Using Performance Measurement Tool to Optimize UEFI Drivers and Systems

UEFI PlugFest – May 13-15, 2014
Seattle, WA
Presented by Jeff Bobzin (Insyde Software)
Agenda

- What is PerformancePkg?
- How Does It Work?
- Real World Example
- What if my IHV test platform does not have a PERF ROM available?
- Late-Loading PERF capability
Performance Measuring Package

• PerformancePkg is available Open Source -
  − Part of EDK2 project on sourceforge.net *(but not in UEFI spec)*
• What can it do?
  − Insert low-overhead measurement start/stop hooks in driver
    • *(low, but not zero overhead. Keep this in mind when nesting.)*
  − Duration measurements based on high accuracy CPU timer
  − Ongoing measurements are stored in central buffer
  − Later able to display results in UEFI Shell
Performance Protocol in Operation

- PEI Measurements
  - Saved in HOB

- DXE CORE
  - DxeCore
    - PerformanceLib

- In-memory database

- Driver
  - DxePerformanceLib
  - dp.efi

- Performance Protocol
PerformanceLib.h Defines PERF Macros

PERF_START_EX(Handle, Token, Module, TimeStamp, Identifier)

- **Handle**: Useful Identifier from context
- **Token**: ASCII String used to id component
- **Module**: ASCII String used to id module
- **TimeStamp**: [OPTIONAL] Caller-supplied start time (use current if 0)
- **Identifier**: Additional 32-bit constant used for class of similar measurement

**ALTERNATE:**

PERF_START(Handle, Token, Module, TimeStamp)

Equal to PERF_START_EX with Identifier as 0
End Measurements

PERF_END_EX(Handle, Token, Module, TimeStamp, Identifier)
PERF_END(Handle, Token, Module, TimeStamp)

PERF_END_EX, is matched to PERF_START_EX by Handle, Token, Module, Identifier
PERF Macros In Code

DxeMain.c

```
//
// Initialize the DXE Dispatcher
//
PERF_START (NULL,"CoreInitializeDispatcher", "DxeMain", 0) ;
CoreInitializeDispatcher ();
PERF_END (NULL,"CoreInitializeDispatcher", "DxeMain", 0) ;
```

Dp.efi

```
==[ General ]==========
Index   Name                 Description       Time (us)
------- -------- --------------- --------------- 
   3:          PreMem       2174540
  55:         PostMem      376006
   56:         DisMem       17215
   66:   DxeMain CoreInitializeDispatcher     3337
```
MACRO On/Off

This PCD turns On/Off the PERF MACROs, so they can remain in production source

gEfiMdePkgTokenSpaceGuid.PcdPerformanceLibraryPropertyMask|1
### dp.efi output snippets

**DP Build Version:** 2.3  
**System Performance Timer Frequency:** 2,296,380 (KHz)

---

**==[ Major Phases ]========**

<table>
<thead>
<tr>
<th>Phase</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEC</td>
<td>28701 (us)</td>
</tr>
<tr>
<td>PEI</td>
<td>2556 (ms)</td>
</tr>
<tr>
<td>DXE</td>
<td>15432 (ms)</td>
</tr>
<tr>
<td>BDS</td>
<td>1215 (ms)</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>19231 (ms)</strong></td>
</tr>
</tbody>
</table>

---

**==[ Drivers by Handle ]========**

<table>
<thead>
<tr>
<th>Index</th>
<th>Handle</th>
<th>Driver Name</th>
<th>Description</th>
<th>Time(us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>757</td>
<td>[ A0]</td>
<td>VariableRuntimeDxe</td>
<td>StartImage:</td>
<td>2394</td>
</tr>
<tr>
<td>868</td>
<td>[ AB]</td>
<td>SmbiosDxe</td>
<td>StartImage:</td>
<td>1820</td>
</tr>
<tr>
<td>1024</td>
<td>[ BD]</td>
<td>PiSmmIpl</td>
<td>StartImage:</td>
<td>2355</td>
</tr>
</tbody>
</table>
## Special Measurement Categories

### General

<table>
<thead>
<tr>
<th>Index</th>
<th>Name</th>
<th>Description</th>
<th>Time(us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3:</td>
<td>PreMem</td>
<td></td>
<td>2174540</td>
</tr>
<tr>
<td>55:</td>
<td>PostMem</td>
<td></td>
<td>376006</td>
</tr>
<tr>
<td>56:</td>
<td>DisMem</td>
<td></td>
<td>17215</td>
</tr>
<tr>
<td>66:</td>
<td>DxeMain</td>
<td>CoreInitializeDispatcher</td>
<td>3337</td>
</tr>
<tr>
<td>67:</td>
<td>DxeMain</td>
<td>CoreDispatcher</td>
<td>15041351</td>
</tr>
<tr>
<td>786:</td>
<td></td>
<td>StartImage:</td>
<td>2558</td>
</tr>
<tr>
<td>4165:</td>
<td>BDS</td>
<td>PlatformBds</td>
<td>1213860</td>
</tr>
<tr>
<td>54745:</td>
<td></td>
<td>PostBDS</td>
<td>284242</td>
</tr>
</tbody>
</table>

### Cumulative

<table>
<thead>
<tr>
<th>Name</th>
<th>Count</th>
<th>Duration</th>
<th>Average</th>
<th>Shortest</th>
<th>Longest</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadImage</td>
<td>206</td>
<td>36707</td>
<td>178</td>
<td>12</td>
<td>8653</td>
</tr>
<tr>
<td>StartImage</td>
<td>203</td>
<td>14780092</td>
<td>72808</td>
<td>2</td>
<td>14656754</td>
</tr>
<tr>
<td>DB:Start</td>
<td>63</td>
<td>1063861</td>
<td>16886</td>
<td>0</td>
<td>519179</td>
</tr>
<tr>
<td>DB:Support</td>
<td>54298</td>
<td>82441</td>
<td>1</td>
<td>0</td>
<td>57195</td>
</tr>
</tbody>
</table>
Adding A Cumulative Category

In dp.c

/// Items for which to gather cumulative statistics.

PERF_CUM_DATA CumData[] = {
    PERF_INIT_CUM_DATA (LOAD_IMAGE_TOK),
    PERF_INIT_CUM_DATA (START_IMAGE_TOK),
    PERF_INIT_CUM_DATA (DRIVERBINDING_START_TOK),
    PERF_INIT_CUM_DATA (DRIVERBINDING_SUPPORT_TOK)
    //add your custom cumulative here, and rebuild dp.efi
};
Is PERF Useful for IHVs?

• For System PERF, PerformanceProtocol must be enabled when DxeMain is built
• So what if no PERF build available for my test platform?
• Can IHV use PerformancePkg to measure IHV driver?
• Usage Goal:
  1. Boot To Shell
  2. Run Driver to Start Performance Protocol
  3. Test IHV Driver in Shell
PerfDxe.c

```c
#include <Uefi.h>
#include <Library/PerformanceLib.h>

EFI_STATUS
EFI_API
PerfDxeInit(
    IN EFI_HANDLE ImageHandle,
    IN EFI_SYSTEM_TABLE *SystemTable
)
{
    return EFI_SUCCESS;
}
```

The Driver C Source is simple. How does this enable PERF?
PerfDxe.inf

[Defines]
INF_VERSION                  = 0x00010006
BASE_NAME                    = PerfDxe
FILE_GUID                    = 8C270BB5-CEBB-4A39-BAFB-5BCA79303C77
MODULE_TYPE                  = UEFI_DRIVER
VERSION_STRING               = 1.0
ENTRY_POINT                  = PerfDxeInit

#
# The following information is for reference only and not required by the build tools.
#
# VALID_ARCHITECTURES         = IA32 X64 IPF EBC
#
[Sources]
  PerfDxe.c

[Packages]
  MdePkg/MdePkg.dec
  MdeModulePkg/MdeModulePkg.dec
  PerformancePkg/PerformancePkg.dec

[LibraryClasses]
  UefiDriverEntryPoint
  PerformanceLib

Nothing unusual in the inf either!
PerformancePkg.dsc Overrides Are the Key!

[Components]
PerformancePkg/Dp_App/Dp.inf

//add lines below ...
PerformancePkg/PerfDxe/PerfDxe.inf {

.LibraryClasses>
    UefiDriverEntryPoint|MdePkg/Library/UefiDriverEntryPoint/UefiDriverEntryPoint.inf
    TimerLib|PerformancePkg/Library/TscTimerLib/DxeTscTimerLib.inf

.PerformanceLib|MdeModulePkg/Library/DxeCorePerformanceLib/DxeCorePerformanceLib.inf

<PcdsFixedAtBuild>
    gEfiMdePkgTokenSpaceGuid.PcdPerformanceLibraryPropertyMask|1
    gEfiMdeModulePkgTokenSpaceGuid.PcdMaxPeiPerformanceLogEntries|64

build -p PerformancePkg\PerformancePkg.dsc -a X64 -t VS2010x86 -b DEBUG
PerfDxe.c – final version

```c
#include <Uefi.h>
#include <Library/UefiBootServicesTableLib.h>
#include <Library/PerformanceLib.h>

EFI_STATUS
EFIAPI
PerfDxeInit(
    IN EFI_HANDLE                            ImageHandle, 
    IN EFI_SYSTEM_TABLE                      *SystemTable   
)
{
    EFI_STATUS                       Status;
    EFI_EVENT                        Timeout;
    UINTN Index;                     
    Status = gBS->CreateEvent (EVT_TIMER,0,NULL,NULL,&Timeout);
    PERF_START_EX(ImageHandle,"CALIBRATE","PerfDxe",0,0);
    Status = gBS->SetTimer (Timeout, TimerRelative, 10000000);
    gBS->WaitForEvent(1,&Timeout,&Index);
    PERF_END_EX(ImageHandle,"CALIBRATE","PerfDxe",0,0);
    return EFI_SUCCESS;
}
```

Final Version Includes Calibration!
### dp.efi output after load of PerfDxe.efi

<table>
<thead>
<tr>
<th>Index</th>
<th>Handle</th>
<th>Driver Name</th>
<th>Description</th>
<th>Time (us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[1CC]</td>
<td>PerfDxe</td>
<td>CALIBRATE</td>
<td>991985</td>
</tr>
<tr>
<td>2</td>
<td>[1CF]</td>
<td>MyDriver</td>
<td>START</td>
<td>39567</td>
</tr>
</tbody>
</table>
Summary

• PerformancePkg is a EDK-II tool
• Measurement start/stop macros are added to driver source
• Helps to Identify slow sections
• If test platform is not PERF enabled, there is a Shell-Loading PERF capability
For more information on the Unified EFI Forum and UEFI Specifications, visit
http://www.uefi.org