

Contents

Acknowledgements — v

Preface — vii

Chapter 1 – Introduction — 1

- Terminology — 4
- Short History of EFI — 5
- EFI Becomes UEFI—The UEFI Forum — 6
- PIWG and USWG — 8
- Platform Trust/Security — 11
- Embedded Systems: The New Challenge — 12
 - How the Boot Process Differs between a Normal Boot and an Optimized/Embedded Boot — 13
- Summary — 14

Chapter 2 – Basic UEFI Architecture — 15

- Objects Managed by UEFI-based Firmware — 15
- UEFI System Table — 16
- Handle Database — 16
- Protocols — 18
 - Working with Protocols — 21
 - Multiple Protocol Instances — 21
 - Tag GUID — 21
- UEFI Images — 22
 - Applications — 25
 - OS Loader — 25
 - Drivers — 26
- Events and Task Priority Levels — 27
- Summary — 30

Chapter 3 – UEFI Driver Model — 31

- Why a Driver Model Prior to OS Booting? — 31
- Driver Initialization — 32
- Host Bus Controllers — 33
- Device Drivers — 35
- Bus Drivers — 36
- Platform Components — 38
- Hot Plug Events — 38
 - Pseudo Code — 41
 - Device Driver — 41

Bus Driver that Creates All of Its Child Handles on the First Call to
Start() — 42

Bus Driver that Is Able to Create All or One of Its Child Handles on Each Call
to Start(): — 43

Additional Innovations — 47

Security — 47

Manageability — 48

Networking — 49

Summary — 52

Chapter 4 – Protocols You Should Know — 53

EFI OS Loaders — 55

Device Path and Image Information of the OS Loader — 56

Accessing Files in the Device Path of the OS Loader — 57

Finding the OS Partition — 58

Getting the Current System Configuration — 60

Getting the Current Memory Map — 61

Getting Environment Variables — 62

Transitioning to an OS Kernel — 63

Summary — 63

Chapter 5 – UEFI Runtime — 65

Isn't There Only One Kind of Memory? — 66

How Are Runtime Services Exposed? — 69

Time Services — 70

Why Abstract Time? — 70

Get Time — 70

Set Time — 71

Get Wakeup Time — 72

Set Wakeup Time — 72

Virtual Memory Services — 72

Set Virtual Address Map — 73

ConvertPointer — 73

Variable Services — 74

GetVariable — 74

GetNextVariableName — 75

SetVariable — 75

Miscellaneous Services — 77

Reset System — 78

Get Next High Monotonic Count — 79

UpdateCapsule — 79

QueryCapsuleCapabilities — 80

Summary — **80**

Chapter 6 – UEFI Console Services — 81

Simple Text Input Protocol — **83**

Simple Text Input Ex Protocol — **86**

Simple Text Output Protocol — **87**

Remote Console Support — **89**

Console Splitter — **92**

Network Consoles — **93**

Summary — **95**

Chapter 7 – Different Types of Platforms — 97

Summary — **110**

Chapter 8 – DXE Basics: Core, Dispatching, and Drivers — 111

DXE Core — **112**

Hand-Off Block (HOB) List — **114**

DXE Architectural Protocols — **115**

EFI System Table — **117**

EFI Boot Services Table — **118**

EFI Runtime Services Table — **119**

DXE Services Table — **119**

Global Coherency Domain Services — **120**

GCD Memory Resources — **120**

GCD I/O Resources — **122**

DXE Dispatcher — **123**

The *a priori* File — **125**

Dependency Grammar — **125**

DXE Drivers — **126**

Boot Device Selection (BDS) Phase — **127**

Console Devices — **128**

Boot Devices — **129**

Boot Services Terminate — **129**

Summary — **130**

Chapter 9 – Some Common UEFI and PI Functions — 131

Architectural Protocol Examples — **132**

CPU Architectural Protocol — **133**

Real Time Clock Architectural Protocol — **135**

Timer Architectural Protocol — **135**

Reset Architectural Protocol — **136**

Boot Device Selection Architectural Protocol — **137**

Variable Architectural Protocol	138
Watchdog Timer Architectural Protocol	138
PCI Protocols	139
PCI Host Bridge Resource Allocation Protocol	139
PCI Root Bridge I/O	143
PCI I/O	145
Block I/O	147
Disk I/O	149
Simple File System	150
EFI File Protocol	151
Configuration Infrastructure	152
Using the Configuration Infrastructure	153
Driver Model Interactions	154
Provisioning the Platform	155
Summary	156
Chapter 10 – Platform Security and Trust	157
Trust Overview	157
Trusted Platform Module (TPM) and Measured Boot	160
What Is a Trusted Building Block (TBB)?	163
What Is the Point of Measurements?	168
UEFI Secure Boot	169
UEFI Executable Verification	170
UEFI Networking	173
UEFI User Identification (UID)	176
Hardware Evolution: SRTM-to-DRTM	177
Platform Manufacturer	178
Vulnerability Classification	180
Roots of Trust/Guards	180
Summary	181
Chapter 11 – Boot Device Selection	183
Firmware Boot Manager	185
Related Definitions	188
Globally-Defined Variables	188
Default Behavior for Boot Option Variables	191
Boot Mechanisms	191
Boot via Simple File Protocol	192
Boot via LOAD_FILE Protocol	193
Summary	194

Chapter 12 – Boot Flows — 195

Defined Boot Modes — 196

Priority of Boot Paths — 196

Reset Boot Paths — 198

Intel® Itanium® Processor Reset — 198

Non-Power-On Resets — 199

Normal Boot Paths — 199

Basic G0-to-S0 and S0 Variation Boot Paths — 200

S-State Boot Paths — 200

Recovery Paths — 201

Discovery — 201

General Recovery Architecture — 202

Special Boot Path Topics — 203

Special Boot Paths — 203

Special Intel Itanium® Architecture Boot Paths — 203

Intel Itanium® Architecture Access to the Boot Firmware Volume — 203

Architectural Boot Mode PPIs — 207

Recovery — 207

Discovery — 208

Summary — 208

Chapter 13 – Pre-EFI Initialization (PEI) — 209

Scope — 209

Rationale — 210

Overview — 210

Phase Prerequisites — 212

Temporary RAM — 212

Boot Firmware Volume — 212

Security Primitives — 213

Concepts — 213

PEI Foundation — 213

Pre-EFI Initialization Modules (PEIMs) — 214

PEI Services — 215

PEIM-to-PEIM Interfaces (PPIs) — 215

Simple Heap — 216

Hand-Off Blocks (HOBs) — 216

Operation — 217

Dependency Expressions — 218

Verification/Authentication — 219

PEIM Execution — 219

Memory Discovery — 219

Intel® Itanium® Processor MP Considerations — 220

Recovery —	220
S3 Resume —	221
The “Terse Executable” and Cache-as-RAM —	222
Example System —	223
Summary —	226

Chapter 14 – Putting It All Together—Firmware Emulation — 227

Virtual Platform —	228
Emulation Firmware Phases —	230
Hardware Pass-Through —	235
Summary —	236

Chapter 15 – Reducing Platform Boot Times — 237

Proof of Concept —	240
Marketing Requirements —	241
What Are the Design Goals? —	242
Platform Policy —	242
What Are the Supported OS Targets? —	243
Do We Have to Support Legacy Operating Systems? —	243
Do We Have to Support Legacy Option ROMs? —	243
Are We Required to Display an OEM Splash Screen? —	244
What Type of Boot Media Is Supported? —	244
What Is the BIOS Recovery/Update Strategy? —	245
When Processing Things Early —	245
Is There a Need for Pre-OS User Interaction? —	246
Additional Details —	246
Adjusting the BIOS to Avoid Unnecessary Drivers —	246
What Is the Boot Target? —	247
Steps Taken in a Normal and Optimized Boot —	247
Loading a Boot Target —	248
Organizing the Flash Effectively —	249
Minimize the Files Needed —	249
Summary —	250
The Primary Adjustments —	250
Suggested Next Steps —	251

Chapter 16 – Embedded Boot Solution — 253

CE Device Landscape —	253
CE Device Boot Challenges —	254
In-Vehicle Infotainment —	256
Other Embedded Platforms —	257
Generic Requirements —	258

Boot Strategies —	259
Power Management —	261
Boot Storage Devices —	261
Security —	263
Manageability —	267
Summary —	268

Chapter 17 – Manageability — 269

Overall Management Framework —	269
Dynamic In-Band —	271
Out-of-Band —	271
Distributed Management Task Force (DMTF) —	271
UEFI Error Format Standardization —	272
UEFI Error Format Overview —	276
Error Record Types —	276
Windows Hardware Error Architecture and the Role of UEFI —	277
Technology Intercepts: UEFI, IPMI, Intel® AMT, WS-MAN —	281
Intelligent Platform Management Interface (IPMI) —	281
Intel® Active Management Technology (Intel AMT) —	283
Web Services Management Protocol (WS-MAN) —	285
Other Industry Initiatives —	285
The UEFI/IPMI/Intel® AMT/WS-MAN Bridge —	286
IPMI Error Records to UEFI —	287
UEFI Error Records to IPMI —	287
Intel® AMT and IPMI —	287
Future Work —	288
Configuration Namespace —	288
Namespace Entries —	292
Summary —	293

Appendix A – Data Types — 295

Appendix B – Status Codes — 297

Index — 301