

Software Project Report For SIC/XE Assembler

[Phase 1]

Prepared by:

Omar Nasr	4730
Sherif Rafik	4635
Mohamad Tarek	4774
Mahmoud Salem	4643
Mohamed Harraz	4608

Introduction

Overview

This document defines the requirement for the SIC/XE Assembler that is being developed for Systems Programming course at Faculty of Engineering, Alexandria University. The purpose of this document is to represent the system requirements for operating the software as well as an overlook on the designing and implementation processes.

SIC/XE Assembler is a Graphical User Interface (GUI) based application for converting code written in assembly language into the object code used by the famous hypothetical microprocessor "SIC/XE".

Requirements:

It is required to process the code written in assembly language to:

- Handle all of the SIC/XE supported instruction
- Process a code that meets the SIC/XE specifications
- Check for possible errors
- Generate a list file to review instructions, addresses and errors
- Generate the object code to be passed to the microprocessor

Phase I of the project covers only generating the list file for the assembled code which is reviewed by the programmer to find and correct errors before generating the object code that will be passed to the microprocessor.

Requirements Specifications

This software uses Java Development Kit which needs some system requirements to be met.

Supported Operating Systems

Windows

- 10 or above
- 8.x
- 7 SP1
- Vista SP2
- Server 2008 R2 SP1 (64-bit)
- Server 2012 and 2012 R2 (64-bit)

MAC OS X

- 10.8.3 +
- 10.9 +

Linux

- Oracle Linux 5.5+
- Oracle Linux 6.x (32-bit), 6.x (64-bit)
- Oracle Linux 7.x (64-bit)2 (8u20 and above)

- Red Hat Enterprise Linux 5.5+, 6.x (32-bit), 6.x (64-bit)
- Red Hat Enterprise Linux 7.x (64-bit) (8u20 and above)
- Suse Linux Enterprise Server 10 SP2+, 11.x
- Suse Linux Enterprise Server 12.x (64-bit) (8u31 and above)
- Ubuntu Linux 12.04 LTS, 13.x
- Ubuntu Linux 14.x (8u25 and above)
- Ubuntu Linux 15.04 (8u45 and above)
- Ubuntu Linux 15.10 (8u65 and above)

Environment

The software requires installation of

- Java Development Kit JDK 1.8 (or above)
- Java SE Runtime Environment JRE
- Java Virtual Machine JVM

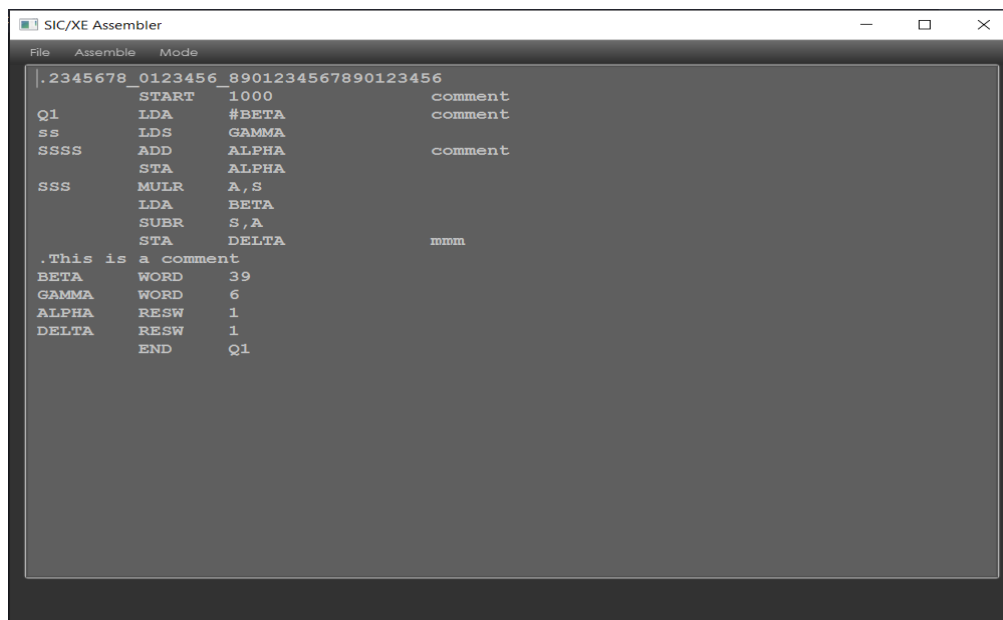
Design Aspects

Overall Design

The project follows an **MVC (model view controller)** design making it reusable and easily editable with an organized hierarchy. The main functionalities of the program are controlled by implementing the **Singleton** Design Pattern to make sure everything is globally handled.

Graphical User Interface

The program provides a smooth, user friendly interface to load written assembly programs from the file system or edit new ones in the editor in order to process them to generate the required output files needed to run programs by the microprocessor



```

SIC/XE Assembler
File  Assemble  Mode
.2345678_0123456_8901234567890123456
      START      1000          comment
Q1    LDA        #BETA         comment
ss    LDS        GAMMA
SSSS  ADD        ALPHA         comment
      STA        ALPHA
SSS   MULR       A,S
      LDA        BETA
      SUBR       S,A
      STA        DELTA         mmm
.This is a comment
BETA  WORD       39
GAMMA WORD       6
ALPHA RESW       1
DELTA RESW       1
      END        Q1
  
```

Data Structures

The mainly used data structures are **ArrayLists** as the are perfectly suitable to the program, because:

- They provide an efficient access time especially with the relatively small amount of data to manipulate
- They are dynamic which allows the user to add more valid instructions when needed

Algorithm

The main functionality of the program is processing a given code to get specific outputs. This is performed in sequential steps:

- The user specifies the input data (either by loading an existing file or by writing the program in the editor)
- The program gets the editing mode selected by the user to proceed with the processing stage
- Input data is processed line by line to check its validity as follows
 1. Check if any of the line components is misplaced
 2. Check that the label is verified
 3. verify the mnemonic
 4. Identify the addressing mode
 5. Verify the operand(s)
- finally, the output files are generated

Assumptions & User Requirements

SIC/XE specifications are met as described below and knowing them is a must for the user to be able to use the program efficiently

Registers

symbol	number	use
<i>A</i>	0	Accumulator: for arithmetic and I/O
<i>X</i>	1	Indexing
<i>L</i>	2	subroutine linkage to save the return address
<i>B</i>	3	base register
<i>S</i>	4	General working register
<i>T</i>	5	General working register
<i>F</i>	6	floating-point accumulator
<i>PC</i>	8	program counter
<i>SW</i>	9	status word

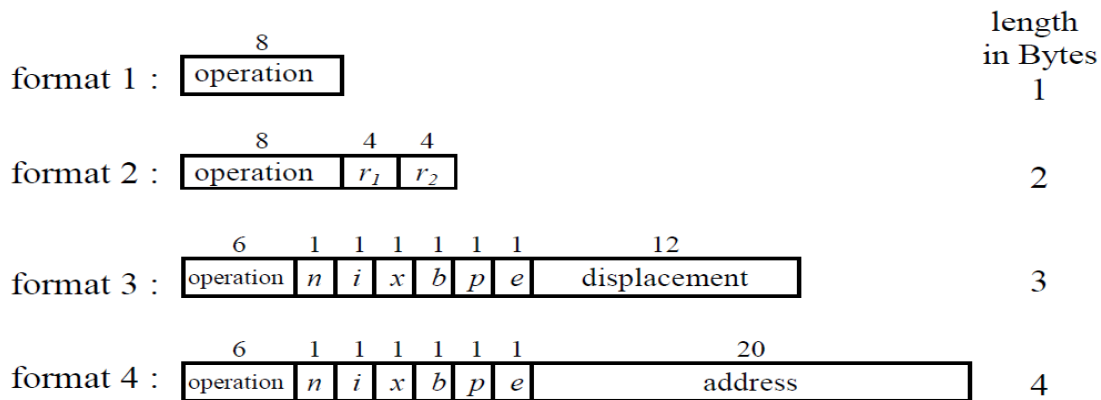
Memory:

- Consists of 8-bit bytes
- Any three consecutive bytes from a word (24-bit)
- Words are addressed by the location of the lowest numbered byte
- Memory size is 220 bytes on the SIC / XE system (1 MB)

Data Format

- Integer are stored in 24-bits as binary numbers using two's complement notation.
- Characters are stored using 8-bits ASCII code

Instruction Format



Addressing Modes

- immediate addressing
- direct addressing with/without indexing
- indirect addressing
- Base relative addressing
- Program counter relative addressing

THE USER IS REQUIRED TO:

- have good experience with assembly language and SIC/XE specifications
- be familiar with the Instruction Set as well as the required format
- provide an assembly program as a text file (file.txt) or as a written code in the editor

Sample runs

Assembling using the restricted format mode

```
File Assemble Mode
.2345678_0123456_8901234567890123456
START 1000 comment
Q1 LDA #BETA comment
ss LDS GAMMA
SSSS ADD TEST comment
STA ALPHA
SSS MULR A,S
LDA BETA
SUBR S,A
STA DELTA mmm
.This is a comment
BETA WORD 39
GAMMA WORD 6
ALPHA RESW 1
DELTA RESW 1
END Q1

-----
S T A R T O F P A S S I
-----
LINES ADDRESS LABEL MNEMONIC ADDR_MODE OPERAND1 OPERAND2 COMMENTS
0 .2345678_0123456_8901234567890123456
1 1000 START 1000 comment
2 1000 Q1 L A # BETA comment
ERROR: Missing or misplaced operation mnemonic
3 1000 ss LDS GAMMA
4 1003 SSSS ADD TEST comment
ERROR: Wrong operand type
5 1006 STA ALPHA
6 1009 SSS MULR A S
7 100B LDA BETA
8 100E SUBR S A
9 1010 STA DELTA mmm
10 .This is a comment
11 1013 BETA WORD 39
12 1016 GAMMA WORD 6
13 1019 ALPHA RESW 1
14 101C DELTA RESW 1
15 101F END Q1
```

Assembling using the free format mode

```
File Assemble Mode
.2345678_0123456_8901234567890123456
Q1 LDA #BETA ;comment
ss LDS GAMMA ;comment
SSSS ADD ALPHA ;comment
STA ALPHA
SSS MULR A,S
LDA BETA
SUBR S,A
STA DELTA ;mmm
.This is a comment
BETA WORD 39
GAMMA WORD 6
ALPHA RESW 1
DELTA RESW 1
END Q1

-----
S T A R T O F P A S S I
-----
LINES ADDRESS LABEL MNEMONIC ADDR_MODE OPERAND1 OPERAND2 COMMENTS
0 .2345678_0123456_8901234567890123456
1 1000 START 1000 comment
2 1000 Q1 LDA # BETA comment
3 1003 ss LDS GAMMA
4 1006 SSSS ADD ALPHA comment
5 1009 STA ALPHA
6 100C SSS MULR A S
7 100E LDA BETA
8 1011 SUBR S A
9 1013 STA DELTA mmm
10 .This is a comment
11 1016 BETA WORD 39
12 1019 GAMMA WORD 6
13 101C ALPHA RESW 1
14 101F DELTA RESW 1
15 1022 END Q1

inline comments should be preceded with ;
```