Table of Contents

Chapter 0: Front Matter 1

Dedication 1
Introduction 1
  Who is this book for? 2
  Chapter summaries 3
  The project 4
  Acknowledgements 5
  Contact us 6

Chapter 1: Linking and Loading 7

What do linkers and loaders do? 7
Address binding: a historical perspective 7
Linking vs. loading 10
  Two-pass linking 12
  Object code libraries 15
  Relocation and code modification 17
Compiler Drivers 18
  Linker command languages 19
Linking: a true-life example 20
Exercises 25

Chapter 2: Architectural Issues 27

Application Binary Interfaces 27
Memory Addresses 28
  Byte Order and Alignment 28
Address formation 30
Instruction formats 31
Procedure Calls and Addressability 32
  Procedure calls 33

Data and instruction references 36
IBM 370 37
SPARC 40
SPARC V8 40
SPARC V9 42
Intel x86 43
Paging and Virtual Memory 45
The program address space 48
Mapped files 49
Shared libraries and programs 51
Position-independent code 51
Intel 386 Segmentation 53
Embedded architectures 55
Address space quirks 56
Non-uniform memory 56
Memory alignment 57
Exercises 57

Chapter 3: Object Files 59

What goes into an object file? 59
Designing an object format 60
The null object format: MS-DOS .COM files 61
Code sections: Unix a.out files 61
a.out headers 64
Interactions with virtual memory 65
Relocation: MS-DOS EXE files 72
Symbols and relocation 74
Relocatable a.out 75
Relocation entries 78
Symbols and strings 80
a.out summary 82
Unix ELF 82
Relocatable files 85
ELF executable files 92
ELF summary 94

IBM 360 object format ... 94
ESD records ... 95
TXT records ... 97
RLD records ... 97
END records ... 98
Summary ... 98
Microsoft Portable Executable format ... 99
PE special sections ... 105
Running a PE executable ... 107
PE and COFF ... 107
PE summary ... 108
Intel/Microsoft OMF files ... 108
OMF records ... 110
Details of an OMF file ... 111
Summary of OMF ... 114
Comparison of object formats ... 114
Project ... 115
Exercises ... 117

Chapter 4: Storage allocation ... 119

Segments and addresses ... 119
Simple storage layout ... 120
Multiple segment types ... 121
Segment and page alignment ... 124
Common blocks and other special segments ... 125
Common ... 125
C++ duplicate removal ... 127
Initializers and finalizers ... 130
IBM pseudo-registers ... 131
Special tables ... 134
X86 segmented storage allocation ... 134
Linker control scripts ... 136
Embedded system storage allocation ... 138
Storage allocation in practice ... 138
Storage allocation in Unix a.out linkers ... 139

Storage allocation in ELF ... 141
Storage allocation in Windows linkers ... 144
Exercises ... 146
Project ... 147

Chapter 5: Symbol management ... 149

Binding and name resolution ... 149
Symbol table formats ... 150
Module tables ... 153
Global symbol table ... 154
Symbol resolution ... 157
Special symbols ... 158
Name mangling ... 158
Simple C and Fortran name mangling ... 158
C++ type encoding: types and scopes ... 160
Link-time type checking ... 163
Weak external and other kinds of symbols ... 164
Maintaining debugging information ... 164
Line number information ... 164
Symbol and variable information ... 165
Practical issues ... 166
Exercises ... 167
Project ... 167

Chapter 6: Libraries ... 169

Purpose of libraries ... 169
Library formats ... 169
Using the operating system ... 169
Unix and Windows Archive files ... 170
Unix archives ... 170
Extension to 64 bits ... 174
Intel OMF libraries ... 174
Creating libraries ... 176
Searching libraries ... 177

Performance issues 179
Weak external symbols 179
Exercises 181
Project 181

Chapter 7: Relocation 183

Hardware and software relocation 183
Link time and load time relocation 184
Symbol and segment relocation 185
  Symbol lookups 186
Basic relocation techniques 186
  Instruction relocation 188
    X86 instruction relocation 189
    SPARC instruction relocation 189
  ECOFF segment relocation 191
  ELF relocation 193
  OMF relocation 193
Relinkable and relocatable output formats 194
Other relocation formats 194
  Chained references 195
  Bit maps 195
  Special segments 196
Relocation special cases 197
Exercises 197
Project 198

Chapter 8: Loading and overlays 201

Basic loading 201
Basic loading, with relocation 202
Position-independent code 203
  TSS/360 position independent code 203
  Per-routine pointer tables 206
  Table of Contents 207
  ELF position independent code 208

PIC costs and benefits ... 212
Bootstrap loading ... 213
Tree structured overlays ... 214
Defining overlays ... 217
Implementation of overlays ... 220
Overlay fine points ... 222
Data ... 222
Duplicated code ... 222
Multiple regions ... 223
Overlay summary ... 223
Exercises ... 223
Project ... 224

Chapter 9: Shared libraries ... 227

Binding time ... 230
Shared libraries in practice ... 231
Address space management ... 231
Structure of shared libraries ... 232
Creating shared libraries ... 233
Creating the jump table ... 234
Creating the shared library ... 235
Creating the stub library ... 235
Version naming ... 237
Linking with shared libraries ... 238
Running with shared libraries ... 238
The malloc hack, and other shared library problems ... 240
Exercises ... 243
Project ... 244

Chapter 10: Dynamic Linking and Loading ... 247

ELF dynamic linking ... 248
Contents of an ELF file ... 248
Loading a dynamically linked program ... 253
Starting the dynamic linker ... 253

Finding the libraries ... 254
Shared library initialization ... 255
Lazy procedure linkage with the PLT ... 256
Other peculiarities of dynamic linking ... 258
Static initializations ... 258
Library versions ... 259
Dynamic loading at runtime ... 260
Microsoft Dynamic Link Libraries ... 260
Imported and exported symbols in PE files ... 261
Lazy binding ... 266
DLLs and threads ... 267
OSF/1 pseudo-static shared libraries ... 267
Making shared libraries fast ... 268
Comparison of dynamic linking approaches ... 270
Exercises ... 271
Project ... 271

Chapter 11: Advanced techniques ... 273

Techniques for C++ ... 273
Trial linking ... 274
Duplicate code elimination ... 276
Database approaches ... 278
Incremental linking and relinking ... 278
Link time garbage collection ... 281
Link time optimization ... 282
Link time code generation ... 284
Link-time profiling and instrumentation ... 284
Link time assembler ... 285
Load time code generation ... 285
The Java linking model ... 287
Loading Java classes ... 288
Exercises ... 290
Project ... 291

Chapter 12: References 293

Perl books 295