Index

! (literal-character operator) 235
!= (not equal operator) 178
” (double quotation marks) 109, 353
$ (current address operator) 368
% (expansion operator) 235, 248, 357
& (substitution operator) 238, 372
&& (logical AND operator) 178
’ (single quotation mark) 109, 353
( ) (parentheses) 106
+ (plus operator) 63, 66, 352, 370
. (dot operator) 126
. (structure-member operator) 64, 67, 352, 370
.186 directive 38
.286 directive 38
.286P directive 38
.287 directive 38
.386 directive
    FLAT, with 26, 36
    processor mode, specifying 38, 336
    segment mode, setting 46, 68
.386P directive 38
.387 directive 38
.486 directive
    FLAT, with 36
    processor mode, specifying 38
    segment mode, setting 46, 68
.486P directive 38
.8087 directive 38
: (colon) 22, 352, 354
: (segment-override operator) 50, 59–60, 64
:: (double colon) 197, 215, 352–354
; (semicolon) 21
;; (double semicolon) 227
< (less than operator) 178
< > (angle brackets) *See* Angle brackets
= = (equal operator) 178
> (greater than operator) 178
? (question mark initializer)
    array elements 109
    described 368
    variables 87
@ (at sign) 10
@@: (anonymous label) 170
[ ] (brackets) 107
[ ] (index operator) 63
\ (backslash character), MASM code 22
\ (line-continuation character) 121
{} (curly braces) 121, 131
|| (logical OR operator) 178
32-bit programming 335
80186 processor 3
80188 processor 3
80286 processor 3
80287 math coprocessor 3, 135
80386 processor 3, 335
80387 math coprocessor 3, 135
80486 processor 3, 135
8086-based processors 2–3
8087 math coprocessor 3, 135
8088 processor 3

A

AAD instruction 160
AAM instruction 160
ABS operand 220
Accessing data with pointers *See* Pointer variables
ADC instruction 92–94
ADD instruction 92–94
ADDR operator 197
Addresses
    displacement of 65
    dynamic 79
    effective 65
    errors in 54
    far 57, 74, 80
    near 57, 80
    physical 7
    registers, loading into 80
    relocatable 57
    segmented 7–8, 53
Addressing
    direct registers, used in 62–63
    indirect registers, used in 65, 68
    scaling operands 70
    specifying 60
Aliases 87, 369
ALIGN directive 3
Align types 45
    See also individual entries
.ALPHA directive 47
AND instruction 27, 99, 100
Angle brackets (< >)
    default parameters 230
    epilogues 202
    FOR loops 242

FORC loops 244

Angle brackets (< >) *(continued)*
macro text delimiters 234
prologues 202
records 131
structures and unions 121
Anonymous label (@@) 170
API (Application Programming Interface) 257
Architecture, segmented 2, 5
Architecture, unsegmented 5
Arguments
errors 196
macro 252
mixed-language programs, passing in 314
qualifiedtypes, with 16
stack, on 182
Arrays
accessing elements in 105
declaring 105
defined 105
defining 15
DUP, declaring with 106, 124
instructions for processing 110
length of 108
multiple-line declarations for 105
number of bytes in 108
referencing 108, 316
size of elements 108
with DUP operator *See* DUP operator
with SIZEOF operator *See* SIZEOF operator
with TYPE operator *See* TYPE operator
ASCIIZ 267
Assembly
actions during 23
conditional *See* Conditional assembly
INCLUDE files 212
language
book list xviii
mixed-language programs 312
listing files *See* Listing files
two-pass 358
Assembly pointers *See* Conditional assembly
Assembly-time variables 233
ASSUME directive
.MODEL, generated with 37
code segments, changing 357
enhancements 344
general-purpose registers 77
segment registers, setting 49–55, 58–59, 357
AT address combine type 46
/AT command-line option, ML 36
At sign (@) 10

B

Backslash character (\) 22
Backus-Naur Form *See* BNF grammar
Base Pointer (BP) register 73
Basic calling conventions 308–310
Basic/MASM programs 328–332
Binary Coded Decimals
  calculating with 156–160
  defining 156
  instructions for 156–160
  packed 158
  unpacked 159–160
Bits
  mask 99–102
  rotating 100
  shifting 100
BNF grammar 16, 379–380
BOUND instruction 108, 204
BP (Base Pointer) register 73
Brackets ([ ]) 107
.BREAK directive 173, 176
BSF instruction 100
BSR instruction 100
BYTE
  align type 45
  directive 86

C

C calling convention 309
C++/MASM programs 322–323
C/MASM programs 315–321
CALL instruction 180
Calling conventions 309
  Basic 308–310
  directives, specifying 37
  FORTRAN 308–310
  (list) 308
  mixed-language programming 308–309
  Pascal 310
  STDCALL 311
  SYSCALL 308–311
CARRY? flag as operand 178
Case sensitivity
  enforcing 348
  macro functions, predefined 245
  MASM statements 22
  radix specifiers 11
  reserved words 9, 407
  specifying
    command-line options, in 25
    language type 348
OPTION directive 25

Case sensitivity *(continued)*
symbols, predefined 10
CASEMAP
ALL argument, OPTION directive 25
NONE argument, OPTION directive 25
NOTPUBLIC argument, OPTION directive 25
CATSTR directive 245–247
CATSTR, compared with TEXTEQU directive *See* TEXTEQU directive
@CatStr predefined string function 245–247
CBW instruction 90
CDQ instruction 90
CLC instruction 104
Cleaning the stack 185
CLI instruction 5, 209
Client program 257, 266
CMC instruction 104
CMP instruction 166
CMPS instruction 110–114, 353
CMPSB instruction 114
.CODE directive 33, 40–42
Code segment *See* Segments, code
Code, near or far 57
@CodeSize predefined symbol 40
CodeView for Windows 264
Combine types
(list) 46
See also individual entries
.COM files
relocatable segment expression, lacking 62
starting address 56
tiny model, using 36, 46–47
COMM directive 16, 211, 217–218
Command-line driver, ML xvi
Command-line options *See* ML command-line options
COMMENT directive 22
Comments
extended lines, in 346
macros, in 227
source code 21–22
COMMON combine type 46
Communal variables 217
Compact model *See* Memory models, compact
Compatibility, MASM 5.1 *See* MASM 5.1 compatibility
Conditional assembly
assembly behavior, changing 23
conditions, testing for 28
directives 28
pointers 83, 187
Conditional-error directives (table) 29
Conditional jumps 164–170
Conditions, testing for conditional assembly *See* Conditional assembly
Constants
defined 11
expressions 12
immediate 61
integer 11–12
size 363
size of 12
symbolic 12
.CONST directive 33, 39–40
.CONTINUE directive 173, 176
Coprocessors
architecture 140–144
control registers 156
data format in registers 140
defined 135
described 3, 139
instructions
arithmetic 148–150
data transfer 146
described 146
(list) 414
overview 141
program control 151–155
memory access 145
operand formats
classical stack 141
memory 142
overview 141
register 143
register-pop 144
specifying 37, 140
status word register 156
steps for using 145
/Cp command-line option, ML 10, 245
@Cpu predefined symbol 254
Curly braces ({})
records 131
structures and unions 121
Current address operator ($) 368
@CurSeg predefined symbol 39, 219
CWD instruction 90
CWDE instruction 90
/Cx command-line option, ML 158

D

DAA instruction 162
DAS instruction 162
.DATA directive 33, 39–40
.DATA? directive 33, 39–40
@data predefined symbol 39
Data segment *See* Segments, data
@DataSize predefined symbol 39, 83

Data types
arrays *See* Arrays
attributes for 15
Binary Coded Decimals 159
defined 14
defining 87
directives 14
floating-point 136
initializers, as 14
integers, allocating memory for 85–86
new features, MASM 6.1 344
qualifiedtypes 15, 214
real 14, 136
signed 14, 86
strings *See* Strings
structures 117
unions 117
user-defined 15
Data, near or far 57, 58
Data-sharing methods 211
Data-sharing methods, multiple-module programs *See* Multiple-module programs
Date, system 11
DB directive 86
DD directive 86
DEC instruction 92–94
DF directive 86
DGROUP group name
.MODEL, defined by 34, 39, 51
DS registers, initializing to 56
MS-DOS programs, for 41–42
near data, accessing 57–58
segment 35–37, 51–52, 57
Direct memory operands
loading offset of 82
overview 60–64
Directives
.286P 38
.287 38
.386 *See* .386 directive
.386P 38
.387 38
.486P 38
.8087 38
.ALPHA 47
ALIGN 3
.BREAK 173, 176
BYTE 86
CATSTR 245–247
.CODE 33, 40–42
COMM 16, 211, 217–218
COMMENT 22
Conditional assembly 28
Conditional error 29, 358

Directives *(continued)*
.CONST 33, 39–40
.CONTINUE 173, 176
.DATA 33, 39–40
.DATA? 33, 39–40
Data declarations, for 87
Data types, for 14
Data-sharing *See* EXTERN directive
DB 86
DD 86, 136
Decision 171
DF 86
.DOSSEG 47
DQ 86, 136
DT 86, 136
DW 86
DWORD 86
ECHO 236
.ELSE 171
ELSE 28
.ELSEIF 171
ELSEIF 28
ELSEIF1 29, 358
ELSEIF2 29, 358
END 33, 56
.ENDIF 171
ENDIF 28
ENDM 227–239
ENDP 180–181, 206
ENDS 44
.ENDW 173
EQU 12, 369
.ERR 30
.ERR1 30, 358
.ERR2 30, 358
.ERRB 30, 231
.ERRDEF 30
.ERRDIF 30
.ERRE 30
.ERRIDN 29
.ERRNB 29, 231
.ERRNDEF 29
.ERRNZ 29
EVEN 3
.EXIT 33, 41–43
EXITM 248
EXTERN *See* EXTERN directive
EXTERNDEF *See* EXTERNDEF directive
FARDATA 33, 39–40
.FARDATA 39–40
.FARDATA? 33, 39–40
Floating-point 136
FOR 242–243, 249
FORC 244

Directives *(continued)*
FWORD 86
GROUP 51–52
.IF 171
IF 28–29
IF1 29, 358
IF2 29, 358
IFB 29, 231
IFDEF 29, 359
IFDIF 29
IFE 29
IFIDN 29
IFNB 29, 231
IFNDEF 29, 359
INCLUDE 212
INCLUDELIB 222
INSTR 245–246
INVOKE *See* INVOKE directive
LABEL 16
LOCAL 188–191, 232
loop-generating 173
.MODEL *See* .MODEL directive
.MSFLOAT 361
Naming conventions 37
.NO87 38, 349
obsolete 361
OPTION *See* OPTION directive
ORG 56
POPCONTEXT 255, 349
PROC 180–184, 193, 206, 312
PUBLIC 185, 211, 220
PUSHCONTEXT 255, 349
QWORD 86
.RADIX 11
REAL4 136–137
REAL8 136–137
REAL10 136–137
RECORD 130–131
Renamed since MASM 5.1 350
.REPEAT 173–177
REPEAT 240
SBYTE 86
SDWORD 86
SEGMENT 44–47
Segment order, controlling 47
.SEQ 47
SIZESTR 245–246
STACK *See* STACK directive
.STARTUP *See* .STARTUP directive
STARTUP *See* .STARTUP directive
STRUCT 118–129
SUBSTR 245–246
SWORD 86
TBYTE 86, 159

Directives *(continued)*
TEXTEQU *See* TEXTEQU directive
UNION 118–119, 122, 125–129
.UNTIL 173
.UNTILCXZ 173
.WHILE 173–177
WHILE 241
WORD 86
Directives: 36–38, 46
Displacement 66
Distance attributes 15
DIV instruction 97–98
Division 97, 102
DLLs
client program 257, 266
data segment 265–269
defined 257, 266
example 267–268
extension name 266
heap 261–262, 265–267
IMPLIB utility 258
initialization 261–262, 268–269
loading 258–260
programming requirements 260–261, 267
prologue and epilogue 264–267
stacks in 46, 264–267
summary 266
termination 262–264, 270
Document conventions vii
DOS *See* MS-DOS
.DOSSEG directive 47
Dot (.) operator *See* Structure-member operator
DOTNAME argument, OPTION directive 25
Double colon (::) 197, 215
Double quotation marks (") 109
Double semicolon (;;) 227
Doublewords 86
DQ directive 86
DT directive 86
DUP operator
arrays, with 106, 124
record variables, with 131
structures and unions, with 121
DW directive 86
DWORD
align type 45
directive 86
Dynamic-link libraries *See* DLLs

E

ECHO directive 236
.ELSE directive 171
ELSE directive 28

.ELSEIF directive 171
ELSEIF directive 28
ELSEIF1 directive 358
ELSEIF2 directive 29, 358
EMULATOR argument, OPTION directive 27, 157
Emulator libraries 155–156
END directive 33, 56
.ENDIF directive 171
ENDIF directive 28
ENDM directive 227–239
ENDP directive 180–181, 206
ENDS directive 44
.ENDW directive 173
ENTER instruction 183
Environment
    target 4
    variables
        INCLUDE 213
        LIB 222
        returning values of 10
/EP command-line option, ML 342
EPILOGUE argument, OPTION directive 26, 201–203
Epilogue code
    defined 198
    macros 201–202, 264–265
    PROC statement, specifying arguments in 185
    procedures, with 26
    RET instruction 357
    standard 199
    user-defined 201
EQ operator 365
EQU directive 12, 369
Equal directive (=) 12
Equates, predefined *See* Predefined symbols
.ERR directive 29
.ERR1 directive 30, 358
.ERR2 directive 30, 358
.ERRB directive 29, 231
.ERRDEF directive 29
.ERRDIF directive 29
.ERRE directive 29
.ERRIDN directive 29
.ERRNB directive 29, 231
.ERRNDEF directive 29
.ERRNZ directive 29
Error detection 196
ERROR operand 49–50
Errors, argument passing 196
ESC instruction 360
EVEN directive 3
Executable (.EXE) files, controlling size of 223
Exit codes, Windows operating system 263
.EXIT directive 33, 41–43
EXITM directive 248

Expansion operator (%) 235–236, 248, 357
Explicit loading 258
Exponent bias 139
EXPORT operand 185
EXPORTS statement 261, 270
EXPR16 argument, OPTION directive 13, 26, 361, 373
EXPR32 argument, OPTION directive 13, 26, 373
Expression operators 178
Expressions
  assembly-time evaluation 23
  constant 12
  loop conditions, evaluating 179
  OPTION M510 behavior 364, 373
  order of evaluation 14
  size 366, 373
  word size 13, 26
Extension, filename 266
EXTERN directive
  data-sharing 211
  executable file size, limiting 223
  module-specific 220
  overview 16
  positioning 218
  procedure prototypes, declaring 193
External declarations 216–218
External variables 217, 369
EXTERNDEF directive
  data-sharing 211
  overview 16
  positioning 218
  procedure prototypes, declaring 193
  symbols, declaring 214–215

F

Far addresses, invoking 57, 74, 80–81, 197
Far code 57
Far data 58–60
.FARDATA directive 33, 39–40
.FARDATA? directive 39–40
FAR operator 169, 185
Far pointer 74, 80–81
FARSTACK operand
  example 35
  grouping 34
  in Windows-based programs 266
  MS-DOS program, initializing 43
  special cases, setting for 37
Farwords 86
FCOM instruction 153
Fields, statements in 21–22
Files
  .COM
  relocatable segment expression, lacking 62

Files *(continued)*
.COM *(continued)*
starting address 56
tiny model, using 36, 46–47
executable 24
include 212–213, 348
line numbers 11
naming 11
Flags
CARRY? 178
operands, as 178
OVERFLOW? 178
PARITY? 178
SIGN? 178
stack, saving on 73
ZERO? 178
Flags register *See* Registers, flags
Flat model *See* Memory models, flat
FLAT operand 46, 49–50
FLD1 instruction 147
FLDZ instruction 147
Floating-point
calculations 3
constants
decimal form 137
encoded hexadecimal format 137
syntax for defining 136
emulation 157–158
IEEE format 139
instructions
arithmetic 148–149
controlling 26
data transfer 147
not emulated (list) 158
program control 152–153, 156
operations 146
values
double precision 139
single precision 139
variables
IEEE format 138
Microsoft binary format 138
.MSFLOAT format 138
ranges 136
FOR directive 242–243, 249
FORC directive 244
FORCEFRAME operand 200–201
FORTRAN calling convention 308–310
FORTRAN/MASM programs 323–326
/Fpi command-line option, ML 26, 157
Frame 62
FS register 17
FTST instruction 153

Full segment definitions
described 32
segment registers, initializing 54–56
using 44–51
Full segment defintions *See* .STARTUP directive
FWORD directive 86
FXCH instruction 144

G

Global variables 211
GROUP directive 51–52
Groups
defined 51
DGROUP 51
SEG operator, returned by 62
GS register 17

H

H2INC 318
Heap space 261–262, 265–267
HEAPSIZE statement 261, 271
Help, online *See* Microsoft Advisor
HIGH operator 356
HIGHWORD operator 346
Huge model *See* Memory models, huge

I

/I command-line option, ML 213
Identifiers
ABS, using 220
naming restrictions 9, 346, 353, 357, 368
OPTION DOTNAME 373
OPTION NOKEYWORD 376
IDIV instruction 97–98
IEEE format 139
.IF directive 171
IF directive 28–29
IF1 directive 29, 358
IF2 directive 29, 358
IFB directive 29, 231
IFDEF directive 29, 359
IFDIF directive 29
IFE directive 29
IFIDN directive 29
IFNB directive 29, 231
IFNDEF directive 29, 359
Immediate operands 60–62
IMPLIB utility 258
Implicit loading 258
Import libraries 258

IMPORTS statement 266
IMUL instruction 95–96
IN instruction 5
INC instruction 92–94
INCLUDE directive 212
INCLUDE environment variable 213
Include files
  assembling 213
  nested 213
  overview 212, 348
INCLUDELIB directive 222
Index operator ([ ]) 63
Indirect memory operands 60, 64–70
Indirect procedure calls *See* INVOKE directive
Initializers
  allocating 87
  directives for 15
  multiple-line 346
Instance 261, 266
INSTR directive 245–246
@InStr predefined string function 245–246
Instruction Pointer (IP) register 20, 57, 161
Instructions
  ADC 92–94
  ADD 92–94
  AND 26, 99–100
  arithmetic 378
  bit-test 354
  BOUND 108, 204
  BSF 100
  BSR 100
  CALL 180
  CBW 90
  CDQ 90
  CLC 104
  CLI 5, 209
  CMC 104
  CMP 166
  CMPS 110–114, 353
  CMPSB 114
  conditional-jump 165–167
  coprocessor 377
  CWD 90
  CWDE 90
  DAA 162
  DAS 162
  DEC 92–94
  default segments, requiring 49
  DIV 97–98
  encodings, changes to 377–378
  ENTER 183
  ESC 360
  FCOM 153
  FLD1 147
Instructions *(continued)*
  FLDZ 147
  floating-point *See* Floating-point instructions
  FTST 153
  FXCH 144
  IDIV 97–98
  IMUL 95–96
  IN 5
  INC 92–94
  INT 204–205
  INTO 207
  JCXZ 170–173
  JECXZ 170–173
  JMP 49, 162
  JO 165
  jump 165–167, 170, 173
  LAHF 73
  LDS 81
  LEA 82, 104
  LEAVE 183
  LES 81
  (list) 412
  LOCK 353, 363
  LODS 110–115, 353
  logical 99–102
  LOOP 172
  LOOPE 172
  LOOPNE 172
  LOOPNZ 172
  LOOPZ 172
  MOV 49, 82, 89
  MOVS 110–113, 353
  MOVSX 92
  MOVZX 92
  MUL 95–96
  NOP 377
  NOT 99–100
  obsolete 360
  operands for 60
  OR 26, 99–100, 168
  OUT 5
  POP 49, 71
  POPA 74
  POPAD 74
  POPF 73
  POPFD 73
  privileged 2, 38
  PUSH 49, 71
  PUSHA 74
  PUSHAD 74
  PUSHF 73
  PUSHFD 73
  RCL 101–104
  RCR 101–104

Instructions *(continued)*
REP 110–112, 363
REPE 110–112, 363
REPNE 110–112, 353, 363
REPNZ 110–112, 353, 363
REPZ 110–112, 363
RET 378
RETF 181, 378
RETN 181, 378
ROL 101–104
ROR 101–104
SAL 101–104
SAR 101–104
SBB 92–94
SCAS 110–115, 353
SHL 101–104
SHR 101–104
STC 104
STI 5, 209
STOS 110–113, 353
SUB 92–94
TEST 167–168
timing xvii, 399–400
XCHG 90
XLAT 116
XLATB 116
XOR 26, 99–100
Integers
adding 92–94
allocating memory for 85–86
Binary Coded Decimal (BCD) 159
bit operations on 99
constants, defining 11–12
dividing 97–98
exchanging 90
hexadecimal 12
initializing 87
memory format 86
moving 89
multiplying 95–96
operations with 88
popping off stack 71
pushing onto stack 71
radix specifiers for 11
sign-extending 90
signed 86
size of 86
stack 71
subtracting 92–94
translating 116
types, defining 14, 86
value range 86
@Interface predefined symbol 37
Interrupt vector 205

Interrupt-enable flag 205
Interrupts
CLI instruction 209
handlers 206–207
INT instruction 204–205
MS-DOS 204, 285
operation 206
overview 204
redefining 207
STI instruction 209
vector table 205
INTO instruction 207
INVOKE directive
actions 194
ADDR, invoking 197
arguments, widening 196
error detection 196
far addresses, invoking 197
generated code, checking 198
indirect procedure calls 198
mixed-language programs 312–313
procedures, calling 193–197, 216
type conversions 194–195

J

JCXZ instruction 170–173
JECXZ instruction 170–173
JMP instruction 49, 162
JO instruction 165
Jumps
anonymous 170
automatic 169
conditional
bit status 167
comparisons 166
extending 26, 169
flag status 165–166
instructions (list) 165–167
overview 164
zero value 168
directives for 171
extension, automatic 26, 169
instructions 165–167
optimization, automatic 162
overview 161
unconditional
indirect operands 163
jump tables 163
overview 162

L

LABEL directive 16

Labels
anonymous 170
code
length 346
OPTION M510 behavior 363
OPTION NOSCOPED 375
procedures, in 357
referencing 352
size 346
visibility 354
LAHF instruction 73
LANGUAGE
BASIC argument, OPTION directive 26
C argument, OPTION directive 26
FORTRAN argument, OPTION directive 26
PASCAL argument, OPTION directive 26
STDCALL argument, OPTION directive 26
SYSCALL argument, OPTION directive 26
LANGUAGE argument, OPTION directive 193
Language attributes
.MODEL directive, with 34, 37
OPATTR operator 253
OPTION directive, with 26
Large model *See* Memory models, large
LDS instruction 81
LEA instruction 82, 104
LEAVE instruction 183
Length of strings *See* LENGTHOF operator
LENGTH operator 356–357, 364
LENGTHOF operator
number of items, returning 110, 124, 132, 346
structures, defining 108
unions, with 125
LES instruction 81
Libraries
C run-time 271
emulator 155–156
overview 221
source files, specifying in 222
LIBRARY statement 270
Line-continuation character (\) 121
LINK, command-line options *See* individual entries
Linkage specification 322–323
Linking
actions during 24, 45
segment order in 48
Listing files
code generated 399
command-line options 397–399
error messages 400
examples 401
generating 397
PWB options 397–399
reading 399, 405

Listing files *(continued)*
symbols used in (list) 400
tables in 405–406
Literal-character operator (!) 235
LJMP argument, OPTION directive 27
LOADDS operand 200–201
Loading local address variables *See* Local variables
Loading, actions during 24
Local addresses, loading *See* Local variables
LOCAL directive 188–191, 232
Local variables
creating 188
loading addresses of 82
procedures, in 188
LOCK instruction 353, 363
LODS instruction 110–115, 353
Logical AND 178
Logical instruction 99–100
Logical line 22
Lookup tables 241
LOOP instruction 172
LOOPE instruction 172
LOOPNE instruction 172
LOOPNZ instruction 172
Loops
conditions
expression evaluation 179
precedence 179
PTR operator in 178
relational operators for (list) 178
signed operands 178
writing 178
controlling 176
directives
.REPEAT 173–177
.WHILE 173–177
instructions (list) 172
macros
FOR 242–243, 249
FORC 244
REPEAT 240
WHILE 241
LOOPZ instruction 172
LOW operator 356
LOWWORD operator 346, 366
LROFFSET operator 344

M

M510 argument, OPTION directive
compatibility with MASM 5.1 26, 353–370
expression word size, setting 13
structures, with 119

Macros
arguments
commas 352, 372
quotation marks 353
testing 29, 252
variable 242, 249
calling 227
checking argument types with 253
comments (;;) 227
expansion 23
functions
defined 248
epilogues 201
EXITM 248
prologues 201
returning values 248
local symbols in 232
loops
FOR 242–243, 249
FORC 244
REPEAT 240
WHILE 242–243
MASM 5.1 behavior 25, 356, 372
nested 251
new features 351
operators
behavior in macro functions 251
expansion (%) 235–236, 248, 357
(list) 234
literal-character (!) 235
substitution (&) 238, 352, 372
OPTION OLDMACROS 372
parameters
default values 230
procedure parameters, compared to 234
required 229
substitution 238
passing arguments to 228, 235
predefined string functions 11
procedures
defined 226
functions, compared to 228
recursive 255
redefining 251
text
defined 226
forward referencing 356
numeric equates, compared to 234
OPTION M510 behavior 370
syntax 226
VARARG keyword 242, 249, 351
writing 227
Mask
defined 99

Mask *(continued)*
logic instructions, with 102
record operators, with 133
MASK operator 133
MASM 5.1 compatibility
address fixups 26
macro behavior 25, 356, 372
OPTION directive, specifying 25
overview xvi
structures 25
updating code 353–360
MASM utility xvi, 342
Math coprocessor *See* Coprocessors
Medium model *See* Memory models, medium
Memory
access 64
allocation 24
virtual 5
MEMORY combine type 46
Memory models
attributes 35
compact 36
described 34
determining 10
far code segments 40
far data segments 40
flat 36, 58, 336
huge 36
large 36
medium 36
model-independent code 83
near code segments 40
small 36
specifying in PROC statement 185
tiny 36, 46–47
Memory-resident programs *See* TSRs
Microsoft Advisor xiii, 342
Minus operator (–) 64
Mixed-language programming
argument passing 314
assembly procedures 312
Basic/MASM programs 328–332
C prototypes, converting with H2INC 318
C++/MASM programs 322–323
C/MASM programs 315–321
calling conventions
Basic 308–310
FORTRAN 308–310
(list) 308
Pascal 310
STDCALL 311
SYSCALL 308–311
column-major order 315

Mixed-language programming *(continued)*
compatible data types
Basic (list) 328
C (list) 315
FORTRAN (list) 323
external data 314
FORTRAN/MASM programs 323–326
initialization code 313, 321
INVOKE, using 312–313
naming conventions 308–309
overview 307
register preservation 314
row-major order 315
ML command-line options
/AT 36
/Cp 10, 245
described xvi
/EP 342
/Fpi 26, 157
/I 213
listing options (list) 397
overview xvi
/X 213
/Zm 62, 119
/Zp 119
Mode, real, protected *See* Real mode; Protected mode
.MODEL directive
attributes 34–35
DGROUP 51
language types, specifying 26, 308
memory model, defining 35–36
mode default 46
overview 34
positioning 46
simplified segment directives 33
@Model predefined symbol 35, 83
Module-definition file
described 270
statements
EXPORTS 261, 270
HEAPSIZE 261, 271
IMPORTS 266
LIBRARY 270
STUB 266
Module-specific EXTERN directive *See* EXTERN directive
MOV instruction 49, 82, 89
MOVS instruction 110–113, 353
MOVSX instruction 92
MOVZX instruction 92
MS-DOS interrupts 204, 285
MS-DOS operating system 2–6
MUL instruction 95–96
Multiple-module programs
alternatives to include files 219

Multiple-module programs *(continued)*
COMM, using 217
data-sharing methods 211
declaring symbols public and external 214
EXTERN with library routines 223
external declarations, positioning 218
EXTERNDEF, using 214
include files 212–213
libraries 221–222
modules 212
PROTO, using 216
PUBLIC and EXTERN, using 220
sharing symbols with include files 212
Multiplex interrupt 291, 304
Multiplication
instructions 95
shift operations 102

N

Naming conventions
directives 37
(list) 308
mixed-language programming 308–309
Naming restrictions 9
Naming restrictions, identifers *See* Identifiers
NE operator 365
Near address 57, 80
NEAR operator 169, 185
NEARSTACK operand
ASSUME statement 54
default stack type 37, 42
described 35
New features, MASM 6.1 xiv–xv, 342–351
NMAKE 270
.NO87 directive 38, 349
NODOTNAME argument, OPTION directive 25
NOEMULATOR argument, OPTION directive 27
NOKEYWORD argument, OPTION directive 9, 27, 353, 376
NOLJMP argument, OPTION directive 27, 170
NOM510 argument, OPTION directive 25
NONUNIQUE operand 118, 126
NOOLDMACROS argument, OPTION directive 26
NOOLDSTRUCTS argument, OPTION directive 26
NOP instruction 377
NOREADONLY argument, OPTION directive 27
NOSCOPED argument, OPTION directive 26, 362, 375
NOSIGNEXTEND argument, OPTION directive 27, 378
NOT instruction 99–100
NOTHING operand 49–50
Number of items with LENGTHOF operator *See* LENGTHOF operator
Numeric equates, compared to text macros 234

O

OFFSET
  FLAT argument, OPTION directive 27
  GROUP argument, OPTION directive 27
  SEGMENT argument, OPTION directive 27, 62
OFFSET operator 61, 82, 356, 374
Offsets
  accessing data with 74
  addresses 7
  described 5–7
  determining 23–24, 360, 374
  fixups for 26
OLDMACROS argument, OPTION directive 25, 239, 361, 372
OLDSTRUCTS argument, OPTION directive
  MASM 5.1 compatibility 25, 361, 370–372
  structures, with 119, 126
Online help *See* Microsoft Advisor
OPATTR operator 252–253
Operands
  ABS 220
  direct memory 60–64
  EXPORT 185
  FAR 15
  FARSTACK *See* FARSTACK operand
  FLAT 46, 49–50
  FORCEFRAME 244
  immediate 60–62
  indirect memory 60, 64–70
  NEAR 15
  PRIVATE READONLY 44–45
  registers 61
  size 66, 355
  USE16 44–46
  USE32 44–46
Operating systems
  (list) 4
  .MODEL, specifying with 34
  multitasking 6
  types *See* MS-DOS, Windows operating systems
Operators
  ADDR 197
  current address ($) 368
  dot (.) 126, 352, 370
  EQ 365
  expansion (%) 235–236, 248, 357
  expressions, in 12–13
  FAR 169, 185
  HIGH 356
  HIGHWORD 346
  index ([ ]) 63
  instructions, compared to 13
  LENGTH 356–357, 364
  LENGTHOF 346
Operators *(continued)*
  LOW 356
  LOWWORD 346, 366
  LROFFSET 344
  macro 251
  MASK 133
  minus (–) 64
  NE 365
  NEAR 169, 185
  OFFSET 61, 82, *See* OFFSET operator
  OPATTR 252–254
  plus (+) 63, 66
  precedence 14
  PTR *See* PTR operator
  PTR, example *See* PTR operator
  relational 357, 365
  relational (list) 178
  SEG 50, 62, 363
  segment-override (:) 59, 64
  SHORT 169
  SIZE 364–365
  size *See* PTR operator
  SIZEOF 86, 346
  structure-member (.) 64–67, 126, 352, 370
  substitution (&) 238
  .TYPE 252, 360
  TYPE 86
  WIDTH 133
OPTION directive
  CASEMAP 25
  described 23
  DOTNAME 25, 361, 373
  emulation mode 157
  EMULATOR 26, 157
  EPILOGUE 26, 201–203
  EXPR16 OPTION directive 13, 26, 361, 373
  EXPR32 OPTION directive 13, 26, 373
  LANGUAGE 26, 193
  language types, specifying 308
  list of arguments for 25
  LJMP 26
  M510 *See* M510 argument, OPTION directive
  NODOTNAME 25
  NOEMULATOR 26
  NOKEYWORD *See* NOKEYWORD argument, OPTION directive
  NOLJMP 27, 170
  NOM510 25
  NOOLDMACROS 26
  NOOLDSTRUCTS 26
  NOREADONLY 27
  NOSCOPED 26, 362, 375
  NOSIGNEXTEND 27, 378

OPTION directive *(continued)*
OFFSET 26, 62, 362, 374–375
OLDMACROS 25, 237
OLDSTRUCTS *See* OLDSTRUCTS argument, OPTION directive
PROC 185, 375
procedure use 26
PROLOGUE 26, 201–203
READONLY 26
SCOPED 25
SETIF2 25, 29–30
using 25, 361
OR instruction 27, 99–100, 168
ORG directive 56
OUT instruction 5
OVERFLOW? flag as operand 178

P

PAGE align type 45
PARA align type 45
Parentheses [( )] 106
PARITY? flag as operand 178
Pascal convention 310
Physical line 22
Plus operator (+) 66, 352, 370
Pointer variables 74–78
Pointers
accessing data with 74
arguments, as 80
copying 79
far 74, 80–81
initializing 78
location 74
operations 78
TYPEDEF, defined with 15, 75–78
types, to 15
Pointers and conditional Assembly *See* Conditional assembly
Pointers defined by TYPEDEF *See* TYPEDEF directive
POP instruction 49, 71
POPA instruction 74
POPAD instruction 74
POPCONTEXT directive 255, 349
POPF instruction 73
POPFD instruction 73
Positioning
EXTERN directive *See* EXTERN directive
EXTERNDEF directive *See* EXTERNDEF directive
Precedence operators 14
Predefined equates *See* Predefined symbols
Predefined functions for macros 11
Predefined string functions
@CatStr 245–247
@InStr 245–246

Predefined string functions *(continued)*
@SizeStr 245–246
@SubStr 245–246
Predefined symbols 39, 83
@Codesize 40
@Cpu 254
@CurSeg 39, 219
@Data 39
@DataSize 39, 83
@Interface 37
(list) 10, 409
@Model 35, 83
@stack 37
@Wordsize 39
case sensitivity 9–10
new to MASM 6.1 (list) 343
PRIVATE operand 185
Privilege levels 5
Problems, reporting xx
PROC
EXPORT argument, OPTION directive 25
PRIVATE argument, OPTION directive 25, 362
PUBLIC argument, OPTION directive 25, 185
PROC directive 180–184, 193, 206, 312
PROC statements with visibility *See also* Visibility
PROC with RET instruction *See* RET instruction
Procedure prototypes
declaring *See* EXTERNDEF directive
defined with *See* PROTO directive
defined with PROTO directive *See* PROTO directive
writing *See* PROTO directive
Procedures
arguments
far pointers 197
near addresses 197
passing 182
pointers 80
type conversions 195, 196
CALL instruction 180
calling *See* INVOKE directive
calls
indirect 198
optimizing 181
defining 180
epilogues 26
EXTERNDEF directive 214–215
See also EXTERNDEF directive,
include files 214
INVOKE directive 193–197, 216
libraries 221
local variables 188–192
See also Local variables
Macro *See* Macros, procedures
new features 347

Procedures *(continued)*
OPTION PROC 375
overview 180
parameters
declaring 184–186
variable numbers of 186–188, 194
PROC attributes, specifying 185
prologues 26
PROTO directive 193, 214, 216
See also PROTO directive
prototypes, writing 193
RET instruction 180
RETF instruction 181
RETN instruction 181
syntax description 184
VARARG keyword 186–188, 194
visibility 25, 375
Processors
See also Real mode; Protected mode
8086-based 2–3
.MODEL directive 37
modes, determining 10
target 2
timing xvii, 399–400
Product assistance xx
Program Segment Prefix (PSP) 56
Programming, MASM 6.1 practices 352
Programs
exiting 41
mixed-language 307
starting 41
PROLOGUE argument, OPTION directive 25, 201–203
Prologue code
arguments, specifying 185
code labels in 357
defined 198
macros for 201–203, 264–265
standard 199
user-defined 26, 201
Protected mode
described 2–7, 335
flat model 335
read-only segments 45
PROTO directive
include files 211, 214–216
procedure prototypes, defined with 193
procedure prototypes, writing 312
Prototypes
procedure
directives for 193
overview 193
qualifiedtypes, defined with 15
PTR operator
example 92

PTR operator *(continued)*
OPTION M510 behavior 365
pointer to type, as 15
signed number, specifying 178
size 66, 88
TYPEDEF, used with 75
PUBLIC combine type 45
PUBLIC directive 185, 211, 220
PUSH instruction 49, 71
PUSHA instruction 74
PUSHAD instruction 74
PUSHCONTEXT directive 255, 349
PUSHF instruction 73
PUSHFD instruction 73

Q

Quadwords 86
Qualifiedtypes
BNF grammar 16
defined 15
pointers, defining 75–76
prototypes, as 15
rules for use 15–16
Question mark initializer (?)
array elements 109
described 368
variables 87
Quotation marks (' or ") 109
QWORD directive 86

R

.RADIX directive 11
Radix specifiers
(list) 11
OPTION M510 behavior 367
RCL instruction 101–104
RCR instruction 101–104
Read-only code 27
READONLY argument, OPTION directive 26
READONLY operand 44–45
Real mode 2, 4, 7
Real numbers *See* Floating-point
REAL4 directive 136–137
REAL8 directive 136–137
REAL10 directive 136–137
RECORD directive 130–131
Records
defined 129
field ranges 354
LENGTH operator 357
operators 133–134
RECORD syntax 130–131

Records with SIZEOF operator *See* SIZEOF operator
Records with TYPE operator *See* TYPE operator
Recursive macros 255
Register operands 61
Registers
  16-bit 16–17, 67
  32-bit 335
  base 65–70
  coprocessor 140
  copying pairs of 82
  division (table) 98
  Eflags 20
  extended 17
  flags 20
  FS 17
  general purpose 19
  GS 17
  index 65–69
  indirect addressing 65
  indirect operands 67–68
  initializing 44
  Instruction Pointer (IP) 20, 57, 161
  (list) 409
  loading addresses into 80
  mixed 16-bit, 32-bit 70
  pointers as 77
  scaling 67–69
  segment *See* Segment registers
  Stack Pointer (SP) 19
  Stack Segment (SS) 73
  stacks, saving on 74
  types, defined with ASSUME 77
Relational operators (list) 178
Relocatable
  addresses 57
  expressions 62, 65
REP instruction 110–112, 363
REPE instruction 110–112, 363
Repeat blocks 239
.REPEAT directive 173
REPEAT directive 240
REPNE instruction 110–112, 353, 363
REPNZ instruction 110–112, 353, 363
Reporting problems xx
REPZ instruction 110–112
Reserved words
  described 8, 26
  (list) 407
  OPTION M510 behavior 362
  OPTION NOKEYWORD 376
RET instruction
  epilogue code, generating 200, 378
  instruction encodings, changes to 357
  PROC, with 180

RETF instruction 181, 378
RETN instruction 181, 378
ROL instruction 101–104
ROM-BIOS interrupts *See* Interrupts
ROR instruction 101–104
Rotate instructions 101
Routines, interrupt 206

S

SAL instruction 101–104
SAR instruction 101–104
SBB instruction 92–94
SBYTE directive 86
Scaling factor 107
Scaling index registers 67–69
SCAS instruction 110–112, 115, 353
Scope within visibility *See also* Visibility
SCOPED argument, OPTION directive 26
SDWORD directive 86
SEG operator 49, 62, 363
SEGMENT
    FLAT argument, OPTION directive 27
    USE16 argument, OPTION directive 27
    USE32 argument, OPTION directive 27
Segment arithmetic 7
SEGMENT directive 44–47
Segment mode, setting *See* .386 directive; .486 directive
Segment registers
    32-bit 335
    assigning 59, 62
    ASSUME directive 49–55, 58–59, 357
    changing 57
    default 60, 64
    described 18
    FS 18
    GS 18
    initializing 43, 54–57
    MS-DOS, under 24, 43
    near code 57
    restoring 59
    segment-override operator (:) 50, 59–60, 64
Segment registers
    initializing *See* STARTUP directive
    setting *See* STACK directive
Segment selectors 5
Segment-override operator (:) 50, 59–60, 64
Segmented architecture 2, 5
Segments
    32-bit 36, 335
    accessing data 74
    aligning 44–45
    class types 44, 47–48

Segments *(continued)*
code
creating 40
far 40
memory model support for 36
near 40
combining 40, 44–46
current 10
data
creating 39
default 49, 54–55, 59
far 40
memory model support for 36
near 39
defined 31
described 5–7
determining order of 47–48
determining position of 23–24
determining size of 44
fixups for 26
full segment definitions, defining 32
groups, defining 51
initializing 55
location of 6
naming 40
ordering with the linker 48
protection 6
READONLY 45
simplified segment directives 37–42
size, determining 10
types 44
USE16 44
USE32 44
values 55
word size, setting 46
Selector 335
Semicolon (;), comments 21
.SEQ directive 47
SETIF2 argument, OPTION directive 25, 29–30
Shift instructions 100
SHL instruction 101–104
SHORT operator 169
SHR instruction 101–104
Sign-extending integers 90
SIGN? flag as operand 178
Signed data 14, 91
Signed numbers, specifying *See* PTR operator
Significand 139
Simplified segment directives
code segments 41
code, starting and ending 42
data segments 40
described 32
language convention 36

Simplified segment directives *(continued)*
memory model 35
.MODEL, defining with 34
operating system 35
processor 38
segment registers, initializing 54–56
stack 39
stack distance 37
using 33
Single quotation mark (') 109
Size attribute, segments
FLAT 46
USE16 46
USE32 46
Size mismatch 355
Size of strings *See* SIZEOF operator
SIZE operator 364, 365
@SizeStr predefined string function 245–246
SIZEOF operator
arrays, with 108
described 346
records, with 132
strings, with 110
structures, with 124
types 86
unions, with 125
SIZESTR directive 245–246
Small model *See* Memory models, small
Source code, statements in 21
SP (Stack Pointer) register 19, 71–73
SS (Stack Segment) register 73
STACK combine type 45
.STACK directive
described 33
segment registers, setting 56
Stack distance 37
Stack frame 73, 200, 264–265
Stack Pointer (SP) register 19
@stack predefined symbol 37
Stack Segment (SS) register 73
Stacks
cleaning 185
creating 38
described 71
distance 37
far 10
FARSTACK 35, 37
in DLLs 264–267
local variables on 188–191
near 10
NEARSTACK 33, 35–37
operations with 72–74
operators 71
passing arguments 182

Stacks *(continued)*
pointer 71–73
POP instructions 71
PUSH instructions 71
saving flags 73
saving registers 74
segment register 18
separate 46
trace 264
.STARTUP directive
described 33
initializing segments 54–56
program, starting 41–42
segment address 37
Statements
case sensitivity 22
syntax 21
Status flags, saving 73
STC instruction 104
STDCALL calling convention 311, 336
STI instruction 5, 209
STOS instruction 110–113, 353
Strings
declaring 109
defined 105
defining 15
initializing 109
instructions
processing, for 110
requirements (table) 112, 353
length of 110
multiple-line declarations for 109
overview 111
predefined functions for macros 11
See also Predefined string functions
size of 110
type of 110
STRUCT directive 118–129
Structure-member operator (.) 64–67, 126, 352, 370
Structures
alignment of fields 118–119
array initializers 122
arrays 124
compatibility with MASM 5.1 25, 118
current address operator ($) 368
default field values 122
defined 117
fields
accessing 64, 67, 371
initializing 118
naming 119, 352, 372
initializers, as 123
MASM 5.1 behavior 25, 355, 370
memory allocation 117

Structures *(continued)*
nested 128–129
new features 345
operators 124
OPTION M510 behavior 366
OPTION OLDSTRUCTS 370
redeclaration 124, 355
referencing fields in 126
steps for using 118
string initializers 122, 368
syntax
types 118
variables 121
Structures with LENGTHOF operator *See* LENGTHOF operator
Structures with SIZEOF operator *See* SIZEOF operator
Structures with TYPE operator *See* TYPE operator
STUB statement 266
SUB instruction 92–94
Substitution operator (&) 238, 372
SUBSTR directive 245–246
@SubStr predefined string function 245–246
SWORD directive 86
Symbol table, listing files 405
Symbols
declaring public and external 214, 220
external 369
naming 346, 368
predefined 9–11
Symbols, declaring by EXTERNDEF directive *See* EXTERNDEF directive
Syntax, MASM 6.1 statements 21
SYSCALL calling convention 308–311
System date 11
System time 11

T

Tables, lookup 241
Target environment 4
TBYTE directive 86, 159
Terminate-and-Stay-Resident programs *See* TSRs
TEST instruction 167–168
Testing for zero 168
Text delimiters *See* Angle brackets
Text macros *See* Macros, text
TEXTEQU directive
aliases 369
CATSTR, compared with 247
syntax 226
Time, system 11
Timing (cycle/second) xvii, 399–400
Tiny model *See* Memory models, tiny
Trap flag 205

TSRs
active
described 275
interrupt handlers in 275
MS-DOS functions, calling 285
MS-DOS functions, interrupting 286, 302
deinstalling 292, 305
described 273
errors, trapping 288–289
examples
ALARM.ASM 279–280, 284
SNAP.ASM 293–305
existing data, preserving 290, 303
hardware events, auditing 275–276, 299
interrupt handlers 275
monitoring
Critical Error flag 287
system status 277, 300
MS-DOS internal stacks (lists) 286
multiplex interrupt 290, 304
passive 274
Type conversions *See* INVOKE directive
Type of strings *See* TYPE operator
TYPE operator
and OPATTR 252–253
arrays, with 108
compatibility 360, 365
records, with 132
string, with 110
structures, with 124
types 86
unions, with 125
TYPEDEF directive
aliases, created by 87, 137
BNF, from 380
data types, defining 87
indirect operands, defining 163
pointers, defined by 15, 75–78
procedure declarations 193
procedure prototypes 193
qualifiedtypes 16
TYPEDEF, used with PTR operator *See* PTR operator
Types, data *See* Data types

U

Unconditional jumps 162
UNION directive 118–119, 122, 125–129
Unions
arrays as initializers 122
arrays of 124
defined 117
fields 119, 127–129
memory allocation 117
Unions *(continued)*
nested 128–129
operators 125
referencing fields in 126
steps for using 118
strings as initializers 122
types 118
variables 121, 127
Unpacked BCD numbers 160
Unsegmented architecture 5
Unsigned data 91
.UNTIL directive 173
.UNTILCXZ directive 173
USE16 operand 44–46
USE32 operand 44–46
USES in PROC statement 184
Utilities
IMPLIB 258
MASM 342
ML xvi

V

VARARG keyword
macros, used in 242, 249, 351
procedures, used with 186–188, 194
Variables
assembly-time 233
communal 217
environment 10, 213, 222
external 217, 369
floating-point 136–138
global 211
initializing 87
integers, allocating memory for 85–86
local address, loading 82
naming restrictions 9
Virtual memory 5
Virtual-86 mode 2, 335
Visibility
PROC statement 25, 185
scope, within 9

W

WDEB386 debugger 264
WEP (Windows Exit Procedure) 263–264, 270
.WHILE directive 173
WHILE directive 241
WIDTH operator 133
Windows operating system
API 257, 262

applications 258, 261
DLLs 261
Windows operating system *(continued)*
exit codes 263
MS-DOS, compared 4
programming for 4
protected mode 2, 6
SDK 268
task header 265, 269
Windows NT 3–5
WORD align type 45
WORD directive 86
Word size
default 13, 363, 373
expressions, in 13, 26
@WordSize predefined symbol 39
Words, reserved *See* Reserved words

X

XCHG instruction 90
/X command-line option, ML 213
XLAT instruction 116
XLATB instruction 116
XOR instruction 27, 99–100

Z

ZERO? flag as operand 178
/Zm command-line option, ML 62, 119
/Zp command-line option, ML 119