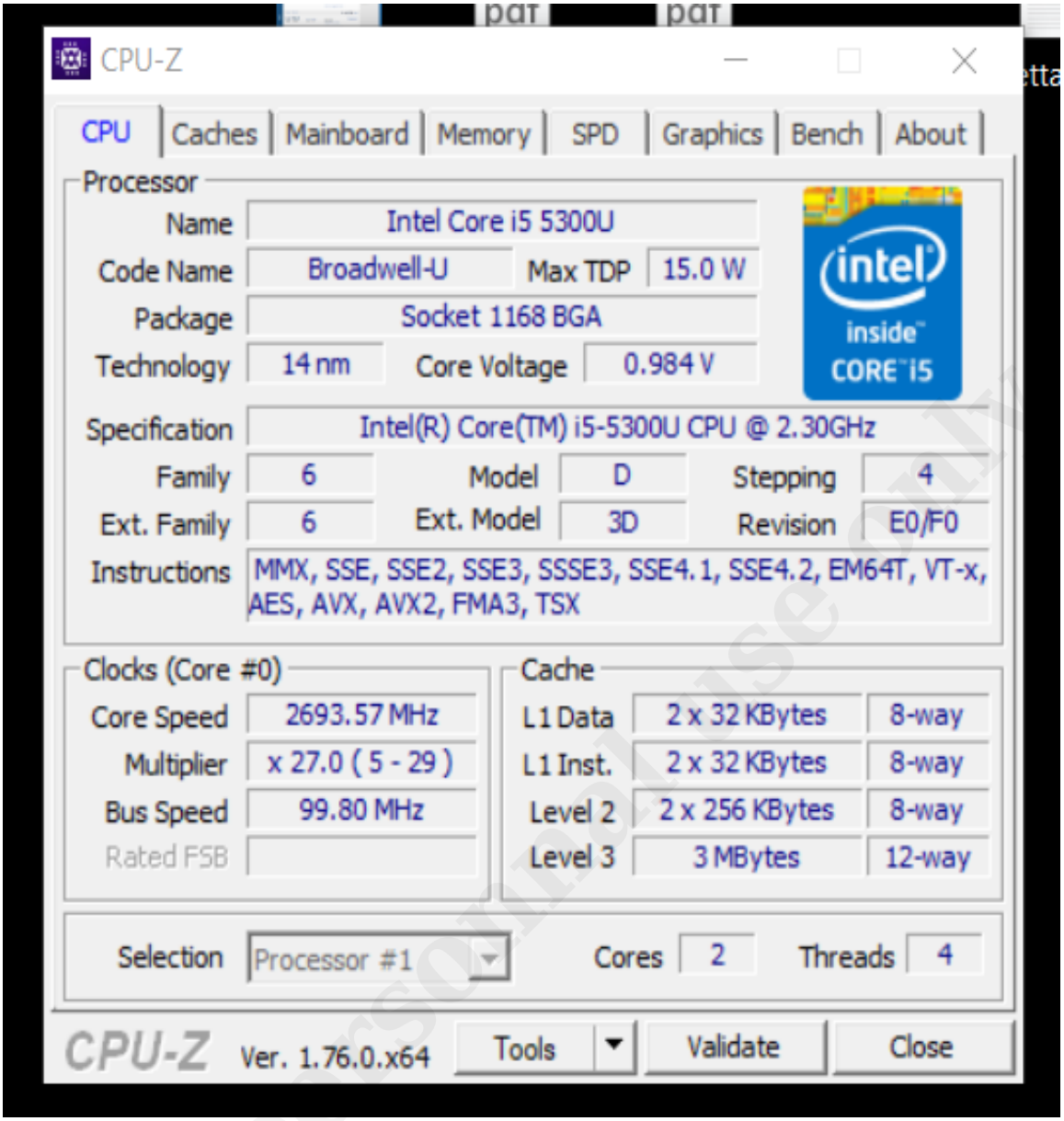
Simple CPUID Utility For UEFI Shell

Finnbarr P. Murphy
(fpm@fpmurphy.com)

The *CPUID* (CPU IDentification) opcode (*OFA2*) is a processor supplementary instruction for the IA32 and IA64 Intel architectures which enables software to determine processor type and the presence or absence of specific processor features. It was first implemented by Intel in the 1993 Pentium processor.

CPUID returns processor identification and feature information in the *EAX*, *EBX*, *ECX*, and *EDX* registers, as determined by input entered in *EAX* (and *ECX* in some cases.) On IA64 processors, *CPUID* clears the high 32 bits of the *RAX*, *RBX*, *RCX* and RDX registers in all modes. No flags are affected.

There are many utilities available which use *CPUID* to display processor-specific information on both Microsoft Windows and GNU/Linux platforms. One of my favorites is *CPU-Z*.



For many years there was no utility like *CPU-Z* available for use from the UEFI shell. In 2016, the situation improved because Intel contributed the source code for a UEFI application to display *CPUID* leaf information to TianoCore. The source code for this application can be found under *.../UefiCpuPkg/Application/Cpuid*.

Here is sample output from this utility:

```
UEFI CPUID Version 0.5
CPUID_SIGNATURE (Leaf 00000000)
  EAX:00000014  EBX:756E6547  ECX:6C65746E  EDX:49656E69
  Eax                              MaximumLeaf: 14
  Signature = GenuineIntel
CPUID_VERSION_INFO (Leaf 00000001)
  EAX:000306D4  EBX:00100800  ECX:77FAFBFF  EDX:BFEBFBFF
  Family = 6  Model = 3D  Stepping = 4
  Eax                                SteppingId: 4
  Eax                                     Model: D
  Eax                                  FamilyId: 6
  Eax                             ProcessorType: 0
  Eax                           ExtendedModelId: 3
  Eax                           ExtendedFamilyId: 0
```



```
  Ebx                                BrandIndex: 0
  Ebx                             CacheLineSize: 8
  Ebx MaximumAddressableIdsForLogicalProcessors: 10
  Ebx                         InitialLocalApicId: 0
  Ecx                                      SSE3: 1
  Ecx                                   PCLMULQDQ: 1
  Ecx                                      DTES64: 1
  Ecx                                     MONITOR: 1
  Ecx                                      DS_CPL: 1
  Ecx                                         VMX: 1
  Ecx                                         SMX: 1
  Ecx                                         TM2: 1
  Ecx                                       SSSE3: 1
  Ecx                                     CNXT_ID: 0
  Ecx                                        SDBG: 1
  Ecx                                         FMA: 1
  Ecx                                   CMPXCHG16B: 1
  Ecx                         xTPR_Update_Control: 1
  Ecx                                        PDCM: 1
  Ecx                                        PCID: 1
  Ecx                                         DCA: 0
  Ecx                                      SSE4_1: 1
  Ecx                                      SSE4_2: 1
  Ecx                                      x2APIC: 1
  Ecx                                       MOVBE: 1
  Ecx                                      POPCNT: 1
  Ecx                                TSC_Deadline: 1
  Ecx                                        AESNI: 1
  Ecx                                        XSAVE: 1
  Ecx                                     OSXSAVE: 0
  Ecx                                         AVX: 1
  Ecx                                        F16C: 1
  Ecx                                      RDRAND: 1
  Edx                                         FPU: 1
  Edx                                         VME: 1
  Edx                                          DE: 1
  Edx                                         PSE: 1
  Edx                                         TSC: 1
  Edx                                         MSR: 1
  Edx                                         PAE: 1
  Edx                                         MCE: 1
  Edx                                         CX8: 1
  Edx                                        APIC: 1
  Edx                                         SEP: 1
  Edx                                        MTRR: 1
  Edx                                         PGE: 1
  Edx                                         MCA: 1
  Edx                                        CMOV: 1
  Edx                                         PAT: 1
  Edx                                      PSE_36: 1
  Edx                                         PSN: 0
  Edx                                       CLFSH: 1
  Edx                                          DS: 1
  Edx                                        ACPI: 1
  Edx                                         MMX: 1
  Edx                                        FXSR: 1
  Edx                                         SSE: 1
  Edx                                        SSE2: 1
  Edx                                          SS: 1
  Edx                                         HTT: 1
  Edx                                          TM: 1
  Edx                                         PBE: 1
CPUID_CACHE_INFO (Leaf 00000002)
  EAX:76036301  EBX:00F0B5FF  ECX:00000000  EDX:00C30000
  TLB      Data TLB: 2 MByte or 4 MByte pages, 4-way set associative, 32 entries and a sep
arate 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, 64 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.
CPUID_SERIAL_NUMBER (Leaf 00000003)
  Not Supported
CPUID_CACHE_PARAMS (Leaf 00000004, Sub-Leaf 00000000)
  EAX:1C004121  EBX:01C0003F  ECX:0000003F  EDX:00000000
  Eax                                 CacheType: 1
  Eax                                CacheLevel: 1
  Eax                    SelfInitializingCache: 1
  Eax                    FullyAssociativeCache: 0
  Eax MaximumAddressableIdsForLogicalProcessors: 1
  Eax    MaximumAddressableIdsForProcessorCores: 7
  Ebx                                  LineSize: 3F
  Ebx                             LinePartitions: 0
  Ebx                                       Ways: 7
  Ecx                               NumberOfSets: 3F
  Edx                                 Invalidate: 0
  Edx                           CacheInclusiveness: 0
  Edx                         ComplexCacheIndexing: 0
CPUID_CACHE_PARAMS (Leaf 00000004, Sub-Leaf 00000001)
  EAX:1C004122  EBX:01C0003F  ECX:0000003F  EDX:00000000
  Eax                                 CacheType: 2
  Eax                                CacheLevel: 1
  Eax                    SelfInitializingCache: 1
  Eax                    FullyAssociativeCache: 0
  Eax MaximumAddressableIdsForLogicalProcessors: 1
  Eax    MaximumAddressableIdsForProcessorCores: 7
  Ebx                                  LineSize: 3F
  Ebx                             LinePartitions: 0
  Ebx                                       Ways: 7
  Ecx                               NumberOfSets: 3F
  Edx                                 Invalidate: 0
  Edx                           CacheInclusiveness: 0
  Edx                         ComplexCacheIndexing: 0
CPUID_CACHE_PARAMS (Leaf 00000004, Sub-Leaf 00000002)
  EAX:1C004143  EBX:01C0003F  ECX:000001FF  EDX:00000000
  Eax                                 CacheType: 3
  Eax                                CacheLevel: 2
  Eax                    SelfInitializingCache: 1
  Eax                    FullyAssociativeCache: 0
  Eax MaximumAddressableIdsForLogicalProcessors: 1
  Eax    MaximumAddressableIdsForProcessorCores: 7
  Ebx                                  LineSize: 3F
  Ebx                             LinePartitions: 0
  Ebx                                       Ways: 7
  Ecx                               NumberOfSets: 1FF
  Edx                                 Invalidate: 0
  Edx                           CacheInclusiveness: 0
  Edx                         ComplexCacheIndexing: 0
CPUID_CACHE_PARAMS (Leaf 00000004, Sub-Leaf 00000003)
  EAX:1C03C163  EBX:02C0003F  ECX:00000FFF  EDX:00000006
  Eax                                 CacheType: 3
  Eax                                CacheLevel: 3
  Eax                    SelfInitializingCache: 1
  Eax                    FullyAssociativeCache: 0
  Eax MaximumAddressableIdsForLogicalProcessors: F
  Eax    MaximumAddressableIdsForProcessorCores: 7
  Ebx                                  LineSize: 3F
  Ebx                             LinePartitions: 0
  Ebx                                       Ways: B
  Ecx                               NumberOfSets: FFF
  Edx                                 Invalidate: 0
  Edx                           CacheInclusiveness: 1
  Edx                         ComplexCacheIndexing: 1
```



```
CPUID_MONITOR_MWAIT (Leaf 00000005)
  EAX:00000040  EBX:00000040  ECX:00000003  EDX:11142120
  Eax                    SmallestMonitorLineSize: 40
  Ebx                     LargestMonitorLineSize: 40
  Ecx                        ExtensionsSupported: 1
  Ecx                           InterruptAsBreak: 1
  Edx                                   C0States: 0
  Edx                                   C1States: 2
  Edx                                   C2States: 1
  Edx                                   C3States: 2
  Edx                                   C4States: 4
  Edx                                   C5States: 1
  Edx                                   C6States: 1
  Edx                                   C7States: 1
CPUID_THERMAL_POWER_MANAGEMENT (Leaf 00000006)
  EAX:00000077  EBX:00000002  ECX:00000009  EDX:00000000
  Eax                   DigitalTemperatureSensor: 1
  Eax                       TurboBoostTechnology: 1
  Eax                                       ARAT: 1
  Eax                                        PLN: 1
  Eax                                       ECMD: 1
  Eax                                        PTM: 1
  Eax                                        HWP: 0
  Eax                           HWP_Notification: 0
  Eax                         HWP_Activity_Window: 0
  Eax         HWP_Energy_Performance_Preference: 0
  Eax                 HWP_Package_Level_Request: 0
  Eax                                       HDC: 0
  Ebx                       InterruptThresholds: 2
  Ecx              HardwareCoordinationFeedback: 1
  Ecx                       PerformanceEnergyBias: 1
CPUID_STRUCTURED_EXTENDED_FEATURE_FLAGS (Leaf 00000007, Sub-Leaf 00000000)
  EAX:00000000  EBX:021C2FBB  ECX:00000000  EDX:00000000
  Ebx                                   FSGSBASE: 1
  Ebx                              IA32_TSC_ADJUST: 1
  Ebx                                       SGX: 0
  Ebx                                      BMI1: 1
  Ebx                                       HLE: 1
  Ebx                                      AVX2: 1
  Ebx                              FDP_EXCPTN_ONLY: 0
  Ebx                                      SMEP: 1
  Ebx                                      BMI2: 1
  Ebx                        EnhancedRepMovsbStosb: 1
  Ebx                                    INVPCID: 1
  Ebx                                        RTM: 1
  Ebx                                     RDT_M: 0
  Ebx                           DeprecateFpuCsDs: 1
  Ebx                                       MPX: 0
  Ebx                                     RDT_A: 0
  Ebx                                    RDSEED: 1
  Ebx                                       ADX: 1
  Ebx                                      SMAP: 1
  Ebx                                  CLFLUSHOPT: 0
  Ebx                                       CLWB: 0
  Ebx                          IntelProcessorTrace: 1
  Ebx                                       SHA: 0
  Ecx                                PREFETCHWT1: 0
  Ecx                                       UMIP: 0
  Ecx                                        PKU: 0
  Ecx                                      OSPKE: 0
  Ecx                                      MAWAU: 0
  Ecx                                      RDPID: 0
  Ecx                                    SGX_LC: 0
CPUID_DIRECT_CACHE_ACCESS_INFO (Leaf 00000009)
  EAX:00000000  EBX:00000000  ECX:00000000  EDX:00000000
CPUID_ARCHITECTURAL_PERFORMANCE_MONITORING (Leaf 0000000A)
  EAX:07300403  EBX:00000000  ECX:00000000  EDX:00000603
  Eax                           ArchPerfMonVerID: 3
```



```
  Eax               PerformanceMonitorCounters: 4
  Eax            PerformanceMonitorCounterWidth: 30
  Eax                        EbxBitVectorLength: 7
  Ebx                        UnhaltedCoreCycles: 0
  Ebx                       InstructionsRetired: 0
  Ebx                  UnhaltedReferenceCycles: 0
  Ebx                 LastLevelCacheReferences: 0
  Ebx                     LastLevelCacheMisses: 0
  Ebx                BranchInstructionsRetired: 0
  Ebx               AllBranchMispredictRetired: 0
  Edx          FixedFunctionPerformanceCounters: 3
  Edx      FixedFunctionPerformanceCounterWidth: 30
CPUID_EXTENDED_TOPOLOGY (Leaf 0000000B, Sub-Leaf 00000000)
  EAX:00000001  EBX:00000002  ECX:00000100  EDX:00000000
  Eax                              ApicIdShift: 1
  Ebx                         LogicalProcessors: 2
  Ecx                              LevelNumber: 0
  Ecx                                LevelType: 1
  Edx                                x2APIC_ID: 0
CPUID_EXTENDED_TOPOLOGY (Leaf 0000000B, Sub-Leaf 00000001)
  EAX:00000004  EBX:00000004  ECX:00000201  EDX:00000000
  Eax                              ApicIdShift: 4
  Ebx                         LogicalProcessors: 4
  Ecx                              LevelNumber: 1
  Ecx                                LevelType: 2
  Edx                                x2APIC_ID: 0
CPUID_EXTENDED_STATE (Leaf 0000000D, Sub-Leaf 00000000)
  EAX:00000007  EBX:00000240  ECX:00000340  EDX:00000000
  Eax                                     x87: 1
  Eax                                     SSE: 1
  Eax                                     AVX: 1
  Eax                                     MPX: 0
  Eax                                 AVX_512: 0
  Eax                                IA32_XSS: 0
  Eax                                    PKRU: 0
  Ebx                     EnabledSaveStateSize: 240
  Ecx                   SupportedSaveStateSize: 340
  Edx                     XCR0_Supported_32_63: 0
CPUID_EXTENDED_STATE (Leaf 0000000D, Sub-Leaf 00000001)
  EAX:00000001  EBX:00000000  ECX:00000000  EDX:00000000
  Eax                               XSAVEOPT: 1
  Eax                                 XSAVEC: 0
  Eax                                 XGETBV: 0
  Eax                                 XSAVES: 0
  Ebx        EnabledSaveStateSize_XCR0_IA32_XSS: 0
  Ecx                                   XCR0: 0
  Ecx                                     PT: 0
  Ecx                                 XCR0_1: 0
  Edx                 IA32_XSS_Supported_32_63: 0
CPUID_INTEL_RDT_MONITORING (Leaf 0000000F, Sub-Leaf 00000000)
  EAX:00000000  EBX:00000000  ECX:00000000  EDX:00000000
  Ebx                       Maximum_RMID_Range: 0
  Edx                             L3CacheRDT_M: 0
CPUID_INTEL_RDT_MONITORING (Leaf 0000000F, Sub-Leaf 00000001)
  EAX:00000000  EBX:00000000  ECX:00000000  EDX:00000000
  Ebx                OccupancyConversionFactor: 0
  Ecx                       Maximum_RMID_Range: 0
  Edx               L3CacheOccupancyMonitoring: 0
  Edx           L3CacheTotalBandwidthMonitoring: 0
  Edx           L3CacheLocalBandwidthMonitoring: 0
CPUID_INTEL_RDT_ALLOCATION (Leaf 00000010, Sub-Leaf 00000000)
  EAX:00000000  EBX:00000000  ECX:00000000  EDX:00000000
  Ebx                         L3CacheAllocation: 0
  Ebx                         L2CacheAllocation: 0
CPUID_INTEL_RDT_ALLOCATION (Leaf 00000010, Sub-Leaf 00000001)
  EAX:00000000  EBX:00000000  ECX:00000000  EDX:00000000
  Eax                            CapacityLength: 0
  Ebx                      AllocationUnitBitMap: 0
```



```
  Ecx                     CosUpdatesInfrequent: 0
  Ecx                   CodeDataPrioritization: 0
  Edx                        HighestCosNumber: 0
CPUID_INTEL_RDT_ALLOCATION (Leaf 00000010, Sub-Leaf 00000002)
  EAX:00000000  EBX:00000000  ECX:00000000  EDX:00000000
  Eax                           CapacityLength: 0
  Ebx                     AllocationUnitBitMap: 0
  Edx                        HighestCosNumber: 0
CPUID_INTEL_PROCESSOR_TRACE (Leaf 00000014, Sub-Leaf 00000000)
  EAX:00000000  EBX:00000001  ECX:00000001  EDX:00000000
  Eax                           MaximumSubLeaf: 0
  Ebx                                Cr3Filter: 1
  Ebx                            ConfigurablePsb: 0
  Ebx                     IpTraceStopFiltering: 0
  Ebx                                      Mtc: 0
  Ebx                                  PTWrite: 0
  Ebx                           PowerEventTrace: 0
  Ecx                                     RTIT: 1
  Ecx                                     ToPA: 0
  Ecx                        SingleRangeOutput: 0
  Ecx                   TraceTransportSubsystem: 0
  Ecx                                      LIP: 0
CPUID_EXTENDED_FUNCTION (Leaf 80000000)
  EAX:80000008  EBX:00000000  ECX:00000000  EDX:00000000
  Eax                  MaximumExtendedFunction: 80000008
CPUID_EXTENDED_CPU_SIG (Leaf 80000001)
  EAX:00000000  EBX:00000000  ECX:00000121  EDX:2C100800
  Ecx                               LAHF_SAHF: 1
  Ecx                                   LZCNT: 1
  Ecx                               PREFETCHW: 1
  Edx                          SYSCALL_SYSRET: 1
  Edx                                      NX: 1
  Edx                                 Page1GB: 1
  Edx                                  RDTSCP: 1
  Edx                                      LM: 1
CPUID_BRAND_STRING1 (Leaf 80000002)
  EAX:65746E49  EBX:2952286C  ECX:726F4320  EDX:4D542865
CPUID_BRAND_STRING2 (Leaf 80000003)
  EAX:35692029  EBX:3033352D  ECX:43205530  EDX:40205550
CPUID_BRAND_STRING3 (Leaf 80000004)
  EAX:332E3220  EBX:7A484730  ECX:00000000  EDX:00000000
Brand String = Intel(R) Core(TM) i5-5300U CPU @ 2.30GHz
CPUID_EXTENDED_CACHE_INFO (Leaf 80000006)
  EAX:00000000  EBX:00000000  ECX:01006040  EDX:00000000
  Ecx                         CacheLineSize: 40
  Ecx                       L2Associativity: 6
  Ecx                             CacheSize: 100
CPUID_EXTENDED_TIME_STAMP_COUNTER (Leaf 80000007)
  EAX:00000000  EBX:00000000  ECX:00000000  EDX:00000100
  Edx                           InvariantTsc: 1
CPUID_VIR_PHY_ADDRESS_SIZE (Leaf 80000008)
  EAX:00003027  EBX:00000000  ECX:00000000  EDX:00000000
  Eax                     PhysicalAddressBits: 27
  Eax                       LinearAddressBits: 30
```

As you can see from the above output, the application is very verbose and output is displayed by leaf (and sub-leaf if appplicable.)

As far as I am concerned the output from the Intel application is too verbose to be useful for most people. For this reason, I wrote my own version of *Cpuid* which only displays the most useful *CPUID* information in a terse format.

Here is the output from my version of *Cpuid* for the same processor:

```
```



```
fs1:> cpuid.efi

    Signature: GenuineIntel
   CPU String: Intel(R) Core(TM) i5-5300U CPU @ 2.30GHz
       Family: 0x6
        Model: 0x3D
     Stepping: 0x4
     Features: ACPI AESNT APIC AVX CLFSH CMOV CMPXCHG16B CX8 DE DS DS_CPL
               DTES64 F16C FMA FPU FXSR HTT LAHF_SAHF LM LZCNT MCA MCE
               MONITOR MMX MOVBE MSR MTRR NX PAE PAGE1GB PAT PBE PCID
               PCLMULQDQ PDCM PGE POPCNT PREFETCHW PSE PSE_36 RDRAND
               RDTSCP SDBG SEP SMX SS SSE SSE2 SSE3 SSE4_1 SSE4_2 SSSE3
               SYSCALL_SYSRET TSC TSC_DEADLINE TM TM2 VME VMX X2APIC XSAVE
               XTPR_UPDATE_CONTROL
fs1:>
```

Here is the source code for the utility:

```


//
//  Copyright (c) 2017 - 2018   Finnbarr P. Murphy.   All rights reserved.
//  Portions Copyright (c) 2016, Intel Corporation.   All rights reserved.
//
//  Show concise CPUID information about a processsor.
//
//  License: BSD license applies to code copyrighted by Intel Corporation.
//           BSD 2 clause license applies to all other code.
//
//
#include <Uefi.h>
#include <Library/BaseLib.h>
#include <Library/UefiLib.h>
#include <Library/BaseMemoryLib.h>
#include <Library/MemoryAllocationLib.h>
#include <Register/Cpuid.h>
#define UTILITY_VERSION L"20180403"
#define WIDTH 60
#undef DEBUG
//
// Display Processor Signature
//
VOID
ProcessorSignature( VOID )
{
    UINT32 Eax, Ebx, Ecx, Edx;
    CHAR8  Signature[13];
    AsmCpuid( CPUID_SIGNATURE, &amp;Eax, &amp;Ebx, &amp;Ecx, &amp;Edx );
#ifdef DEBUG
    Print(L"  EAX:%08x  EBX:%08x  ECX:%08x  EDX:%08x\n", Eax, Ebx, Ecx, Edx);
#endif
    *(UINT32 *)(Signature + 0) = Ebx;
    *(UINT32 *)(Signature + 4) = Edx;
    *(UINT32 *)(Signature + 8) = Ecx;
    Signature[12] = 0;
    Print(L"    Signature: %a\n", Signature);
}
//
// Display Processor Brand String
//
VOID
ProcessorBrandString( VOID )
{
    CPUID_BRAND_STRING_DATA Eax, Ebx, Ecx, Edx;
    UINT32                  BrandString[13];
    AsmCpuid(CPUID_BRAND_STRING1, &amp;Eax.Uint32, &amp;Ebx.Uint32, &amp;Ecx.Uint32, &amp;
Edx.Uint32);
#ifdef DEBUG
```

```
    Print(L"  String1:  EAX:%08x  EBX:%08x  ECX:%08x  EDX:%08x\n", Eax.Uint32, Ebx.Uint32,
 Ecx.Uint32, Edx.Uint32);
#endif
    BrandString[0] = Eax.Uint32;
    BrandString[1] = Ebx.Uint32;
    BrandString[2] = Ecx.Uint32;
    BrandString[3] = Edx.Uint32;
    AsmCpuid(CPUID_BRAND_STRING2, &amp;Eax.Uint32, &amp;Ebx.Uint32, &amp;Ecx.Uint32, &amp;
Edx.Uint32);
#ifdef DEBUG
    Print(L"  String2:  EAX:%08x  EBX:%08x  ECX:%08x  EDX:%08x\n", Eax.Uint32, Ebx.Uint32,
 Ecx.Uint32, Edx.Uint32);
#endif
    BrandString[4] = Eax.Uint32;
    BrandString[5] = Ebx.Uint32;
    BrandString[6] = Ecx.Uint32;
    BrandString[7] = Edx.Uint32;
    AsmCpuid(CPUID_BRAND_STRING3, &amp;Eax.Uint32, &amp;Ebx.Uint32, &amp;Ecx.Uint32, &amp;
Edx.Uint32);
#ifdef DEBUG
    Print(L"  String3:  EAX:%08x  EBX:%08x  ECX:%08x  EDX:%08x\n", Eax.Uint32, Ebx.Uint32,
 Ecx.Uint32, Edx.Uint32);
#endif
    BrandString[8]  = Eax.Uint32;
    BrandString[9]  = Ebx.Uint32;
    BrandString[10] = Ecx.Uint32;
    BrandString[11] = Edx.Uint32;
    BrandString[12] = 0;
    Print (L"   CPU String: %a\n", (CHAR8 *)BrandString);
}
//
// Display Processor Version Information
//
VOID
ProcessorVersionInfo( VOID )
{
    CPUID_VERSION_INFO_EAX Eax;
    CPUID_VERSION_INFO_EBX Ebx;
    CPUID_VERSION_INFO_ECX Ecx;
    CPUID_VERSION_INFO_EDX Edx;
    UINT32                 DisplayFamily;
    UINT32                 DisplayModel;
    AsmCpuid(CPUID_VERSION_INFO, &amp;Eax.Uint32, &amp;Ebx.Uint32, &amp;Ecx.Uint32, &amp;E
dx.Uint32);
#ifdef DEBUG
    Print(L"  VersionInfo:  EAX:%08x  EBX:%08x  ECX:%08x  EDX:%08x\n", Eax.Uint32, Ebx.Uin
t32, Ecx.Uint32, Edx.Uint32);
#endif
    DisplayFamily = Eax.Bits.FamilyId;
    if (Eax.Bits.FamilyId == 0x0F) {
      DisplayFamily |= (Eax.Bits.ExtendedFamilyId << 4);
    }
    DisplayModel = Eax.Bits.Model;
    if (Eax.Bits.FamilyId == 0x06 || Eax.Bits.FamilyId == 0x0f) {
        DisplayModel |= (Eax.Bits.ExtendedModelId << 4);
    }
    Print(L"       Family: 0x%x\n", DisplayFamily);
    Print(L"        Model: 0x%x\n", DisplayModel);
    Print(L"     Stepping: 0x%x\n", Eax.Bits.SteppingId);
}
//
// Display Available Processor Features
//
VOID
ProcessorFeatures( VOID )
{
    CPUID_EXTENDED_CPU_SIG_ECX xEcx;
    CPUID_EXTENDED_CPU_SIG_EDX xEdx;
```



```
    CPUID_VERSION_INFO_EAX     Eax;
    CPUID_VERSION_INFO_EBX     Ebx;
    CPUID_VERSION_INFO_ECX     Ecx;
    CPUID_VERSION_INFO_EDX     Edx;
    BOOLEAN                    FirstRow = TRUE;
    UINT32                     xEax;
    UINT32                     Col = 0;
    CHAR16                     Features[1000];
    CHAR16                     Buf[80];
    UINT8                      Width = WIDTH;
    CHAR16                     *f = Features;
    ZeroMem( Features, 1000 );
    AsmCpuid(CPUID_VERSION_INFO, &amp;Eax.Uint32, &amp;Ebx.Uint32, &amp;Ecx.Uint32, &amp;E
dx.Uint32);
#ifdef DEBUG
    Print(L"  Features:  EAX:%08x  EBX:%08x  ECX:%08x  EDX:%08x\n", Eax.Uint32, Ebx.Uint32,
 Ecx.Uint32, Edx.Uint32);
#endif
    AsmCpuid(CPUID_EXTENDED_CPU_SIG, &amp;xEax, NULL, &amp;xEcx.Uint32, &amp;xEdx.Uint32);
#ifdef DEBUG
    Print (L"  Extended Features:  EAX:%08x  EBX:%08x  ECX:%08x  EDX:%08x\n", xEax, 0, xEc
x.Uint32, xEdx.Uint32);
#endif
    // Presorted list. No sorting routine!
    if (Edx.Bits.ACPI) StrCat(Features, L" ACPI");            // ACPI via MSR Support
    if (Ecx.Bits.AESNI) StrCat(Features, L" AESNT");
    if (Edx.Bits.APIC) StrCat(Features, L" APIC");
    if (Ecx.Bits.AVX) StrCat(Features, L" AVX");              // Advanced Vector Extensio
ns
    if (Edx.Bits.CLFSH) StrCat(Features, L" CLFSH");          // CLFLUSH (Cache Line Flus
h) Instruction Support
    if (Edx.Bits.CMOV) StrCat(Features, L" CMOV");            // CMOV Instructions Extens
ion
    if (Ecx.Bits.CMPXCHG16B) StrCat(Features, L" CMPXCHG16B"); // CMPXCHG16B Instruction S
upport
    if (Ecx.Bits.CNXT_ID) StrCat(Features, L" CNXT_ID");      // L1 Cache Adaptive Or Sha
red Mode Support
    if (Edx.Bits.CX8) StrCat(Features, L" CX8");              // CMPXCHG8 Instruction Sup
port
    if (Ecx.Bits.DCA) StrCat(Features, L" DCA");              // Direct Cache Access
    if (Edx.Bits.DE) StrCat(Features, L" DE");                // Debugging Extensions
    if (Edx.Bits.DS) StrCat(Features, L" DS");                // Dubug Store Support
    if (Ecx.Bits.DS_CPL) StrCat(Features, L" DS_CPL");        // CPL Qual. Debug Store Su
pport
    if (Ecx.Bits.DTES64) StrCat(Features, L" DTES64");        // 64-bit Debug Store Suppo
rt
    if (Ecx.Bits.F16C) StrCat(Features, L" F16C");            // 16-bit FP Conversion Ins
tructions Support
    if (Ecx.Bits.FMA) StrCat(Features, L" FMA");              // Fused Multiply-Add
    if (Edx.Bits.FPU) StrCat(Features, L" FPU");              // Floating Point Unit
    if (Edx.Bits.FXSR) StrCat(Features, L" FXSR");            // FXSAVE/FXRSTOR Support
    if (Edx.Bits.HTT) StrCat(Features, L" HTT");              // Max APIC IDs reserved fi
eld is Valid
    if (xEcx.Bits.LAHF_SAHF) StrCat(Features, L" LAHF_SAHF");
    if (xEdx.Bits.LM) StrCat(Features, L" LM");
    if (xEcx.Bits.LZCNT) StrCat(Features, L" LZCNT");
    if (Edx.Bits.MCA) StrCat(Features, L" MCA");              // Machine Check Architectu
re
    if (Edx.Bits.MCE) StrCat(Features, L" MCE");              // Machine Check Exception
    if (Ecx.Bits.MONITOR) StrCat(Features, L" MONITOR");      // Monitor/Mwait Support (S
SE3 supplements)
    if (Edx.Bits.MMX) StrCat(Features, L" MMX");              // Multimedia Extensions
    if (Ecx.Bits.MOVBE) StrCat(Features, L" MOVBE");          // Move Data After Swapping
 Bytes Instruction Support
    if (Edx.Bits.MSR) StrCat(Features, L" MSR");              // Model Specific Registers
    if (Edx.Bits.MTRR) StrCat(Features, L" MTRR");            // Memory Type Range Regist
ers
    if (xEdx.Bits.NX) StrCat(Features, L" NX");
```



```
    if (Ecx.Bits.OSXSAVE) StrCat(Features, L" OSXSAVE");       // OS has set bit to suppor
t CPU extended state management using XSAVE/XRSTOR
    if (Edx.Bits.PAE) StrCat(Features, L" PAE");               // Physical Address Extensi
ons
    if (xEdx.Bits.Page1GB) StrCat(Features, L" PAGE1GB");
    if (Edx.Bits.PAT) StrCat(Features, L" PAT");               // Page Attribute Table
    if (Edx.Bits.PBE) StrCat(Features, L" PBE");               // Pending Break Enable
    if (Ecx.Bits.PCID) StrCat(Features, L" PCID");             // Process Context Identifi
ers
    if (Ecx.Bits.PCLMULQDQ) StrCat(Features, L" PCLMULQDQ");   // Support Carry-Less Multi
plication of Quadword instruction
    if (Ecx.Bits.PDCM) StrCat(Features, L" PDCM");             // Performance Capabilities
    if (Edx.Bits.PGE) StrCat(Features, L" PGE");               // Page Global Enable
    if (Ecx.Bits.POPCNT) StrCat(Features, L" POPCNT");         // Return the Count of Numb
er of Bits Set to 1 instruction
    if (xEcx.Bits.PREFETCHW) StrCat(Features, L" PREFETCHW");
    if (Edx.Bits.PSE) StrCat(Features, L" PSE");               // Page Size Extensions (4M
B memory pages)
    if (Edx.Bits.PSE_36) StrCat(Features, L" PSE_36");         // 36-Bit (> 4MB) Page Size
 Extension
    if (Edx.Bits.PSN) StrCat(Features, L" PSN");               // Processor Serial Number
Support
    if (Ecx.Bits.RDRAND) StrCat(Features, L" RDRAND");         // Read Random Number from
hardware random number generator instruction Support
    if (xEdx.Bits.RDTSCP) StrCat(Features, L" RDTSCP");
    if (Ecx.Bits.SDBG) StrCat(Features, L" SDBG");             // Silicon Debug Support
    if (Edx.Bits.SEP) StrCat(Features, L" SEP");               // SYSENTER/SYSEXIT Support
    if (Ecx.Bits.SMX) StrCat(Features, L" SMX");
    if (Edx.Bits.SS) StrCat(Features, L" SS");
    if (Edx.Bits.SSE) StrCat(Features, L" SSE");
    if (Edx.Bits.SSE2) StrCat(Features, L" SSE2");
    if (Ecx.Bits.SSE3) StrCat(Features, L" SSE3");
    if (Ecx.Bits.SSE4_1) StrCat(Features, L" SSE4_1");
    if (Ecx.Bits.SSE4_2) StrCat(Features, L" SSE4_2");
    if (Ecx.Bits.SSSE3) StrCat(Features, L" SSSE3");                           // Supple
mental SSE-3
    if (xEdx.Bits.SYSCALL_SYSRET) StrCat(Features, L" SYSCALL_SYSRET");
    if (Edx.Bits.TSC) StrCat(Features, L" TSC");                               // Time S
tamp Counter
    if (Ecx.Bits.TSC_Deadline) StrCat(Features, L" TSC_DEADLINE");             // TSC De
adline Timer
    if (Edx.Bits.TM) StrCat(Features, L" TM");                                 // Automa
tic Clock Control (Thermal Monitor)
    if (Ecx.Bits.TM2) StrCat(Features, L" TM2");                               // Therma
l Monitor 2
    if (Edx.Bits.VME) StrCat(Features, L" VME");                               // Virtua
l 8086 Mode Enhancements
    if (Ecx.Bits.VMX) StrCat(Features, L" VMX");                               // Hardwa
re Virtualization Support
    if (Ecx.Bits.x2APIC) StrCat(Features, L" X2APIC");                         // x2APIC
 Support
    if (Ecx.Bits.XSAVE) StrCat(Features, L" XSAVE");                           // Save P
rocessor Extended States
    if (Ecx.Bits.xTPR_Update_Control) StrCat(Features, L" XTPR_UPDATE_CONTROL"); // Change
 IA32_MISC_ENABLE Support
    Print(L"     Features:");
    // Not the most elegant output folding code but it works!
    ZeroMem( Buf, 80 );
    while (*f) {
         Buf[Col++] = *f++;
         if (Col > Width) {
             while (Col >= 0 &amp;&amp; Buf[Col] != L' ') {
                 Buf[Col] = CHAR_NULL;
                 f--; Col--;
             }
             if (FirstRow) {
                 Print(L"%s\n", Buf);
                 FirstRow = FALSE;
```



```
            } else {
                Print(L"             %s\n", Buf);
            }
            ZeroMem(Buf,80);
            Col = 0;
         }
    }
    if (Col) {
        if (FirstRow) {
            Print(L"%s\n", Buf);
        } else {
            Print(L"             %s\n", Buf);
        }
    }
}
VOID
Usage( BOOLEAN ErrorMsg )
{
    if ( ErrorMsg ) {
        Print(L"ERROR: Unknown option(s).\n");
    }
    Print(L"Usage: Cpuid [ -V | --version ]\n");
}
INTN
EFIAPI
ShellAppMain( UINTN Argc,
              CHAR16 **Argv )
{
    EFI_STATUS Status = EFI_SUCCESS;
    if (Argc == 2) {
        if (!StrCmp(Argv[1], L"--version") ||
            !StrCmp(Argv[1], L"-V")) {
            Print(L"Version: %s\n", UTILITY_VERSION);
            return Status;
        } else if (!StrCmp(Argv[1], L"--help") ||
            !StrCmp(Argv[1], L"-h")) {
            Usage(FALSE);
            return Status;
        } else {
            Usage(TRUE);
            return Status;
        }
    }
    if (Argc > 2) {
        Usage(TRUE);
        return Status;
    }
    Print(L"\n");
    ProcessorSignature();
    ProcessorBrandString();
    ProcessorVersionInfo();
    ProcessorFeatures();
    Print(L"\n");
    return Status;
}
```


Note, my utility does not attempt to support non-Intel processor-specific features. This is because I did not have any non-Intel platforms available for testing purposes.

By the way, *CPUID* can be invoked at any privilege level to serialize instruction execution, which guarantees that any modifications to flags, registers, and memory for previous instructions are completed before the next instruction is fetched and executed. See the section entitled *Serializing Instructions* in Chapter 8 (*Multiple-Processor Management*) of the Intel 64 and IA-32 Architectures Software Developer's Manual, Volume 3A.

Enjoy!

