**Software Design Document (SDD)**

**Assignment #2 DIS Disassembler for XE computer**

**CS530, Fall 2021**

**Team:**

Note – can include things like who took lead/support for each task during the project

James Lee, cssc3719,

Joshua Boltz, cssc3745

Ayoub Rammo, cssc3733

**Overview & Goals:**

With this project, we aim to be consistently communicating with each other and progressing further into the project in a timely fashion. We plan on meeting once a week and discussing any major errors that arise as we are coding. With the goal of wanting to learn the ins and outs of this project, we believe that it is a key factor to talk about these errors and fix them as a group to learn through experience.

**Project Description:**

Our program will take in SIC/XE assembler listing files and output and executable object file/s for the XE machine. It will also print out a separate file for the ESTAB.

**Plan of Action and Milestones:**

Note – tasks, milestone (dates) for each task, dependencies/critical path

1. We plan on meeting once or twice a week to keep in check with our errors and bugs.
2. A main task is the creation of the function that will store 5 variables listed below. All three of us will be creating our own function for this and will decide on one function to keep and move forward with.
3. Each Person’s Tasks:
   1. James Lee: Will be in charge of Converter Pass 1(Finish by 3/28/21), Main Function (Finish by 3/18/21), and Memory Check (Finish by 3/28/21).
   2. Joshua Boltz: Will be in charge of Converter Pass 2 (Finish by 3/28/21)
   3. Ayoub Rammo: Will be in charge of Converter Pass 3 (Finish by 3/28/21), Converter function (Finish by 3/21/21).

**Requirements: (error messages), (input argument bounds), etc.**

* Must have at least one input argument to create an output file, otherwise prints error and terminates.
* Scan the file and run a check on the memory mapping.
* Checks to make sure that the format 3 and format 4 instructions are making memory references within the scope of the program’s memory space, and if not, it will output an error message and terminate the program.
* The program will also print out the ESTAB in a separate file.
* We created a separate function that will store 5 variables: Loc, Symbol, OP, Variable, Object Code.

**System Design/Specification:**

* FUNCTION NAMES WILL BE CHANGED FOR FINAL DRAFT!!
* Main Function:
  + This function will check to see if there are 0 arguments, or more. With the case of 0 arguments, the main will print out an error and then terminate the program. Otherwise, it will continue onto the converter function.
* Converter Function:
  + This function will read through the argument list and call passes on each of the listing files that were inputted as arguments. This function will also print the estab after every function has been run through and completed. It will also output a 0 or 1 value depending on whether the memory check contains an error or not.
* Pass 1 Function:
  + This function will fill up the empty symtab created in main function. After which it will also find the ExtDef, ExtRef, start location and end location to add to the symtab. The function will then go on to return the filled in symtab.
* Pass 2 Function:
  + This function will create the header record, D record, R record, and text records. It will also call the memory check function whenever an ExtDef or ExtRef is prevalent.
* Pass 3 Function:
  + This function will be creating the modification records for each Extref and will also create a modification record when an ExtDef is subtracted to or from a non-relative address. This function will also be creating the end record.
* Memory Check:
  + This function will check to see if the addresses are in bound and valid. This will then return either a 0 or 1 depending on whether the bounds were met or not.

A picture containing text

Description automatically generated

**Development Environment:**

Joshua Boltz - MobaXTerm

James Lee – Visual Studio Code

Ayoub Rammo – Visual Studio Code

**Run/Test Environment:**

Using a ssh terminal or linux terminal, we will test out our code by running sample input SIC/XE assembler listing files through our code to receive desired output. We will also be adding breakpoints throughout the code in order to get a better picture on where errors will lie and how we can go about fixing them. We will be using the gcc compiler to finalize and run our executable object code.