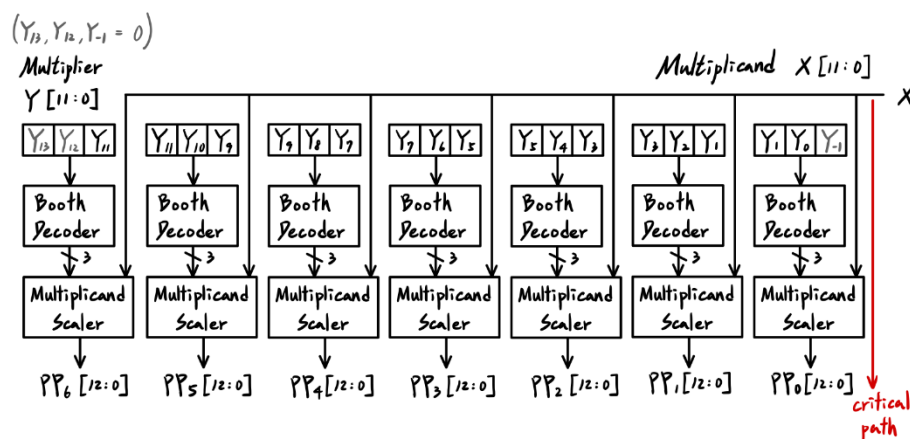


# Digital Integrated Circuits homework 7 電子所 陳柏翔 313510156

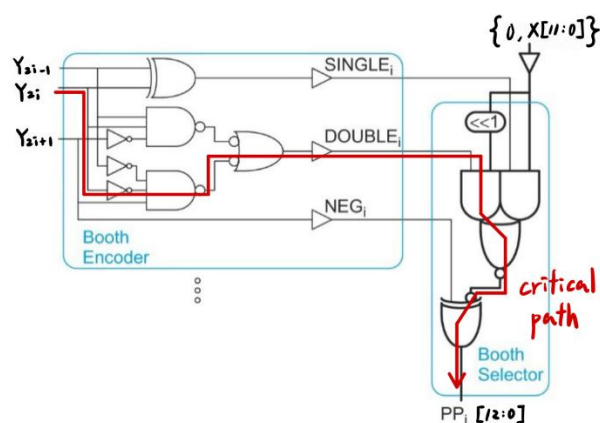
- Design a **12-bit unsigned multiplier** ( $X$ : 12 bits,  $Y$ : 12 bits,  $S$ : 24 bits,  $P$ : 24 bits)  
 $S = X \times Y + P$  by using **Radix-4 Booth multipliers**. The goal is to have minimum critical path delay time with the least gate count.

- Show your **block diagram** as shown in Fig.1 below (example for the case of 8-bit; booth decoder and Multiplicand scale corresponding to Booth Encoder and Booth selector of Fig.10.80). For each block, you shall **show its logic design diagram**. Indicate the **critical path**. Explain your **design concepts**.

- Block diagram (& Critical path):**



- Logic design for each block (& Critical path):**



## Booth Encoder (Left) & Booth Selector (Right)

(此作業中的 Booth Decoder block) (此作業中的 Multiplicand Scaler block)

- Design concepts:**

如同課程中所學的 Radix-4 Booth multiplier 的操作原理，將  $Y$  向前與向後各看一個 Bit，組成為  $(Y_{2i+1}, Y_{2i}, Y_{2i-1})$  並從  $i = 0, 1, 2, \dots$  開始掃描，

此外還需要補上 0，也就是設  $Y_{13}, Y_{12}, Y_{-1} = 0$ ，當掃完所有的三個 Bits 群組後，放入 Booth Decoder 產生  $SINGLE_i, DOUBLE_i, NEG_i$ ，再和  $X$  經過 Multiplicand Scaler 產生  $-2X, -X, 0, X, 2X$  五種情形(如下 Table)，可以讓 CSA array 的層數大幅降低，達到高的运算速度：

$Y_{2i+1}$	$Y_{2i}$	$Y_{2i-1}$	$PP_i$	$SINGLE_i$	$DOUBLE_i$	$NEG_i$
0	0	0	0	0	0	0
0	0	1	+X	1	0	0
0	1	0	+X	1	0	0
0	1	1	+2X	0	1	0
1	0	0	-2X	0	1	1
1	0	1	-X	1	0	1
1	1	0	-X	1	0	1
1	1	1	0	0	0	1

其中：

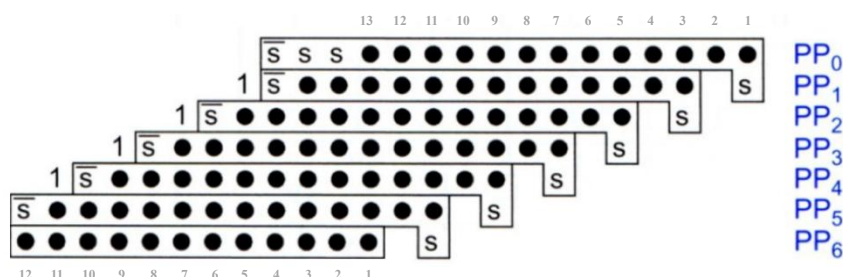
$$SINGLE_i = Y_{2i-1} \cdot \overline{Y_{2i}} + \overline{Y_{2i-1}} \cdot Y_{2i} = Y_{2i-1} \oplus Y_{2i}$$

$$DOUBLE_i = Y_{2i-1} \cdot Y_{2i} \cdot \overline{Y_{2i+1}} + \overline{Y_{2i-1}} \cdot \overline{Y_{2i}} \cdot Y_{2i+1}$$

$$NEG_i = Y_{2i+1}$$

而 SINGLE 的意思就是乘上 1 倍；DOUBLE 的意思就是乘上 2 倍(即左移 1 位)；兩者都為 0 的話就是乘上 0；而 NEG 的意思是乘上負號。最後再根據這個規則經過 Multiplicand Scaler 來輸出 Partial Products ( $PP_i$ )。

- (2) Shown the Radix Booth-4 encoded partial products with **simplified sign extension** like that shown in Fig.10.82.



先把 sign bit 做 signed extension，再將所有的 1 先加起來，便可化簡成上圖。在 0~5 層的 Partial Products 會有 13 bits 而非 12 bits 是因為可能有兩倍的情況，而第 6 層的掃描數值 ( $Y_{13}, Y_{12}, Y_{11}$ ) 中的前兩個為 0，因此 Partial Product 僅可能是 0 倍或 1 倍，故只需要 12 bits。

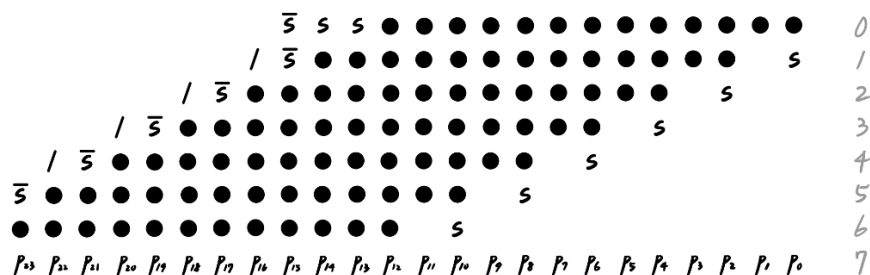
- (3) Design the partial products with **Dadda method** shown in Fig.5.19. Carry-ripple adder is used in the last stage of CPA. Show the **critical path** and indicate the **delay time** in terms of number of HA and FA. For area, show the **number of FA and HA** used. Explain your **design concepts**.

• **前述:**

為避免混淆接下來要做的設計，依照最初的題意應為設計一個 MAC ( $S = X \times Y + P$ )而不是一個 Multiplier ( $= X \times Y$ )。因此需要注意的是，如同課程講義中最後一頁所述，不該把 MAC 當成一個 Multiplier 串接一個 Adder 來實現，而是實現在一起才對，所以下方的設計是以 MAC ( $S = X \times Y + P$ )為主(會多出一排)而不是只有 Multiplier ( $= X \times Y$ )之後再接上了一個獨立的 CPA。

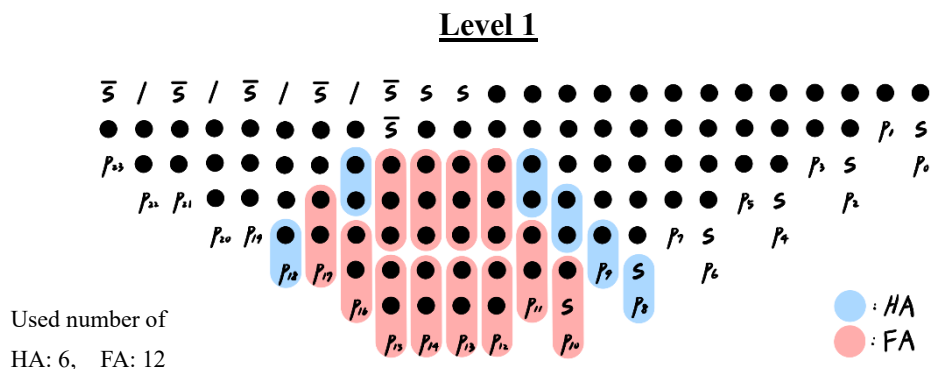
• **Design concepts:**

首先整理一下  $S = X \times Y + P$  的 Dot diagram 變成下圖：

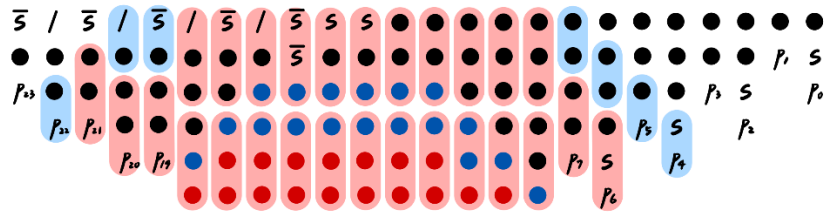


其中第 0~6 層都是屬於 Multiplier ( $= X \times Y$ )的部分，而第 7 層則是額外加的數值  $P$  的一層，因此總共會有 8 層。

根據 Dadda method，會使用 HA 和 FA 來逐漸合併數值，合併後的目標最大層數為 2, 3, 4, 6, 9, ... 是從最終目標層數 2 開始每次乘上  $(3/2)$  取 floor 的數值所形成的序列。由於這邊的最大層數是 8，因此 CSA tree 的 Level 1 目標是將最大層數從 8 縮減到 6、接著 Level 2 再縮減到 4、接著 3... 依此類推。設計方法與過程如下圖：



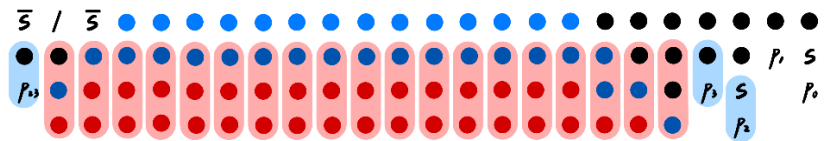
## Level 2



Used number of HA: 7, FA: 27

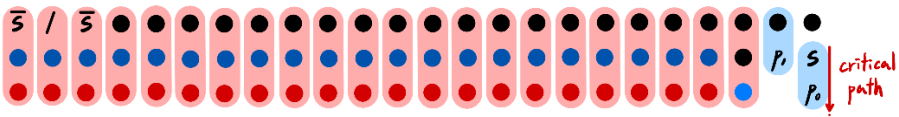
● : 前一層的 Sum  
● : 前一層的 Carry

## Level 3



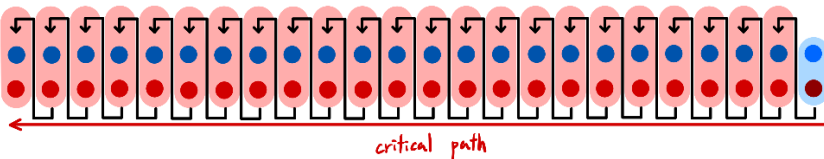
Used number of HA: 3, FA: 19

## Level 4



Used number of HA: 2, FA: 22

## Level 5 (CPA)



Used number of HA: 1, FA: 23

### • Area:

Level	Half Adder	Full Adder
1	6	12
2	7	27
3	3	19
4	2	22
5	1	23
Total	19	103

### • Delay time & Critical path:

由於在前面數層的 Dadda tree 相加中都遠不如最後一層 CPA 的 Delay，因此 Critical path 如上圖中最後兩層所標示的紅線所示，是從 Level 4 的 HA 到 Level 5 的 HA 再一路經過整排 FA 的路徑。而 Delay time 就是 Critical path 上所花的時間： $2 \cdot t_{HA} + 23 \cdot t_{FA}$