專案(一)

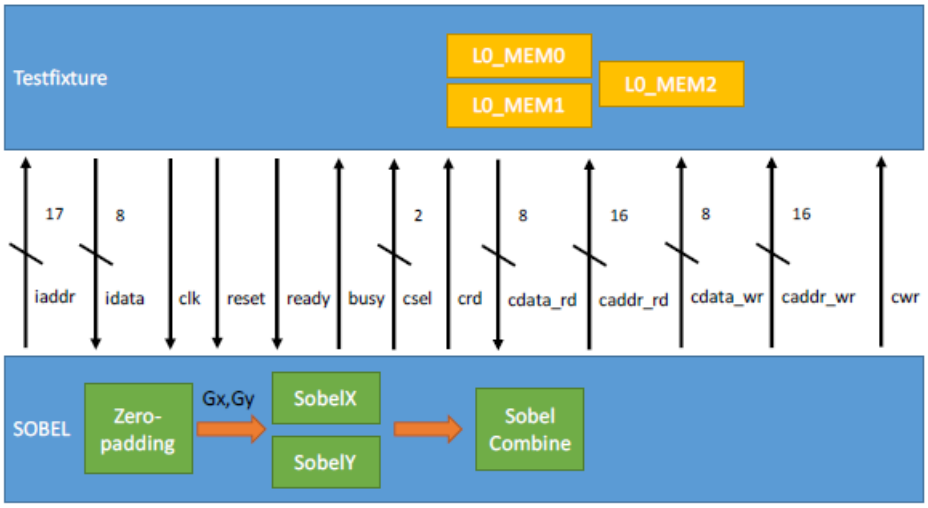
專案名稱 - Sobel operator 邊緣偵測 IP 實現

簡介

索伯算子（Sobel operator）是圖像處理中的算子之一，有時又稱為索伯-費德曼算子或索貝濾波器，在影像處理及電腦視覺領域中常被用來做邊緣檢測。在技術上，它是一離散性差分算子，用來運算圖像亮度函數的梯度之近似值。在圖像的任何一點使用此算子，索伯算子的運算將會產生對應的梯度向量或是其範數。概念上，索伯算子就是一個小且是整數的濾波器對整張影像在水平及垂直方向上做捲積，因此它所需的運算資源相對較少，另一方面，對於影像中的頻率變化較高的地方，它所得的梯度之近似值也比較粗糙。此次作業請實作一圖像邊緣偵測系統，利用 Gx和Gy對圖像進行捲積得出sobelX圖像及sobelY圖像，再利用得出的sobelX圖像及sobelY圖像相加除以二得出sobelCombine的圖像。

設計規格

Block overview



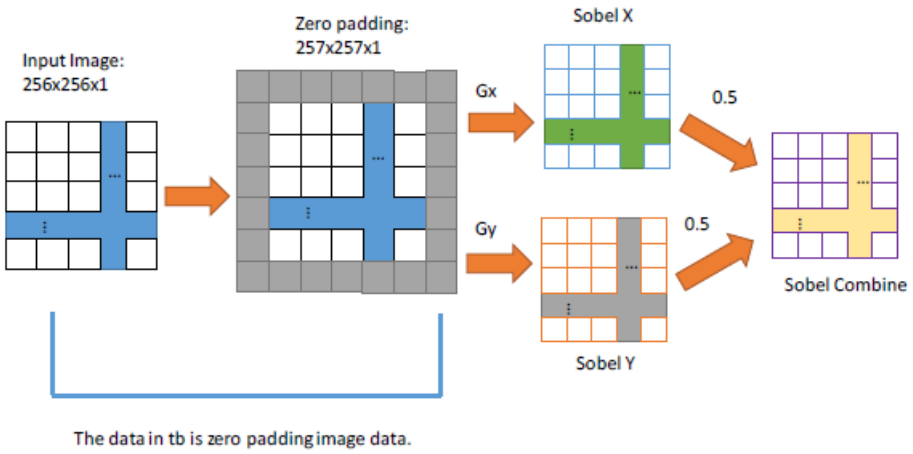
圖一、系統方塊圖

I/O Interface

Name	I/O	Width	Description
clk	I	1	系統時脈訊號。本系統為同步於時脈正緣之同步設計。
reset	I	1	高位準"非"同步(active high asynchronous)之 系統重置信號
ready	I	1	灰階圖像準備完成指示訊號。當訊號為 High 時，表示灰階 圖像準備完成，此時 SOBEL 才可以開始向 testfixture 發送 輸入灰階圖 像資料索取位址。
busy	O	1	系統忙碌指示訊號。當 SOBEL 接收到 ready 訊號為 High，且 SOBEL 準備開始動作時， 需將此訊號設為 High，表示準備開始進行輸 入灰階圖像資料索取；待所有運算處理完成 且輸出結果寫回 testfixture 後，需再將訊號 設為 Low 表示 動作結束。
iaddr	O	17	輸入灰階圖像位址訊號。指示欲索取哪個灰 階圖像像素 (pixel)資料的位址。
idata	I	8	輸入灰階圖像像素資料訊號，由 8bits 整數 組成，為無號數。testfixture 將 iaddr 所指示 的位址之像素資料用此訊號送給 SOBEL。
crd	O	1	SOBEL 運算輸出記憶體讀取致能訊號。當時 脈正緣觸發 時，若此訊號為 High，表示要 進行讀取動作。testfixture 會將 caddr_rd 位址 指示之資料讀取到 cdata_rd 上。
cdata_rd	I	8	CONV 運算結果記憶體讀取訊號，由 8 bits 整數(MSB)組成，為無號數。testfixture 將記 憶體資料傳送至 SOBEL 電路。
caddr_rd	O	16	SOBEL 運算結果記憶體讀取位址。 SOBEL 電路各層的運算 結果利用此訊號 指示將要讀取 testfixture 中所內建輸出結果 之記憶體的哪個位址。
cwr	O	1	SOBEL 運算輸出記憶體寫入致能訊號。當時 脈正緣觸發時，若此訊號為 High，表示要進 行寫入動作。testfixture 會將 cdata_wr 內容 寫到 caddr_wr 所指示之位址。
cdata_wr	O	8	SOBEL 運算結果記憶體寫出訊號，由 8bits 整數(MSB)組成，為無號數。SOBEL 電路的

			運算結果利用此訊號輸出至 testfixture。
caddr_wr	O	16	SOBEL 運算結果記憶體寫入位址。SOBEL 電路的運算 結果利用此訊號指示將要寫入到 testfixture 中所內建輸出結果之記憶體的哪個位址。
csel	O	2	SOBLE 運算處理結果寫入/讀取記憶體選擇訊號。此訊號指示目前寫入/讀取資料為 SOBEL 電路中哪一個的運算結果。說明如下: 2'b00:表示沒有選擇記憶體。 2'b01: 寫入/讀取 Sobel X 結果。 2'b10: 寫入/讀取 Sobel Y 結果。 2'b11:寫入/讀取 Sobel combine 結果。

Function Description



$$\mathbf{G_x} = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \quad \mathbf{G_y} = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

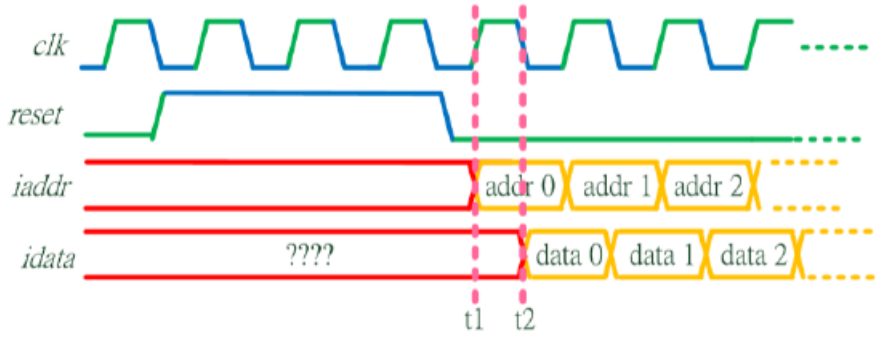
圖二、運算流程圖

本系統的輸入圖片大小為 256x256 存放於 testfixture 的記憶體中，灰階圖像各 pixels 與其記憶體的對應方式如下圖四.說明。動作時序上 SOBEL 電路需利用 iaddr 發送欲索取圖像資料的位址到 testfixture(如圖三 t1 時間點), testfixture 在每個時脈負緣後會將 iaddr 所指示位址之 pixel 資料利用 idata 送入 SOBEL 電路(如圖三 t2 時間點)。

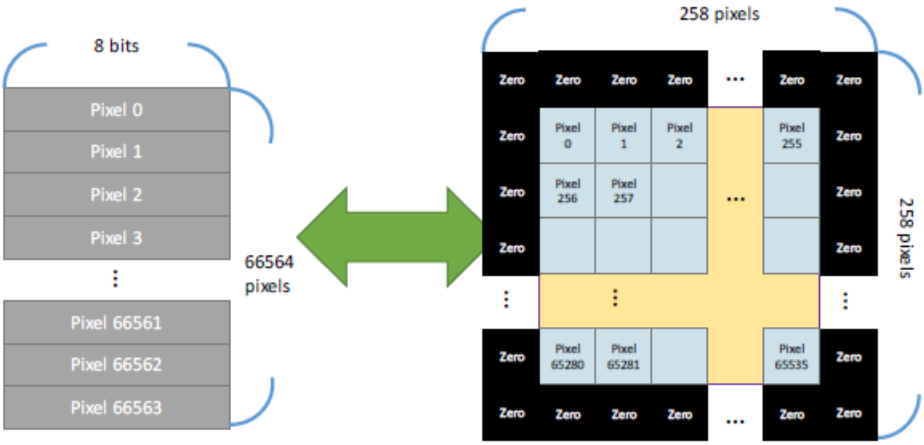
本系統已經將 zero_padding 後的資料存於記憶體中，利用 Gx、Gy 分別作捲積，得出 Sobel X 及 Sobel Y 的圖，在做 Sobel 運算時，若值超過 255 就將其設定為 255, 若小於 0 就將其值設定為 0, 得出 Sobel X 及 Sobel Y 的圖後, 利用(sobel X + sobel Y)/2 然後四捨五入的方式得到 Sobel combine，當結束運算，將 busy 訊號拉為 0，之後會開始驗證答案是否正確。

本系統的記憶體存取方式，各層輸出資料記憶體 L0_MEM0、L0_MEM1、L0_MEM2 皆為 RAM model 且控制方式及時序皆相同，都可進行寫入及讀取動作。採用不同的 csel 設定值啟動各層輸出相對應的記憶體，使用 cwr 作為寫入致能訊號, crd 作為讀取致能訊號。讀取時, 使用 caddr_rd 為記憶體位址, cdata_rd 作為讀取資料訊號。

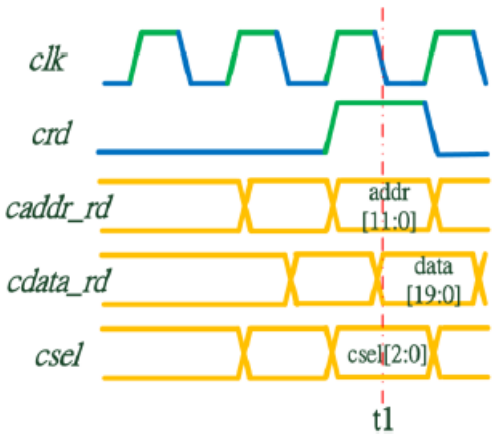
動作時序如下圖五說明，當時脈正緣觸發時若 crd 為 High, 則會在觸發後立刻將 caddr_rd 所指示位址的資料讀取到 cdata_rd 上(如圖五 t1 時間點)。寫入時，使用 caddr_wr 為記憶體位址，cdata_wr 作為寫入資料訊號。動作時序如下圖六說明，當時時脈正緣觸發時若 cwr 為 High，則會將這時 cdata_wr 的資料寫入到 caddr_wr 所指示位址上(如圖六 t1 時間點)。最終將 SobelX 的資料存入記憶體 L0_MEM0（2'b01），SobelY 的資料存入記憶體 L0_MEM1（2'b10 ），Sobel Combine 的資料存入記憶體 L0_MEM2（2'b11）。



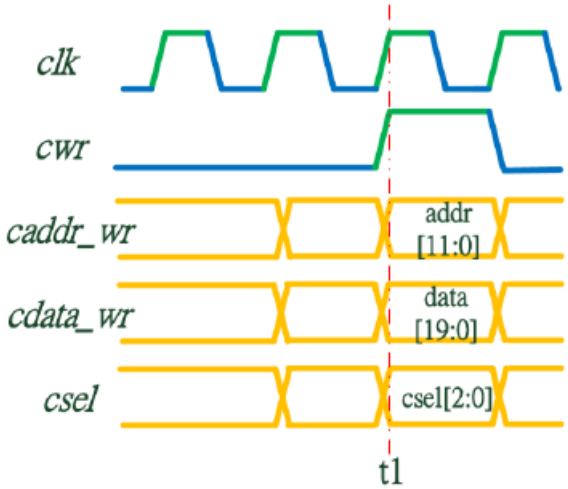
圖三、灰階圖像記憶體時序圖



圖四、圖像對應記憶體位址



圖五、輸出資料記憶體 L0_MEM0、L0_MEM1、L0_MEM2 讀取動作時序圖



圖六、輸出資料記憶體 L0_MEM0、L0_MEM1、L0_MEM2 寫入動作時序圖

Result

Gate-Level simulation

```
-------------------------------------------------------

-------------------- S U M M A R Y ----------------

Congratulations! Sobel X data have been generated successfully! The result is PASS!!

Congratulations! Sobel Y data have been generated successfully! The result is PASS!!

Congratulations! Sobel combine data have been generated successfully! The result is PASS!!

-------------------------------------------------------

** Note: $finish    : E:/ModelSim projects/DIC_HW5/testfixture.v(198)
   Time: 24772779922 ps  Iteration: 0  Instance: /testfixture
```

Synthesis result

Flow Summary

Flow Status	Successful - Sun Jun 28 14:13:30 2020
Quartus II Version	10.0 Build 262 08/18/2010 SP 1 SJ Full Version
Revision Name	SOBEL
Top-level Entity Name	SOBEL
Family	Cyclone II
Device	EP2C70F896C8
Timing Models	Final
Met timing requirements	N/A
Total logic elements	492 / 68,416 (< 1 %)
Total combinational functions	476 / 68,416 (< 1 %)
Dedicated logic registers	125 / 68,416 (< 1 %)
Total registers	125
Total pins	81 / 622 (13 %)
Total virtual pins	0
Total memory bits	0 / 1,152,000 (0 %)
Embedded Multiplier 9-bit elements	0 / 300 (0 %)
Total PLLs	0 / 4 (0 %)