Final Report of Computer Organization & Assembly Languages:

tdb - A TOY Machine Debugger

B94902117　馮禹翰

1. 動機

這門課在前半段教的 TOY Machine Architecture 確實讓我受益匪淺，特別是在對於 CPU 的結構與運作上獲得了很重要的觀念。之後教授笑著說他用八個小時完成了一套針對 TOY Machine 的組合語言以及其組譯器和連結器（linker），並且出給大家一個大作業，讓許多人在連續假期難以露出如同教授般陽光的微笑。追究其原因，其實是 TOY Machine 的模擬器（simulator）沒有提供良好的除錯（debug）環境。要在幾十行的程式碼（code）當中尋找錯誤（bug）雖然不是太難的事情，不過對於才剛開始接觸組合語言的我們就有得受了。

因此藉由期末專案的機會，希望能夠練習寫一些有點「實用性質」的程式，一方面萬一以後自己又需要碰到 TOY Machine programming 的時候可以輕鬆許多，另一方面也期望自己的程式可以讓更多第一次接觸組合語言程式設計的同學能夠擁有較爲完整的測試與除錯環境。

2. 模擬器架構

用來進行模擬 TOY Machine CPU 核心的部分爲 class GToyMachine。此一類別與助教用來評分作業的 toyvm 在功能上頗爲類似，不過因爲我需要用來處理中斷點（break point）和其他除錯器（debugger）必要的基本功能，加上這是我自己的作業，所以最後模擬器核心的類別（class）全部都是由本人所撰寫完成。以下就 GToyMachine 類別的數個較爲重要的方法（method）進行簡短說明：

int GToyMachine::RunInstruction();

此方法爲該類別的核心，即執行一個單位的指令（instruction）。回傳值（return value）爲執行完此指令後 TOY Machine 的狀態，可能的狀態包括了「停止（halt）」、「執行中（running）」、「中斷（interrupted）」。其中「中斷」之狀態是用來處理中斷點的，若此方法被呼叫且正要執行一個設有中斷點的指令時，此方法會暫停執行該指令並回傳此一狀態。若中斷的指令被第二次執行，則此方法會按照正常的程序執行之並在完成之後回傳「執行中」狀態。

void GToyMachine::SetBreakPoint(unsigned char Address);

顧名思義，這一方法是用來設定一個位於記憶體位置爲 Address 的中斷點。此類別所定義的中斷點均是「read/write/execute」型式的；也就是說，被設立中斷點的記憶體位置無論被讀取、寫入、執行，都會引發中斷（interrupt）的行爲。

int GToyMachine::ClearBreakPoint(unsigned char Address);
用來清除設定位於 Address 的一個中斷點。

int GToyMachine::DumpMemory(unsigned char Address, unsigned short Length, word* Dest);
此方法可以將部份記憶體區塊的內容提取（dump）出來至呼叫者（caller）所指定的記憶體緩衝區。Address 爲欲提取的 TOY Machine 記憶體起始位址，Length 爲希望提取資料的記憶體單位長度，Dest 則爲呼叫者要用來存放此一資料的緩衝區位址。

int GToyMachine::DumpRegisters(word* Dest);
提取全部共 16 個 TOY Machine 暫存器的內容至呼叫者所指定的 Dest。

int GToyMachine::Load(const char* Path);
載入一個 *.toy 檔案至模擬器的記憶體，同時並初始化暫存器當中的數值。若檔案開啓成功則回傳 0，若開啓失敗則會回傳 -1。

void GToyMachine::PrintInstruction();
將 instruction pointer（IP）所指向的指令反組譯（disassemble）並顯示於標準輸出介面上。

3. 除錯器介面

由於過去有使用過 GNU gdb 的經驗，因此我刻意模仿了 gdb 若干我認爲基本且必要的功能於 tdb 當中。以下則簡單說明 tdb 所支援的諸指令（command）：

quit
離開 tdb 程式。

run
執行 TOY Machine 程式直到程式呼叫 halt 指令或是到達一中斷點。

nexti
執行下一個指令（instruction）。

setbp address
在 TOY Machine 記憶體位址 address 處設定一個中斷點，被設立中斷點的記憶體位置無論被讀取、寫入、執行，都會引發中斷（interrupt）的行爲。

clrbp address
將 TOY Machine 位於記憶體位置 address 處的中斷點移除。

listbp
列出所有已設立之中斷點。

disasm
將目前 instruction pointer 所指向的指令反組譯並輸出。

dumpreg
輸出 TOY Machine 的十六個一般需求(general purpose)暫存器以及 instruction pointer 之內容。

dumpmem
輸出 TOY Machine 記憶體當中的所有資料。

reset
重新設定 TOY Machine 模擬器回到初始狀態，並清除所有中斷點。

help
顯示所有可用的（available）除錯器指令清單與簡短說明。