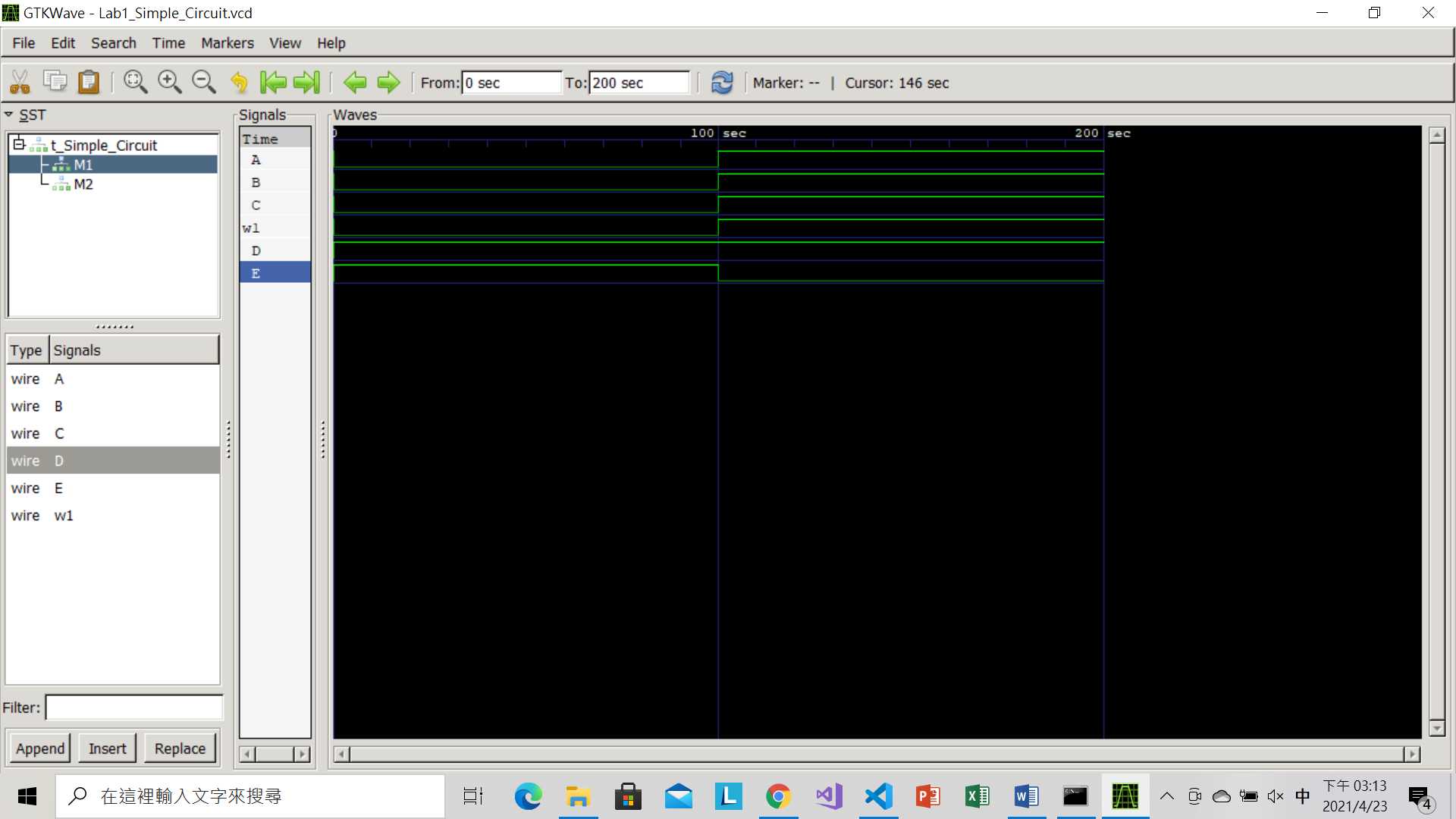
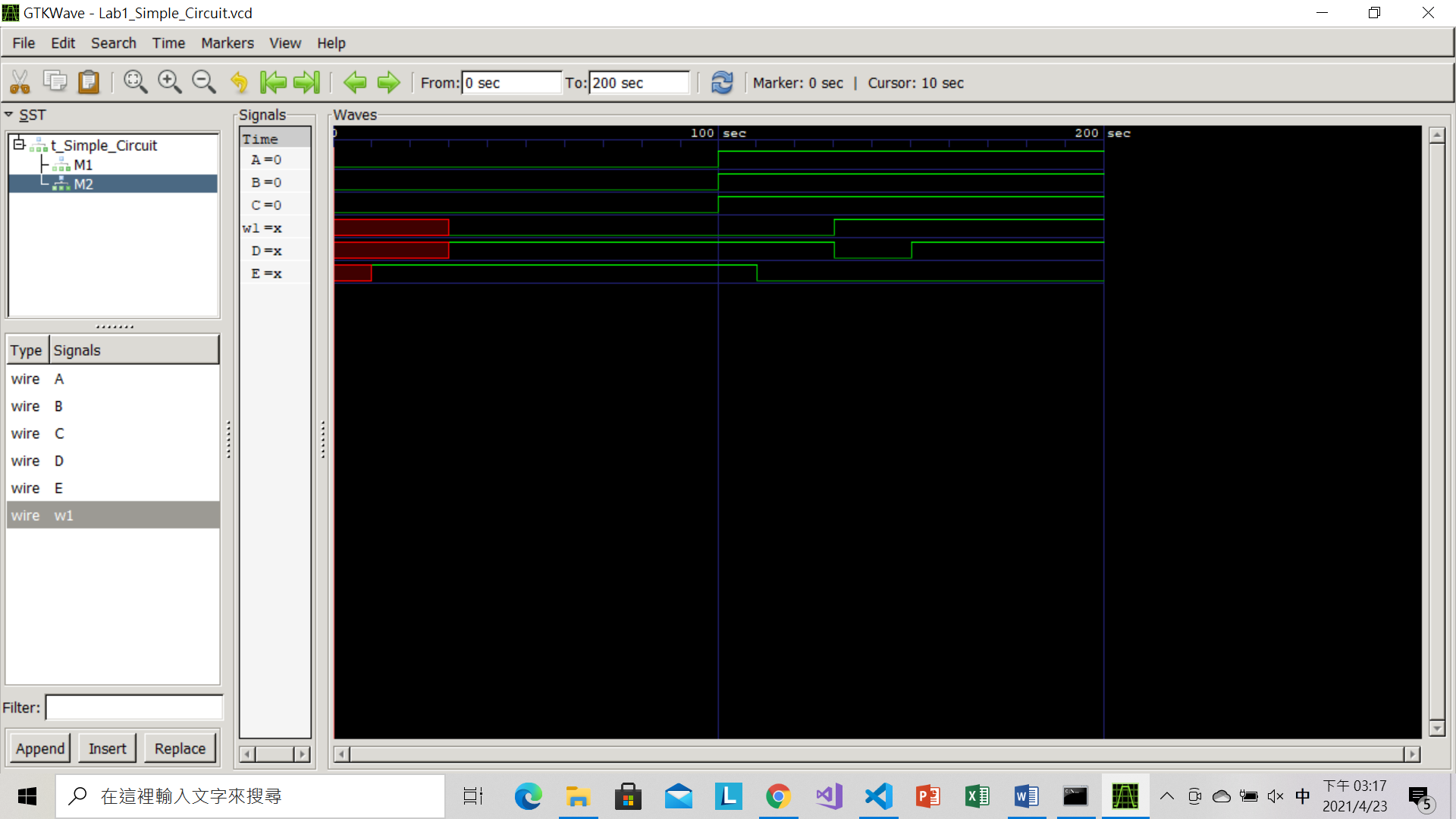
**1.A(a):**

Lab1\_A\_Simple\_Circuit Lab1\_A\_Simple\_Circuit \_prop\_delay

Propagation delay 的圖形和第一個的差別在於output和wire值改變的時間延遲

1.W1是經延遲30個時間單位的and gate輸出 故在input A B改變為1後30個時間單位才改變值

2.E是經由延遲10個時間單位的not gate輸出 故在input C 改變為1為10個時間單位才改變值

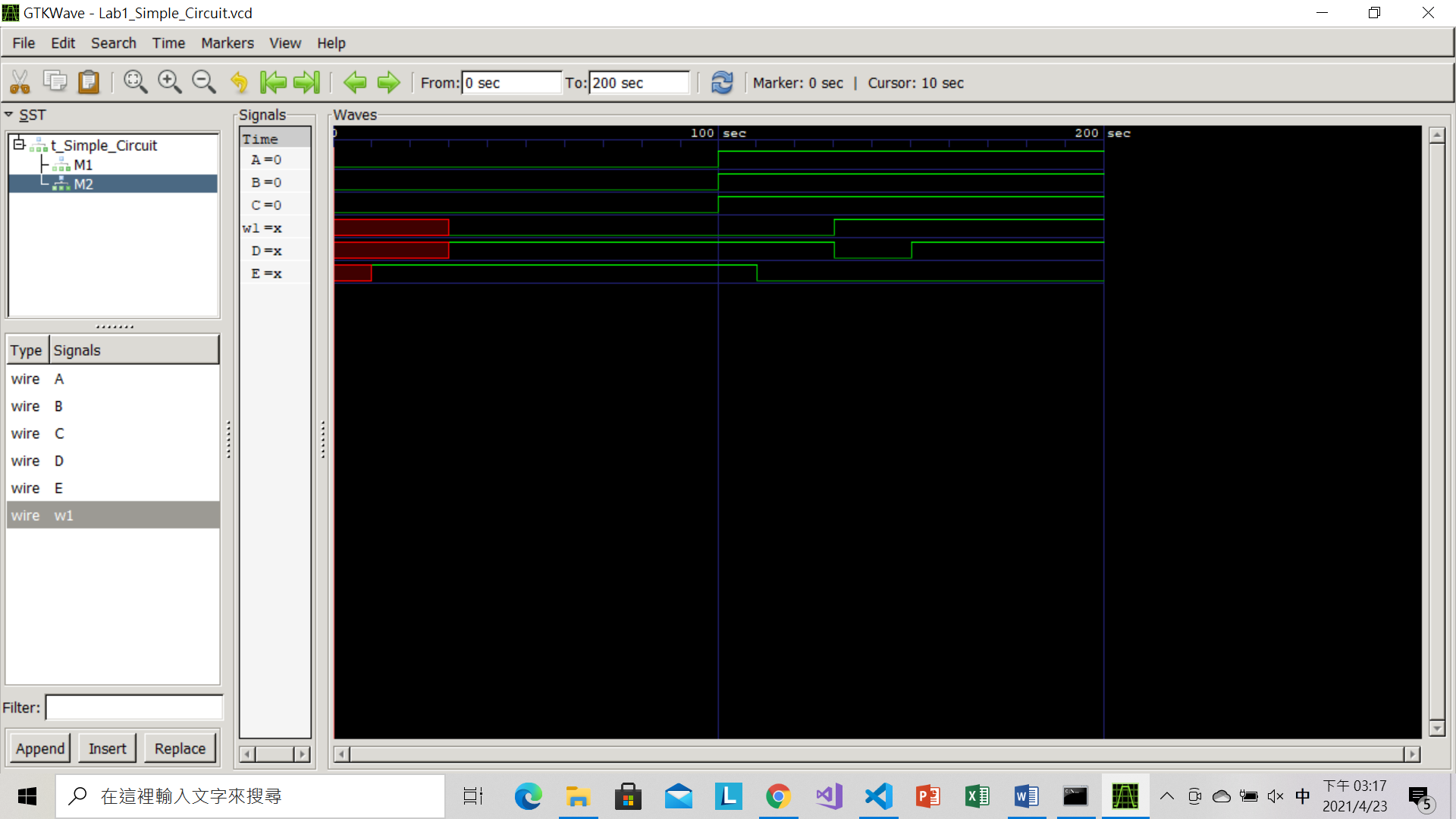
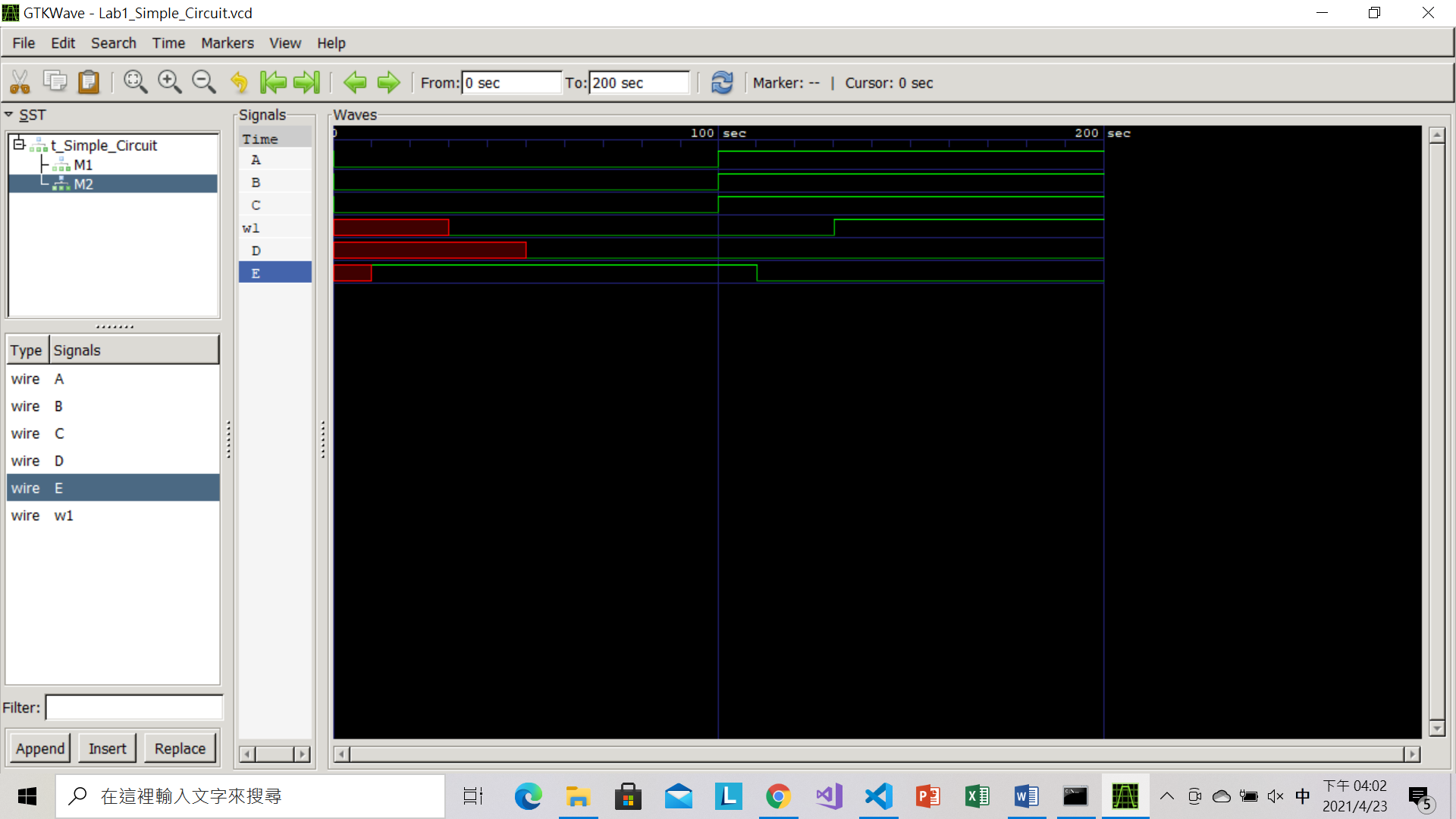
3.D是經由延遲20個時間單位的or gate輸出 其input分別為W1及E 又or gate任一input為1其

結果必為1 故E在先輸入or gate時再經過20個時間單位 即輸出D值

**2.A(b)**

Lab1\_A\_Simple\_Circuit\_prop\_delay

Before After

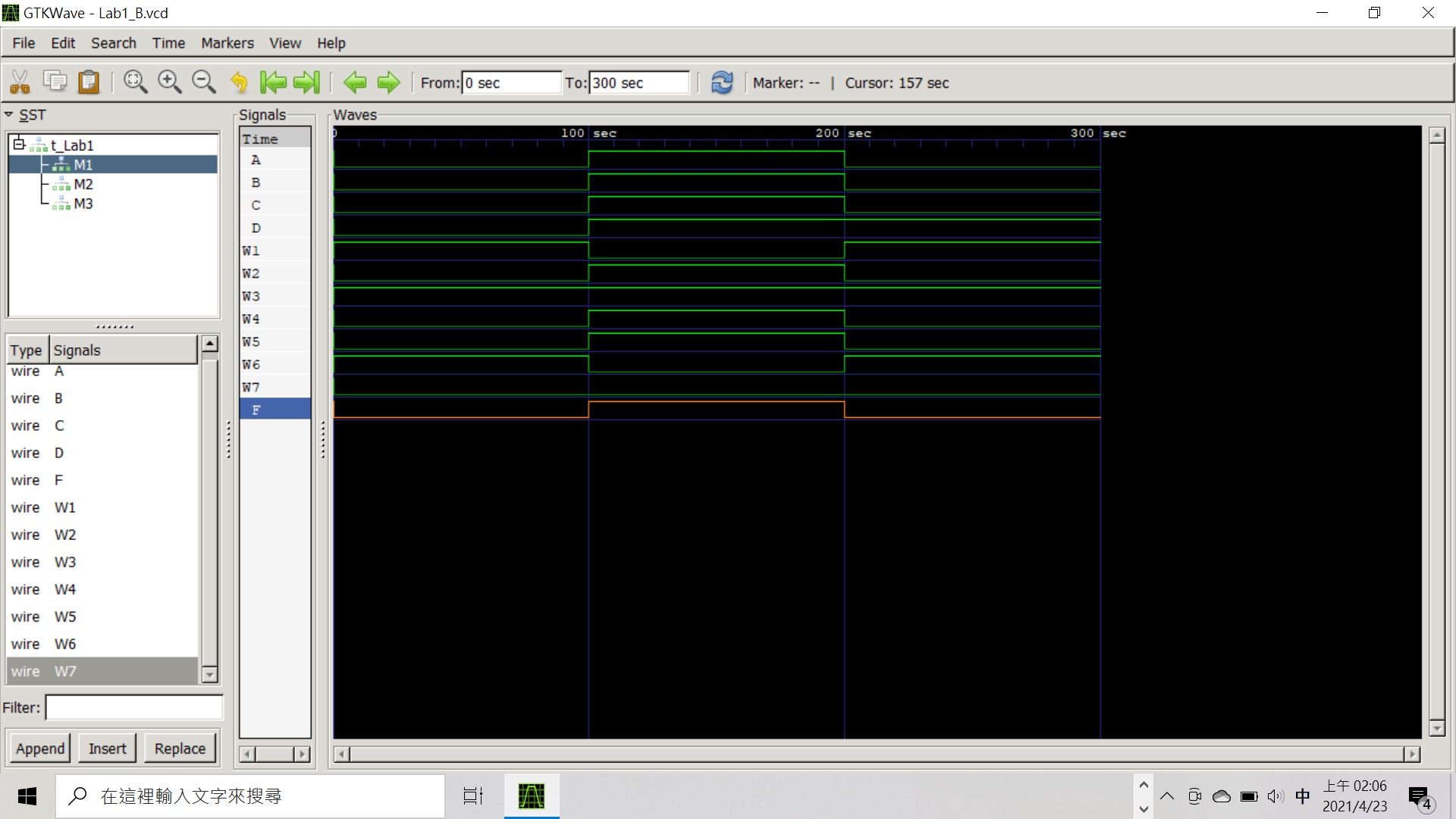
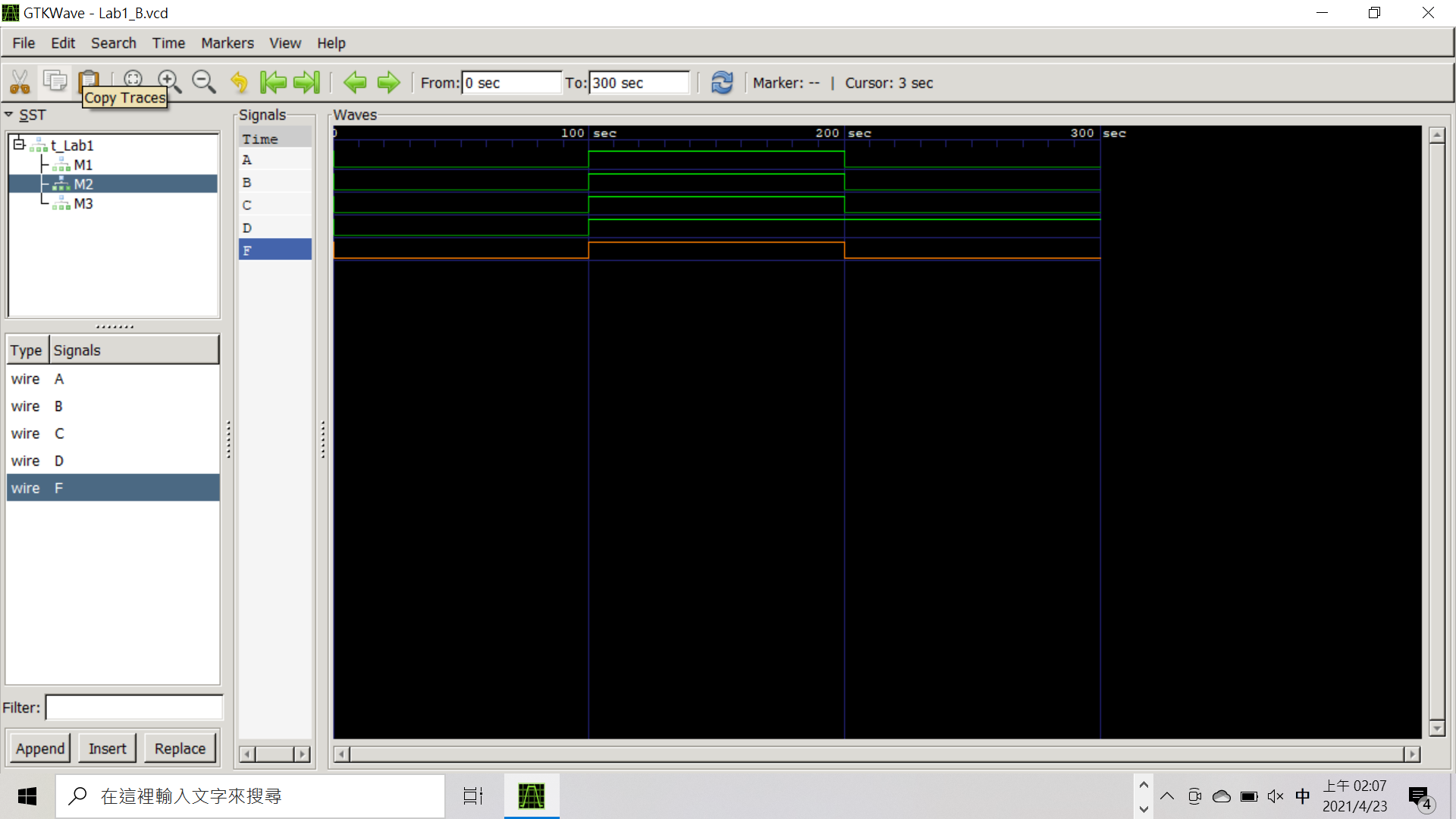
將and or的statement互換後 D變為經由and gate輸出 而and gate為任一input為1其結果必為0

然而先輸入的E為1 故須等待延遲30個時間單位的第二個輸入W1傳入後 再經過20個時間單位

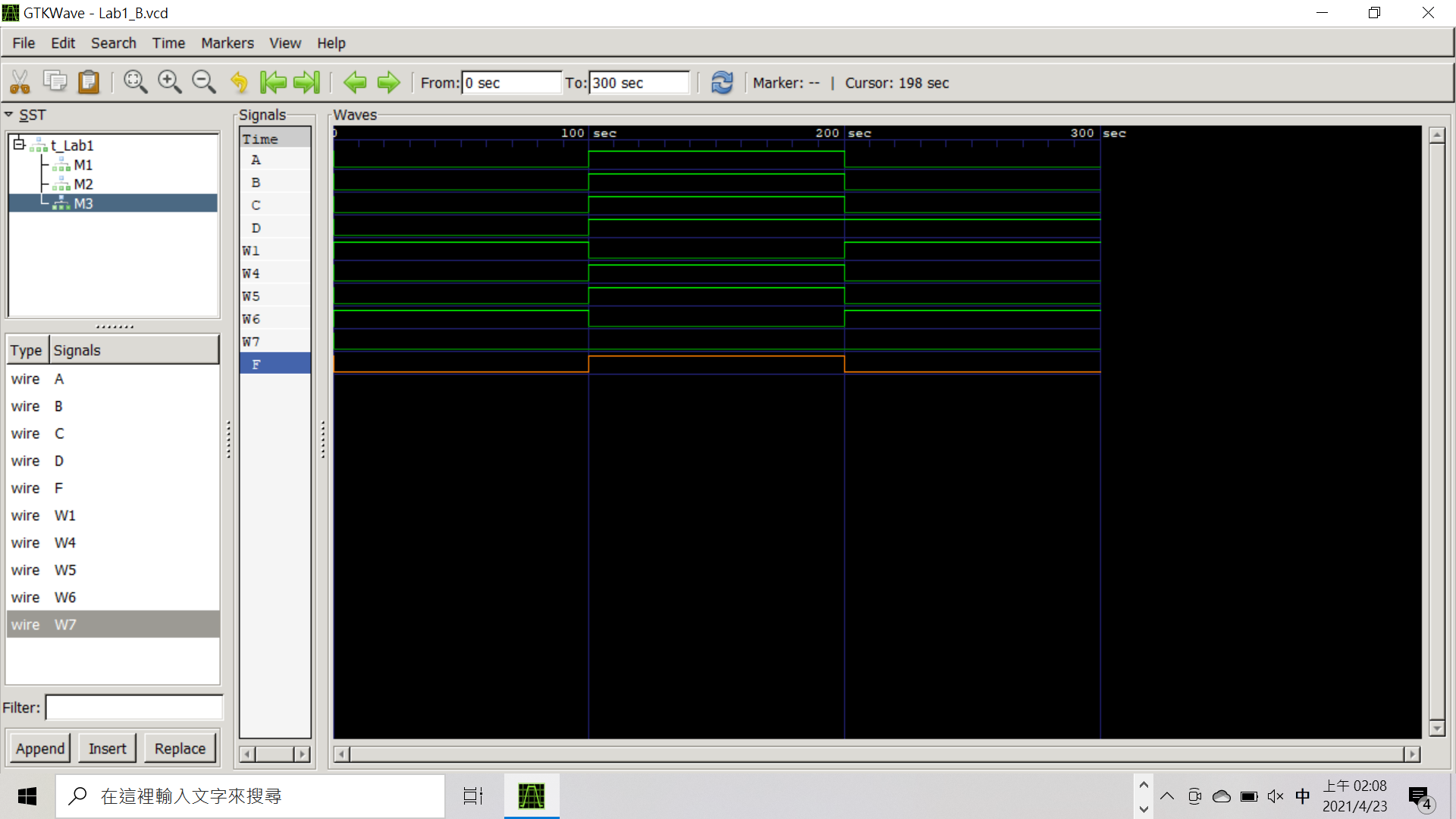
才產生輸出D 故兩圖差別在於 前圖D輸出時間為(10+20) 後圖D輸出時間為(30+20)時間單位

**3.B(d)**

Lab1\_B\_gatelevel Lab1\_B\_dataflow

Lab1\_B\_gatelevel\_UDP



由input值和各線路及output值檢查 三線路圖皆正確 三圖差別在於使用到的線路數量不同

1. gatelevel使用線路數量最多 因每個gate的input output都需細項定義
2. dataflow使用線路數量最少 因再assign的時候即一次將多個gate assign成一個statement
3. UDP可將部分線路使用user defined的方式簡化 故相對gatelevel線路數量更少

**4.**

由圖一可知

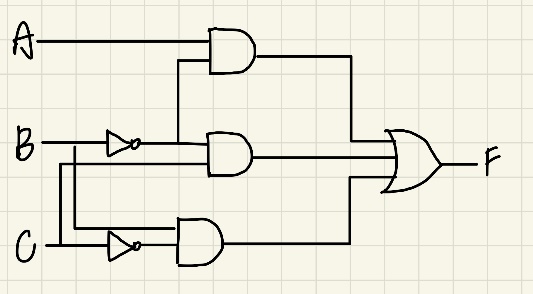
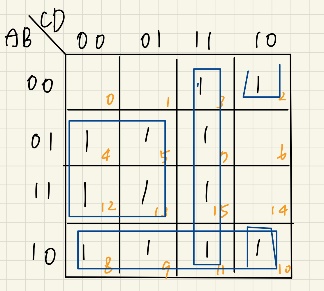
F=(A+B)(C’+D)+(A+C)B’ GIF=14

而F的minterm有(2.3.4.5.7.8.9.10.11.12.13.15)

畫出F的Kmap後可得

F=AB’+BC’+CD+B’CD’=AB’+BC’+C(D+B’D)=AB’+BC’+B’C GIF=2+2+2+2+3=11

因可求得gate input count的更小值 故圖一飛最小之實作

**5.**

(1)再命令提示字元中要vvp檔開啟vcd檔時一直失敗 後面才發現在testbench的地方要先initial使

用dump函數打開一個vcd數據庫才能使用

(2)在進行user defined primitive建置table的時候 對於輸入的C值要為C還是C’進行了討論 最後決

定使用C’ 因為前方的not gate並未圈入框內 故應另外執行

1. 一開始在testbench的地方指使值在100格時間單位後全數改變成1一次 圖形變化單調難以針對
2. 圖形做討論 因次我又新增了第二次值的改變 並將最後的輸出F以不同顏色的線路表示 因此更能清楚看出圖形的變化

(4)在進行此作業時 因為要不斷叫出gtkwave觀察波型圖 也讓我更熟悉如何使用命令提示字元 且在

實際打verilog的code還有觀察波形圖的變化時 我也更清楚了各個gate的運左模式及輸出變化

實作後總是能讓我比起上課聽講更清楚的融會貫通 且我覺得這個作業的難度不僅僅是在使用新的

編碼程式verilog 清楚的了解老師的題目希望要求做到的是甚麼也讓我下了不少功夫