EFI Update & EFI Application Toolkit

Providing a jump start to EFI application development and a uniform pre-boot environment

Yosi Govezensky
Program Manager
Microcomputer Software Labs
Intel Corp.

February 15-17, 2000
Agenda

- EFI Application Toolkit
  - Objectives & Components
  - Development Model
  - Timeline

- EFI Update
EFI Overview

- Interface specification
  - Implementation agnostic
- Abstracts BIOS from OS
  - Decouples development
- Compatible by design
  - Evolution, not revolution
- Modular and extensible
  - OS-Neutral value add
- Complements existing interfaces

Flexible to meet existing and future needs
Toolkit Objectives

- Enable rapid application development for EFI
  - Help reduce the porting effort
- Create rich pre-boot environment industry can count on
- Enable 3rd party market
  - e.g. platform utilities
- Allow industry contribution
- Uniform environment between IA-32 and IA-64

Jump start EFI application development
Toolkit Components

- EFI shell
- Common Utilities
- Libc
- Remote Debugger
- Network Stack
- Multiple Processor Test Support
- Platform Management
- Security Service

Sources are provided

Useful tools for EFI application development
Toolkit Components

EFI Shell
Utilities
Libc
Network
MP Test
Management
Security

EFI API

EFI

Hardware

Intel

Labs
EFI Shell

- Interactive console interface
- Application launch
- Load EFI protocols and drivers
- Simple scripting capability
- Automatic execution of startup batch file
- Console redirection to files
Common Utilities

- Core utilities
  - `cp, rm, mkdir, type, …`

- Extended utilities
  - Network utilities
    - FTP client and server, ping
  - Text editor

- Scripting interpreter (Python)
Libc

- FreeBSD port
- System I/O - open(), read(), write(), close(), stat(),
- Standard I/O - fopen()…, printf()…, gets()…
- String/Char - strcmp()…, isupper()…, atoi()…,
- Memory - malloc(), calloc(), realloc(), free()
- Time/Date - time()…, asctime()…, ctime()…
- Math - sqrt(), pow(), sin(), cosin(), log(),…
Remote Debugger

- Enable debugging
  - EFI applications, drivers, and protocols
- User interface
  - Provided by “commercial” debugger
  - Requires debugger with remote debug capability
- Toolkit agent
  - Can be customized for remote protocols
  - Intel Enhanced Debugger (EDB) reference
    - EDB is included in the IA-64 SDK

"Firmware” source debug
Network Stack

- Port of FreeBSD TCP/IP stack
- Supports standard protocols
  - IPv4, ICMP, ARP, UDP, TCP
- Socket library interface
- Implemented as an EFI protocol

Network functionality in FW
MP Test Support

- Execute non-EFI based code on non-boot processors
  - Primarily used for diagnostics and manufacturing test
  - Cannot make use of EFI services on non-boot processors in first release

- Inter-processor synchronization functions
Platform Management

- SMBIOS Interface
  - Library routines for parsing SMBIOS tables

- IPMI 1.0 Interface
  - EFI protocol for sending and receiving IPMI messages
**Toolkit Components**

**Security Service**

- Limited but extensible platform security
  - Service that application may choose to use
  - Common security interface
  - Customizable security policies

- Provide reference implementation
  - Unix style password file
  - One-way encryption
    - Given user and password.. “is this user authenticated?”

---

Rudimentary application security
Development Model

- Toolkit adopters are free to make source code changes
  - There are no restrictions on differentiation

- Collaboration
  - We will incorporate changes that are voluntarily returned and benefit industry as a whole
  - There is a desire to keep the toolkit environment uniform

- New releases to be posted on the web

Continuously improve, while maintaining basic compatibility
Toolkit Timeline

1999
Dec
EFI Shell and Utilities

Feb
Libc, Network Stack, Platform Management, and Python

2000
Apr
Remote Debugger

Jun
MP Test Support

Aug
Security Service And Updated Extended Utilities

Incremental releases

Tools to help you are ready now
Agenda

- EFI Application Toolkit
  - Objectives & Components
  - Development Model
  - Timeline

- EFI Update
EFI Update

1999
Dec
Feb
2000
Q2
Q3
Q4

Key Milestones

- 0.92 review ends
- 0.92 Spec
- 1.0 Spec
- 1.01 Spec
- 1.1 Spec
- ...And future revs as needed

- Specification
  - 0.92 is “Release Candidate” (RC) for 1.0

- Reference Implementation
  - To be posted following specification
EFI Specification – Next

- Version 1.01
  - Graphics Protocol (bitblit)
  - Pointer Protocol (mouse)
  - Debug Protocol
- Version 1.1
  - Architecture independent PCI Option ROM

EFI continues to evolve
PCI Option ROM

- Already in 0.92:
  - EFI Driver as PCI Option ROM image
  - Multiple formats in a single Option ROM

- Goals for 1.1
  - CPU architecture neutral
  - One C source – multiple targets
  - Regular EFI Driver
  - Small image size
  - Small interpreter in Firmware

One IHV SKU fits all
Summary

- EFI continues to evolve
- Toolkit provides useful tools
- Development model
  - Continuously improve while maintaining basic compatibility
- Tools to help you are ready now
EFI on the Web

- EFI Homepage
  - register for EFI mailing list
  - provide feedback on the specification
  - sample implementation, EFI toolkit and docs

- EFI FAT32 and PE/COFF Image
  - http://www.microsoft.com/hwdev/specs/
Call for Action

- Download latest EFI Specification
- “Pass the word” on the toolkit…
  - Make sure EFI developers in your company are aware of the toolkit
  - Intended to speed up development
- Feedback on EFI applications plans
  yosi.govezensky@intel.com