

*presented by*



**Innovative Software Tools & Methods to  
Profile, Test and Optimize UEFI Firmware  
Improving Test Coverage and Debug Results**

UEFI US Fall Plugfest – September 20 - 22, 2016  
Presented by Kevin Davis (Insyde Software)

# Agenda



- Introduction
- Intel Processor Trace Buffer
- Driver Profiling
- Code Coverage Display
- Questions?

# Debug Models



- Hardware Level Debuggers
  - Physical Access to board required
  - Complexity of connection to board
  - No or minor impact to software execution
- Software Level Debuggers
  - No need to have physical access to the board
  - Some impact to software execution
  - Complexity of sharing a processor

# Define Software Terms



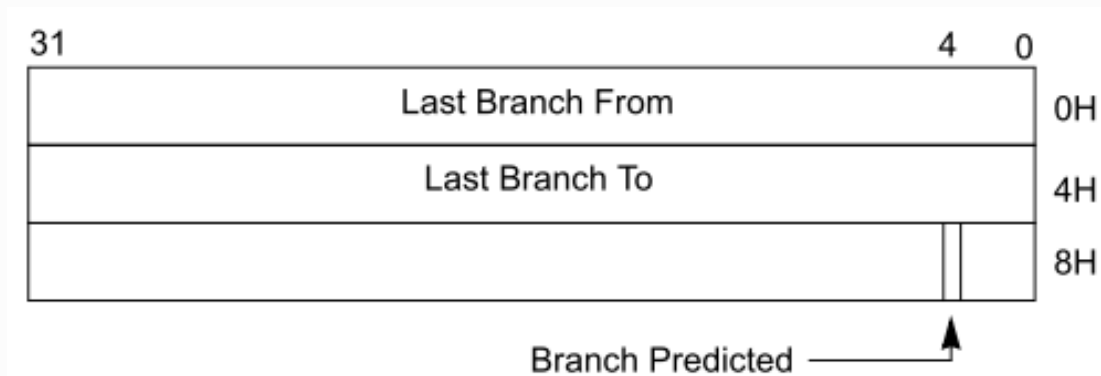
- **Host Machine** – runs the User Interface of the Software Debugger to the engineer
- **Target Platform** – runs the BIOS interface of the Software Debugger

# Innovation of Branch Tracing

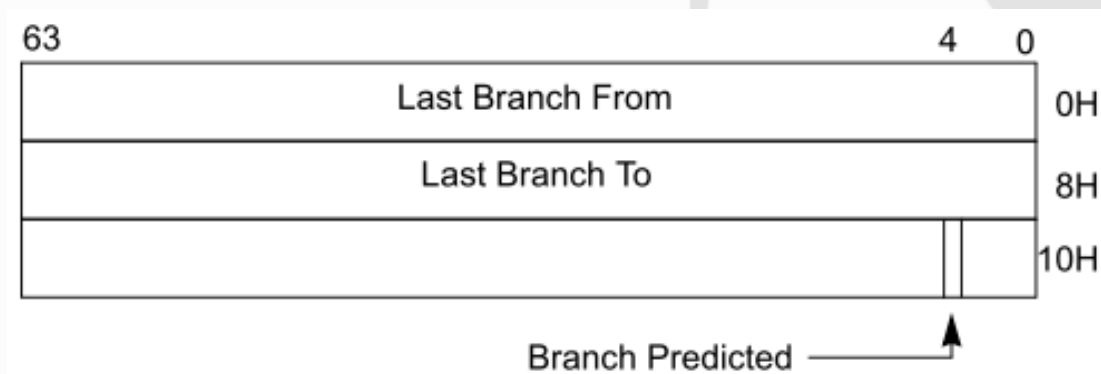


- Previously to trace, SW debuggers broke on either each instruction or at branches
  - On each break:
    - Transmit data to host or store locally
    - Check various debug control conditions
    - Restart next instruction
  - Very slow execution
- Now with Branch Tracing
  - SW debugger stops on a condition
  - Unwinds execution trace to previous IP or size of the Trace buffer
  - Target code executes much faster

# Branch Records



32-bit Branch Trace Record Format



64-bit Branch Trace Record Format

# Solving new problems



- More features can be created in a software debugger
  - Unravel call trees
  - Highlight areas called by Oss
  - Find long hardware I/O loops
  - Figure out where a crash started
  - Test sequence code coverage
  - Profile specific drivers



# Profiling Drivers





# Steps to profile a driver



## Update to Bios

Update debug engine to support Profiling



Build bios with non-optimized and generating debug info

## Debugger

Setup Profiling buffer size from Project Setting dialog



Enable/Disable Profiling from Profiling menu

## Log Analysis

Select Profiling Analyzer from Profiling menu to start profiling analysis



The reports will be displayed on Report view after completion

# Setup Trace Buffer sizes



Profiling Driver Options

PEI Buffer Size(24K ~ 4M)	200000
DXE Buffer Size(24K ~ 8M)	400000

OK Cancel

A screenshot of a BIOS dialog box titled "Profiling Driver Options". The dialog box has a purple border and contains two rows of settings. The first row is "PEI Buffer Size(24K ~ 4M)" with a value of "200000". The second row is "DXE Buffer Size(24K ~ 8M)" with a value of "400000". At the bottom of the dialog box are two buttons: "OK" on the left and "Cancel" on the right. A red oval is drawn around the entire dialog box content.

# Profiling Execution Flow Example



The screenshot displays two windows from the Insyde tool. The left window, titled 'Insyde - Execution Paths', shows a tree view of execution paths. The right window, titled 'Insyde - Code Coverage', shows the corresponding C code with executed lines highlighted in green.

**Execution path displayed in tree view**

- GenericMemoryTestEntryPoint
  - L(1):2398
  - L(0):2410
  - DxeInitializeDriverLib
    - L(1):47
    - L(0):50
    - EfiCommonLibSetMem
      - L(1):54
      - L(1):55
      - L(0):56
      - L(1):61
      - L(1):56
      - L(1):2414
    - EfiCommonLibSetMem
      - L(1):2419
      - L(0):2425
    - EfiLibGetSystemConfigurationTable
      - L(1):2426
      - > L(0):2431
      - > L(1):2446
      - > L(1):2431
      - > L(0):2436
      - > L(1):2446
      - > L(1):2431

**Executed code colored in green**

```
2410: DxeInitializeDriverLib (ImageHandle, SystemTable);
2411:
2412: mSelfHandle = ImageHandle;
2413:
2414: EfiZeroMem (&mGenericMemoryTestPrivate, sizeof (GENERIC_MEMORY_TEST_PRIVATE));
2415:
2416: //
2417: // Use the generic pattern to test compatible memory range
2418: //
2419: mGenericMemoryTestPrivate.MonoPattern = GenericMemoryTestMonoPattern;
2420: mGenericMemoryTestPrivate.MonoTestSize = GENERIC_CACHELINE_SIZE;
2421:
2422: //
2423: // Get the platform boot mode
2424: //
2425: Status = EfiLibGetSystemConfigurationTable (&gEfiHobListGuid, &HobList);
2426: if (EFI_ERROR (Status)) {
2427:     return Status;
2428: }
2429: //[-start-110427-IB02960376-add]//
2430: for (Hob.Raw = HobList; !END_OF_HOB_LIST(Hob); Hob.Raw = GET_NEXT_HOB(Hob)) {
2431:     if (GET_HOB_TYPE (Hob) == EFI_HOB_TYPE_RESOURCE_DESCRIPTOR) {
2432:         ResourceMemoryHob = Hob.ResourceDescriptor;
2433:
2434:         if (((ResourceMemoryHob->ResourceType == EFI_RESOURCE_FIRMWARE_DESCRIPTOR) ||
2435:             (ResourceMemoryHob->ResourceType == EFI_RESOURCE_MEMORY_DESCRIPTOR) ||
2436:             ((ResourceMemoryHob->ResourceAttribute) & EFI_RESOURCE_ATTRIBUTE_ALLOCATED) != 0) &
2437:             ResourceMemoryHob->PhysicalStart <= BaseAddress &
2438:             ResourceMemoryHob->PhysicalStart + ResourceMemoryHob->ResourceLength >= BaseAddress + Length) {
2439:             Status = gDS->GetMemorySpaceDescriptor (BaseAddress, Length, GcdDescriptor.Attributes | EFI_MEMORY_RUNTIME);
2440:             if (!EFI_ERROR (Status)) {
2441:                 Status = gDS->SetMemorySpaceAttributes (BaseAddress, Length, GcdDescriptor.Attributes | EFI_MEMORY_RUNTIME);
2442:             }
2443:         }
2444:     }
2445: }
2446:
2447: }
2448: }
2449: }
2450: }
```



# Code Coverage



# Need for Code Coverage



- Validate that all code is being tested
  - If code isn't being tested during validation, code path is either:
    - Unnecessary – remove it!
    - Not being tested – probably buggy, might cause problems in the field
      - Fix the test & test the code

# Code Coverage Example



Insyde - Execution Paths x Insyde - Code Coverage x

file:///D:/Projects/bt\_analyzer/bt\_analyzer/out/top.html

module/driver package	coverage		
	function	branch	state
S3Resume	0.00%		
MemoryQpInit	0.00%		
DxeIpl	0.00%		
FindFv	0.00%		

CoreAcquireGcdMemoryLock	280
CoreReleaseGcdMemoryLock	280
CoreAcquireGcdIoLock	0
CoreReleaseGcdIoLock	0
AlignValue	0
PageAlignAddress	0
PageAlignLength	0
CoreAllocateGcdMapEntry	
CoreInsertGcdMapEntry	
CoreMergeGcdMapEntry	
CoreCleanupGcdMapEntry	140
CoreSearchGcdMapEntry	280
CoreCommitGcdMapEntry	0
ConvertToCpuArchAttributes	140
CoreConvertSpace	140

```

475: |
476: | EFI_LIST_ENTRY *Link;
477: |
478: | if (TopEntry->Signature == 0) {
479: |   CoreFreePool (TopEntry);
480: | }
481: | if (BottomEntry->Signature == 0) {
482: |   CoreFreePool (BottomEntry);
483: | }
484: |
485: | Link = StartLink;
486: | while (Link != EndLink->ForwardLink) {
487: |   CoreMergeGcdMapEntry (Link, FALSE, Map);
488: |   Link = Link->ForwardLink;
489: | }
490: | CoreMergeGcdMapEntry (EndLink, TRUE, Map);
491: |
492: | return EFI_SUCCESS;
493: |
494: |
495: EFI_STATUS
496: CoreSearchGcdMapEntry (
497:   IN EFI_PHYSICAL_ADDRESS  BaseAddress,
498:   IN UINT64                Length,
499:   OUT EFI_LIST_ENTRY      **StartLink,
500:   OUT EFI_LIST_ENTRY      **EndLink,
501:   IN EFI_LIST_ENTRY       *Map
502: )
503: {
504:   /**+
505:   Routine Description:
506:     Search a segment of memory space in GCD map. The result is a range of GCD en
507:   Arguments:
508:     509:
509:     510:
510:     511: BaseAddress      - The start address of the segment.
511:     512:
512:     513: Length          - The length of the segment.
513:     514:
514:     515: StartLink       - The first GCD entry involves this segment of memory spac
515:

```

Executed

Not executed

undetermined

Coverage information by module/driver

Summary of functions in a module/driver

# Call to action



- Contact your Debugger vendor
  - Ask about their support for tracing code execution
- Profile your drivers to understand their execution flow
- Verify that your code is executed during your testing

Thanks for attending the  
UEFI US Fall Plugfest 2016



For more information on  
the Unified EFI Forum and  
UEFI Specifications, visit  
<http://www.uefi.org>

*presented by*







Backup

**References**

**Pointers to more information**

# References



- Material heavily borrowed from:
  - Intel 64, ia, 32 Architectures Software Developer Manual #325462
  - Insyde Software DDT Developer Manual
- Pointer
  - <http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html>
  - <https://www.insyde.com/products/developertools>