



‘sct-next’ ***SCT for modern platforms***

Presented by Olivier Martin

presented by

ARM®

Agenda



UEFI SCT overview

Why 'sct-next'?

Steps toward 'sct-next'

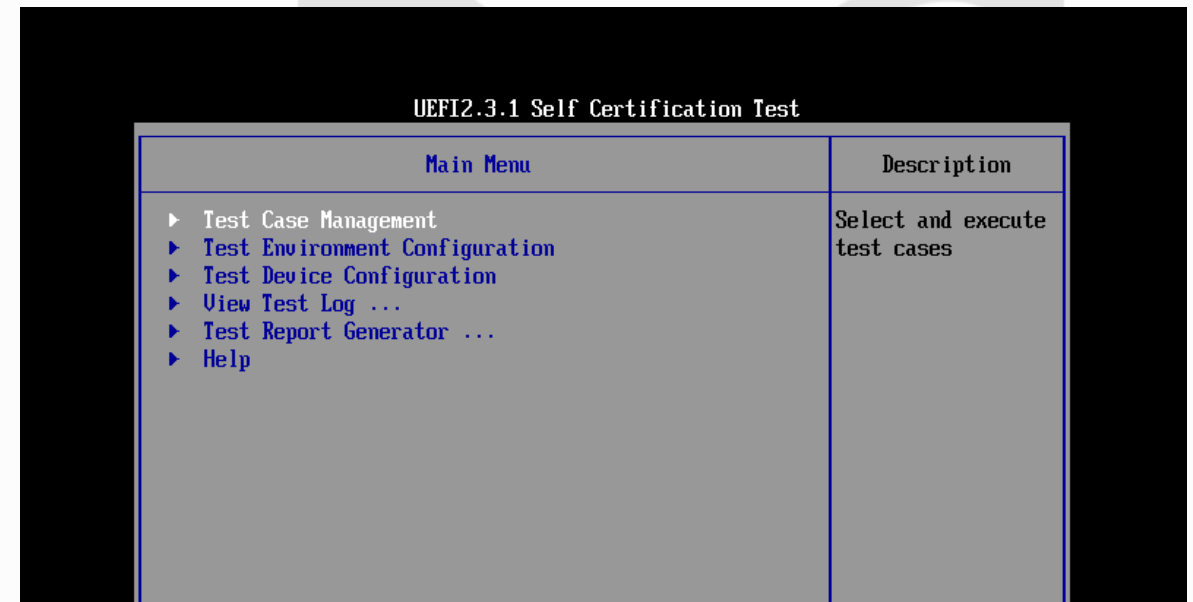
After 'sct-next'?



UEFI SCT Overview



- **Self Certification Test**
- Driven by **UEFI Test Working Group** – utwg@uefi.org

A screenshot of the UEFI2.3.1 Self Certification Test interface. The title bar reads 'UEFI2.3.1 Self Certification Test'. Below it is a table with two columns: 'Main Menu' and 'Description'. The 'Main Menu' column contains a list of options with right-pointing arrowheads: 'Test Case Management', 'Test Environment Configuration', 'Test Device Configuration', 'View Test Log ...', 'Test Report Generator ...', and 'Help'. The 'Description' column contains the text 'Select and execute test cases'.

What is SCT?



- A Test Framework running in UEFI
... a set of UEFI drivers and applications
- Test if the UEFI firmware complies with the latest UEFI specification



Why 'sct-next'?



SCT legacy



- Require EDK Shell (sometimes named as EFI Shell 1.0)

“I think EDK shell is not actively maintained now. We should migrate to UEFI shell ASAP.” (answer from one of the owner of EDK Shell)

- Require EdkCompatibilityPkg

- Accumulation of years of development



Steps toward 'sct-next'



New Development Process



- From an obscure ZIP file to an 'open' Github repository

- Allow code review, issue tracker
- Development more transparent to encourage contributions

- How to contribute:

<https://sourceforge.net/apps/mediawiki/tianocore/index.php?title=ArmPkg/HowToContributeSct>

Step 1: Clean the code



- Remove dead code & leftovers

```
327  **/  
328  {  
329      EFI_STATUS status;  
330      DEBUG ((EFI_D_INFO | EFI_D_LOAD, "~~~~~"));  
331      EfiInitializeTestLib (ImageHandle, SystemTable);  
332      InitializeLib (ImageHandle, SystemTable);
```

```
795  
796      if(EFI_ERROR(Status))  
797      {  
798          EntsPrint(L"Fuckn");  
799          EFI_DEADLOOP();  
800      }  
801
```



Step 1: Clean the code

- Remove dead code
- Remove duplicated implementation





an the code

- Remove duplication

The screenshot displays a code analysis tool interface with several overlapping windows showing source code. The windows include:

- ParseConf.c**: Shows a `MemSet` function signature with parameters `*Dest`, `Char`, and `Count`.
- misc.c**: Shows a `SetMem` function signature with parameters `*Buffer`, `Size`, and `Value`.
- PxeBaseCodeBBTestMain.c**: Shows a `MemSet` function signature with parameters `*b`, `c`, and `len`.
- TestProfileSupport.c**: Shows a `MemSet` function signature with parameters `*b`, `c`, and `len`.
- BlockIo2BBTestMain.c**: Shows a `MemSet` function signature with parameters `*b`, `c`, and `len`.
- BlockIo2BBTestMain.c**: Shows a `MemSet` function signature with parameters `*b`, `c`, and `len`.
- EntsMisc.c**: Shows a `EntsSetMem` function signature with parameters `*Buffer`, `Size`, and `Value`.

At the bottom, a search results pane shows:

- 'EfiSetMem' - 66 matches in working set 'SctPkg' (*.c)
- PxeBaseCodeBBTestFunction.c (4 matches)
 - 1,065: EfiSetMem (&BclpFilter, sizeof (BclpFilter), 0);
 - 1,143: EfiSetMem (&BclpFilter, sizeof (BclpFilter), 0);
 - 613: EfiSetMem (&BclpFilter, sizeof (BclpFilter), 0);
 - 827: EfiSetMem (&MacFilter, sizeof (MacFilter), 0);

Step 1: Clean the code



- Remove dead code
- Remove duplicated implementation
- New helper library

```
EfiTestUtilityLib.h
491
492 INTN
493 CompareGuid(
494     IN EFI_GUID          *Guid1
495     IN EFI_GUID          *Guid2
496 );
497
```

```
Debug.c
60
61 BOOLEAN
62 EfiCompareGuid (
63     IN EFI_GUID *Guid1,
64     IN EFI_GUID *Guid2
65 )
66
```

Step 2: Run SCT on EFI Shell 2.0



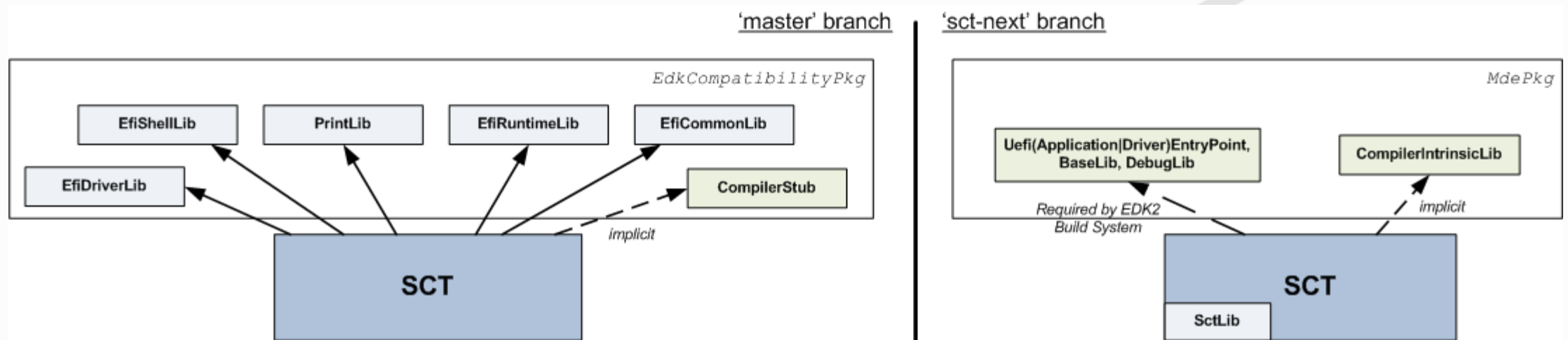
- Abstraction to still be able to run on EDK Shell



Step 3: Disconnect from EdkCompatibilityPkg



- Reduce dependency on the legacy framework



- Avoid to maintain legacy code

Step 4: Performance improvement



- Run SCT faster to run SCT more often
- To not use the excuse of the long SCT execution time to not contribute to SCT...



Step 5: Validation



- Automation
- Regression
- Debugging

The image shows a Jenkins web interface for a project named 'ap-uefi-sct-run'. The main area displays build parameters: 'SCT_SEQUENCE' set to 'ALL' and 'EDK2_SHELL' with a dropdown menu showing 'ALL', 'minimal.seq', and 'bench.seq'. A 'Build' button is visible. On the left, a 'Build History' table lists recent builds with their IDs and timestamps. Overlaid on the Jenkins interface are two debugger windows. The top window, 'DS-5 Debug', shows a hardware breakpoint at address 0x0B6E208C0 in the file 'uefi.c' at line 942. The bottom window, 'Fast Models - CLCD', displays the 'UEFI2.3.1 Self Certification Test' main menu with options: 'Test Case Management', 'Test Environment Configuration', and 'Test Device Configuration'.

Build #	Timestamp
#33	09-May-2014 16:41
#32	07-May-2014 12:16
#31	07-May-2014 11:56
#30	01-May-2014 21:45
#29	28-Apr-2014 15:48

Main Menu	Description
▶ Test Case Management	Select and execute test cases
▶ Test Environment Configuration	
▶ Test Device Configuration	

Results [1/2]



- Parity between the 'master' and 'sct-next' branches
- Found & Fix 12 defects in EFI Shell 2.0
- Found limitation in UEFI Shell 2.0 spec
- Tried (and documented) new Github process
- Inconsistencies in UEFI SCT with the UEFI spec

Results [2/2]



- `$ git diff master..HEAD --shortstat`

1745 files changed, **66**752 insertions(+), **96**577 deletions(-)

- Performance improvement ... work-in-progress



After 'sct-next'?



Next official SCT release?



- Need contribution from UEFI members

... what is a contribution?

- Code changes
- Code review
- Documentation
- Test



46 commits 2 branches 1 release 4 contributors

Your recently pushed branches:

sct-next (2 minutes ago) Compare & pull request

branch: master UEFI-SCT

```
$ git clone https://github.com/UEFI/UEFI-SCT
```

```
Cloning into 'UEFI-SCT'...
```

```
Username for 'https://github.com':
```

```
$ cd UEFI-SCT
```

```
$ git checkout sct-next
```

```
Switched to a new branch 'sct-next'
```

```
$
```

Future of 'sct-next'?



- Do we really need to run SCT from a **Shell**?
- Remove **ALL** the dependencies around *SctPkg*
- More than a **UEFI** spec conformance test...
 - **EFI Shell** spec?
 - **ACPI** spec?
 - Vendor specific test (eg: (**ARM**) **Server Base Specification Architecture**) ?

Questions ?!?



Resources:

- How to contribute to SCT?

<http://sourceforge.net/apps/mediawiki/tianocore/index.php?title=ArmPkg/HowToContributeSct>

- How to debug SCT?

<http://sourceforge.net/apps/mediawiki/tianocore/index.php?title=ArmPkg/Sct>

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



presented by

