tianocore

UEFI & EDK II Training

EDK II Debugging with Windows Lab

tianocore.org

Copy and Paste LabGuide.md

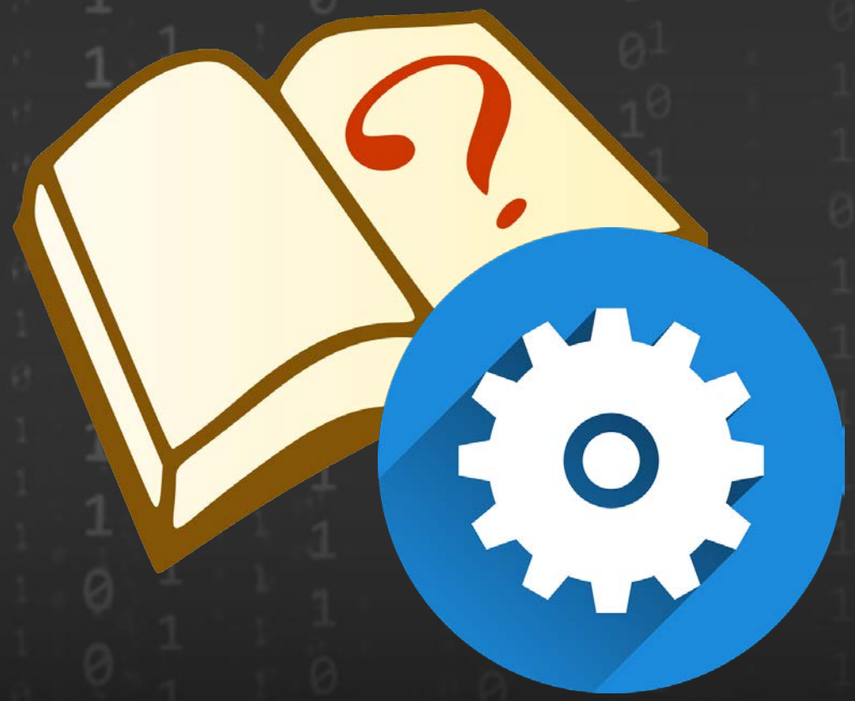
UEFI & EDK II Debugging

Lesson Objective

- Using PCDs to Configure `DebugLib` - LAB
- Change the `DebugLib` instance to modify the debug output - LAB
- Debug EDK II using VS Debugger - LAB

Lab 1 – Adding Debug Statements

In this lab, you'll add debug statements to the previous lab's SampleApp UEFI Shell application



Lab 1: Catch up from previous lab

Skip to next slide if Lab Writing UEFI App Lab completed (Lab Guide)

- Perform Lab Setup from previous Labs (Lab Guide)
- Create a Directory under the workspace `C:/FW/edk2-ws/edk2` "`SampleApp`"
- Copy contents of `C:../FW/LabSampleCode/SampleAppDebug to C:/FW/edk2-ws/edk2/SampleApp`
- Open `C:/FW/edk2/EmulatorPkg/EmulatorPkg.dsc`
- Add the following to the [Components] section:

```
# Add new modules here
 SampleApp/SampleApp.inf
```

- Save and close the file `EmulatorPkg.dsc`

Lab 1: Add debug statements to SampleApp

- Open a VS Command Prompt and type: `cd C:/FW/edk2-ws` then

```
C:/FW/edk2-ws > setenv.bat
C:/FW/edk2-ws > cd edk2
C:/FW/edk2-ws/edk2 > edksetup
```

- Open `C:/FW/edk2-ws/edk2/SampleApp/SampleApp.c`

- Add the following to the include statements at the top of the file after below the last "`include`" statement:

```
#include <Library/DebugLib.h>
```

Lab 1: Add debug statements to SampleApp

Locate the UefiMain function. Then copy and paste the following code after the "`EFI_INPUT_KEY  KEY;`" statement: and before the first `Print()` statement as shown in the screen shot below:

LabGuide.md Slide for Copy and paste

```
DEBUG ((0xffffffff, "\n\nUEFI Base Training DEBUG DEMO\n") );
DEBUG ((0xffffffff, "0xffffffff USING DEBUG ALL Mask Bits Set\n") );

DEBUG ((DEBUG_INIT,     " 0x%08x USING DEBUG DEBUG_INIT\n" , (UINTN)(DEBUG_INIT))  );
DEBUG ((DEBUG_WARN,     " 0x%08x USING DEBUG DEBUG_WARN\n", (UINTN)(DEBUG_WARN))  );
DEBUG ((DEBUG_LOAD,     " 0x%08x USING DEBUG DEBUG_LOAD\n", (UINTN)(DEBUG_LOAD))  );
DEBUG ((DEBUG_FS,       " 0x%08x USING DEBUG DEBUG_FS\n", (UINTN)(DEBUG_FS))  );
DEBUG ((DEBUG_POOL,     " 0x%08x USING DEBUG DEBUG_POOL\n", (UINTN)(DEBUG_POOL))  );
DEBUG ((DEBUG_PAGE,     " 0x%08x USING DEBUG DEBUG_PAGE\n", (UINTN)(DEBUG_PAGE))  );
DEBUG ((DEBUG_INFO,     " 0x%08x USING DEBUG DEBUG_INFO\n", (UINTN)(DEBUG_INFO))  );
DEBUG ((DEBUG_DISPATCH, " 0x%08x USING DEBUG DEBUG_DISPATCH\n",(UINTN)(DEBUG_DISPATCH)));
DEBUG ((DEBUG_VARIABLE, " 0x%08x USING DEBUG DEBUG_VARIABLE\n",(UINTN)(DEBUG_VARIABLE)));
DEBUG ((DEBUG_BM,       " 0x%08x USING DEBUG DEBUG_BM\n", (UINTN)(DEBUG_BM))  );
DEBUG ((DEBUG_BLKIO,    " 0x%08x USING DEBUG DEBUG_BLKIO\n", (UINTN)(DEBUG_BLKIO))  );
DEBUG ((DEBUG_NET,      " 0x%08x USING DEBUG DEBUG_NET\n", (UINTN)(DEBUG_NET))  );
DEBUG ((DEBUG_UNDI,     " 0x%08x USING DEBUG DEBUG_UNDI\n", (UINTN)(DEBUG_UNDI))  );
DEBUG ((DEBUG_LOADFILE, " 0x%08x USING DEBUG DEBUG_LOADFILE\n",(UINTN)(DEBUG_LOADFILE)));
DEBUG ((DEBUG_EVENT,    " 0x%08x USING DEBUG DEBUG_EVENT\n", (UINTN)(DEBUG_EVENT))  );
DEBUG ((DEBUG_GCD,      " 0x%08x USING DEBUG DEBUG_GCD\n", (UINTN)(DEBUG_EVENT)) );
DEBUG ((DEBUG_CACHE,    " 0x%08x USING DEBUG DEBUG_CACHE\n", (UINTN)(DEBUG_CACHE)) );
DEBUG ((DEBUG_VERBOSE,  " 0x%08x USING DEBUG DEBUG_VERBOSE\n", (UINTN)(DEBUG_VERBOSE)) );
DEBUG ((DEBUG_ERROR,    " 0x%08x USING DEBUG DEBUG_ERROR\n", (UINTN)(DEBUG_ERROR))  );
```

Lab 1: Build, Run and Test Result

At the VS Command Prompt

```
$> Build
$> RunEmulator.bat
```

Run the application from the shell

```
Shell> SampleApp
```

Check the VS Debug output

Exit

```
Shell> Reset
```

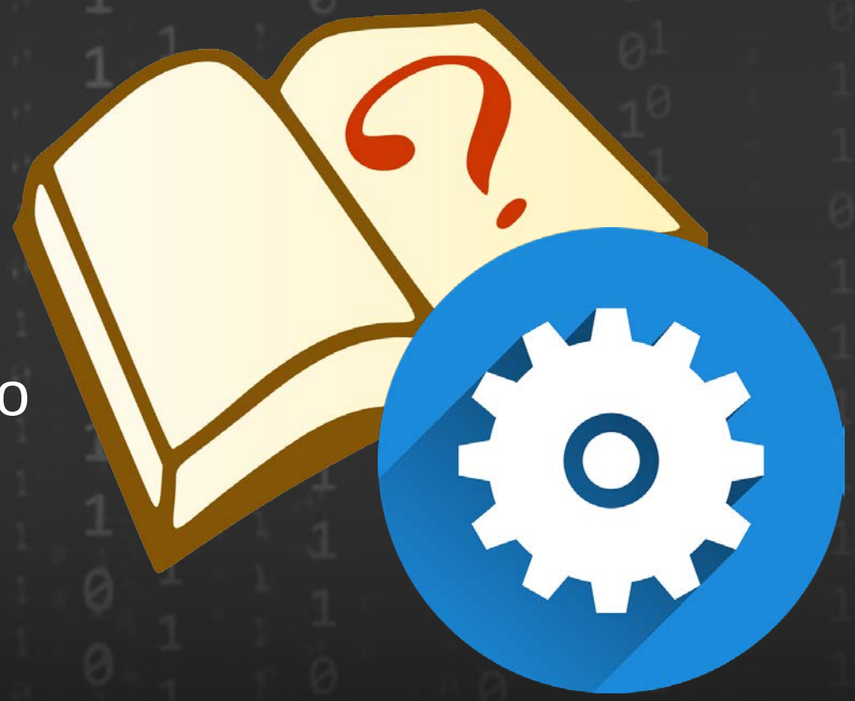
Visual Studio command prompt window output

Developer Command Prompt for VS2015

```
  - 0x00000000073A6000 - 0x0000000000007000
InstallProtocolInterface: 752F3136-4E16-4FDC-A22A-E5F46812F4CA 88C6CEC
InstallProtocolInterface: 4C8A2451-C207-405B-9694-99EA13251341 EB94090

UEFI Base Training DEBUG DEMO
0xffffffff USING DEBUG ALL Mask Bits Set
 0x00000040 USING DEBUG DEBUG_INFO
 0x80000000 USING DEBUG DEBUG_ERROR
```

Lab 2 – Changing PCD Value

In this lab, you'll learn how to use PCD values to change debugging capabilities.



Lab 2: Change PCDs for SampleApp

Open `C:/FW/edk2-ws/edk2/EmulatorPkg/EmulatorPkg.dsc`
Replace `SampleApp/SampleApp.inf` with the following:

```
SampleApp/SampleApp.inf {
   <PcdsFixedAtBuild>
    gEfiMdePkgTokenSpaceGuid.PcdDebugPropertyMask|0xff
    gEfiMdePkgTokenSpaceGuid.PcdDebugPrintErrorLevel|0xffffffff
 }
```

Save and **close** `EmulatorPkg.dsc`

LabGuide.md Slide for Copy and paste

Lab 2: Build, Run and Test Result

At the VS Command Prompt

```
$> Build
$> RunEmulator.bat
```

Run the application from the shell

```
Shell> SampleApp
```

Check the VS Debug output

Exit

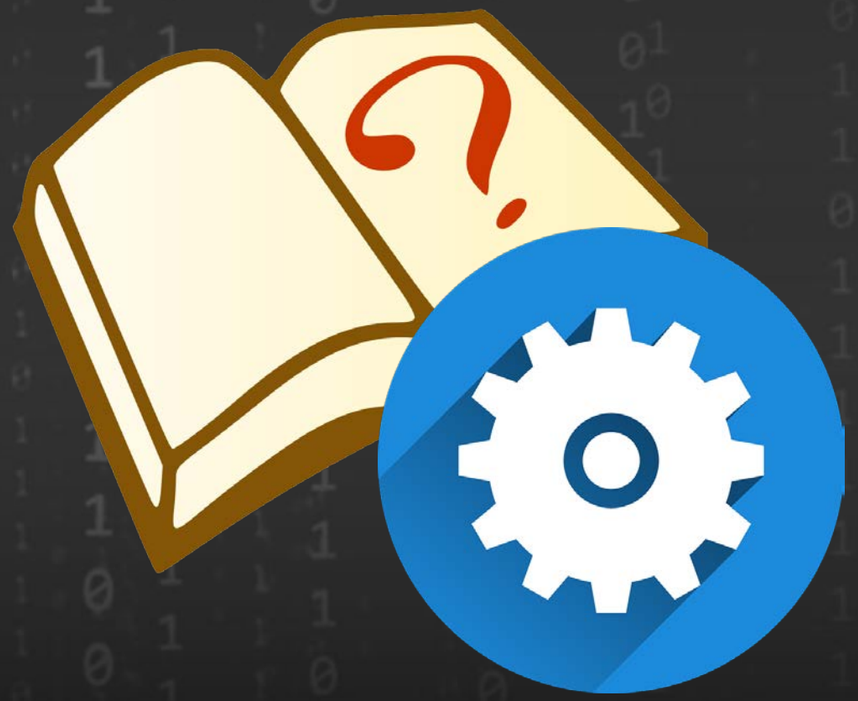
```
Shell> Reset
```

Visual Studio command prompt window output

```
c:\ Developer Command Prompt for VS2015
UEFI Base Training DEBUG DEMO
0xffffffff USING DEBUG ALL Mask Bits Set
 0x00000001 USING DEBUG DEBUG_INIT
 0x00000002 USING DEBUG DEBUG_WARN
 0x00000004 USING DEBUG DEBUG_LOAD
 0x00000008 USING DEBUG DEBUG_FS
 0x00000010 USING DEBUG DEBUG_POOL
 0x00000020 USING DEBUG DEBUG_PAGE
 0x00000040 USING DEBUG DEBUG_INFO
 0x00000080 USING DEBUG DEBUG_DISPATCH
 0x00000100 USING DEBUG DEBUG_VARIABLE
 0x00000400 USING DEBUG DEBUG_BM
 0x00001000 USING DEBUG DEBUG_BLKIO
 0x00004000 USING DEBUG DEBUG_NET
 0x00010000 USING DEBUG DEBUG_UNDI
 0x00020000 USING DEBUG DEBUG_LOADFILE
 0x00080000 USING DEBUG DEBUG_EVENT
 0x00080000 USING DEBUG DEBUG_GCD
 0x00080000 USING DEBUG DEBUG_CACHE
 0x00080000 USING DEBUG DEBUG_VERBOSE
 0x80000000 USING DEBUG DEBUG_ERROR
```

Lab 3 – Library Instances for Debugging

In this lab, you'll learn how to add specific debug library instances.



Lab 3: Using Library Instances for Debugging

Open `C:/FW/edk2-ws/edk2/EmulatorPkg/EmulatorPkg.dsc`
Replace `SampleApp/SampleApp.inf { . . .}` with the following:

```
SampleApp/SampleApp.inf {
   <LibraryClasses>
    DebugLib|MdePkg/Library/UefiDebugLibConOut/UefiDebugLibConOut.inf
 }
```

Save and **close** `EmulatorPkgPkg.dsc`

LabGuide.md Slide for Copy and paste

Lab 3: Build, Run and Test Result

At the VS Command Prompt

```
$> Build
$> RunEmulator.bat
```

Run the application from the shell

```
Shell> SampleApp
```

See that the output from the Debug statements now goes to the console

Exit

```
Shell> Reset
```

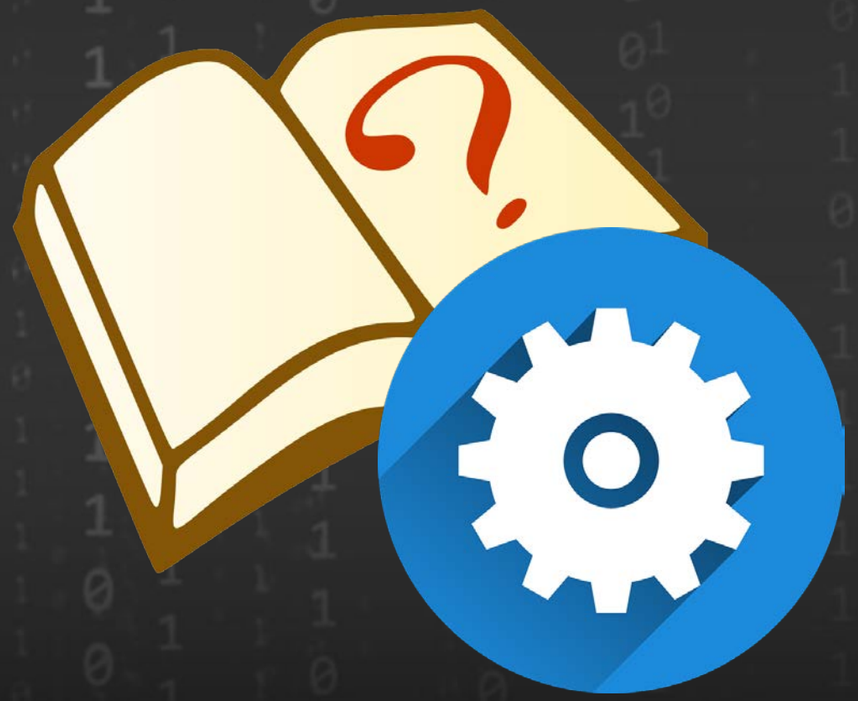
Debug output to console

```
Shell> sampleapp

UEFI Base Training DEBUG DEMO
0xffffffff USING DEBUG ALL Mask Bits Set
 0x00000040 USING DEBUG DEBUG_INFO
 0x80000000 USING DEBUG DEBUG_ERROR
System Table: 0xB7A7C018

Press any Key to continue :
```

EmulatorPkg

Lab 4: Null Instance of `DebugLib`

In this lab, you'll change the `DebugLib` to the Null instance.



Lab 4: Using Null Library Instances

Open `C:/FW/edk2-ws/edk2/EmulatorPkg/EmulatorPkg.dsc`
Replace `SampleApp/SampleApp.inf { . . .}` with the following:

```
SampleApp/SampleApp.inf {
   <LibraryClasses>
   DebugLib|MdePkg/Library/BaseDebugLibNull/BaseDebugLibNull.inf
}
```

Save and **close** `EmulatorPkg.dsc`

LabGuide.md Slide for Copy and paste

Lab 4: Build, Run and Test Result

At the VS Command Prompt

```
$> Build
$> RunEmulator.bat
```

Run the application from the shell

```
Shell> SampleApp
```

Check – now **NO** Debug output

Exit

```
Shell> Reset
```

Visual Studio command prompt window output – NO DEBUG

```
Developer Command Prompt for VS2015
Loading driver at 0x0000618A000 EntryPoint=0x000001C1090 SampleApp.efi
InstallProtocolInterface: BC62157E-3E33-4FEC-9920-2D3B36D750DF 62AF410
ProtectUefiImageCommon - 0x62AF128
  - 0x000000000618A000 - 0x0000000000006000
InstallProtocolInterface: 752F3136-4E16-4FDC-A22A-E5F46812F4CA 7534CEC
```

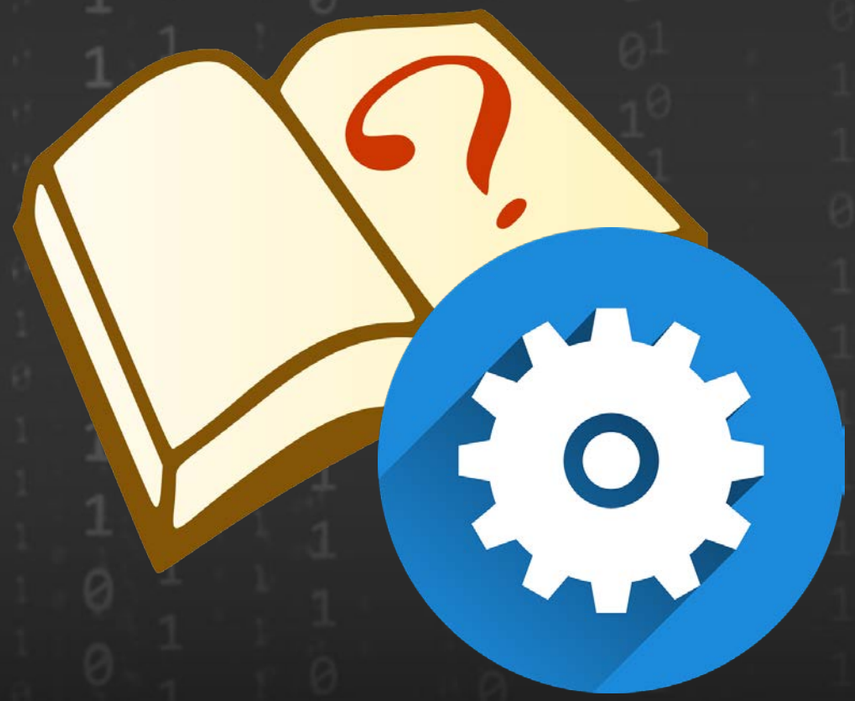
Console window – NO DEBUG

```
Shell> sampleapp
System Table: 0x074CF010

Press any Key to continue :

Enter text. Include a dot ('.') in a sentence then <Enter> to ex
```

Lab 5: Debugging EDK II with VS Debugger

In this lab, you'll learn how setup the VS to debug the EDK II emulation

First make sure you have "Just-in-Time" Debugging enabled in the Visual Studio Application Link



Lab 5: Debug with VS

Edit the SampleApp.c and add an "ASSERT_EFI_ERROR" :
Add the following:

```
EFI_STATUS      Status;
Status = EFI_NO_RESPONSE;
      . . .
ASSERT_EFI_ERROR(Status);
```

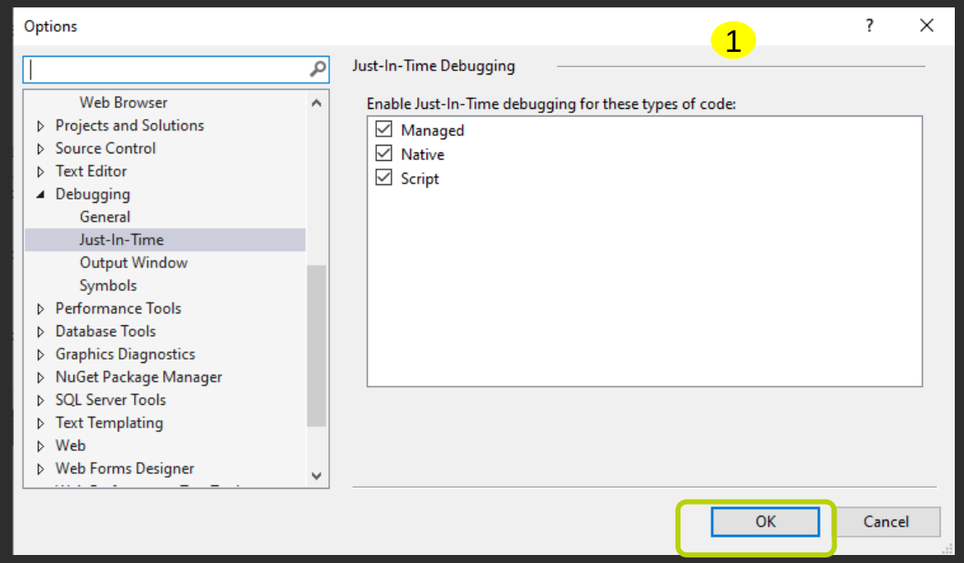
```
EFI_STATUS Status;
Status = EFI_NO_RESPONSE;  // or any EFI Error

DEBUG((0xffffffff, "\n\nUEFI Base Training DEBUG DEMO\n"));
DEBUG((0xffffffff, "0xffffffff USING DEBUG ALL Mask Bits Set\n"));

ASSERT_EFI_ERROR(Status);
```

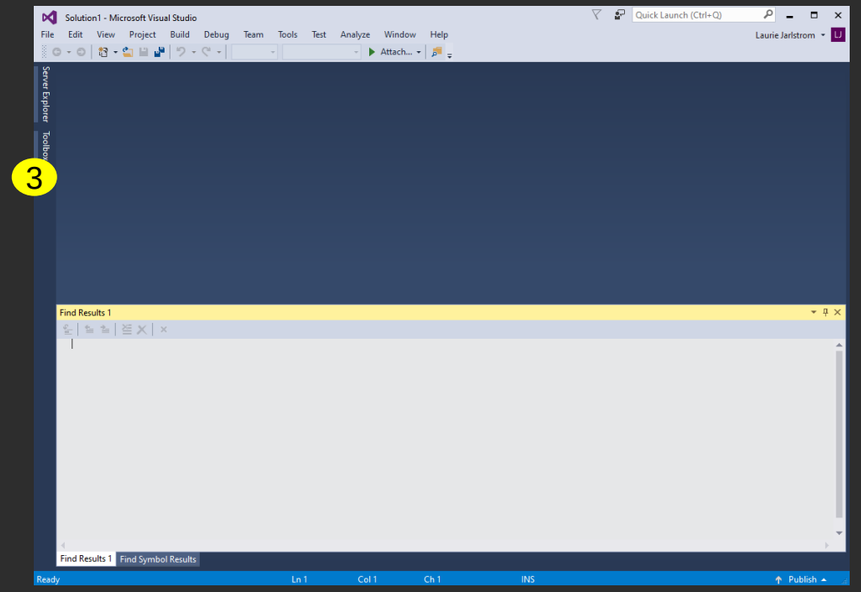
Save SampleApp.c

LabGuide.md Slide for Copy and paste

First Enable Visual Studio Debugging

1. First make sure you have “Just-in-Time” Debugging enabled in the Visual Studio Application Link (note: need to open as Admin)
2. Use the Regedt32 to add the “`Auto`” key: Link
3. Next Open the Visual Studio Application





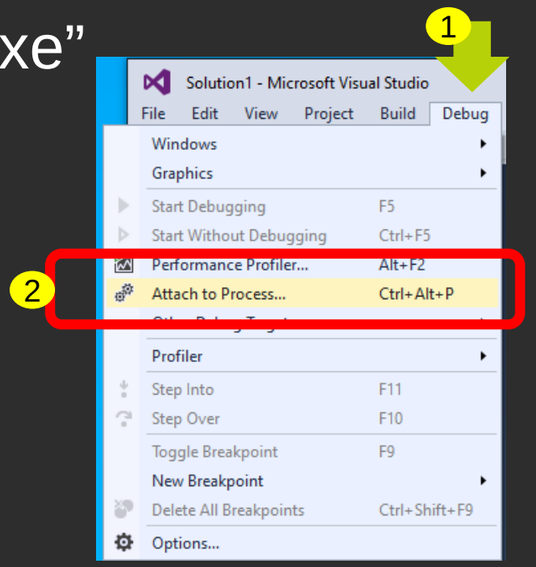
Lab 5: Debug with VS - ASSERT

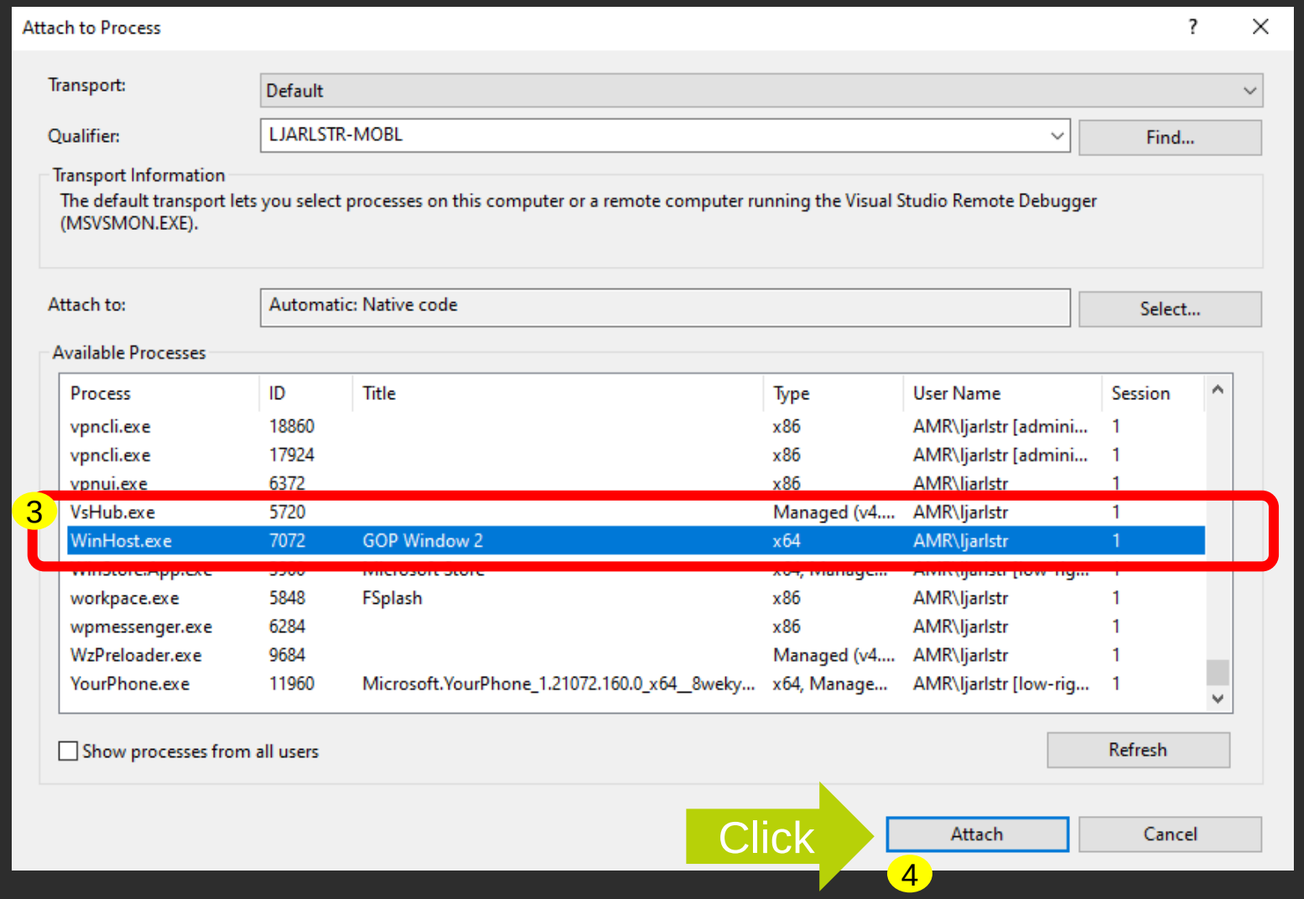
At the VS Command Prompt

```
$> Build
$> RunEmulator.bat
```

Inside the Visual Studio app, Enable the Winhost.exe for Debugging

1. Select “Debug”
2. “Attach to “Process”
3. Find “WinHost.exe”
4. Click “Attach”





Lab 5: Debug with VS - ASSERT

Run the application from the shell

```
Shell> SampleApp
```

Assert in VS Command Prompt

Visual Studio command prompt window output

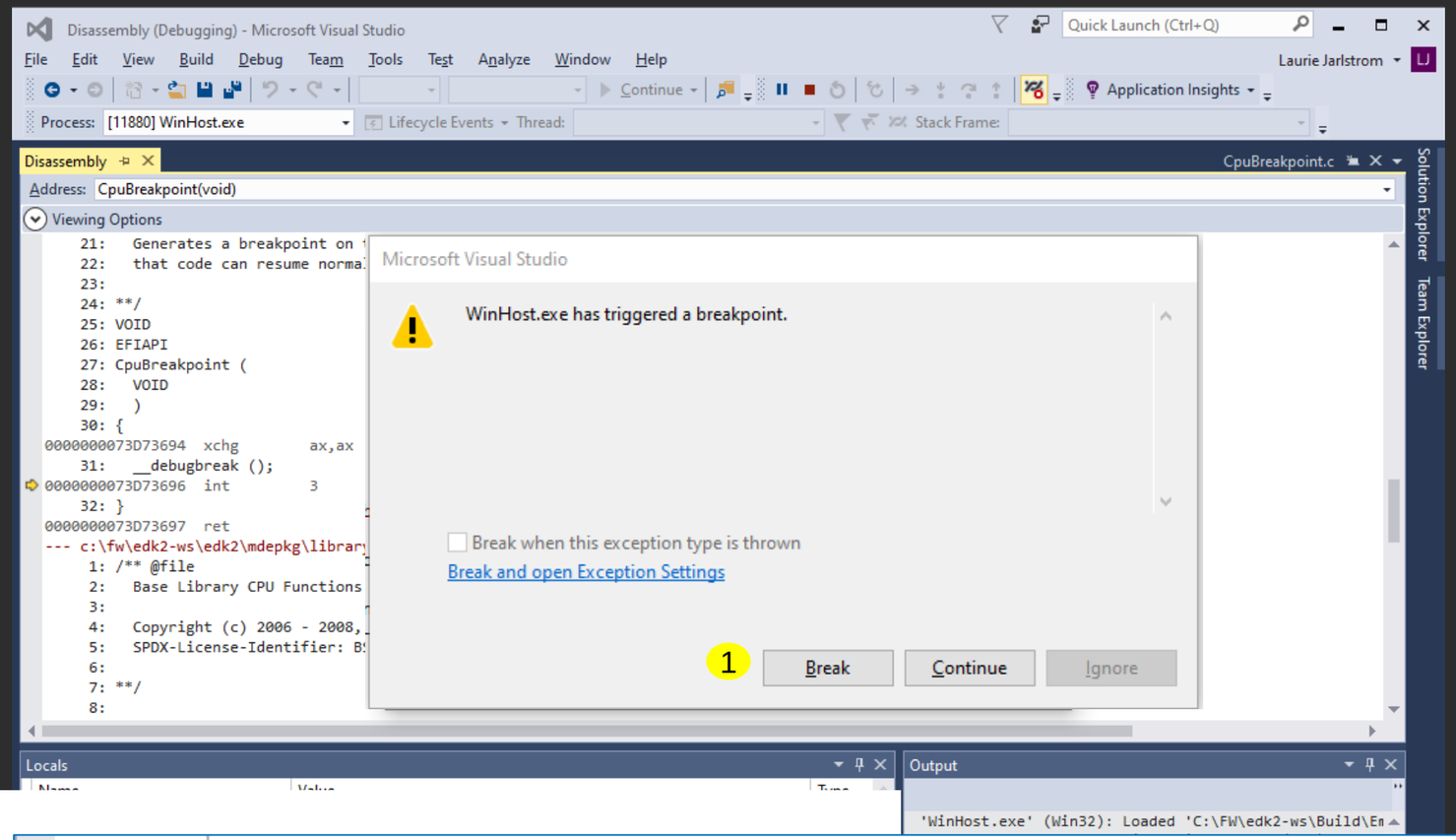
```
Developer Command Prompt for VS2015 - runEmulator.bat
InstallProtocolInterface: 5B1B31A1-9562-11D2-8E3F-00A0C969723B 1D55B83F440
LoadLibraryEx (
  c:\fw\edk2-ws\Build\EmulatorX64\DEBUG_VS2015x86\X64\SampleApp\SampleApp\DEBUG\SampleApp.DLL,
  NULL, DONT_RESOLVE_DLL_REFERENCES)
Loading driver at 0x1D55B7E4000 EntryPoint=0x00077441000 SampleApp.efi
InstallProtocolInterface: BC62157E-3E33-4FEC-9920-2D3B36D750DF 1D55B840018
ProtectUefiImageCommon - 0x5B83F440
  - 0x000001D55B7E4000 - 0x000000000000E000
InstallProtocolInterface: 752F3136-4E16-4FDC-A22A-E5F46812F4CA 1D557D8D628


UEFI Base Training DEBUG DEMO

ASSERT_EFI_ERROR (Status = No Response)

DXE_ASSERT!: [SampleApp] c:\fw\edk2-ws\edk2\SampleApp\SampleApp.c (51): !EFI_ERROR (Status)
```

Lab 5: Debug with VS - ASSERT

Windows* VS Debugger

1. Select “Break”
2. “F10” about 2 times
3. SampleApp.c can be debugged

“F5” to continue
“Shift F5” to Stop debugging




```
24  **/
25  VOID
26  EFIAPI
27  CpuBreakpoint (
28    VOID
29    )
30  {
31    __debugbreak ();
32  }
```

2

“F10” - Step over 2-3 times

```
53
54  DEBUG((0xffffffff, "\n\nUEFI Base Training DEBUG DEMO\n"));
55  DEBUG((0xffffffff, "0xffffffff USING DEBUG ALL Mask Bits Set\n"));
56
57  ASSERT_EFI_ERROR(Status);  ≤1ms elapsed
58
59
60  DEBUG((DEBUG_INIT, " 0x%08x USING DEBUG DEBUG_INIT\n", (UINTN)(DEBUG_INIT)));
```

3

SampleApp.c

Lab 5: Debug with VS - CpuBreakpoint

Edit the `SampleApp.c` and add "`CpuBreakpoint();`" Statement and comment out the "`ASSERT`":

```
CpuBreakpoint();
```

SampleApp.c

Miscellaneous Files | (Global Scope)

```
43      EFI_INPUT_KEY      Key;
44      DEBUG((EFI_D_INFO, "\r\n>>>>>[UefiMain] Entry point: 0x%p <<<<<<\r\n"
45
46      DEBUG((0xffffffff, "\n\nUEFI Base Training DEBUG DEMO\n"));
47      DEBUG((0xffffffff, "0xffffffff USING DEBUG ALL Mask Bits Set\n"));
48
49      //ASSERT_EFI_ERROR(0x8000000000000000);
50
51          CpuBreakpoint();
52
```

Save `SampleApp.c`

LabGuide.md Slide for Copy and paste

Lab 5: Debug with VS

At the VS Command Prompt

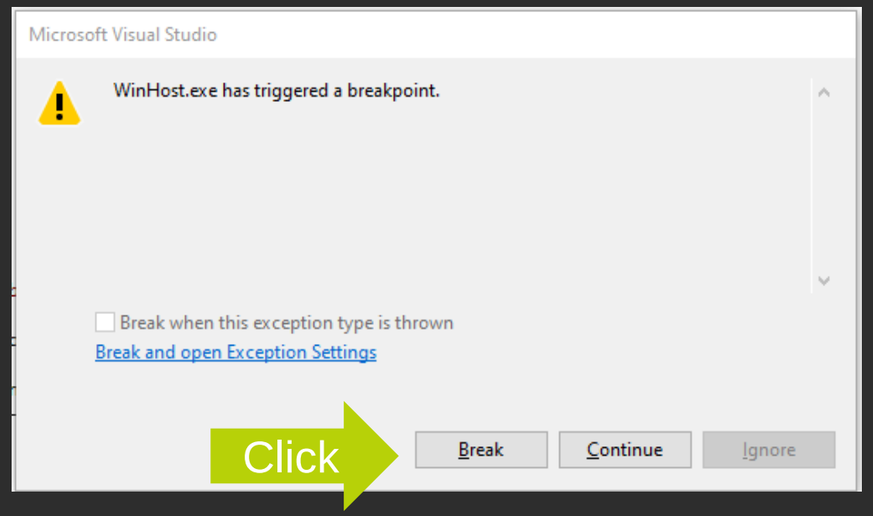
```
$> Build
$> RunEmulator.bat
```

Inside the Visual Studio App, Enable the Winhost.exe for Debugging.
“Cnt-Alt-p” then select “Winhost.exe”

Run the application from the shell

```
Shell> SampleApp
```

VS Debugger pop up, select “Break”
Press “F10” until SampleApp.c shows



Invoke Windows Visual Studio Debugger

WinHost (Debugging) - Microsoft Visual Studio

File Edit View Project Build Debug Team Tools Test Analyze Window Help

Continue | Application Insights

Process: [29848] WinHost.exe | Lifecycle Events | Thread: [27704] Main Thread | Stack Frame: UefiMain

Miscellaneous Files | (Global Scope) | UefiMain(IN EFI_HANDLE ImageHandle, IN EFI_SYSTEM_TABLE * Syste

```
41      UINTN          EventIndex;
42      BOOLEAN        ExitLoop;
43      EFI_INPUT_KEY      Key;
44      DEBUG((EFI_D_INFO, "\r\n>>>>>[UefiMain] Entry point: 0x%p <<<<<<\r\n", UefiMain));
45
46      DEBUG((0xffffffff, "\n\nUEFI Base Training DEBUG DEMO\n"));
47      DEBUG((0xffffffff, "0xffffffff USING DEBUG ALL Mask Bits Set\n"));
48
49      //ASSERT_EFI_ERROR(0x8000000000000000);
50
51         CpuBreakpoint();
52
53
54      DEBUG((DEBUG_INIT, " 0x%08x USING DEBUG DEBUG_INIT\n", (UINTN)(DEBUG_INIT)));  ≤2ms elapsed
55      DEBUG((DEBUG_WARN, " 0x%08x USING DEBUG DEBUG_WARN\n", (UINTN)(DEBUG_WARN)));
56      DEBUG((DEBUG_LOAD, " 0x%08x USING DEBUG DEBUG_LOAD\n", (UINTN)(DEBUG_LOAD)));
57      DEBUG((DEBUG_FS, " 0x%08x USING DEBUG DEBUG_FS\n", (UINTN)(DEBUG_FS)));
58      DEBUG((DEBUG_POOL, " 0x%08x USING DEBUG DEBUG_POOL\n", (UINTN)(DEBUG_POOL)));
59      DEBUG((DEBUG_PAGE, " 0x%08x USING DEBUG DEBUG_PAGE\n", (UINTN)(DEBUG_PAGE)));
60      DEBUG((DEBUG_INFO, " 0x%08x USING DEBUG DEBUG_INFO\n", (UINTN)(DEBUG_INFO)));
61      DEBUG((DEBUG_DISPATCH, " 0x%08x USING DEBUG DEBUG_DISPATCH\n", (UINTN)(DEBUG_DISPATCH)));
62      DEBUG((DEBUG_VARIABLE, " 0x%08x USING DEBUG DEBUG_VARIABLE\n", (UINTN)(DEBUG_VARIABLE)));
63      DEBUG((DEBUG_BM, " 0x%08x USING DEBUG DEBUG_BM\n", (UINTN)(DEBUG_BM)));
64      DEBUG((DEBUG_BLKIO, " 0x%08x USING DEBUG DEBUG_BLKIO\n", (UINTN)(DEBUG_BLKIO)));
65      DEBUG((DEBUG_NET, " 0x%08x USING DEBUG DEBUG_NET\n", (UINTN)(DEBUG_NET)));
66      DEBUG((DEBUG_UNDI, " 0x%08x USING DEBUG DEBUG_UNDI\n", (UINTN)(DEBUG_UNDI)));
```

100 %

Debug using “F5”-Continue
or “F10” - Step over

Locals

Name	Value	Type
EventIndex	1806700961	unsigned
ExitLoop	4 '\x4'	unsigned
ImageHandle	0x000001f1febdfe98	void *
Key	{ScanCode=497 UnicodeChar=0 }	EFI_INPU
ScanCode	497	unsigned
UnicodeChar	0	unsigned
SystemTable	0x000001f1fb10c018 {Hdr={Signature=6076298535811760713 Revision=131142 He	EFI_SYST

Autos | Locals | Watch 1

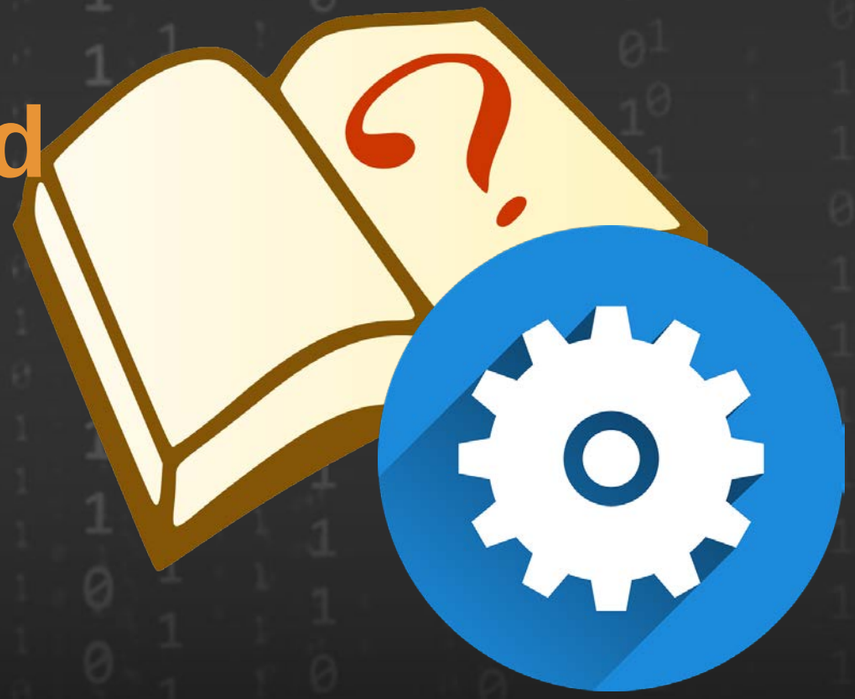
Call Stack

Name
SampleApp.dll!UefiMain(void * ImageHandle, EFI_SYSTEM_TABLE * SystemTable) Line 54
SampleApp.dll!ProcessModuleEntryPointList(void * ImageHandle, EFI_SYSTEM_TABLE * SystemTable) Line 293
SampleApp.dll!_ModuleEntryPoint(void * ImageHandle, EFI_SYSTEM_TABLE * SystemTable) Line 59
DxeCore.dll!CoreStartImage(void * ImageHandle, unsigned __int64 * ExitDataSize, unsigned short * * ExitData) Line 1
Shell.dll!InternalShellExecuteDevicePath(void * const * ParentImageHandle, const EFI_DEVICE_PATH_PROTOCOL * D
Shell.dll!RunCommandOrFile(SHELL_OPERATION_TYPES Type, const unsigned short * CmdLine, unsigned short * Fi
Shell.dll!SetupAndRunCommandOrFile(SHELL_OPERATION_TYPES Type, unsigned short * CmdLine, unsigned short

Call Stack | Breakpoints | Exception Settings | Command Window | Immediate Window | Output | Error List

Lab 6: Debugging EDK II add Debug to Boot Flow

In this lab, you'll learn how add Debug statements to the EDK II Boot flow and check the debug log output



Lab 6: Debug Boot Flow

Edit the MdeModulePkg/Core/Pei/PeiMain/PeiMain.c and add a "DEBUG" print ~line 489 before the call to the PeiDispatcher:

```
DEBUG((DEBUG_INFO, "***********Before call to Pei Dispatcher ********\n"));
```

Save PeiMain.c

```
486    //
487    // Call PEIM dispatcher
488    //
489    DEBUG((DEBUG_INFO, "***********Before call to Pei Dispatcher ********\n"));
490    PeiDispatcher (SecCoreData, &PrivateData);
491
```

Lab 6: Build, Run and Test Result

At the VS Command Prompt

```
$> Build
$> RunEmulator.bat
```

Check the VS Debug output

Exit

```
Shell> Reset
```

Visual Studio command prompt window output

```
c:\ Developer Command Prompt for VS2015
Stack Hob: BaseAddress=0x2279DDE0000 Length=0x20000
Heap Offset = 0x710000 Stack Offset = 0x710000
Loading PEIM 52C05B14-0B98-496C-BC3B-04B50211D680
WARNING: DLL already loaded.  No source level debug c:\fw\edk2-ws\Build\Emulator)
BUG\PeiCore.DLL.
Loading PEIM at 0x227A1DC0000 EntryPoint=0x227A1DC1078 PeiCore.efi
Reinstall PPI: 8C8CE578-8A3D-4F1C-9935-896185C32DD3
Reinstall PPI: 5473C07A-3DCB-4DCA-BD6F-1E9689E7349A
Reinstall PPI: B9E0ABFE-5979-4914-977F-6DEE78C278A6
Install PPI: F894643D-C449-42D1-8EA8-85BDD8C65BDE
**********Before call to Pei Dispatcher ********
Loading PEIM 9B3ADA4F-AE56-4C24-8DEA-F03B7558AE50
WARNING: DLL already loaded.  No source level debug c:\fw\edk2-ws\Build\Emulator)
cd\DEBUG\PcdPeim.DLL.
Loading PEIM at 0x227A1DB1000 EntryPoint=0x227A1DB2004 PcdPeim.efi
Reinstall PPI: 06E81C58-4AD7-44BC-8390-F10265F72480
Reinstall PPI: 4D8B155B-C059-4C8F-8926-06FD4331DB8A
Reinstall PPI: 01F34D25-4DE2-23AD-3FF3-36353FF323F1
Reinstall PPI: A60C6B59-E459-425D-9C69-0BCC9CB27D81
Loading PEIM 6DB075DE-449E-2644-96D0-CC5A1B4C3B2A
LoadLibraryEx (
  c:\fw\edk2-ws\Build\EmulatorX64\DEBUG_VS2015x86\X64\EmulatorPkg\FirmwareVolumeF
  NULL, DONT_RESOLVE_DLL_REFERENCES)
```

Summary

- Using PCDs to Configure `DebugLib` - LAB
- Change the `DebugLib` instance to modify the debug output - LAB
- Debug EDK II using VS Debugger - LAB

tianocore

Questions?

Return to Main Training Page

Return to Training Table of contents for next presentation link

tianocore

tianocore

www.tianocore.org

ACKNOWLEDGEMENTS

BACK UP

ISSUE:

Debugging in Emulator with Windows 7/10 64Bit and Visual Studio does not work?

Symptom: With Windows 7/10 64bit a `CpuBreakpoint()` or `ASSERT` just exits with an error from the “Build Run” command.

Link to fix this issue:
https://github.com/tianocore/tianocore.github.io/wiki/NT32#Debugging_in_Nt32_Emulation_with_Windows_7_and_Visual_Studio_does_not_work

1. Run the RegEdt32
2. Navigate to the HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows NT\CurrentVersion\AeDebug
3. Add a string value entry called "Auto" with a value of "1“