EDK I Build
EFI Shell and Debug Lab

Intel Corporation
Software and Services Group

Agenda

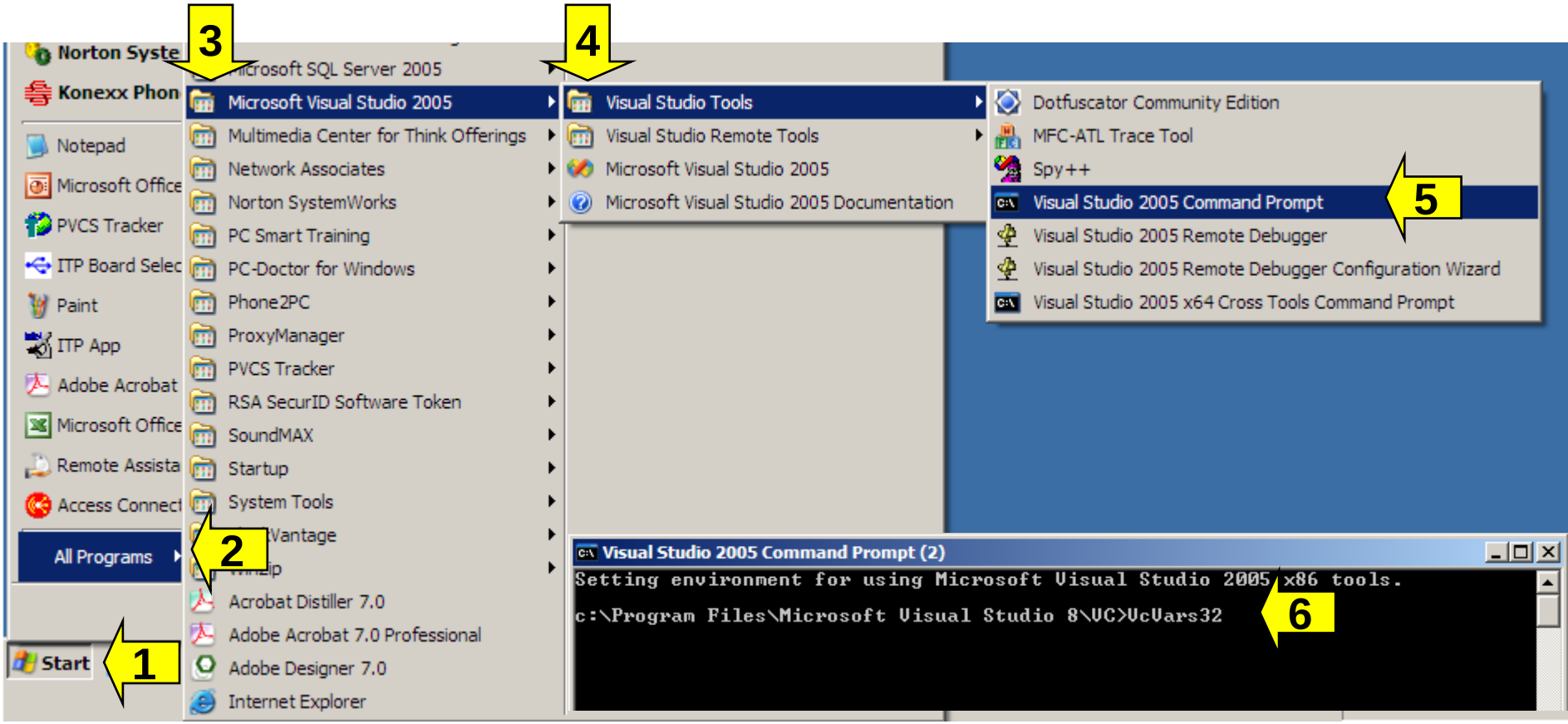
- Generating the Build EDK Lab
- EFI Shell Basics
- Build Results Lab
- EFI Shell – Command Line Tools Lab

NT32 Build Lab

- System requirements:
  - Microsoft Windows XP
  - 256MB+ System Memory
  - 500MB+ Free Space on Hard Drive
  - Visual Studio 2005 (Ver 8.0) Professional or VS .NET 2003
- Create a directory on C: \Fw
- Download the EDK and EFI-Shell from Tianocore.org
  - Go to Documents & files
    - Releases
      - Official releases and download the latest Release .zip
- CD Directory NT32BuildLab contains EDK and EFI-Shell
  - Unzip the “EDK_1.05.Zip” to C:\Fw
  - Unzip the “EfiShell 1.05.Zip” to C:\Fw\Edk\Other\Maintained\Application

Compiler Environment

- Use the Visual Studio command prompt to setup the proper compiler environment
- Run VcVars32 at the CMD Window with VS 2005



NT32 Build Lab

- `CD C:\FW\Edk\Sample\Platform\Nt32\Build`
- Edit the Config.env file for VS 2005 Only change the following line to “YES”
  - USE_VC8 = YES
- Add the EFI Shell to the Build
  - `Edit or Notepad` C:\Fw\Edk\Sample\Platform\Nt32\Build\nt32.dsc check
    - Add the following line in the Shell EFI Library section under “[Libraries.Platform]”
      Other\Maintained\Application\Shell\Library\EfiShellLib.inf
    - Search for other\Maintained\Application\UefiShell\Bin\Shell.inf replace with (or comment)
      #Other\Maintained\Application\UefiShell\Bin\Shell.inf
      Other\Maintained\Application\Shell\Shellfull.inf
  - or copy from CD Nt32BuildLab\Nt32LAB.dsc to C:\Fw\Edk\Sample\Platform\Nt32\Build\nt32.dsc
- Build the NT32
  - `set EDK_SOURCE=C:\FW\EDK`
  - `CD C:\FW\Edk\Sample\Platform\Nt32\`
  - `Build`
- Wait for Build to finish

Agenda

- Generating the Build EDK Lab
- EFI Shell Basics
- Build Results Lab
- EFI Shell – Command Line Tools Lab

EFI Shell Overview

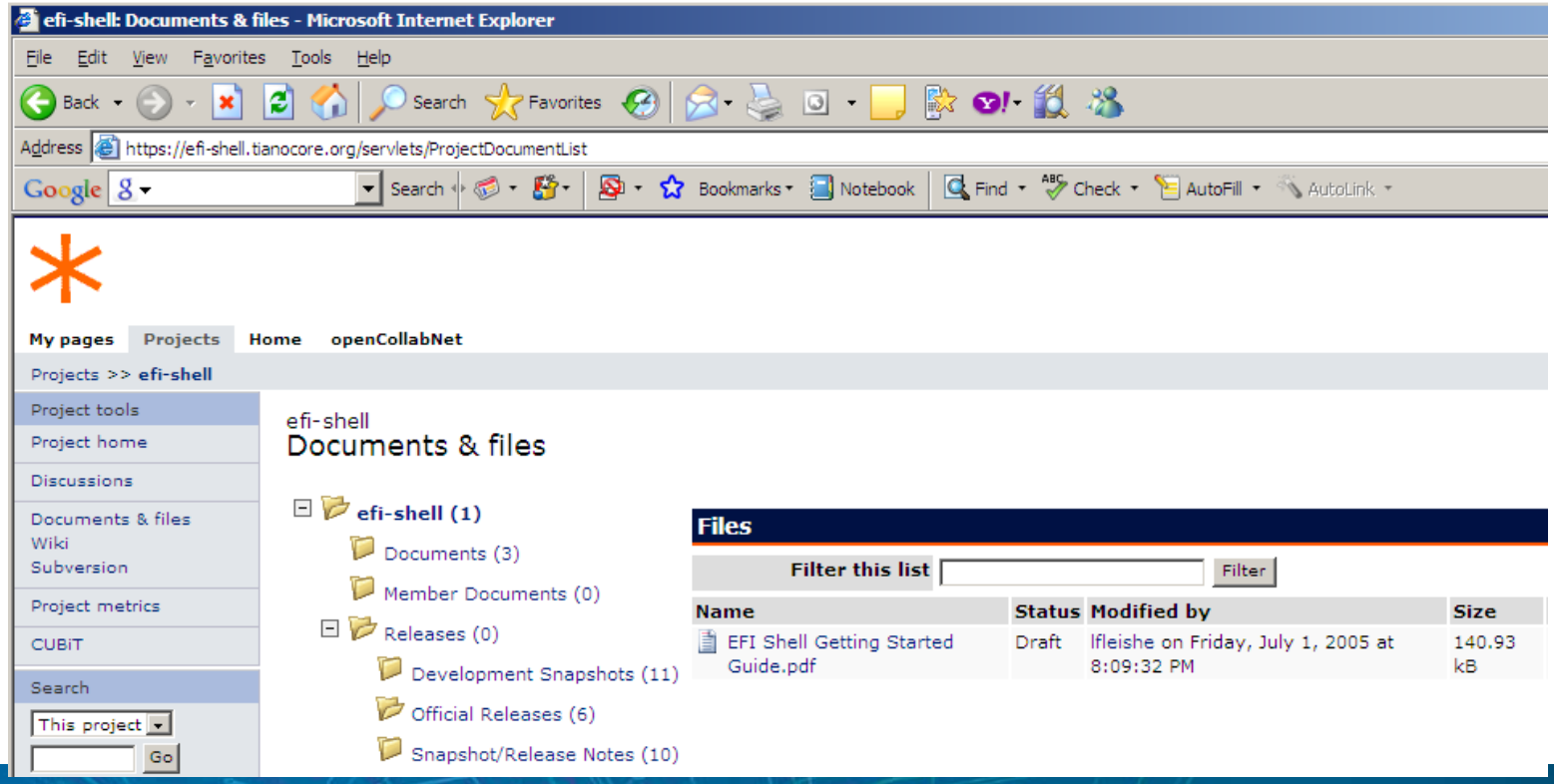
- Interactive way to use UEFI code in system
- Has command line prompt and Scripting
  - Is similar to DOS and Linux* shell but not EXACTLY - its own unique syntax
- Is an EFI executable in itself
- Knows only about EFI file systems that are FATxx
- Shell is a Sub-project on EFI Development Kit (EDK) on the EFI and Framework Open source Community Website
  - Shell programs
  - users' guide
  - EFI Shell Source
- EDK has Binary UEFI Shell for processors IA32, Intel® 64, IA-64

EFI Shell Open Source

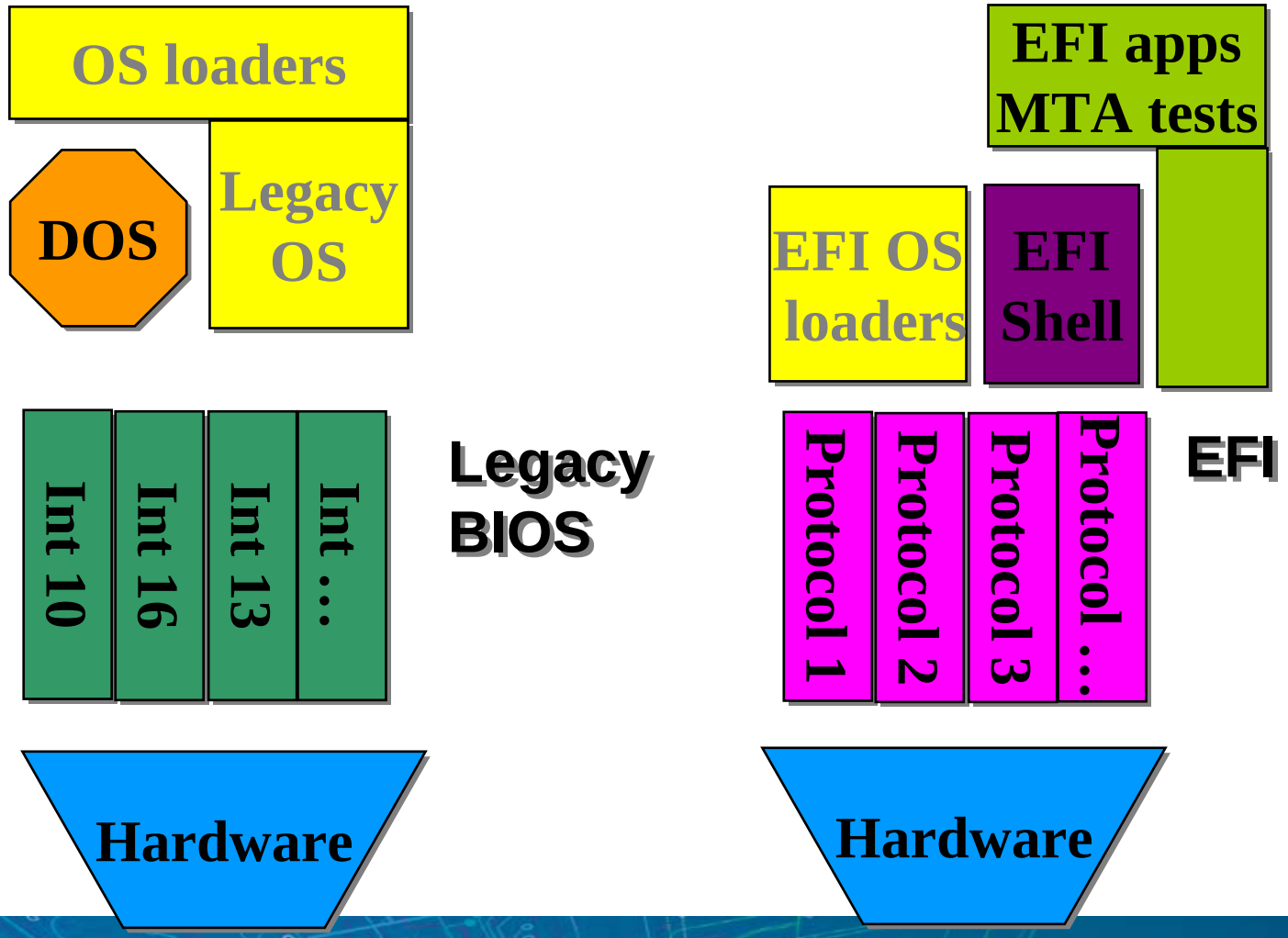
WEB Site: http:\\www.tianocore.org

Project: EFI-Shell

Documents & files : “EFI Shell Getting Started Guide”



Analogy to Old DOS: BIOS



Shell Usage

- Execute preboot programs
  - Setup
  - operating system install
  - Test
  - disk utilities
  - Driver Diagnostics, Configurations
- Move files around between the hard disk, floppy disk, CD-ROM, USB flash devices, and so on
- Load a preboot EFI driver in the system (has an .efi suffix), examples:
  - LAN stack tcpip drivers
  - Update old drivers in flash
  - New drivers for plugin cards
- Shell.efi verses Shell_full.efi
  - Shell.efi smaller to fit in Flash
  - Shell_full.efi Richer commands

EFI File System

Can manipulate EFI system fatxx partition only where boot loader and EFI application are

```
Shell> map
Device mapping table
fs0  : Acpi(PNP0A03,1)/Pci(1F|0)/Pci(2|0)/
 Scsi(Pun0,Lun0)/HD(Part1,Sig8983DFE0-F474-01C2-507B-
 9E5F8078F531)
blk0 : Acpi(PNP0A03,0)/Pci(1F|1)/Ata(Primary,Slave)
blk1 : Acpi(PNP0A03,0)/Pci(1F|1)/Ata(Primary,Master)
blk2 : Acpi(PNP0A03,1)/Pci(1F|0)/Pci(2|0)/Scsi(Pun0,Lun0)
blk3 : Acpi(PNP0A03,1)/Pci(1F|0)/Pci(2|0)/
  Scsi(Pun0,Lun0)/HD(Part1,Sig8983DFE0-F474-01C2-507B-
9E5F8078F531)
blk4 : Acpi(PNP0A03,1)/Pci(1F|0)/Pci(2|0)/
  Scsi(Pun0,Lun0)/HD(Part2,Sig898D07A0-F474-01C2-F1B3-
12714F758821)
blk5 : Acpi(PNP0A03,1)/Pci(1F|0)/Pci(2|0)/
  Scsi(Pun0,Lun0)/HD(Part3,Sig89919B80-F474-01C2-D931-
F8428177D974)
```

Device Path

fs0 :
Acpi(PNP0A03,1)/Pci(1F|0)/Pci(2|0)/Scsi(Pun0,Lun0)/HD(Part1
,
Sig8983DFE0-F474-01C2-507B-9E5F8078F531)

Fs0:

Acpi(PNP0A03,1)
Pci(1F|0)/Pci(2|0)
Scsi(Pun0,Lun0)
HD(Part1,Sig8983DFE0-F474-01C2-507B-9E5F8078F531)

EFI Shell Commands Help ?

dh
help ?
map
mount
load
unload
loadbmp
nshell
ver
memmap
bcfg
Dblk
alias

dmem
dmpstore
err
guid
pci
mm
reset
stall
getmtc
hexedit
Setsize
Set

drivers
devtree
devices
connect
disconnect
openinfo
reconnect
drvcfg
drvdiag
loadpcirom

Main EFI Shell Commands For EFI / UEFI Drivers

- dh
  - Displays handles in the EFI environment
- map
  - Displays or defines mappings
- drivers
  - Displays drivers and attributes in database
- connect / disconnect
  - Start and Stop managing a driver
- drvcfg
  - Run the driver's Configuration program
- drvdiag
  - Run the Driver's Diagnostic program

Agenda

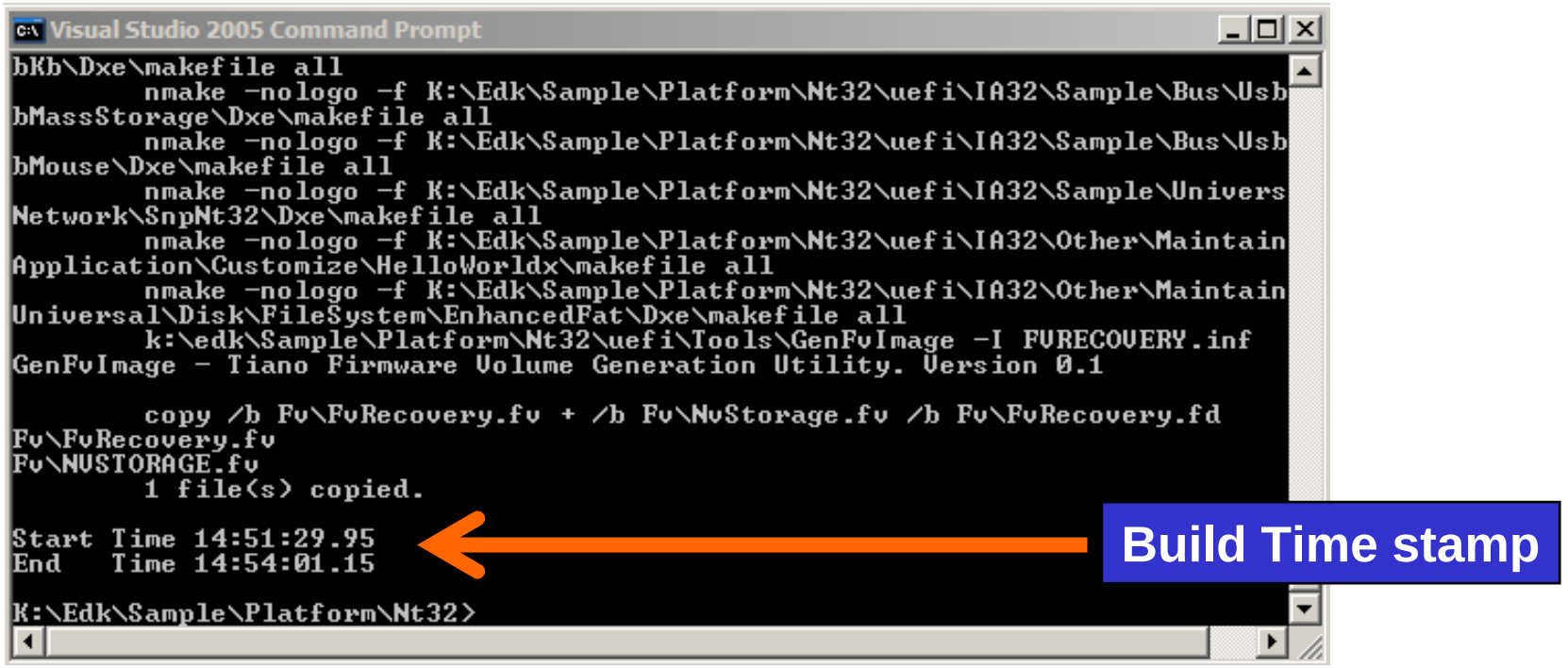
- Generating the Build EDK Lab
- EFI Shell Basics
- Build Results Lab
- EFI Shell – Command Line Tools Lab

Setting Up The Environment

- Local disks & folders can be mounted as file systems under the NT32 environment
- Window names can be modified
- The `System.cmd` in the Build directory Examples:
  - Map a 1.44MB Floppy:
    `set EFI_WIN_NT_PHYSICAL_DISKS=a:RW;2880;512`
  - Set the window name:
    `set EFI_WIN_NT_UGA="UGA Window 1"`
  - Map a system directory as a file system under EFI NT32:
    `set EFI_WIN_NT_FILE_SYSTEM=c:\virtual (needs to a mkdir c:\virtual before you nmake run)`

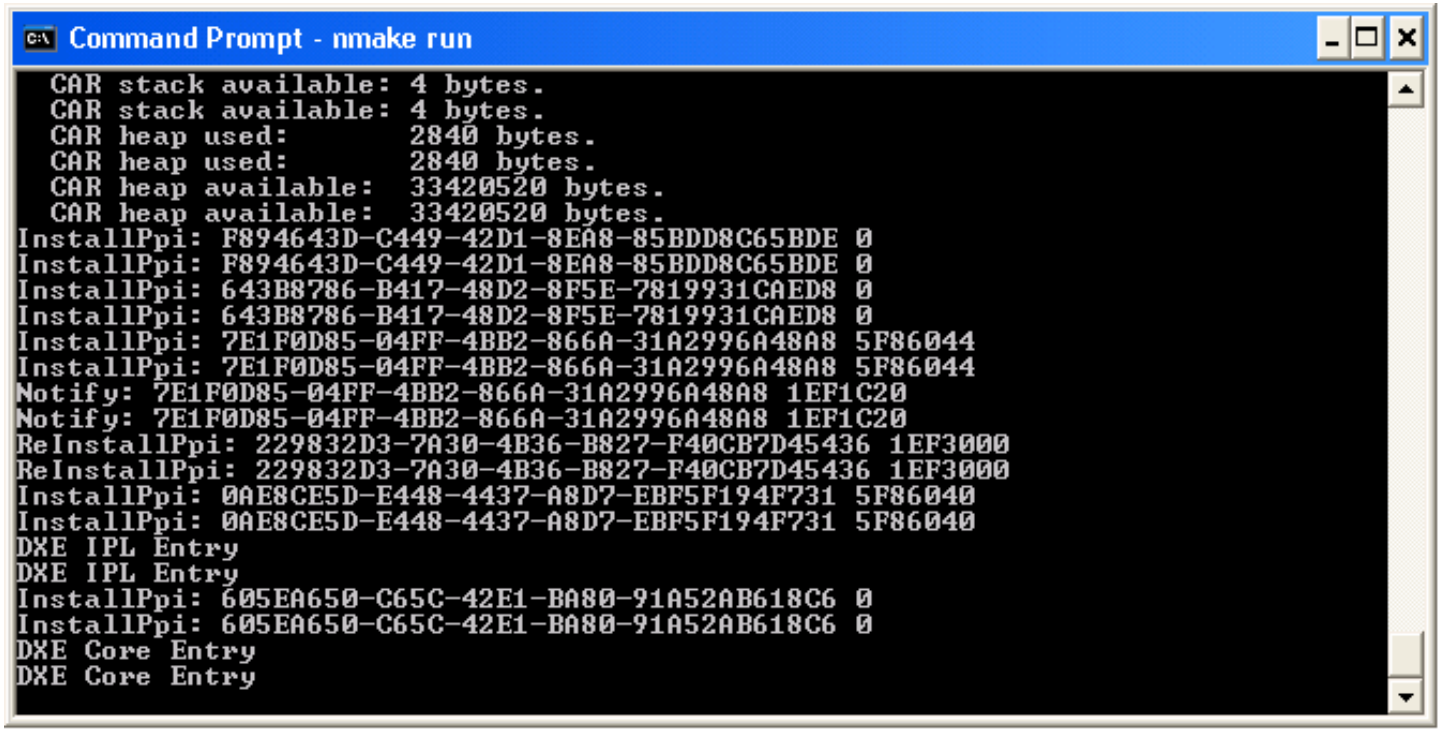
    `Or set it to . to point to the current build dir where all the .efi files are at. This will allow you to run the .efi files from FS0:`
- Run `System.cmd` before running `nmake run` to get file systems

NT32 Build Lab Continued



- After BUILD is complete
  - CD Uefi
  - System
  - Nmake Run

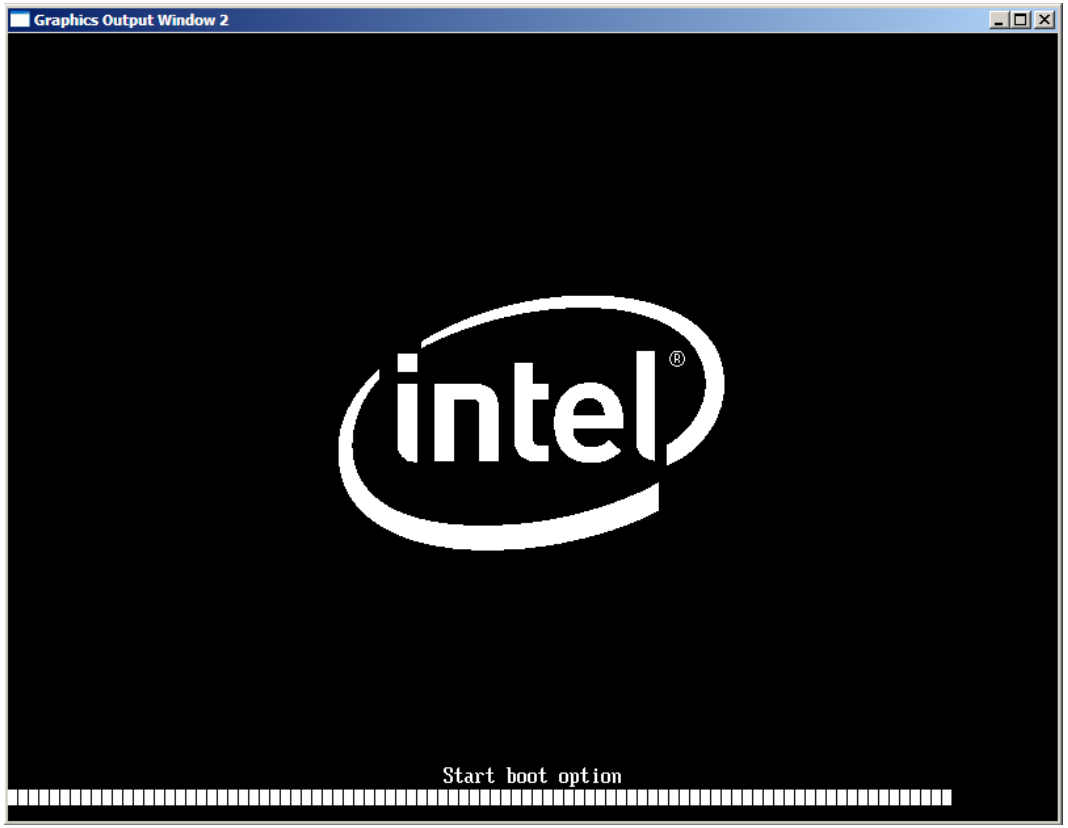
nmake run



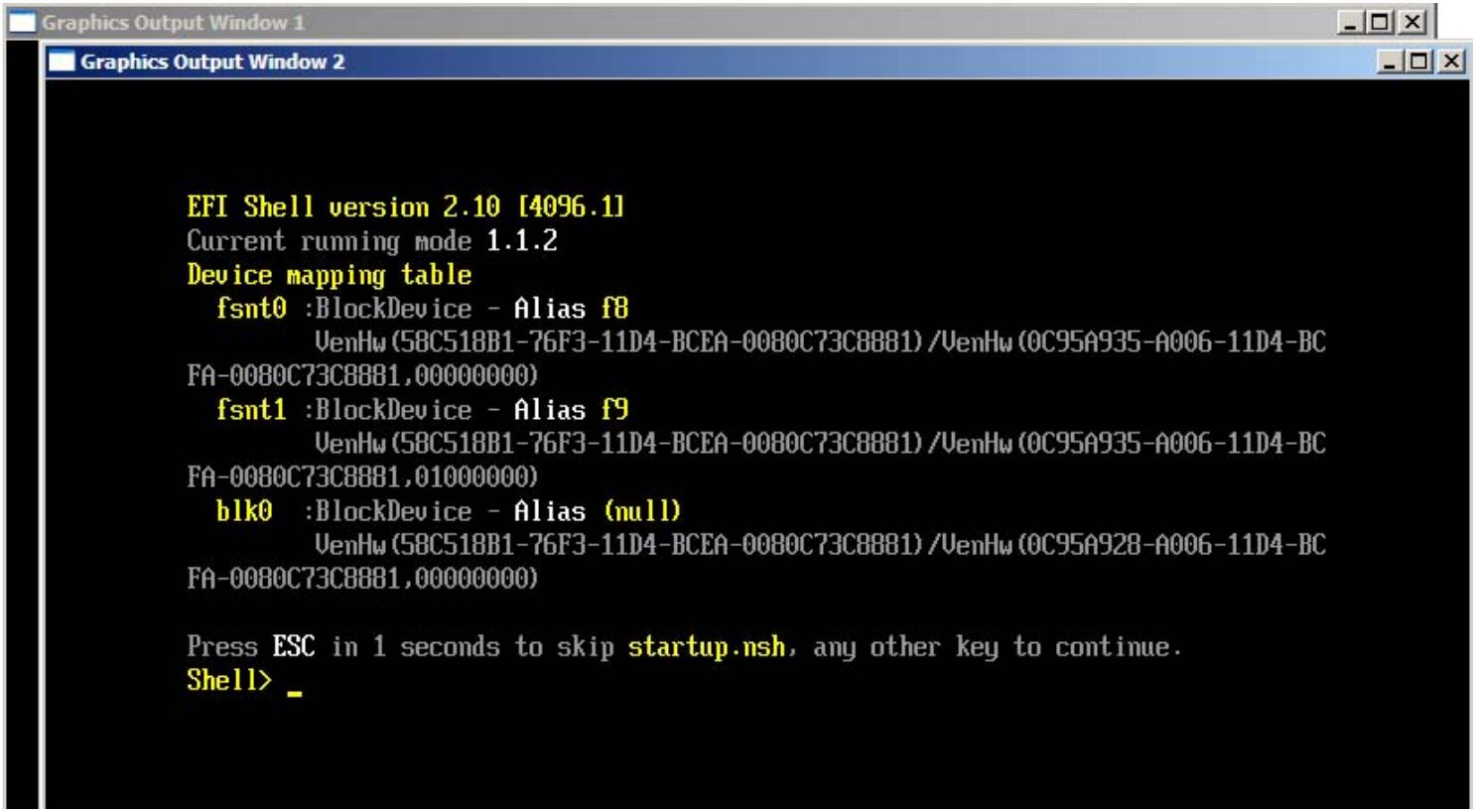
- Start Emulation
  - Emulation runs as Windows application
  - Reset vector code for desktop/server replaced with Windows application startup code
  - Startup code initializes framework and calls dispatcher to start loading drivers

NT32 Emulated Boot

- Splash screen displayed
  - Windows drivers simulate output to screen, input from keyboard and disk operations to Hard Drive



Boot to the EFI Shell



- Load the EFI Shell
- Switch to fsnt0: or F8: for the local disk
  - Mapped to `EFI_WIN_NT_FILE_SYSTEM`

Agenda

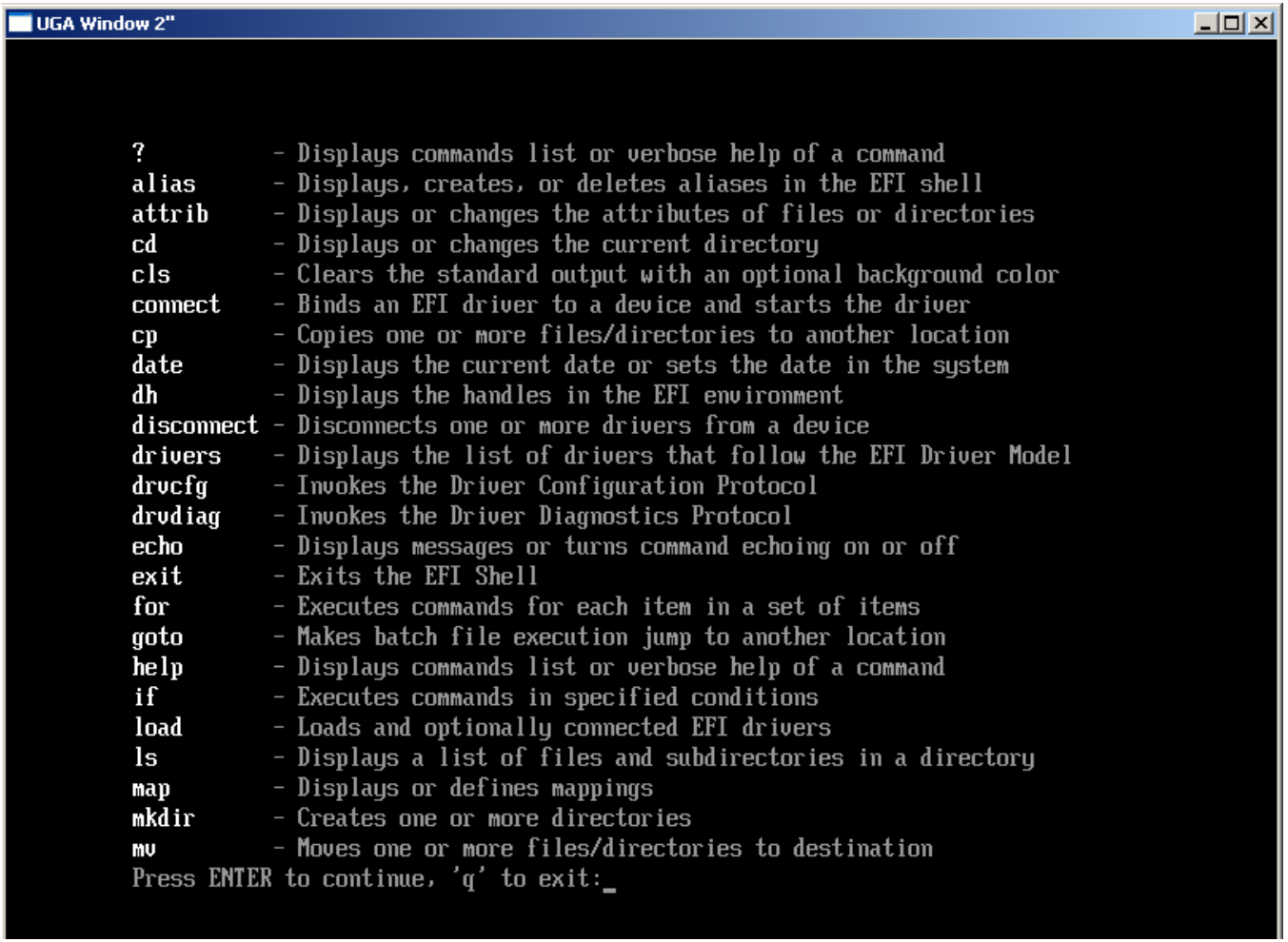
- Generating the Build EDK Lab
- EFI Shell Basics
- Build Results Lab
- EFI Shell – Command Line Tools Lab

EFI Shell Command

- The following EFI Shell commands are useful for Debugging EFI applications
  - Help
  - MM
  - Mem
  - Memmap
  - Drivers
  - Devices
  - Devtree
  - Dh
  - Load

EFI Shell Help

- Help -b



mm – Displays or Modifies memory, I/O or PCI resources

Usage:

MM Address [Width 1|2|4|8] [;MMIO | ;MEM | ;IO | ;PCI] :Value –n

Address	- Starting address for MMIO, MEM, IO or PCI
Width	- Size accessed in bytes (1, 2, 4 or 8)
;MMIO	- MMIO Range (0 – 0xFFFFFFFF_FFFFFFFF)
;MEM	- Memory Range (0 – 0xFFFFFFFF_FFFFFFFF)
;IO	- IO Address Range (0 – 0xFFFF)
;PCI	- PCI Config Address (0x000000ssbbddffrr) ss = SEG, bb = BUS, dd = DEV, ff = FUNC, rr = REGISTER
Value	- Value to write
-n	- Non-interactive mode

mm – Displays or Modifies memory, I/O or PCI resources

continued

Notes:

1. **MEM type is the default**
2. **In interactive mode type a hex value to modify, enter ‘q’ or ‘.’ to exit.**
3. **Use the PCI command to discover the PCI device before using MM to modify PCI configuration space.**
4. **Use ‘-n’ mode inside of shell script files (*.nsh)**
5. **Not all PCI registers are writable. PCI option will not do read-modify write. MM will only write the value posted.**

mem – Displays Contents of System Memory or Device Memory

Usage:

MEM [-b] [address] [size] [;MMIO]

-b	- Display one screen at a time
address	- Starting Address (hex) to display. This needs to be an even address boundary for the processor the command is run on.
size	- Number of bytes to display (hex)
;MMIO	- Memory mapped I/O. Turns on any bits required to force memory access across out to the PCI bus

Note: **Run w/o args to see the system table entry pointer and all other system table pointer addresses.**

memmap – Displays Memory Map Maintained by EFI

Usage:

MEMMAP [-b]

-b - Display one screen at a time

Note:

Use the EFI specification to lookup the memory type.

drivers – Displays List of EFI Drivers

Usage:

DRIVERS [-b] [-lXXX]

-b - Displays one screen at a time

-lXXX - Displays drivers using the ISO 639-2 language specified by XXX

Note:

Run DRIVERS /? to see the display format

devices – Display List of Devices Managed by EFI drivers

Usage:

DEVICES [-b] [-lXXX]

-b - Displays one screen at a time

-lXXX - Displays devices using the ISO 639-2 language specified by XXX

Note:

Run DEVICES /? to see the display format

devtree – displays Tree of Devices Follow the EFI Driver Model

Usage:

DEVTREE [-b] [-d] [lXXX] [DeviceHandle]

-b	- Displays one screen at a time
-d	- Displays device tree using device paths
-lXXX	- Displays drivers using the ISO 639-2 language specified by XXX
DeviceHandle	- Displays device handle below a certain handle

dh – Displays handles in EFI

Usage:

DH [-b] [-d] [-lXXX] [-v] [handle] [-p prot_id]

-b	- Displays one screen at a time
-d	- Displays device tree using device paths
-lXXX	- Displays drivers using the ISO 639-2 language specified by XXX
-v	- Dumps information on all handles
handle	- Dumps information on a specific handle
-p prot_id	- Dumps all handles of a certain protocol

load – Load EFI Drivers

Usage:

LOAD [-nc] file [file…]

-nc	- Load the driver, but do not connect the driver
file	- File that contains the EFI driver (.efi extension)

Note:

- LOAD can handle multiple files & supports wildcards
- Use the ‘UNLOAD’ command to unload an EFI driver

stall – Stall the Processor

Usage:

STALL microseconds

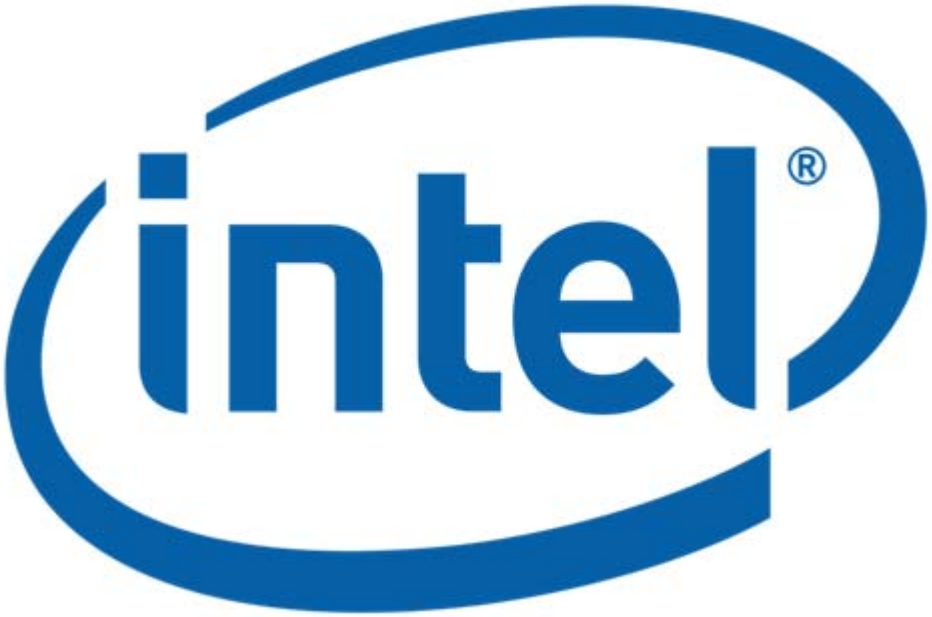
Note:

- The ‘microseconds’ value is in decimal.
- STALL will cause the emulation environment to ‘sleep’ for the specified period.

Additional Resources

- Class CD - Documents
  - Framework R8.x Build.pdf describes build tools
- https://www.TianoCore.org
  - Website for EFI open source resources
    - EFI Developer Kit (EDK)
      - Nt32 emulation environment
    - EFI shell https://efi-shell.tianocore.org/
    - EFI Documentation: https://efi-shell.tianocore.org/servlets/ProjectDocumentList?folderID=50&expandFolder=50&folderID=0

Q & A



Back up