Tanmay Mohit Shrivastav, 21114103
CSN-252

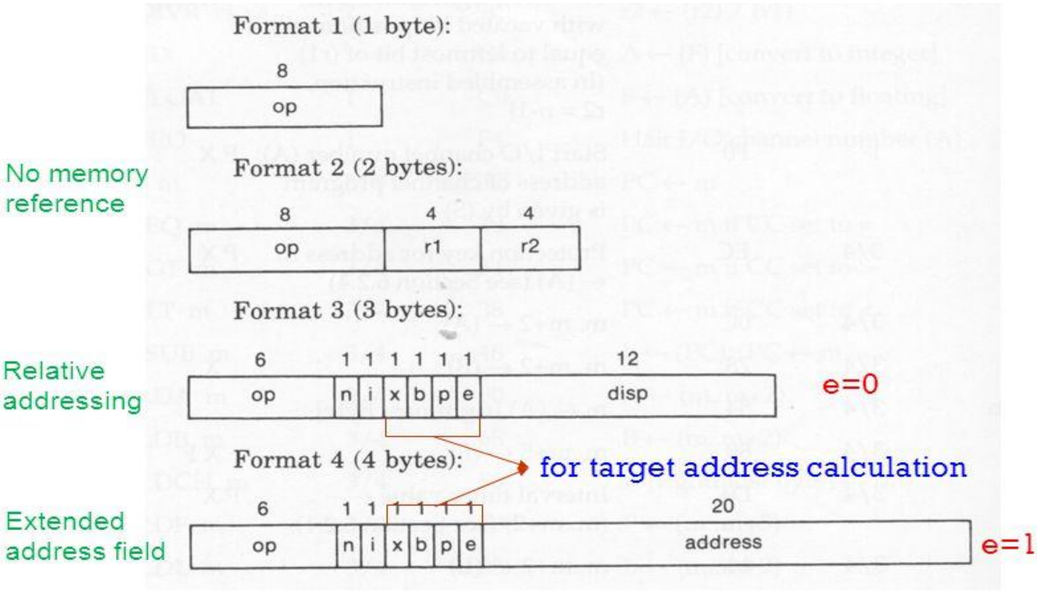
SIC-XE Assembler

C++ implementation

Objective:

The objective of this project is to implement a two-pass assembler for the SIC-XE architecture. The assembler will support all 4 instruction formats and all the various addressing modes. The assembler also includes an implementation for control sections.

The following are the various formats supported by the SIC-XE assembler:



Addressing Modes:

- Base Relative (n=1, i=1, b=1, p=0)
- Program-counter relative (n=1, i=1 b=0, p=1)
- Direct (n=1 ,i=1 ,b=0 ,p=0)
- Immediate (n=0, i=1, x=0)
- Indirect (n=1 ,i=0, x=0)
- Indexing (n=i=0 or 1 & x=1)
- Extended (e=1 format 4, e=0 format 3)

Features Implemented:

1. Literals
2. Symbol Defining Statements
3. Control Sections
4. Expressions

We give input.txt as the input to the assembler. This file contains the machine instructions which the assembler converts into object code.

Execution of the Assembler:

1. Pass1 generates a symbol table and an intermediate file for Pass2.
2. Pass2 generates a listing file containing the input assembly code and address, block number, object code of each instruction.
3. Pass 2 also generates an object program including the following type of record: H, D, R, T, M and E types.
4. An error file is also generated to identify any errors in the assembly program.

Steps required for execution:

- Download the zip file and open the terminal.
- Compile the file pass2.cpp using the command g++ -std=c++11 pass2.cpp.
- Put the executable a.out and the test inputs in the same folder.
- Now change the directory to test_inputs and execute the executable a.out.
- Write the name of the file you want to run in the given space.
- And when we are done, 4 folders will be created corresponding to the error file, intermediate file, listing file and object file.

Conclusion:

I have implemented the assembler in the SIC/XE architecture using C++. This assembler converts the SIC/XE instructions into machine understandable object code along with identification of any errors that might be present in the program