312510188_王思閎_HW2report

此次作業是要在最差的情況下，找到最久的 delay time 和最快的 delay time，而我的程式執行如下:

1. 先將資料讀取進來，並使用 read_netlist 來將資料化簡並保存，其中我創建了兩個 struct，分別是存放 gate 和 wire，而其中有許多資訊，如圖所示，將得到的資料分別處理好，方便後續使用。

```
struct Gate
{
    string label; // 導線名稱
    Wire *ZN = NULL; //輸出
    Wire *A1 = NULL; // A1輸入
    Wire *A2 = NULL; // A2輸入
    int gate_type = -1; // 0代表INVX1, 1代表NOR2X1, 2代表NANDX1
    vector<Wire *> path; // 它前面所有的路徑
    double o_cap = 0;     // 輸出電容
    double trans = -1;    // transition time
    double delay = -1;    // delay time
    double total_delay = 0; // 總delay
};
typedef struct Gate Gate;

struct Wire
{
    int value = -1;       // 這條線傳輸的值
    string label;         // 導線名稱
    Gate *pre = NULL;     // wire的前面gate是哪個
    bool OUT = false;  // 若是true，代表是output導線
    vector<Gate *> next; // 輸出gate有哪些
};
typedef struct Wire Wire;
```

2. 接著開始執行 step1，使用 output_capacitance()來計算每個 gate 的 output transition，遵照題目要求，找到最大的那個。
3. 執行 step2 和 step3，使用 choose_the_worst();指令，先找到這個 gate 是 INV NOR 還是 NAND 再接下去處理，若有多個 input 則採用 total delay 最久的 input，並要注意每條 wire 上還會有 0.005ns 的延遲，其中因為需要使用到內插和外插公式，也有寫一個 interpolation 的 function 來計算。要注意的是，有寫了很多個 if else 涵式來判斷，其中包刮 input 是不是都有輸入進來了、要輸出 1 還是 0delay 會比較久、這個 gate 前面有接線嗎還是它是開頭等等，因為每個 netlist 的不同會有不同的情況，我在測試時也很常發生其中一個 netlist 可以順利執行，但另一個 netlist 卻輸出錯誤訊息，要特別注意有把每個情況都考慮進去。
4. 最後，choose_the_worst();這個 function 同時也會記錄每條 path，紀錄其中經過了哪幾個 gate 和最後 total delay 的時間，所以 step3 也完成。
5. 將所有所需資料輸出，便完成。