CSN – 252

System Software

SIC-XE ASSEMBLER

Name: Sarvesh Baheti

Enrollment Number: 22114087

Course: BTech – CSE (UG – II year)

Batch: O4

Contents:

|  |  |
| --- | --- |
| Topic | Page No |
| Introduction | 2 |
| Steps to run the assembler | 3 |
| Working of the assembler | 4 |
| Shortcomings | 5 |
| Conclusion and images | 5 |

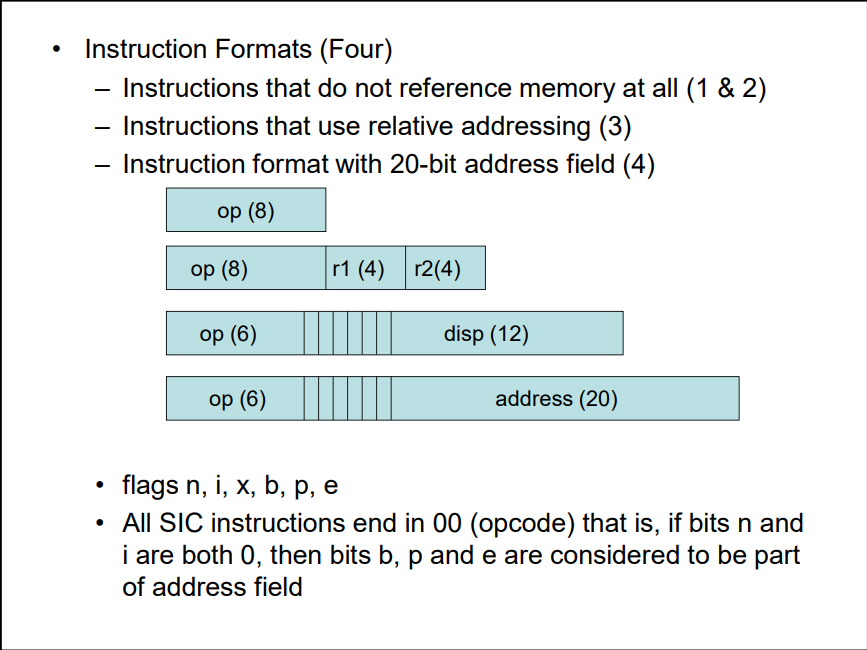
* **Introduction**

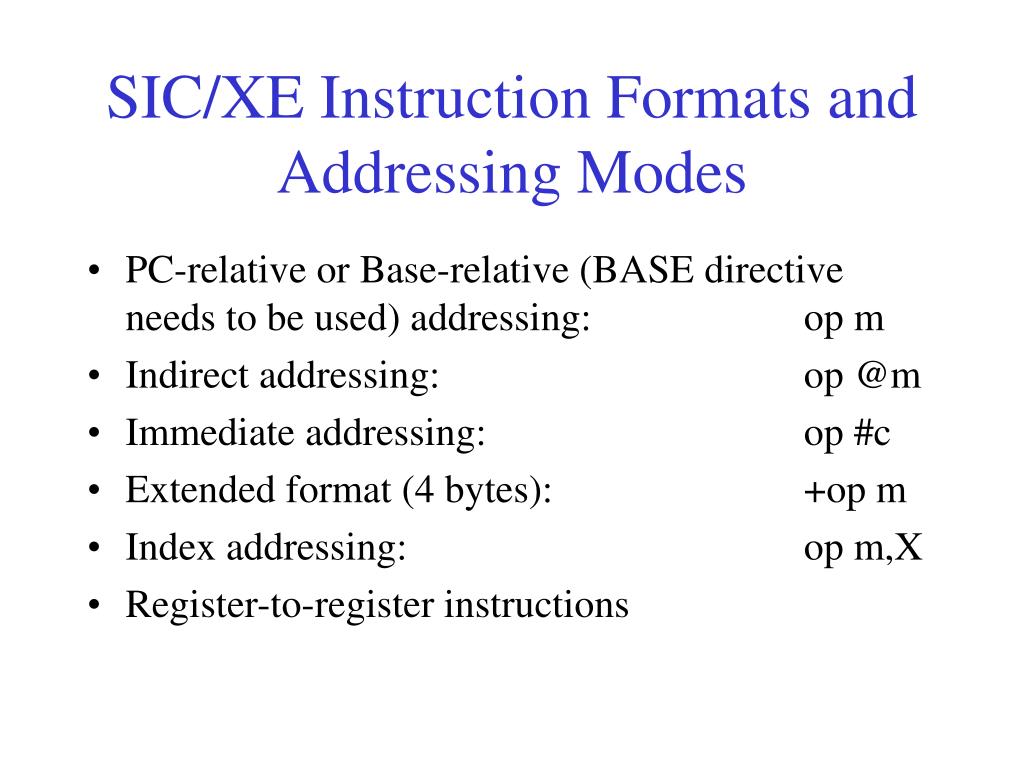
The project implements a SIC-XE assembler, analyzing input assembly language code and generating corresponding object program consisting of header, text, modification, end, define and refer records. This is done by a C++ source file, which simulates the functioning of an assembler by making two passes of the input SIC-XE program.

Following is an overview of different records used in SIC-XE:

1. Header Record: consists of name of the program, starting address and length of the object program.
2. Text Records: they contain assembly of various instructions in assembly code.
3. Modification records: they contain instructions to be modified if the program is relocated. This enables the program to be relocated.
4. Define records: contain names and relative addresses of external symbol defined within that control section.
5. Refer records: contain names of external symbol referred to in this control section.

Instruction formats supported by SIC-XE are:



The Addressing modes supported by the Assembler are: 

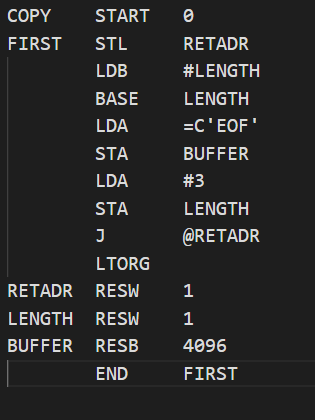
Control Sections in SIC-XE:

Control Sections are parts of the main program that maintain their identity, and are used to implement subroutines. This project can assemble programs consisting of multiple control sections and generate object programs corresponding to all control sections.

* **Steps to run the assembler**

1. Compile assembler.cpp and run the .exe file.
2. Paste the assembly program in input.txt file. The assembly code should be in standard format, with indentation as shown on next page.
3. The program produces:
   1. output.txt: contains the object program generated.
   2. errors.txt: contains the errors detected in pass 1 and pass 2 respectively.
   3. Listing\_File.txt: listing file produced.

Format of input:



* **Working of the assembler**

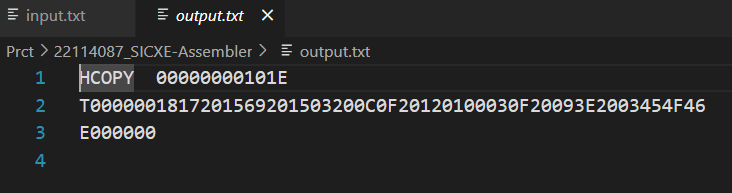
In Pass 1, addresses are assigned to all statements within the program, and the addresses allocated to all labels are stored. Additionally, certain assembler directives are handled during this phase.

Pass 2 assembles instructions by generating object code using tables created in pass1. The data values defined by BYTE, WORD, etc. are assigned values. The assembler directives that were not processed during pass1 are processed. The object program and the assembly listing are written into output.txt and Listing\_File.txt. The assembler generates modification records corresponding to any format 4 instructions and instructions involving external references.

* **Shortcomings**The assembler does not support program control blocks as of now.
* **Conclusion and Images**

This SIC/XE assembler supports control sections. It has been tested against some inputs attached with the project folder

Sample outputs:

 A screenshot of a computer

Description automatically generated