STD_CU43USBSW_V1.1E

STEP TECHNICA CO.,LTD.

STEP TECHNICA

CUnet (MKY43) USB Unit

CU-43USB

Software Manual

Introduction

This document describes API included with CU-43USB unit.
Before using the product, please check the latest information on our website.

• Target readers

・Those who do the programming with CU-43USB to build CUnet.

• Prerequisites

・Network technology
・Semiconductor products (especially microcontrollers and memory).
・Windows Application Programming

• Related manuals

・CUnet Introduction Guide (A Guide to the CUnet Protocol)
・CUnet Technical Guide for Network
・CUnet IC MKY43 User's Manual
・CU-43USB Product Manual

【Note】

Some terms in this manual are different from those used on our website and in our product brochures.
The brochure uses ordinary terms to help many people in various industries understand our products.
Please understand technical information on CUnet Family based on technical documents (manuals).

Revision history

Version	Date	Content	
		Page	Description
1.0E	AUG 2018		Issued the first edition
1.1E	APR 2020	1	Added Windows 10 as supported OS

Table of contents

Tables

Chapter 1 Summary

StepTechnica Co., Ltd., provides an API to access CU-43USB for the user application.
This document is intended for use with firmware version "V_1.0" and API version "1.0.0" of CU-43USB.
Please download the API from StepTechnica website's 'Downloads' page.
URL https://www.steptechnica.com/en/download/index.html

Chapter 2 System requirements

The API works correctly in the following operating systems.
· Windows 10 (64bit / 32bit)
· Windows 8.1 (64bit / 32bit)
· Windows 8 (64bit / 32bit)
· Windows 7 (64bit / 32bit)
This API can be called from Microsoft Visual Studio and VB6 etc.

Chapter 3 Copyright and disclaimer

StepTechnica Co., Ltd. assumes no responsibility for any results caused by using all softwares downloaded from our website.

Use API in proper ways with its instructions.

All specifications and contents are subject to change without prior notice.
We do not guarantee for any compatibility in the future.

We can not support for inquiries regarding OS or a development environment.

If you have found an error, please contact our system development department.

Chapter 4 File structure

The files in “DLL” folder are described in the following.

[DLL]

```
|
+ ---- [cu43usb.dll]    :  DLL body. Before your use, copy it to the system folder of Windows or the
|                           directory in which user program using this DLL is stored.
+ ---- [cu43usb.lib]    :  Import library
+ ---- [cu43usb.h]      :  DLL header file. Please include after than Windows.h.
```

Chapter 5 Limitations

This chapter describes limitations when creating an application using this API.

5.1 Multi-thread

This API Function cannot be used from multiple threads at the same time.
Consider not to generate a collective call if the application has multithreaded structure.

5.2 Timeout in USB communication

In this API, the maximum waiting time timeout of data transmission and reception between CU-43USB is 1 second.
Even if the timeout period is over, sending and receiving may not end in some of the system environments.
Return value of the API function returns an error when timeout has been occurred.

If a timeout occurs, the CUnet network and periodic update function stops in the internal API.
The CUnet network and periodic transmission function was able to stop normally, the following is set to the error code.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET(9)

If it fails to stop CUnet network and periodic transmission function, the following is set to the error code.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET(10)

After the timeout occurs, close the handle which is used in CubCloseHandle function, please reobtain a handle at CubOpenHandle function. Until a handle is reobtained, the return parameter of the API function other than CubGetVer sion, CubCountDevice, CubSearchBoard, CubGetLastError, CubOpenHandle and CubCloseHandle returns an error.
At that time, CubGetLastError function returns CUB_ERR_REACQUISITION_OF_HANDLE(11).

5.3 Power saving mode

This product is not applied to the power saving mode of PC (personal computer).

Chapter 6 API specifications

This chapter describes API specifications.
This API is prepared for easy operation of CU-43USB for user application.
In addition to the normal function to read and write to MKY43, this API has an internal function to sample all global memory and MFR of MKY43 at specified cycle. This function is called "periodic update".

The API function list is shown in Table 6-1.

Table 6-1 API functions

API function	Description
CubGetVersion	Obtains the version number of API
CubGetLastError	Obtains the termination status of API function
CubOpenHandle	Opens a handle of CU-43USB
CubCloseHandle	Closes a handle obtained by CubOpenHandle
CubCountDevice	Obtains the number of CU-43USB connected to PC
CubBoardID	Obtains the board ID
CubSearchBoard	Obtains the number of CU-43USB connected to PC and obtain the board ID
CubResetMKY43	Orders a reset to MKY43
CubStartAutoTrans	Starts periodic update
CubStopAutoTrans	Stops periodic update
CubReadWord	Reads 2 bytes of data from the specified address
CubWriteWord	Writes 2 bytes of data to the specified address
CubReadProtect	Data read from global memory with hazard protect function
CubWriteProtect	Data write to global memory with hazard protect function
CubReadGM	Obtains the latest data of all global memory by periodic update
CubReadMFR	Obtains the latest data of all MFR by periodic update
CubReadData	Reads data of the specified word length from the specified address
CubWriteData	Writes data of the specified word length to the specified address
CubGetFirmwareVersion	Obtains the firmware version number of CU-43USB

6.1 CubGetVersion

Format

UINT CubGetVersion(void);

Function

Obtains the version number of API

Parameter

None

Return value

Version number of API (Hexadecimal BCD code)
(Major Number + Minor Number + Update Number)

Error code

The error code and error factor returned by the CubGetLastError after executing this function is as follows.

CUB_SUCCESS Terminated normally

Note

The configuration of API version number is as shown in Table 6-2.
The reasons for updating the version number are as follows.

Major Number : The revision with no backword compatibility such as API specification change.
Minor Number : The revision with backword compatibility such as an addition of API function.
Update Number : The revision with no specification change such as bug fixes.

Table 6-2 Version numbering

Return value (Example)	Major Number (Bit 15 - 8)	Minor Number (Bit 7 - 4)	Update Number (Bit 3- 0)
0x0102	1	0	2
0x1398	13	9	8

6.2 CubGetLastError

Format

UINT CubGetLastError(void);

Function

Obtains the termination state of the API function called last time

Parameter

None

Return value

Returns the error code defined in cu43usb.h.

Note

The error codes defined in cu43usb.h. are shown in Table 6-3.

Table 6-3 Error code list

Error Code	Value	Content
CUB_SUCCESS	0	Terminated normally
CUB_ERR_DEVICENOTEXIST	1	Device does not exist.
CUB_ERR_ALREADYOPENED	2	Handle has been already opened.
CUB_ERR_CLOSED	3	'CubOpenHandle' has never been called.
CUB_ERR_INVALIDPARAM	4	Called with invalid parameter.
CUB_ERR_NORESOUCE	5	No resource to execute the process
CUB_ERR_FAILED	6	The process failed due to unknown reason.
CUB_ERR_AUTO_TRANS_ALREADY_START	7	Periodic update has already started.
CUB_ERR_AUTO_TRANS_STOP	8	Periodic update has not started.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	9	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	10	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	11	Handle has not been reobtained.
CUB_ERR_NOT_SUPPORT_FIRM_VERSION	12	Unsupported firmware version
CUB_ERR_INVALID_SEQUENCE_NUMBER	13	Invalid sequence number
CUB_NOTCALLYET	99	CUBAPI has never been called.

6.3 CubCountDevice

Format

INT CubCountDevice(void);

Function

Obtains the number of CU-43USB connected to PC

Parameter

None

Return value

Returns the number of CU-43USB connected to PC

-1	:	5 or more
0	:	Not connected
1 to 4	:	1 to 4

Error Code

The error code and error factor returned by the CubGetLastError after executing this function is as follows.

CUB_SUCCESS	Terminated normally

Note

No more than five devices can be connected to a PC.

6.4 CubBoardID

Format

INT CubBoardID(HANDLE CUBHandle);

Function

Obtains board ID of CU-43USB

Parameter

HANDLE CUBHandle	Handle value of CU-43USB

Return value

Succeeded: Board ID (0 to 3) is returned.

Failed : -1 is returned.

Error Code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.5 CubSearchBoard

Format

BOOL CubSearchBoard(BYTE *board_num , BYTE *board_id_list);

Function

Obtains the number of CU-43USB connected to PC and obtain the Board ID list

Parameter

*board_num — Specify the address to the byte type variable in which the number of boards is set.

The meanings of the set values are as follows.

-1	:	5 or more
0	:	Not connected
1 to 4	:	Number of boards identified

*board_id_list — To receive the board ID, specify a pointer to an array with four byte types. It is also possible to specify NULL.
If NULL has been specified, only the number of boards is returned.
The meanings of the set values are as follows.

0x00 to 0x03	:	Board ID
0x80	:	Handle value has already been obtained by CubOpenHandle.
0xFF	:	No board has been identified.

Return value

Succeeded: TRUE(1) is returned.
Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS Terminated normally
CUB_ERR_INVALIDPARAM * board_num is NULL
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET
Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET
Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_INVALID_SEQUENCE_NUMBER
Invalid sequence number
CUB_ERR_FAILED The process failed due to unknown reason.

Addendum

The board ID is set by Option switch. If two or more CU-43USB devices are connected, it can be distinguished by board IDs.
This API function can identify up to four CU-43USB devices. Specify the byte type array as a parameter as shown below.

```
BYTE board_num;
BYTE board_id_list[4];
CubSearchBoard（&board_num, &board_id_list[0]）;
```

As an example, three CU-43USB devices are connected to the PC, and each board IDs are set in sequence ;
1st board ID = 0, 2nd board ID = 1, 3rd board board ID = 2.
board_num = 3;
board_id_list[0] = 0、board_id_list[1] = 2、board_id_list[2] = 1、board_id_list[3] = 0xFF

If the devices have been identified by the PC in sequence with first, third, and second, and run CubSearchBoard, board number and its IDs are returned as follows.
board_num = 3;
board_id_list[0] = 0、board_id_list[1] = 2、board_id_list[2] = 1、board_id_list[3] = 0xFF

6.6 CubStartAutoTrans

Format

BOOL CubStartAutoTrans(HANDLE CUBHandle, WORD MfCnt);

Function

Starts periodic update of all global memory and MFR of CU-43USB

The update cycle can be specified in units of 125 µs.

The updated data is retained inside the API. Retained data can be obtained with CubReadGM, CubReadMFR.

Parameter

HANDLE CUBHandle	Handle value of CU-43USB
WORD MfCnt	Set update cycle. The update cycle can be specified in units of 125 µs from 1 ms to 100 ms. Regarding the update cycle interval, refer to Table 6-4 to set the periodic interval. The setting values other than in Table 6-4 will be an error.

Table 6-4 Update cycle setting list

Setting value	Update cycle（µsec）
8	1,000(1msec)
9	1,125
10	1,250
:	:
797	99,625
798	99,750
799	99,875
800	100,000 (100msec)

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. MfCnt is out of range
CUB_ERR_AUTO_TRANS_ALREADY_START	Periodic update has already started.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

Note

Please be aware that periodic updating may not be executed due to the specifications of the PC, or other applications running on the same PC.

When using CubReadGM, CubReadMFR, please use this API to enable periodic update.

6.7 CubStopAutoTrans

Format

BOOL CubStopAutoTrans(HANDLE CUBHandle);

Function

Stops periodic update of all global memory and MFR of CU-43USB

Parameter

HANDLE CUBHandle	Handle value of CU-43USB

Return value

Succeeded:TRUE(1) is returned.
Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified.
CUB_ERR_AUTO_TRANS_STOP	Periodic update has not started.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.8 CubOpenHandle

Format

HANDLE CubOpenHandle(int index_no);

Function

Opens handles to the CU-43USB

Parameter

int index_no	Index number You can specify an index number from 0 to 3. If just one CU-43USB is connected to PC, set 0. For more information, see "Addendum".

Return value

Succeeded: 1 or greater value is returned.
Failed : -1 (INVALID_HANDLE_VALUE) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_ALREADYOPENED	Handle has been already opened.
CUB_ERR_DEVICENOTEXIST	Device does not exist.
CUB_ERR_NOT_SUPPORT_FIRM_VERSION	Unsupported firmware version
CUB_ERR_REACQUISITION_OF_HANDLE	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

Addendum

It's not necessary to run CubSearchBoard when just one CU-43USB is connected to PC.
If two or more CU-43USB devices are connected to PC, you must run "CubSearchBoard" in advance to check which CU-43USB to manipulate.

As an example, three CU-43USB devices are connected to the PC, and each board IDs are set in sequence ; 1st board ID = 0, 2nd board ID = 1, 3rd board board ID = 2.
To obtain the handle value of Board ID=2, operate as follows.

```
BYTE board_num;
BYTE board_id_list[4];
CubSearchBoard (&board_num, &board_id_list[0]);
```

Assuming that the results of executing in the above was the following.

board_id_list[0]=0, board_id_list[1]=2, board_id_list[2]=1, board_id_list[3]=0xFF

In this case, you see that index number 1 is the board ID=2.

That means 1 is the index number, the parameter of CubOpenHandle.

Close the handle with CubCloseHandle at finishing the program.

6.9 CubCloseHandle

Format

BOOL CubCloseHandle(HANDLE CUBHandle);

Function

Closes the handle obtained by CubOpenHandle

Stops periodic update as well if it's running

Parameter

HANDLE CUBHandle	Handle value of CU-43USB

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.10 CubResetMKY43

Format

BOOL CubResetMKY43(HANDLE CUBHandle);

Function

Resets MKY43

Parameter

HANDLE CUBHandle	Handle value of CU-43USB

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.11 CubReadWord

Format

BOOL CubReadWord(HANDLE CUBHandle, const ULONG Adr, WORD *Dat);

Function

Reads 2 bytes of data from the specified address

Parameter

HANDLE CUBHandle	Handle value of CU-43USB
const ULONG Adr	Address value Input conditions are the following. ・Multiples of 2 ・Input range : 0x0000 - 0xF02
WORD *Dat	The storage address of read data

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. Adr is out of range. Adr value is not multiple of 2. NULL has been specified to *Dat.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.12 CubWriteWord

Format

BOOL CubWriteWord(HANDLE CUBHandle, const ULONG Adr, const WORD Dat);

Function

Writes 2 bytes of data from the specified address

Parameter

HANDLE CUBHandle	The handle value of CU-43USB
const ULONG	Adr address value Input conditions are the following. Multiples of 2 Input range : 0x0000 - 0xF02
cconst WORD Dat	Write data

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. Adr is out of range. Adr value is not multiple of 2.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.13 CubReadProtect

Format

BOOL CubReadProtect (HANDLE CUBHandle, WORD BlockNo, void *Data);

Function

Obtains the latest data of all control words by periodic update

Error is returned when CubReadProtect has been called while periodic update was stopping.

Parameter

HANDLE CUBHandle	The handle value of CU-43USB
WORD BlockNo	Memory block number Input conditions are the following. ・Input range : 0 to 63
void *Data	The storage address of 8 bytes read data

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. BlockNo is out of range. NULL has been specified to *Data.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.14 CubWriteProtect

Format

BOOL CubWriteProtect (HANDLE CUBHandle, WORD BlockNo, void *Data);

Function

Writes data to global memory using hazard protect function

Parameter

HANDLE CUBHandle	The handle value of CU-43USB
WORD BlockNo	Memory block number Input conditions are the following. ・Input range : 0 to 63
void *Data	The storage address of 8 bytes write data

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. NULL has been specified to *Data.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.15 CubReadGM

Format

BOOL CubReadGM (HANDLE CUBHandle, void*Data);

Function

Obtains the latest data of all global memory by periodic update

Error is returned when CubReadGM has been called while periodic update was stopping.

Parameter

HANDLE CUBHandle	The handle value of CU-43USB
void *Data	The storage address of 512 bytes data

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CUBGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. NULL has been specified to *Data.
CUB_ERR_AUTO_TRANS_STOP	Periodic update is stopping.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_FAILED	The process failed due to unknown reason.

Note

CubReadGM is an API that obtains data by periodic update, which is not accessed to MKY43 in direct.
When obtaining control word from MKY43 directly, use "CubReadWord", "CubReadData","CubReadProtect".

6.16 CubReadMFR

Format

BOOL CubReadMFR (HANDLE CUBHandle, void*Data);

Function

Obtains the latest data of MFR by periodic update

Error is returned when CubReadMFR has been called while periodic update was stopping.

Parameter

HANDLE CUBHandle	The handle value of CU-43USB
void *Data	The storage address of 8 bytes data

Return value

Succeeded:TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. NULL has been specified to *Data.
CUB_ERR_AUTO_TRANS_STOP	Periodic update is stopping
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_FAILED	The process failed due to unknown reason.

Note

CubReadMFR is an API that obtains data by periodic update, which is not accessed to MKY43 in direct. When obtaining MFR from MKY43 directly, use "CubReadWord", "CubReadData".

6.17 CubReadData

Format

BOOL CubReadData (HANDLE CUBHandle, WORD Adr, WORD WordLen, void *Data);

Function

Reads data of the specified word length from the specified address

Parameter

HANDLE CUBHandle	The handle value of CU-43USB
WORD Adr	Address value Input conditions are the following Multiples of 2 Input range : 0x0000 - 0x07FE
WORD WordLen	Word length Input conditions are the following Input range : 0x0001 - 0x0400
void *Data	The storage address of read data

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. Specified Adr is out of range. Adr value is not multiple of 2. Specified WordLen is out of range. Specified read range exceeded 0x800. NULL has been specified to *Data.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.18 CubWriteData

Format

BOOL CubWriteData(HANDLE CUBHandle, WORD Adr, WORD WordLen, void *Data);

Function

Writes data of the specified word length to the specified address

Parameter

HANDLE CUBHandle	The handle value of CU-43USB
WORD Adr	Address value Input conditions are the following Multiples of 2 Input range : 0x0000 - 0x07FE
WORD WordLen	Word length Input conditions are the following. • Input range : 0x0001 - 0x0400
void *Data	The storage address of write data

Return value

Succeeded: TRUE(1) is returned.

Failed : FALSE(0) is returned.

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified. Specified Adr is out of range. Adr value is not multiple of 2. Specified WordLen is out of range. Specified write range exceeded 0x800. NULL has been specified to *Data.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

6.19 CubGetFirmwareVersion

Format

INT CubGetFirmwareVersion(HANDLE CUBHandle);

Function

Obtains the firmware version number of CU-43USB

Parameter

HANDLE CUBHandle	The handle value of CU-43USB.

Return value

Version number of firmware (Hexadecimal BCD code)
(Major Number + Minor Number + Update Number)

Error code

The error codes and error factors returned by the CubGetLastError after executing this function are as follows.

CUB_SUCCESS	Terminated normally
CUB_ERR_INVALIDPARAM	Invalid CUBHandle is specified.
CUB_ERR_USB_TIMEOUT_SUCCESS_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication was successfully stopped.
CUB_ERR_USB_TIMEOUT_FAILED_STOP_CUNET	Timeout has occurred during USB communication, and CUnet communication failed to be stopped.
CUB_ERR_FAILED	The process failed due to unknown reason.
CUB_ERR_REACQUISITION_OF_HANDLE	Handle has not been reobtained.
CUB_ERR_INVALID_SEQUENCE_NUMBER	Invalid sequence number
CUB_ERR_FAILED	The process failed due to unknown reason.

Note

The configuration of firmware version number is as shown in Table 6-5.
The reasons for updating the version number are as follows.

Major Number : The revision with no backword compatibility such as firmware specification change.
Minor Number : The revision with backword compatibility such as an addition of firmware function.
Update Number : The revision with no specification change such as bug fixes.

Table 6-5 Version numbering

Return value (Example)	Major Number (Bit 15 - 8)	Minor Number (Bit 7 - 4)	Update Number (Bit 3- 0)
0x0102	1	0	2
0x1398	13	9	8

Chapter 7 Appendix

7.1 Sample program

The sample of initializing and finishing program to control CU-43USB is the following.
For the structure and functions of MKY43 register, please refer to "Chapter 5: Register Reference" described in "MKY43 User's Manual".

```
int main（int argc, char argv[]）
{

    unsigned char buf[0x580];
    unsigned char board_count;
    unsigned char board_id_list[4];
    /** Check an API version number */
    int version=CubGetVersion（）；
    if（version < 0x100 || version > 0x199）{
      printf（"This version of cu43usb.dll is incompatible\n"）；
      exit（1）；
    }
    /** Search CU-43USB
    * Up to four CU-43USB devices can be identified. When five or more devices are connected to PC,
      it returns an error.
    * The number of CU-43USB devices connected to PC and its Board ID list are returned.
    * It's not necessary to execute CubSearchBoard when just one CU-43USB device is connected to PC.
    */
    if(CubSearchBoard(&board_count,&board_id_list[0]) == FALSE){
      exit（1）；
    }

    if（board_count == 0）{
      printf（"No CU-43USB is connected to PC. \n"）；
      exit（1）；
    } else if（board_count == 0xFF）{
      printf（"Number of  CU-43USB devices connected to PC exceeded the limit.\n"）；
      exit（1）；
    }

    /** A handle corresponded with CU-43USB to control is generated.
    * If only one CU-43USB is connected to PC, open handle with 0 parameter.
    */
    HANDLE dev_handle;
    dev_handle=CubOpenHandle（0）；
    if（dev_handle == INVALID_HANDLE_VALUE）{
      printf（"Failed to obtain a handle value to CU-43USB.\n"）；
      exit（1）；
    }

    memset（buf, 0, sizeof（buf））；
```

```
        /** Initializing CU-43USB */
        // Clear global memory
        CubWriteData（dev_handle, 0, 0x100, buf）;
        // Clear mail sending buffer
        CubWriteData（dev_handle, 0x200, 0x80, buf）;
        // Clear mail receiving buffer 0
        CubWriteData（dev_handle, 0x400, 0x80, buf）;
        // Clear mail sending buffer 1
        CubWriteData（dev_handle, 0x500, 0x80, buf）;

        // Transfer to GM mode
        CubWriteWord（dev_handle, 0x366, 0x8000）;
        // Change setting to SA-1, OWN-1, BPS-3Mbps
        CubWriteWord（dev_handle, 0x356, 0x0141）;
        // Release GMM mode
        CubWriteWord（dev_handle, 0x366, 0）;

        /** Start communication
        * Set "1" to START bit of SCR to start network.
        */
        CubWriteWord（dev_handle, 0x366, 0x0100）;

        /** Start periodic update. It's not necessary to execute when CubReadGM, CubReadMFR are not used.
        * Data sending at 4000μs（4msec）cycle
        */
        CubStartAutoTrans（dev_handle, 32）;

        /** -- Describe user process here -- **/

        /** Stop periodic update.（It's not necessary to execute when CubStartAutoTrans is not used.）*/
        CubStopAutoTrans（dev_handle）;

        /** Stop CUnet communication **/
        // Set 0 to SCR to stop CUnet communication.
        CubWriteWord（dev_handle, 0x366, 0x0000）;

        /** Close the generated handle. */
        CubCloseHandle（dev_handle）;
        return 0;
}
```

Notes

1. The information in this document is subject to change without prior notice.
   Before using this product, please confirm that this is the latest version of this document.
2. Technical information in this document, such as explanations and circuit examples, are just for references to use this product in a proper way.
   When actually using this product, always fully evaluate the entire system according to the design purpose based on considerations of peripheral circuits and environment.
   We assume no responsibility for any incompatibility between this product and your system.
3. We assume no responsibility whatsoever for any losses or damages arising from the use of the information, products, and circuits in this document, or for infringement of patents and any other rights of a third party.
4. When using this product and the information and circuits in this document, we do not guarantee the right to use any property rights, intellectual property rights, and any other rights of a third party.
5. This product is not designed for use in critical applications, such as life support systems.
   Contact us when considering such applications.
6. No part of this document may be copied or reproduced in any form or by any means without prior written permission from StepTechnica Co., Ltd.

■Developed and manufactured by

StepTechnica Co., Ltd.

757-3, Shimofujisawa, Iruma, Saitama
https://www.steptechnica.com/en/index.html
info@steptechnica.com

CUnet（MKY43）USB Unit
CU-43USB
Software Manual

Document No STD_CU43USBSW_V1.1E
Issued: April 2020