Dynamic Tables Framework: A Step Towards Automatic Generation of Advanced Configuration and Power Interface (ACPI) & System Management BIOS (SMBIOS) Tables

Spring 2018 UEFI Seminar and Plugfest
March 26-30, 2018
Presented by Sami Mujawar (Arm Ltd.)
Agenda

• Introduction
• Dynamic Tables Framework
• Status & Plans
• Links
• Questions
Introduction
Background

- Number of firmware builds for a platform.

  `big/`  
  `Juno x big.LITTLE/ x Display/ x Debug/ = Firmware`  
  `LITTLE Headless Release`  
  `3 x 2 x 2 = 12`  
  `(MADT) (FADT) (6 ACPI variants)`

- Multiple firmware builds for similar platforms/hardware variants.
- Erroneous tables generated while handcrafting tables.
Goals

• Configurable firmware builds.
• Unify firmware build for similar platforms.
• Minimize/eliminate human induced errors.
• Ability to validate, and generate firmware that complies with relevant specifications.
Simplicity

• Select which ACPI tables to install.
• Provide hardware information.
• Framework generates the ACPI tables.
• Allow use of pre-generated tables.
Dynamic Tables Framework
Framework Architecture

Table Manager

ACPI Table Factory
- MADT
- GTDT
- Table<n>

SMBIOS Table Factory
- OEM
- RAW
- Type0
- Type1
- Type<n>

Key
- Platform Specific/OEM Modules
Configuration Manager

- Platform specific implementation.
- Collates the platform hardware information required for building the tables.
- Populates an abstract Platform Information Repository.
- Provides a list of tables to be installed.
- Implements the Configuration Manager Protocol that provides an interface to access the platform information.
Table Manager

<table>
<thead>
<tr>
<th>Table Manager</th>
<th>ACPI Table Factory</th>
<th>SMBIOS Table Factory</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MADT</td>
<td>OEM</td>
</tr>
<tr>
<td></td>
<td>GTDT</td>
<td>RAW</td>
</tr>
<tr>
<td>Table&lt;n&gt;</td>
<td></td>
<td>OEM</td>
</tr>
<tr>
<td>ACPI Table Generators</td>
<td>Standard/Generic Implementation</td>
<td>SMBIOS Table Generators</td>
</tr>
</tbody>
</table>

- Drives the table generation and installation.
  - Retrieves the list of tables to install from the Configuration Manager.
  - Gets the required table generators from the Table Factory.
  - Invokes the table generator to build the tables.
  - Installs the tables.
The table factory is responsible for managing the list of available table generators.

The table generators register with the table factory.
Table Generators

- Implement table specific logic for constructing the tables.
- Uses Configuration Manager Protocol interface to fetch the required platform information for building the table.
- Perform checks to see if the information provided to generate the table is consistent with the specifications.
### Table Generator Types

**Standard**
- Generic implementation
- RAW

**OEM**
- Allows customization

<table>
<thead>
<tr>
<th>ACPI Table Generators</th>
<th>SMBIOS Table Generators</th>
</tr>
</thead>
<tbody>
<tr>
<td>MADT</td>
<td>Type0</td>
</tr>
<tr>
<td>GTDT</td>
<td>Type1</td>
</tr>
<tr>
<td>Table&lt;n&gt;</td>
<td>Type&lt;n&gt;</td>
</tr>
</tbody>
</table>

**Platform Specific/OEM Modules**

**Key**
- Platform Specific/OEM Modules
Table generation sequence

:Dynamic Firmware Table Manager
:ACPI Table Factory
:Table Factory
:Configuration Manager
UEFI

Loop

Table List

Get Table List

Get Generator (e.g. MADT)

Build Table

Get Hardware1 Info

Get Hardware2 Info

ACPI Table Protocol InstallAcpiTable
Status & Plans
Current Status

• Prototyped on Juno and FVP model platforms.
• ACPI 6.2 with support for the following tables:
  – DBG2
  – FADT
  – GTDT
  – IORT
  – MADT
  – MCFG
  – SPCR
  – RAW (DSDT/SSDT)
Future Plans

• Supporting more ACPI tables.
• Hardware information parser.
• Adding SMBIOS support.
Hardware Information Parser

• Parser capable of loading hardware information from a suitable hardware description format represented using, for example, XML or JSON.
• Enables firmware customisation from generated system descriptions.
Links

• Dynamic Tables Framework
  https://github.com/tianocore/edk2-staging/tree/dynamictables
  https://github.com/tianocore/edk2-platforms/tree/devel-dynamictables

• ACPICA Patch

• ACPIview
Questions?
Impact?

• Memory footprint?
  – RAM requirement may increase somewhat.
    • Memory footprint only increases at boot time (no runtime services impact).
  – Image Size may be slightly increased.
    • Can be optimized by excluding generators that are not required for the platform.

• Boot time?
  – Possible optimization using:
    • Cached information.
    • Tables pre-generated using dynamic tables framework, possibly extracted using ACPIview.
Thanks for attending the Spring 2018 UEFI Plugfest

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

presented by

arm