

# Contents

<b>Introduction</b> . . . . .	<b>xiii</b>
New and Extended Features in MASM 6.1 . . . . .	xiii
MASM Features New Since Version 5.1 . . . . .	xiv
MASM Features New Since Version 6.0 . . . . .	xv
ML and MASM Command Lines . . . . .	xvi
Compatibility with Earlier Versions of MASM . . . . .	xvi
A Word About Instruction Timings . . . . .	xvii
Books for Further Reading . . . . .	xviii
Document Conventions . . . . .	xix
Getting Assistance and Reporting Problems . . . . .	xx
<b>Chapter 1 Understanding Global Concepts</b> . . . . .	<b>1</b>
The Processing Environment . . . . .	1
8086-Based Processors . . . . .	2
Operating Systems . . . . .	4
Segmented Architecture . . . . .	5
Segment Protection . . . . .	6
Segmented Addressing . . . . .	7
Segment Arithmetic . . . . .	7
Language Components of MASM . . . . .	8
Reserved Words . . . . .	8
Identifiers . . . . .	9
Predefined Symbols . . . . .	10
Integer Constants and Constant Expressions . . . . .	11
Operators . . . . .	13
Data Types . . . . .	14
Registers . . . . .	16
Statements . . . . .	21
The Assembly Process . . . . .	22
Generating and Running Executable Programs . . . . .	23
Using the OPTION Directive . . . . .	24
Conditional Directives . . . . .	28
<b>Chapter 2 Organizing Segments</b> . . . . .	<b>31</b>
Physical Memory Segments . . . . .	32
Logical Segments . . . . .	32
Using Simplified Segment Directives . . . . .	33

---

Defining Basic Attributes with <code>.MODEL</code> . . . . .	34
Specifying a Processor and Coprocessor . . . . .	38
Creating a Stack. . . . .	38
Creating Data Segments . . . . .	39
Creating Code Segments. . . . .	40
Starting and Ending Code with <code>.STARTUP</code> and <code>.EXIT</code> . . . . .	41
Using Full Segment Definitions . . . . .	44
Defining Segments with the <code>SEGMENT</code> Directive. . . . .	44
Controlling the Segment Order . . . . .	47
Setting the <code>ASSUME</code> Directive for Segment Registers. . . . .	49
Defining Segment Groups . . . . .	51
<b>Chapter 3 Using Addresses and Pointers . . . . .</b>	<b>53</b>
Programming Segmented Addresses . . . . .	53
Initializing Default Segment Registers. . . . .	53
Near and Far Addresses . . . . .	57
Operands . . . . .	60
Register Operands . . . . .	61
Immediate Operands . . . . .	61
Direct Memory Operands . . . . .	62
Indirect Memory Operands. . . . .	64
The Program Stack. . . . .	71
Saving Operands on the Stack. . . . .	71
Saving Flags on the Stack . . . . .	73
Saving Registers on the Stack (80186–80486 Only). . . . .	74
Accessing Data with Pointers and Addresses . . . . .	74
Defining Pointer Types with <code>TYPDEF</code> . . . . .	75
Defining Register Types with <code>ASSUME</code> . . . . .	77
Basic Pointer and Address Operations . . . . .	78
<b>Chapter 4 Defining and Using Simple Data Types . . . . .</b>	<b>85</b>
Declaring Integer Variables . . . . .	85
Allocating Memory for Integer Variables . . . . .	85
Data Initialization . . . . .	87
Working with Simple Variables . . . . .	88
Copying Data. . . . .	89
Adding and Subtracting Integers . . . . .	92
Multiplying and Dividing Integers. . . . .	95
Manipulating Numbers at the Bit Level. . . . .	98
Logical Instructions . . . . .	99
Shifting and Rotating Bits . . . . .	100

Multiplying and Dividing with Shift Instructions . . . . .	102
<b>Chapter 5 Defining and Using Complex Data Types . . . . .</b>	<b>105</b>
Arrays and Strings . . . . .	105
Declaring and Referencing Arrays. . . . .	105
Declaring and Initializing Strings . . . . .	108
Processing Strings . . . . .	110
Structures and Unions . . . . .	117
Declaring Structure and Union Types . . . . .	118
Defining Structure and Union Variables. . . . .	121
Referencing Structures, Unions, and Fields . . . . .	126
Nested Structures and Unions . . . . .	128
Records . . . . .	129
Declaring Record Types . . . . .	130
Defining Record Variables . . . . .	131
Record Operators . . . . .	133
<b>Chapter 6 Using Floating-Point and Binary Coded Decimal Numbers . . . . .</b>	<b>135</b>
Using Floating-Point Numbers . . . . .	136
Declaring Floating-Point Variables and Constants. . . . .	136
Storing Numbers in Floating-Point Format. . . . .	138
Using a Math Coprocessor . . . . .	139
Coprocessor Architecture. . . . .	140
Instruction and Operand Formats . . . . .	141
Coordinating Memory Access. . . . .	145
Using Coprocessor Instructions . . . . .	146
Using An Emulator Library. . . . .	155
Using Binary Coded Decimal Numbers . . . . .	156
Defining BCD Constants and Variables . . . . .	157
BCD Calculations on a Coprocessor . . . . .	157
BCD Calculations on the Main Processor . . . . .	158
<b>Chapter 7 Controlling Program Flow . . . . .</b>	<b>161</b>
Jumps. . . . .	161
Unconditional Jumps . . . . .	162
Conditional Jumps. . . . .	164
Loops. . . . .	172
Loop-Generating Directives . . . . .	173
Writing Loop Conditions . . . . .	178
Procedures . . . . .	180
Defining Procedures . . . . .	180

Passing Arguments on the Stack . . . . .	182
Declaring Parameters with the PROC Directive . . . . .	184
Using Local Variables. . . . .	188
Creating Local Variables Automatically . . . . .	190
Declaring Procedure Prototypes . . . . .	193
Calling Procedures with INVOKE . . . . .	194
Generating Prologue and Epilogue Code. . . . .	198
MS-DOS Interrupts . . . . .	204
Calling MS-DOS and ROM-BIOS Interrupts . . . . .	204
Replacing an Interrupt Routine . . . . .	206
<b>Chapter 8 Sharing Data and Procedures Among Modules and Libraries . . . . .</b>	<b>211</b>
Selecting Data-Sharing Methods. . . . .	211
Sharing Symbols with Include Files . . . . .	212
Organizing Modules . . . . .	212
Declaring Symbols Public and External . . . . .	214
Positioning External Declarations. . . . .	228
Using Alternatives to Include Files . . . . .	219
PUBLIC and EXTERN . . . . .	220
Other Alternatives . . . . .	221
Developing Libraries. . . . .	221
Associating Libraries with Modules . . . . .	222
Using EXTERN with Library Routines . . . . .	223
<b>Chapter 9 Using Macros . . . . .</b>	<b>225</b>
Text Macros. . . . .	226
Macro Procedures . . . . .	226
Creating Macro Procedures. . . . .	227
Passing Arguments to Macros . . . . .	228
Specifying Required and Default Parameters . . . . .	229
Defining Local Symbols in Macros . . . . .	232
Assembly-Time Variables and Macro Operators . . . . .	233
Text Delimiters and the Literal-Character Operator . . . . .	234
Expansion Operator . . . . .	235
Substitution Operator . . . . .	237
Defining Repeat Blocks with Loop Directives . . . . .	239
REPEAT Loops. . . . .	240
WHILE Loops. . . . .	241
FOR Loops and Variable-Length Parameters . . . . .	242
FORC Loops. . . . .	244
String Directives and Predefined Functions . . . . .	245

Returning Values with Macro Functions. . . . .	248
Returning Values with EXITM. . . . .	248
Using Macro Functions with Variable-Length Parameter Lists. . . . .	249
Expansion Operator in Macro Functions . . . . .	251
Advanced Macro Techniques . . . . .	251
Defining Macros within Macros . . . . .	251
Testing for Argument Type and Environment . . . . .	252
Using Recursive Macros . . . . .	255
<b>Chapter 10 Writing a Dynamic-Link Library For Windows . . . . .</b>	<b>257</b>
Overview of DLLs. . . . .	257
Loading a DLL. . . . .	258
Building a DLL . . . . .	260
DLL Code . . . . .	261
DLL Data. . . . .	265
DLL Stack . . . . .	265
DLL Extension Names . . . . .	266
Summary . . . . .	266
Example of a DLL: SYSINFO . . . . .	267
Entry Routine for SYSINFO . . . . .	268
Expanding SYSINFO . . . . .	270
<b>Chapter 11 Writing Memory-Resident Software. . . . .</b>	<b>273</b>
Terminate-and-Stay-Resident Programs. . . . .	273
Structure of a TSR . . . . .	274
Passive TSRs . . . . .	274
Active TSRs. . . . .	275
Interrupt Handlers in Active TSRs. . . . .	275
Auditing Hardware Events for TSR Requests . . . . .	275
Monitoring System Status . . . . .	277
Determining Whether to Invoke the TSR . . . . .	279
Example of a Simple TSR: ALARM . . . . .	279
Using MS-DOS in Active TSRs . . . . .	285
Understanding MS-DOS Stacks . . . . .	285
Determining MS-DOS Activity. . . . .	285
Interrupting MS-DOS Functions. . . . .	286
Monitoring the Critical Error Flag. . . . .	287
Preventing Interference . . . . .	288
Trapping Errors . . . . .	288
Preserving an Existing Condition. . . . .	289
Preserving Existing Data . . . . .	290

---

Communicating Through the Multiplex Interrupt . . . . .	290
The Multiplex Handler . . . . .	291
Using the Multiplex Interrupt Under MS-DOS Version 2.x. . . . .	292
Deinstalling a TSR . . . . .	292
Example of an Advanced TSR: SNAP . . . . .	293
Building SNAP.EXE . . . . .	294
Outline of SNAP . . . . .	295
<b>Chapter 12 Mixed-Language Programming . . . . .</b>	<b>307</b>
Naming and Calling Conventions . . . . .	308
Naming Conventions . . . . .	309
The C Calling Convention. . . . .	309
The Pascal Calling Convention . . . . .	310
The STDCALL and SYSCALL Calling Conventions. . . . .	311
Writing an Assembly Procedure For a Mixed-Language Program . . . . .	312
The MASM/High-Level-Language Interface. . . . .	313
The C/MASM Interface . . . . .	315
The C++/MASM Interface . . . . .	322
The FORTRAN/MASM Interface . . . . .	323
The Basic/MASM Interface . . . . .	328
<b>Chapter 13 Writing 32-Bit Applications . . . . .</b>	<b>335</b>
32-Bit Memory Addressing . . . . .	335
MASM Directives for 32-Bit Programming. . . . .	336
Sample Program. . . . .	337

## Appendixes

<b>Appendix A Differences Between MASM 6.1 and 5.1. . . . .</b>	<b>341</b>
New Features of Version 6.1 . . . . .	342
The Assembler, Environment, and Utilities. . . . .	342
Segment Management . . . . .	343
Data Types . . . . .	344
Procedures, Loops, and Jumps . . . . .	347
Simplifying Multiple-Module Projects . . . . .	348
Expanded State Control . . . . .	349
New Processor Instructions. . . . .	350
Renamed Directives . . . . .	350
Macro Enhancements. . . . .	351
MASM 6.1 Programming Practices . . . . .	352
Compatibility Between MASM 5.1 and 6.1. . . . .	352

Rewriting Code for Compatibility . . . . .	353
Using the OPTION Directive . . . . .	361
Changes to Instruction Encodings . . . . .	377
<b>Appendix B BNF Grammar . . . . .</b>	<b>379</b>
<b>Appendix C Generating and Reading Assembly Listings. . . . .</b>	<b>397</b>
Generating Listing Files . . . . .	397
Precedence of Command-Line Options and Listing Directives. . . . .	399
Reading the Listing File . . . . .	399
Generated Code . . . . .	399
Error Messages. . . . .	400
Symbols and Abbreviations . . . . .	400
Reading Tables in a Listing File . . . . .	404
<b>Appendix D MASM Reserved Words . . . . .</b>	<b>407</b>
Operands and Symbols. . . . .	407
Special Operands for the 80386/486 . . . . .	409
Predefined Symbols . . . . .	409
Registers. . . . .	409
Operators and Directives . . . . .	410
Processor Instructions . . . . .	412
8086/8088 Processor Instructions. . . . .	412
80186 Processor Instructions . . . . .	413
80286 Processor Instructions . . . . .	413
80286 and 80386 Privileged-Mode Instructions . . . . .	413
80386 Processor Instructions . . . . .	413
80486 Processor Instructions . . . . .	414
Instruction Prefixes . . . . .	414
Coprocessor Instructions . . . . .	414
8087 Coprocessor Instructions . . . . .	414
80287 Privileged-Mode Instruction . . . . .	415
80387 Instructions. . . . .	415
<b>Appendix E Default Segment Names . . . . .</b>	<b>417</b>
<b>Glossary . . . . .</b>	<b>421</b>
<b>Index. . . . .</b>	<b>435</b>

# Figures and Tables

## Figures

1.1 Segment Allocation. . . . .	6
1.2 Calculating Physical Addresses . . . . .	8
1.3 Registers for 8088-80286 Processors . . . . .	17
1.4 Extended Registers for the 80386/486 Processors . . . . .	18
1.5 Flags for 8088-80486 Processors. . . . .	20
3.1 Stack Status Before and After Pushes and Pops . . . . .	72
4.1 Integer Formats . . . . .	87
4.2 Shifts and Rotates . . . . .	101
6.1 Encoding for Real Numbers in IEEE Format . . . . .	138
6.2 Coprocessor Data Registers. . . . .	140
6.3 Status of the Register Stack. . . . .	142
6.4 Status of the Register Stack and Memory Locations . . . . .	143
6.5 Status of the Previously Initialized Register Stack . . . . .	144
6.6 Status of the Already Initialized Register Stack . . . . .	144
6.7 Status of the Register Stack: Main Memory and Coprocessor. . . . .	148
6.8 Coprocessor Control Registers. . . . .	154
6.9 Coprocessor and Processor Control Flags. . . . .	155
7.1 Program Arguments on the Stack . . . . .	183
7.2 Local Variables on the Stack. . . . .	190
7.3 Operation of Interrupts . . . . .	206
8.1 Using EXTERNDEF for Variables. . . . .	215
8.2 Using PROTO and INVOKE . . . . .	217
8.3 Using PUBLIC and EXTERN. . . . .	221
11.1 Time Line of Interaction Between Interrupt Handlers for a Typical TSR. . . . .	278
11.2 Flowchart for SNAP.EXE: Installation Phase . . . . .	296
11.3 Flowchart for SNAP.EXE Resident Phase . . . . .	297
11.4 Flowchart for SNAP.EXE Deinstallation Phase. . . . .	298
12.1 C String Format . . . . .	316
12.2 C Stack Frame. . . . .	320
12.3 FORTRAN String Frame . . . . .	324
12.4 FORTRAN Stack Frame. . . . .	327
12.5 Basic String Descriptor Format . . . . .	330
12.6 Basic Stack Frame . . . . .	333
B.1 BNF Definition of the TYPEDEF Directive. . . . .	380



**Tables**

1.1 8086 Family of Processors . . . . . 2

1.2 The MS-DOS and Windows Operating Systems Compared . . . . . 4

1.3 Operator Precedence . . . . . 14

2.1 Attributes of Memory Models . . . . . 35

3.1 Indirect Addressing with 16-Bit Registers . . . . . 68

4.1 Division Operations . . . . . 97

5.1 Requirements for String Instructions . . . . . 112

6.1 Ranges of Floating-Point Variables . . . . . 136

6.2 Coprocessor Operand Formats . . . . . 141

6.3 Control-Flag Settings After Comparison or Test . . . . . 151

7.1 Conditional Jumps Based on Comparisons of Two Values . . . . . 167

9.1 MASM Macro Operators . . . . . 234

11.1 MS-DOS Internal Stacks . . . . . 286

12.1 Naming and Calling Conventions . . . . . 309

12.2 Register Conventions for Simple Return Values . . . . . 317

A.1 Requirements for String Instructions . . . . . 353

C.1 Options for Generating or Modifying Listing Files . . . . . 398

C.2 Symbols and Abbreviations in Listings . . . . . 400

C.3 Symbols in Timing Column . . . . . 401