

## 積體電路設計 Homework #3 Report

B06602035 生工三 李晴妍

### 一、Discussion

#### 1. 將 inputs 轉成 10 bits

```
assign a0={i0[5],i0[5],i0[5],i0};
assign a1={i1[5],i1[5],i1[5],i1};
assign a2={i2[5],i2[5],i2[5],i2};
assign a3={i3[5],i3[5],i3[5],i3};
assign a4={i4[5],i4[5],i4[5],i4};
assign a5={i5[5],i5[5],i5[5],i5};
assign a6={i6[5],i6[5],i6[5],i6};
assign a7={i7[5],i7[5],i7[5],i7};
assign a8={i8[5],i8[5],i8[5],i8};
```

原本 inputs 是 6 bits，在 two's complement 時 range : -32~31。因為 inputs 一共有 9 個，所以 range 會在 -288~279 之間。為了避免 overflow，10 bits ( range: -512~511 )剛好是包含 input range 的最小位數，所以在計算前先將所有 inputs 都轉成 10 bits。

#### 2. 用 Carry safe adder 將九個 Inputs 相加

```
module CSAdd_9(I1, I2, I3, I4, I5, I6, I7, I8, I9, 0);
```

a. Carry safe adder: 每三個 inputs 會產生一個 c 和 s

(I1, I2, I3) => (c1, s1)

(I4, I5, I6) => (c2, s2)

(I7, I8, I9) => (c3, s3)

b. s 和 c 各自相加

(c1, c2, c3) => (c4, s4)

(s1, s2, s3) => (c5, s5)

c. 先加 c4, s4, c5

(c4, s4, c5) => (c6, s6)

d. 再加 c6, s6, s5

(c6, s6, s5) => (c7, s7)

e. Ripple-carry adder: output

Output = c7 + s7

最開始我全部用 ripple-carry adder 去計算九個 inputs 和，這樣每次 10 bits 相

加就需要 10 個 FA，全部需要 80 個 FA，時間大概是 9.08ns。

後來換成 7 bits + 7 bits 相加 (為了避免 overflow 要多加 1 bits)，也就是

{i0,i1},{i2,i3},{i4,i5},{i6,i7}。再 8 bits + 8 bits、9 bits + 9 bits，10 bits + 10 bits 得

到最後的和，有點類似 tree adder，總共需要 63 個 FA，也一樣都是 ripple-

carry adder，時間大概在 8.03ns。

```
// 6+6 bits -> change to 7 bits
wire c1,c2,c3,c4;
wire[6:0] sum1,sum2,sum3,sum4;
add7 add7_1(c1,sum1,{i0[5],i0},{i1[5],i1},1'b0);
add7 add7_2(c2,sum2,{i2[5],i2},{i3[5],i3},1'b0);
add7 add7_3(c3,sum3,{i4[5],i4},{i5[5],i5},1'b0);
add7 add7_4(c4,sum4,{i6[5],i6},{i7[5],i7},1'b0);

// 7+7 bits -> change to 8 bits
wire c5,c6;
wire[7:0] sum5,sum6;
add8 add8_1(c5,sum5,{sum4[6],sum4},{sum2[6],sum2},1'b0);
add8 add8_2(c6,sum6,{sum3[6],sum3},{sum1[6],sum1},1'b0);

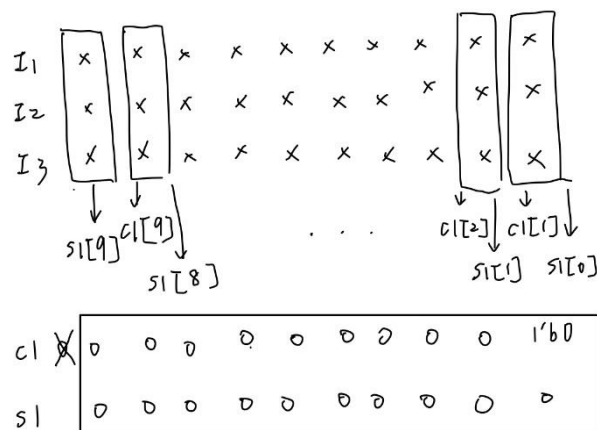
// 8+8 bits -> change to 9 bits
wire c7;
wire[8:0] sum7;
add9 add9_1(c7,sum7,{sum5[7],sum5},{sum6[7],sum6},1'b0);

// 9+9 bits -> change to 10 bits
wire c8;
wire[9:0] sum8;
add10 add10_1111(c8,sum8,{sum7[8],sum7},{i8[5],i8[5],i8[5],i8[5],i8},1'b0);
```

最後就是用 carry safe adder，減少等待 bit 傳送的時間。

{i1[0],i2[0],i3[0]}=>{c1[1],s1[0]}，以此類推。最後一個{i1[9],i2[9],i3[9]}=>{s1[9]}

忽略進位 c(因為已經全部換成 10bits)，而最開始的 c1[0]要 assign 1'b0。



這樣一次計算 9 個 inputs 會比一次 3 個 inputs 少兩次最後 c 跟 s 加總(ripple carry adder)的時間。

### 3. 除 9 大約等於 $(1/16 + 1/32 + 1/64 + 1/512)$

$$1/9 = 0.111111$$

$$1/16 + 1/32 + 1/64 + 1/512 = 0.111$$

Divider 我是用 shift + add 實現，將 sum 右移 4 bits、5 bits、6 bits、9 bits 後相加會最接近  $\text{sum}/9$  的值。

- a. 先將 sum 分成 6bits 整數位跟 6bits 小數位

$\text{sum}/16 = \{\text{int}16, \text{digit}16\}$

$\text{sum}/32 = \{\text{int}32, \text{digit}32\}$

$\text{sum}/64 = \{\text{int}64, \text{digit}64\}$

$\text{sum}/512 = \{\text{int}512, \text{digit}512\}$

之所以取 6bits 是因為/16 右移 4bits，小數點前面還保留了 6bits。

而 digit512 的後三位可以省略，因為沒有其他可以跟它相加，也就沒有進位問題，不會影響到最後的 average。

- b. 整數部分先做相加

$\text{Int}16 + \text{int}32 + \text{int}64 + \text{int}512 = \text{sum}3$

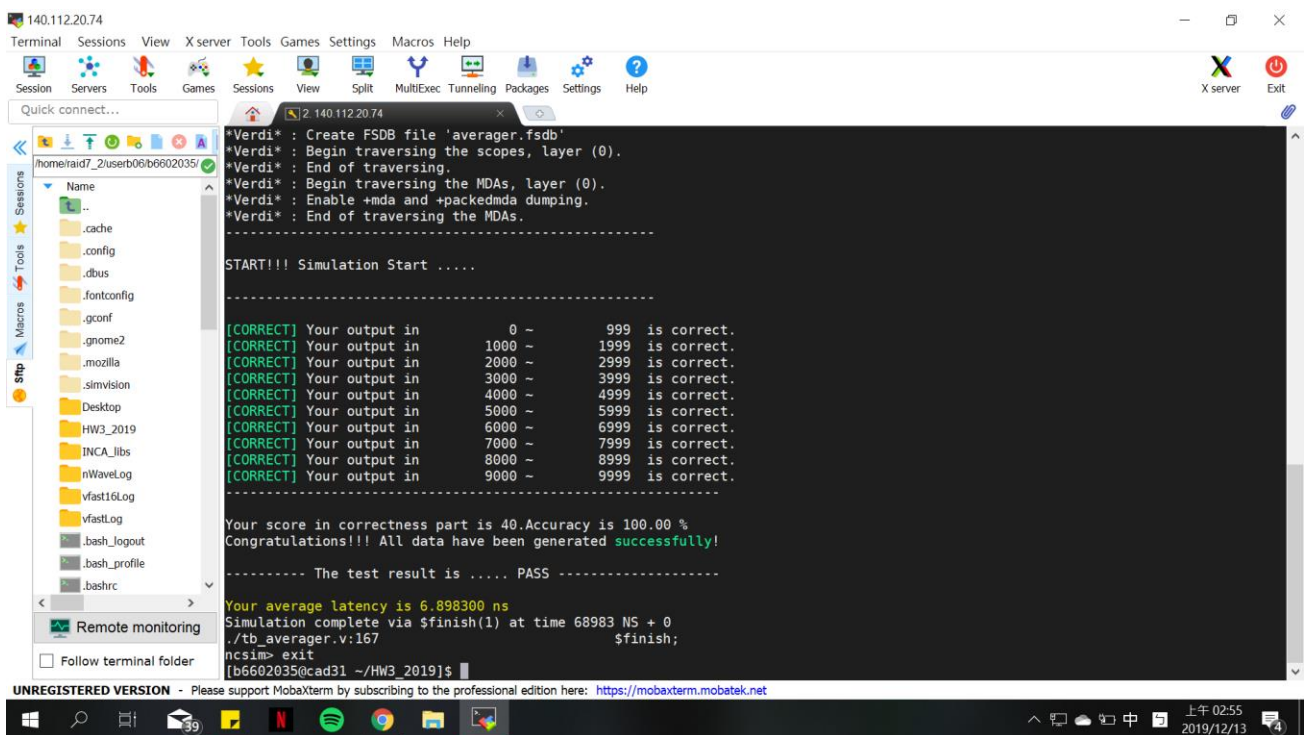
- c. 小數部分相加，進位 c 要保留

$\text{Digit}16 + \text{digit}32 + \text{digit}64 + \text{digit}512 = \text{sum}6$

- d.  $\text{Average} = \text{sum}3 + \text{sum}6 +$  小數相加的進位 + 四捨五入

$\text{Average} = \text{sum}3 + \text{sum}6 + c + \text{sum}6[5]$

因為要考慮四捨五入的進位問題，所以要看看小數總和的第一位是不是  $1(2^{-1} = 0.5)$ ，是的話 average 就要加 1。



```
*Verdi* : Create FSDB file 'averager.fsdb'
*Verdi* : Begin traversing the scopes, layer (0).
*Verdi* : End of traversing.
*Verdi* : Begin traversing the MDAs, layer (0).
*Verdi* : Enable +mda and +packedmda dumping.
*Verdi* : End of traversing the MDAs.

START!!! Simulation Start .....

[CORRECT] Your output in      0 ~      999 is correct.
[CORRECT] Your output in    1000 ~     1999 is correct.
[CORRECT] Your output in    2000 ~     2999 is correct.
[CORRECT] Your output in    3000 ~     3999 is correct.
[CORRECT] Your output in    4000 ~     4999 is correct.
[CORRECT] Your output in    5000 ~     5999 is correct.
[CORRECT] Your output in    6000 ~     6999 is correct.
[CORRECT] Your output in    7000 ~     7999 is correct.
[CORRECT] Your output in    8000 ~     8999 is correct.
[CORRECT] Your output in    9000 ~     9999 is correct.

Your score in correctness part is 40. Accuracy is 100.00 %
Congratulations!!! All data have been generated successfully!

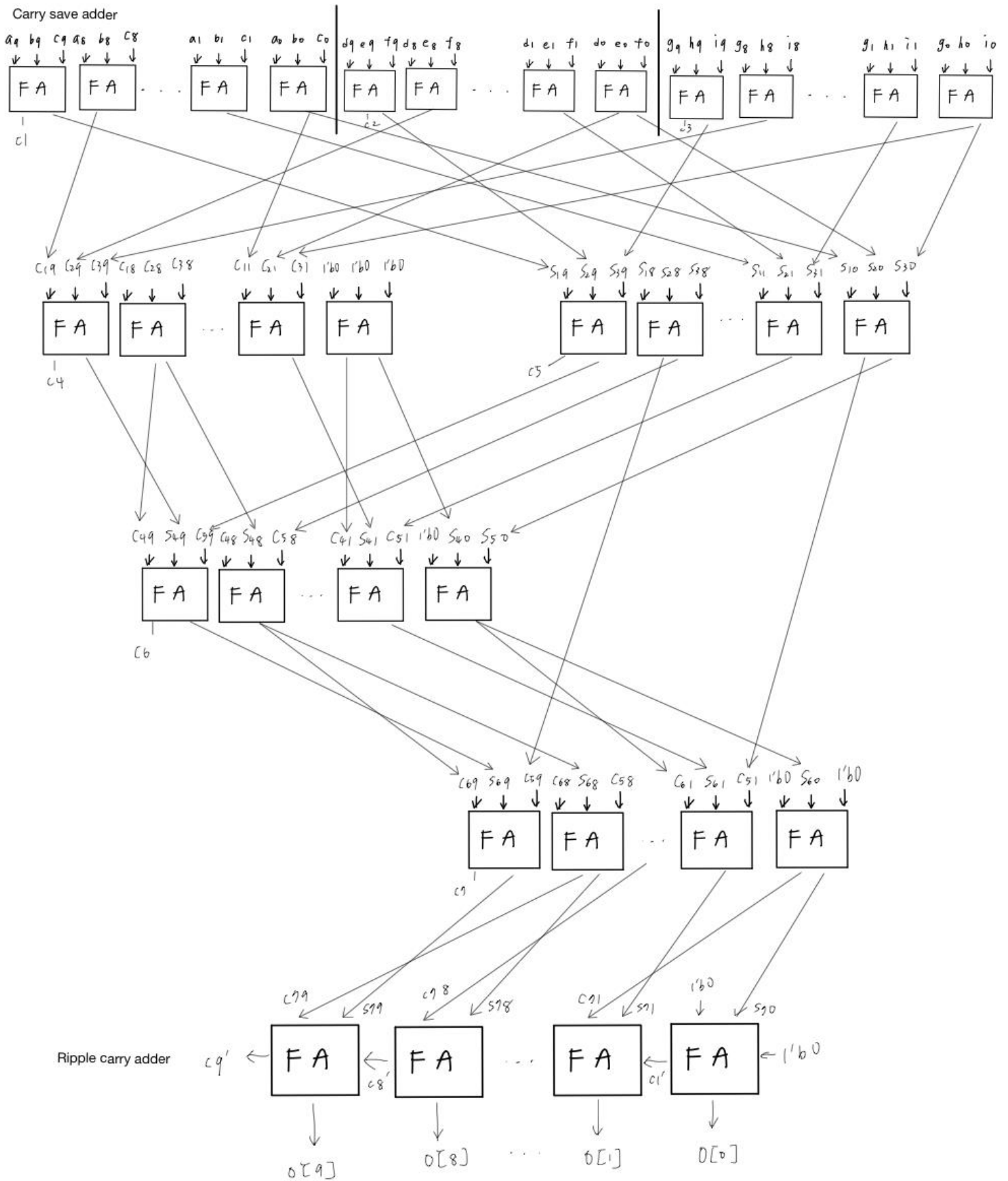
----- The test result is ..... PASS -----

Your average latency is 6.898300 ns
Simulation complete via $finish(1) at time 68983 NS + 0
./tb_averager.v:167                               $finish;
ncsim> exit
[b6602035@cad31 ~/HW3_2019]$
```

## 二、Circuit diagram

```
module CSAdd_9(I1, I2, I3, I4, I5, I6, I7, I8, I9, O);
```

Carry save adder



6 bits to 10 bits

$$I = \{x_5, x_4, x_3, x_2, x_1, x_0\}$$

$$\Rightarrow I = \{a_9, a_8, a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0\}$$

