如圖十三所示。

》在整個處理過程中，busy皆維持為high。並在輸出完成後，將busy設回low以接受新指令輸入

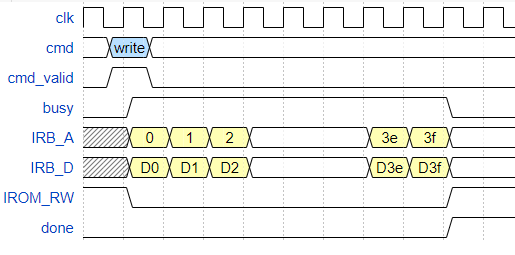
圖十三、其他控制指令之時序規格圖

●寫入指令(write)之時序規格圖，如圖十四所示。

》執行寫入指令時控制器會將處理完的影像資料寫入IRB。

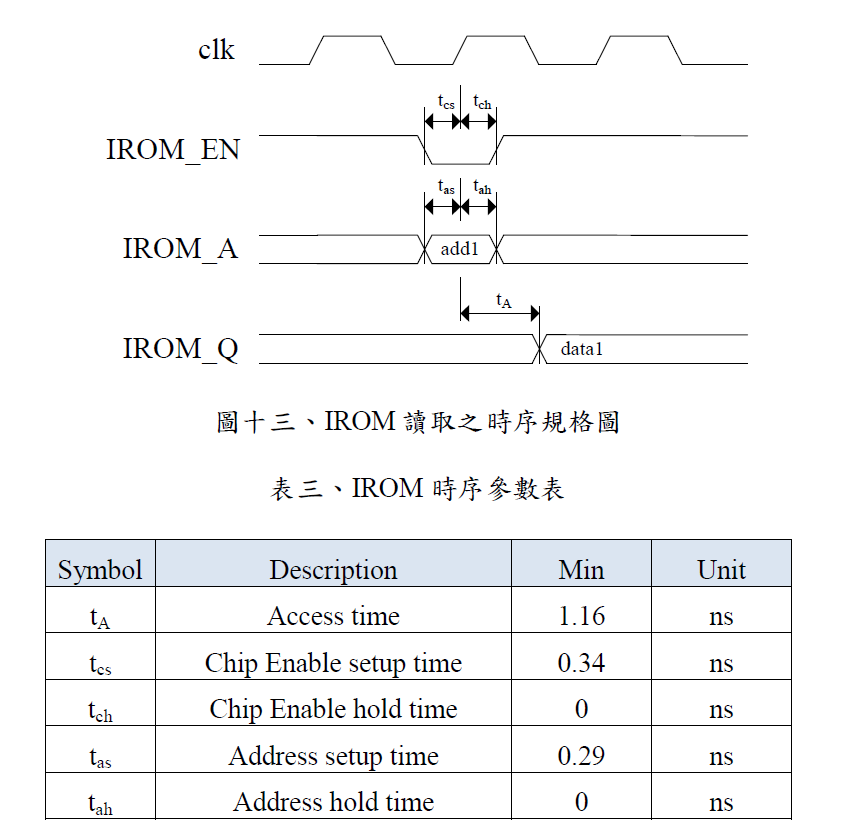
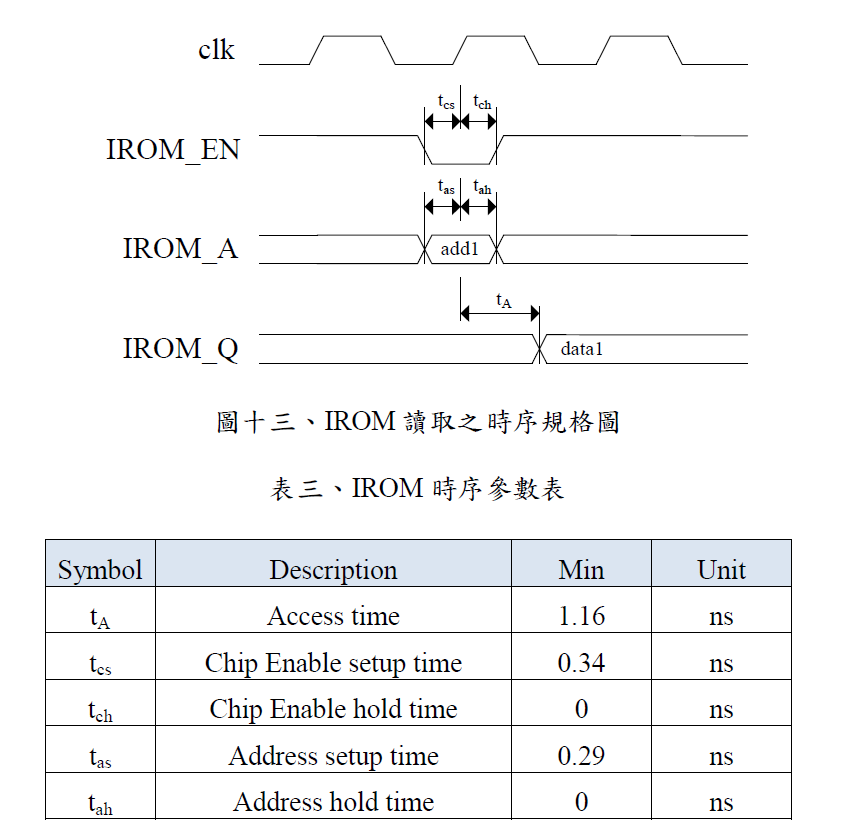
》當IRB\_RW為low時，表示對IRB寫入，即可輸入位址信號將影像資料寫入IRB。

》在整個處理過程中，busy皆維持為high。並在輸出完成後，將busy設回low以接受新指令輸入。

》寫入完成後，並將done信號設為high，表示寫入完成，此時testfixture會拿寫入IRB的資料與golden pattern比對。

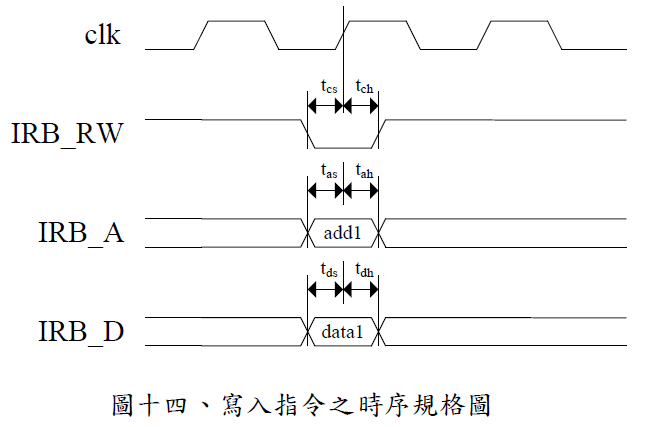
圖十四、寫入指令之時序規格圖

**2.5 IROM**與**IRB**之時序規格

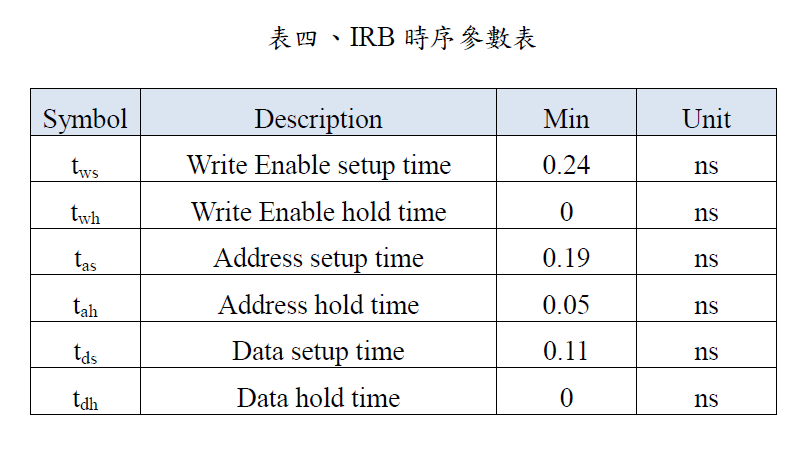
IROM 讀取動作主要是以IROM\_EN 啟動IROM 後，輸入address 讀取資料，其資料讀取之波形時序圖，如圖十五所示。特別注意若沒有要進行讀取時，請將IROM\_EN 保持為high。最後，時序規格數值整理於表三。

圖十五、IROM讀取之時序規格圖

表三、IROM時序參數表

IRB 讀取動作主要是以IRB\_RW 這個信號來控制，其資料讀取之波形時序圖，如圖十六所示。特別注意若沒有要進行寫入時，請將IRB\_RW 保持為low。最後，時序數值整理於表四。

圖十六、IRB寫入指令之時序規格圖

**表三、IRB時序參數表

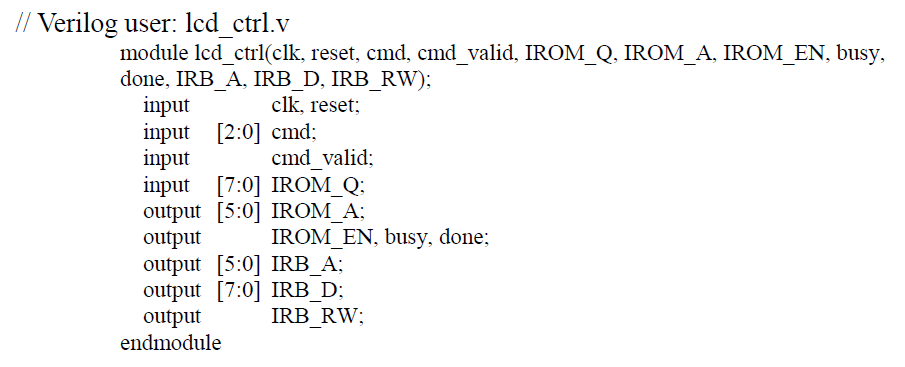
附錄

***附錄A 設計檔案說明***

1. 以下表六為設計檔案

表六、設計檔案

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | **檔名** | | |  | | --- | | **說明** | |
| testfixture .v | 測試樣本檔(testbench)。此測試樣本檔定義了時脈週期與測試樣本之輸入信號，module名稱為test |
| lcd\_ctrl.v | 影像顯示控制器lcd\_ctrl的設計檔範本，已包含系統輸/出入埠之宣告  注意!!!繳交之檔案，包含檔名、top module name、port name皆不能更改，在評分時因為檔名或是top module name及port name不同之問題而無法模擬，將視為設計錯誤 |
| IROM.v | ROM模擬檔 (simulation model) |
| IRB.v | Register Bank模擬檔 (simulation model) |
| cmd1.dat | 第一組測試樣本之指令 |
| cmd2.dat | 第二組測試樣本之指令 |
| image1.dat | 第一組測試樣本輸入檔案 |
| tb1\_goal.dat | 第二組測試樣本之IRB比對檔 |
| image2.dat | 第一組測試樣本輸入檔案 |
| tb2\_goal.dat | 第二組測試樣本之IRB比對檔 |

2. 請使用***lcd\_ctrl.v***，進行影像顯示控制電路之設計。其模組名稱、輸出/入埠宣告如下列範本所示︰

3.測試樣本可依下面範例來進行模擬：

* ncverilog 指令範例如下：

*ncverilog testfixture.v lcd\_ctrl.v +define+tb1*

* 上述指令中*+define+tb1* 指的是使用第一組測試樣本模擬，若須使用其它測試樣本請自行修改此參數。以第二組測試樣本為例：*+define+tb2*。
* 關於模擬時使用的一些記憶體，因已經以include方式加在testfixture.v 裡，所以不需加在模擬指令裏。
* 若RTL模擬時，要避免時序檢查以減少錯誤訊息，可於模擬指令中加入*+notimingchecks*

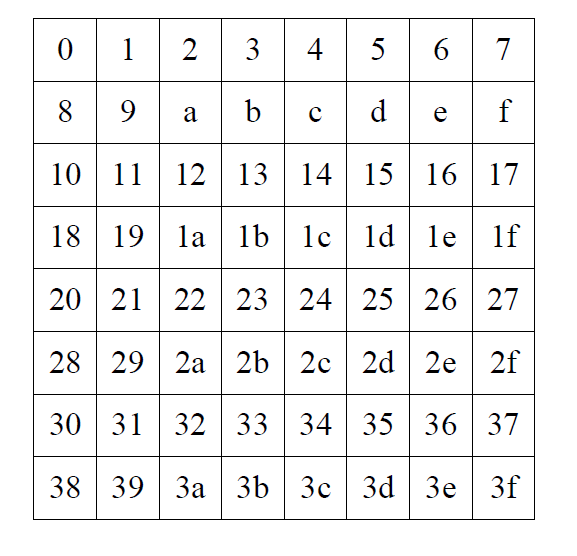
範例如：***ncverilog testfixture.v lcd\_ctrl.v +define+tb1 +notimingchecks***

附錄***B*** 測試樣本

本次作業提供兩組測試樣本，為方便設計者除錯之用，將測試樣本之影像資料及指令輸入詳列如下 :

◎測試樣本一(tb1)

●相關資料：image1.dat, cmd1.dat, tb1\_goal.dat

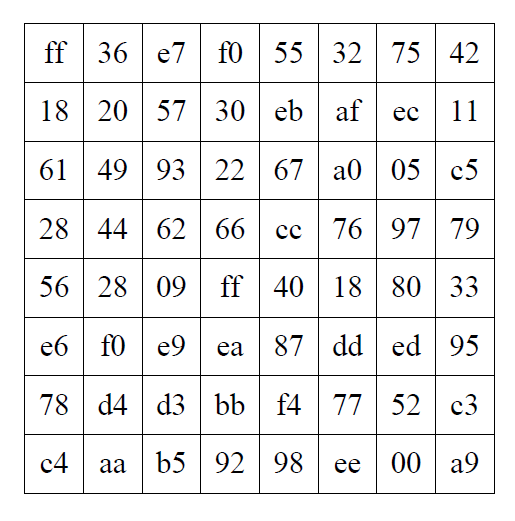
●影像資料：

■ 指令輸入順序：

shift up, average, shift left, mirror x, shift down, shift right, mirror y ,write

◎測試樣本二(tb2)

●相關資料：image2.dat, cmd2.dat, tb2\_goal.dat

●影像資料：

■ 指令輸入順序：

shift up ,shift up ,mirror y ,shift up ,mirror x ,shift up ,shift left ,shift down ,shift left ,average ,shift left

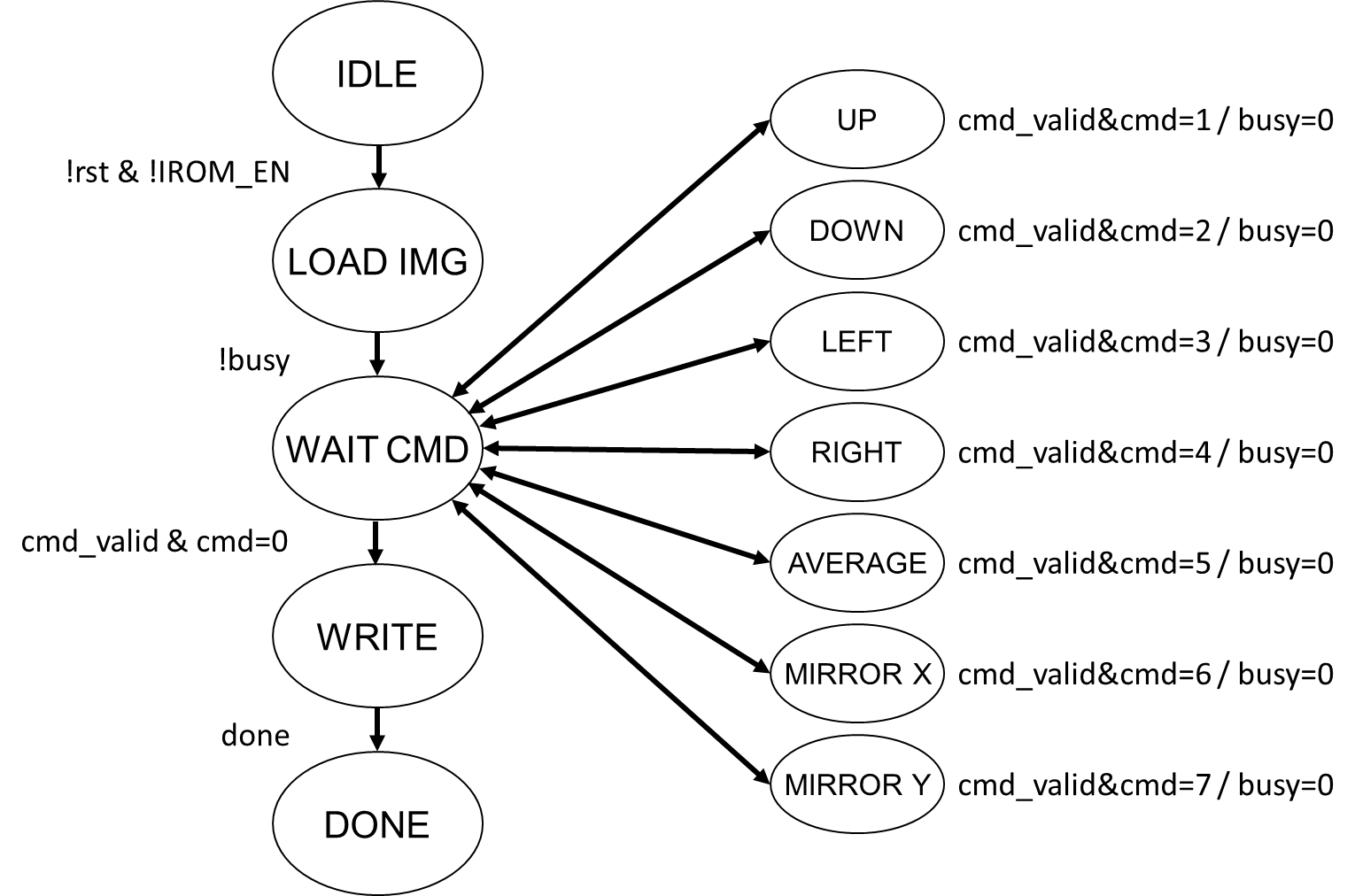
,mirror y ,shift down ,shift down ,mirror x ,shift right ,average ,shift down ,mirror y ,shift down ,mirror x

,shift right ,average ,shift right ,mirror y ,shift right ,mirror x ,shift down ,average ,shift right ,mirror

x ,shift right ,mirror y ,shift up ,average ,shift up ,shift up ,shift left ,mirror x ,shift up ,average ,mirror y

,shift right ,mirror y ,write

**附錄*C 有限狀態機　Finite-state machine***

同學可參考圖十七，也可以自行設計Finite state machine。

圖十七、有限狀態機

**附錄Ｄ繳交檔案**

|  |  |  |
| --- | --- | --- |
| ***Design Stage*** | ***File*** | ***Description*** |
| RTL Simulation | lcd\_ctrl.v | Verilog synthesizable RTL code |
| Report | StudentID.pdf | Design report |

繳交期限4/24 13:00