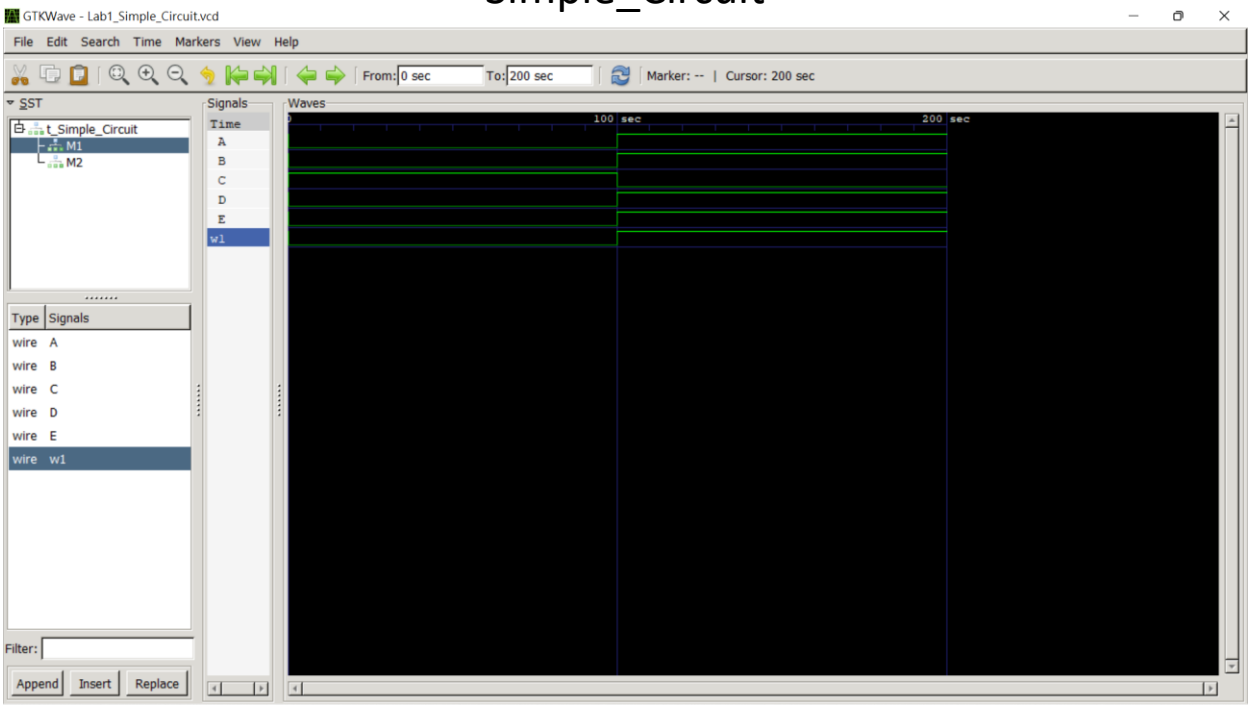
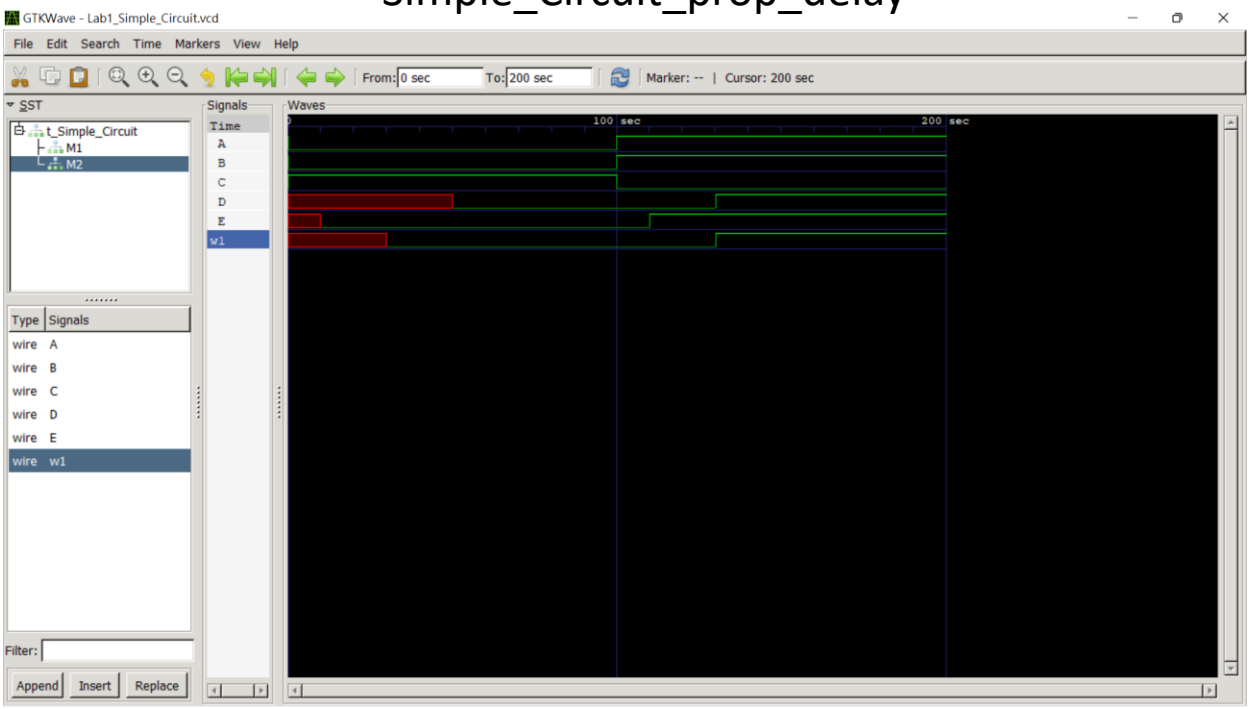
(1)兩者之間的差異來自於propagation delay，其中，因為G1的delay是30個單位，所以在A、B皆由0轉變為1後，w1會延遲30個單位才做變化，而G2的delay為10個單位，所以E在C由1轉變為0後10個單位才做變化，最後G3因為輸入的E由0轉變為1，再經G2延遲20單位，所以D會在C變為0後延遲30單位由0轉變為1。

Simple_Circuit

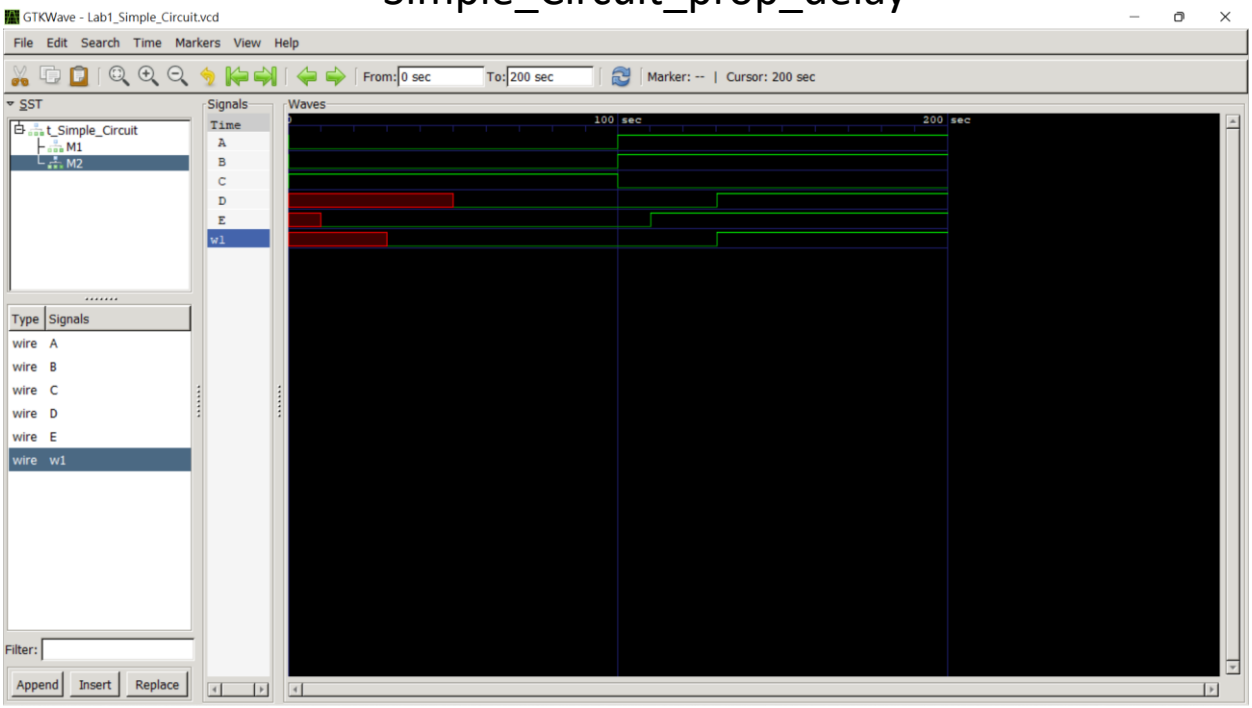


Simple_Circuit_prop_delay

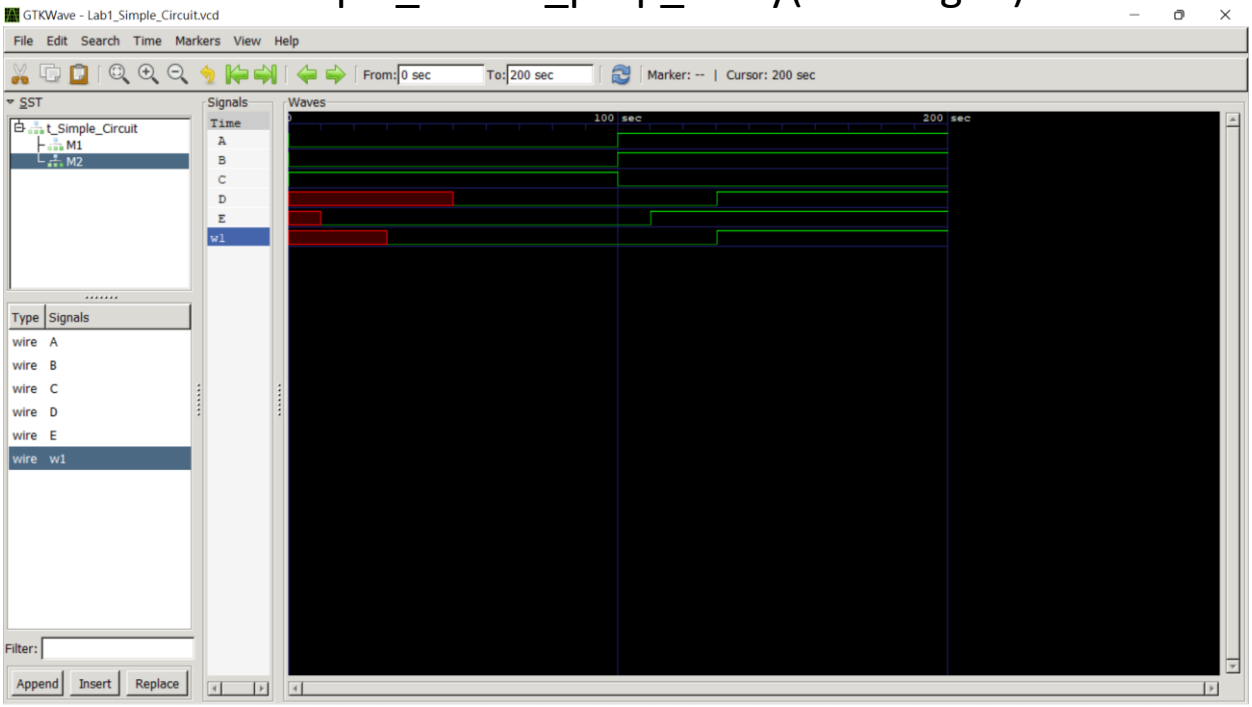


(2)以下兩個波形圖分別為and、or兩個敘述交換前後的波形圖，可以看到兩者的結果相同，由此可知，在模組當中的敘述並沒有順序性，即便在交換過後先看到w1的輸入才看到w1的輸出，對於模組而言仍然沒有產生影響。

Simple_Circuit_prop_delay

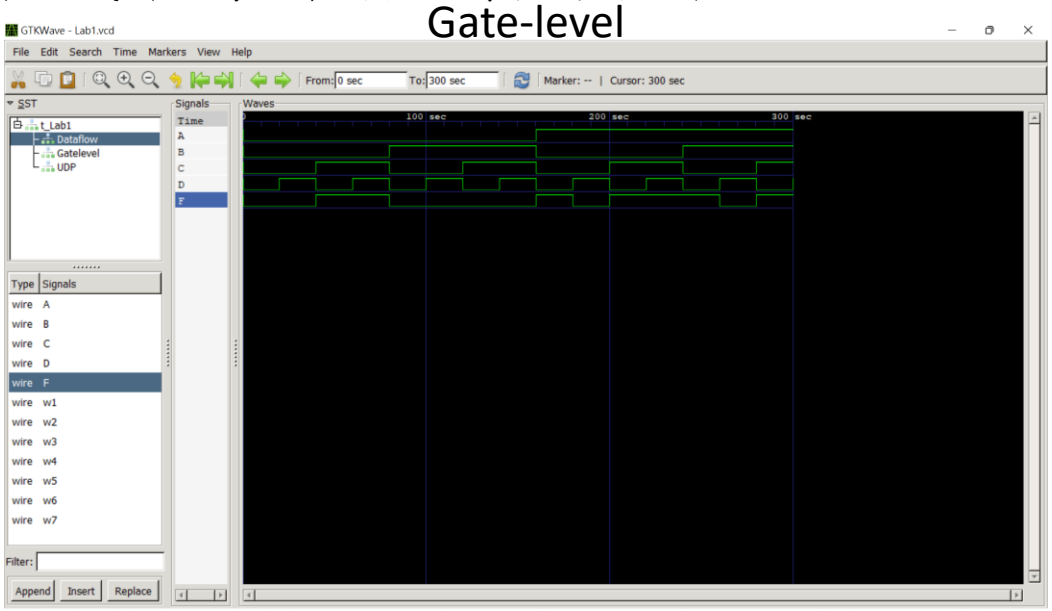


Simple_Circuit_prop_delay(exchanged)

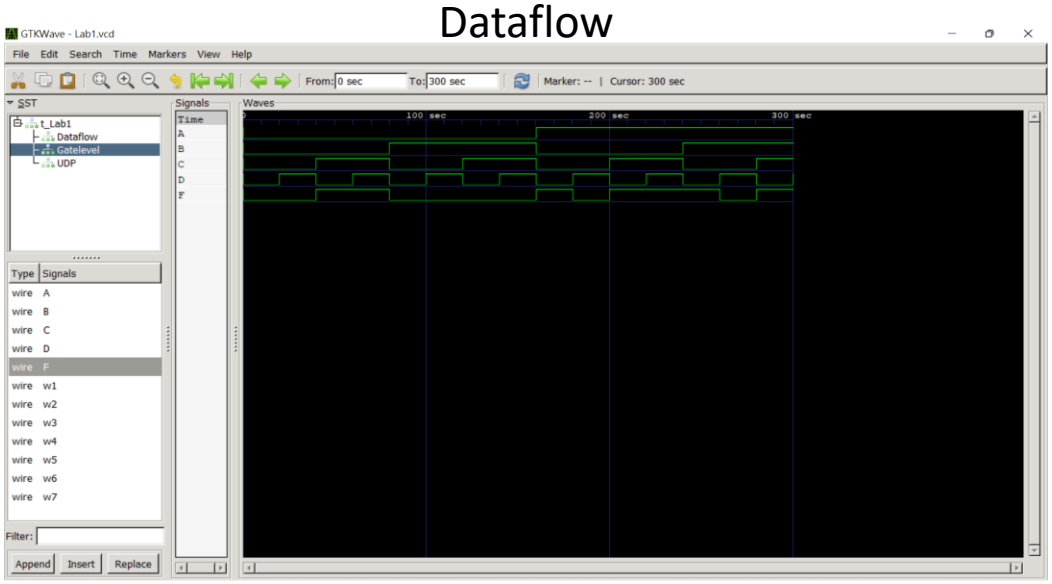


(3)以下三張圖片分別為Gate-level、Dataflow、UDP三種方式所撰寫出來的模組的波形圖，在Testbench當中測試四個Input共16種組合，以20單位為間隔，可以發現三者所產生的波形圖相同，即三種方式所撰寫出來的模組有相同的結果。

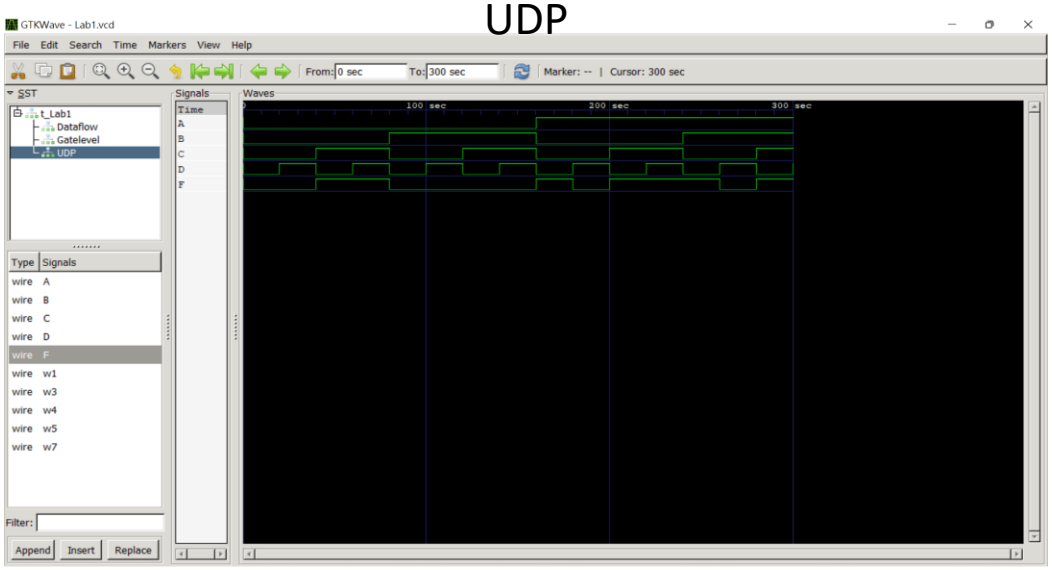
Gate-level



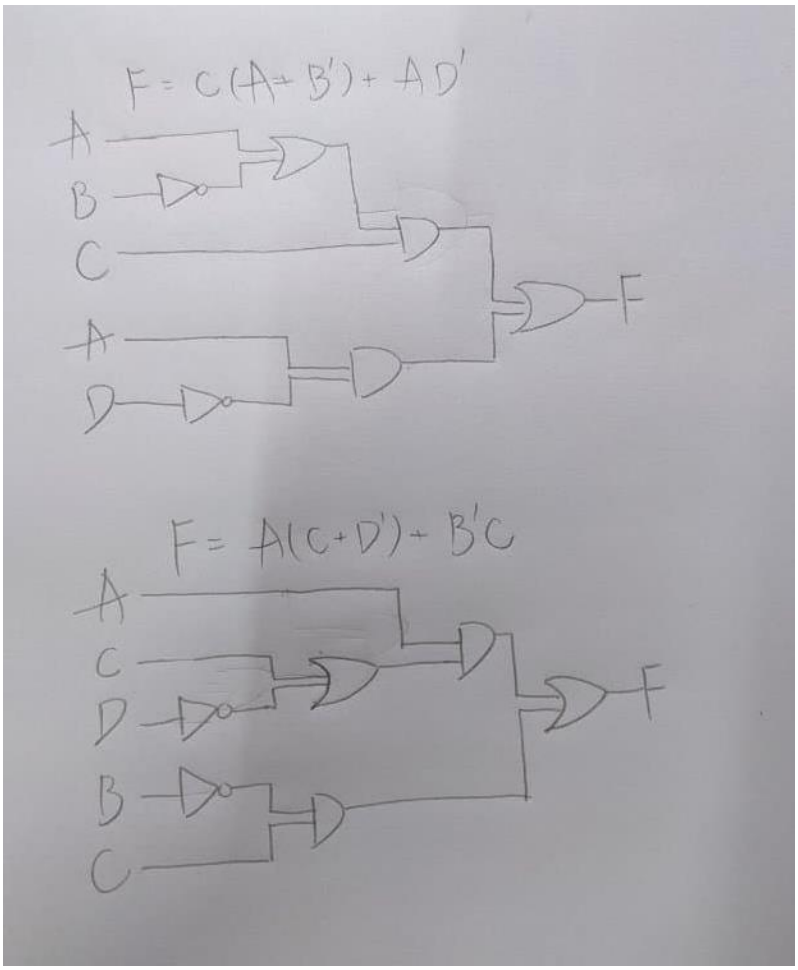
Dataflow



UDP



(4)否，該函式GIC最小之實作為F = C(A+B')+AD'或是F = A(C+D')+B'C，兩者的GIC皆為10，兩者電路圖如下圖所示。



(5)首先在終端機執行的時候，開file的輸入方法研究很久，一開始不知道後面可以直接寫兩個v檔，所以沒辦法一次跑出兩個圖形，直到後來寫B部分的時候才發現可以一次執行多個v檔。接著花了一些時間在研究Testbench的寫法，上網查了許多資料發現可以用localparam去固定每個周期的時間，方便測試所有Input組合。也花了不少時間理解三種方式的主要差異。

雖然過程當中大部分的內容都需要看許多資料才能夠理解並自己將模組寫出來，但因為老師給的資料很清楚，告訴大家在終端機要輸入甚麼可以執行，也給了簡單的例子讓大家測試、學習，所以在寫的過程當中雖然不算簡單但並不會覺得徬徨無助，也很有效的幫助我釐清三種撰寫方式的主要差異，並進一步了解module及testbench的用法。