國 立 台 灣 科 技 大 學

計 算 機 組 織

指 導 教 授：陳雅淑 教 授

**------------------------------------------------------------------------**

計算機組織作業PA1報告

班 級 ： 四電機三甲

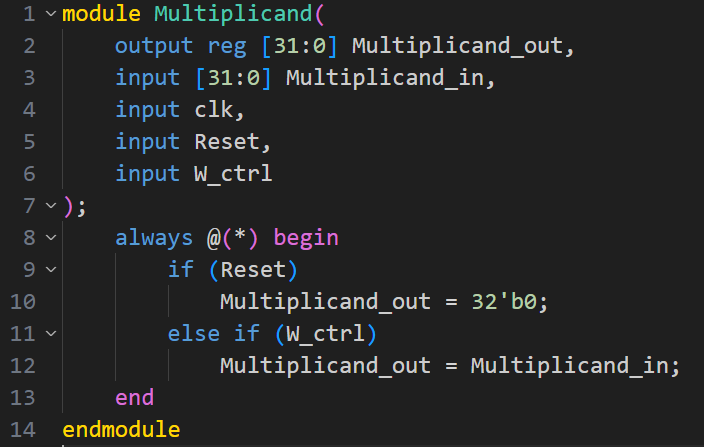
學 生 ： 呂佳祐

學 號 ： B11107055

乘法器 ： Area: 2408.630、Slack: 4.3850

除法器 ： Area: 2570.358、Slack: 4.5649

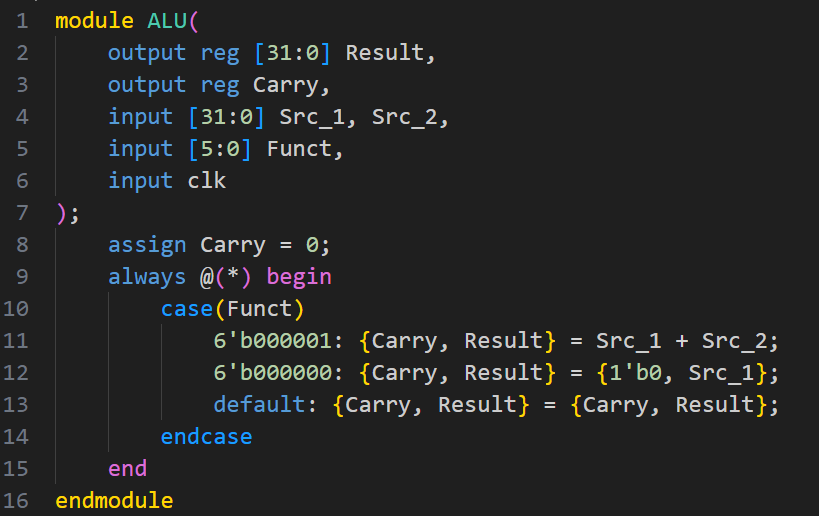
1. Multiplier
2. 32-bit Multiplicand Register



Multiplicand.v程式碼

Multiplicand Register用於暫存外部輸入之被乘數值。若Reset為1時，則輸出至ALU的Multiplicand\_out重設為0；若W\_ctrl為1時，則暫存器會將被乘數值寫入進ALU。

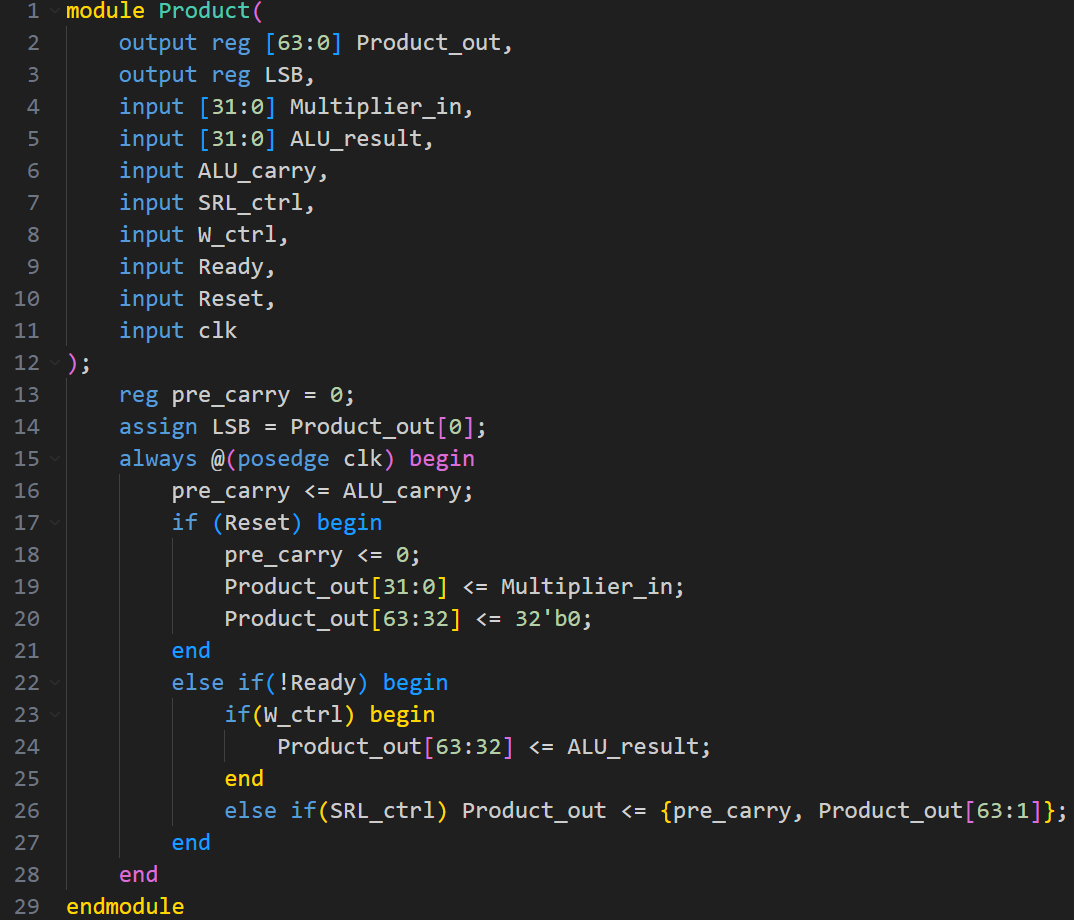
1. Arithmetic Logic Unit



ALU.v程式碼

ALU將輸入訊號Src\_1與Src\_2根據Funct值做對應的運算。當Funct為”000000”時，將輸出結果Result重設為Src\_1的值，並將Carry重整為0；若Funct為”000001”時，將Src\_1與Src\_2相加，並將加法結果傳入Result，進位位元傳入Carry。

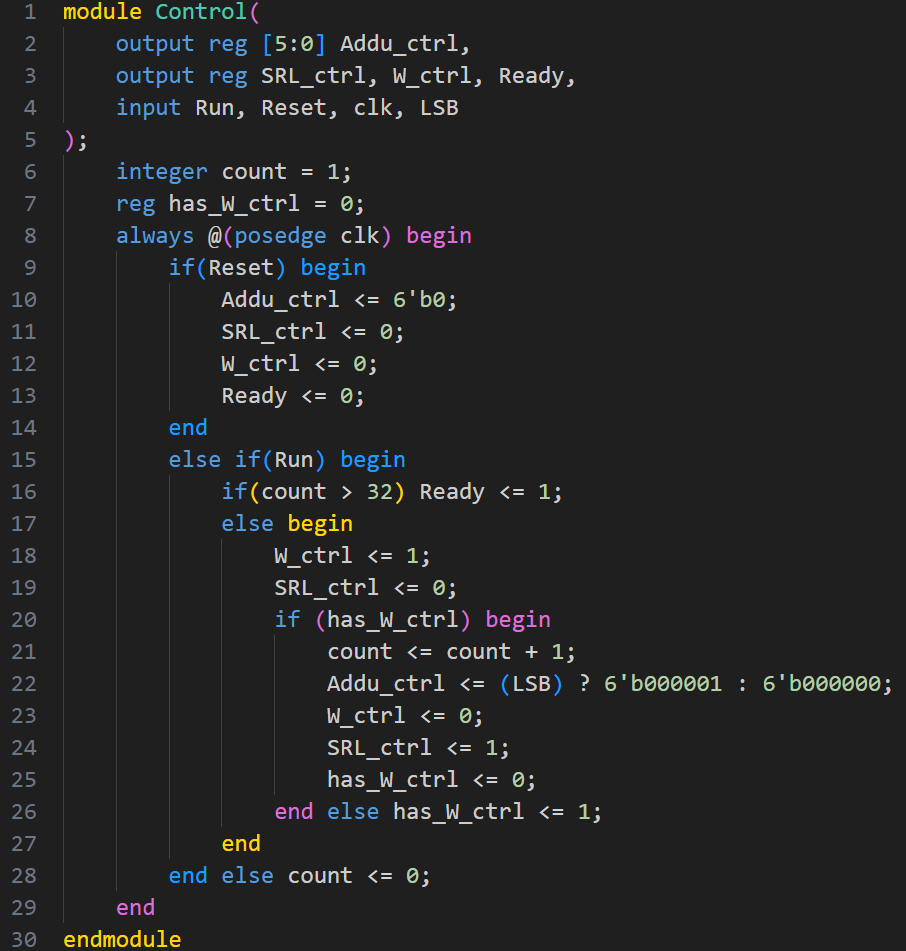
1. 64-bit Product Register



Product.v程式碼

Product Register用於儲存乘積。為了節省元件，將乘數合併進暫存器的前32-bit裡，並透過Controller的控制，經過一系列演算後，會得到正確乘積。暫存器提供乘積的第一位元給控制器參考，使其得以決策。當Reset為1時，初始化暫存器，將旗標重設為0，並將輸入的乘數傳進乘積的前32-bit，並將後32-bit設為0。程式演算未完成時，若W\_ctrl為1，則將ALU的結果寫進暫存器；若SRL\_ctrl為1時，則將乘積向左移一位元。

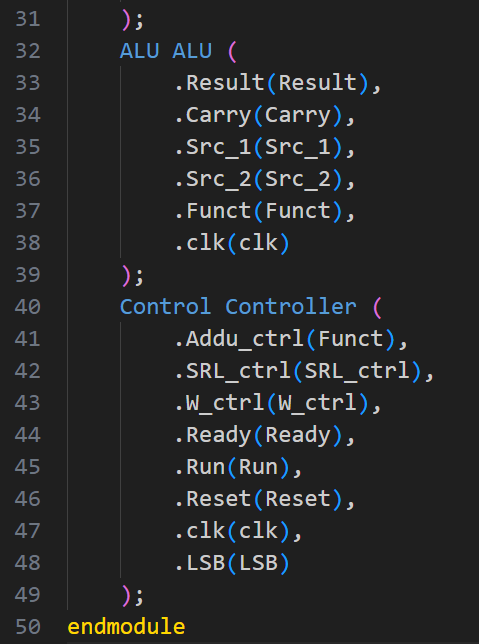
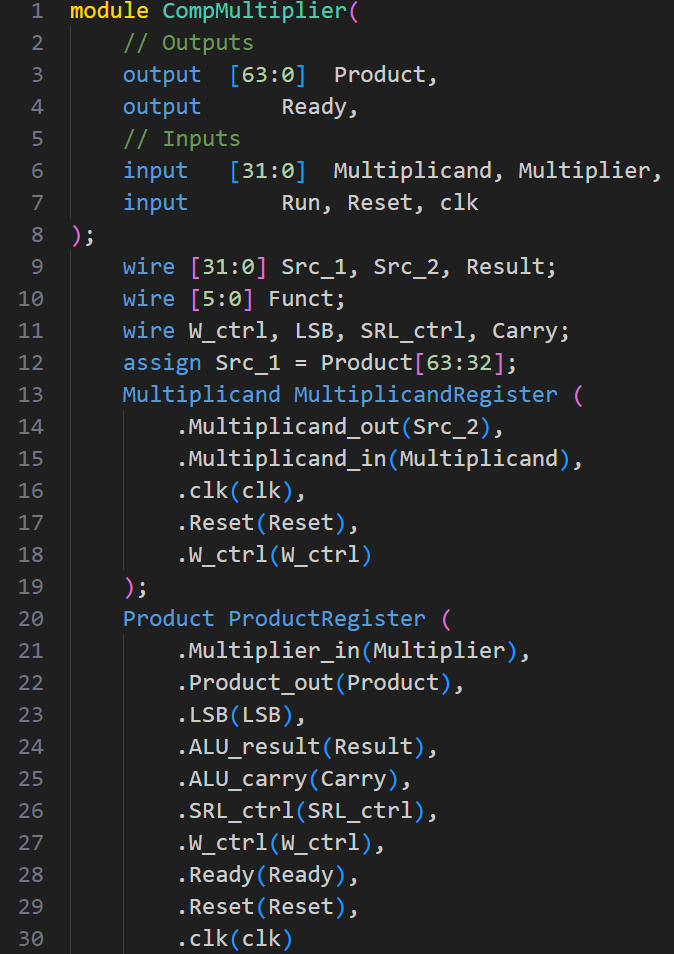
1. Controller



Control.v程式碼

Controller根據輸入、現在狀態與旗標，決定其輸出，使其控制各模組以完成乘法演算。控制器在每一repetition會先檢查乘積的第一位元是否為1，若是，則向ALU下加法指令，令被乘數與乘積左半部相加，並將相加結果放回乘積左半部，再將整個乘積向右移；若不是1，則直接將整個乘積向右移。每次repetition都會先將W\_ctrl設為1，使Product Register都能先重載上回計算後，ALU輸出的結果。每次repetition都會計數迴圈次數，若超過32次，則代表運算完成，將Ready設為1。

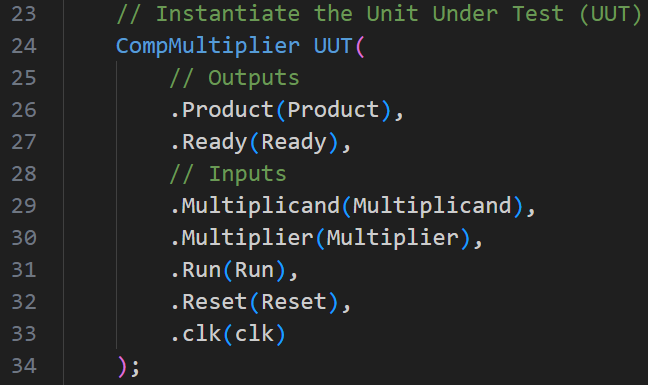
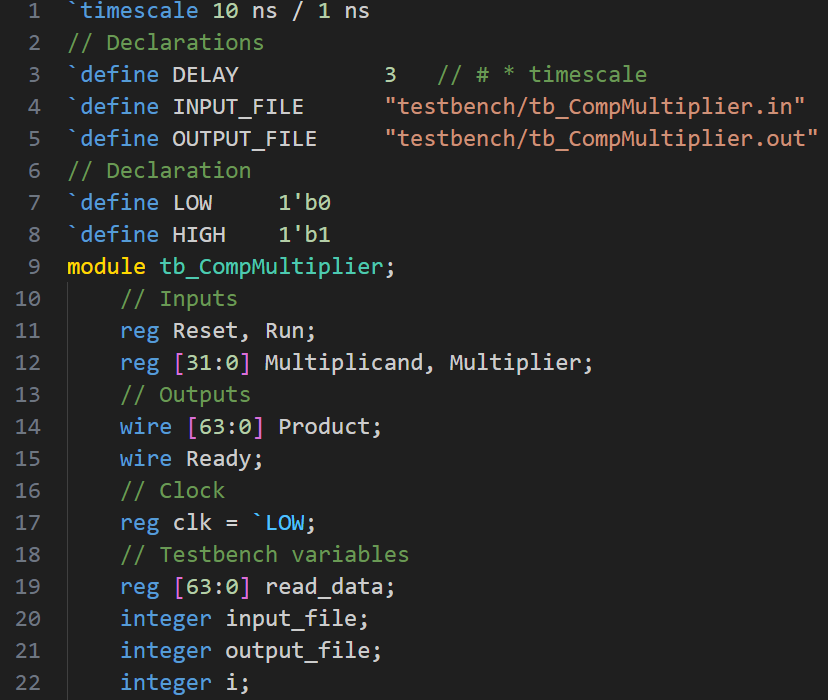
1. 32-bit Unsigned Complete Multiplier
   1. 主程式



CompMultiplier.v程式碼

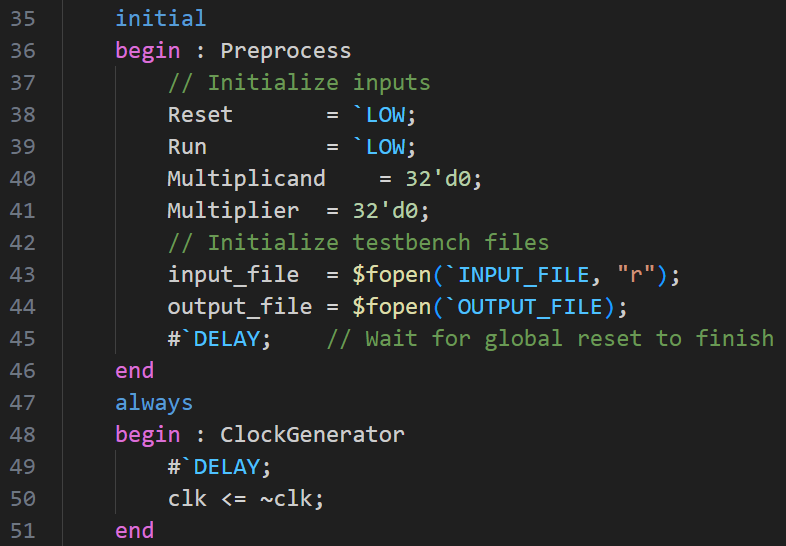
CompMultiplier將各module透過內部接線，使對應接腳相連，組合成一個功能完整的乘法器。

* 1. Testbench and Simulation



tb\_CompMultiplier.v程式碼—初始化

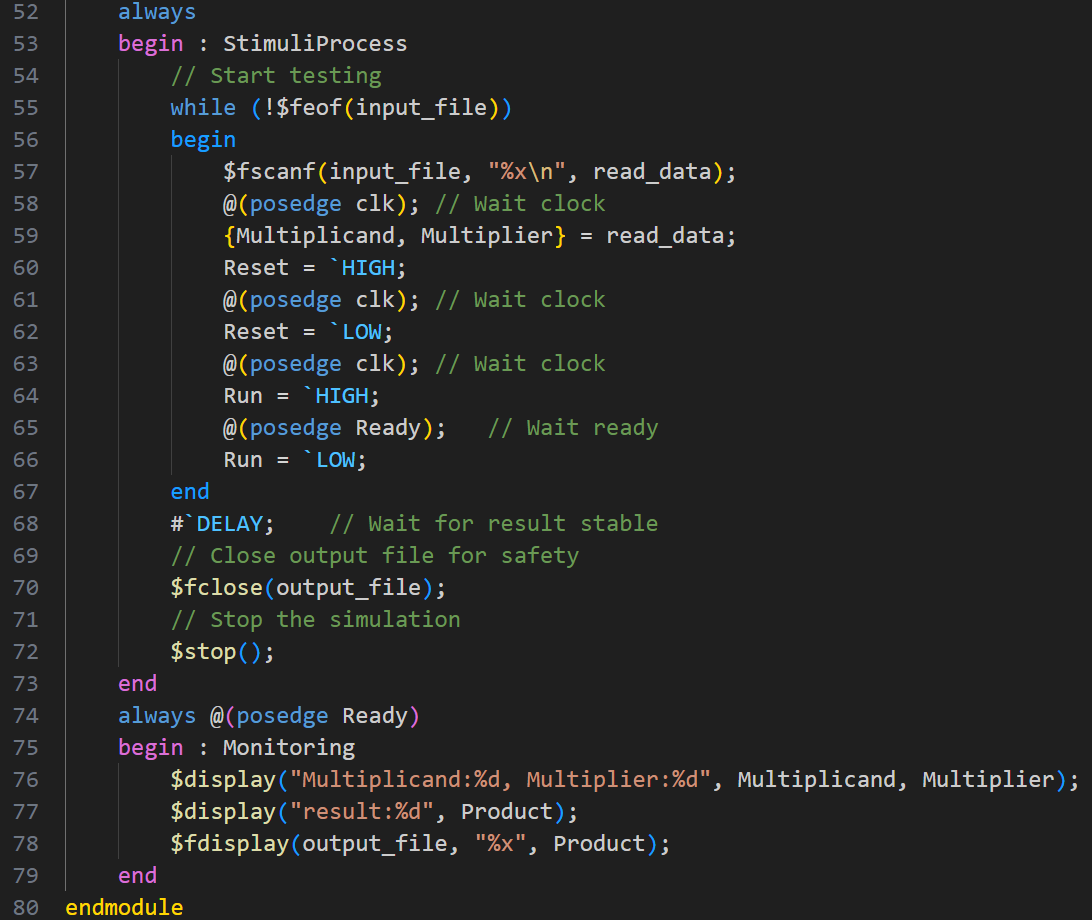
設定時間單位10ns、時間精度1ns。設定各常數與變數，並接線至CompMultiplier模組。



tb\_CompMultiplier.v程式碼—檔案處理與時鐘

Preprocess負責將各變數設為0，並開啟tb\_CompMultiplier.in、tb\_CompMultiplier.out檔案。

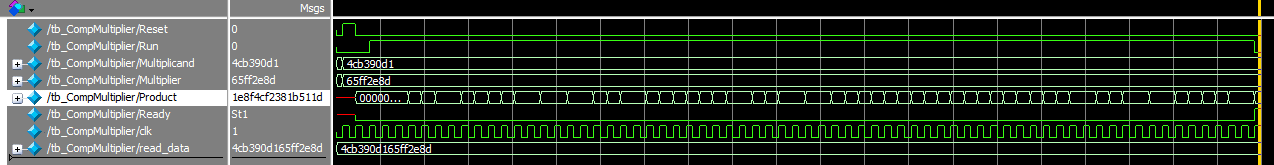
ClockGenerator產生週期為6個時間單位的時鐘。



tb\_CompMultiplier.v程式碼—模擬測試與輸出結果

StimuliProcess將從tb\_CompMultiplier.in讀取的測試向量，輸入至CompMultiplier。每筆輸入資料都會等到Controller將Ready設為1，才會測試下筆資料。

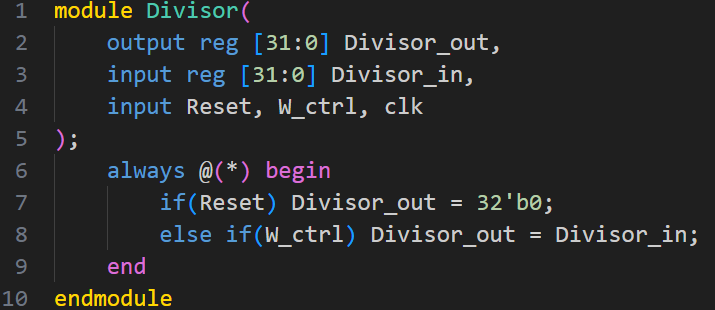
Monitoring將運算結果輸出至終端，並一一儲存於tb\_CompMultiplier.out檔。



tb\_CompMultiplier.v模擬波型

可以從波型看到乘積在演算過程中變化了很多次，直到最後算出結果為止。從結果來看4CB390D1 \* 65FF2E8D確實是1E8F4CF2381B511D，而且Ready也有在運算結束後變為1，可以斷定乘法器運算功能正常。

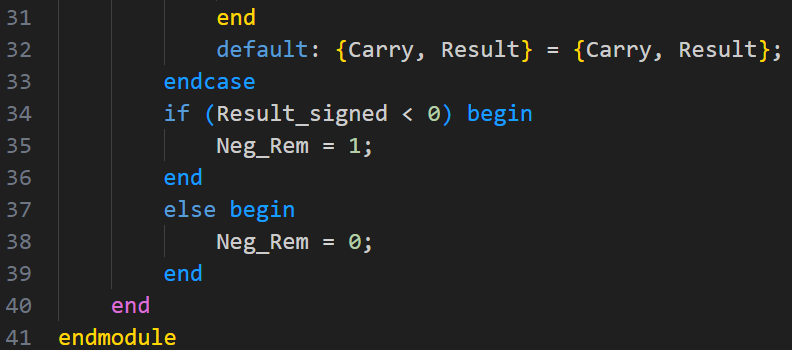
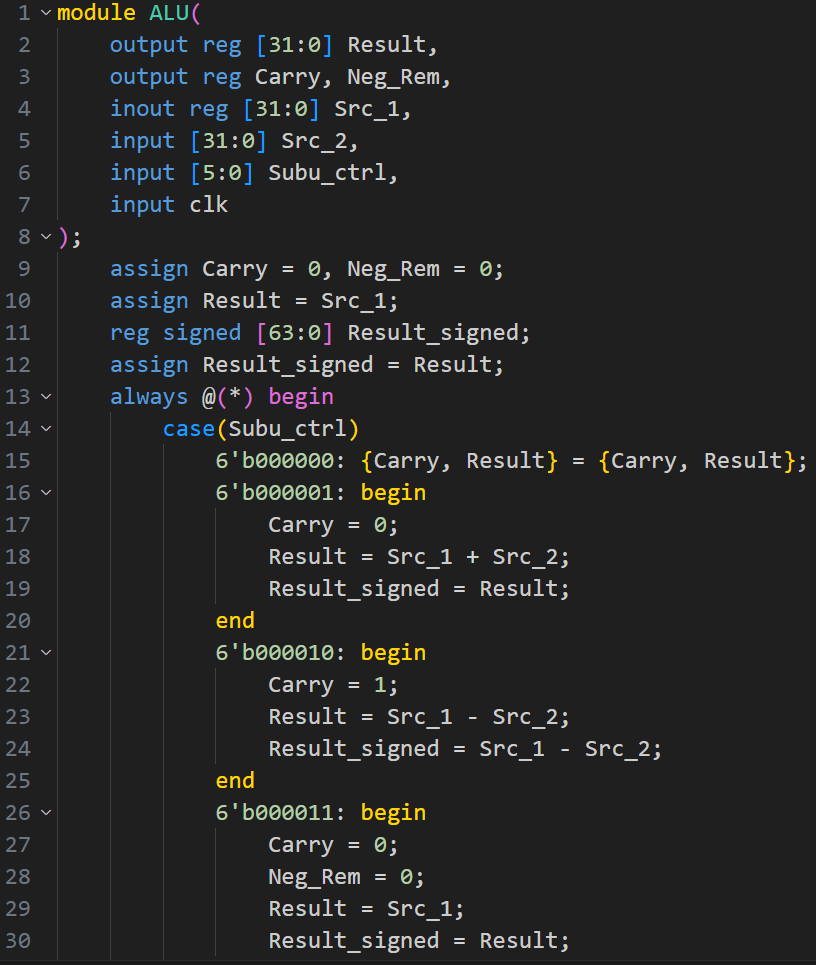
1. Divider
2. 32-bit Divisor Register



Divisor.v程式碼

Divisor Register用於暫存來自輸入的除法數訊號，並將訊號傳遞給ALU運算。當Reset為1時，Divisor\_out會被重設為0；當W\_ctrl為1時，則Divisor\_out會被設為輸入訊號Divisor\_in的值。

1. Arithmetic Logic Unit

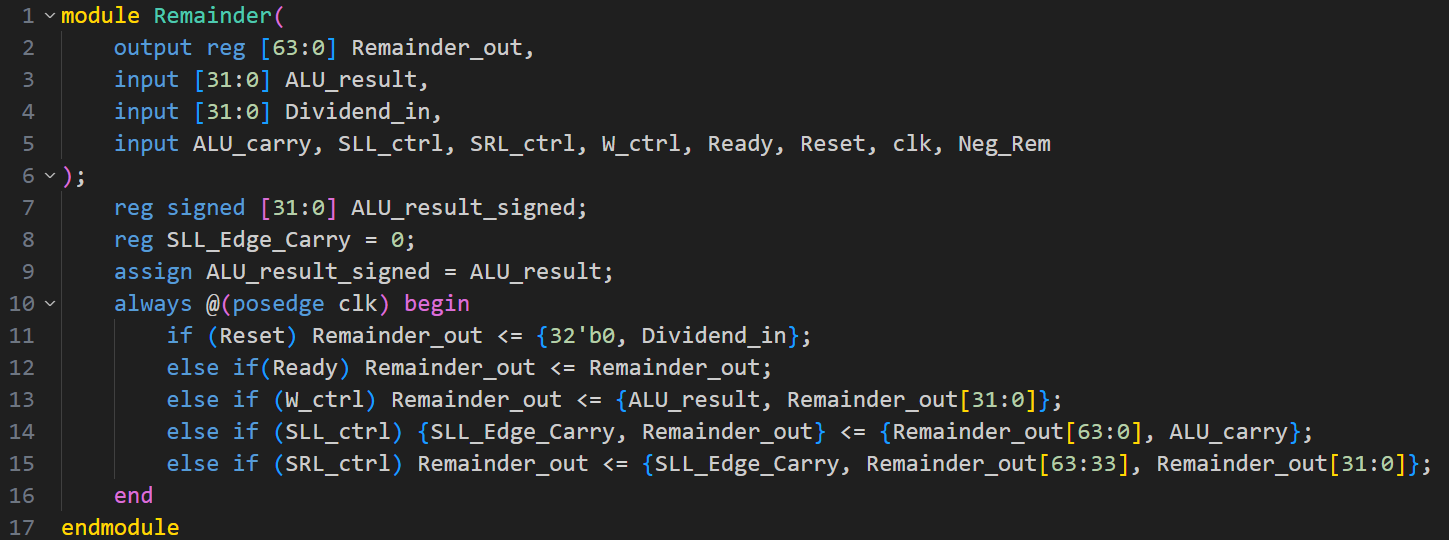


ALU.v程式碼

ALU用於對兩輸入Src\_1、Src\_2的值做運算，並依接收到的Funct值，對應其指定的運算功能。此外，為使Remainder Register與Controller得知運算結果是否為負數，以判斷應對措施，而輸出Neg\_Rem旗標。以下表格整理出各Funct值對應功能。

|  |  |
| --- | --- |
| Funct (Binary) | 對應功能 |
| 000000 | 使輸出結果與Carry保持原本狀態。 |
| 000001 | 相加兩輸入訊號，並輸出結果。使Carry保持為1。 |
| 000010 | 相減兩輸入訊號，並輸出結果。使Carry保持為0。 |
| 000011 | 重設所有輸出，使其回到初始狀態。 |

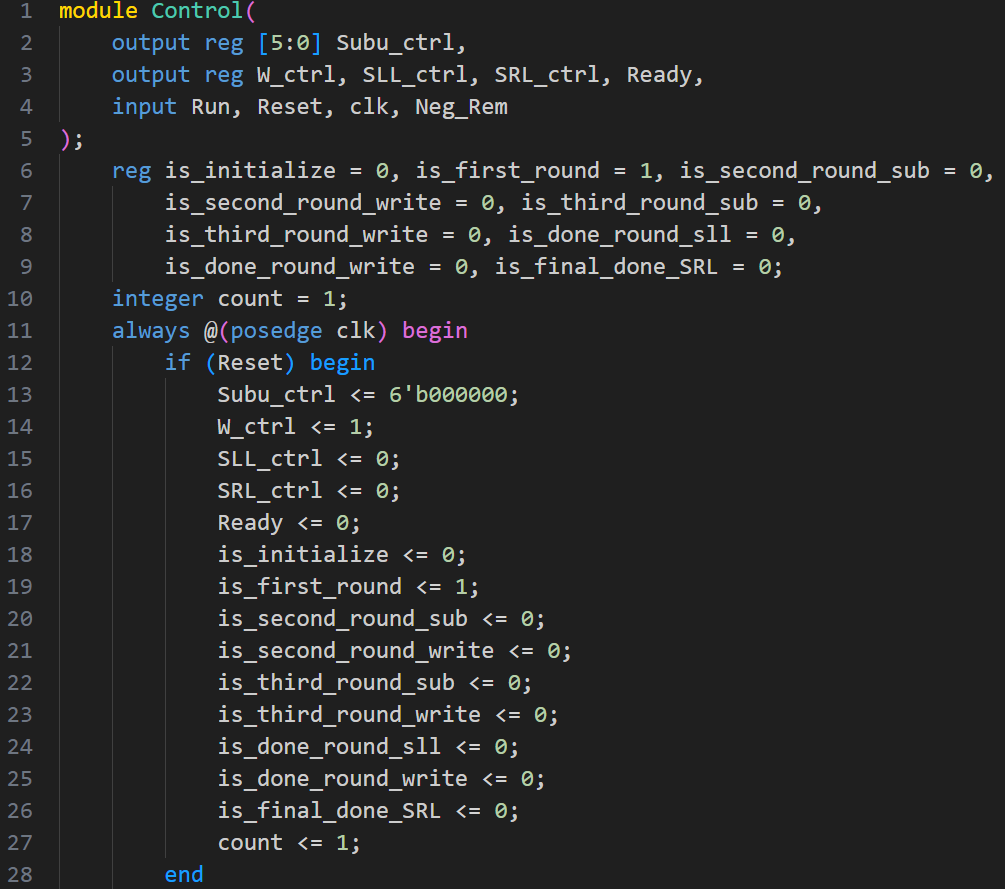
1. 64-bit Remainder Register



Remainder.v程式碼

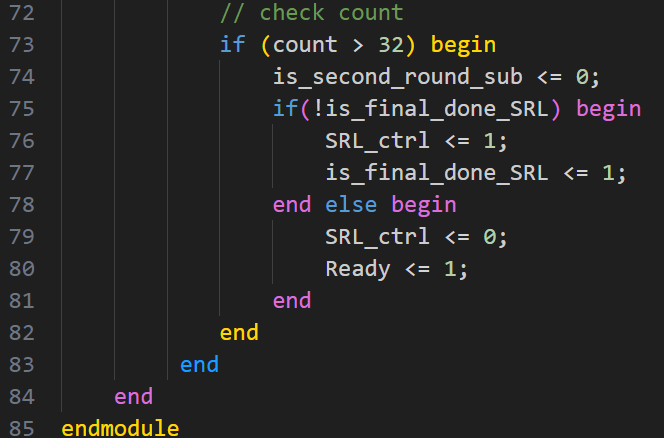
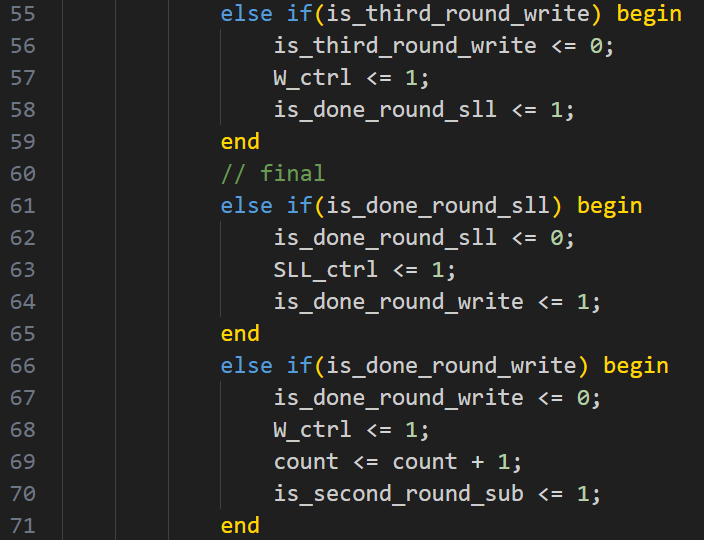
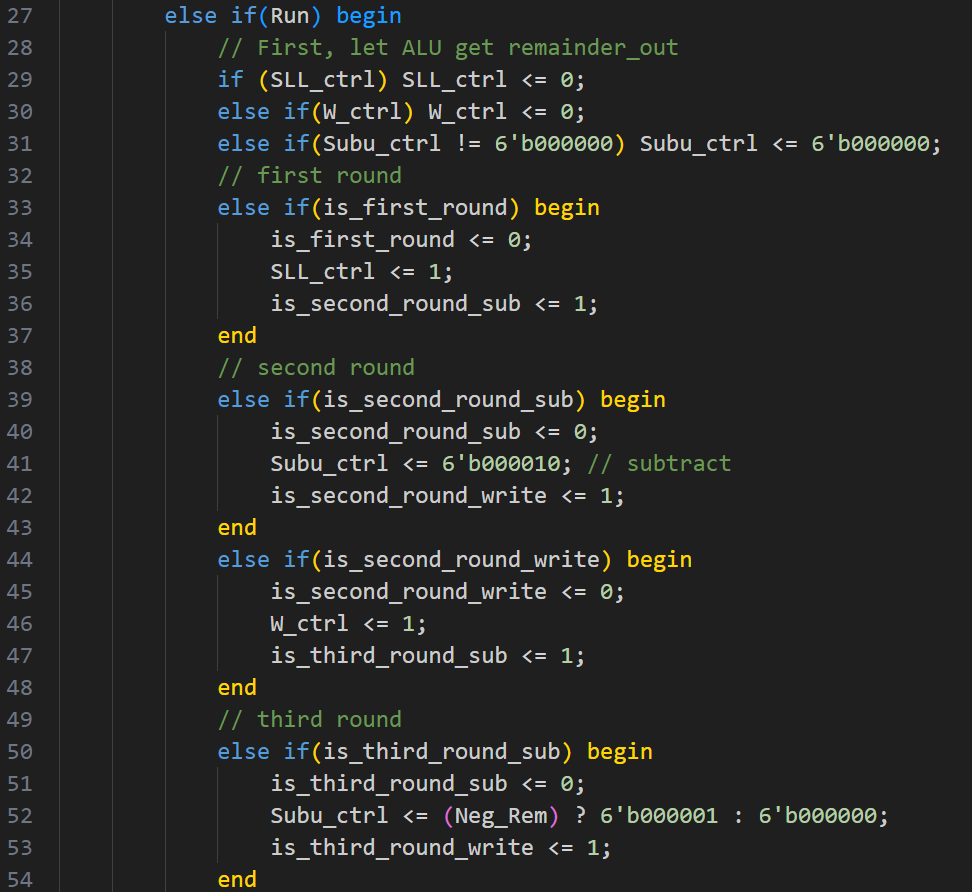
Remainder Register用於暫存餘數與商的值，為節省空間，一開始會將被除數放入暫存器的後32-bit中。當Reset為1時，Remainder\_out會被重設至左半部為0、右半部為被除數。若Ready為1時，代表整體除法運算完成，則Remainder\_out保持原本狀態。若W\_ctrl為1時，暫存器讀取ALU運算結果，並放入Remainder\_out左半部，右半部則維持原狀。若SLL\_ctrl為1，則整體Remainder\_out向左移；最右一位元會被移入Carry，當ALU相減後結果為負數，則Carry為0，若為正數，則Carry為1。且左移後會保存被移出的位元，以確保最後右移時不會出錯。若SRL\_ctrl為1，則僅Remainder\_out左半部往右移。

1. Controller



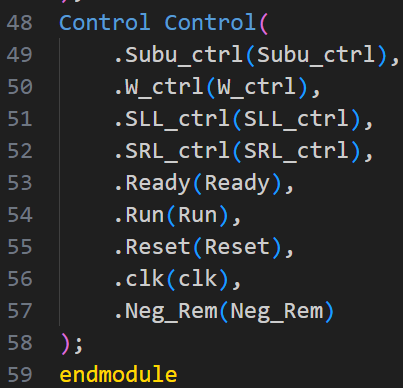
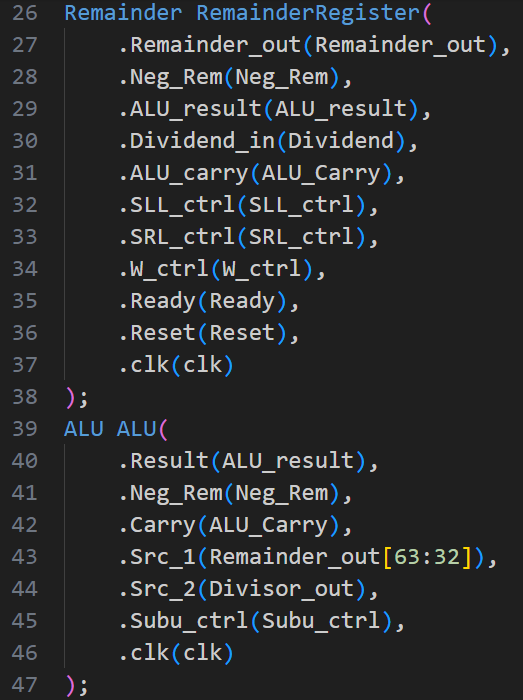
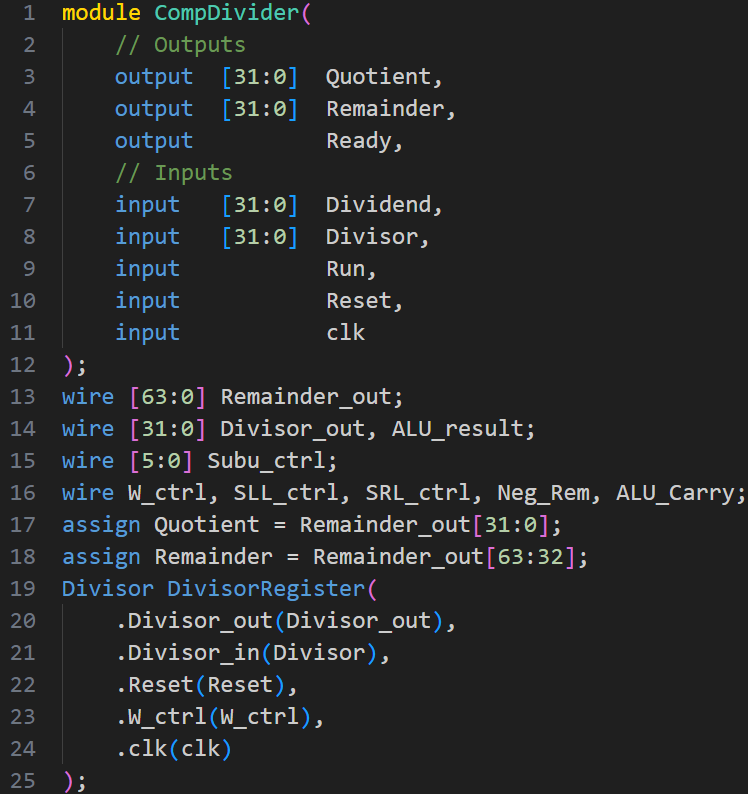
Control.v程式碼

Controller根據輸入值、當前狀態，判斷其應輸出的值，相當於個模組的樞紐。當Reset為1時，所有旗標及變數初始化。當Reset結束，開始執行程式時，會分各狀態一一執行。且每次W\_ctrl、SLL\_ctrl設為1後，會等待1個時鐘週期的時間，才進入下個狀態，這麼做是為了能讓Remainder及ALU正常讀入運算後的值。所有狀態執行完後，代表一個repetition的結束，這時會將計數器加一，並判斷迴圈次數有無大於32。若有，則會先將Remainder左半部向右移，再將Ready設為1；若無，則繼續進入下個循環，並從狀態二(second round)開始。



Control.v程式碼

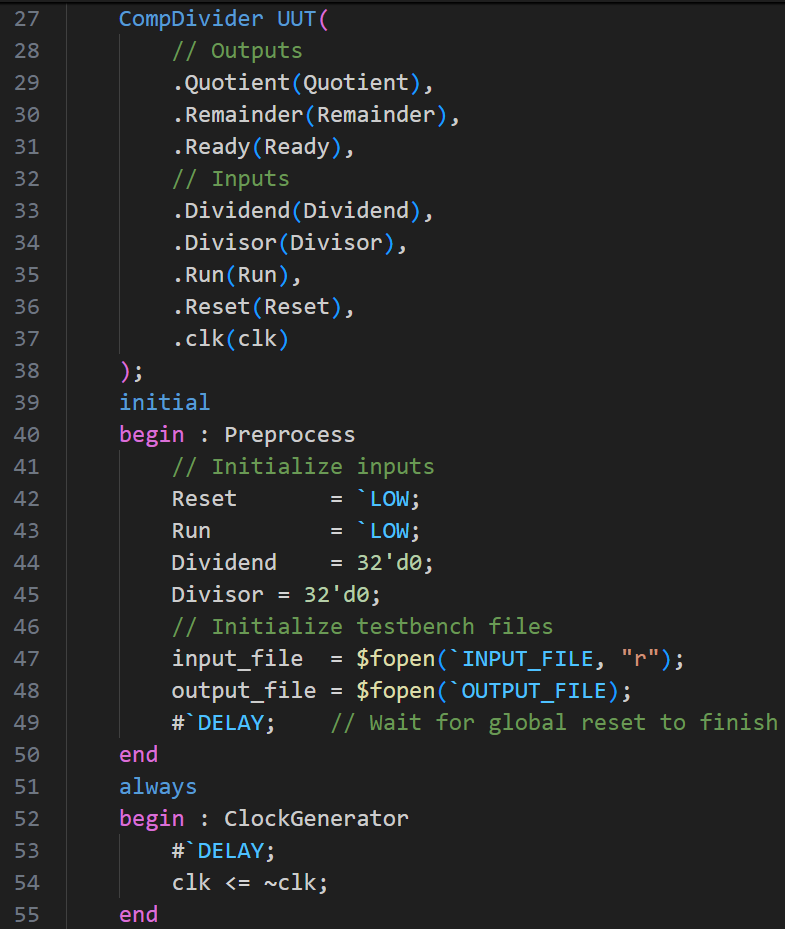
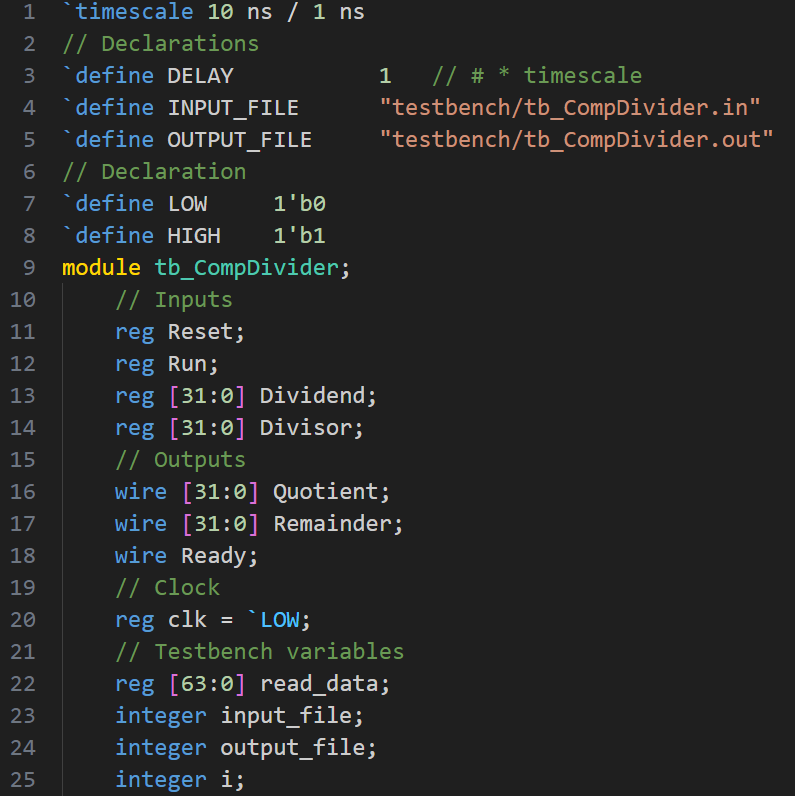
1. 32-bit Unsigned Complete Divider
   1. 主程式



CompDivider.v程式碼

CompDivider將各module以內部接線連接，組成一個完整的除法器。

* 1. Tesetbench and Simulation

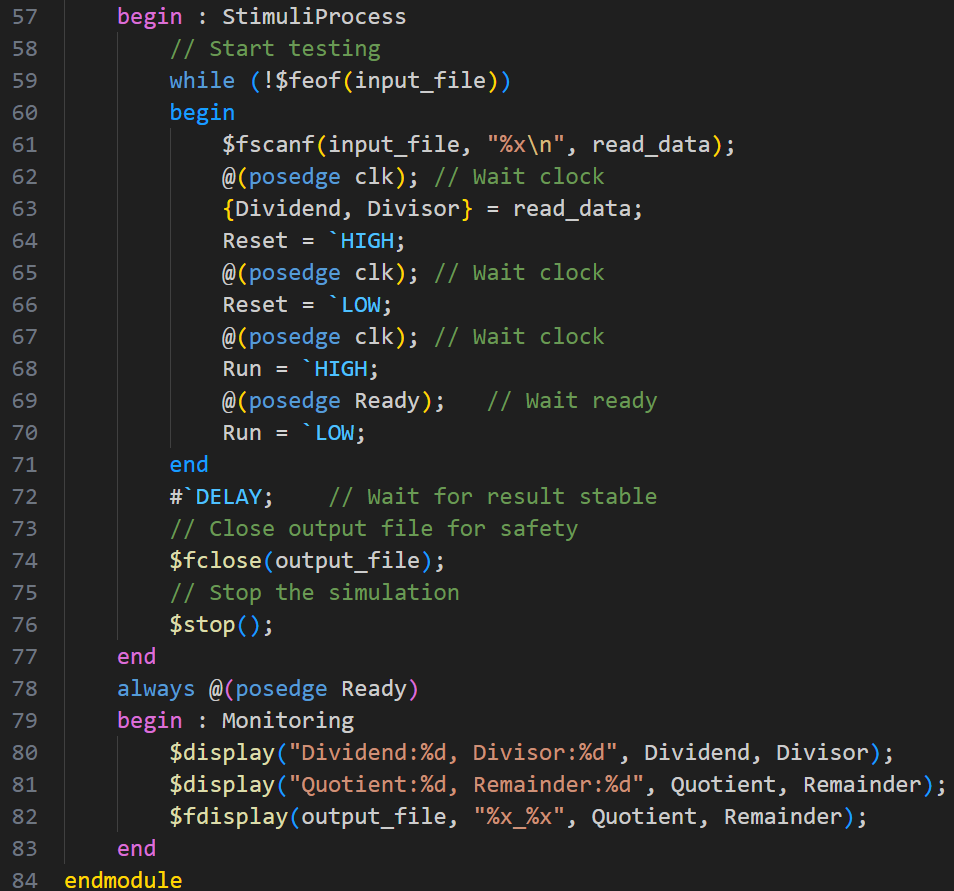


tb\_CompDivider.v程式碼—初始化、連接、預處理與時鐘設定

Testbench一開始將時間單位設為10ns、時間精度1ns。初始化各常數與變數，並接線至CompDivier。

Preprocess將輸入至CompDivider的訊號初始化，並開啟input檔與output檔。

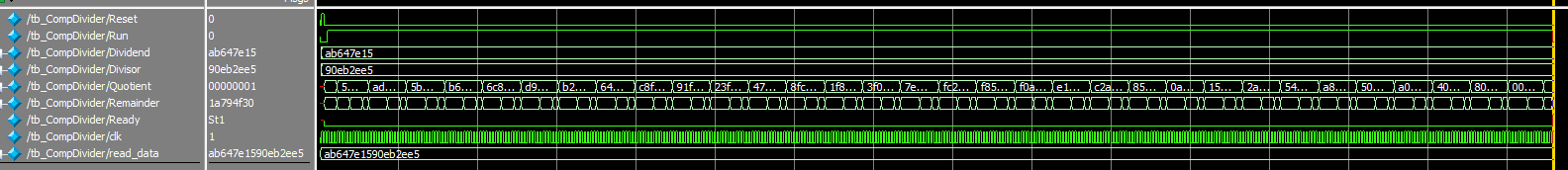
ClockGenerator設定時鐘週期為兩個時間單位。



tb\_CompDivider.v程式碼—模擬程序與輸出結果

StimuliProccess將input檔的測試向量一一輸入至除法器中，在每個資料運算時，都會等到Ready變為1，才會結束，送入下個訊號。

Monitoring會將結果輸出至終端並記錄於output檔中。



tb\_CompDivider.v模擬波型

可以看到輸入的被除數為90EB2EE5、除數為AB647E15，其模擬後所得的商為1、餘數為1A794F30，跟理論值相符，且Ready也有在運算完成後被設為1，可以判斷此除法器功能正常。

1. Conclution and Insight

乘法器與除法器的運作方式，因為要考慮到電路的限制與晶片的面積，所以沒有那麼的直覺。但仔細觀察這些算法的步驟，對比實際上用手去算，可以發現算法有它本身的邏輯在，在此可以感嘆能想出這些算法的人，他們是多麼的厲害。