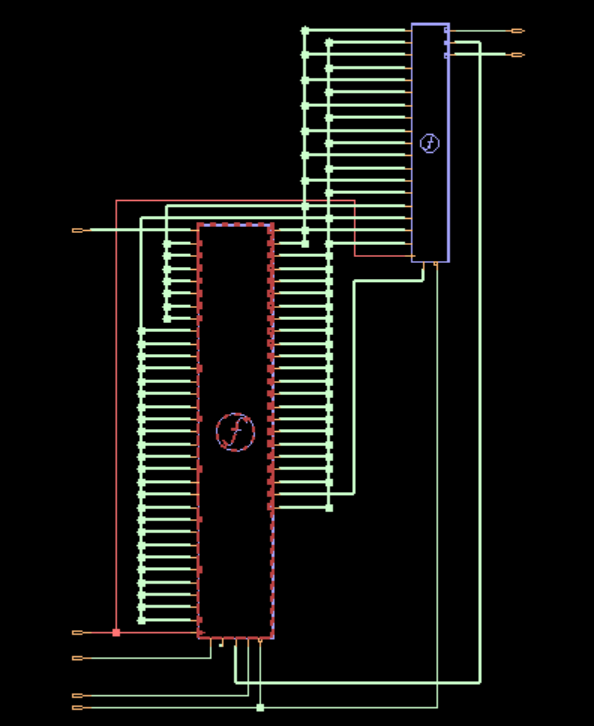
**Block Diagram**

out\_valid

out\_data



image\_valid

rst\_n

filter\_valid

clk

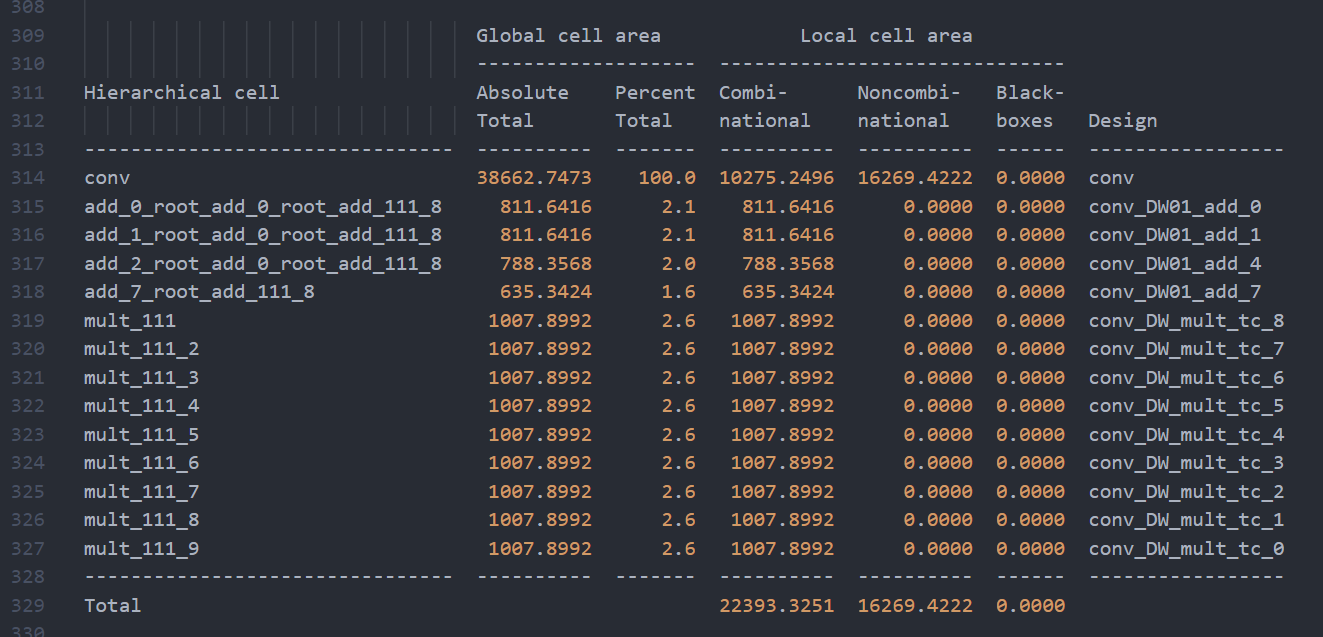
in\_data

shift operations on image and filter

out\_data & multiplication

operations

**02乘法器**



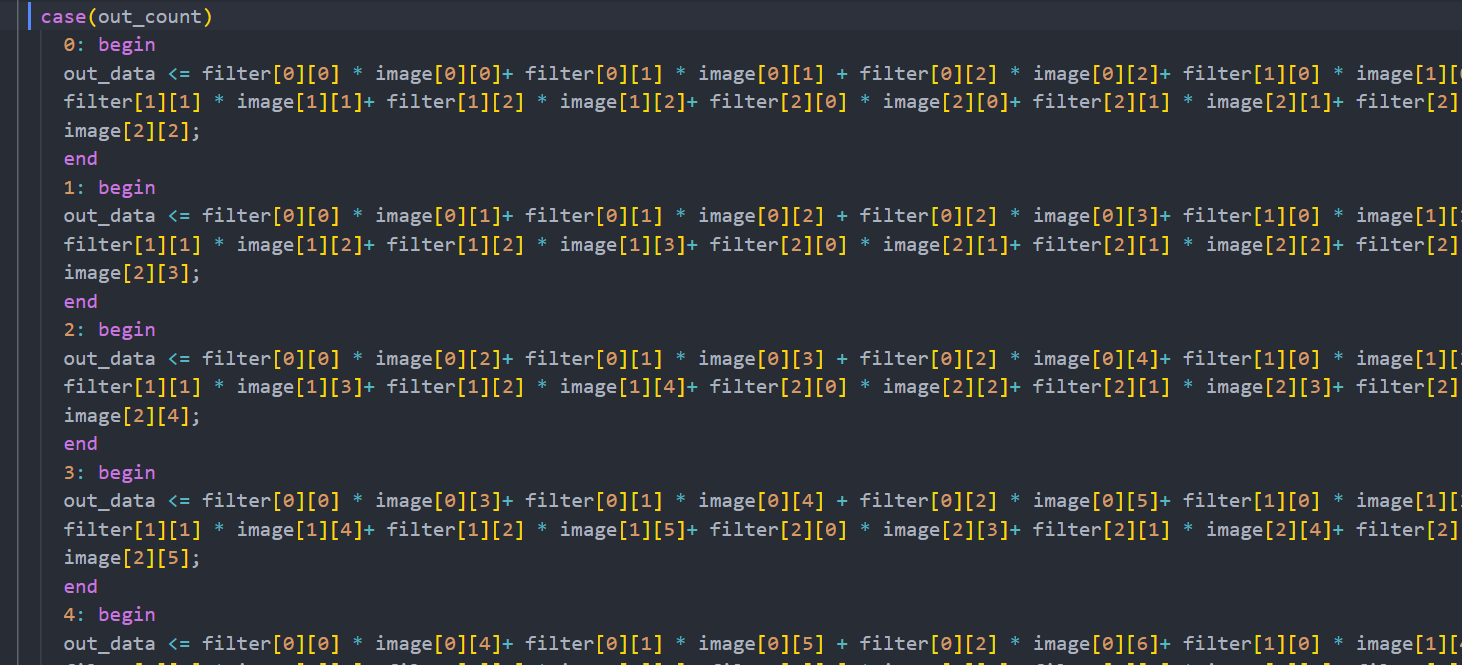
最終的設計我用到了9個乘法器，和filter的格子數一樣 (對filter中每個數字做乘法)，而可從這個synthesis report中看到，一個乘法器的面積約為1000，占了combinational area很大的一部分。

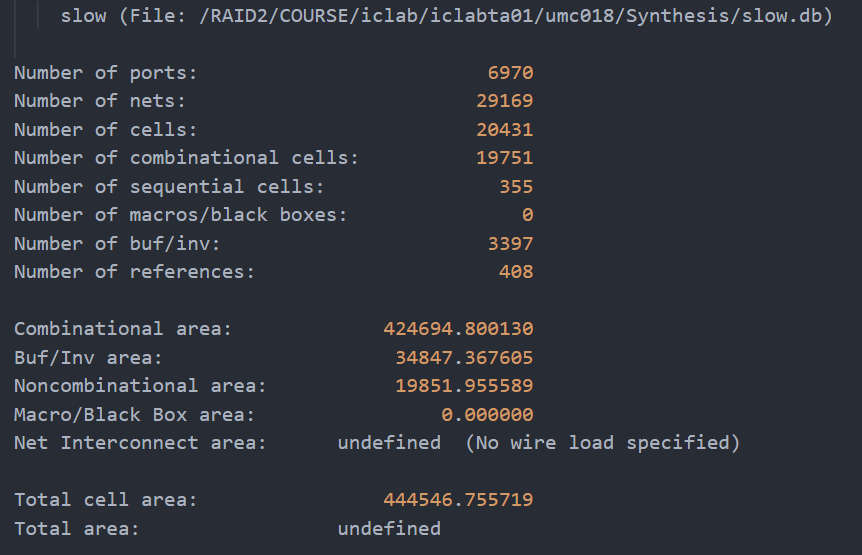
**遇到困難與如何解決/設計方法**

我的設計共做了三次，而以下會大致介紹每一次的概念、方法，以及針對一、二的改良:

* 設計一:

我用了最笨也是最圖法煉鋼的方式，就是直接在case判斷裏頭做各項的乘法，出來的面積大得驚人，這是我第一次寫出六位數子的設計！而之所以會有這樣的情況應該是因為利用太多的乘法器，如上可知，一個乘法器面積約為1000，若以這樣的寫法，做乘法的面積就大約1000\*25\*9 = 225000，可想而知為甚麼會有那麼大的電路。



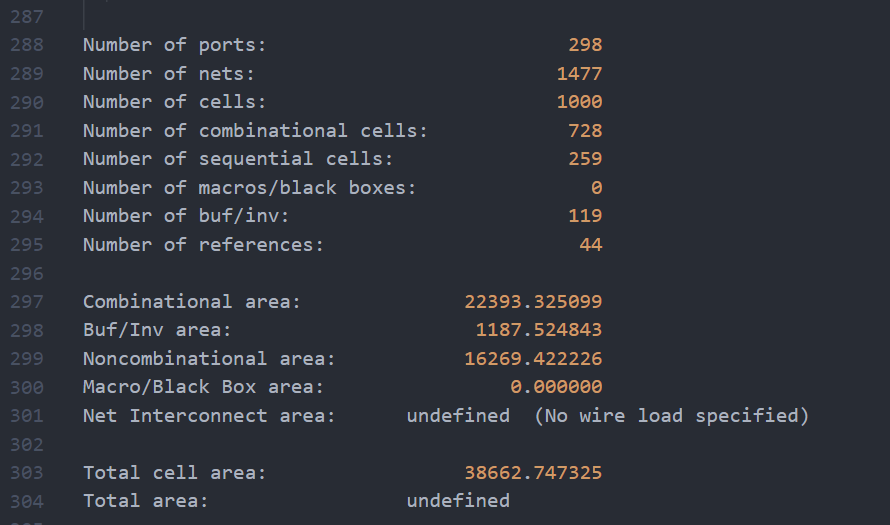


* 設計二:

我首先對乘法的面積進行改良，把乘法拉出case外面來做，另外設9個reg，而用case判斷甚麼時候把哪9個數字存進這些reg和filter進行乘法加總，這次的面積大概6萬多，而面積都主要為combinational。

* 設計三(最終設計):

這次的改良我針對case的條件判斷進行修正，因為combinational的面積依然很大，所以我直接對image做shift來代替25個的條件判斷，而當下被shift到乘法位置的那9個數字就和filter做乘法，一樣是使用9個乘法器，能夠再減了一些面積。結果如下:



**心得**

這次的作業讓我真正體會到乘法器面積的龐大，將來在做設計的時候，若能降低運算面積改為較簡單的operation像是這次的shift就要盡量使用，避免像我這次第一次設計出這麼恐怖的電路。所以從這次的作業我學到了: 要做出良好的設計，一定要對實質電路有相當的概念和認知，像是常用到的blocks (multiplier, adder, shift, mux等) 大概都會合成出多大的面積。