組譯器專案報告

學號：41147011S

姓名：鄭聿喬

系級：資工 115

程式架構

本組譯器專案實現了一個針對 SIC/XE 虛擬機的兩次通過組譯器，包含以下主要組成部分與功能模組：

1. 操作碼表 (OPTAB)：
   儲存所有支援的操作碼及其對應的機器碼，例如 LDA、STA、COMP 等。
2. 寄存器表 (REGISTER_TABLE)：
   定義所有支援的寄存器名稱與其對應的編號，例如 A、X、L 等。
3. 檔案讀取模組：
   從輸入檔案讀取組合語言程式碼，並進行前置處理去除空行與多餘空格。
4. 符號表生成模組 (Symbol Table Generation)：
   第一遍組譯中處理標籤並建立符號表，儲存每個標籤對應的地址。
5. 目標碼生成模組 (Object Code Generation)：
   第二遍組譯中根據操作碼與操作數計算目標碼，並依規格生成目標程式碼的各類記錄（例如 H、T、M、E 記錄）。
6. 組譯器指令處理：
   支援 START、END、BYTE、WORD、RESB、RESW 等指令。
   基本實現 BASE 指令，用於設定基底寄存器。
7. 格式處理：
   支援格式 1、2 和 3/4 的指令，並正確處理立即定址 (#)、間接定址 (@)、簡單定址，以及 X 位元的處理。
8. 錯誤檢測：
   檢測重複標籤的定義，並提供錯誤訊息。
   處理 BASE 指令中未定義的標籤情況。
9. 目標程式碼輸出：

按照固定格式生成 H、T、M、E 記錄，並處理每行目標碼的長度限制。

學到的與經歷的

我在剛開始撰寫的過程非常沒有頭緒，對這個最深的印象就是第三次作業的手動轉換，由於不限程式語言，所以我直接選用感覺最簡易的 Python 3 ，過程中我有參考[網路上的資源](https://hackmd.io/@tommygood/SIC-XE-Assembler?utm_source=preview-mode&utm_medium=rec#SIC-XE-Assembler)，一步一步 debug，我過程中遇到很多次地址不對，format 判斷錯誤，還有位元運算的 nixbpe 的 flag 設置錯誤，解決後還是有一些指令錯誤，但最後我還是成功完成。

完成後我非常有成就感，也真正了解這是如何運作，因此我非常感謝這次的專題，不僅讓我學習到課本上的知識並實作，還培養了我解決問題的能力。

程式碼附註

1. 本組譯器的程式碼實現於 Python 3，具有跨平台的可移植性。
2. 執行方式：
   使用 python assembler.py <檔案路徑> 指令執行組譯器，輸入檔案需符合 SIC/XE 的語法規範。

程式碼原創性聲明

本組譯器程式碼完全由我鄭聿喬本人所撰寫，未參考任何他人未經授權的程式碼。

給 G.H.Hwang 的建議與反饋

我認為這個專題，可以考慮放置在學期前半部，雖然很花時間，但卻可以真正了解課堂上所講述的內容，最好的學習就是動手做，也建議可以減少期中期末，用專題的方式取代，因為有些人可能很不擅長考試 ex:我，但可以感受老師上課的用心與辛苦，所以也謝謝老師的教導。

撰寫時間：2024/12/24