

INFORMAL VERIFICATION OF CORRECTNESS OF THE  
SCANNER MODULE OF AN INTERPRETER PROGRAM

by

JAMES NOEL JONES

B. S., Oklahoma State University, 1965

---

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1975

Approved by:

  
Major Professor

TABLE OF CONTENTS

Chapter 1	General	6
1.1	Purpose of the Report	6
1.2	General Background of the Interpreter Program	6
Chapter 2	Specifications and Data Formats	8
2.1	General	8
2.1.1	HEAP Storage Area Specifications	8
2.1.2	Global Assertions	12
2.2	Scanner Specifications	15
2.2.1	General	15
2.2.2	Calls to the Scanner Module	17
	(2.1) Input Assertions	17
	(2.2) Output Assertions	19
2.2.3	Local Data Structures	19
	(3.1) Character Table	19
	(3.2) Keyword Table and Command Language Table	22
	(3.3) Line Stack	22
	(3.4) Symbol Stack	22
	(3.5) Error Stack	22
	(3.6) Symbol	25
	(3.7) RVALUE	25
2.2.4	Required Calls	25
2.2.5	Terminal Communication to User	25

		3
2.2.6	Language Restrictions	25
2.3	Global Procedures	29
2.3.1	General	29
2.3.2	Routine Specifications and Assertions	29
	(2.1) Routine GET	29
	(2.2) Routine EXPAND	31
	(2.3) Routine SYMTAB	32
	(2.4) Routine ERRPRT	34
	(2.5) Routine STAX	36
	(2.6) Routine GETCHR	37
	(2.7) Routine PUTCHR	38
Chapter 3	Assertion Refinement and High Level Design	40
3.1	General	40
	3.1.1 Assertion Refinement	40
	3.1.2 Notation	40
3.2	Global Assertion Refinement	41
3.3	Scanner Assertions	44
	3.3.1 Input Assertions	44
	3.3.2 Output Assertions	47
	3.3.3 Table, Counter, and Flag Initialization and Variable Definition	50
3.4	Global Procedure Assertion Refinement	56
3.5	High Level Design Language with Assertions	60
	3.5.1 SUBROUTINE SCAN	61
	3.5.2 SUBROUTINE LNSCAN	67
	3.5.3 SUBROUTINE FORM	72

	4
3.5.4 SUBROUTINE TABLE	77
3.5.5 SUBROUTINE NUMPAC	79
3.5.6 SUBROUTINE SERROR	81
3.5.7 SUBROUTINE LINFIN	83
Chapter 4 Module Verification	86
4.1 General	86
4.2 Subroutine SCAN	86
4.3 Subroutine LNSCAN	92
4.4 Subroutine FORM	96
4.5 Subroutine TABLE	100
4.6 Subroutine NUMPAC	102
4.7 Subroutine SERROR	103
4.8 Subroutine LINFIN	105
Chapter 5 Conclusions	109
5.1 General	109
5.2 Verification	110
5.3 Application	112
5.4 Recommendations	112
Annex A - References	114
Annex B - Initialization Data for Data Structures	115
Annex C - FORTRAN CODE with Assertions	125

## LIST OF FIGURES

FIGURE		PAGE
2-1	Interpreter Program Hierarchical Chart	9
2-2	HEAP Storage Area Contents	10
2-3	Format for Address Translation Table	11
2-4	Format for Procedure Table Block	13
2-5	Header Format for an Allocated Block in HEAP	14
2-6	Format for Text Block	18
2-7	Format for Token Block	20
2-8	Character Table Format	21
2-9	Keyword Table Format	23
2-10	Command Language Table Format	24
2-11	SCANNER Module Hierarchical Chart	26
4-1	Subroutine SCAN State Transition Diagram	87
4-2	Subroutine LNSCAN State Transition Diagram	94
4-3	Subroutine FORM State Transition Diagram	97
4-4	Subroutine TABLE State Transition Diagram	101
4-5	Subroutine NUMPAC State Transition Diagram	104
4-6	Subroutine SERROR State Transition Diagram	106
4-7	Subroutine LINFIN State Transition Diagram	108
C1-1	Variable Translation Table	126

## Chapter 1 General

### 1.1 Purpose of the Report

The purpose of this report is to provide a documented, informal verification of correctness of the Scanner module of an interpreter program. The verification follows three levels of program development: (1) that of English specifications and assertions, (2) high level design language, and (3) FORTRAN code. Assertions are developed from the Scanner module's specifications and are refined in parallel with program development. Verification of correctness is presented for the program based on the refined assertions. The scope of the verification will include the Scanner subroutine, all subroutines developed directly from the Scanner module, and all external routines used by the Scanner.

### 1.2 General Background of the Interpreter Program

The interpreter program of which the Scanner module is a portion was designed and implemented by the students of CS 286-700, summer 1975 session. The program was divided into the eleven following student groups:

- (1) Interpreter Driver
- (2) Scanner
- (3) Text Editor
- (4) Top Down Parser
- (5) Bottom Up Parser
- (6) Command Line Interpreter
- (7) Operation Functions
- (8) Heap Maintenance
- (9) Stack Functions