DLD – lab03 – Vector Inner Product

104020002 理雙 19 周明曄

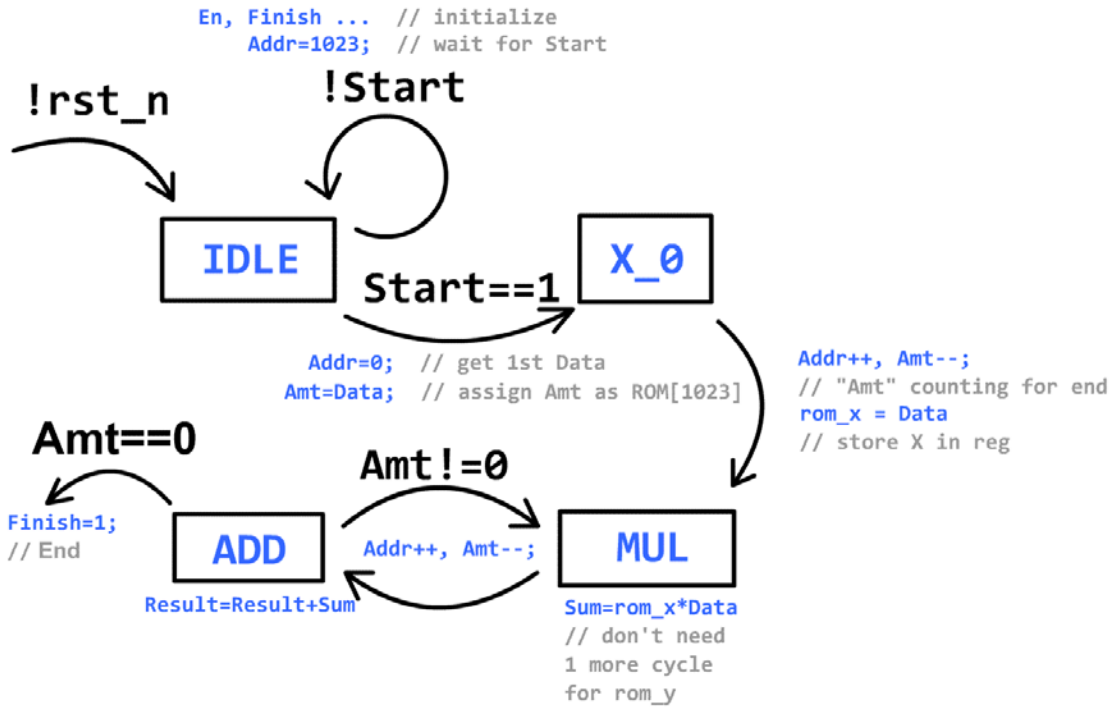
一、目的以及流程圖

$$dot(X, Y) = \sum_{i=0}^{n-1} X_i * Y_i$$

※計算內積及其加總

1. 學會使用 FSM 來操作流程
2. 學會使用時脈 clk 及 d_ff 操作 sequential 電路

二、Finite State Machine



IDLE:

1. 所有變數的初始，並等待 Start 拉起，即開始計算。
2. 等待 Start 拉起的這幾個 cycle 裡，會向 ROM 要取存放在[1023]的資料數，以方便得知計算是否結束。(存放在 Amt 中，做為終止條件)
3. Start 拉起之後，會在此 cycle 中要取第一個 X 參數，並在下一個 cycle 得到。(在 Addr 的 flip-flop 中存放 index,[0])

X_0:

1. 得到 X 參數，存放在 rom_X 中，並要取第一個 Y 參數。(Addr++, Amt--)
2. 無條件進入 MUL

MUL:

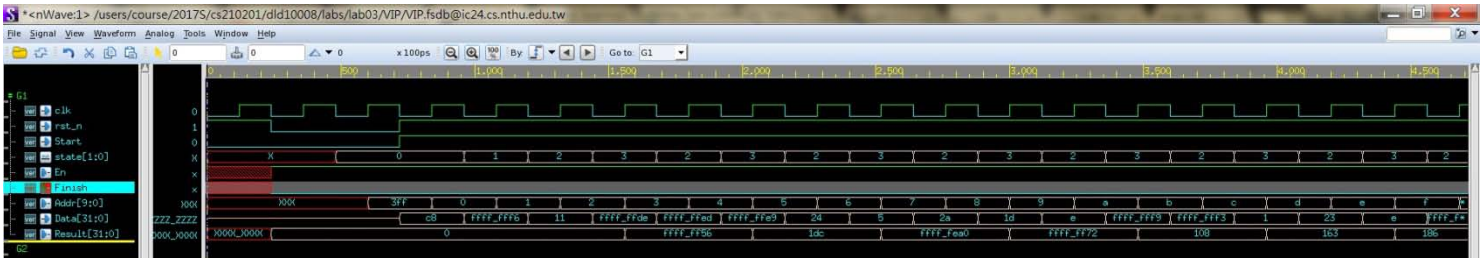
1. 得到 Y 參數，並要取第一個 X 參數。(Addr++, Amt--)
2. 將 rom_X 與要取到的 Y 用 ALU 乘起來存在 Sum 裡。

ADD:

1. 將目前的 Sum 跟上個 cycle 的 Result 加在當前 cycle 的 Result。
2. 檢查 Amt 是否等於 0。

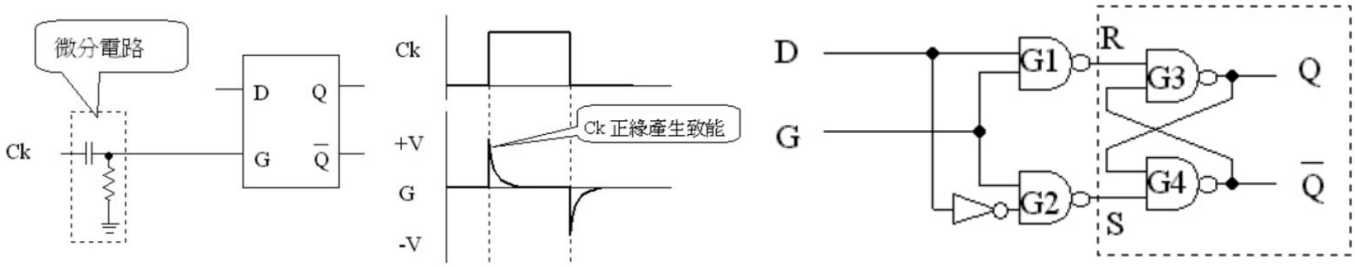
若是，則結束計算，使 Finish 為 1，在下個 cycle 檢查答案正確與否。

若否，則繼續要下一個 X 參數，並重回 MUL state。



三、Synthesis

D flip-flop / D Latch :



※左為 flip-flop，右為 Latch。

※圖片來源: http://www.gauss.com.tw/logic/ch9/index9.htm

兩者差別在於，觸發的訊號，只要觸發存放資料的訊號為任意訊號的都可以被稱做 Latch，而 flip-flop 的觸發訊號，一定要是時脈訊號。

我們合成 reg 得到 Latch 的時候，是因為我們在此 cycle 並沒有對此參數做任何指定或描述，進而讓合成軟體認為我們在未來將會用到此參數。但卻沒有指定，因此用非時脈觸發的型式來存儲，也就是 Latch。

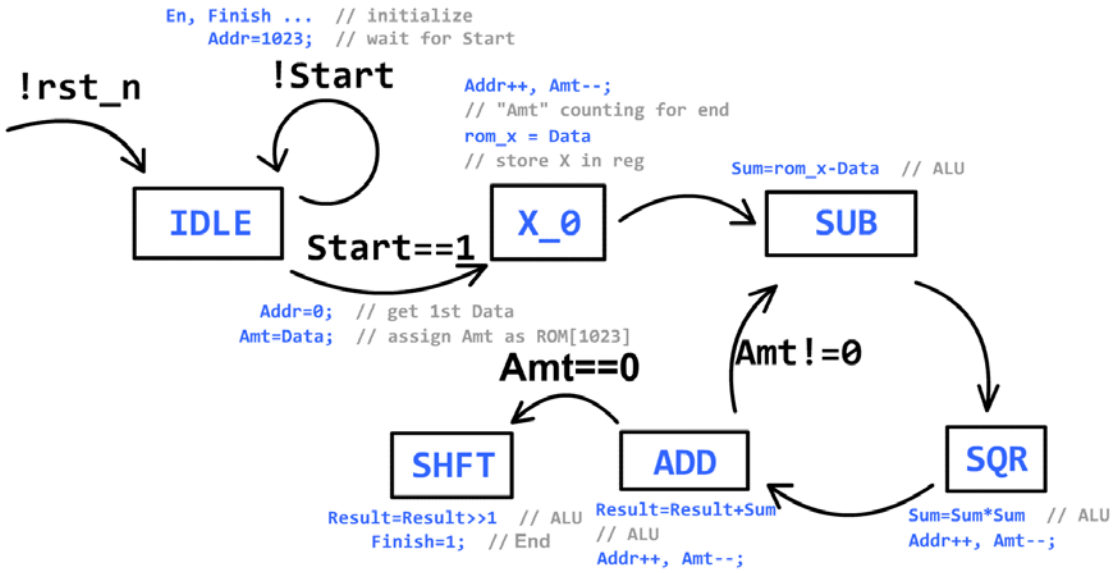
這種情況通常是發生在使用 if 卻沒有給定 else，case 並沒有討論完全，或是遺漏了某些參數的狀況。因此在此次 lab 中，只要先定義好 d flip-flop 的元件，包括 rst 情況，以及一個 cycle 中的清況處理(有指定新值及指定新值，沒指定則指定原值)，再把在 FSM 會用到的所有參數，全都丟到 d flip-flop 裡，就不會合出 latch 了。

※一開始沒有做好細節，Latch 使我非常頭痛，不過後來我好像也沒有寫好沒指定新值則給定原值的部分，卻沒有合出 Latch，覺得問號???

四、Bonus

$$f(X, Y) = 0.5 * \sum_{i=0}^{n-1} (X_i - Y_i)^2$$

FSM:



IDLE:

同 VIP。

X_0:

同 VIP，進入 SUB state。

SUB:

使用 ALU 將 rom_X 與要到的 Data(Y)相減，無條件進入 SQR。

SQR:

將當前 sum 與自己相乘，存於下一個 cycle 會得到的 sum 裡，並要下個 X。

ADD:

將當前 sum 與 Result 相加後存於下一個 cycyle 的 Result。

檢查 Amt 是否為 0，若是，進入 SUB 繼續循環；反之則結束，進入 SHFT。

SHFT:

使用 ALU 右移一個 bit，相當於乘 0.5、除以二。

並且將 Finish 拉起，En 放下。