**Hardware Design and Lab: Lab5**

**Team 28**

**111060013 EECS 26' 劉祐廷**

**111060002 EECS 26' 李侑霖**

**Catalog**

**1. FPGA Question:**

**Mixed Keyboard and Audio Modules Together…………….……P3**

**By 李侑霖**

**2. FPGA Question:**

**Vending Machine…………………………………..…...………….P7**

**By 劉祐廷**

1. **FPGA Question: Mixed Keyboard and Audio Modules Together**
2. **Block Diagram**

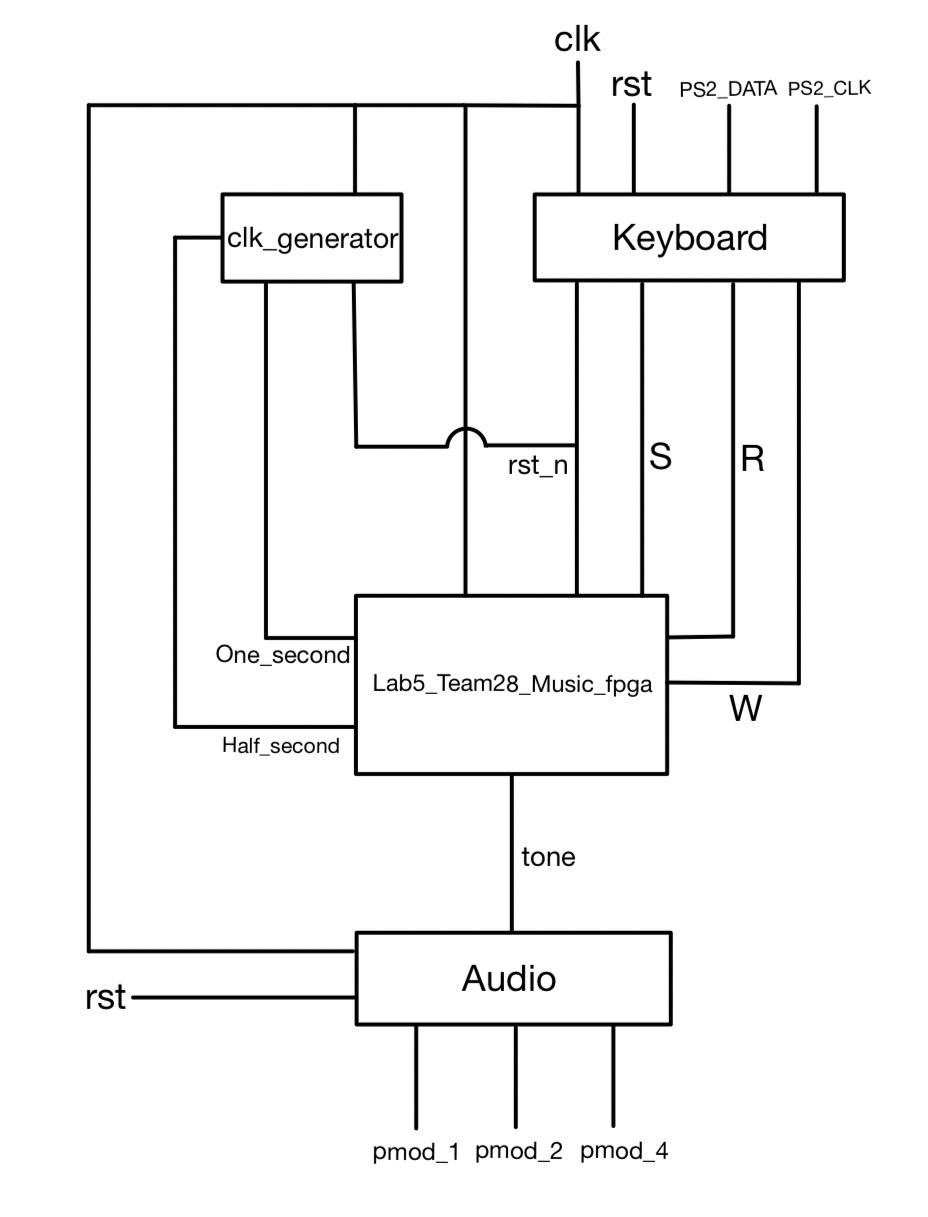


Figure 1.1

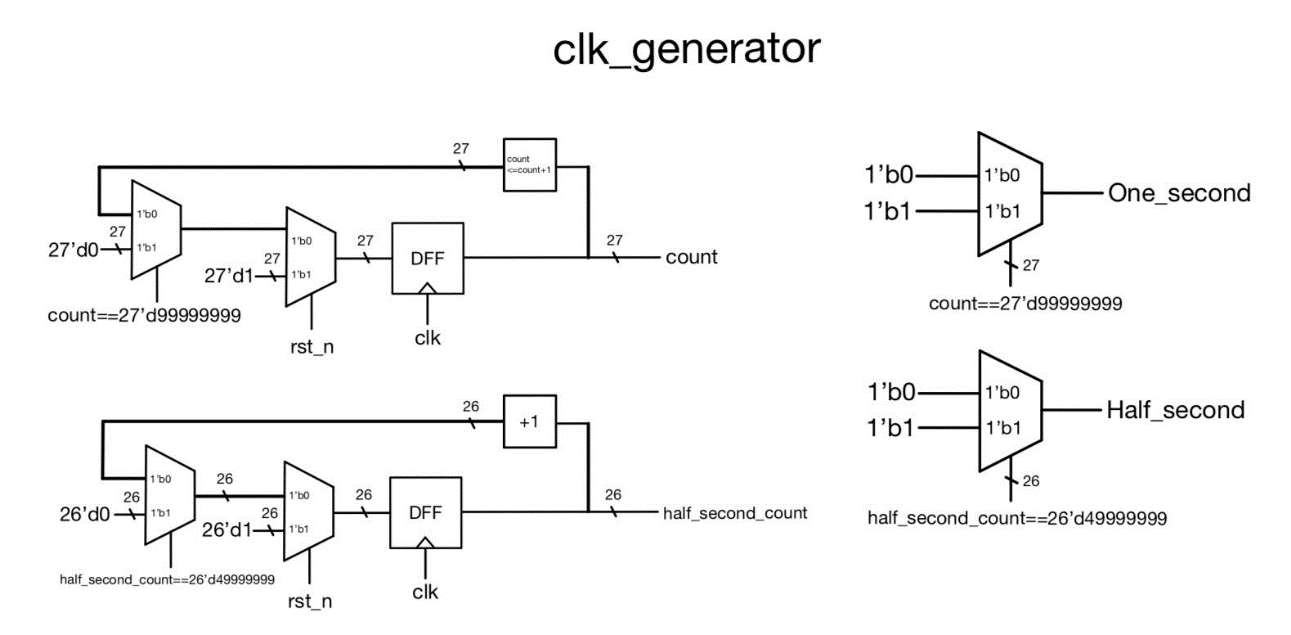


Figure 1.2

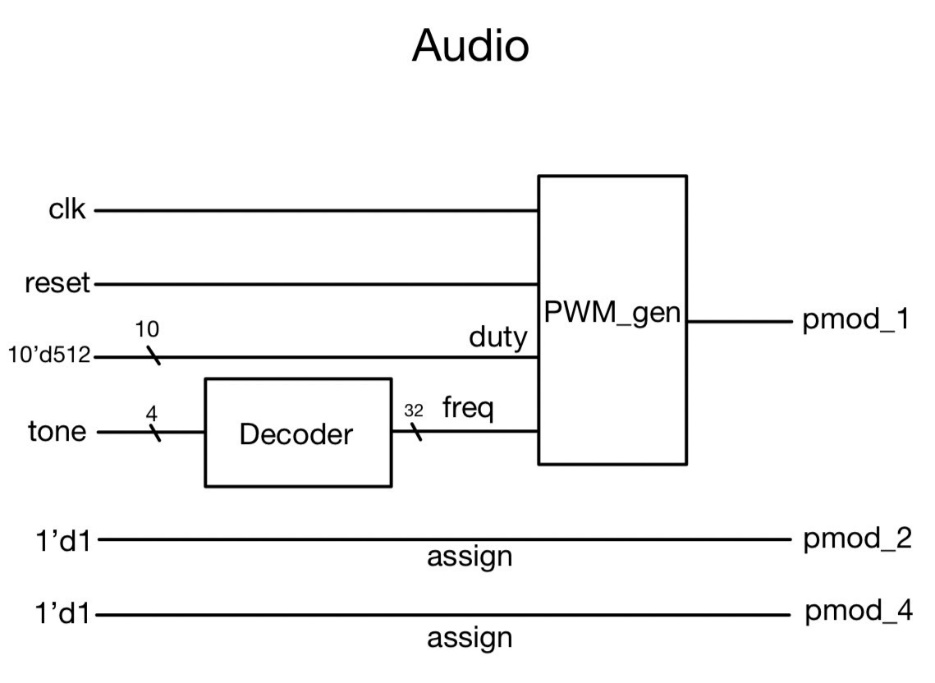
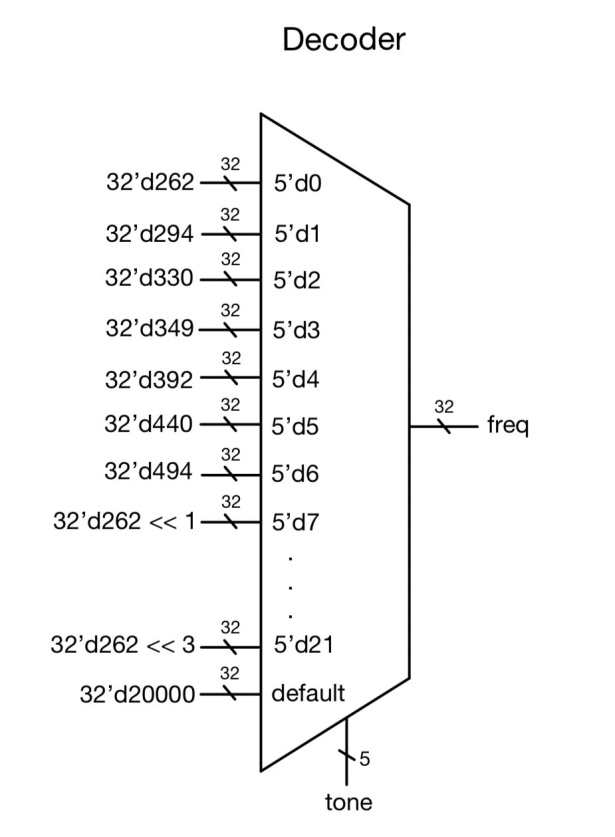


Figure 1.4

Figure 1.3

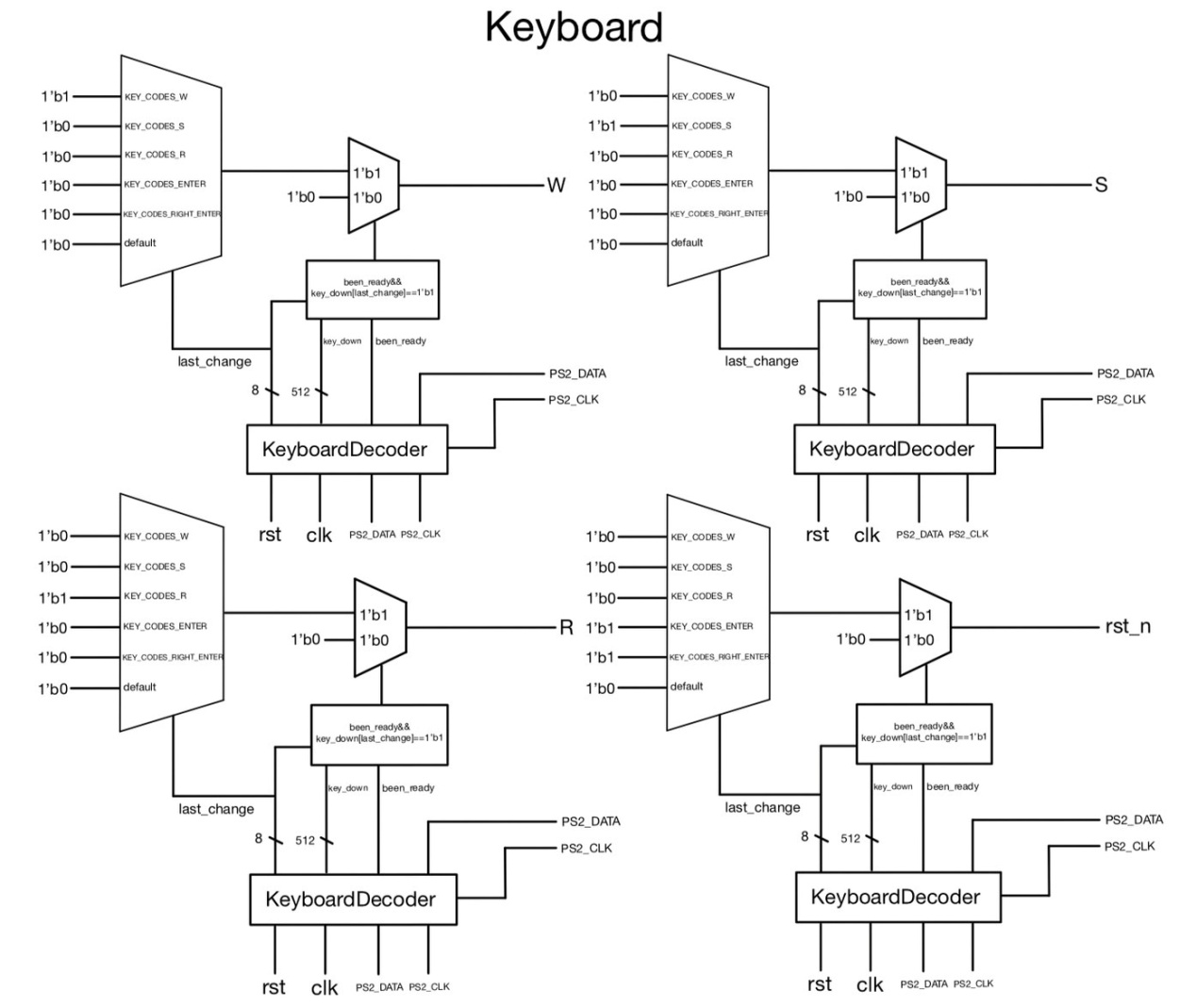


Figure 1.5

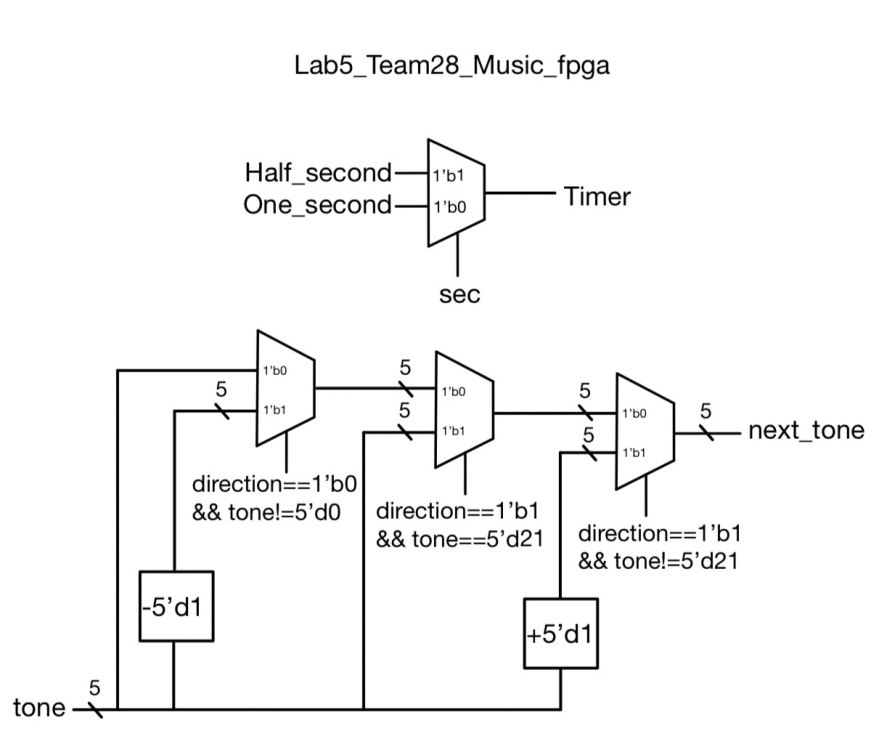


Figure 1.6

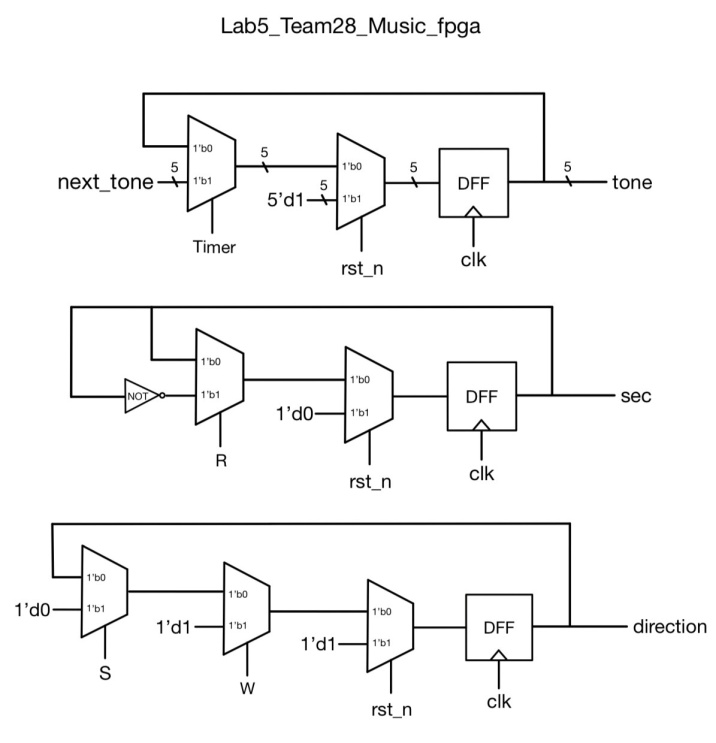


Figure 1.7

1. **Explanation**

**clk\_generator**是計算一秒和半秒的module，一秒對應到27個bits的**count**，當**count**數到27'd99999999時，將**One\_second**設為1，其餘時間設為0；半秒對應到27個bits的**half\_second\_count**，當**half\_second\_count**數到27'd49999999時，將**Half\_second**設為1，其餘時間設為0。(**Figure 1.2**)

**Audio**是用來播放聲音的module，輸入的**tone**，會進入**Decoder**算出**freq** (頻率)，再經過**PWM\_gen**後輸出**pmod\_1**，另外同時assign 1'd1給**pmod\_2**和**pmod\_4**。(**Figure 1.3**)

**Decoder**是製造出**freq**的module，將輸入的**tone**轉變成我們想要的**freq**，其中C4到B4的頻率為圖中的5'd0到5'd6，而C5為C4的2倍、D5為D4的2倍，依此類推，最後的C8則為C4的8倍。(**Figure 1.4**)

**Keyboard**這個module則是將鍵盤的資訊整理後給top module使用，其中輸出的**W**、**S**、**R**、**rst\_n**分別是按了鍵盤上W、S、R、Enter (鍵盤左邊或右邊的Enter都可以) 的訊號。

(**Figure 1.5**)

**Lab5\_Team28\_Music\_fpga**是將其他所有module整理在一起，並在此module中決定音階上下方向、播放聲音的間隔長短、以及**tone**的top module，其中**sec**便是要決定使用哪一種**clk\_generator**輸出的間隔秒數，而他受按鍵R的影響，再利用**sec**決定**Timer** (fpga播放間隔秒數)，而**tone**會受**Timer**影響，每當**Timer**的訊號一來，**next\_tone**便傳給**tone**，使其進入下一個音；此外，音階的上升或下降受**direction**影響，而**direction**受鍵盤W、S影響，若按下W，則**direction**變為1 (上升)，若按下S，則**direction**變為0 (下降)，接著**direction**會決定下一個**tone**的值 (**next\_tone**)。(**Figure 1.6, Figure 1.7**)

1. **What I Have Learned**

這次是第一次組隊進行，是一個很新的體驗，我要學會跟不同人一起合作，當中也會需要不少磨合，感覺像是模擬以後工作時，大家一起合作做一個東西的狀況，整體而言感覺還不錯。另外我也在這次lab學會如何使用鍵盤和喇叭，並利用code輸入和輸出自己想要值和聲音，又學到很多新的東西。

1. **Vending Machine**
2. **Finite State Diagram**

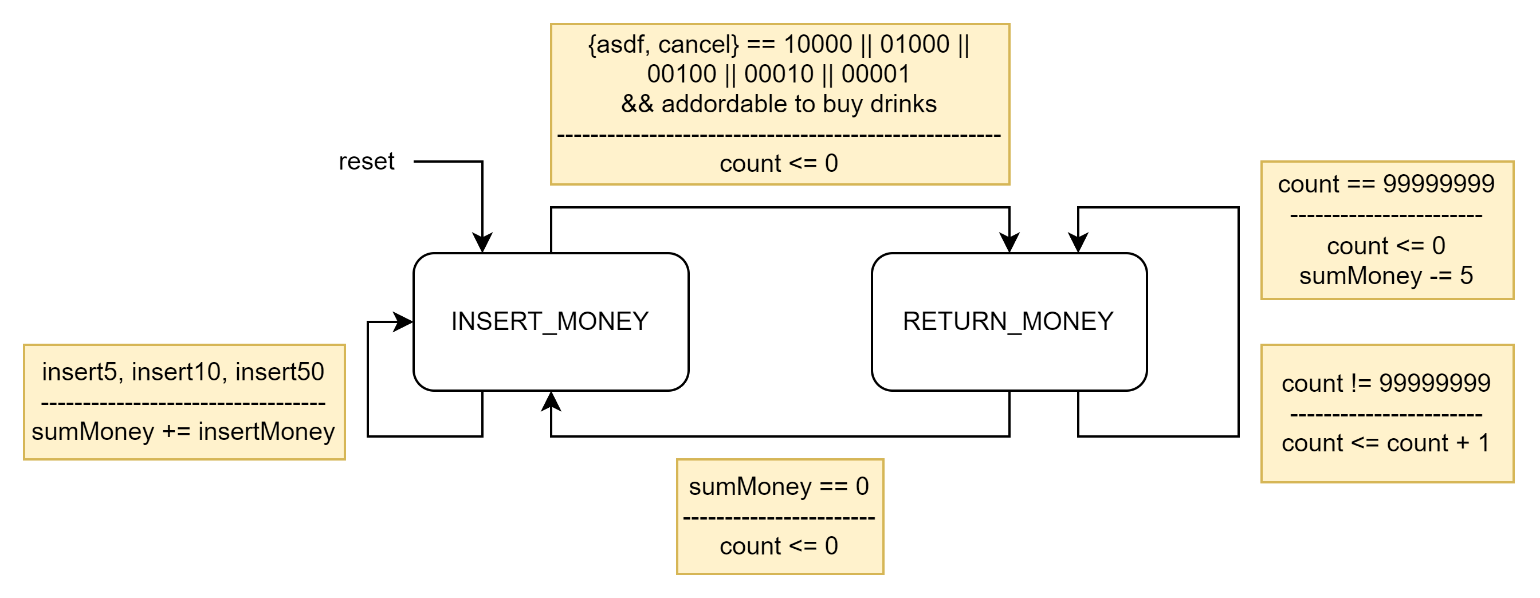
****

Figure 2.1

1. **Block Diagram**

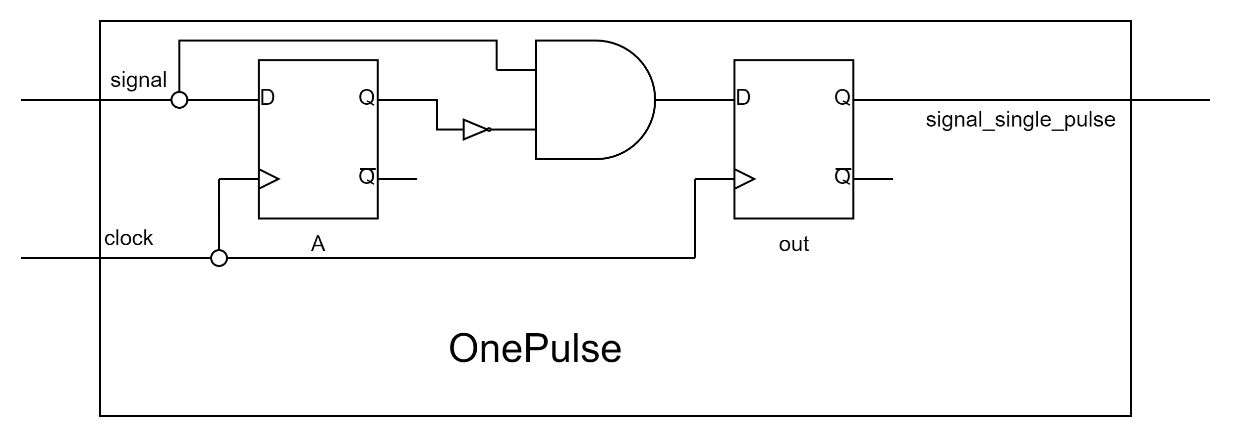


Figure 2.2

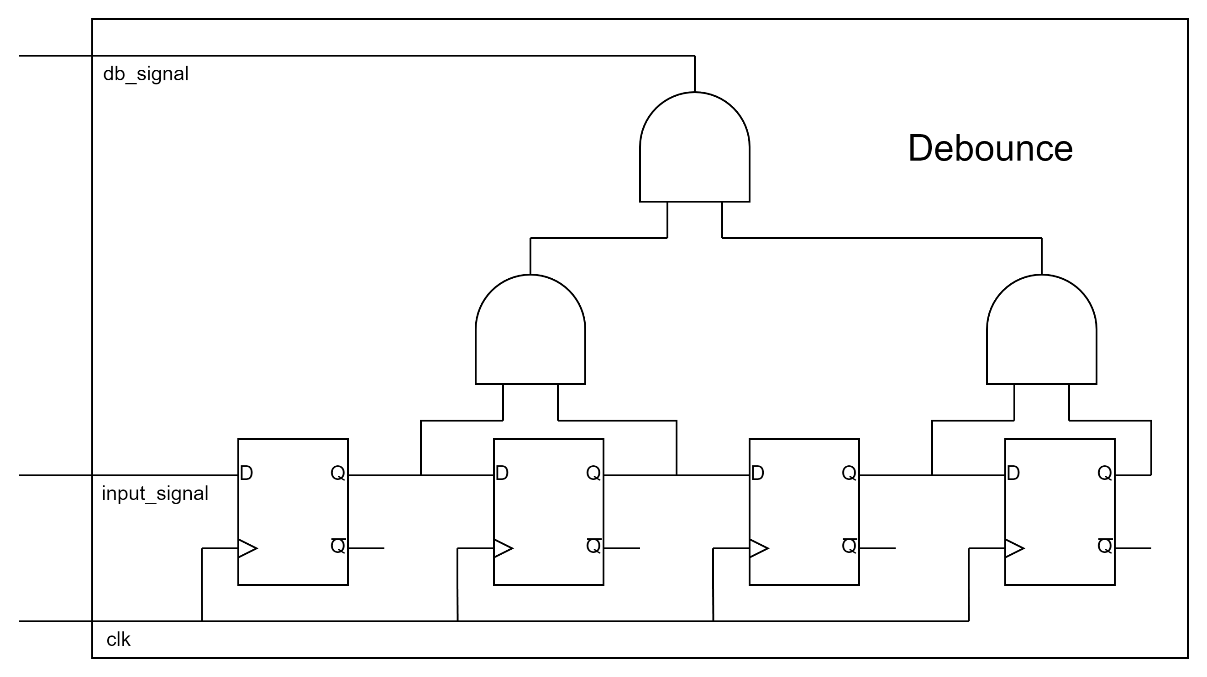


Figure 2.3

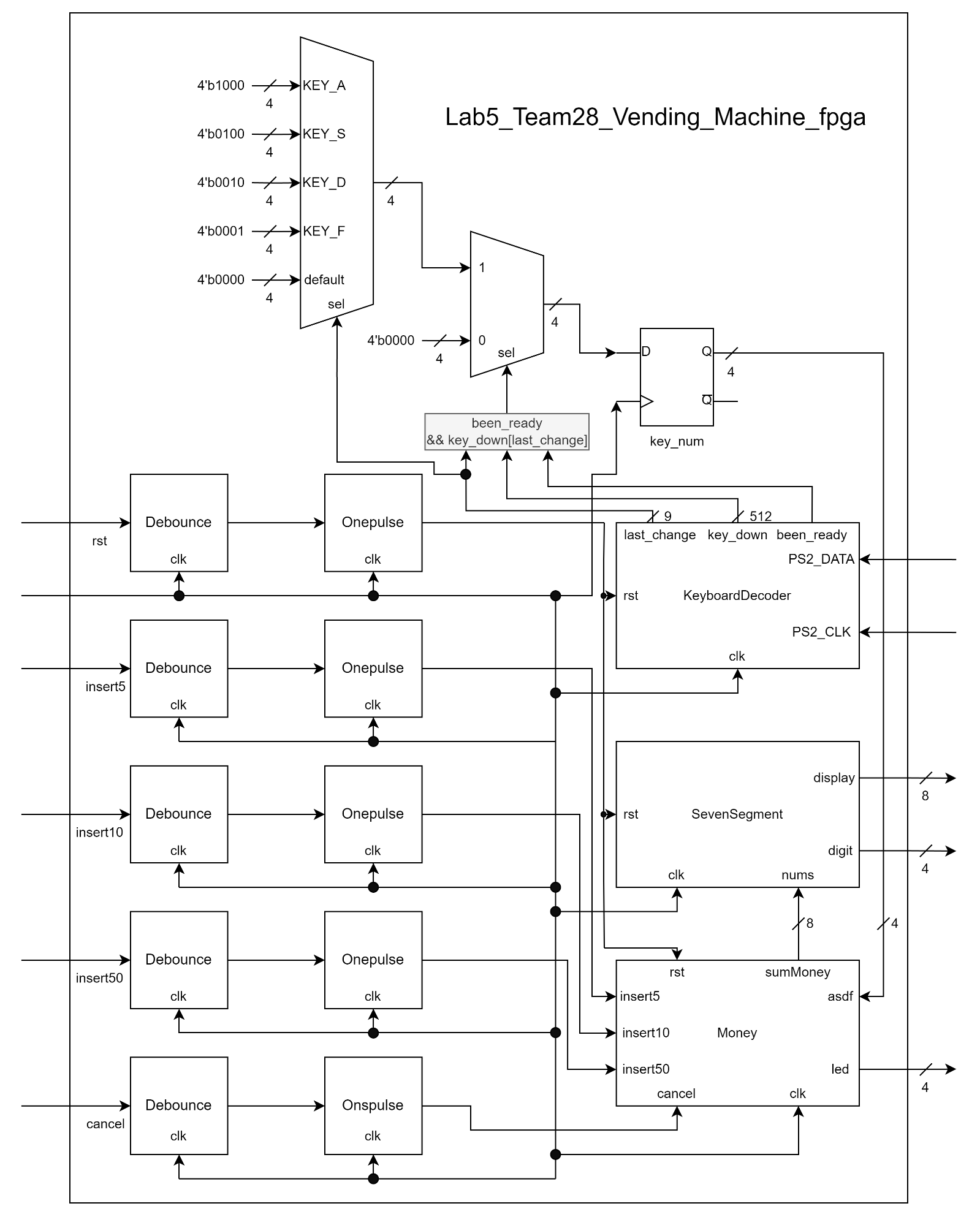


Figure 2.4

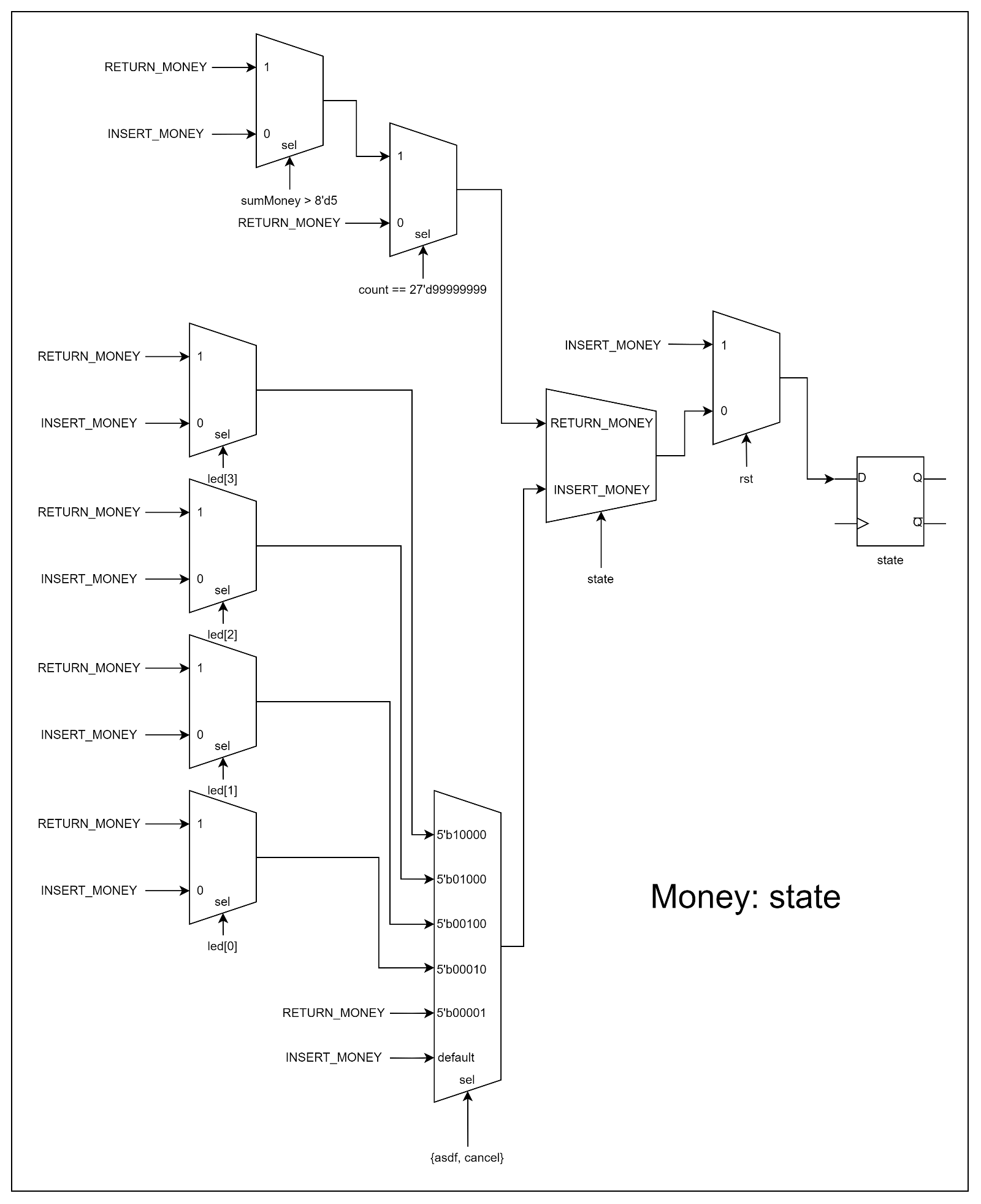


Figure 2.5

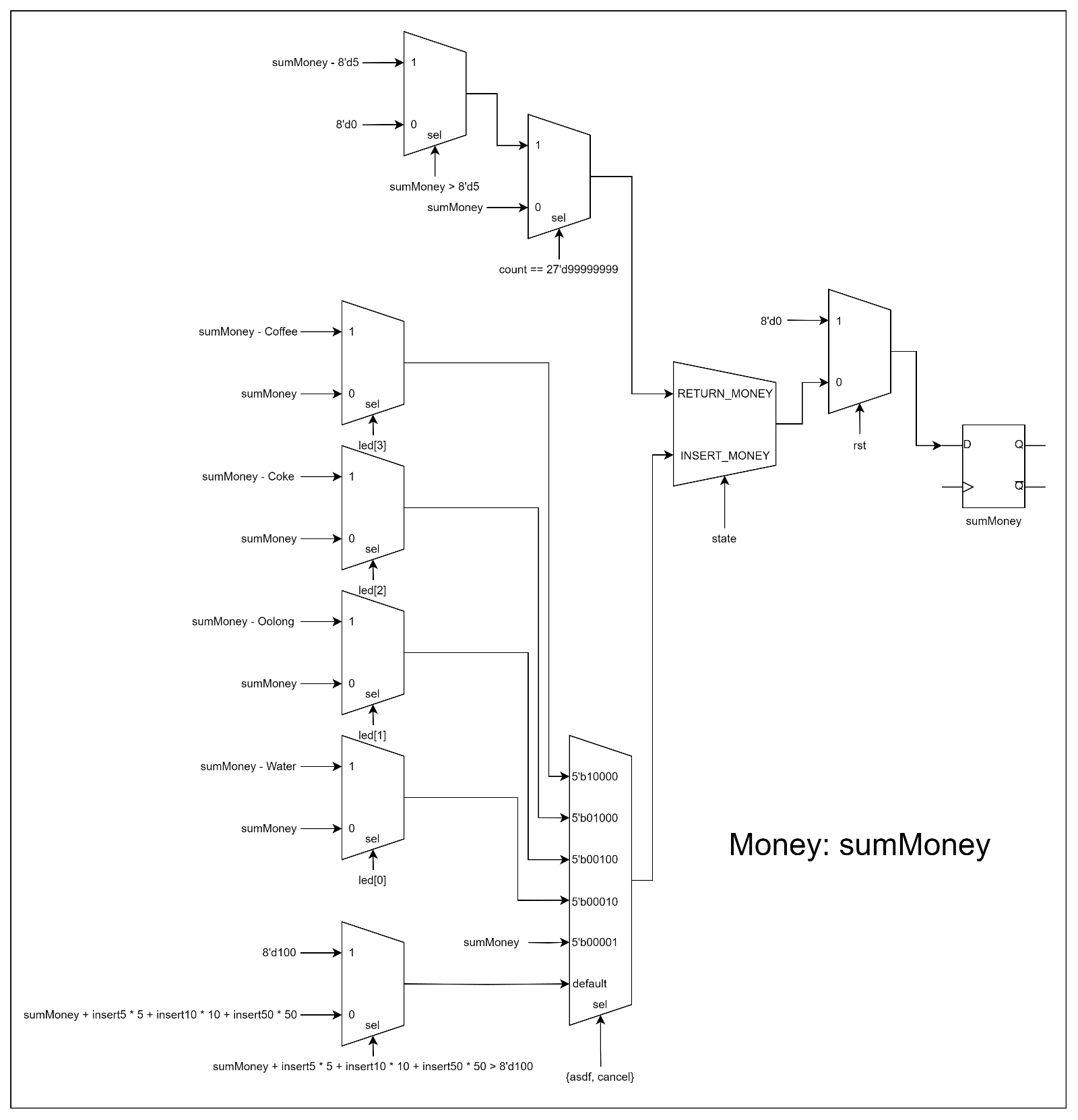


Figure 2.6

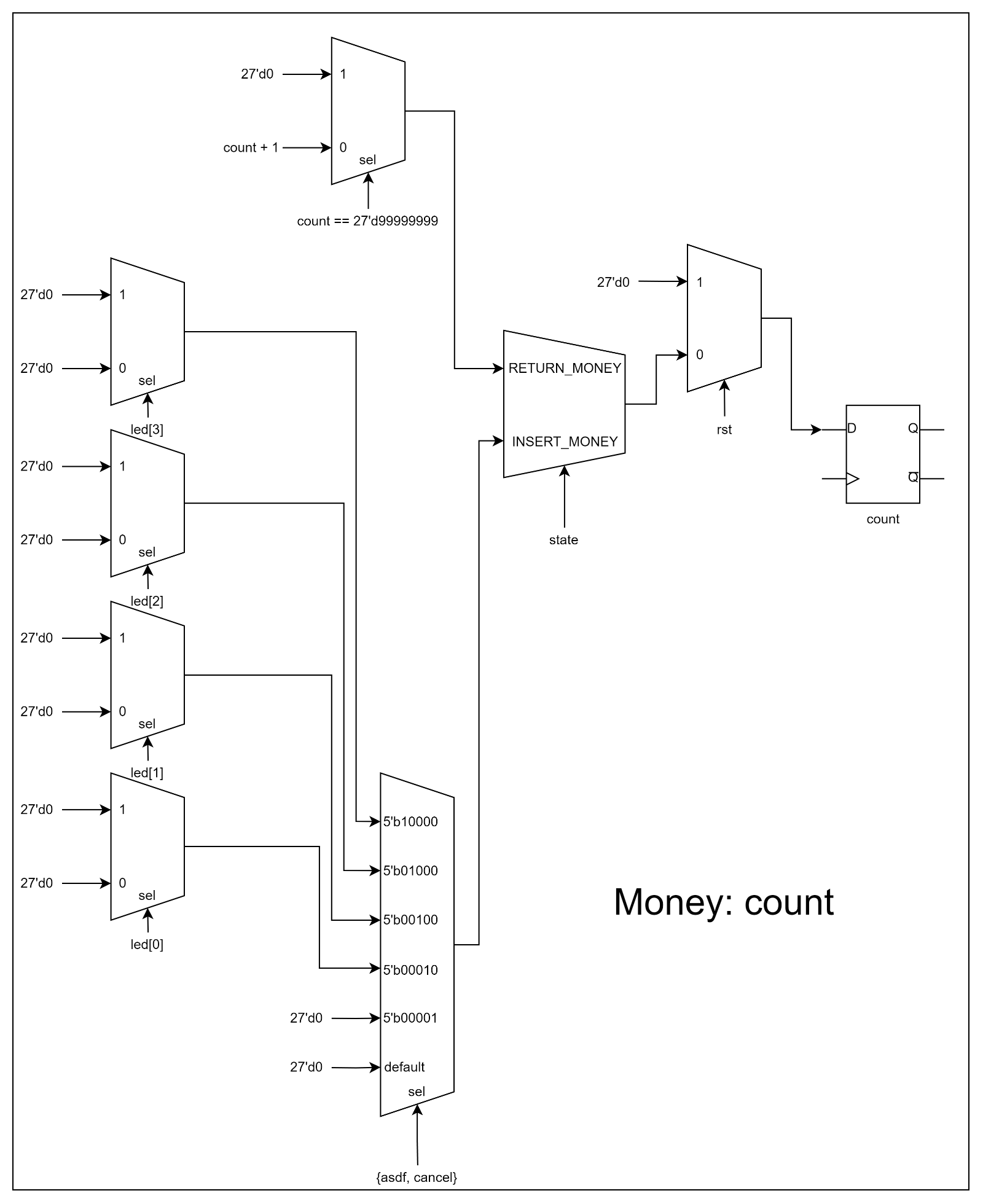


Figure 2.7

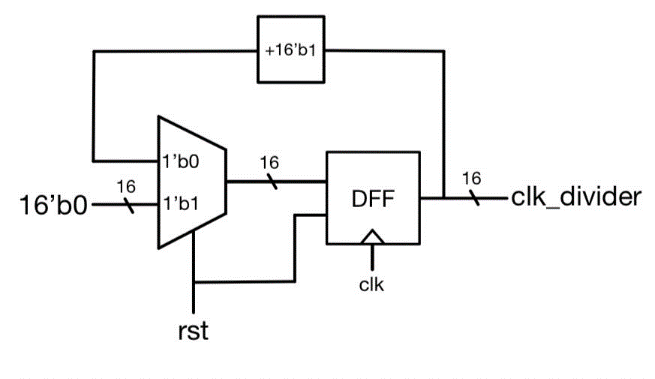
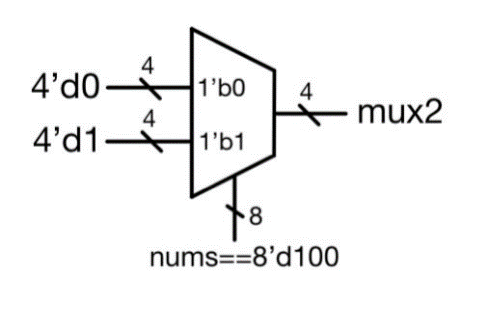


Figure 2.9

Figure 2.8

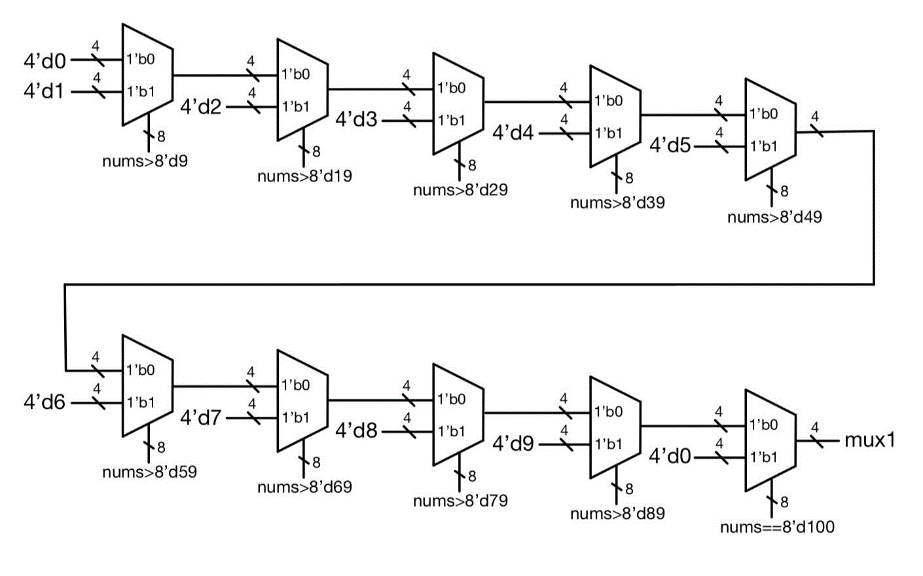
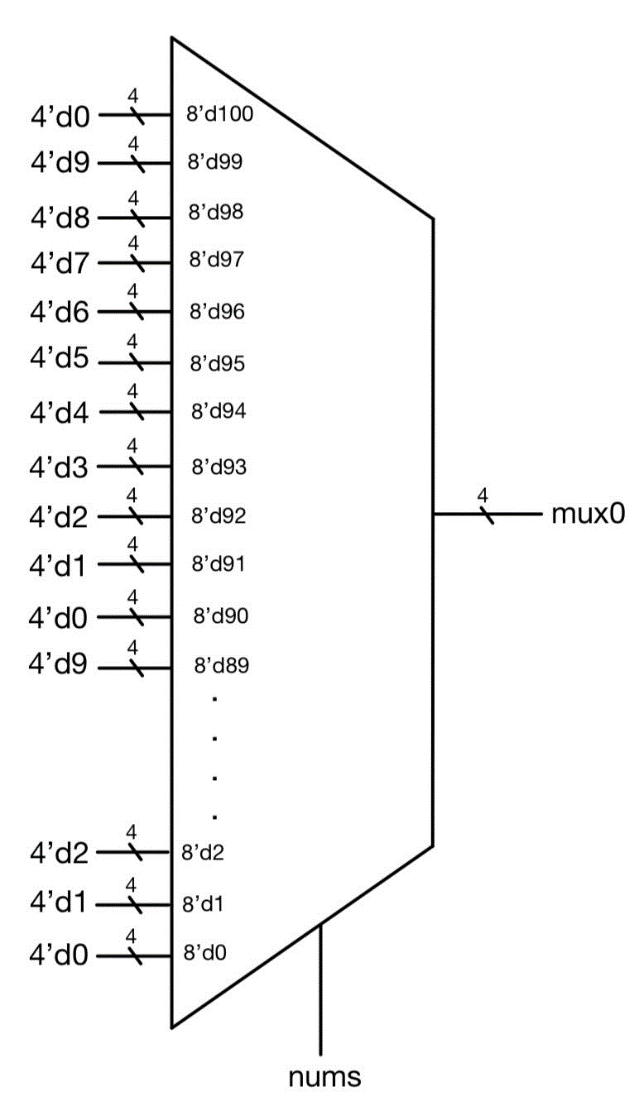


Figure 2.10

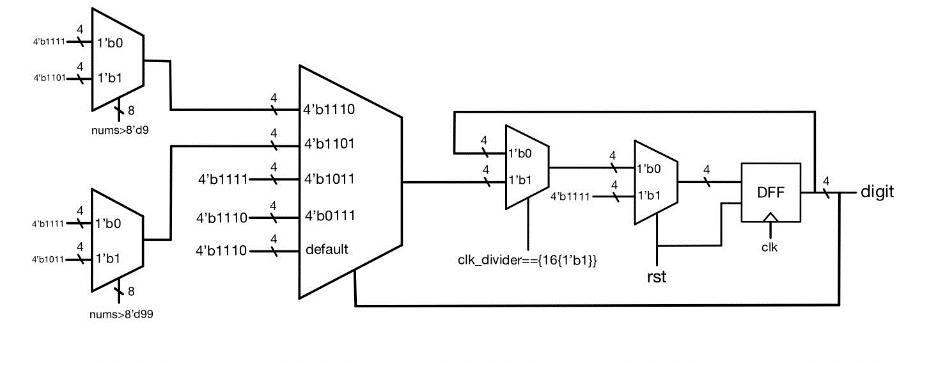
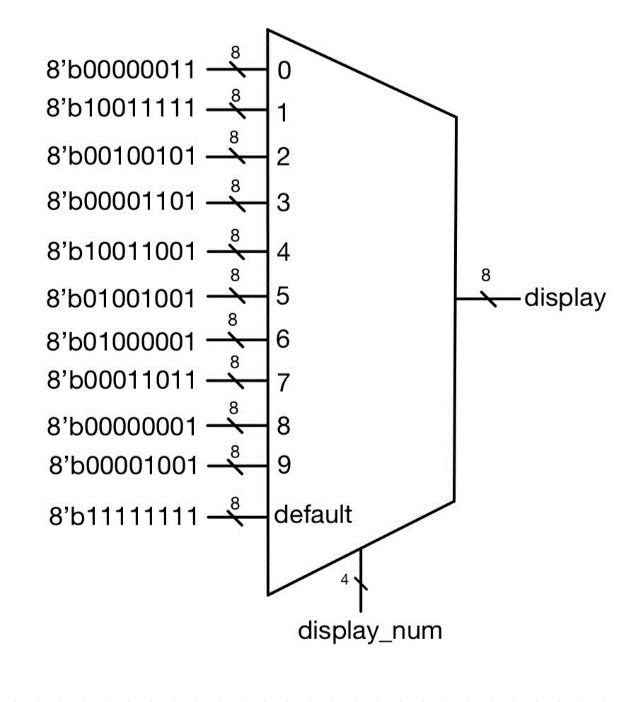


Figure 2.11

Figure 2.12



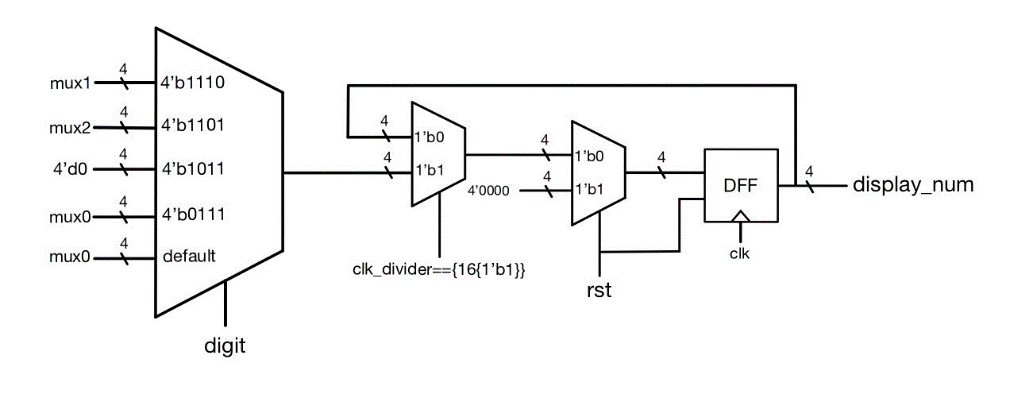


Figure 2.13

Figure 2.14

1. **Explanation**

如**Figure 2.1**所示我將使用狀態分成兩種，第一種是**INSERT\_MONEY**，一開始reset會進到這個狀態，等待使用者投錢，並用七段顯示器顯示使用者投入了多少錢，一旦使用者選擇了飲料而且投進去的錢足夠多，或是使用者按了**cancel**按鈕，則會進入第二種狀態**RETURN\_MONEY**，在這個狀態之下，販賣機對於投錢、選擇飲料和**cancel**按鈕不會做出反應，七段顯示器會以每秒減少5元的速度改變數值，直到**sumMoney**變為0，接者會回到**INSERT\_MONEY**等待後續的操作。整體的電路圖如**Figure 2.4**所示，其中**OnePulse**與**Debounce**是沿用上一個Lab使用的module (**Figure 2.2, Figure 2.3**)。Money這個module的電路圖為**Figure 2.5 ~ Figure 2.7**，大致分為三塊電路：**state, sumMoney, count**。其中**state**用來記錄狀態變化，**sumMoney**用來記錄販賣機內有多少錢，**count**則是計數器，用來devide clock。**SevenSegment**這個module的電路圖為**Figure 2.8 ~ Figure 2.14**，**mux2, mux1, mux0**分別代表百位數、十位數與個位數。

1. **What I Have Learned**

以前玩遊戲鍵盤會分成短按與長壓兩種操作，在這次lab我了解到原來這兩種狀態是使用按下與放開組合成的，或許在期末專題可以用這個特性做出比較多樣的操作組合。另外透過這次的題目，我也更了解如何去設計finite state machine，並且也更熟悉divided clock的使用方法與時機。