**Digital System Design (Fall 2018)**

**Report of Term Project**

Group Number (組別) : 15

Group Member 1 (組員1) : Student ID 0616225 Name 張承遠

Group Member 2 (組員2) : Student ID x Name x

**Title (標題) : 收銀機(Cashier!)**

**A. Problem Description (問題敘述):**

Design a cashier that simulate the real cashier.

It has a keyboard, a micro computer that do calculating, and a screen to print the sum of commodity what customer has bought.

**B. Flowchart or Procedure (流程圖或運作程序) :**

**You must know before operating:**

Cashier operator should input all the cashier command in “data.txt”, then these command will be send to fileio.v and then computer’s command line interface will show the results.

**Process:**

First, input data in “data.txt” in order to simulate the real operation of real cashier.

A file “data.txt” is used to store what cashier operator has input.

The computer of cashier will get input and then calculate the sum of all commodity.

After calculating open the draw of cashier for cashier operator to make change.

Then, close the draw and print the receipt.

**Rule of input in “data.txt”:**

1.In the beginning, you should input a 6-bit binary number that indicate password. (default password is 111111)

2.If successfully open the computer of cashier, you’ll see some welcome sentence. Then, input the customer’s number. (ex: To first customer, input 1, second customer input 2)

3.Then, we can type what customer has bought! However, you should follow the format :

“price of this commodity (space) a multiplication sign (space) numbers of this commodity (space) name of this commodity.” (ex: 100 \* 3 bananas)

4. If the customer have other commodity, type an addition sign to do addition.(+)

5.Then we can keep doing step3 until we have input all commodity that the customer bought.

**6.There are some special feature to help cashier operator to operate:**

I. If we just input a commodity and we find we have some error in the previous input, we can type “--“ to clear the previous commodity.(It can only clear “one” previous input) (清除上一個輸入)

II. If we find that we have more than one error in the previous input, we can type “c” to clear every input before. The sum of commodity will return to 0

7. If we finish all commodity that customer bought, we input “f” means finish, then the screen will show the sum of all commodity that customer bought

8. Input the way customer choose to pay, by credit card or by cash.

9.Congratulation! We have finished the first customer. Then, we go back to step 2 to continue to deal with the next customer.

**C. Block Diagram (方塊圖) :**

Fileio

Keyboard Input

sum

fdone

cdone

Cashier

st

clk

open

Open\_cashier

Generate\_receipt

**D. Definition of Inputs, Outputs, Control Signals, and Status Signals (輸入、輸出、控制訊號、及狀態訊號之定義):**

**Input Signals of “cashier.v”**

St = 1 if cashier is on.

Open = 1 when pressing the open buttom.

**Output Signals of “cashier.v”**

Open\_cashier = 1 cause the cashier open.

Generate\_receipt = 1 cause the cashier to print receipt

Cdone = 1 if successfully finish one turn of cashier’s operation.

**Input Signals of “fileio.v”:**

enable= 1 if there are still commodity in the waiting queue.

init = 0 if the commodity is the first commodity.

Custo = 1 if the previous customer has paid and gone.

Keyboard input signal:

char = + if shopkeeper want to add the amount of commodity.

char = -- if shopkeeper want to clear the previous commodity.

char = f if there are no commodity in waiting queue.

**Output signals of “fileio.v”:**

fdone = 1 if successfully get the input of cashier.

**E. State Machine Chart (SM Chart) or State Graph (狀態機器圖或狀態圖):**

St

Open/open\_cashier

open’

St’

open

Open’

/generate\_receipt

**F. Description of Verilog Code (Verilog電路模組說明):**

“fileio.v”: This file deal with the input/output, it will extract what cashier operator input in “data.txt” and then display the necessary information in command line interface. (ex: the sum of commodity at that time, receipt number and so on)

“cashier.v”: This file will simulate the process of cashier. In state0, wait for st signal become 1. In state1, if open signal become 1, open\_cashier signal become 1 means open the cashier’s draw. In state2, if open signal become 0, open\_cashier signal become 0 means close the cashier’s draw. In state3, directly go back to state0 and then generate\_receipt signal become 1 means print the receipt.

That is the rough process of operate a cashier.

**G. Description of Test Bench (Verilog測試模組說明):**

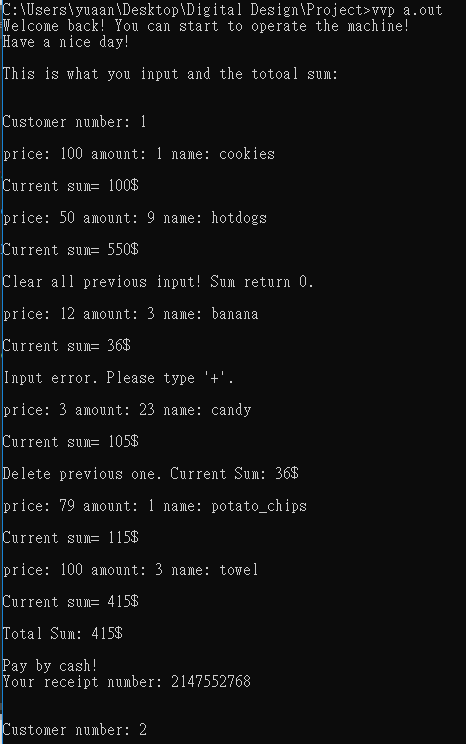
First, wait until “fileio.v” finish its task. (fdone become 1). If fdone become 1, it will start the cashier.(cause st\_cashier = 1)

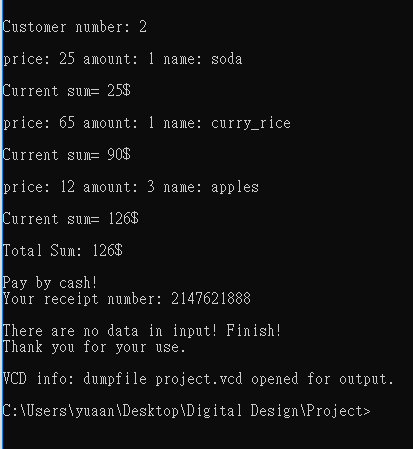
After 500s, open signal become 1 that cause the cashier draw to open and St\_cashier become 0.

After 100s, open signal become 0 that cause the cashier draw to close.

After 1000s, finish.

**H. Simulation Results (模擬結果):**







**I. Conclusions and Discussions (心得、感想、結論、及討論):**

這次的project讓我學到蠻多東西的。首先是關於主題的選定，一開始左思右想都不知道要做甚麼，而且想做的東西也不想跟別人重複或很類似，最後突然想到收銀機，不會跟別人重複而且剛好也是生活上的例子，顯示金額的螢幕可以用上課教過的七段顯示器，鍵盤可以用上課教過的keypad scanner，那時候心想這主題應該蠻適合的！因此那時就匆匆地決定要做收銀機，沒想太多。

然而到了開始認真要做的時候，才發現那時候的我太過理想化了。許多東西如七段顯示器，keypad scanner都不知道該從何取得及如何操作，但若是只有模擬收銀機的過程的話，那也太過簡單了。收銀機的功能其實不多，只有開關抽屜，印發票等等，因此我想到利用File input output的方法來做，利用電腦鍵盤取代收銀機鍵盤，終端命令列取代收銀機顯示螢幕，用主機取待收銀機的計算機，因此這個Project最難的地方其實是處理File input output。

經過這次專案，讓我更了解了Verilog的地位，以及他與其他語言的差別。像是verilog的input都要事先打好在testbench裡面，不像其他語言如C, python可以從終端機收到使用者打的input再進行處理。更重要的是verilog是硬體描述語言，所以若有個硬體來接自己的code會比較有成就感，雖然會需要花更多的時間。同時我也希望之後有機會可以做到這樣的事。這次的專題因為對verilog還不太熟所以做的很普通，但我也學到很多事情，相信在未來這些所學都會被用到。很感謝這次的機會，將來我會繼續進步，愈來愈好的。

**References (參考資料) :** (請說明各參考項目對你的專題提供那方面資料)

1. [收銀機詳細使用方法](https://cht.tw/x/2ugnl)
2. [收銀機實際操作影片](https://www.youtube.com/watch?v=6A_SJ_Q4RzE)
3. [verilog 字串處理](https://www.cnblogs.com/hfyfpga/p/4316262.html)
4. [Verilog 一些範例](https://hom-wang.gitbooks.io/verilog-hdl/content/Chapter_01.html)
5. [Verilog 輸入輸出處理](https://www.cnblogs.com/oomusou/archive/2008/02/11/1066839.html)
6. 上課講義