Secure from the START

phoenix technologies

Phoenix PSI SDK/DDK Overview

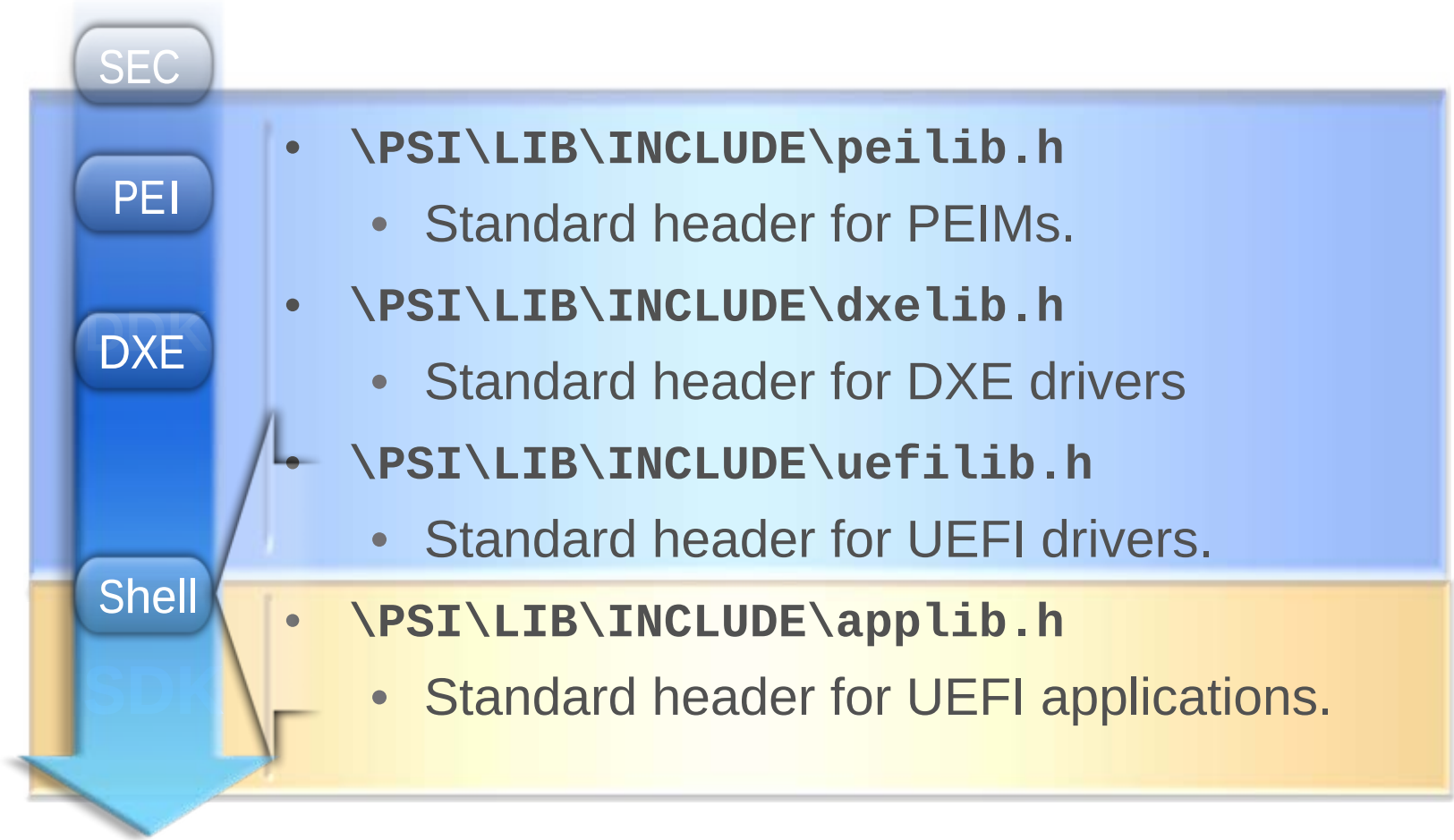
Tim Lewis, Nill Ge

16 March 2007

Agenda

- Phoenix Standard Header Files
- C Library Functions
- EFI Support Functions
- Compiler Support
- Property Setting
- Target Files
- Driver And Application Wizards Overview
- Driver Wizard Demo
- Application Wizard Demo
- Summary

Phoenix Standard Header Files



C Library Functions

- assert.h – Debug assert functions
- conio.h – Intrinsics for port I/O and interrupt disable
  - _inp, _outp, _disable, _enable, etc.
- ctype.h/wctype.h – Character classification functions
- math.h – Floating point math functions
- stdarg.h – Functions for processing variable argument
- string.h – String handling functions

C Library Functions (contd.)

- stdio.h – Standard file read/write functions
  - printf, scanf, fopen, fread, etc.
- stdlib.h – Miscellaneous functions.
- time.h – Standard time handling functions
- wchar.h – Wide-character versions of functions.

Easy to port over existing C code to UEFI.

UEFI Support Functions

Device Paths	EfiAppendDevicePath(), EfiAppendDevicePathInstance(), EfiDevicePathToInstance(), EfiDevicePathToHandle(), EfiDuplicateDevicePath(), EfiGetDevicePathSize(), EfiGetNextDevicePathInstance()
Linked Lists	ListGetFirstNode(), ListGetNextNode(), ListInitialize(), ListInsertHead(), ListInsertTail(), ListIsEmpty(), ListIsNodeAtEnd(), ListIsNodeNull(), ListRemoveNode(), ListSwapNodes()
Simple File System	FileSimpleClose(), FileSimpleOpen(), FileSimpleGetDeviceHandle(), FileSimpleGetHandle(), FileSimpleOpenFromPath(), FileSimpleRead(), FileDevicePath()

UEFI Support Functions (contd.)

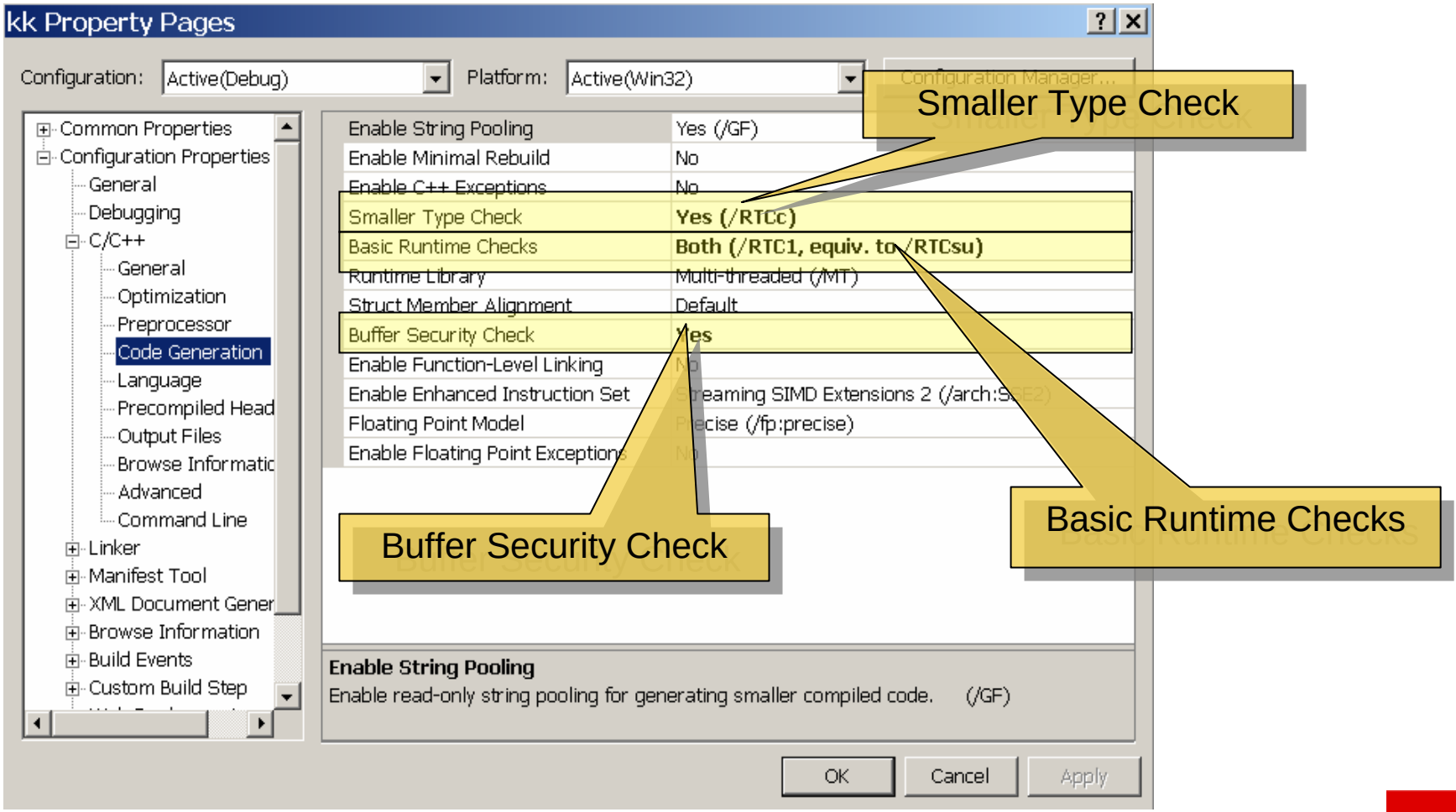
Memory Management	PoolAllocBt(), PoolAllocBtZero(), PoolAllocRt(), PoolDupBt(), PoolDupRt()
CPU	CpuCpuid(), CpuDisableCache(), CpuEnableCache(), CpuGetEflags(), CpuHalt(), CpuInvd(), CpuReadMsr(), CpuReadTsc(), CpuWbinvd(), CpuWriteMsr()
Lock	LockAcquire(), LockAcquireOrFail(), LockInitialize(), LockRelease()
PEI/DXE HOB	PeiBuildHobCpu, PeiBuildHobGuid, GetTimeValue, GetNextGuidHob…

UEFI Support Functions (contd.)

IO/Mem/PCI R/W	IoRead8, MemRead32, PciRead8…
EFI Macros	EFICopyMem, EFISetMem, ,EFIStrCpy, EFILibAllocatePool…
UEFI Global Variables	gST, gBS, gRT, gDS, LoadedImage, GetPeiServices (DDK/SDK will initialize them automatically)
Debug	ASSERT, ASSERT_POINTER, ASSERT_NULL_OR_POINTER DEBUG_ONLY, TRACE, VERIFY, UNUSED_ALWAYS, DbgPrintF, CHECKPOINT

Compiler Support

- Support Of Visual Studio Debug Functions



Compiler Support (contd.)

- Built-In Math Support
- Support For new/delete C++ Operators
- Support Of Visual Studio Performance Enhancement
  - C Compiler Generates Inline Calls To memset.

Tightly integrated with Compiler.

Property Setting

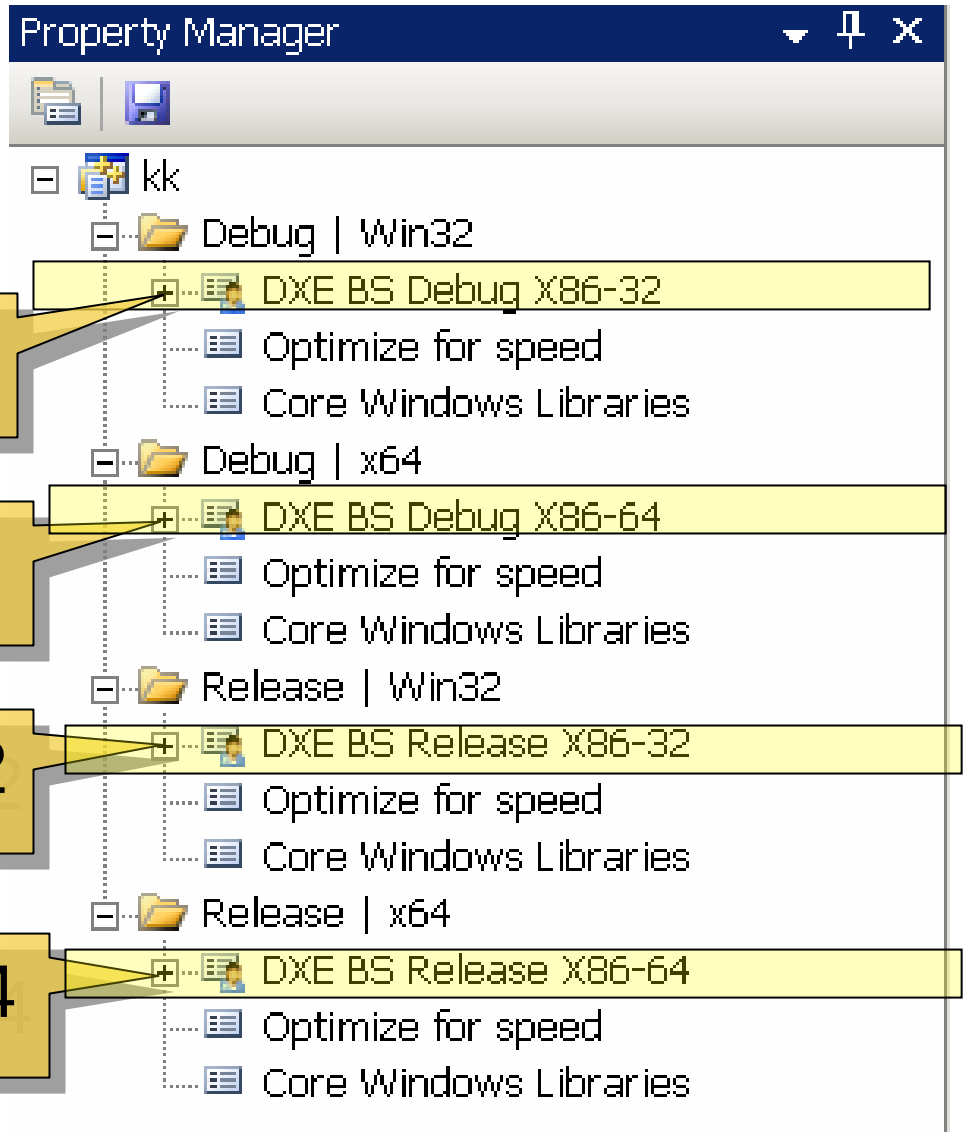
- VSPROPS

DXE BS Debug X86-32

DXE BS Debug X86-64

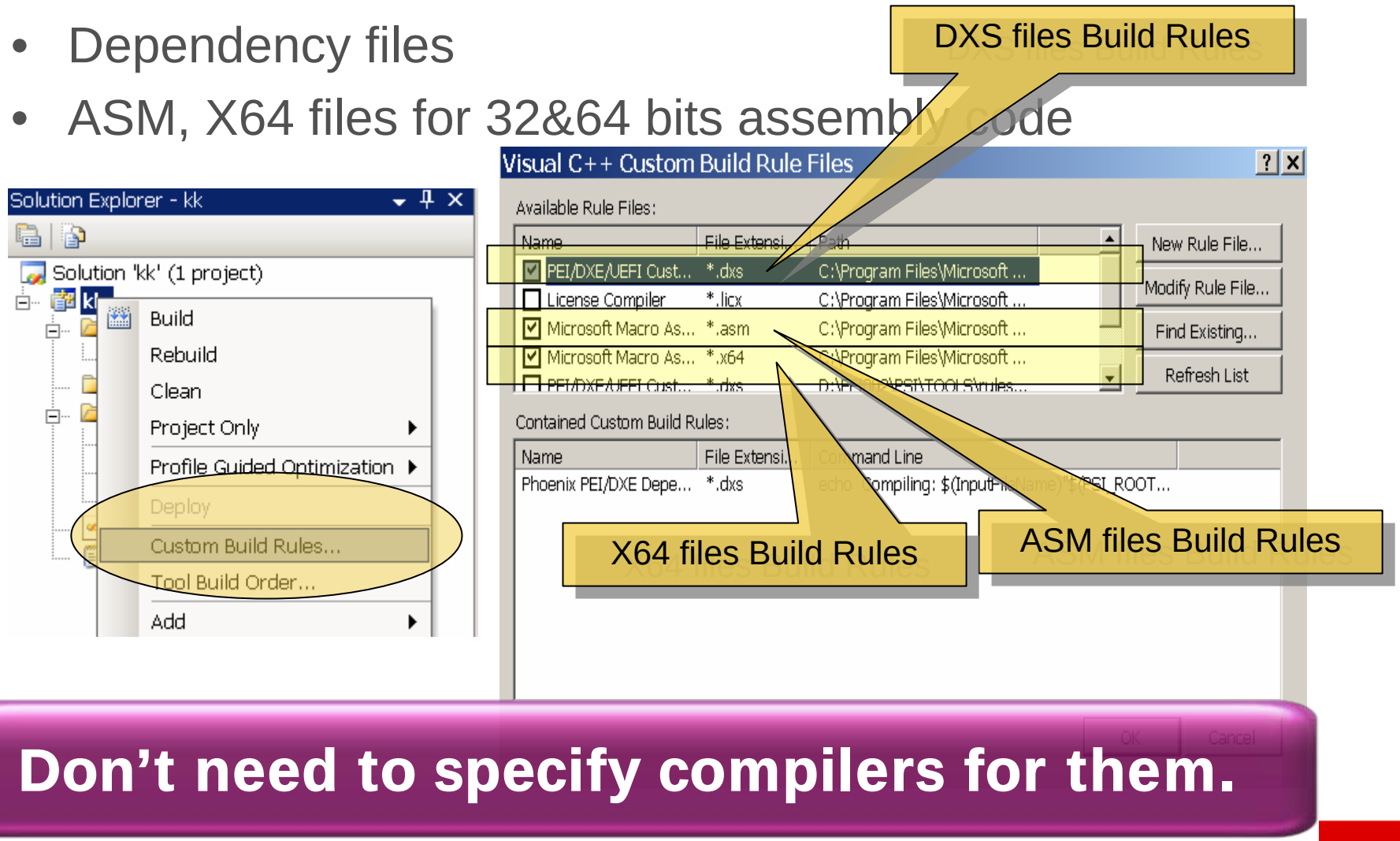
DXE BS Release X86-32

DXE BS Release X86-64



Property Setting (contd.)

- Dependency files
- ASM, X64 files for 32&64 bits assembly code

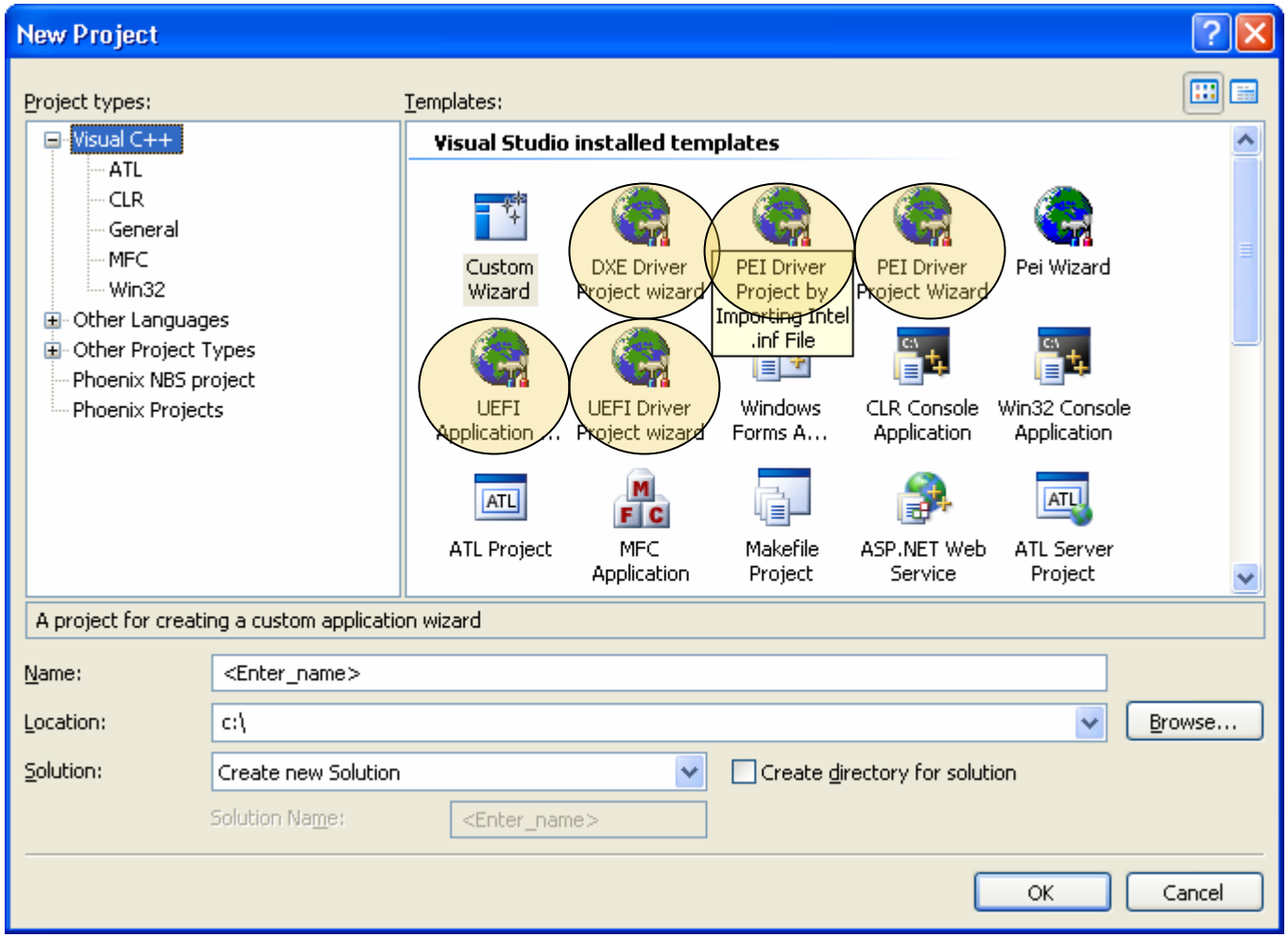


Don’t need to specify compilers for them.

Target Files

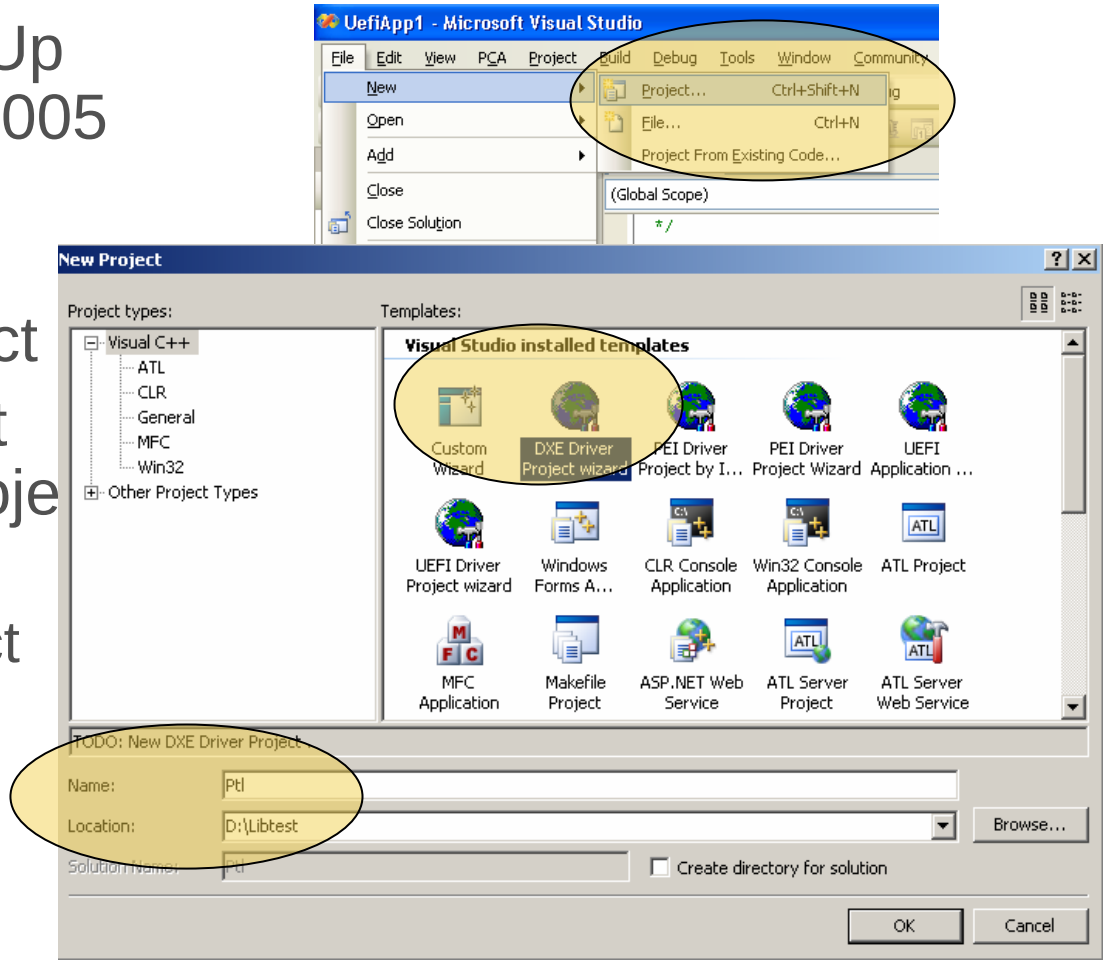
- Phoenix's format drivers and UEFI standard format.
- Debug/Release mode support.
- Merge source into FF files for debug.
- Output PDB files for debug.
- Minimize size for Release files.
- 32-bit and 64-bit(no PEI drivers) support.

Application and Driver Wizards Overview



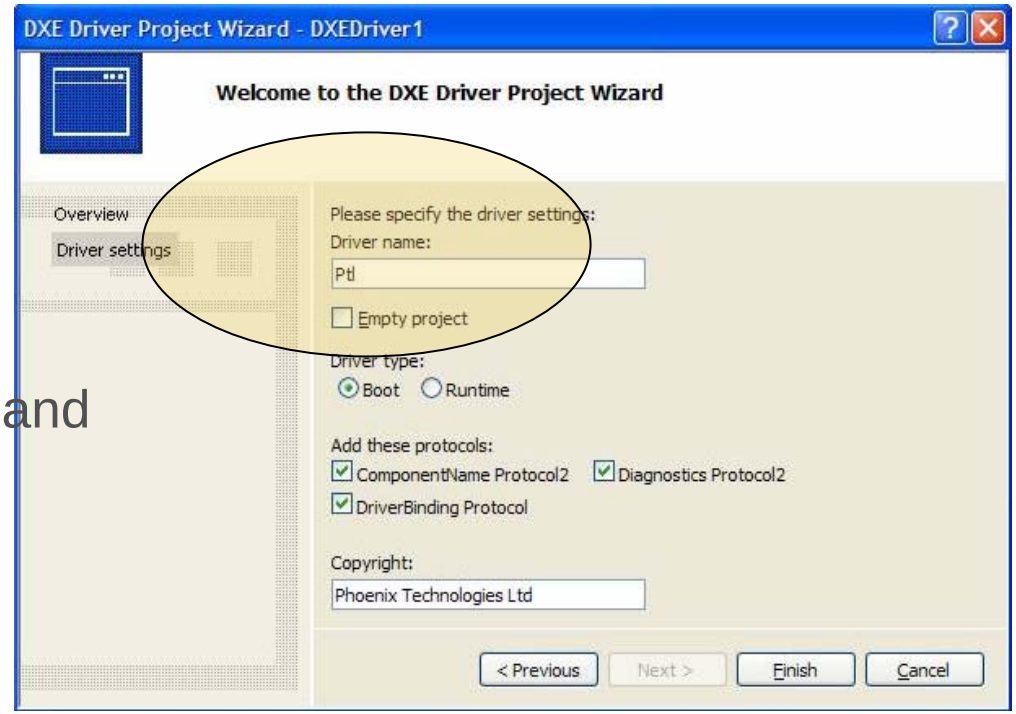
DXE Wizard Demo

- Step #1: Start Up Visual Studio 2005 SP1
- Step #2: Use File|New|Project
- Step #3: Select DXE Driver Proje Wizard
  - Select Project Name and Location
  - Press OK



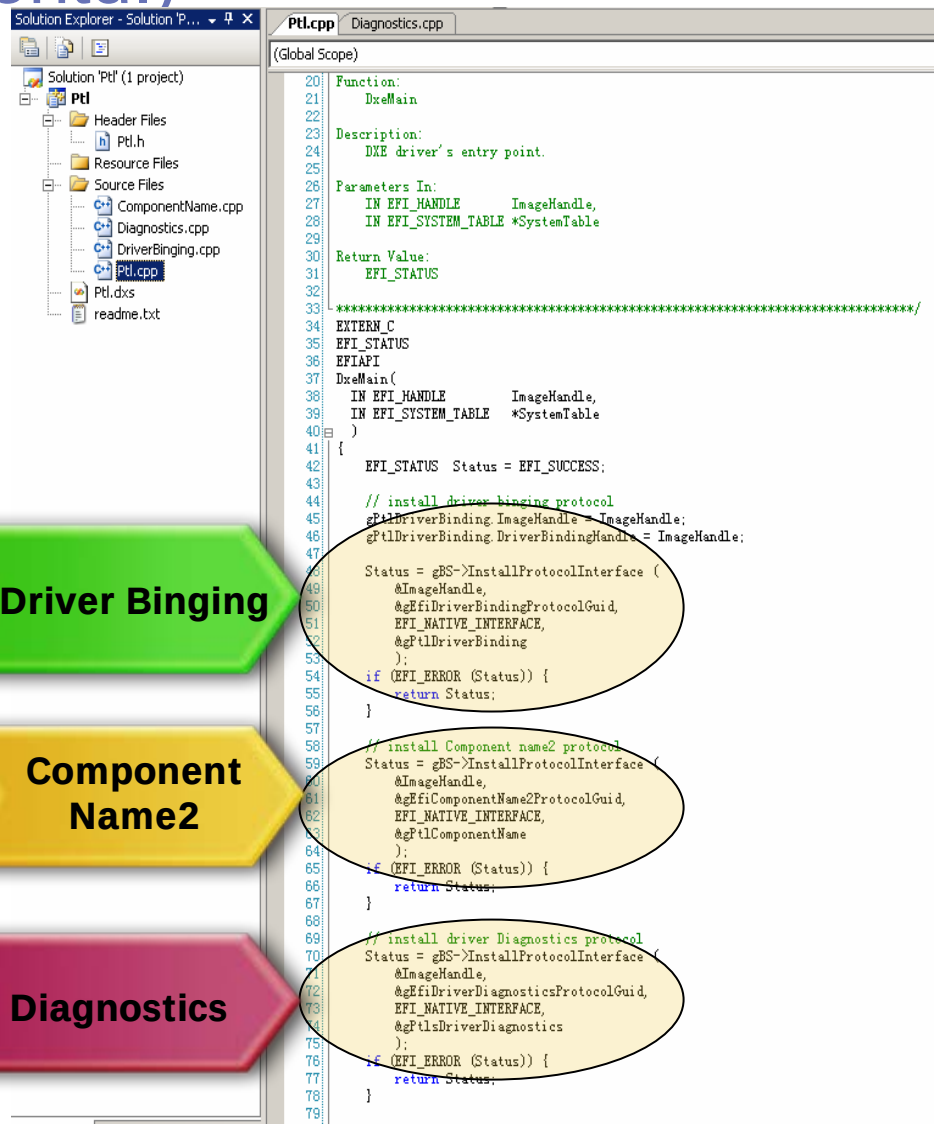
Driver Wizard Demo (contd.)

- Step #4: Change Project Settings
  - Select DXE Driver Settings
  - Change Driver Name and Copyright
  - Press Finish



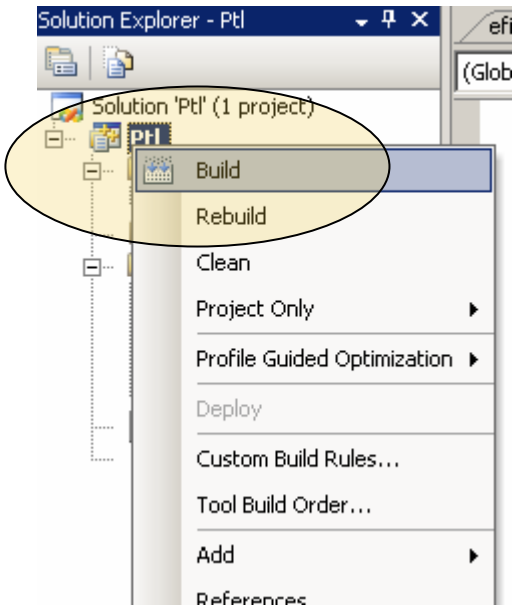
Driver Wizard Demo (contd.)

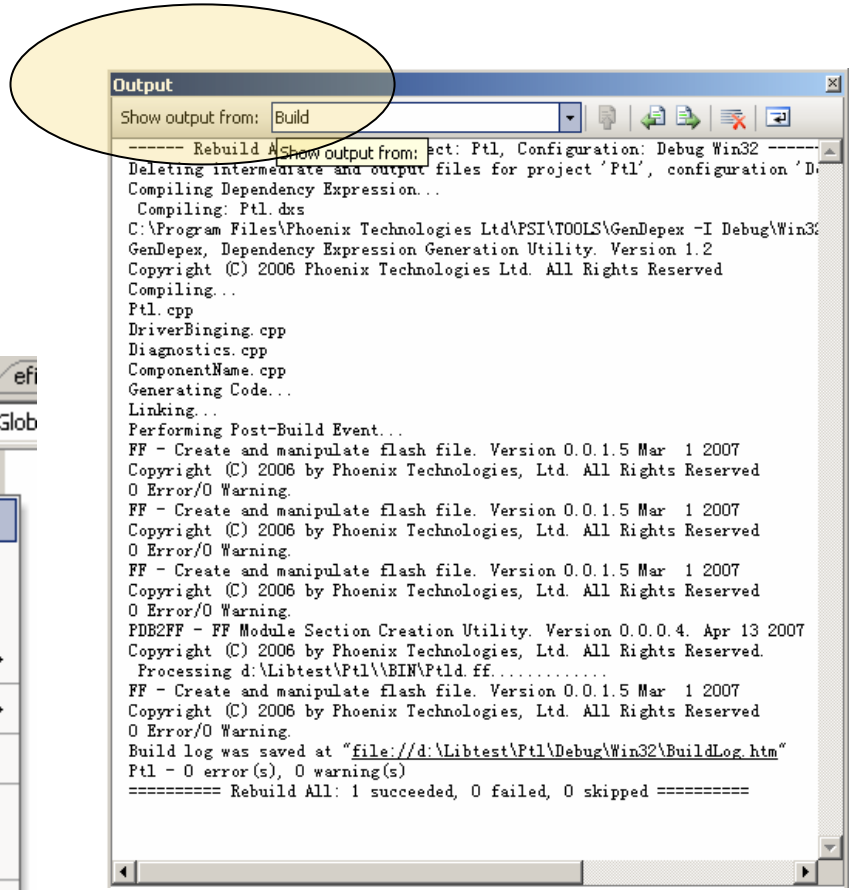
- Step #5: Develop Your Driver
  - Design the Protocol
  - Create a GUID
  - Finish Protocol's Services
  - Finish all Driver Binging Protocol's Services
  - Set Driver Dependence
  - Add the Instruction into readme.txt



Driver Wizard Demo (contd.)

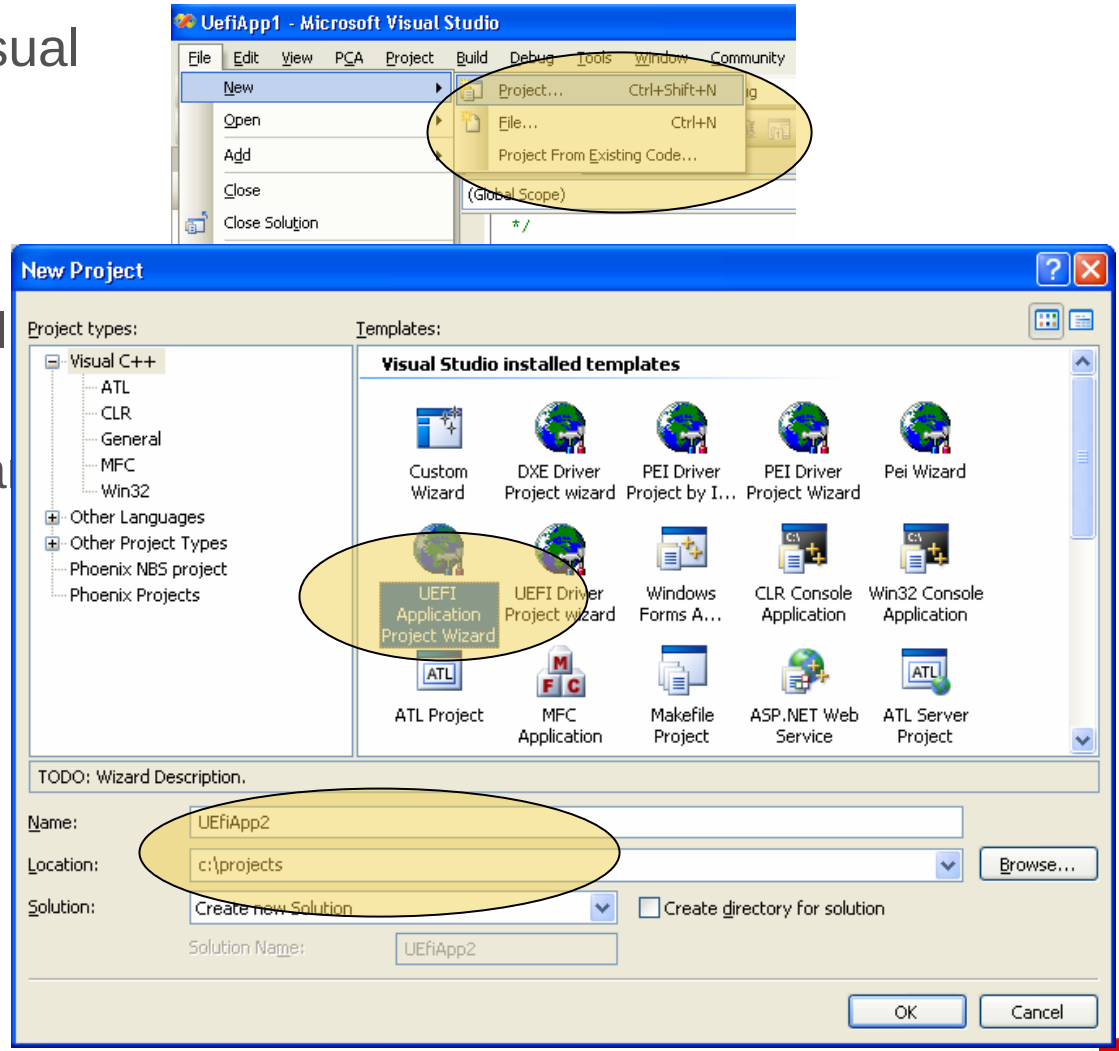
- Step #6: Build The Driver Using Build|Build
- Step #7: Check Target Files





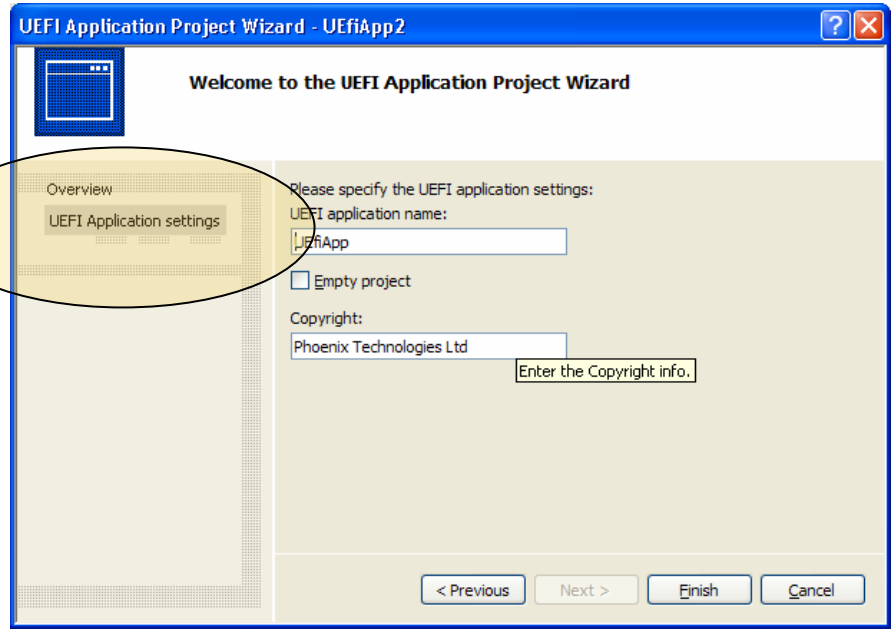
“Hello World” Application Demo

- Step #1: Start Up Visual Studio 2005 SP1
- Step #2: Use File|New|Project
- Step #3: Select UEFI Application Wizard
  - Select Project Name and Location
  - Press OK



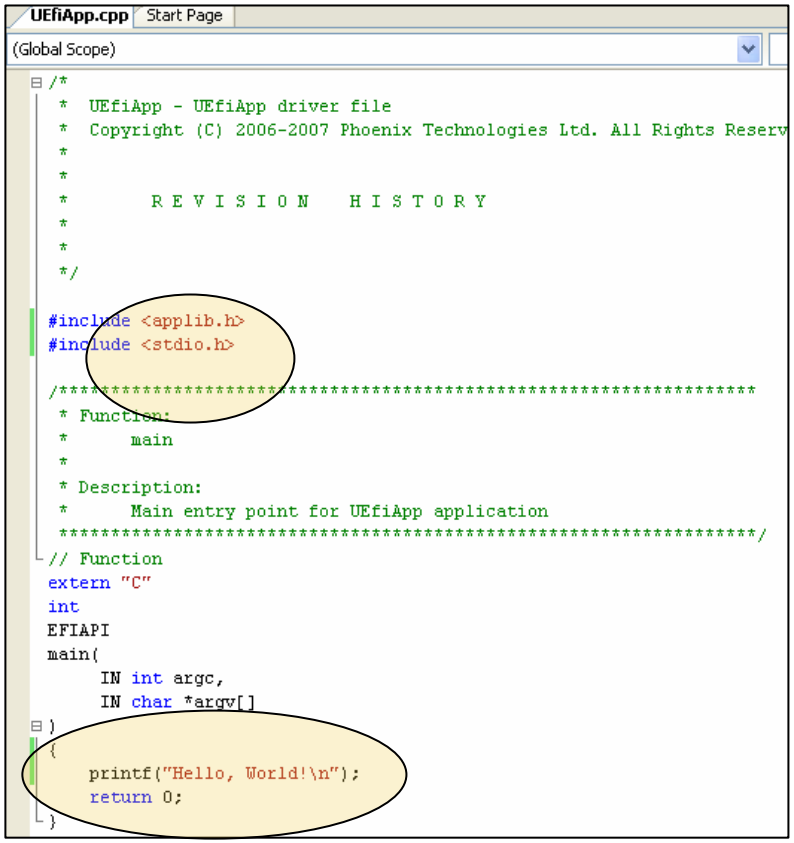
“Hello World” Application Demo (contd.)

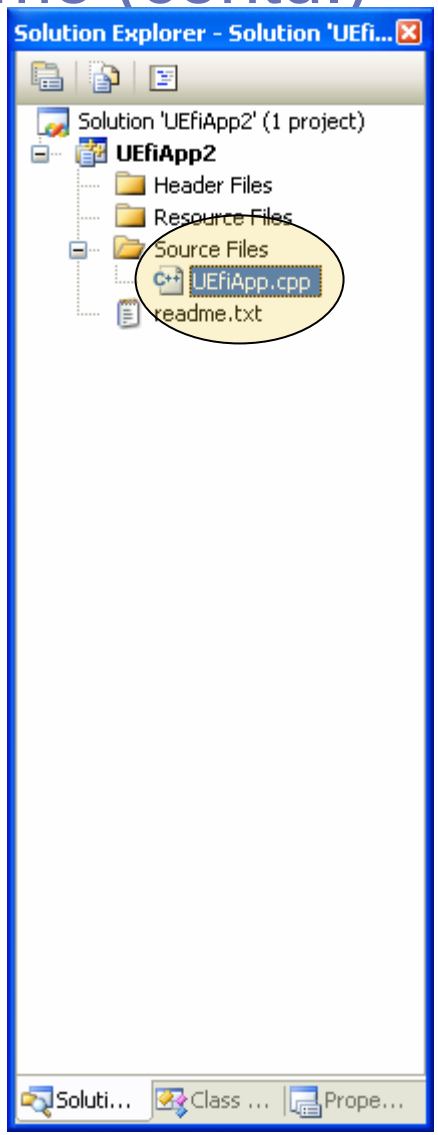
- Step #4: Change Project Settings
  - Select UEFI Application Settings
  - Change Application Name and Copyright
  - Press Finish



“Hello World” Application Demo (contd.)

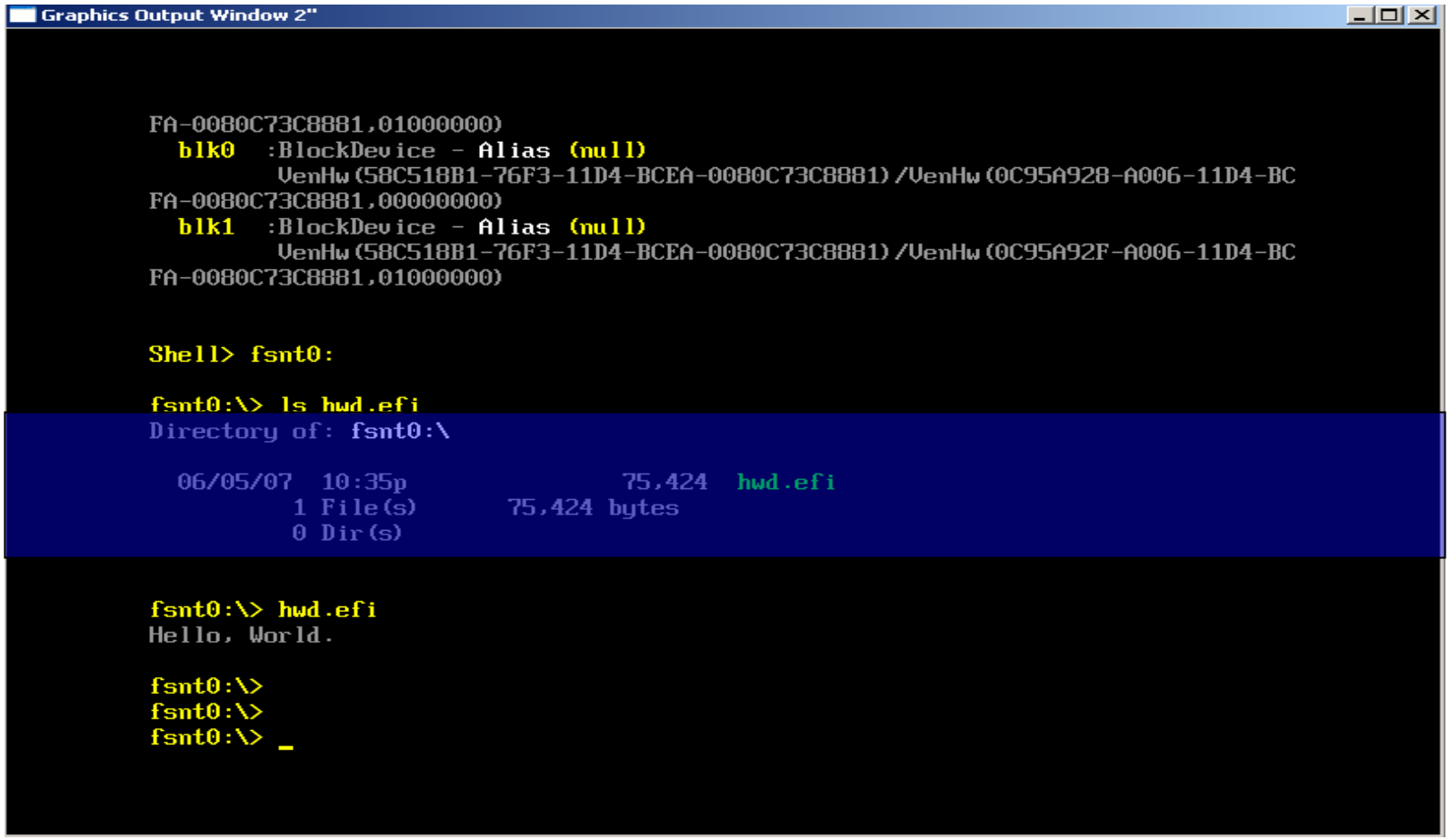
- Step #5: Add Include Files and printf in UefiApp.cpp





“Hello World” Application Demo (contd.)

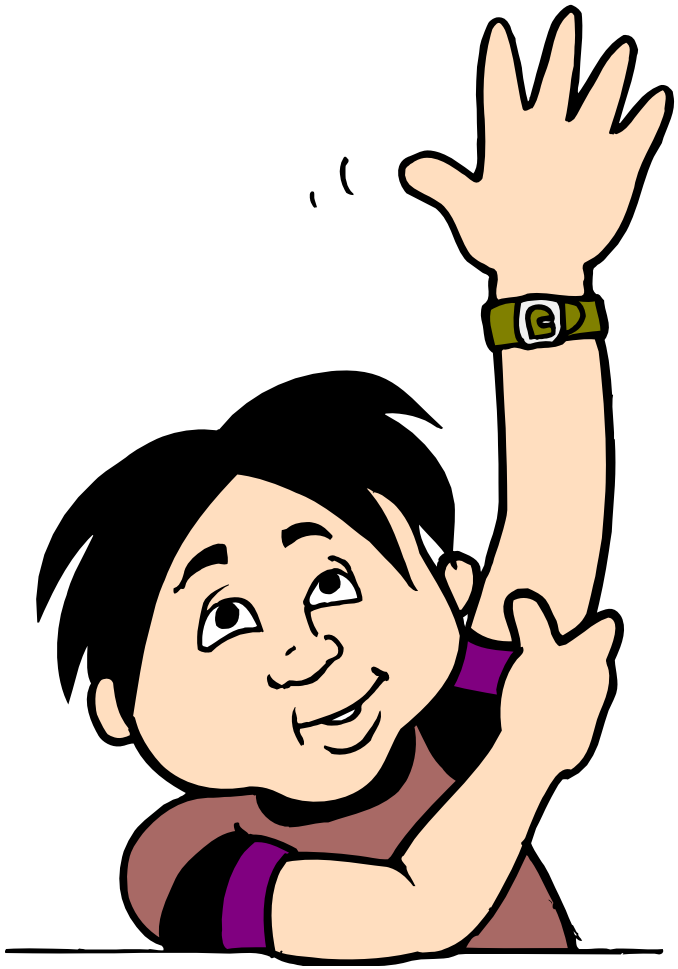
- Step #6: Run It in EDK Simulator



Summary

- VS 2005 environment, it’s easy for engineers.
- Support standard C functions.
- Tightly integrated Compiler Support.
- Easy for debug.
- Driver wizards provide great starting point.
- Special library support for UEFI.

- Phoenix’s BIOS Developer Blog
  - http://blogs.phoenix.com



Questions