

VLSI System Design and Implementation

Midterm Project Report

Student ID: 105062635

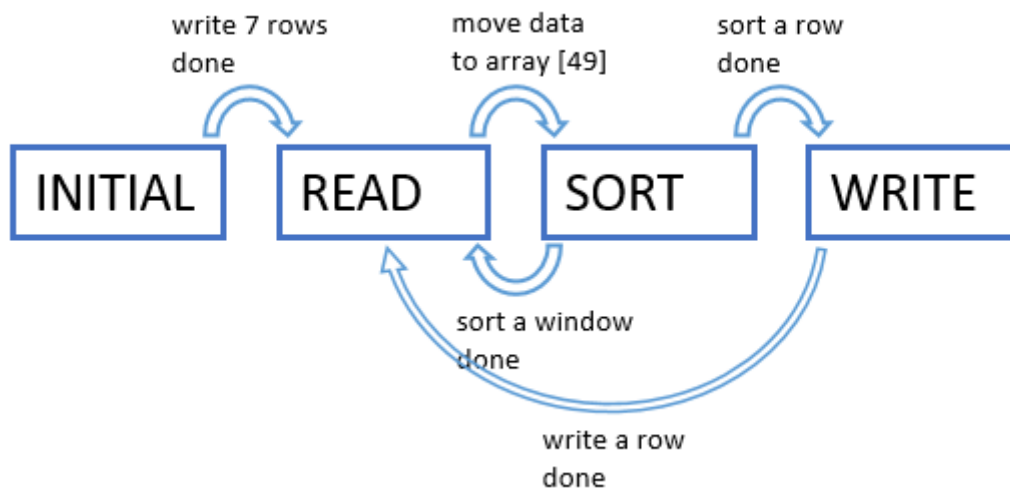
Student: 吳浩寧

Introduction

Median Filter 可應用在影像的處理，為給定一個固定大小的 window，以每個點為中心，找出在 window 內所有數值的中位數，並取代原點的值，做用為過濾信號中的雜訊，也可用來修復影像。本次題目規定的 window 大小為 7 x 7，若遇到邊緣，或角落不足 49 個值則以補 0 處理，處理的資料大小為 128 x 128 圖片。

Design Concept

使用 1KB 的 SRAM，在各個 state 之間轉換的條件如下圖：




```
always @(*) begin
  case (state)
    INIT:  if (Waddr == 937) nextstate = READ;
           else nextstate = INIT;
    READ:  if (colori == 48) nextstate = SORT;
           else nextstate = READ;
    SORT:  if (count == 48 && winx == 127) nextstate = WRITE;
           else if (count == 48) nextstate = READ;
           else nextstate = SORT;
    WRITE: if (posx == 133) nextstate = READ;
           else nextstate = WRITE;
  endcase
end
```

Initial

先讀入最初 7 列數值，為了簡化程式的複雜性，於邊界處我直接將 0 寫入記憶體中，寫完 938 個值即進入 Read State。寫入後 SRAM 的狀態如下圖所示：

000	0	0	0	0	0	0	0	0	...	0	0	0	0	0
134	0	0	0	0	0	0	0	0	...	0	0	0	0	0
268	0	0	0	0	0	0	0	0	...	0	0	0	0	0
402	0	0	0	138	138	140	131	134	...	117	146	0	0	0
536	0	0	0	137	136	136	132	131	...	107	78	0	0	0
650	0	0	0	133	133	134	135	133	...	52	41	0	0	0
784	0	0	0	134	132	132	133	131	...	44	46	0	0	0

window 

利用 posx、posy 記錄現在寫的位置，已判斷要寫入的值是由 Host 寫入抑或 0

```

always @(*) begin
    if(posx <= 2 || posy <= 2 || posx >= 131 || posy >= 131) begin
        Min = 0;
        busy = 1;
    end else begin
        Min = Din;
        busy = 0;
    end
end
end

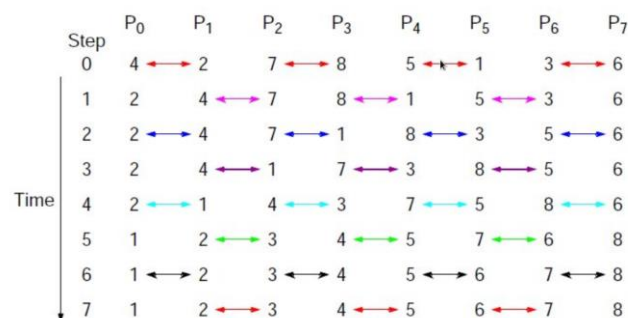
```

Read

從 SRAM 中讀 7 x 7 個數值到一維陣列中，準備進行排序，不完後進入 Sort State。

Sort

使用 Odd-Even Sort 排序法，在偶數步驟從第 1 個數開始兩兩交換，奇數步驟從第 2 個數開始，由於利用硬體平行的計算，可以減少要花的時間，最差情況從第一個位置換到最後也只需 48 個 cycles，完成排序即可以得到中位數，將結果輸出。若已算到一列最後一個 pixel 則進入 Write State，反之回到 Read State。



```

if(count[0] == 0) begin
    for (i = 0; i <= 46; i = i + 2) begin
        if (color[i] >= color[i+1]) begin
            nextcolor[i+1] = color[i];
            nextcolor[i] = color[i+1];
        end else begin
            nextcolor[i] = color[i];
            nextcolor[i+1] = color[i+1];
        end
    end
    nextcolor[48] = color[48];
    nextcount = count + 1;
end else begin
    for (i = 1; i <= 47; i = i + 2) begin
        if (color[i] >= color[i+1]) begin
            nextcolor[i+1] = color[i];
            nextcolor[i] = color[i+1];
        end else begin
            nextcolor[i] = color[i];
            nextcolor[i+1] = color[i+1];
        end
    end
    nextcolor[0] = color[0];
    nextcount = count + 1;
end
end

```

Write

寫入新的一列，將已不會使用到的位置蓋掉。

Result

Synthesis Script

```

read_file -format verilog LMFE.v

create_clock -period 10 -waveform {0 5} [get_ports {clk}]
set_dont_touch_network [get_clocks clk]
set_fix_hold [get_clocks clk]
set_clock_uncertainty 0.1 [get_clocks clk]
set_clock_latency 0.5 [get_clocks clk]

set_input_delay 5 -clock clk [remove_from_collection [all_inputs] [get_ports clk]]
set_output_delay 0.5 -clock clk [all_outputs]

set_load 1 [all_outputs]
set_drive 1 [all_inputs]

set_operating_conditions -max_library slow -max slow
set_wire_load_model -name tsmc13_wl10 -library slow

set_max_fanout 20 [all_inputs]

compile_ultra
write_file -format ddc -hier -output LMFE.ddc
write_file -format verilog -hier -output LMFE_syn.v
write_sdf -version 2.1 -context verilog LMFE.sdf
write_parasitics -format reduced -o LMFE.spef

report_timing > Timing.txt
report_area > Area.txt
report_power > Power.txt

exit

```

Area

```

Number of ports:          21
Number of nets:          3082
Number of cells:         2716
Number of combinational cells: 2248
Number of sequential cells:  467
Number of macros/black boxes: 1
Number of buf/inv:       149
Number of references:    84

```

```

Combinational area:      18374.355001
Buf/Inv area:            933.569986
Noncombinational area:  15054.240097
Macro/Black Box area:   69557.296875
Net Interconnect area:  403248.722748

```

```

Total cell area:        102985.891973
Total area:             506234.614721

```

Gate count = 102985.89/5.09 \approx 20232

Power

```

Cell Internal Power  = 2.6582 mW (98%)
Net Switching Power = 53.8356 uW (2%)
-----
Total Dynamic Power  = 2.7120 mW (100%)
Cell Leakage Power   = 47.8403 uW

```

Power Group	Internal Power	Switching Power	Leakage Power	Total Power	(%)	Attrs
io_pad	0.0000	0.0000	0.0000	0.0000	(0.00%)	
memory	1.6950	5.3692e-04	2.8000e+07	1.7235	(62.45%)	
black_box	0.0000	0.0000	0.0000	0.0000	(0.00%)	
clock_network	0.0000	0.0000	0.0000	0.0000	(0.00%)	
register	0.9597	6.2841e-03	1.3849e+07	0.9799	(35.50%)	
sequential	0.0000	0.0000	0.0000	0.0000	(0.00%)	
combinational	3.4607e-03	4.7014e-02	5.9916e+06	5.6467e-02	(2.05%)	
Total	2.6582 mW	5.3835e-02 mW	4.7840e+07 pW	2.7599 mW		

Timing

Point	Incr	Path
data arrival time		9.95
clock clk (rise edge)	10.00	10.00
clock network delay (ideal)	0.50	10.50
clock uncertainty	-0.10	10.40
Raddr_reg[9]/CK (DFFRX1)	0.00	10.40 r
library setup time	-0.29	10.11
data required time		10.11
data required time		10.11
data arrival time		-9.95
slack (MET)		0.16

Post-synthesis

Annotation completed with 0 Errors and 22 Warnings

SDF statistics: No. of Pathdelays = 9099 Annotated = 99.91% -- No. of Tchecks = 4277 Annotated = 99.91%

	Total	Annotated	Percentage
Path Delays	9099	9091	99.91
\$period	1	1	100.00
\$width	1400	1400	100.00
\$setuphold	2876	2872	99.86

Output pixel: 0 ~ 1000 are correct!
Output pixel: 0 ~ 2000 are correct!
Output pixel: 0 ~ 3000 are correct!
Output pixel: 0 ~ 4000 are correct!
Output pixel: 0 ~ 5000 are correct!
Output pixel: 0 ~ 6000 are correct!
Output pixel: 0 ~ 7000 are correct!
Output pixel: 0 ~ 8000 are correct!
Output pixel: 0 ~ 9000 are correct!
Output pixel: 0 ~ 10000 are correct!
Output pixel: 0 ~ 11000 are correct!
Output pixel: 0 ~ 12000 are correct!
Output pixel: 0 ~ 13000 are correct!
Output pixel: 0 ~ 14000 are correct!
Output pixel: 0 ~ 15000 are correct!
Output pixel: 0 ~ 16000 are correct!

Congratulations! All data have been generated successfully!

-----PASS-----

Simulation complete via \$finish(1) at time 16399741 NS + 0

nLint

The nLint tool output is displayed in a tree view. The root node is 'Clock Domain Analysis', which contains a sub-node 'Total - 6 Error(s), 210 Warning(s), 16 Information(s)'. This sub-node is expanded, showing several categories of linting issues:

- Compilation & Elaboration -1 Error(s)**
 - 16651 - view %s is not defined for instance %s - Compilation & Elaboration - 1 Error(s)
 - LMFE.v(47) : Error : view sram_1024x8_t13 is not defined for instance ram1
- Naming Convention -15 Warning(s), 3 Information(s)**
 - NC.9-1 - Signals names are lowercase letters - Naming Convention - 10 Warning(s)
 - NC.9-5 - Port names are lowercase letters - Naming Convention - 2 Warning(s)
 - NC.12 - Consistent reset signal name - Naming Convention - 1 Warning(s)
 - NC.14-1 - For FSM variables, naming in <fsm_cs>, <fsm_ns> - Naming Convention - 1 Information(s)
 - NC.23 - Consistent ordering of bits for describing multibit buses - Naming Convention - 2 Warning(s)
 - NC.17 - For asynchronous signals, end in _a - Naming Convention - 1 Information(s)
 - NC.14-2 - For FSM variables, naming in <fsm_cs>, <fsm_ns> - Naming Convention - 1 Information(s)
- File Header -1 Warning(s)**
 - FH.1*4 - Check file header format - File Header - 1 Warning(s)
- Comments -5 Error(s), 9 Information(s)**
 - COM.1 - Use comments for port declarations - Comments - 5 Error(s)
 - COM.2 - Comment signal declarations - Comments - 1 Information(s)
 - COM.3 - Use comments for functional sections - Comments - 8 Information(s)
- Coding Style -185 Warning(s), 3 Information(s)**
 - CS.13 - Use parentheses in complex equations - Coding Style - 11 Warning(s)
 - CS.7 - Keep line length within 72 characters - Coding Style - 3 Information(s)
 - CS.1 - Code should be aligned in a tabular format - Coding Style - 48 Warning(s)
 - CS.2 - Use space instead of tab stops for code indentation - Coding Style - 122 Warning(s)
 - CS.5 - Preserve port order - Coding Style - 4 Warning(s)
- Synthesis -8 Warning(s)**
 - SYN.8 - Avoid top level glue logic - Synthesis - 8 Warning(s)
- Design For Test -1 Information(s)**
 - DFT.10-3 - Avoid floating output - Design For Test - 1 Information(s)
- Design Style -1 Warning(s)**
 - D.1.1 - Output signals must be registered - Design Style - 1 Warning(s)

Primetime Script

```
set search_path ". /theda21_2/CBDK_IC_Contest/cur/SynopsysDC/db /usr/synopsys/synthesis/cur/libraries/syn $search_path"
set link_path {* slow.db fast.db sram_1024x8_t13_slow_syn.db}

read_verilog LMFE_syn.v
current_design LMFE
link_design
set_operating_conditions -min slow -max slow

create_clock -period 10 -waveform {0 5} [get_ports clk]
set_dont_touch_network [get_clocks clk]
set_clock_transition -rise 0.1 [get_clocks {clk}]
set_clock_transition -fall 0.1 [get_clocks {clk}]
set_clock_uncertainty 0.1 [get_clocks clk]
set_clock_latency 0.5 [get_clocks {clk}]

set_input_delay 5 -clock clk [remove_from_collection [all_inputs] [get_ports clk]]
set_output_delay 0.5 -clock clk [all_outputs]
set_load 1 [all_outputs]
set_drive 1 [all_inputs]

set_max_fanout 20 [current_design]

set_wire_load_model -name tsmc13_wl10 -library slow

set_dont_touch_network [list clk reset]
set_drive 0 [list clk reset]

read_sdf LMFE.sdf

report_clock > clock.rpt
report_port -input_delay >> clock.rpt
report_port -output_delay >> clock.rpt
check_timing >> clock.rpt

report_constraint -all_violators > timing.rpt
report_timing >> timing.rpt
report_timing -nets -transition_time -capacitance >> timing.rpt
report_timing -nworst 10 -path_type summary >> timing.rpt

exit
```

Timing

Startpoint: winx_reg[3]
(rising edge-triggered flip-flop clocked by clk)

Endpoint: Raddr_reg[9]
(rising edge-triggered flip-flop clocked by clk)

Path Group: clk

Path Type: max

Point	Fanout	Cap	Trans	Incr	Path

data arrival time					9.95
clock clk (rise edge)			0.10	10.00	10.00
clock network delay (ideal)				0.50	10.50
clock reconvergence pessimism				0.00	10.50
clock uncertainty				-0.10	10.40
Raddr_reg[9]/CK (DFFRX1)					10.40 r
library setup time				-0.29 *	10.11
data required time					10.11

data required time					10.11
data arrival time					-9.95

slack (MET)					0.16

Critical Path:

winx→winx+1→nextwinx→nextRaddr→Raddr

not false path

PrimetimePX Script

```
set power_enable_analysis TRUE
set power_analysis_mode averaged

set search_path          ". /theda21_2/CBDK_IC_Constest/cur/SynopsysDC/db /usr/synopsys/synthesis/cur/libraries/syn"
set link_library          " * slow.db fast.db sram_1024x8_t13_slow_syn.db"

read_verilog              LMFE_syn.v
current_design             LMFE
link

read_parasitics           LMFE.spef

check_timing
update_timing
report_timing

check_power
update_power
report_power -hierarchy > vf.rpt

exit
```

Power

Hierarchy	Int Power	Switch Power	Leak Power	Total Power	%
LMFE	1.40e-03	1.19e-04	4.78e-05	1.56e-03	100.0

Summary

Area	Power (mW)	Time (ns)
506234.614721	1.56	16399741

Discussions & Conclusions

這次遇到的問題包括讀 SRAM 時，由於取得結果會慢一個 cycle，所以在判斷可否讀值到陣列中時，須多加個 halt 訊號來暫緩一個 cycle；此外本來我將排序寫成 combinational circuit，直接用 repeat(48)做完，後來才發現這樣會產生 combinational loop，須改為一個 cycle 進行一個 Odd-Even Sort 的步驟才合理。

此外我記憶體的讀、寫都是循環的，每當遇到最後的位置，下個 clock 即會重回到位置 0，好處是可以省下記憶體空間，和多餘的判斷式；另外，我多使用了 Raddrmod 的訊號來記錄 Raddr 除以一列長度的餘數，如此可以避免使用 % 運算子；遇到比較數值大小，則盡量使用合成面積較小的 \geq 、 \leq 而非 $>$ 、 $<$ 。

感覺我這次的寫法還可以有不少改進的方式，比如不用寫多餘 0，可以省下 $3*4*134-4*3*3=1572$ 個 cycles，雖然如此判斷式會變複雜，電路面積也會稍微變大；若在排序同時進行 SRAM 的讀取，可以省下 $(134-7)*(134-7)=16129$ 個 cycles；每次讀取 SRAM 到排序陣列也不用全部重讀，只須讀 window 移動後新增的 7 個數，可以省下 $(134-7)*(42*(134-7))=677418$ 缺點是要多用一個陣列來存排序後的值。