Intel® Platform Innovation Framework for EFI Processor Subclass Specification

Draft for Review

Version 0.9

April 1, 2004

Intel order number xxxxxx-001

Revision History

Revision	Revision History	Date
0.9	First public release.	4/1/04

Contents

1 Introduction

Overview

This specification defines the core code that is required for an implementation of the processor data hub subclass of the Intel® Platform Innovation Framework for EFI (hereafter referred to as the "Framework"). This specification does the following:

- Describes the basic components of the processor data hub subclass and processor subclass data records
- Provides code definitions for type and record definitions for the processor subclass that are architecturally required by the *Intel® Platform Innovation Framework for EFI Architecture Specification*

This specification complies with the System Management BIOS (SMBIOS) Reference Specification, version 2.3.4.

Conventions Used in This Document

This document uses the typographic and illustrative conventions described below.

Data Structure Descriptions

Intel® processors based on 32-bit Intel® architecture (IA-32) are “little endian” machines. This distinction means that the low-order byte of a multibyte data item in memory is at the lowest address, while the high-order byte is at the highest address. Processors of the Intel® Itanium® processor family may be configured for both “little endian” and “big endian” operation. All implementations designed to conform to this specification will use “little endian” operation.

In some memory layout descriptions, certain fields are marked *reserved*. Software must initialize such fields to zero and ignore them when read. On an update operation, software must preserve any reserved field.

The data structures described in this document generally have the following format:

STRUCTURE NAME: The formal name of the data structure.

Summary: A brief description of the data structure.

Prototype: A “C-style” type declaration for the data structure.

Parameters: A brief description of each field in the data structure prototype.

Description: A description of the functionality provided by the data structure, including any limitations and caveats of which the caller should be aware.

Related Definitions: The type declarations and constants that are used only by this data structure.

Pseudo-Code Conventions

Pseudo code is presented to describe algorithms in a more concise form. None of the algorithms in this document are intended to be compiled directly. The code is presented at a level corresponding to the surrounding text.

In describing variables, a *list* is an unordered collection of homogeneous objects. A *queue* is an ordered list of homogeneous objects. Unless otherwise noted, the ordering is assumed to be First In First Out (FIFO).

Pseudo code is presented in a C-like format, using C conventions where appropriate. The coding style, particularly the indentation style, is used for readability and does not necessarily comply with an implementation of the *Extensible Firmware Interface Specification*.

Typographic Conventions

This document uses the typographic and illustrative conventions described below:

Plain text	The normal text typeface is used for the vast majority of the descriptive text in a specification.
Plain text (blue)	In the online help version of this specification, any plain text that is underlined and in blue indicates an active link to the cross-reference. Click on the word to follow the hyperlink. Note that these links are *not* active in the PDF of the specification.
Bold	In text, a **Bold** typeface identifies a processor register name. In other instances, a **Bold** typeface can be used as a running head within a paragraph.
Italic	In text, an *Italic* typeface can be used as emphasis to introduce a new term or to indicate a manual or specification name.
`BOLD Monospace`	Computer code, example code segments, and all prototype code segments use a **`BOLD Monospace`** typeface with a dark red color. These code listings normally appear in one or more separate paragraphs, though words or segments can also be embedded in a normal text paragraph.
`Bold Monospace`	In the online help version of this specification, words in a **`Bold Monospace`** typeface that is underlined and in blue indicate an active hyperlink to the code definition for that function or type definition. Click on the word to follow the hyperlink. Note that these links are *not* active in the PDF of the specification. Also, these inactive links in the PDF may instead have a **`Bold Monospace`** appearance that is underlined but in dark red. Again, these links are not active in the PDF of the specification.
`Italic Monospace`	In code or in text, words in *`Italic Monospace`* indicate placeholder names for variable information that must be supplied (i.e., arguments).
`Plain Monospace`	In code, words in a `Plain Monospace` typeface that is a dark red color but is not bold or italicized indicate pseudo code or example code. These code segments typically occur in one or more separate paragraphs.

text text text — In the PDF of this specification, text that is highlighted in yellow indicates that a change was made to that text since the previous revision of the PDF. The highlighting indicates only that a change was made since the previous version; it does not specify what changed. If text was deleted and thus cannot be highlighted, a note in red and highlighted in yellow (that looks like *(Note: text text text.)*) appears where the deletion occurred.

See the master Framework glossary in the Framework Interoperability and Component Specifications help system for definitions of terms and abbreviations that are used in this document or that might be useful in understanding the descriptions presented in this document.

See the master Framework references in the Interoperability and Component Specifications help system for a complete list of the additional documents and specifications that are required or suggested for interpreting the information presented in this document.

The Framework Interoperability and Component Specifications help system is available at the following URL:

http://www.intel.com/technology/framework/spec.htm

2 Design Discussion

Overview

This specification describes a group of data records that have similar characteristics. The data records are records that will be input to the data hub to be consumed by one or more drivers.

This specification complies with the *Intel® Platform Innovation Framework for EFI Data Hub Specification* and it is assumed that the consumer of this specification is well versed in the concept of the data hub. This specification describes in more detail how to implement a driver that will log all the processor-related data records and how to create drivers that will consume this data.

This specification also specifies the format of each data record. Each data record must be capable of being used by current and potential consumers of the data—in other words, the format should also be consumable by all potential agents. The unit of measurement, if applicable, should be the most common unit of measurement for the specific data record, and the range of values for a data record should be usable for the foreseeable future.

Scope

This specification is the contract processor-related data between the processor data drivers and the processor data consumers. This specification covers all data structures that are processor related. For multiprocessor (MP) systems and systems using Intel® processors with Hyper-Threading Technology, this specification includes all the data related to all processors in the system. Besides the central microprocessor, it also includes data related to math coprocessors, digital signal processors, and video processors.

The processor subclass refers to the processor and other processor-related devices that might be required. Although cache memory is tightly tied to processors, cache memory does not belong to the processor subclass. Instead, cache memory belongs to the cache subclass because it has inherent characteristics distinct to a subclass in itself.

The data driver (processor drivers in this case) does not have to declare all the record types that are listed in this specification but should declare only those data types that are applicable. If the data is not applicable (for example, the socket type for a processor soldered down on the board), the data consumer should make the necessary adjustment and handle such scenarios.

A specific record defines the semantic context of the data. All records related to the processor are documented in this specification. The record type is numbered sequentially and a new record type can be appended to the end of the list. The record type in this specification defines the semantic context of the data. The data records in this specification comply to the **`EFI_SUBCLASS_TYPE1_HEADER`**`.Version` field of 1. Type **`EFI_SUBCLASS_TYPE1_HEADER`** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Data records can be added or deleted as long as backward compatibility is maintained. If a data record needs to be updated, a new entry to this specification can be added. At some point when the objectives of this subclass are not accomplished or become inconsistent, an entirely new subclass with a new Globally Unique Identifier (GUID) will be introduced.

NOTE

The data records and structures sometimes are System Management BIOS (SMBIOS)–centric as the initial document is based on the SMBIOS requirements.

Compliancy Requirements

None of the record numbers described in this document are required by the *Extensible Firmware Interface Specification.* Some record number entries may be required by other industry-wide specifications such as the *System Management BIOS (SMBIOS) Reference Specification.*

Some **ENUM** definitions are controlled by the Distributed Management Task Force, Inc. (DMTF) organization. Refer to their Web site at www.dmtf.org/standards/dmi* and select the link to *Master MIF*.

Header Information

Processor Subclass Definition

The processor subclass belongs to the data class and is identified as the processor subclass by the GUID. See Processor Subclass Definition in Code Definitions for the definition.

Data Record Header

Each data record that is logged or read starts with a standard header of type **EFI_DATA_RECORD_HEADER**. The format of the header is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Specification*.

Subclass Header

Each data record that is a member of the data class starts with a standard header of type **EFI_SUBCLASS_TYPE1_HEADER**. The format of the header is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*. This header follows the data record header **EFI_DATA_RECORD_HEADER**, which is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Specification*.

The *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide* provides generic descriptions of the fields in **EFI_SUBCLASS_TYPE1_HEADER**. The following explanation further clarifies the descriptions for the processor subclass:

- The *Instance* is the instance number of the logical processor produced by the same *ProducerName* in the system, as multiple logical processors can exist in the system.
- The *SubInstance* is the instance number of the *RecordType* of the processor *Instance*. In some *RecordType*s, *SubInstance* may mean a different cache level that is associated with a processor *Instance* (see Cache Association, for example).

Raw Data

The raw data follows the **EFI_SUBCLASS_TYPE1_HEADER** header and its definition is specific to the *RecordNumber*. The syntax of the raw data is defined in Data Record Number in Code Definitions. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Data Record Number

The **EFI_SUBCLASS_TYPE1_HEADER** is followed by a data record. The data record format is specific to a *RecordNumber*. The sections in Data Record Number in Code Definitions define the data *RecordNumber*s for the processor subclass.

All generic macros are defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*. **STRING_REF** is defined in the *Intel® Platform Innovation Framework for EFI Human Interface Infrastructure Specification*.

3 Code Definitions

Introduction

This section contains the basic definitions of the data record header fields that are specific to the processor subclass, as well as definitions of processor data records. The following data types and data records are defined in this section:

- EFI_PROCESSOR_SUBCLASS
- EFI_PROCESSOR_SUBCLASS_VERSION
- EFI_PROCESSOR_CORE_FREQUENCY_DATA
- EFI_PROCESSOR_FSB_FREQUENCY_DATA
- EFI_PROCESSOR_VERSION_DATA
- EFI_PROCESSOR_MANUFACTURER_DATA
- EFI_PROCESSOR_SERIAL_NUMBER_DATA
- EFI_PROCESSOR_ID_DATA
- EFI_PROCESSOR_TYPE_DATA
- EFI_PROCESSOR_FAMILY_DATA
- EFI_PROCESSOR_VOLTAGE_DATA
- EFI_PROCESSOR_APIC_BASE_ADDRESS_DATA
- EFI_PROCESSOR_APIC_ID_DATA
- EFI_PROCESSOR_APIC_VERSION_NUMBER_DATA
- EFI_PROCESSOR_MICROCODE_REVISION_DATA
- EFI_PROCESSOR_STATUS_DATA
- EFI_PROCESSOR_SOCKET_TYPE_DATA
- EFI_PROCESSOR_SOCKET_NAME_DATA
- EFI_PROCESSOR_CACHE_ASSOCIATION_DATA
- EFI_PROCESSOR_MAX_CORE_FREQUENCY_DATA
- EFI_PROCESSOR_ASSET_TAG_DATA
- EFI_PROCESSOR_MAX_FSB_FREQUENCY_DATA
- EFI_PROCESSOR_PACKAGE_NUMBER_DATA
- EFI_PROCESSOR_CORE_FREQUENCY_LIST_DATA
- EFI_PROCESSOR_FSB_FREQUENCY_LIST_DATA
- EFI_PROCESSOR_HEALTH_STATUS_DATA

Header Information

Processor Subclass Definition

EFI_PROCESSOR_SUBCLASS

Summary

The processor subclass belongs to the data class and is identified as the processor subclass by the GUID.

GUID

```
#define EFI_PROCESSOR_SUBCLASS_GUID \
{ 0x26fdeb7e, 0xb8af, 0x4ccf, 0xaa, 0x97, 0x2, 0x63, 0x3c, 0xe4,
0x8c, 0xa7 }
```

Class

```
#define        EFI_PROCESSOR_SUBCLASS        EFI_DATA_CLASS_DATA
```

Description

Summary

The processor subclass belongs to the data class and is identified as the processor subclass by the GUID.

For this subclass, the values defined above are used as follows:

- **EFI_DATA_RECORD_HEADER**.*DataRecordGuid* = **EFI_PROCESSOR_SUBCLASS_GUID**, which is the GUID that is specific to the processor subclass.
- **EFI_DATA_RECORD_HEADER**.*DataRecordClass* = **EFI_DATA_CLASS_DATA**. The "class" may be equal to the GUID "class" or a superset of the GUID "class."

Type **EFI_DATA_CLASS_DATA** is defined in **EFI_DATA_RECORD_HEADER**, which is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Specification*.

Subclass Version

EFI_PROCESSOR_SUBCLASS_VERSION

Summary

Indicates the version of the processor subclass.

Prototype

```
#define EFI_PROCESSOR_SUBCLASS_VERSION   0x0100
```

Description

This value indicates the version of the processor subclass. It is used in **EFI_SUBCLASS_TYPE1_HEADER**.*Version*. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Data Record Number

Core Frequency

EFI_PROCESSOR_CORE_FREQUENCY_DATA

Summary

This data record refers to the internal frequency of the processor.

Prototype

```
typedef EFI_EXP_BASE10_DATA     EFI_PROCESSOR_CORE_FREQUENCY_DATA;
```

Description

This data record refers to the internal frequency of the processor. The unit of measurement of this data record is in Hertz. For asynchronous processors, the content of this data record is zero.

Type **EFI_EXP_BASE10_DATA** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_FREQUENCY_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide.*

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_FREQUENCY_RECORD_NUMBER      0x00000001
```

Front Side Bus Frequency

EFI_PROCESSOR_FSB_FREQUENCY_DATA

Summary

This data record refers to the external bus frequency of the processor.

Prototype

```
typedef EFI_EXP_BASE10_DATA     EFI_PROCESSOR_FSB_FREQUENCY_DATA;
```

Description

This data record refers to the external bus frequency of the processor. The unit of measurement of this data record is in Hertz. For asynchronous processors, the content of this data record is zero.

Type **EFI_EXP_BASE10_DATA** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_BUS_FREQUENCY_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide.*

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_BUS_FREQUENCY_RECORD_NUMBER     0x00000002
```

Version

EFI_PROCESSOR_VERSION_DATA

Summary

This data record refers to the external name/brand of the processor.

Prototype

```
typedef STRING_REF          EFI_PROCESSOR_VERSION_DATA;
```

Summary

This data record refers to the external name/brand of the processor. This data record is a string token. The string is retrievable from a string database through the Human Interface Infrastructure (HII) Protocol's **GetString()** function.

Type **STRING_REF** and **EFI_HII_PROTOCOL.GetString()** are defined in the *Intel® Platform Innovation Framework for EFI Human Interface Infrastructure Specification*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_VERSION_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_VERSION_RECORD_NUMBER  0x00000003
```

Manufacturer

EFI_PROCESSOR_MANUFACTURER_DATA

Summary

This data record refers to the entity that is manufacturing the processor.

Prototype

```
typedef STRING_REF          EFI_PROCESSOR_MANUFACTURER_DATA;
```

Description

This data record refers to the entity that is manufacturing the processor. This data record is a string token. The string is retrievable from a string database through the HII Protocol's **GetString()** function.

Type **STRING_REF** and **EFI_HII_PROTOCOL.GetString()** are defined in the *Intel® Platform Innovation Framework for EFI Human Interface Infrastructure Specification*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_MANUFACTURER_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_MANUFACTURER_RECORD_NUMBER     0x00000004
```

Serial Number

EFI_PROCESSOR_SERIAL_NUMBER_DATA

Summary

This data record refers to the unique serial number of the processor that is identifiable through software or the serial number marked on the processor.

Prototype

```
typedef STRING_REF          EFI_PROCESSOR_SERIAL_NUMBER_DATA;
```

Description

This data record refers to the unique serial number of the processor that is identifiable through software or the serial number marked on the processor. This data record is a string token. The string is retrievable from a string database through the HII Protocol's **GetString()** function.

Type **STRING_REF** and **EFI_HII_PROTOCOL.GetString()** are defined in the *Intel® Platform Innovation Framework for EFI Human Interface Infrastructure Specification*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_SERIAL_NUMBER_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_SERIAL_NUMBER_RECORD_NUMBER      0x00000005
```

ID

EFI_PROCESSOR_ID_DATA

Summary

This data record refers to the unique ID that identifies a set of processors.

Prototype

```
typedef struct {
  EFI_PROCESSOR_SIGNATURE      Signature;
  EFI_PROCESSOR_MISC_INFO      MiscInfo;
  UINT32                       Reserved;
  EFI_PROCESSOR_FEATURE_FLAGS  FeatureFlags;
} EFI_PROCESSOR_ID_DATA;
```

Parameters

Signature

Identifies the processor. Type **EFI_PROCESSOR_SIGNATURE** is defined in "Related Definitions" below.

MiscInfo

Provides additional processor information. Type **EFI_PROCESSOR_MISC_INFO** is defined in "Related Definitions" below.

Reserved

Reserved for future use.

FeatureFlags

Provides additional processor information. Type **EFI_PROCESSOR_FEATURE_FLAGS** is defined in "Related Definitions" below.

Summary

This data record refers to the unique ID that identifies a set of processors. This data record is 16 bytes in length. The data in this structure is processor specific and reserved values can be defined for future use. The consumer of this data should not make any assumption and should use this data with respect to the processor family defined in the Family record number.

See the following for more information on processor IDs:

- SMBIOS 2.3.4, Type 4, section 3.3.5.3
- Manuals and other technical documentation on Intel® processors:
  - — IA-32 processors: http://developer.intel.com/design/pentium4/index.htm
  - — Itanium® processor family: http://developer.intel.com/design/itanium/family/index.htm

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_ID_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_ID_RECORD_NUMBER      0x00000006
```

NOTE

See the IA-32 Intel® Architecture Software Developer's Manual for descriptions of the fields in the definitions below. Also note that these fields can be processor specific.

```
//*************************************************************
// EFI_PROCESSOR_SIGNATURE
//*************************************************************
typedef struct {
  UINT32  ProcessorSteppingId:4;
  UINT32  ProcessorModel:     4;
  UINT32  ProcessorFamily:    4;
  UINT32  ProcessorType:      2;
  UINT32  ProcessorReserved1: 2;
  UINT32  ProcessorXModel:    4;
  UINT32  ProcessorXFamily:   8;
  UINT32  ProcessorReserved2: 4;
} EFI_PROCESSOR_SIGNATURE;


//*************************************************************
// EFI_PROCESSOR_MISC_INFO
//*************************************************************
typedef struct {
  UINT32  ProcessorBrandIndex   :8;
  UINT32  ProcessorClflush      :8;
  UINT32  LogicalProcessorCount :8;
  UINT32  ProcessorDfltApicId   :8;
} EFI_PROCESSOR_MISC_INFO;
```

```
//*******************************************************
// EFI_PROCESSOR_FEATURE_FLAGS
//*******************************************************
typedef struct {
  UINT32  ProcessorFpu:       1;
  UINT32  ProcessorVme:       1;
  UINT32  ProcessorDe:        1;
  UINT32  ProcessorPse:       1;
  UINT32  ProcessorTsc:       1;
  UINT32  ProcessorMsr:       1;
  UINT32  ProcessorPae:       1;
  UINT32  ProcessorMce:       1;
  UINT32  ProcessorCx8:       1;
  UINT32  ProcessorApic:      1;
  UINT32  ProcessorReserved1: 1;
  UINT32  ProcessorSep:       1;
  UINT32  ProcessorMtrr:      1;
  UINT32  ProcessorPge:       1;
  UINT32  ProcessorMca:       1;
  UINT32  ProcessorCmov:      1;
  UINT32  ProcessorPat:       1;
  UINT32  ProcessorPse36:     1;
  UINT32  ProcessorPsn:       1;
  UINT32  ProcessorClfsh:     1;
  UINT32  ProcessorReserved2: 1;
  UINT32  ProcessorDs:        1;
  UINT32  ProcessorAcpi:      1;
  UINT32  ProcessorMmx:       1;
  UINT32  ProcessorFxsr:      1;
  UINT32  ProcessorSse:       1;
  UINT32  ProcessorSse2:      1;
  UINT32  ProcessorSs:        1;
  UINT32  ProcessorReserved3: 1;
  UINT32  ProcessorTm:        1;
  UINT32  ProcessorReserved4: 2;
} EFI_PROCESSOR_FEATURE_FLAGS;
```

Type

EFI_PROCESSOR_TYPE_DATA

Summary

This data record refers to the general classification of the processor.

Prototype

```
typedef enum {
  EfiProcessorOther = 1,
  EfiProcessorUnknown = 2,
  EfiCentralProcessor = 3,
  EfiMathProcessor = 4,
  EfiDspProcessor = 5,
  EfiVideoProcessor = 6
} EFI_PROCESSOR_TYPE_DATA;
```

Description

This data record refers to the general classification of the processor. This data record is 4 bytes in length.

The type definition enumeration for **EFI_PROCESSOR_TYPE_DATA** is in SMBIOS 2.3.4, Table 3.3.5.1, Type 4, Offset 05h.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_TYPE_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_TYPE_RECORD_NUMBER     0x00000007
```

Family

EFI_PROCESSOR_FAMILY_DATA

IMPORTANT NOTE

Enumerated values are controlled by the DMTF, not this specification.

Summary

This data record refers to the family of the processor as defined by the DMTF.

Prototype

```
typedef enum {
  EfiProcessorFamilyOther        = 1,
  EfiProcessorFamilyUnknown      = 2,
  EfiProcessorFamily8086         = 3,
  EfiProcessorFamily80286        = 4,
  EfiProcessorFamilyIntel386     = 5,
  EfiProcessorFamilyIntel486     = 6,
  EfiProcessorFamily8087         = 7,
  EfiProcessorFamily80287        = 8,
  EfiProcessorFamily80387        = 9,
  EfiProcessorFamily80487        = 0x0A,
  EfiProcessorFamilyPentium      = 0x0B,
  EfiProcessorFamilyPentiumPro   = 0x0C,
  EfiProcessorFamilyPentiumII    = 0x0D,
  EfiProcessorFamilyPentiumMMX   = 0x0E,
  EfiProcessorFamilyCeleron      = 0x0F,
  EfiProcessorFamilyPentiumIIXeon = 0x10,
  EfiProcessorFamilyPentiumIII   = 0x11,
  EfiProcessorFamilyM1           = 0x12,
  EfiProcessorFamilyM2           = 0x13
  EfiProcessorFamilyM1Reserved2 = 0x14,
  EfiProcessorFamilyM1Reserved3  = 0x15,
  EfiProcessorFamilyM1Reserved4  = 0x16,
  EfiProcessorFamilyM1Reserved5  = 0x17,
  EfiProcessorFamilyAmdDuron     = 0x18,
  EfiProcessorFamilyK5           = 0x19,
  EfiProcessorFamilyK6           = 0x1A,
  EfiProcessorFamilyK6-2         = 0x1B,
  EfiProcessorFamilyK6-3         = 0x1C,
  EfiProcessorFamilyAmdAthlon    = 0x1D,
  EfiProcessorFamilyK6-2Plus     = 0x1E,
  EfiProcessorFamilyK5Reserved6  = 0x1F,
  EfiProcessorFamilyPowerPC      = 0x20,
  EfiProcessorFamilyPowerPC601   = 0x21,
  EfiProcessorFamilyPowerPC603   = 0x22,
  EfiProcessorFamilyPowerPC603Plus = 0x23,
```

```
EfiProcessorFamilyPowerPC604   = 0x24,
EfiProcessorFamilyPowerPC620   = 0x25,
EfiProcessorFamilyPowerPC704   = 0x26,
EfiProcessorFamilyPowerPC750   = 0x27,
EfiProcessorFamilyAlpha2       = 0x30,
EfiProcessorFamilyAlpha21064   = 0x31,
EfiProcessorFamilyAlpha21066   = 0x32,
EfiProcessorFamilyAlpha21164   = 0x33,
EfiProcessorFamilyAlpha21164PC = 0x34,
EfiProcessorFamilyAlpha21164a  = 0x35,
EfiProcessorFamilyAlpha21264   = 0x36,
EfiProcessorFamilyAlpha21364   = 0x37,
EfiProcessorFamilyMips         = 0x40,
EfiProcessorFamilyMIPSR4000    = 0x41,
EfiProcessorFamilyMIPSR4200    = 0x42,
EfiProcessorFamilyMIPSR4400    = 0x43,
EfiProcessorFamilyMIPSR4600    = 0x44,
EfiProcessorFamilyMIPSR10000   = 0x45,
EfiProcessorFamilySparc        = 0x50,
EfiProcessorFamilySuperSparc   = 0x51,
EfiProcessorFamilymicroSparcII = 0x52,
EfiProcessorFamilymicroSparcIIep = 0x53,
EfiProcessorFamilyUltraSparc   = 0x54,
EfiProcessorFamilyUltraSparcII = 0x55,
EfiProcessorFamilyUltraSparcIIi = 0x56,
EfiProcessorFamilyUltraSparcIII = 0x57,
EfiProcessorFamilyUltraSparcIIi = 0x58,
EfiProcessorFamily68040        = 0x60,
EfiProcessorFamily68xxx        = 0x61,
EfiProcessorFamily68000        = 0x62,
EfiProcessorFamily68010        = 0x63,
EfiProcessorFamily68020        = 0x64,
EfiProcessorFamily68030        = 0x65,
EfiProcessorFamilyHobbit       = 0x70,
EfiProcessorFamilyCrusoeTM5000 = 0x78,
EfiProcessorFamilyCrusoeTM3000 = 0x79,
EfiProcessorFamilyWeitek       = 0x80,
EfiProcessorFamilyItanium      = 0x82,
EfiProcessorFamilyAmdAthlon64  = 0x83,
EfiProcessorFamilyAmdOpteron   = 0x84,
EfiProcessorFamilyPARISC       = 0x90,
EfiProcessorFamilyPaRisc8500   = 0x91,
EfiProcessorFamilyPaRisc8000   = 0x92,
EfiProcessorFamilyPaRisc7300LC = 0x93,
EfiProcessorFamilyPaRisc7200   = 0x94,
EfiProcessorFamilyPaRisc7100LC = 0x95,
EfiProcessorFamilyPaRisc7100   = 0x96,
EfiProcessorFamilyV30          = 0xA0,
EfiProcessorFamilyPentiumIIIXeon = 0xB0,
```

```
  EfiProcessorFamilyPentiumIIISpeedStep = 0xB1,
  EfiProcessorFamilyPentium4     = 0xB2,
  EfiProcessorFamilyIntelXeon    = 0xB3,
  EfiProcessorFamilyAS400        = 0xB4,
  EfiProcessorFamilyIntelXeonMP  = 0xB5,
  EfiProcessorFamilyAMDAthlonXP = 0xB6,
  EfiProcessorFamilyAMDAthlonMP = 0xB7,
  EfiProcessorFamilyIntelItanium2 = 0xB8,
  EfiProcessorFamilyIBM390       = 0xC8,
  EfiProcessorFamilyG4           = 0xC9,
  EfiProcessorFamilyG5           = 0xCA,
  EfiProcessorFamilyi860         = 0xFA,
  EfiProcessorFamilyi960         = 0xFB
} EFI_PROCESSOR_FAMILY_DATA;
```

Summary

This data record refers to the family of the processor as defined by the DMTF. This data record is 4 bytes in length.

The type definition enumeration for **EFI_PROCESSOR_FAMILY_DATA** is in SMBIOS 2.3.4, Table 3.3.5.2, Type 4, Offset 06h.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_FAMILY_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_FAMILY_RECORD_NUMBER        0x00000008
```

Voltage

EFI_PROCESSOR_VOLTAGE_DATA

Summary

This data record refers to the core voltage of the processor being defined.

Prototype

```
typedef EFI_EXP_BASE10_DATA          EFI_PROCESSOR_VOLTAGE_DATA;
```

Description

This data record refers to the core voltage of the processor being defined. The unit of measurement of this data record is in volts.

Type **EFI_EXP_BASE10_DATA** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_VOLTAGE_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_VOLTAGE_RECORD_NUMBER       0x00000009
```

APIC Base Address

EFI_PROCESSOR_APIC_BASE_ADDRESS_DATA

Summary

This data record refers to the base address of the Advanced Programmable Interrupt Controller (APIC) of the processor being defined.

Prototype

```
typedef EFI_PHYSICAL_ADDRESS   EFI_PROCESSOR_APIC_BASE_ADDRESS_DATA;
```

Description

This data record refers to the base address of the APIC of the processor being defined. This data record is a physical address location.

Type **EFI_PHYSICAL_ADDRESS** is defined in **AllocatePages()** in the *EFI 1.10 Specification.*

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_APIC_BASE_ADDRESS_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_APIC_BASE_ADDRESS_RECORD_NUMBER  0x0000000A
```

Processor APIC ID

EFI_PROCESSOR_APIC_ID_DATA

Summary

This data record refers to the ID of the APIC of the processor being defined.

Prototype

```
typedef UINT32              EFI_PROCESSOR_APIC_ID_DATA;
```

Description

This data record refers to the ID of the APIC of the processor being defined. This data record is a 4-byte entry.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_APIC_ID_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_APIC_ID_RECORD_NUMBER 0x0000000B
```

APIC Version Number

EFI_PROCESSOR_APIC_VERSION_NUMBER_DATA

Summary

This data record refers to the version number of the APIC of the processor being defined.

Prototype

```
typedef UINT32              EFI_PROCESSOR_APIC_VERSION_NUMBER_DATA;
```

Description

This data record refers to the version number of the APIC of the processor being defined. This data record is a 4-byte entry.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_APIC_VER_NUMBER_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_APIC_VER_NUMBER_RECORD_NUMBER    0x0000000C
```

Microcode Update Revision

EFI_PROCESSOR_MICROCODE_REVISION_DATA

Summary

This data record refers to the revision of the processor microcode that is loaded in the processor.

Prototype

```
typedef struct {
  EFI_PROCESSOR_MICROCODE_TYPE ProcessorMicrocodeType;
  UINT32                       ProcessorMicrocodeRevisionNumber;
} EFI_PROCESSOR_MICROCODE_REVISION_DATA;
```

Parameters

ProcessorMicrocodeType

Identifies what type of microcode the data is. Type **EFI_PROCESSOR_MICROCODE_TYPE** is defined in "Related Definitions" below.

ProcessorMicrocodeRevisionNumber

Indicates the revision number of this microcode.

Description

This data record refers to the revision of the processor microcode that is loaded in the processor. This data record is a 4-byte entry.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_MICROCODE_REVISION_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_MICROCODE_REVISION_RECORD_NUMBER 0x0000000D;


//*******************************************************
// EFI_PROCESSOR_MICROCODE_TYPE
//*******************************************************
typedef enum {
  EfiProcessorIa32Microcode = 1,
  EfiProcessorIpfPalAMicrocode = 2,
  EfiProcessorIpfPalBMicrocode = 3,
} EFI_PROCESSOR_MICROCODE_TYPE;
```

Following is a description of the fields in the above definition.

EfiProcessorIa32Microcode	Refers to the system configuration data as described by the *IA-32 BIOS Writer's Guide.*
EfiProcessorIpfPalAMicrocode	Refers to the Processor Abstraction Layer A (PAL-A) code for the Itanium® processor family.
EfiProcessorIpfPalBMicrocode	Refers to the Processor Abstraction Layer B (PAL-B) code for the Itanium processor family.

Processor Status

EFI_PROCESSOR_STATUS_DATA

Description

This data record refers to the status of the processor.

Prototype

```
typedef struct {
  UINT32            CpuStatus                 :3;
  UINT32            Reserved1                 :3;
  UINT32            SocketPopulated           :1;
  UINT32            Reserved2                 :1;
  UINT32            ApicEnable                :1;
  UINT32            BootApplicationProcessor  :1;
  UINT32            Reserved3                 :22;
} EFI_PROCESSOR_STATUS_DATA;
```

Parameters

CpuStatus

Indicates the status of the processor. Type **EFI_CPU_STATUS** is defined in "Related Definitions" below.

Reserved1

Reserved for future use. Should be set to zero.

SocketPopulated

Indicates if the processor is socketed or not.

Reserved2

Reserved for future use. Should be set to zero.

ApicEnable

Indicates if the APIC is enabled or not.

BootApplicationProcessor

Indicates if this processor is the boot processor.

Reserved3

Reserved for future use. Should be set to zero.

Description

This data record refers to the status of the processor.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_STATUS_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_STATUS_RECORD_NUMBER         0x0000000E


//*************************************************************
// EFI_CPU_STATUS
//*************************************************************
typedef enum {
  EfiCpuStatusUnknown = 0,
  EfiCpuStatusEnabled = 1,
  EfiCpuStatusDisabledByUser = 2,
  EfiCpuStatusDisabledbyBios = 3,
  EfiCpuStatusIdle = 4,
  EfiCpuStatusOther = 7
} EFI_CPU_STATUS;
```

Socket Type

EFI_PROCESSOR_SOCKET_TYPE_DATA

Summary

This data record refers to the socket type of the processor.

Prototype

```
typedef enum {
  EfiProcessorSocketOther   = 1,
  EfiProcessorSocketUnknown = 2,
  EfiProcessorSocketDaughterBoard = 3,
  EfiProcessorSocketZIF     = 4,
  EfiProcessorSocketReplacePiggyBack = 5,
  EfiProcessorSocketNone    = 6,
  EfiProcessorSocketLIF     = 7,
  EfiProcessorSocketSlot1   = 8,
  EfiProcessorSocketSlot2   = 9,
  EfiProcessorSocket370Pin  = 0xA,
  EfiProcessorSocketSlotA   = 0xB,
  EfiProcessorSocketSlotM   = 0xC,
  EfiProcessorSocket423     = 0xD,
  EfiProcessorSocketA462    = 0xE,
  EfiProcessorSocket478     = 0xF,
  EfiProcessorSocket754     = 0x10,
  EfiProcessorSocket940     = 0x11
} EFI_PROCESSOR_SOCKET_TYPE_DATA;
```

Description

This data record refers to the socket type of the processor.

The type definition enumeration for **EFI_PROCESSOR_SOCKET_TYPE_DATA** is in SMBIOS 2.3.4, Table 3.3.5.5, Type 4, Offset 0x19.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_SOCKET_TYPE_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_SOCKET_TYPE_RECORD_NUMBER    0x0000000F
```

Socket Name

EFI_PROCESSOR_SOCKET_NAME_DATA

Summary

This data record refers to the socket identifier of the processor.

Prototype

```
typedef STRING_REF          EFI_PROCESSOR_SOCKET_NAME_DATA;
```

Description

This data record refers to the socket identifier of the processor. This data record is a string token. The string is retrievable from a string database through the HII Protocol's **GetString()** function.

Type **STRING_REF** and **EFI_HII_PROTOCOL.GetString()** are defined in the *Intel® Platform Innovation Framework for EFI Human Interface Infrastructure Specification*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_SOCKET_NAME_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_SOCKET_NAME_RECORD_NUMBER    0x00000010
```

Cache Association

EFI_PROCESSOR_CACHE_ASSOCIATION_DATA

Summary

This data record refers to the cache *Instance* and *SubInstance* to which a processor is associated.

Prototype

```
typedef EFI_INTER_LINK_DATA    EFI_PROCESSOR_CACHE_ASSOCIATION_DATA;
```

Description

This data record refers to the cache *Instance* and *SubInstance* to which a processor is associated.

Type **EFI_INTER_LINK_DATA** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_CACHE_ASSOCIATION_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide.*

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_CACHE_ASSOCIATION_RECORD_NUMBER   0x00000011
```

Processor Maximum Frequency

EFI_PROCESSOR_MAX_CORE_FREQUENCY_DATA

Summary

This data record refers to the maximum processor frequency supported by this system.

Prototype

```
typedef EFI_EXP_BASE10_DATA   EFI_PROCESSOR_MAX_CORE_FREQUENCY_DATA;
```

Description

This data record refers to the maximum processor frequency supported by this system. The unit of measurement of this data record is in Hertz. For asynchronous processors, the content of this data record is zero.

Type **EFI_EXP_BASE10_DATA** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_MAX_FREQUENCY_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide.*

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_MAX_FREQUENCY_RECORD_NUMBER     0x00000012
```

Processor Asset Tag

EFI_PROCESSOR_ASSET_TAG_DATA

Summary

This data record refers to the asset tag of the processor. This data record is a string token.

Prototype

```
typedef STRING_REF          EFI_PROCESSOR_ASSET_TAG_DATA;
```

Description

This data record refers to the asset tag of the processor. This data record is a string token. The string is retrievable from a string database through the HII Protocol's **GetString()** function.

Type **STRING_REF** and **EFI_HII_PROTOCOL.GetString()** are defined in the *Intel® Platform Innovation Framework for EFI Human Interface Infrastructure Specification*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_ASSET_TAG_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*************************************************************
// Record number
//*************************************************************
#define EFI_PROCESSOR_ASSET_TAG_RECORD_NUMBER      0x000000013
```

Processor Maximum Bus Frequency

EFI_PROCESSOR_MAX_FSB_FREQUENCY_DATA

Summary

This data record refers to the maximum processor front side bus frequency that is supported by this system.

Prototype

```
typedef EFI_EXP_BASE10_DATA    EFI_PROCESSOR_MAX_FSB_FREQUENCY_DATA;
```

Description

This data record refers to the maximum processor front side bus frequency that is supported by this system. The unit of measurement of this data record is in Hertz. For asynchronous processors, the content of this data record is zero.

Type **EFI_EXP_BASE10_DATA** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_MAX_FSB_FREQUENCY_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide.*

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_MAX_FSB_FREQUENCY_RECORD_NUMBER  0x00000014
```

Processor Package Number

EFI_PROCESSOR_PACKAGE_NUMBER_DATA

Summary

This data record refers to the package number of this processor.

Prototype

```
typedef UINTN  EFI_PROCESSOR_PACKAGE_NUMBER_DATA;
```

Description

This data record refers to the package number of this processor. Multiple logical processors can exist in a system and each logical processor can be correlated to the physical processor using this record type.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_PACKAGE_NUMBER_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_PACKAGE_NUMBER_RECORD_NUMBER     0x00000015
```

Core Frequency List

EFI_PROCESSOR_CORE_FREQUENCY_LIST_DATA

Summary

This data record refers to the list of frequencies that the processor core supports.

Prototype

```
typedef EFI_EXP_BASE10_DATA  EFI_PROCESSOR_CORE_FREQUENCY_LIST_DATA;
```

Description

This data record refers to the list of frequencies that the processor core supports. The list of supported frequencies is determined by the firmware based on hardware capabilities—for example, it could be a common subset of all processors and the chipset. The unit of measurement of this data record is in Hertz. For asynchronous processors, the content of this data record is zero.

The list is terminated by -1 in the *Value* field of the last element. A *Value* field of zero means that the processor/driver supports automatic frequency selection.

Type **EFI_EXP_BASE10_DATA** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_FREQUENCY_LIST_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide.*

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_FREQUENCY_LIST_RECORD_NUMBER     0x00000016
```

Front Side Bus Frequency List

EFI_PROCESSOR_FSB_FREQUENCY_LIST_DATA

Summary

This data record refers to the list of supported frequencies of the processor external bus.

Prototype

```
typedef EFI_EXP_BASE10_DATA  *EFI_PROCESSOR_FSB_FREQUENCY_LIST_DATA;
```

Description

This data record refers to the list of supported frequencies of the processor external bus. The list of supported frequencies is determined by the firmware based on hardware capabilities—for example, it could be a common subset of all processors and the chipset. The unit of measurement of this data record is in Hertz. For asynchronous processors, the content of this data record is **NULL**.

The list is terminated by -1 in the *Value* field of the last element. A *Value* field of zero means that the processor/driver supports automatic frequency selection.

Type **EFI_EXP_BASE10_DATA** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_FSB_FREQUENCY_LIST_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide.*

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_FSB_FREQUENCY_LIST_RECORD_NUMBER 0x00000017
```

Processor Health Status

EFI_PROCESSOR_HEALTH_STATUS_DATA

Summary

This data record refers to the health status of the processor.

Prototype

```
typedef enum {
  EfiProcessorHealthy = 1,
  EfiProcessorPerfRestricted = 2,
  EfiProcessorFuncRestricted = 3
} EFI_PROCESSOR_HEALTH_STATUS_DATA;
```

Parameters

EfiProcessorHealthy

No hardware failures have occurred in testing that would affect either the performance or functionality of the processor.

EfiProcessorPerfRestricted

A hardware failure has occurred in testing that does not affect the functionality of the processor, but performance may be degraded.

EfiProcessorFuncRestricted

A hardware failure has occurred in testing that affects the functionality of the processor, but firmware code can still run. The performance of the processor may also be restricted.

Description

This data record refers to the health status of the processor as defined in **EFI_PROCESSOR_HEALTH_STATUS_DATA** above.

For this data record, **EFI_SUBCLASS_TYPE1_HEADER**.*RecordType* = **EFI_PROCESSOR_HEALTH_STATUS_RECORD_NUMBER**. Type **EFI_SUBCLASS_TYPE1_HEADER** is defined in the *Intel® Platform Innovation Framework for EFI Data Hub Subclass Design Guide*.

Related Definitions

```
//*******************************************************
// Record number
//*******************************************************
#define EFI_PROCESSOR_HEALTH_STATUS_RECORD_NUMBER  0x00000018
```