|  |
| --- |
| **EECS 2070 02 Digital Design Labs 2019**  **Lab 5** |
| **學號：107000115 姓名：林珈卉** |

1. 實作過程
   1. 設計概念
      1. 實作出模擬的販賣機，共有兩個商品可購買
      2. 按下按鈕可以投入 10 元或是 5 元硬幣
      3. 若投入的金額足夠，相對應的商品燈號會亮起
      4. 按下商品的按鈕， 7-segment display 的左邊兩個 digit 會顯示該商品的價格
      5. 在金額足夠的狀況下，再按一次相同商品的按鈕，等同於購買該商品
      6. 購買商品後，找錢會以退回 10 元硬幣為優先，再退回 5 元硬幣
      7. 找錢結束後，會回到一開始可以投錢的狀態
   2. 實作方法
      1. 程式分為四個 state，依序為「初始」、「投幣」、「購買」及「找錢」狀態
      2. 「初始」及「購買」只會維持一個 clock cycle，就跳入下一個狀態
      3. 使用 clock divider 製造出週期為 、、 倍的三種 clock，分別命名為 clk\_13、clk\_16、clk\_26
      4. 在「初始」及「投幣」兩個狀態，使用 clk\_16，其餘兩個使用 clk\_26
      5. 宣告 wire 變數 clk\_FSM，根據現在狀態，assign 為 clk\_16 或是 clk\_26
      6. 將按鈕的 input signal money\_5、money\_10、cancel、drink\_A、drink\_B 接上 debounce module 及 onepulse module，命名為 m5\_1、m10\_1、cancel\_1、A\_1、B\_1
      7. 宣告 reg 變數
         1. [3:0] value, BCD0, BCD1, BCD2, BCD3，用來控制 7-segment display
         2. [1:0] state, next\_state，用來記錄目前的狀態，「初始」、「投幣」、「購買」及「找錢」
         3. [6:0] money, next\_money，紀錄目前已投入金額，即目前機器裡還有多少錢還沒找回
         4. [9:0] next\_drop，配合 output signal next\_drop
         5. item\_1, item\_2, next\_item\_1, next\_item\_2，紀錄哪個商品已備按過，當作購買商品的判斷條件，及扣除金額時的判斷條件
      8. 設計三個 always block
         1. 一個 always block 以 posedge clk\_FSM trigger，負責將 next\_xxx 值送給 state、money、drop\_money、item\_1 及 item\_2
         2. 一個 always@\*，最上面先寫上 next\_xxx = xxx 當作 default
            1. 使用 case(state) 判斷現在的狀態
            2. 「初始」狀態，將所有 next\_xxx 的訊號 reset
            3. 「投幣」狀態，用 if else 判斷現在按下的按鈕為何

按下投幣按鈕，即 m5\_1 或 m10\_1 為 1，若 money 在範圍內，next\_money = money + 5 或 10

按下商品按鈕，即 A\_1 或 B\_1 為 1，

若商品已經被按過，而且已投入金額也足夠，next\_money = money - 價格，並將 next\_state 設為「購買」

else 即把 next\_item\_1 或 \_2 設為 1，另一個為 0

按下取消按鈕，next\_state 直接進入「找錢」狀態

* + - * 1. 「購買」狀態，使用 clk\_26，next\_state =「找錢」
        2. 「找錢」狀態，使用 if else 判斷

若 money >= 10，next\_money = money - 10，next\_drop = 10’b1111111111

若 money >= 5，next\_money = money - 5，

next\_drop = 10’b1111100000

* + - * 1. 最後面判斷 rst 是否為 1，next\_state 直接回到「初始」狀態
      1. 另一個 always@\* 控制顯示器的四個 digit
         1. 若在「購買」狀態，則根據買的商品顯示商品名稱
         2. 其他狀態的 BCD0 以及 BCD1 為 money 的個位數和十位數

若在「投幣」狀態，判斷上一個被按下的商品按鈕是哪一個後，BCD2 和 BCD3 顯示商品價格

其他狀態的 BCD2 和 BCD3 皆顯示 0

1. 學到的東西與遇到的困難

* 原本的 item\_1、item\_2 是合在一起，寫成 reg item，判斷為 0 或是 1，極為不同的商品，但沒辦法判斷「沒有選擇商品」的時候，而且有時會出錯，拆成兩個變數比較適當。

1. 想對老師或助教說的話

* 謝謝助教在我 debug 到崩潰的時候，幫我檢查 code，也指出問題所在，提醒我 reset 的部分要再修改，還有提到 clock assign 可能會出一些問題。
* 想請問老師，後面的課程是否會提到 clock assign 會出的問題？