Software Design Document (SDD)
Assignment #2 Assembler for SIC/XE computer
CS530, Spring 2024

Team:
Carlos Lopez, cssc4002, RED ID: 827117558
David Granda-Ventura, csscxxxx, RED ID: 824371438

Overview:
In this project, this team will design and implement a two pass assembler program for a SIC/XE machine that takes SIC/XE source files and generates listing and SYMTAB files. This will be done by following the assembler algorithm procedure from Chapter 2 of the textbook. The format for the listing file will be based on that from Chapter 2 of the textbook, as well.

Goals:
1. Open multiple SIC/XE source files (sample.sic), or handle no input
2. Use two pass assembler logic from text book to scan each file
3. Process every assembler directive and assemble each instruction for each file
4. Print the SYMTAB created while assembling instructions into a ".st" file for each file
5. Create listing file from assembled instructions into a ".l" file for each file

Project Description:
The purpose of this team project is to demonstrate essential software engineering and course learning concepts with C/C++ programming skills by creating a two-pass SIC/XE assembler. This assembler will be able to open given XE source files and generate a listing file and SYMTAB file from the instructions. This assembler will use the OPTAB and SYMTAB data structures to look up mnemonic operations codes to translate to machine language and store addresses to labels, respectively. These addresses are found by using the LOCCTR variable that is initialized to the address specified in the START statement of the input source file.

Plan of Action and Milestones:

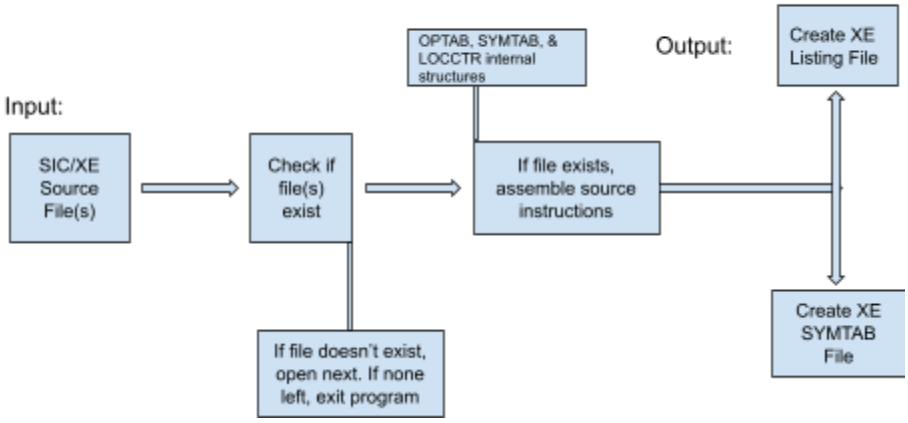
	Milestones	Target Date
Phase 1	Project Brainstorm - Specifications gathering - Identify requirements/goals	2/29/24 – 3/5/24
Phase 2	Pseudocode - Outline design layout - Determine roles in project - Assign work to be split up	3/5/24 – 3/12/24
Phase 3	Initial Stage Coding - Begin coding from pseudocode - Reanalyze time constraints - Determine possible additional requirements - Source file input is ready to be open and read	3/12/24 – 3/26/24
Phase 4	Last Stage Coding - Debug initial stage code - Reassess any additional requirements and build on it - Find areas of concern - If needed, contact the resource system for help (office hours, peers, etc.) - Prepare for finalization phase	3/26/24 – 4/9/24
Phase 5	Finalization - Debugging - Correct output formatting - Perform testing to validate completeness	4/9/24 – 4/18/24

Requirements:

- Knowledge of C/C++ Programming Language
    - C/C++ source files and include/header file
- C++ Programming Language and GCC Compiler Installed
- MAC OS or Windows 10 Operating Systems
- Makefile and README file
- Test files

System Design/Specifications:

Data Flow Diagram



Development Environment:

- Operating System: MAC OS System
- Compiler: GCC Compiler
- EDORAS from SDSU

Run/Test Environment:

We will use the provided sample test files found on the assignment page on Canvas first to test our program first. But, we will also make our own test case source files. During the final development phase, we will compile our program on EDORAS and test it on the server to ensure completeness and correctness.

Code and Test Plan:

1. Code Review:
    a. From the code we have split up, we will do peer code reviews to check for errors.
    b. Ensure that requirements are being met and reassess our deadlines.
2. Unit Testing:
    a. Each method of our program(s) will be tested including adding print statements to our code to verify everything is working as intended.
3. Integration Testing:
    a. In late phase 4, we will prepare for the finalization phase of the project by combining our code.
    b. We will debug our complete code and test with files provided and files created.