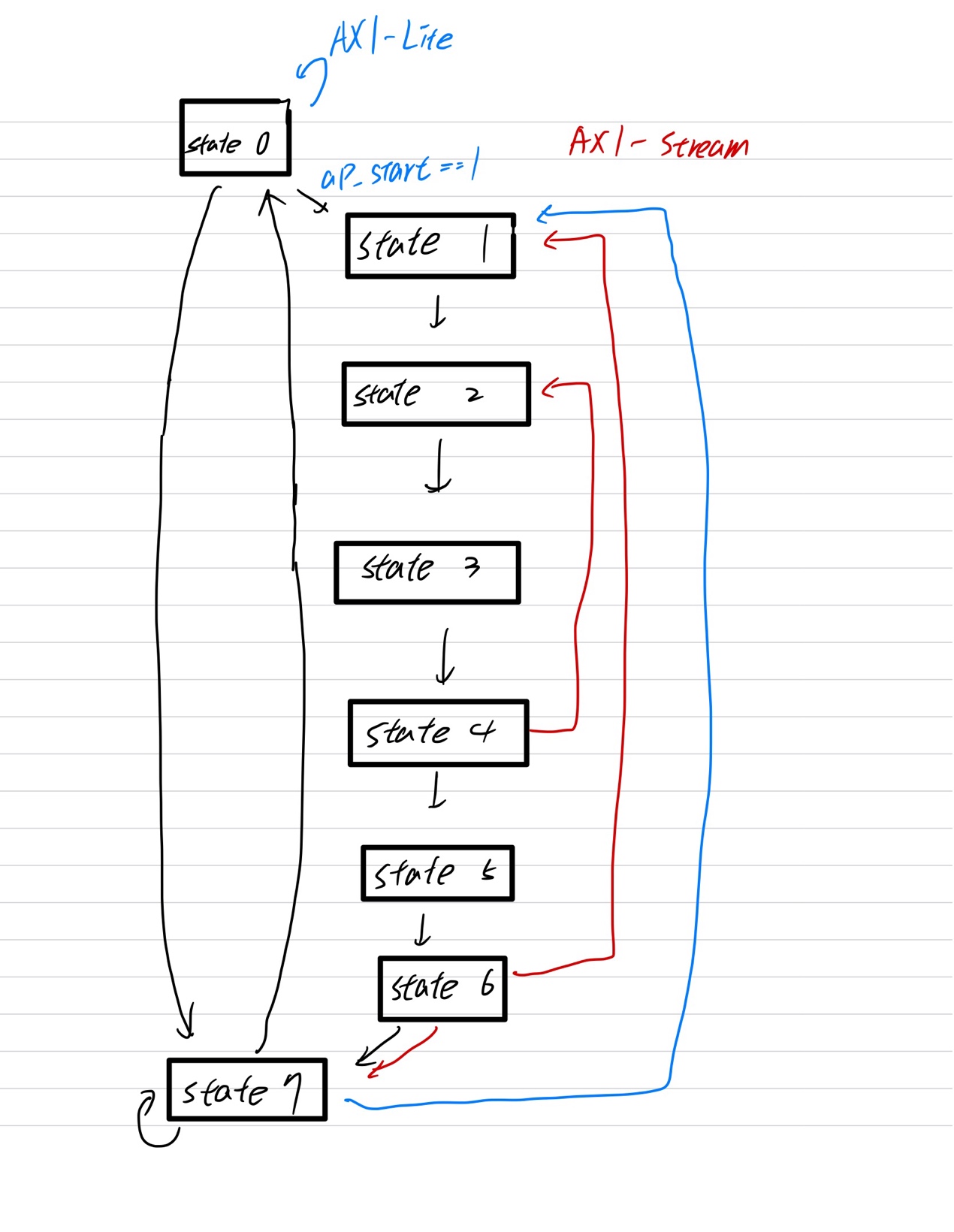
111061589 紀皓洋 LAB3

Block diagram

我全部使用ＦＳＭ的設計，



Stage 0 會將所有暫存器的值歸零，同時利用AXI Lite協議將所需資料傳送進來，並直接存到BRAM去。

因為這裡是傳11個tap值進來，會剛剛好存滿11。而其他像是data\_length或是ap\_start，我使用reg 暫存他們代表的意義。

每一次完成stage 0會有三種選擇，可以選擇繼續接收tap num，或是當AP\_start 升起時，開始 AXI Stream的傳輸，或是進行check 。

先說 Stage 7 ，這個階段會做一些地址判定，

如果是ap的地址，那麼根據是否做完600個data會有不同的值，對應的意義分別是ap\_done , ap\_valid。如果做完ap\_done會升起，如果ＦＩＲ停止運算，ap\_valid升起。

如果不是ap的地址，那便會從ＢＲＡＭ中提取tap，並透過AXI lite來傳送值回去給testbench，目的是做check的動作。

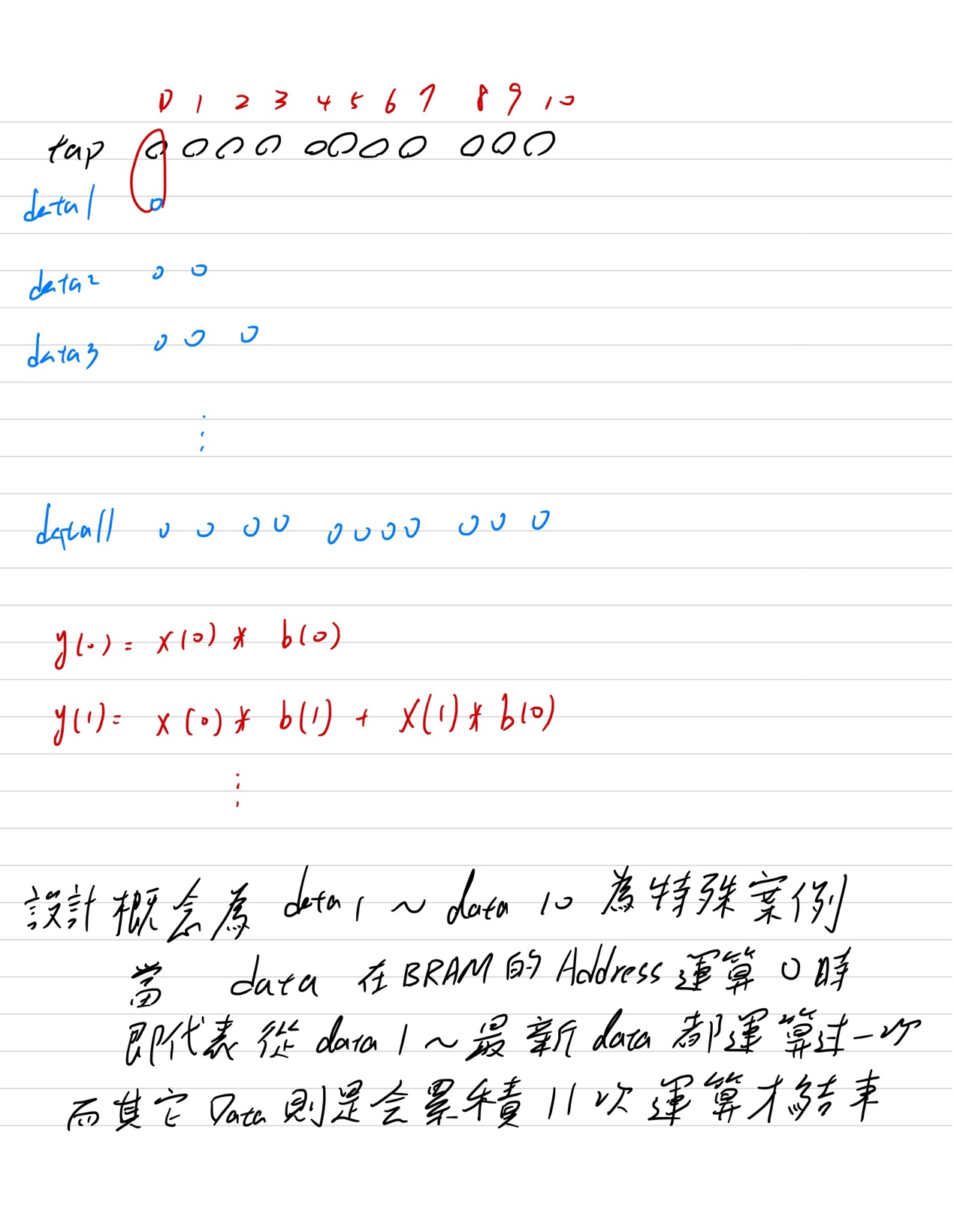
Stage 1 代表Data透過AXI Stream 傳送到ＦＩＲ中，再將其存到ＢＲＡＭ中。與此同時會將計數計加一，方便我觀察目前進行到第幾個資料。

另外這裡也代表新的一筆資料進來，所以累加器等需要歸零的暫存器都會在這粒進行歸零與初始化。

同時也對下一次新DATA的ＢＲＡＭ地址作更新。

Stage 2就是很簡單地從ＢＲＡＭ中提取值出來

Stage 3 就是將提取出來的值做運算。

Stage 4 會判斷是屬於前十筆資料還是其他需要做11次資料存取，會做這個判斷跟我的設計概念有關。另外他會再是否計算已結束，還是要繼續運算。

Stage 5 就是當運算結束後，將Data送回testbench。

Stage 6就是判斷是否還有資料需要運算，如果有，那就回Stage1，若沒有就Stage 7 等待測試。另外一種情況是因為testbench想要測試AP\_valid的狀態，因此當還沒運算結束但卻想測試時，我有再拉一條線去Stage 7等待測試，阿如果測完，發現還沒運算完，就會再回去Stage 1繼續新一輪的運算。

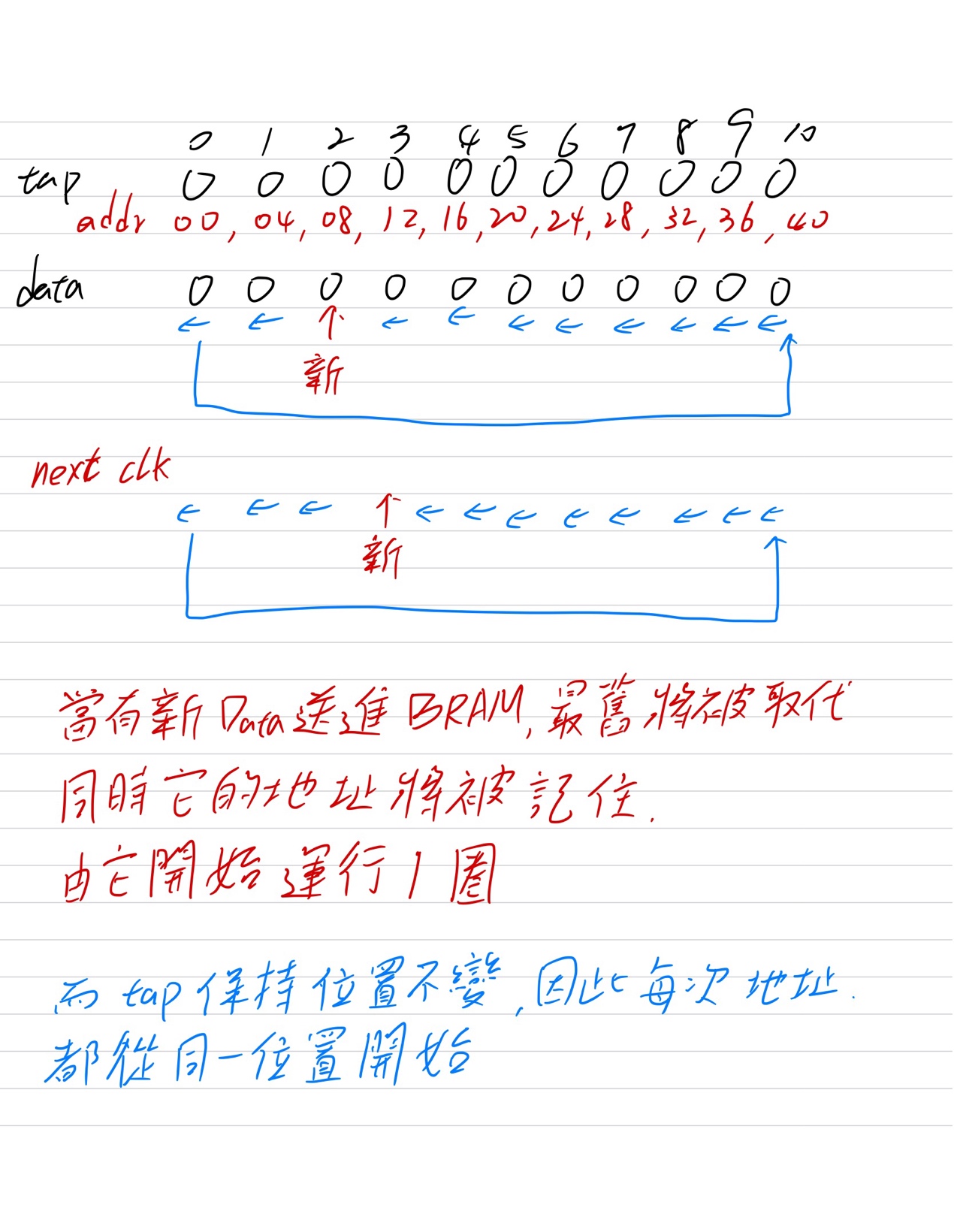


How to receive data-in and tap parameters and place into SRAM

AXI-Lite 我會做handshake，當valid為1，將ready拉起，此時已完成handshake，接著接受data的同時，直接將其存入BRAM。但如果不是存tap，那我就不會存入BRAM，因為也用不到。

AXI-Stream，其概念應該是完成一次handshake就可以不停地送值，達到類似Burst的效果，但因為我的一次計算就需要大量的時間，根本用不到他的優點，因此我還是會對每一次data進來時做一次handshake。送值出去也會再做一次handshake。

How to access shiftram and tapRAM to do computation

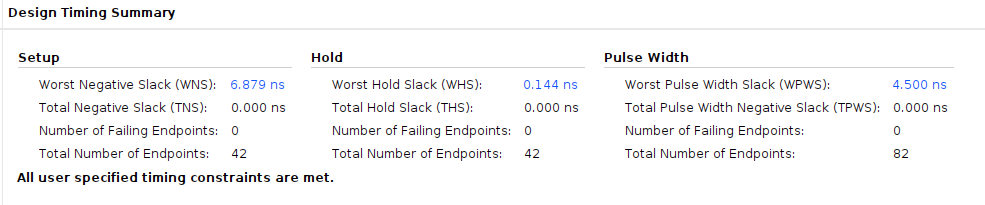


但正如前面所提到的特殊前十筆資料，我會做另外的處理，以data的地址作為判斷依據。

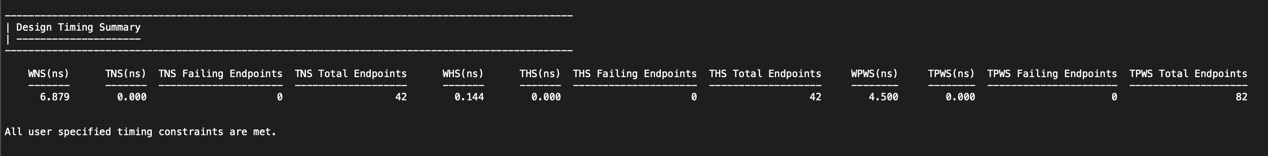
How ap\_done is generated

在stage 7都會判斷是否做完，若做完就會將ap\_done拉起，若沒做完就不會。

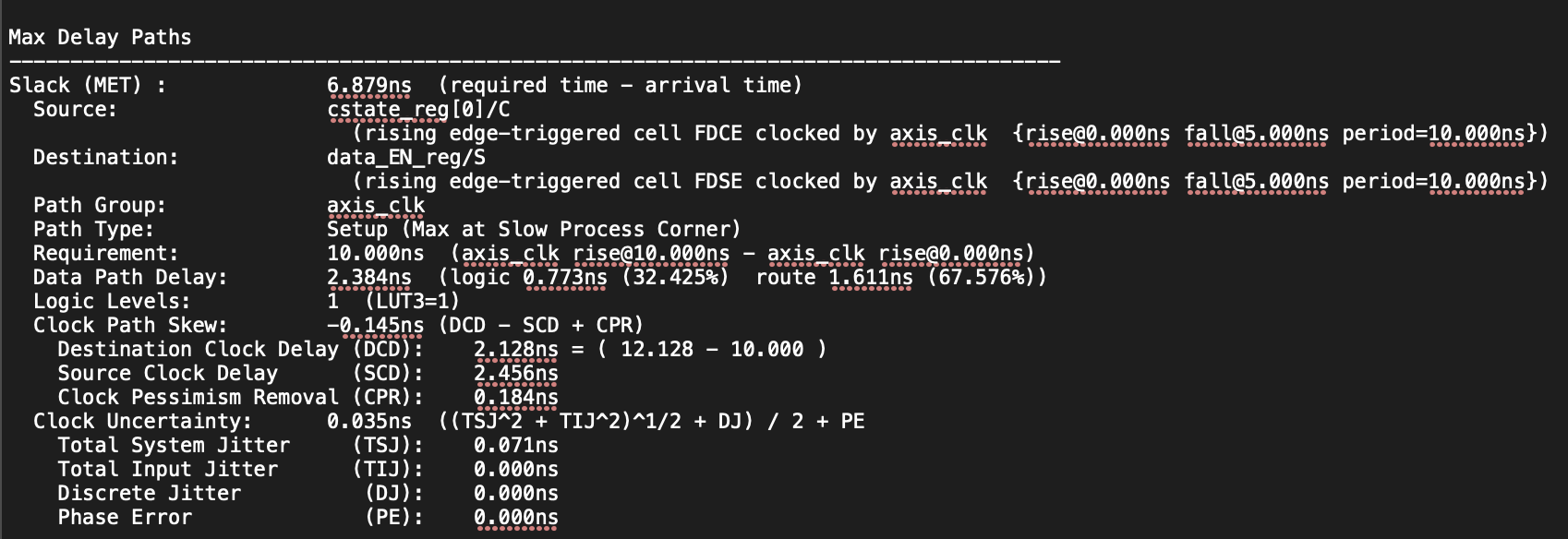
Slack



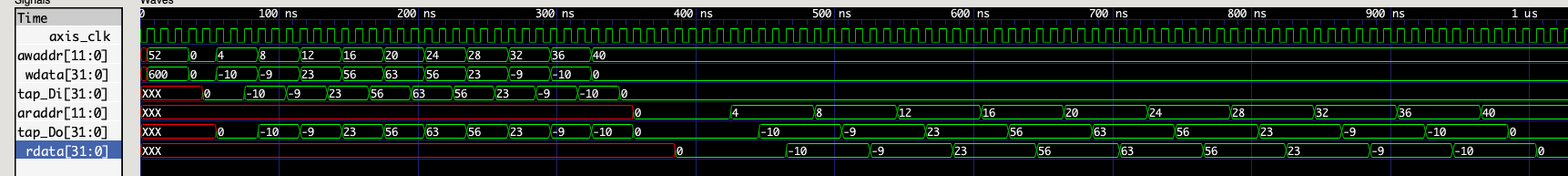
Timing Summery



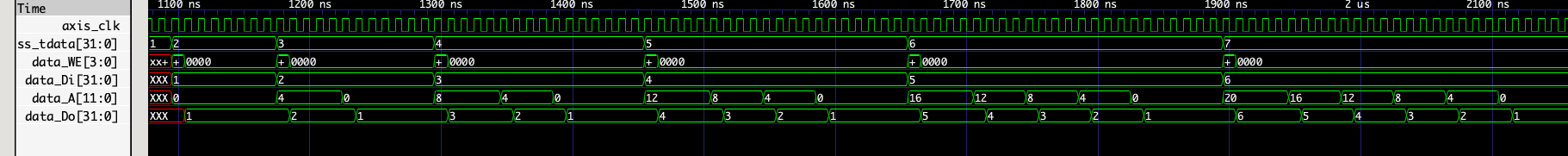
Max Delay Path



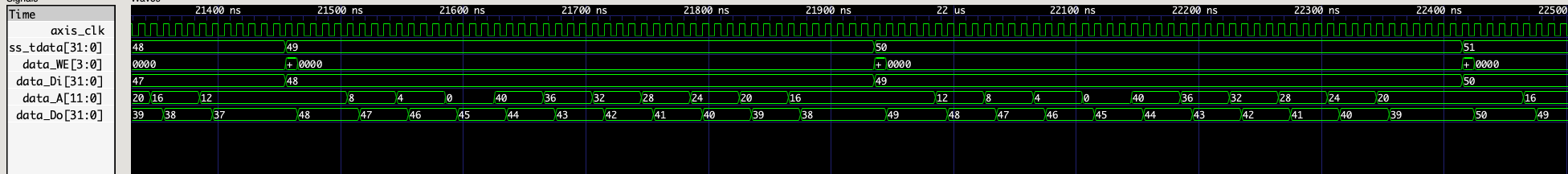
Read the tap from testbench and write it into BRAM



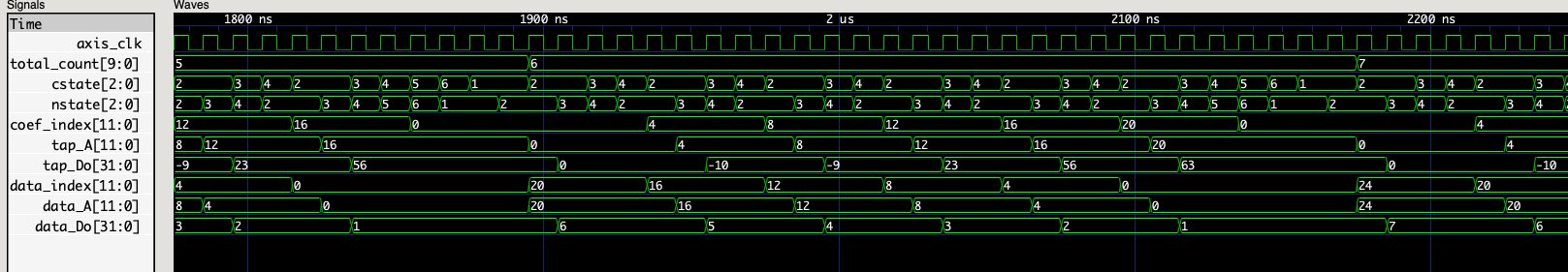
Read from BRAM and send back to testbench



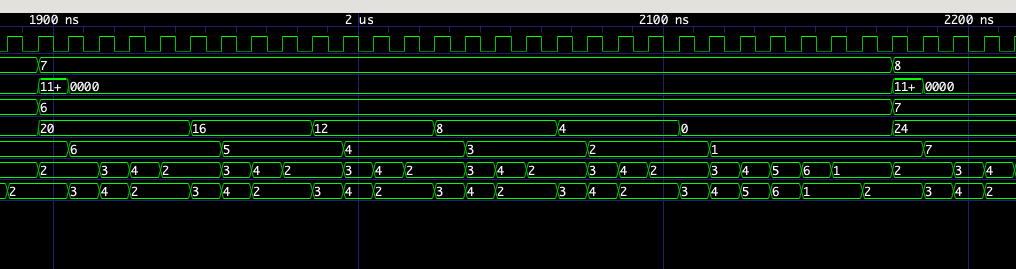
Write the new data from testbench into BRAM



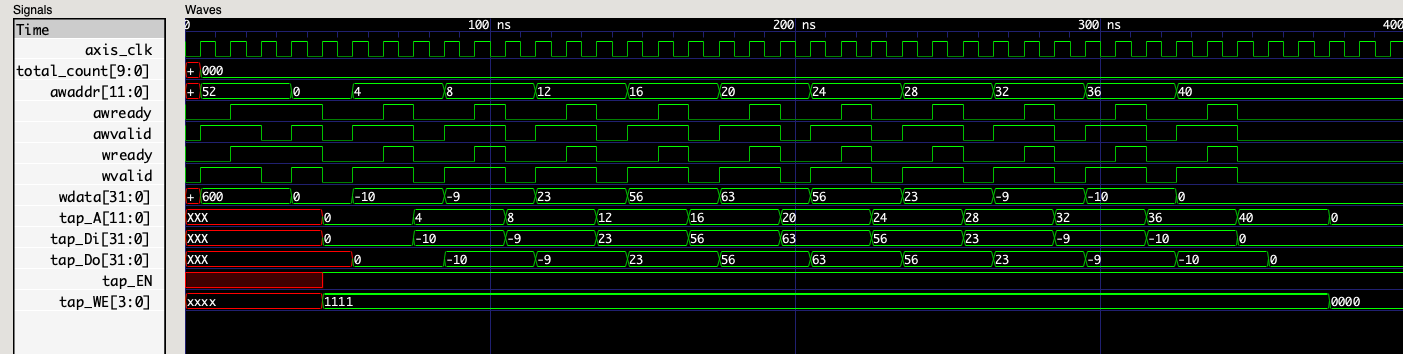
The data and tap read from BRAM



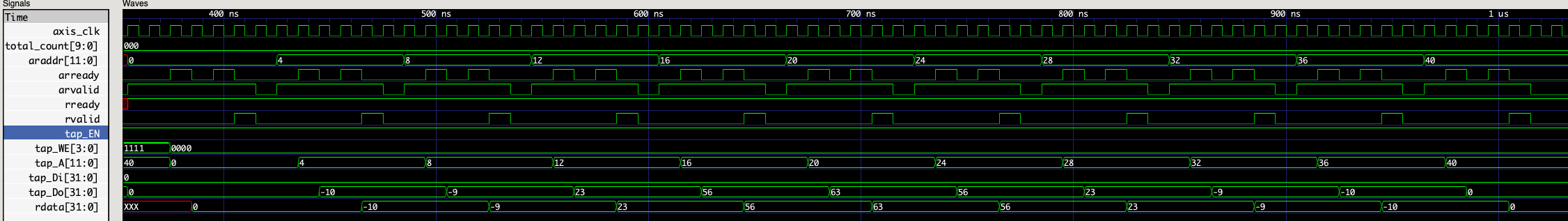
The data and tap read from BRAM but for the first 10 data



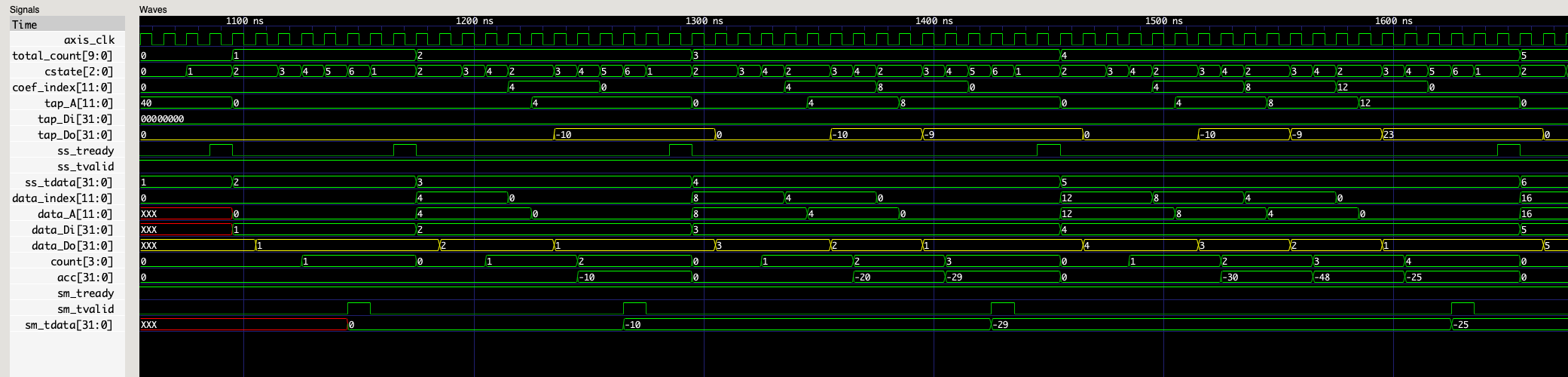
AXI-Lite write Handshake



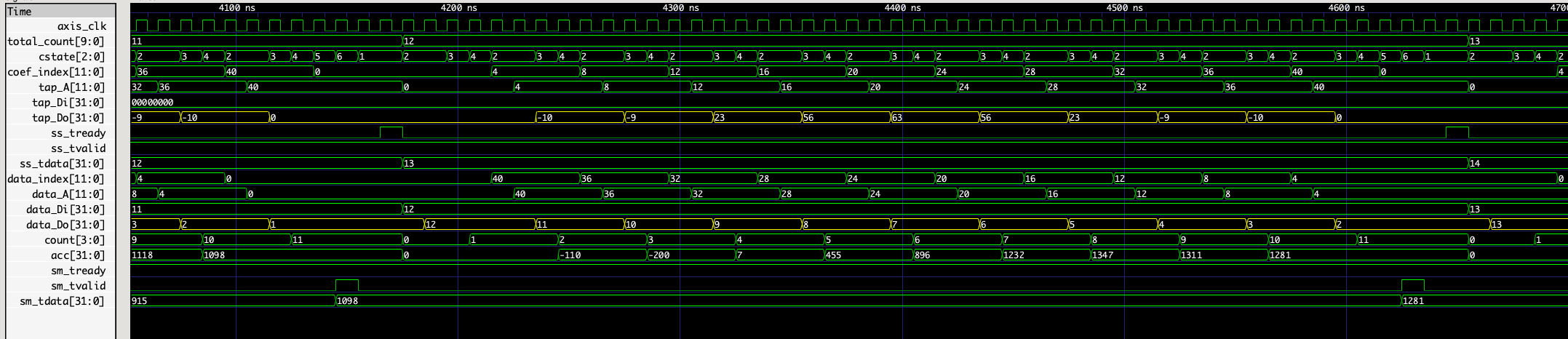
AXI-Lite read Handshake



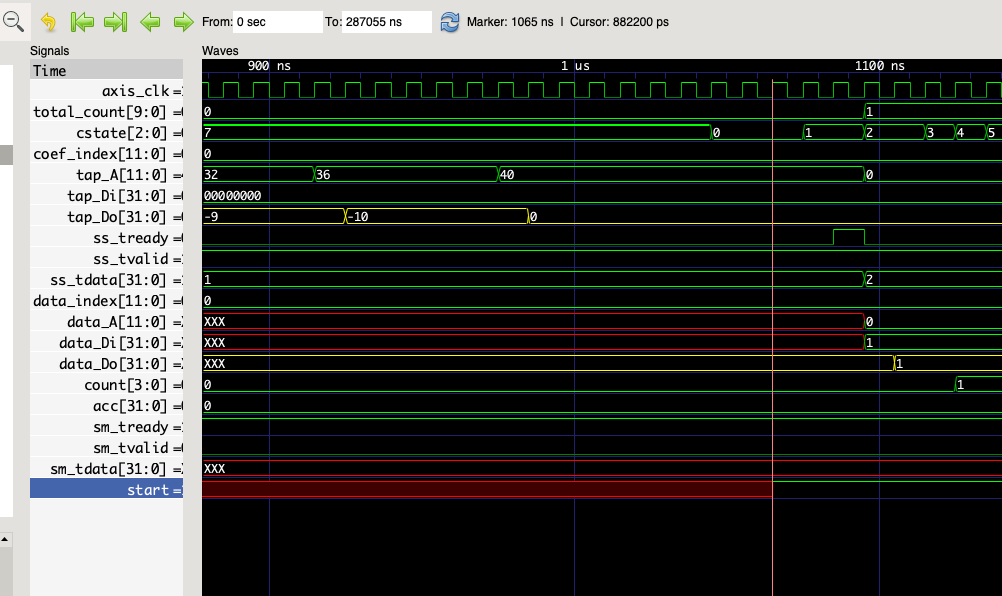
Calculate for the acc with first 10 data



Calculate for the acc with other data

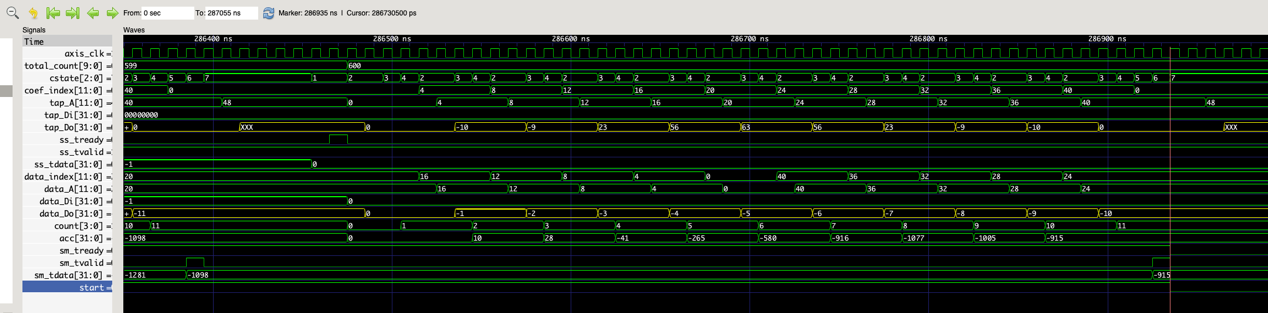


Ap\_start



­­­­­­­

AP\_done



心得：

此作業的第一步是先了解AXI-Lite與AXI-Stream的協議，對於handshake的概念也是一個新的學習。

第二步是了解BRAM的概念，並如何去運用與接線。是一個能在同一個CLK給他input又輸出output 的記憶體，很酷。

第三步是了解如何運用這僅有的11 words記憶體來儲存data與tap，取值並運算，對於address的存取與轉換，我認為關鍵在於我用了三個reg來存取地址，第一個是存去新data的地址，另外兩個分別儲存data運算時的地址與tap運算時的地址。

第四步就是對於ap這個東西該如何去接收與判定。

我一開始並沒有使用ＦＳＭ的設計，因此狀態寫得很亂，但還是有寫出來，但因為繳交期限有延後，因此我有再寫一個FSM的版本，我自己也比較滿意FSM的版本，寫出來比較清楚，對於整個流程也更加清晰。

遲交原因：

我是因為看到繳交期限有修改，所以才延遲繳交，如果事先有說明清楚，我會在昨天繳交。當然，學校停電也有影響到一部分，因為我都是在學校的電腦寫作業，但我是因為看到繳交期限有延期，所以才沒有及時繳交，如果只要在繳交時間前完成作業繳交，我想我都會不停地去嘗試更好的設計，即便最後並沒有獲得更好的成績。

希望助教能通融並理解我們，停電與公告都是我遲交的原因，謝謝。