|  |
| --- |
| **Computer-Aided Design for VLSI Design** |
| Homework3（Student ID: 61047046S｜Name: 林宣佑） |

1. Provide a simple explanation of your code.

這次的作業定義了一個名為HW3的entity作為主程式，另外定義了Counter以及Controller兩個entity作為component，並引入它們的功能來實現紅綠燈的控制。

HW3：

在entity HW3中定義了它的實體。它具有一個input：clk（時脈），三個output：glight（綠燈）、ylight（黃燈）和rlight（紅燈）。clk用於驅動整個系統的時脈，而glight、ylight和rlight則用於輸出控制紅綠燈的狀態。

在architecture Behavioral of HW3中，先是定義了Counter及Controller互相溝通用的三個signal：reset、enable、complete，以及用來定義計數器上限的count。接下來引入了Counter和Controller兩個component。

接下來定義Counter\_inst和Controller\_inst來實例化Counter和Controller兩個component，並使用port map將兩個component與主程式相互連接，由此將它們的功能集成到主程式中，並且可以透過clk、reset、enable和complete訊號進行相對應的操作和控制。

Counter負責進行計數操作。它接收clk訊號作為時脈輸入，reset訊號用於重置計數器，enable訊號用於判斷計數器是否啟用，計數器的計數值可由主程式設定上限。

Controller則負責控制紅綠燈的狀態，它接收clk訊號作為時脈輸入，reset和enable訊號由Controller生成並輸出給Counter，complete訊號則由Counter輸出並作為輸入給Controller，另外，glight、ylight、rlight訊號是用於輸出控制紅綠燈的狀態。

主程式中的reset、enable和complete訊號在Counter和Controller之間作用為傳遞和控制。當HW3收到clk訊號時，它會同時將clk訊號傳遞給Counter和Controller，並從Controller接收到glight、ylight和rlight訊號。根據Counter和Controller的操作和狀態轉換，HW3會相應地設定紅綠燈的狀態並更新下一個狀態。

總結來說，HW3作為紅綠燈控制器的主程式，它將Counter和Controller作為組件引入並使用它們的功能來實現紅綠燈的控制。Counter負責計數操作，Controller負責控制紅綠燈的狀態。通過主程式中的訊號傳遞和控制，紅綠燈的狀態會根據Counter和Controller的操作進行切換和更新，整個專案也是一個有限狀態機。

Counter：

Counter在這個紅綠燈控制系統中負責計數操作，在entity Counter中定義了它的實體，它具有三個input：clk（時脈）、reset（重置）、enable（啟用），兩個output：complete（完成）和count（計數）。

* clk是時脈訊號，用於驅動計數器的計數操作。
* reset是重置訊號，當reset為高電位（'1'）時，計數器將被重置為初始值。
* enable是啟用訊號，當enable為高電位（'1'）時，計數器開始計數。
* complete是完成訊號，當計數器的計數值達到最大值時，complete將被設置為高電位（'1'），表示計數操作已完成。
* count是計數訊號，它是一個buffer integer型態，用於輸出計數器的計數值。

在architecture Behavioral of Counter中，我們定義了一個process，用於控制計數器的操作。在這個過程中，我們偵測clk訊號的變化。

* 當clk訊號的rising edge觸發時，首先檢查reset訊號是否為高電位（'1'），若是則我們將計數器的輸出值重置為全零，即count <= 0、complete <=（'0'），若否則接著
* 檢查enable訊號是否為高電位（'1'），若是則我們將計數器的值加1，即count <= count + 1，並檢查計數器的值是否已經達到最大值，即count = MAX\_COUNT。如果是，則將complete訊號設置為高電位（'1'），表示計數操作已經完成。
* 如果計數器的值尚未達到最大值，則將complete訊號設置為低電位（'0'）。
* 最後，我們將計數器的值轉換為buffer integer型態，並將其賦值給count訊號，以便在Counter中輸出計數值。

總結來說，Counter組件通過使用clk訊號作為時脈輸入，reset訊號用於重置計數器，enable訊號用於啟用計數器的計數操作，並通過complete和count訊號輸出計數器的狀態。它實現了計數器的功能，並在計數值達到最大值時發出完成訊號。

Controller：

Controller在這個紅綠燈控制系統中負責控制紅綠燈的狀態，在entity Controller中定義了它的實體，它具有兩個input：clk（時脈）、complete（完成），五個output：reset（重置）、enable（啟用）、glight（綠燈）、ylight（黃燈）和rlight（紅燈）。

* clk是時脈訊號，用於驅動Controller的操作。
* reset是重置訊號，由Controller組件生成並輸出，用於重置Counter組件。
* enable是啟用訊號，由Controller組件生成並輸出，用於啟用Counter組件的計數操作。
* complete是完成訊號，由Counter組件輸出並作為輸入給Controller組件，表示計數操作是否已完成。
* glight、ylight和rlight分別是綠燈、黃燈和紅燈的輸出訊號。

在architecture Behavioral of Controller中，我們定義了一個狀態類型State，其中包含六個狀態：green1、green2、red1、red2、yellow1 和 yellow2，初始狀態為green1。

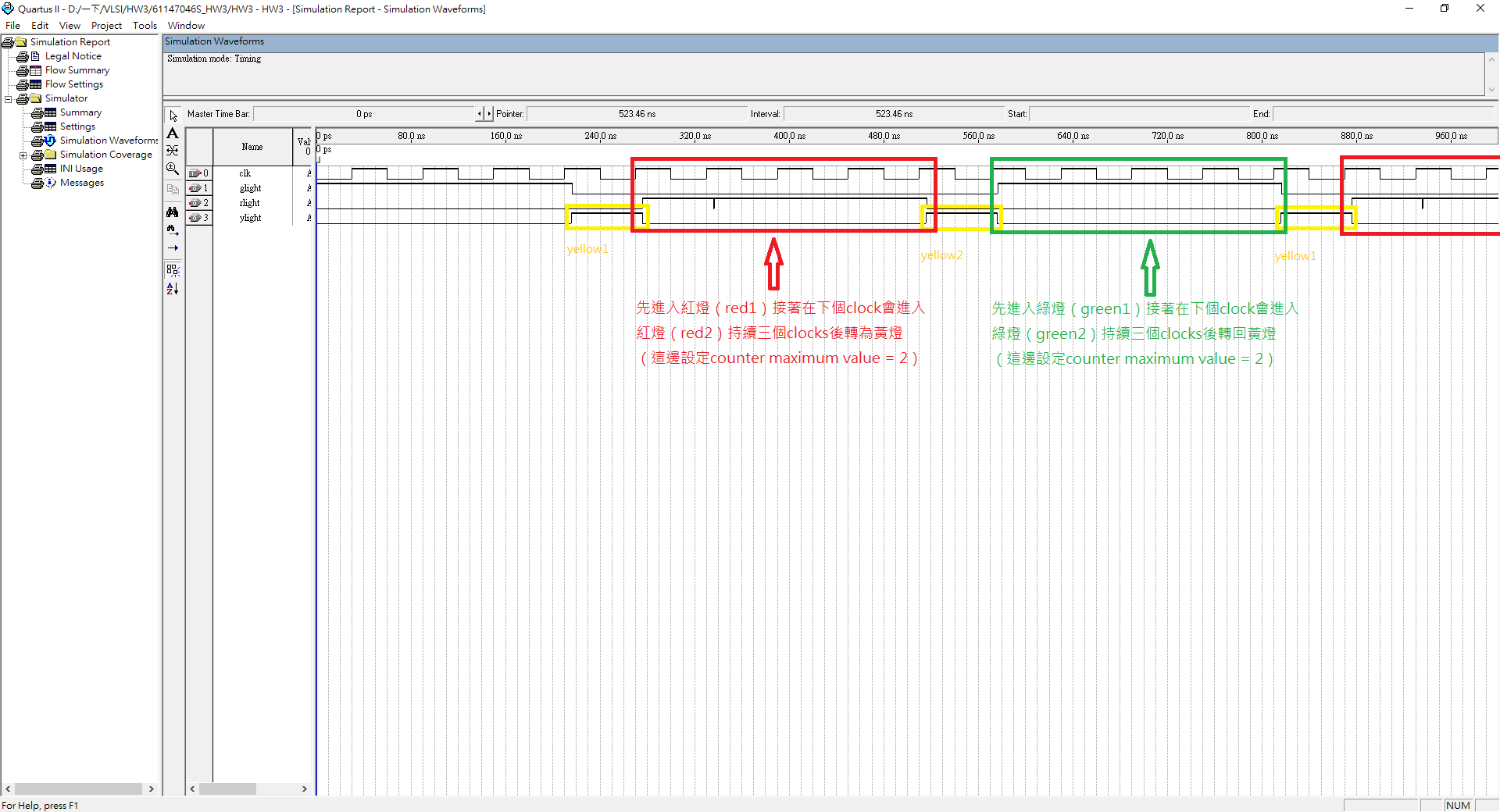
在主要的process中，我們檢測時脈訊號的rising edge。如果是上升訊號則將current\_state更新為next\_state，即進入下一個狀態。

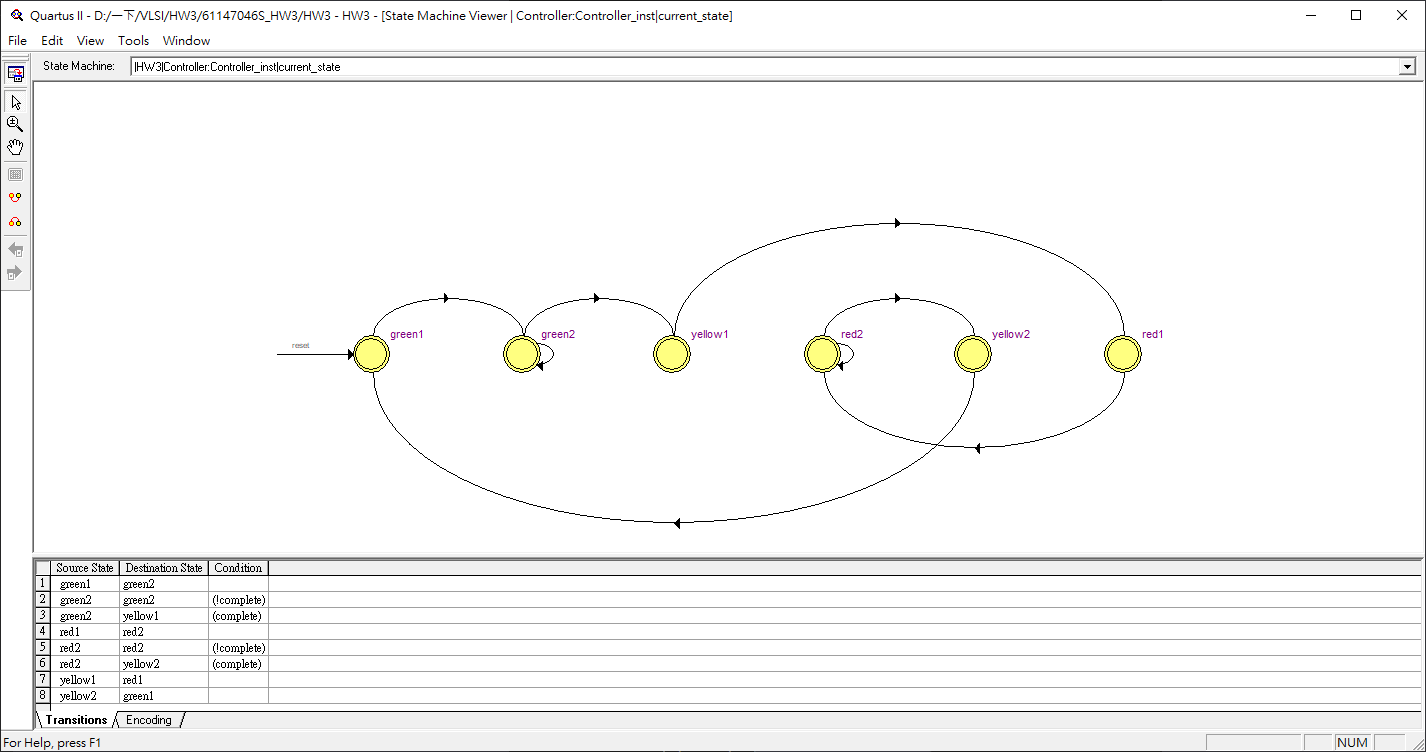
在第二個 process 中，我們根據current\_state的值進行分支判斷，並對輸出訊號進行設置。根據不同的狀態，我們設置對應的綠燈、黃燈和紅燈訊號的值，同時設置reset、enable訊號的值以跟Counter進行溝通。

在每個狀態的最後，我們使用next\_state變數指定下一個狀態。這樣在下一個時脈週期中，current\_state會被更新為next\_state，從而使控制系統進入下一個狀態。

總結來說，Controller組件根據紅綠燈系統的狀態進行控制，根據不同的狀態設置對應的輸出訊號，同時控制計數器的啟用。它根據時脈訊號和計數器的完成訊號進行狀態轉換，實現了紅綠燈系統的控制邏輯。

1. Waveform diagram here (Simulation Results)





1. Reflections and discussions

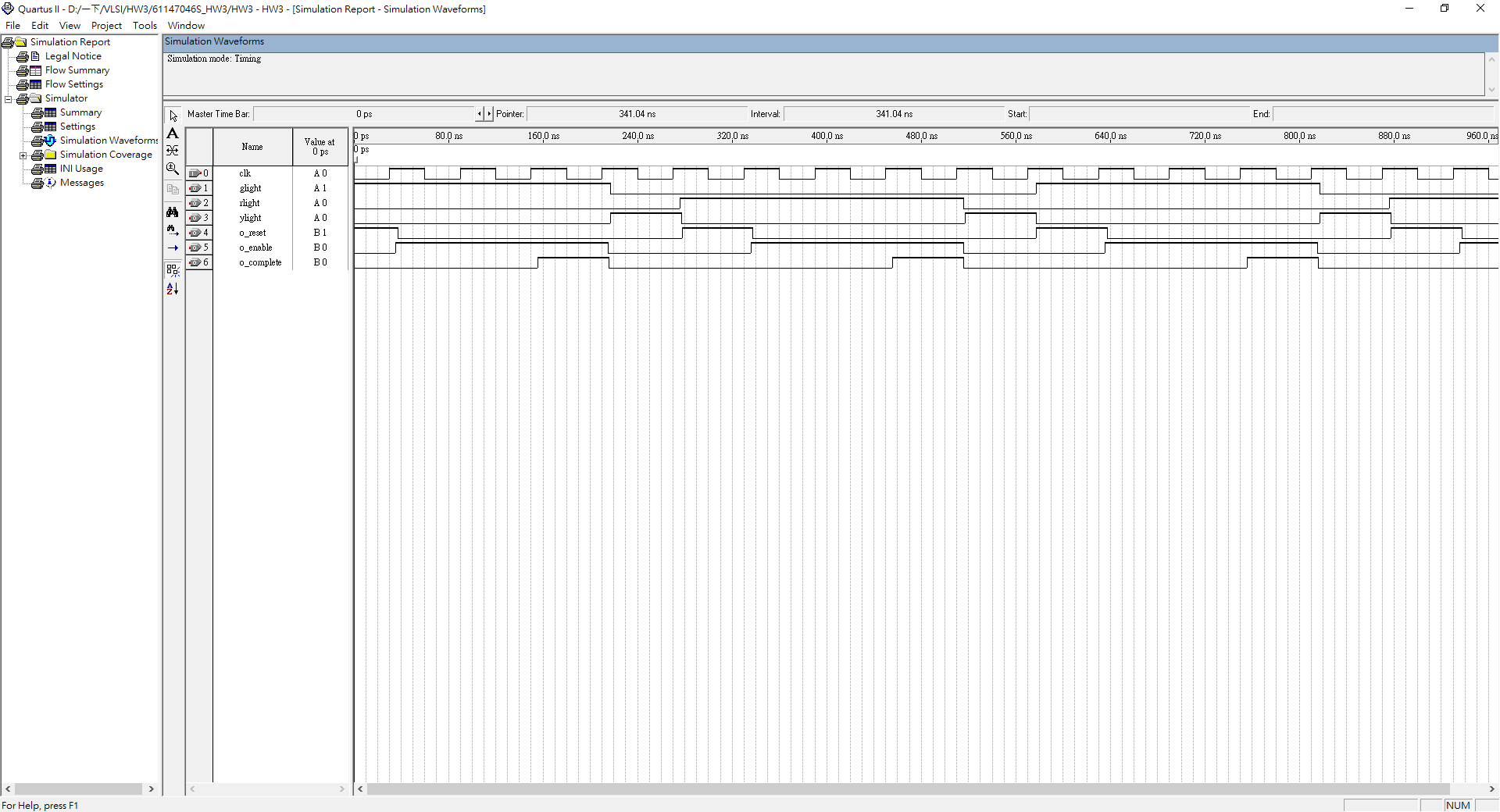
這個紅綠燈控制電路的設計過程中，我們延伸了上次的component用法。透過將系統切分為主程式及Counter和Controller兩個組件，實現了模組化和可重用的程式碼。

在Counter組件中，我們使用了一個計數器來實現紅綠燈燈號的切換。透過時脈信號的驅動，計數器可以在每個時脈週期上升一次，並在達到最大值時產生完成信號。這個計數器的設計簡單而直觀，可以應用於各種計數操作。

在Controller組件中，我們使用了狀態機的設計方法來控制紅綠燈的狀態轉換。透過定義不同的狀態和對應的輸出信號，我們可以在每個狀態中設定紅綠燈的亮滅以及計數器的啟用。

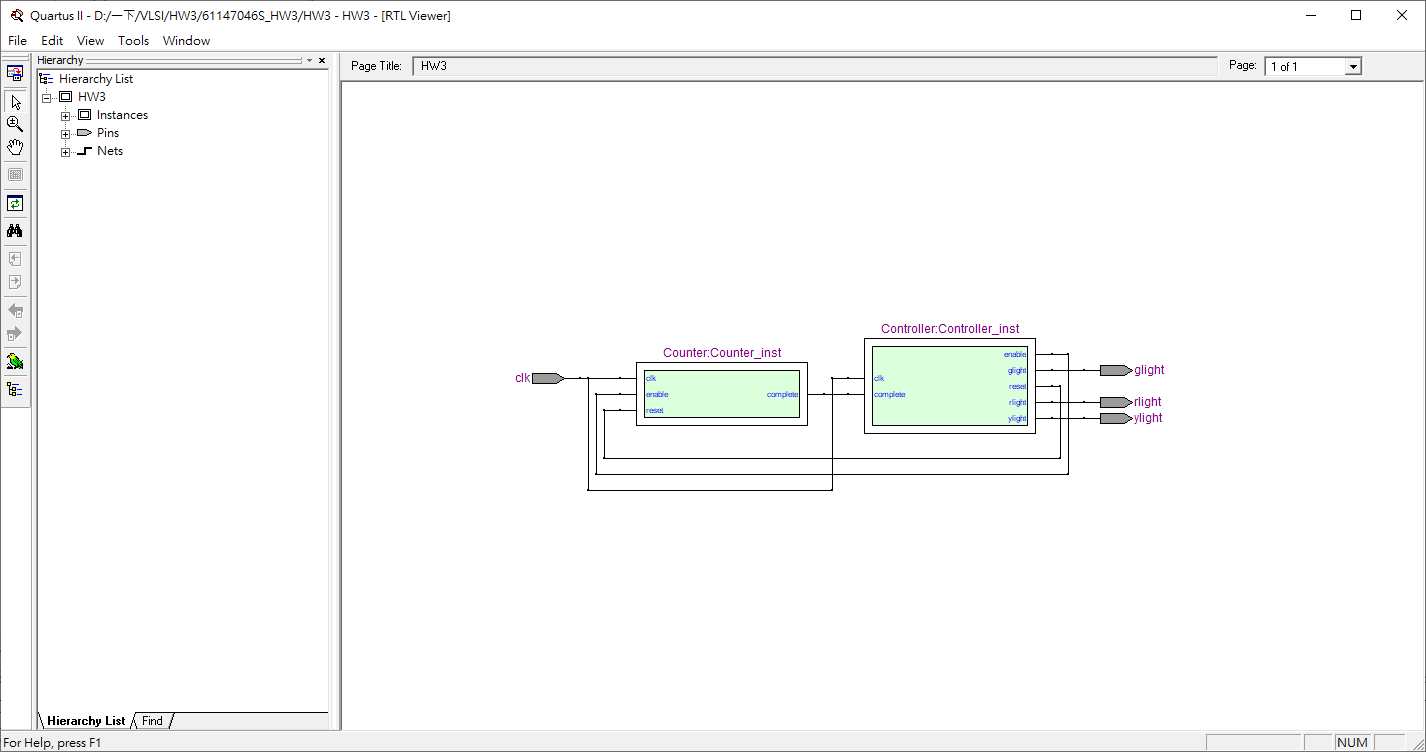
這次的作業使用狀態機的設計方法來實現複雜的控制邏輯。狀態機的設計使得控制系統的狀態轉換清晰可讀，並提供了靈活性和可擴展性。透過定義不同的狀態和對應的輸出控制，我們可以有效的控制紅綠燈系統的運作，輕鬆調整狀態的邏輯、順序。

另外有趣的是在此次實驗中一開始是先套用學姊的範例輸入counter maximum value = 2，但顯示出來的波形圖數量卻不一樣，於是自己另外加上三個output，觀察reset、enable、complete訊號之關聯性。



以上這張波型圖雖然跟學姊的範例波型圖一樣，但學姊設定的counter maximum value為2，而我則是設定為1。根據波型圖可觀察到進入green1/red1時發送reset訊號會消耗一個clock，根據我撰寫Counter的程式邏輯count: buffer integer range 0 to MAX\_COUNT，因為0 to 1所以在green2/red2時會等待兩個clocks，最後等待complete訊號也會消耗一個clock，故1+2+1=4 clocks。

附圖為產生出的RTL電路圖：



可觀察到Counter與Controller的enable、reset、complete訊號相互連接，且由Controller輸出綠、紅、黃三個燈號的訊號。