



# Compiler Design

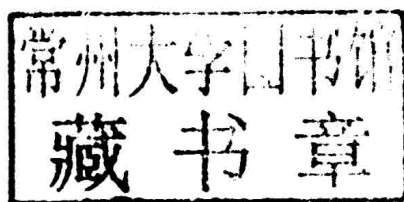
H.S. Mohan



Alpha  
Science

# COMPILER DESIGN

H. S. Mohan



Alpha Science International Ltd.  
Oxford, U.K.

## **Compiler Design**

232 pgs.

### **H. S. Mohan**

Professor and Head

Department of Information Science and Engineering

SJB Institute of Technology

Kengeri, Bangalore

Copyright © 2014

---

ALPHA SCIENCE INTERNATIONAL LTD.

7200 The Quorum, Oxford Business Park North

Garsington Road, Oxford OX4 2JZ, U.K.

**[www.alphasci.com](http://www.alphasci.com)**

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior written permission of the publisher.

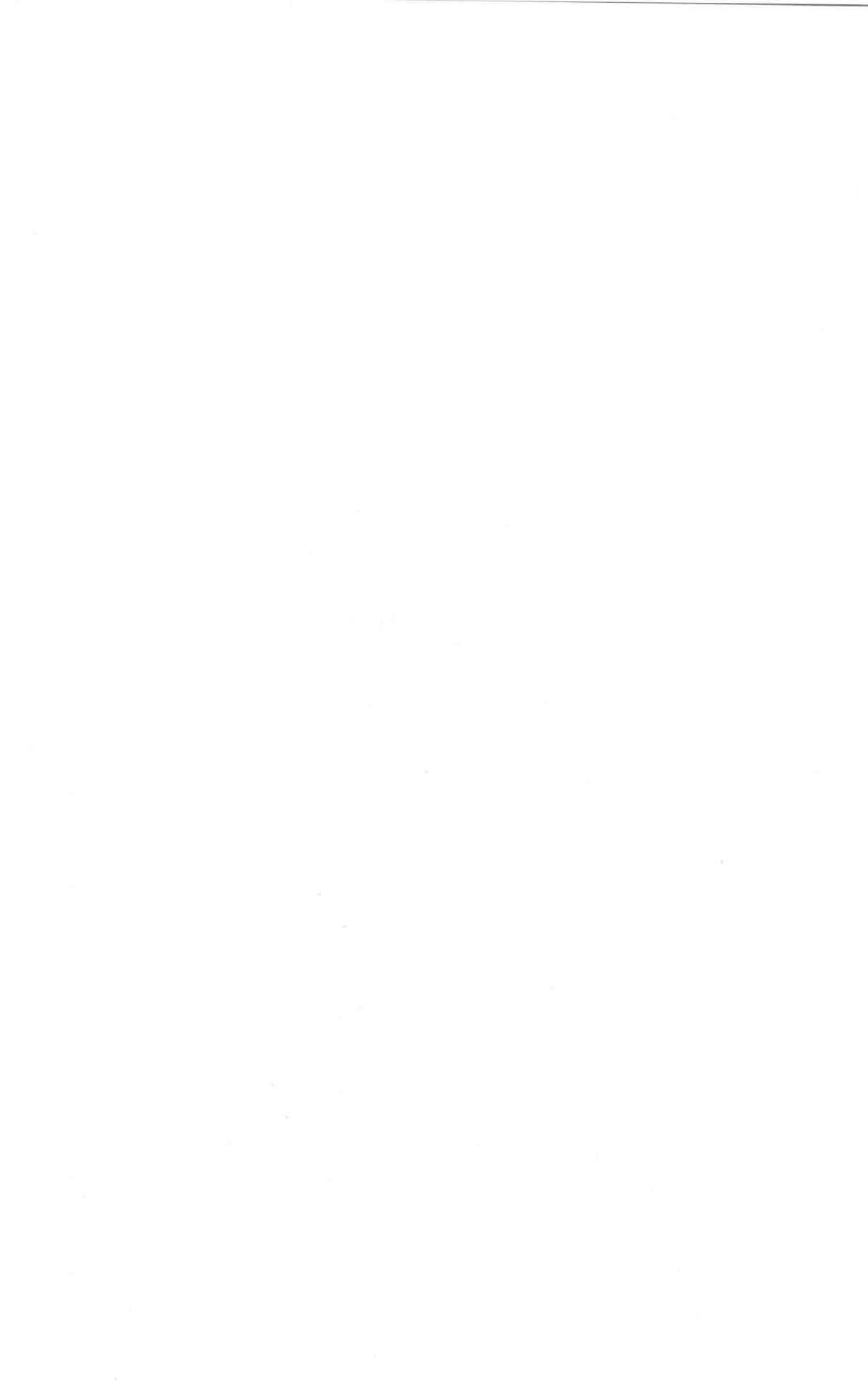
ISBN 978-1-84265-857-4

Printed in India

# COMPILER DESIGN



Dedicated  
to  
My Beloved Mother  
Yashodha





## PREFACE

Computer understands only Machine code. So it is necessary to have a intermediate to convert the source language into the machine code. So the design of compilers is very important in the engineering fields.

Overwhelming response from my students and other teachers of various engineering colleges who have referred my notes inspired me to write this book.

The book is written as a text, with problems and exercises. In every chapter of this book, more importance is given to the concepts and many problems are solved which covers all varieties of problems in a simpler and easier techniques.

The book covers the syllabus of Undergraduate and Postgraduate students of CSE and ISE branches of almost all universities.

Any suggestion for the improvement of the book will be acknowledged and well appreciated. Suggestions can be emailed to [mohan\\_kit@yahoo.com](mailto:mohan_kit@yahoo.com)

**H.S. Mohan**







## ACKNOWLEDGEMENTS

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible because “Success is the abstract of hard work and perseverance, but steadfast of all is encouragement guidance”. So I would like to acknowledge all those whose guidance and encouragement served as a beacon light and crowned my efforts with success.

I would like to express my pranam's to his Divine soul Padmabhushana Sri Sri Sri Dr. Balagangadharanatha Maha Swamiji and pranam's to his Holiness Sri Sri Sri Nirmalanandanatha Maha Swamiji, the president of Sri Adichunchanagiri shikshana trust ® for their Blessings.

I would like to express my profound grateful to Reverend Sri Sri Prakashnath Swamiji, Managing Director, SJBIT, Bangalore for his blessings.

I am grateful to Dr. Puttaraju, Principal for his kind co-operation and encouragement.

I am grateful to Dr. V. Rajappa, Founder, Director Prajwal Academy of Technical Educations and Neelakanta V. for their moral support and encouragement given to me for writing this book.

I would like to render my heartfelt gratitude to my parents, my wife Bindiya M.K. and My daughter Aditi Mohan and my son Adithya Mohan for their kind cooperation, valuable moral support and encouragement given to me and the role they have played in completing my book.

I am grateful to all my friends, colleagues and others who are directly or indirectly involved for their inspiration, encouragement and support to successfully complete this book.

Finally I convey my thanks to publisher of this book, for their constructive criticism and suggestions for improvement in the manuscript.

**H. S. Mohan**





# CONTENTS

<i>Preface</i> .....	<i>vii</i>
<i>Acknowledgements</i> .....	<i>ix</i>
<b>1. Introduction to Compilers</b> .....	<b>1.1—1.10</b>
1.1 Compilers .....	1.1
1.2 Other Applications .....	1.1
1.3 A Language Processing System .....	1.2
1.4 The Structure of a Compiler or Phases of a Compiler .....	1.3
1.5 Compiler Construction Tools .....	1.8
1.6 The Evolution of Programming Languages .....	1.9
1.7 Applications of Compiler Technology .....	1.10
<b>2. Lexical Analysis</b> .....	<b>2.1—2.10</b>
2.1 Role of the Lexical Analyzer .....	2.1
2.2 Tokens, Patterns and Lexemes .....	2.2
2.3 Input Buffering .....	2.3
2.4 Specification of Tokens .....	2.4
2.5 Regular Definitions .....	2.5
<b>3. Syntax Analysis—I</b> .....	<b>3.1—3.56</b>
3.1 Introduction .....	3.1
3.2 Context-Free Grammar: (CFG) .....	3.3

3.3	Leftmost and Rightmost Derivations .....	3.7
3.4	Context Free Grammar Versus Regular Expressions .....	3.15
3.5	Left Recursion.....	3.19
3.6	Left Factoring.....	3.25
3.7	Parsing Techniques .....	3.29
3.8	Error Recovery in Predictive Parsing .....	3.47
<b>4.</b>	<b>Syntax Analysis–II.....</b>	<b>4.1—4.30</b>
4.1	Bottom–up–Parsing.....	4.1
4.2	Handles .....	4.4
4.3	Shift Reduce Parser.....	4.6
4.4	Conflicts During Shift-Reduce Parsing.....	4.9
4.5	Introduction to LR Parsing.....	4.10
4.6	Viable Prefixes .....	4.29
<b>5.</b>	<b>Syntax Analysis–III .....</b>	<b>5.1—5.24</b>
5.1	More Powerful LR Parsers.....	5.1
5.2	LALR Parser (Lookahead LR Parser).....	5.8
5.3	Using Ambiguous Grammar to (Reduce) Resolve Shift Reduce Conflict.....	5.19
<b>6.</b>	<b>Syntax Directed Translation .....</b>	<b>6.1—6.26</b>
6.1	Syntax Directed Definition (SDD).....	6.1
6.2	Evaluation Orders for SDDs .....	6.11
6.3	Applications of Syntax Directed Translation.....	6.16
<b>7.</b>	<b>Intermediate—Code Generation .....</b>	<b>7.1—7.36</b>
7.1	Introduction.....	7.1
7.2	Directed Acyclic Graphs for Expressions (DAG) .....	7.2
7.3	Implements of Three-Address Codes.....	7.6
7.4	Type Checking .....	7.10
7.5	Unification.....	7.12
7.6	Syntax Directed Translation.....	7.13
7.7	Evaluation Orders for SDDs .....	7.23
7.8	Applications of Syntax Directed Translation.....	7.28
7.9	Syntax-Directed Translation Schemes (SDT's) .....	7.33
7.10	Eliminating Left Recursion from SDT's.....	7.36
<b>8.</b>	<b>Run-Time Environments .....</b>	<b>8.1—8.15</b>
8.1	Storage Organization.....	8.1

8.2	Stack Allocation of Space .....	8.3
8.3	Access to Nonlocal Data on the Stack .....	8.7
8.4	Heap Management .....	8.9
8.5	Introduction to Garbage Collection .....	8.12
<i>Appendix</i> .....		<i>A.1—A.7</i>
<i>Index</i> .....		<i>I.1—I.2</i>



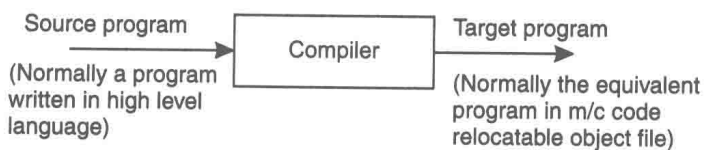
## CHAPTER

# 1

## INTRODUCTION TO COMPILERS

### 1.1 COMPILERS

A Compiler is a program, takes a program written in a source language and translates it into equivalent program in a target language.



### 1.2 OTHER APPLICATIONS

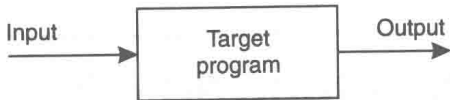
In addition to the development of Compiler, the techniques used in compiler design can be applicable to many problems in Computer Science.

1. Techniques used in Lexical Analyzer can be used in Text editors, information retrieval system and Pattern Recognition Programs.
2. Techniques used in a parser can be used in a query processing system such as SQL.
3. Many software having a complex front end may need techniques used in Compiler Design.

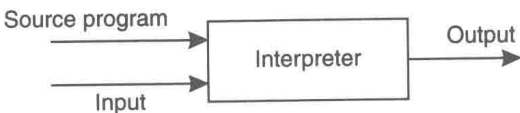


- 4. Most of the techniques used in Compiler Design may be used in Natural Language Processing (NLP) Systems.

If the *target program* is an executable Machine language program. It can then be called by the user to process inputs and process outputs.



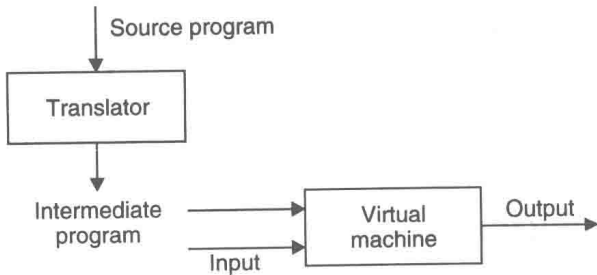
An **interpreter** is another common kind of Language Processor. Instead of producing a target program as a transition an interpreters appear to directly execute the operations specified in the source. Program on inputs supplied by the user.



**Example**

A java language Processor combine compilation and interpretation.

A Java Source Program may first be compiled into an intermediate form called *Byte Code*. The Byte codes are then interpreted by a virtual machine.



**Figure:** A hybrid compiler

**1.3 A LANGUAGE PROCESSING SYSTEM**

In addition to a compiler, several other Programs may be required to create an executable Target Program as shown in figure.

- A source program may be divided into modules stored in separate files. The task of collecting the source program is sometimes entrusted to a separate program called a *preprocessor*. The preprocessor may also expand shorthands called *Macros*.