Name: Digvijay Jondhale
Roll No: PC-32
PAN: 1032201770
Batch: C1

SSC Lab Assignment No. 2

Title : Design of Pass 2 of Two Pass Assembler.

Aim : Design suitable data structurs and implement Pass 2 of 2 Pass Assembler for pseudo machine.

Objective: Design suitable data structures and implement Pass 2 of 2 Pass Assembler for pseudo machine. Subset should consist of a few instructions from each category and few assembler directives.

Theory: Design Specification of an Assembler:

In addition to the analysis phase (Pass I), as mentioned earlier, the design specification of an assembler also includes the synthesis phase, which covers the generation of machine code or object code from the processed source code. The synthesis phase typically includes:

Code Generation: In this phase, the assembler generates the actual machine code instructions using the information collected during pass I, such as the symbol table.

Object File Generation: It creates an object file or an executable file containing the machine code, instructions and possibly additional information needed for loading and execution.

Memory Allocation: Determines the memory addresses for the generated machine code instructions, especially for instructions that use labels and symbols.

Relocation: Handles relocatable code, adjusting memory addresses as needed for loading at different addresses.

Design of a Two-Pass Assembler - Algorithm for Pass 2:

Pass 2 in a two-pass assembler is responsible for generating machine code and performing additional tasks. The algorithm for Pass 2 typically involves:

Initialize the location counter (LC) to the starting address, as specified in Pass 1.

Read the intermediate representation (usually created during Pass 1), which includes the source code, labels, operation codes, and operands.

Process each line of the intermediate representation.

Translate each assembly instruction into its corresponding machine code representation using the information from the OPTAB and the symbol table.

Handle addressing modes, immediate values, labels, and symbols, and generate the appropriate binary code for each instruction.

Write the generated machine code to the object file or executable file.

Increment the LC as instructions are processed.

Continue until all instructions have been processed.

Perform any necessary back-patching or relocation adjustments as specified during Pass 1.

Error Listing and Error Handling:

Error listing and error handling are critical aspects of any assembler. Error handling in Pass 2 involves checking for errors in the assembly code, such as invalid instructions, undefined symbols, or addressing errors and providing meaningful error messages to aid in debugging. Here's how error listing and handling work:

Error Detection: During Pass 2, the assembler checks each instruction for syntax and semantic errors.

Error Detection : During Pass 2, the assembler checks each instruction for syntex and semantic errors.

Error Reporting : When an error is detected, the assembler generates an error message indicating the line number, type of error, and a description of the problem.

Continued Processing : The assembler may continue processing the code to identify and report multiple errors in a single pass.

Error Listing : An error listing is a report generated by the assembler that lists all detected errors, their line numbers and descriptions.

Graceful Termination : In case of severe or fatal errors, the assembler should terminate gracefully to avoid generating incomplete or incorrect machine code.