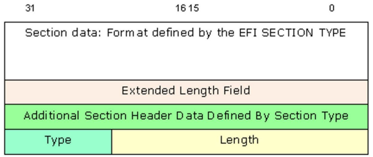
UEFI Shell Applications and Diagnostic Tools

Problem:

The Unified Extensible Firmware Interface (UEFI) defines the set of interfaces and data structures that the platform firmware must implement. It also describes the set of interfaces and data structures that operating systems may use in booting. The UEFI provides data tables that contain platform related information and boot and runtime service calls that are available to the operating system loader and the operating system. The goal of this project is to build custom firmware images and UEFI Shell applications that can be loaded and executed from the UEFI Shell.

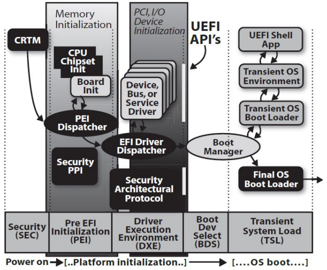
Approach:

We used a software toolchain provided by Intel to compile, assemble, and link C source code files into PE32/PE32+/COFF images that were processed to EFI format. We used a Python script that would parse tokens in DEC, DSC, and INF metadata files to generate .efi executable images. We created UEFI Shell applications for CPUID output processing, memory profiling, and SMI handler profiling.

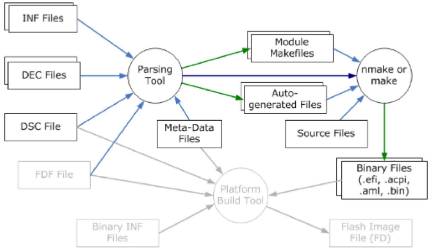


31
16 15
0
Section data: Format defined by the EFI SECTION TYPE
Extended Length Field
Additional Section Header Data Defined By Section Type
Type
Length

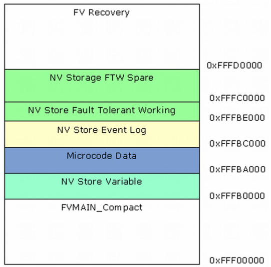
General EFI Section Format



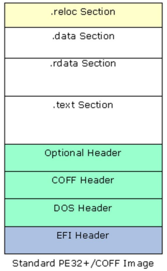
Platform Initialization (PI) Boot Sequence



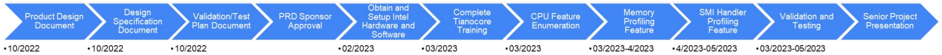
Platform build process flow



Typical IA32/X64 Flash Device Layout



Standard PE32+/COFF Image Format


- Product Design Document • 10/2022
- Design Specification Document • 10/2022
- Validation/Test Plan Document • 10/2022
- PRD Sponsor Approval
- Obtain and Setup Intel Hardware and Software • 02/2023
- Complete Tianocore Training • 03/2023
- CPU Feature Enumeration • 03/2023
- Memory Profiling Feature • 03/2023-4/2023
- SMI Handler Profiling Feature • 4/2023-05/2023
- Validation and Testing • 03/2023-05/2023
- Senior Project Presentation

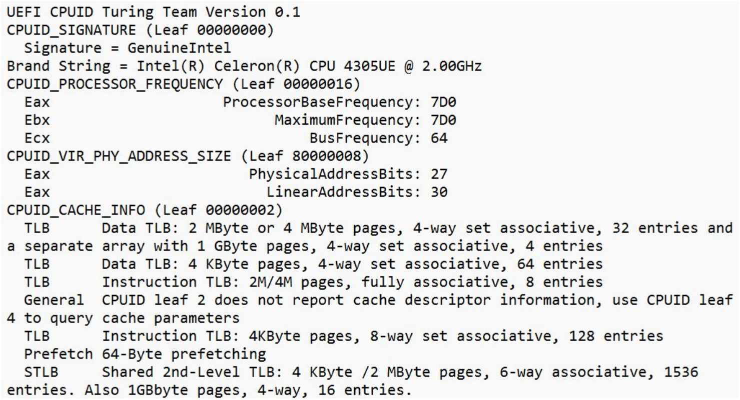
Hardware Materials:

- Aaeon UP Xtreme Board and Power Supply
- USB and Serial cable
- USB to TTL 3.3V UART Converter Cable with FTDI
- Test lead wire for FTDI 2 10 Pin USB Up Xtreme
- USB 3.0 Super-Speed A/A Debugging Cable
- Acclamator 256 GB SSD

Software Materials:

- Visual Studio
- Python
- Git
- NASM
- IASL
- Simics
- Tera Term

Results:



```
UEFI CPUID Turing Team Version 0.1
CPUID_SIGNATURE (Leaf 00000000)
  Signature = GenuineIntel
Brand String = Intel(R) Celeron(R) CPU 4305UE @ 2.00GHz
CPUID_PROCESSOR_FREQUENCY (Leaf 00000016)
  Eax                    ProcessorBaseFrequency: 7D0
  Ebx                          MaximumFrequency: 7D0
  Ecx                              BusFrequency: 64
CPUID_VIR_PHY_ADDRESS_SIZE (Leaf 80000008)
  Eax                       PhysicalAddressBits: 27
  Eax                        LinearAddressBits: 30
CPUID_CACHE_INFO (Leaf 00000002)
  TLB       Data TLB: 2 MByte or 4 MByte pages, 4-way set associative, 32 entries and
a separate array with 1 GByte pages, 4-way set associative, 4 entries
  TLB       Data TLB: 4 KByte pages, 4-way set associative, 64 entries
  TLB       Instruction TLB: 2M/4M pages, fully associative, 8 entries
  General   CPUID leaf 2 does not report cache descriptor information, use CPUID leaf
4 to query cache parameters
  TLB       Instruction TLB: 4KByte pages, 8-way set associative, 128 entries
  Prefetch 64-Byte prefetching
  STLB      Shared 2nd-Level TLB: 4 KByte /2 MByte pages, 6-way associative, 1536
entries. Also 1GBbyte pages, 4-way, 16 entries.
```

UpXtreme UEFI Shell Cpuid.efi Console Output