ACKNOWLEDGEMENTS

We would like to acknowledge R.V.C.E. for providing us the opportunity to work on this mini-project.

We also express our heartfelt thanks to Dr. Shobha G, Head of Department, Computer Science & Engineering, for providing us with the required facilities and information which helped us in the completion of our project.

We would personally like to thank **Prof. Suma B** and **Prof. Srividya M S** for providing timely assistance & guidance.

We would also like to acknowledge the co-operation shown by the lab administrators and lab assistants towards the completion of our project.

**MEDA SAI KHEERTHANA MEGGHA S**

1RV13CS080 1RV13CS081

**ABSTRACT**

Assemblers are the simplest of a class of systems programs called translators. A translator is simply a program, which translates from one (computer) language to another (computer) language. In the case of an assembler, the translation is from assembly language to object language (which is input to a loader). Notice that an assembler, like all translators, adds nothing to the program which it translates, but merely changes the program from one form to another. The use of a translator allows a program to be written in a form that is convenient for the programmer, and then the program is translated into a form convenient for the computer. Assembly language is almost the same as machine language. The major difference is that assembly language allows the declaration and use of symbols to stand for the numeric values to be used for opcodes, fields, and addresses. An assembler inputs a program written in assembly language, and translates all of the symbols in the input into numeric values, creating an output object module, suitable for loading. The object module is output to a storage device, which allows the assembled program to be read back into the computer by the loader. This report presents the implementation of a one-pass assembler for 8086 using lex and yaac.

**TABLE OF CONTENTS**

**Topics Page Nos.**

**1. INTRODUCTION**…………………………………………………………….11.1 Overview of One pass assembler for 8086……..……………………………...1

1.2 Scope…………………………………………………………………………..2

1.3 Methodology……………………………………………...................................3

**2. SYSTEM REQUIREMENTS**……………………………………………...42.1 Functional Requirement ………………………………………………………5

2.2 Non Functional Requirement ………………………………………………....6

**3. SYSTEM DESIGN** …………..………….. ………………………………….7

3.1 DFD ………………………………………………………………………….7

3.2 Algorithms……………………………………………………………………8

3.3 Data structures………………………………………………………………..8

**4. IMPEMENATION**………………………………………………..………….9 4.1 Source Code…………………………………………………………………...9

4.2 Result Discussion …………………………………………………………….23

**5. CONCLUSION**

5.1 Limitation…………………………………………………………………….24

5.2 Scope for Further Enhancements……………………………….....................24

**6. REFERENCE**………………………………………………………………...25

**7. APPENDIX- A (Snapshots)** ………………………………………………26