

Intel® Converged Security Engine Firmware Integrated Clock Controller (ICC) Tool

Tools User Guide

February 2021

Revision 1.21

Intel Confidential



You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at [Intel.com](https://www.intel.com), or from the OEM or retailer.

No computer system can be absolutely secure. Intel does not assume any liability for lost or stolen data or systems or any damages resulting from such losses.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at [intel.com](https://www.intel.com), or from the OEM or retailer.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or visit www.intel.com/design/literature.htm.

By using this document, in addition to any agreements you have with Intel, you accept the terms set forth below.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

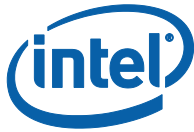
*Other names and brands may be claimed as the property of others.

Copyright © 2021, Intel Corporation. All rights reserved.



Contents

1	Introduction	5
1.1	Terminology	5
1.2	Reference Documents.....	6
2	Intel® ICCS SDK	7
2.1	Intel® ICCS SDK	7
2.2	Intel® ICCS Control Library SDK Features	7
2.3	Intel® Integrated Clock Controller Service Data Structures.....	7
2.3.1	ICC_HECI_CLOCK_ID Type	8
2.3.2	ErrorCodes Type.....	8
2.3.3	ICC_GET_CLOCK_SETTINGSEx Type	8
2.3.4	ICC_SET_CLOCK_SETTINGSEx Type	9
2.3.5	GET MPHY Version	9
2.3.6	IccLibGetMphyVersion	10
2.3.7	IccLibGetMphySettingsWrapper	10
2.3.8	IccLibSetMphySettingsWrapper	10
2.4	Intel® Integrated Clock Controller Service API – Clock Manipulation Interface	11
2.4.1	Get Current Clock Settings Wrapper.....	12
2.4.2	Set Current Clock Settings Wrapper	12
2.4.3	Set Current Clock Settings WrapperEX	13
3	Building an Executable	14



Revision History

Revision Number	Description	Revision Date
0.5	<ul style="list-style-type: none">Initial release.	January 2018
0.6	<ul style="list-style-type: none">Added Set/Get Register commandsAdded Set/Get mPHY commandsRemoved Set/Get Record CommandsRemoved Set/Get chipset Init commandsListed examples for newly added commands	February 2018
0.7	<ul style="list-style-type: none">Align revision number	April 2018
0.8	<ul style="list-style-type: none">Align revision number	September 2018
0.8	<ul style="list-style-type: none">Reformatted all tools.	March 2019
0.8	<ul style="list-style-type: none">Removed CCT tool	July 2019
1.0	<ul style="list-style-type: none">Updated revision to 1.0	April 2020
1.1	<ul style="list-style-type: none">Added Set Current Clock Settings WrapperEXRemoved WDT	August 2020
1.2	<ul style="list-style-type: none">Updated the IccLibGetMphyVersion APIUpdated format for IccLibGetMphySettingsWrapper APIAdded guidance to build an executable (Chapter 3).	January 2021
1.21	<ul style="list-style-type: none">Added back Get MPHY Version	February 2021

§ §



1 Introduction

The purpose of the document is to provide guidance on the usage of the tools provided for Intel® Converged Security and Management Engine (Intel® CSME) Firmware Integrated Clock Controller (ICC) included within the Intel firmware kit.

1.1 Terminology

Acronym or Term	Definition
API	Application Programming Interface
BIOS	Basic Input Output System
CPU	Central Processing Unit
DLL	Dynamic Link Library
FW	Firmware
Intel® CSME	Intel® Converged Security and Management Engine
Intel® FIT	Intel® Flash Image Tool
Intel® ICCS	Intel® Integrated Clock Controller Services
Intel® MEI	Intel® Management Engine Interface (formerly HECI)
PCH	Platform Controller Hub
Permanent UOB	UOB that is applied on every boot.
UOB	Update on Boot. A record of ICC registers setting that is applied on the next platform boot.



1.2 Reference Documents

Document	Document Location
SPI Programming Guide	FW release kit
Intel® Converged Security Engine Firmware Bring Up Guide	FW release kit
External Design Specification (EDS)	RDC

§



2 Intel® ICCS SDK

2.1 Intel® ICCS SDK

Intel® Integrated Clock Controller Services (Intel® ICCS) provides a lot of flexibility for OS applications. To ease OS application development and to avoid erratic programming of ICC, Intel provides an ICC Control Library and abstracts ICC hardware from clock tuning applications such as BIOS.

2.2 Intel® ICCS Control Library SDK Features

ICC HW is only accessible to Intel® Management Engine (Intel® ME) and is accessible indirectly to the host software through a set of Intel® Management Engine Interface (Intel® MEI) APIs that are known to the Intel® Integrated Clock Controller Service. Intel does not expose those Intel MEI APIs and does not recommend OS applications to use them to keep platform stability.

Example of application that may call Intel® Integrated Clock Controller Service:

- Intel® Extreme Tuning Utility (Intel® XTU). This application can overclock or underclock platform BCLK (Processor clock).

2.3 Intel® Integrated Clock Controller Service Data Structures

Intel® Integrated Clock Controller Service provides a simplified ICC data structures and APIs for clock manipulation. The new data structure has significantly been reduced and simplified compared to previous generation of ICC control library. The ICC data structure is described in this section.

Table 2. Intel® Integrated Clock Controller Service API Data Structures

Name	Type	Description
ICC_HECI_CLOCK_ID	Enum	Defines the clock id for applicable clocks
ErrorCodes	UINT32	Returns error codes from Intel® Integrated Clock Controller Service API calls
ICC_GET_CLOCK_SETTIN GSEx	Struct	Contains the current clock setting
ICC_SET_CLOCK_SETTIN GSEx	Struct	Contains updatable clock setting
GET_MPHY_VERSION	Struct	Contains the current ChipsetInit version
IccLibGetMphyVersion	UINT32	Retrieves the MPHY version



Name	Type	Description
ICCLibGetMphySettingsWrapper	Method	Returns MPHY table with current settings
ICCLibSetMphySettingsWrapper	Method	Changes MPHY table current settings to the requested value

2.3.1 ICC_HECI_CLOCK_ID Type

The ICC_HECI_CLOCK_ID data structure provides the applicable clock to be selected with the following structure.

```
typedef enum
{
    ICC_HECI_PCIE_CLOCK_ID = 0,
    ICC_HECI_BCLK_CLOCK_ID = 1,
    ICC_HECI_WMPHY_CLOCK_ID = 2
}ICC_HECI_CLOCK_ID;
```

Table 3. Intel ICC_HECI_CLOCK_ID type

Name	Description
ICC_HECI_PCIE_CLOCK_ID	PCIe Clock (CPUBCLK Signal to CPU)
ICC_HECI_BCLK_CLOCK_ID	BCLK Clock (CPUBCLK Signal to CPU)
ICC_HECI_WMPHY_CLOCK_ID	White Mountain PLL

2.3.2 ErrorCodes Type

The ErrorCodes data structure provides description of the return error code from the Intel® Integrated Clock Controller Service.

The returned values are represented in UINT32 the following function is required to parse the values into char type:

```
const char* GetErrorStringByCode(const UINT32 errorCode);
```

2.3.3 ICC_GET_CLOCK_SETTINGSEx Type

The ICC_GET_CLOCK_SETTINGSEx structure provides all the clock settings details as the following:

```
typedef struct _ICC_GET_CLOCK_SETTINGSEx
{
    UINT32 Frequency;
    UINT32 UserFrequency;
    UINT32 MaxFrequency;
    UINT32 MinFrequency;
    UINT8 SscMode;
```




```

    UINT8  SscPercent;
    UINT8  MaxSscPercent;
    UINT16 CurrentFlags;
    UINT16 SupportFlags;
} ICC_GET_CLOCK_SETTINGSEx;

```

The returned values are represented in UINT32 the following function is required to retrieve the clock settings :

```

UINT32 IccLibGetCurrentClockSettingsWrapper (const ICC_HECI_CLOCK_ID
clockId, ICC_GET_CLOCK_SETTINGSEx * const clockSettings);

```

2.3.4 ICC_SET_CLOCK_SETTINGSEx Type

The ICC_SET_CLOCK_SETTINGSEx structure provides all the updatable clock settings as the following:

```

typedef struct _ICC_SET_CLOCK_SETTINGSEx
{
    UINT32 UserFrequency;
    UINT8  SscPercent;    // encoding example: 1.28% -> SSC_SPREAD value is
128
    BOOL   SetToDefault;
    BOOL   ForcePowerFlow;
} ICC_SET_CLOCK_SETTINGSEx;

```

The set values are represented in UINT32 the following function is required to configure the clock setting:

```

UINT32 IccLibSetCurrentClockSettingsWrapper(const ICC_HECI_CLOCK_ID
clockId, ICC_SET_CLOCK_SETTINGSEx * clockSettings);

```

2.3.5 GET MPHY Version

The GET_MPHY_VERSION structure provides the MPHY table information as the following:

```

typedef union _GET_MPHY_VERSION
{
    UINT32 data;
    struct
    {
        UINT32 CRC : 16;
        UINT32 Ver : 8;
        UINT32 Product_and_Stepping : 8;
    } Fields;
} GET_MPHY_VERSION;

```



The returned values are represented in UINT32 the following function is required to retrieve the MPHY version:

```
UINT32 IccLibGetMphyVersion(GET_MPHY_VERSION *survTable)
```

2.3.6 IccLibGetMphyVersion

The IccLibGetMphyVersion returned values are represented in UINT32. The method format is the following:

@param[out] **version** MPHY table version

@return **status**

```
UINT32 IccLibGetMphyVersion(GET_MPHY_VERSION *version)
```

2.3.7 IccLibGetMphySettingsWrapper

The IccLibGetMphySettingsWrapper returned values are represented in UINT32. The method format is the following:

@param[in] **length** num of bytes to read

@param[in] **offset** start offset

@param[out] **buffer** current settings of mphy table

@param[out] **bytesRead** num of bytes read

@param[out] **mphyTotalSize** size of MPHY table

@return **status**

```
UINT32 IccLibGetMphySettingsWrapper(UINT32 length, UINT32 offset, UINT8  
*buffer,UINT32 *bytesRead, UINT32* mphyTotalSize);
```

2.3.8 IccLibSetMphySettingsWrapper

The IccLibSetMphySettingsWrapper changes MPHY table current settings to the requested value,

* leaving all others intact. Error will be returned on range violation.

The method format is the following:

@param[in] **mphyFileName** the name of the settings file

@return **status**

```
UINT32 IccLibSetMphySettingsWrapper(char *mphyFileName);
```



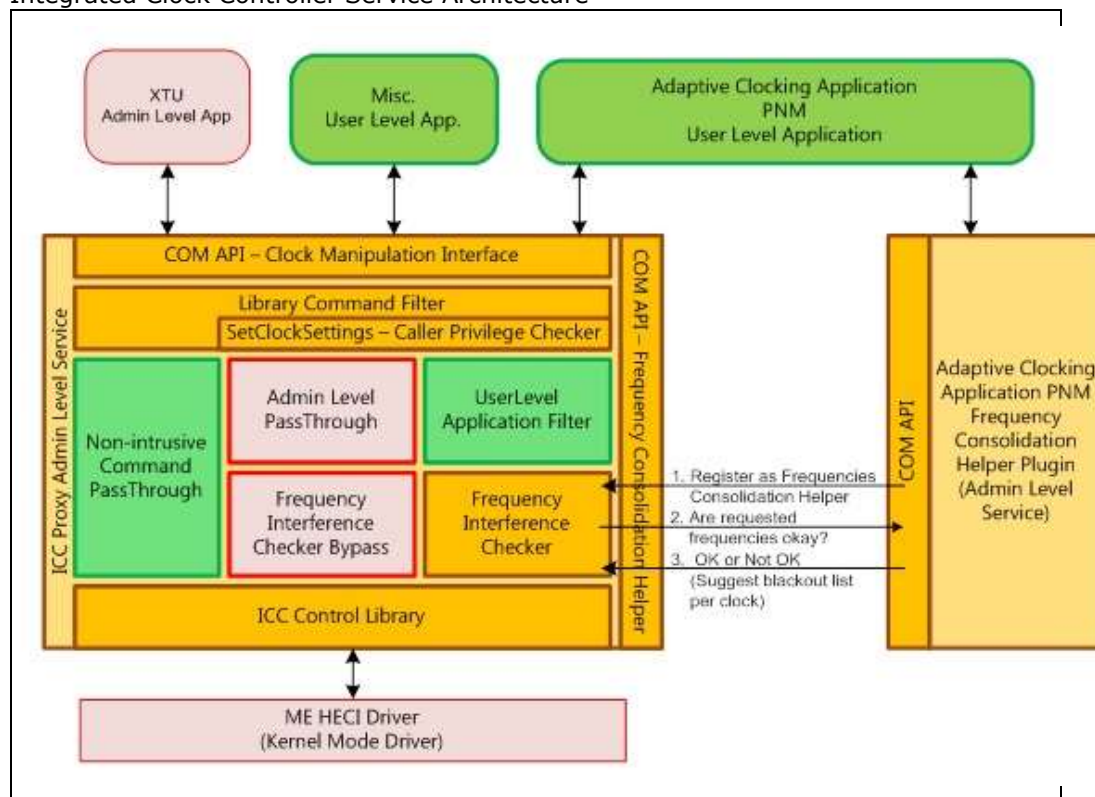
2.4 Intel® Integrated Clock Controller Service API – Clock Manipulation Interface

The Intel® Integrated Clock Controller Service provides a new set of simplified APIs in form of COM. The API exposed by the Intel® Integrated Clock Controller Service is used for communication between Intel® Integrated Clock Controller Service and client applications.

It is assumed that an application would only configure ICC clocks that “belong” to it. There is no provision in the library to lock a specific clock from a specific application.

Each API in the following section provides detail about Input and Output parameters for each API, as well as function prototype. For this guide, example in the following programming language will be provided: IDL and C++. Other programming languages are supported by the API, but an example will not be provided by this SDK user guide.

Note: Please note that Intel MEI Driver installation is required. Figure 1. Intel® Integrated Clock Controller Service Architecture





2.4.1 Get Current Clock Settings Wrapper

Through this function, the host application can get the runtime settings of the ICC clock identified by ICC_HECI_PCIE_CLOCK_ID parameter. The request returns the runtime settings of the clock.

Definition:

```
UINT32 IccLibGetCurrentClockSettingsWrapper(const ICC_HECI_CLOCK_ID  
clockId, ICC_GET_CLOCK_SETTINGSEX * const clockSettings);
```

Table 4. Get Clock Runtime Settings Parameters

Type	Field	Description
Input	clockId	Clock identifier
Output	clockSettings	Runtime settings of the clock. Meaningful only if successful status is received.

2.4.2 Set Current Clock Settings Wrapper

Host application calls this function to change the settings of one of the ICC clocks.

For host application (with Admin level) call, the request will be verified against the ICC Clock Range Definition Record.

For host application (with user level) call, the request will be verified against the ICC enhanced SKU Clock Range definition.

Note: If the requested frequency is not supported by the HW, it will be automatically rounded by Intel® Integrated Clock Controller Service to the nearest valid frequency. If there are two nearest valid frequencies (up and down), the lower value will be chosen.

Definition:

```
UINT32 IccLibSetCurrentClockSettingsWrapper(const ICC_HECI_CLOCK_ID  
clockId, ICC_SET_CLOCK_SETTINGSEX * clockSettings);
```

Table 5. Set Clock Runtime Settings Parameters

Type	Field	Description
Input	clockId	Clock identifier
Input	clockSettings	Runtime configuration for the clock.



2.4.3 Set Current Clock Settings WrapperEX

If dynamic tuning is supported, the host can immediately change settings of a given clock to the requested value, leaving all others intact. Error will be returned on range violation.

Definition

```
UINT32 IccLibSetCurrentClockSettingsWrapperEX(const ICC_HECI_CLOCK_ID  
clockId, ICC_SET_CLOCK_SETTINGSEX* clockSettings, ICC_SET_CLOCK_SETTING_TYPE  
type);
```

Table 6. Set Clock Runtime Settings Parameters

Type	Field	Description
Input	clockId	Clock identifier
Output	clockSettings	Runtime settings of the clock. Meaningful only if successful status is received.



3 *Building an Executable*

The following are general steps to build an executable file with Intel® ICC SDK files:

1. Create a new Visual Studio* project
2. Copy the following files to the project folder:
 - IccSdk.lib
 - Icc_sdk_api.h
3. Add to icc_sdk_api.h file:

```
#ifndef COMPILE_WITH_ICC_SDK_BINARIES  
  
#define COMPILE_WITH_ICC_SDK_BINARIES  
  
#endif
```
4. Link the library to the exe file:
 - Properties → Linker → Input → Additional Dependencies → Edit
 - ".\..\ADD_PATH_TO_LIB_FOLDER\IccSdk.lib"

