# Design Notes

## Overview

The assembler is structured in a modular and straightforward manner, employing a two-pass assembly approach to convert assembly source code into machine code.

## Key Classes

1. Assembler

* Core class implementing the assembler logic.
* Maintains the current address and the symbol table.
* Coordinates the assembly process via two main methods:
  + firstPass(String inputFile): Reads and cleans input source, extracts symbols/labels, and tracks addresses.
  + secondPass(ArrayList<String> inputLines): Parses instructions, encodes machine code, and generates output files.

1. OpCodeTables

* Static utility class containing mappings of instruction mnemonics to their binary opcode values.
* Provides opcode lookups for different instruction categories, e.g., arithmetic, load/store, etc.

## Important Methods in Assembler

* parseOperands(String operandString): Parses and trims instruction operands from a raw string into an array.
* lsInstructionParse(String[] instructionComponents): Handles parsing and encoding of load/store and related instructions into machine code.
* Instruction Handlers: Multiple methods such as:
  + handleArithmeticLogic(): Processes arithmetic and logic instructions.
  + handleShiftRotate(): Handles shift and rotate instructions.
  + handleIO(): Processes input/output instructions.
  + handleMisc(): Handles miscellaneous instructions like HALT and TRAP.
  + handleLSOther(): Passes load/store instructions to parsing logic.
  + writeDataToFile(), generateListing(): Manage output file writing for assembled code and listings.

## Assembly Flow

* The assembler reads the entire source code during the first pass, collecting symbol definitions and assigning addresses.
* In the second pass, each instruction line is parsed and converted to corresponding machine code using the symbol table and opcode mappings.
* Final outputs include a listing file for debugging and a load file with raw machine code.

A diagram of a process

AI-generated content may be incorrect.