

目次

序文

第1章 プログラムの実行と終了	1
はじめはhello.c	2
hello.cソースファイルの準備	4
GNU開発ツールの確認	5
gccコマンドによる自動ビルド	6
コマンド探索の仕組み	7
a.outの起動	7
環境変数とシェル変数	9
作業ディレクトリでのプログラム実行	12
ユーザレベルによるPATH環境変数の違い	14
/etc/profileによるPATH環境変数の初期化	18
main関数におけるreturn引数の意味	20
終了ステータスの確認	21
エラーコードとしての終了ステータス	21
固定終了ステータスを返すコマンド	22
/etc/passwdにおける/bin/falseの活用	24
OpenBSDにおける/etc/passwd管理	25
第2章 gccドライバによる自動ビルド	31
11368バイトが意味するもの	32
gccコマンドの正体と中間ファイル	33
-save-tempsオプションによる中間ファイルの保存	33
-vオプションによるビルド作業のモニター	33
gcc内部で実行される6つの処理	35
specsファイルとconfigureオプション	35
ビルド4工程と入出力ファイルの関係	37
fileコマンドによる中間ファイルの分類	38
GCCバージョン2.95/4.04のビルド処理	39
ビルド作業の流れ	42
gccオプションスイッチによるビルド4工程の制御	44
中間ファイルの生成・削除	44
ビルド制御スイッチ	46
動的リンクと静的リンク	47
11368バイト vs 471021バイト	47
実行可能ファイルができるまで	50
第3章 プリプロセス	53
プリプロセッサの起動	54
-Eオプションによるプリプロセス	54
Cコンパイラによるプリプロセス	55
cc1コマンドの格納場所	56
マニュアル操作によるプリプロセス	58
コマンド置換機能の活用	59
GNUCバージョンのマクロ定義	60
インクルードの意義	63
#include <stdio.h>の意味	63
プロトタイプ宣言の効用	65
プロトタイプ宣言の直書き	67

インクルード処理の解析	68
ラインマークの意味	68
インクルード状況の可視化	71
cppコマンドの活用	73
ヘッダーファイル探索パス	74
<stdio.h>の格納場所	74
ユーザによるヘッダーファイル探索パスの制御	77
システムヘッダーファイルとローカルヘッダーファイル	81
定義済みマクロ	84
<hr/>	
第4章 コンパイル	91
Cコンパイラの起動	92
自動コンパイル	92
手動コンパイル	93
標的システムによるアセンブリ言語ソースリストの違い	95
GNU開発ツールと標的システム	98
機種依存性から見た開発ツールの違い	99
<hr/>	
第5章 アセンブル	103
アセンブラの起動	104
手動アセンブル	104
自動アセンブル	105
オブジェクトファイルの中身	107
実行コードの逆アセンブル	109
固定文字列データ	110
セクション構造	111
<hr/>	
第6章 静的リンク	115
hello.oが実行できない理由	117
再配置可能と実行可能の違い	117
プログラムヘッダ	119
未定義シンボル_startとprintf	121
未定義シンボルによるリンクエラー	121
printf関数の格納場所	123
2種類のCライブラリ	123
静的Cライブラリlibc.a	123
printf.oオブジェクトファイル	125
printf.oの抽出	126
手作業による静的リンク	127
リンクの連鎖	130
隠されたGCCランタイムライブラリ	133
__udivdi3, __umoddi3シンボルの正体	133
libgcc.aの格納場所	136
_startシンボル未定義による致命的エラー	137
crtファイル	138
crtファイルの格納場所	139
crtファイルの中身	139
crt1.oのリンク	141
crt1.o, crt1.o, crtn.oのリンク	142
静的リンクの構成要素	144

第7章 動的リンク	147
共有ライブラリ版libc	149
libc.so	149
共有ライブラリ内部のシンボル情報	150
共有型Cライブラリのリンク	151
crtファイルのリンク	153
実行不能のa.out	154
"No such file"エラーの原因	154
プログラムヘッダの異常	154
インタプリタによる実行可能ファイルのロード	156
LD_DEBUG環境変数によるELFローダの確認	158
ELFローダの実体と実行	159
lddコマンド	161
LD_TRACE_LOADED_OBJECTS環境変数	161
ELFローダの指定	162
共有ライブラリの探索機構	164
実験用共有ライブラリのビルド	165
実験用共有ライブラリの呼び出し	166
リンクのライブラリ探索パス	167
ELFローダのライブラリ探索	174
LD_LIBRARY_PATH環境変数	177
/etc/ld.so.cacheキャッシュファイル	180
ld.so.cacheの再構築	182
動的リンクの利点	184
ディスク資源の節約効果	184
ライブラリ更新の容易さ	185
動的リンクのアキレス腱	186
BSDとLinuxにおける危機意識の隔たり	186
第8章 GNU開発ツールの全体像	191
静的リンク	193
プリプロセス	193
コンパイル	195
アセンブル	195
libc.a, libgcc.aのリンク	196
crtファイルのリンク	197
静的リンクにおけるファイルサイズの変遷	198
動的リンク	199
libgcc.aおよびlibc.soのリンク	199
crtファイルのリンク	200
ELFローダの指定	200
動的リンクにおけるファイルサイズの変遷	201
GNU開発ツールのパッケージ別分類	202
マニュアル操作によるビルド手順のまとめ	204
参考資料	208
GNU開発ツール・コマンド一覧	209
索引	213