

# Forth user's manual

## AIMI 65

advanced interactive microcomputer



Rockwell  
International

...where science gets down to business

# **Forth User's manual**

**ATM 65**  
advanced interactive microcomputer



...where science gets down to business

©Rockwell International Corporation 1981  
All Rights Reserved  
Printed in the U.S.A.

Document No. 2965ON72  
June 1981

TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Title</u>	<u>Page</u>
4.13	Input Words	
4.13.1	Input a Character from the Keyboard with KEY .....	4-60
4.13.2	Input a String from the Keyboard with EXPECT .....	4-62
4.13.3	Set the Active Input Device with ?IN .....	4-63
4.13.4	Input a Character from the Active Input Device with GET .....	4-63
4.13.5	Input a String from the Active Input Device with READ .....	4-64
4.13.6	Test for Terminal Input with ?TERMINAL .....	4-64
5	Advanced Operations	
5.1	Other Single-Precision Arithmetic Operations .....	5-1
5.1.1	Modulus Operators MOD and /MOD .....	5-1
5.1.2	Absolute ABS and Negate NEGATE .....	5-1
5.1.3	Increment and Decrement 1+ , 2+ , 1- , 2- ..	5-2
5.1.4	Minimum MIN and Maximum MAX .....	5-2
5.2	Unsigned, Mixed and Double-Precision Arithmetic .....	5-3
5.2.1	Entering Double-Precision Numbers .....	5-3
5.2.2	Printing Double-Precision Numbers .....	5-4
5.2.3	Other 32-Bit FORTH Operators .....	5-6
5.2.4	Unsigned Compare <U> .....	5-7
5.2.5	Unsigned Multiply U* and Divide U/ .....	5-7
5.2.6	Mixed Mode Operations M* M/ and M/MOD ..	5-8
5.2.7	Scaling .....	5-9
5.3	Output Formatting .....	5-10
5.3.1	S->D , <# , #S , SIGN , and #> .....	5-10
5.3.2	# and HOLD .....	5-11
5.4	Strings .....	5-12
5.4.1	Address String Data with COUNT .....	5-13
5.4.2	Output String Data with TYPE .....	5-13
5.4.3	Input String Data with EXPECT .....	5-14
5.4.4	Suppress Trailing Blanks with -TRAILING .....	5-14
5.4.5	Interpret a Number with (NUMBER) .....	5-15
5.4.6	Input a Number with NUMBER .....	5-15
5.5	Dictionary Structure .....	5-15
5.5.1	FORTH Word Structure .....	5-16
5.5.2	Handling FORTH Word Addresses .....	5-19
5.5.3	FORTH Word Handling Examples .....	5-19

TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Title</u>	<u>Page</u>
5.6	Vocabularies .....	5-20
5.6.1	More on VLIST .....	5-20
5.6.2	CONTEXT and CURRENT Specify Vocabularies ..	5-21
5.6.3	Use LATEST and HERE to Check Directory ..	5-22
5.6.4	Application Libraries .....	5-23
5.7	Immediate Words .....	5-25
5.8	Creating Your Own Data/Operation Types .....	5-26
6	AIM 65 FORTH ASSEMBLER	
6.1	The Assembly Process .....	6-1
6.1.1	CODE Definitions .....	6-4
6.1.2	Assembly-Time Versus Run-Time .....	6-4
6.1.3	CODE-Definition Example .....	6-5
6.2	Assembler Op-codes .....	6-7
6.2.1	Single Mode Op-Codes .....	6-7
6.2.2	Multi-Mode Op-Codes .....	6-7
6.3	Addressing Modes .....	6-8
6.4	R6502 Conventions .....	6-9
6.4.1	Stack Addressing .....	6-9
6.4.2	Return Stack .....	6-10
6.5	FORTH Registers .....	6-11
6.5.1	Assembly Registers .....	6-11
6.5.2	CPU Registers .....	6-12
6.5.3	XSAVE .....	6-12
6.5.4	N Area .....	6-13
6.5.5	SETUP .....	6-14
6.6	Control Flow .....	6-14
6.6.1	Conditional Looping .....	6-15
6.6.2	Conditional Execution .....	6-17
6.6.3	Conditional Nesting .....	6-18
6.6.4	Some Nesting Examples .....	6-19
6.7	Return of Control .....	6-21
6.8	Assembler Security .....	6-23
6.8.1	Assembler Tests .....	6-23
6.8.2	Bypassing Security .....	6-24
6.9	Adding Assembly Code to Colon-Definition .....	6-24

TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Title</u>	<u>Page</u>
7	Handling Interrupts in FORTH	
7.1	Types of Interrupt Handlers .....	7-1
7.2	Machine Level Interrupt Handling .....	7-1
7.2.1	CODE-Definition Form .....	7-4
7.2.2	Code Fragment Form .....	7-5
7.2.3	Interrupt Disable/Enable Words .....	7-5
7.2.4	Example .....	7-6
7.3	Interpretive Interrupt Handling .....	7-6
7.3.1	Interrupt Service Subroutine .....	7-6
7.3.2	Interrupt Processing Word .....	7-7
7.3.3	Example .....	7-8
7.3.4	Points to Remember .....	7-10
8	Programming the R6522 VIA	
8.1	VIA Organization and Registers .....	8-1
8.2	Simple I/O with the VIA .....	8-4
8.2.1	Considerations .....	8-4
8.2.2	Examples .....	8-5
8.3	Recognizing Status Signals .....	8-8
8.3.1	Considerations .....	8-8
8.3.2	Examples .....	8-9
8.4	Producing Output Strokes .....	8-11
8.4.1	Considerations .....	8-11
8.4.2	Options .....	8-12
8.4.3	Examples .....	8-12
8.5	VIA Interrupts .....	8-16
8.5.1	Considerations .....	8-16
8.5.2	Examples .....	8-16
9	TTY and CRT Operation	
9.1	TTY Operation .....	9-1
9.1.1	Switching Control to the TTY .....	9-1
9.1.2	Switching Control to the AIM 65 .....	9-1
9.2	CRT Terminal Operation Using the 20 MA Current Loop Interface .....	9-2
9.2.1	Switching Control to the CRT Terminal .....	9-2
9.2.2	Switching Control to the AIM 65 .....	9-3

TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Title</u>	<u>Page</u>
10	Preparing an Application Program for PROM Installation .....	10-1
10.1	Using AIM 65 Mnemonic Entry .....	10-3
10.2	Using the FORTH Assembler .....	10-12
11	Using an Audio Cassette Recorder	
11.1	Handling Program Source Code Files .....	11-1
11.1.1	Listing Program Source Code .....	11-1
11.1.2	Reading Program Source Code .....	11-3
11.1.3	Compiling Program Source Code .....	11-4
11.2	Handling Program Object Code Files .....	11-5
11.2.1	Dumping Program Object Code .....	11-5
11.2.2	Loading Program Object Code .....	11-6
11.3	Handling Data Files .....	11-6
11.3.1	Using Recorder Remote Control .....	11-10
11.3.2	Using AIM 65 FORTH Format .....	11-10
11.3.3	Using AIM 65 Monitor Format .....	11-12
12	Interfacing to Mass Storage .....	12-1
12.1	Overview .....	12-1
12.1.1	Mass Storage Terminology .....	12-1
12.1.2	Buffer Variables .....	12-4
12.2	Steps to Set up Block and Data Buffers .....	12-4
12.3	Creating Screens .....	12-6
12.3.1	Creating and Testing a One Screen Buffer .....	12-6
12.3.2	Creating and Testing a Two Screen Buffer .....	12-10
12.4	Interface Words .....	12-11
12.5	Using Mass Storage .....	12-14
12.5.1	Data Storage and Retrieval -the Virtual RAM ..	12-15
12.5.2	Program Loading and Overlays .....	12-16
12.6	Source Code Editings .....	12-18
13	Notes on Style and Program Development	
13.1	General .....	13-1
13.2	Example Program .....	13-2

TABLE OF CONTENTS (Continued)

<u>Section</u>	<u>Title</u>	<u>Page</u>
APPENDIX A	AIM 65 FORTH Functional Summary .....	A-1
APPENDIX B	AIM 65 FORTH Glossary .....	B-1
APPENDIX C	AIM 65 FORTH Assembler Functional Summary .....	C-1
APPENDIX D	AIM 65 FORTH Assembler Glossary .....	D-1
APPENDIX E	Error Messages Recovery .....	E-1
APPENDIX F	Page Zero And One Memory Map .....	F-1
APPENDIX G	USER Variables RAM Map .....	G-1
APPENDIX H	ASCII Character Set .....	H-1
APPENDIX I	FORTH String Words .....	I-1
APPENDIX J	USER 24-Hour Clock Program in FORTH .....	J-1
APPENDIX K	Measuring FORTH Word Execution Time .....	K-1
APPENDIX L	AIM 65 FORTH Versus FIG-FORTH .....	L-1
APPENDIX M	AIM 65 FORTH ROM Check-sum Program .....	M-1
APPENDIX N	Selected Bibliography .....	N-1

LIST OF FIGURES

<u>Figure</u>	<u>Title</u>	<u>Page</u>
2-1	AIM 65 FORTH Memory Map .....	2-2
4-1	VLIST of AIM 65 FORTH Words .....	4-3
4-2	Stack Diagram of Postfix Example .....	4-12
6-1	VLIST of AIM 65 FORTH Assembler Words .....	6-2
7-1	Machine Level Interrupt Handling .....	7-2
7-2	Interpretive Interrupt Handling .....	7-3
8-1	R6522 VIA Organization .....	8-2
8-2	R6522 VIA Interrupt Enable Register .....	8-17
8-3	R6522 VIA Interrupt Flag Register .....	8-18
10-3	Application Program Dictionary Linkage .....	10-2
J-1	24-Hour Clock Program Using a Machine Level Interrupt Handler .....	J-4
J-2	VLIST of 24-Clock Program Using a Machine Level Interrupt Handler .....	J-7
J-3	24-Hour Clock Program Using an Interpretive Interrupt Handler .....	J-8
J-4	VLIST of 24-Hour Clock Program Using an Interpretive Interrupt Handler .....	J-11

## SECTION 1

### INTRODUCTION

FORTH is a unique programming system that is well suited to a variety of applications. Because it was originally developed for real-time control applications, FORTH has features that make it ideal for machine and process control, data acquisition, energy and environmental management, automatic testing, and other similar applications. The speed performance of assembly language is required in many of these applications, however a high-level language is often desired to improve program development productivity and program reliability. FORTH is designed to satisfy both speed and programming efficiency requirements.

FORTH can be called a computer language, an operating system, an interactive compiler, a data structure, or an interpreter, depending upon your point of view. It was designed to combine the strengths of both compilers and interpreters. The result is a unique language based on pre-defined operations that minimizes software development time and costs, supports structured programming and program modularity, compiles interactively to ease debugging and to reduce programming errors, compacts into small object code, and executes extremely fast. Additional words may be defined to allow usage by non-programmers.

AIM 65 FORTH in ROM combines the benefits of FORTH and the features of the AIM 65 Microcomputer with its resident printer, display, keyboard, and interactive Monitor and Text Editor firmware, to produce a standalone development and run-time system.

### LIST OF TABLES

<u>Table</u>	<u>Title</u>	<u>Page</u>
8-1	R6522 VIA Memory Assignments .....	8-3
12-1	Buffer Variables and Access Words .....	12-5

