Message Digest Algorithm 5 User Guide

Version 1.20 BETA

For use with Message Digest Algorithm 5 (MD5) module versions 1.07 and above

Date: 11-Jan-2017 13:33

Table of Contents

1 System Overview

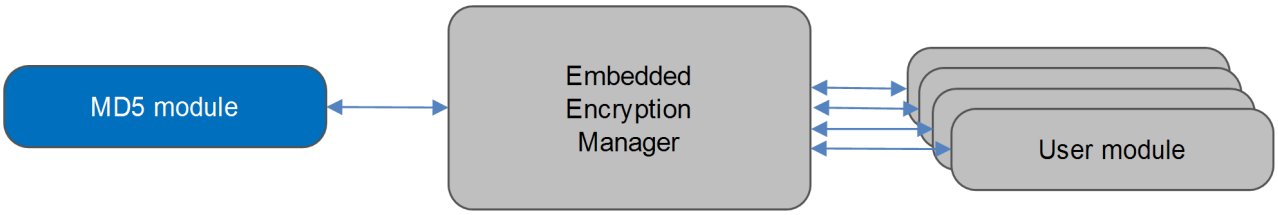
1.1 Introduction

This guide is for those who want to generate hash codes for data by using the MD5 Message Digest Algorithm. The MD5 module implements the MD5 hash algorithm. It supports MD5-HMAC and MD5-HMAC-96.

Note: The MD5 method, which produces a 16 byte hash value, has been superseded by the SHA hash algorithms.

You register the MD5 module with HCC's Embedded Encryption Manager (EEM), making it usable by other applications (for example, HCC's TLS/DTLS) through a standard interface. The EEM is the core component of HCC's encryption system.

The system structure is shown below:



Note:

- Although every attempt has been made to simplify the system's use, to get the best results you must understand clearly the requirements of the systems you design.
- HCC Embedded offers hardware and firmware development consultancy to help you implement your system; contact sales@hcc-embedded.com.

1.2 Feature Check

The main features of the MD5 module are the following:

- Conforms to the HCC Advanced Embedded Framework.
- Conforms to the HCC Coding Standard including full MISRA compliance.
- Designed for integration with both RTOS and non-RTOS based systems.
- Conforms to the HCC Embedded Encryption Manager (EEM) standard and is compatible with the EEM.
- Supports MD5-HMAC and MD5-HMAC-96.
- Can be verified using the HCC Encryption Test Suite.

1.3 Packages and Documents

Packages

The table below lists the packages that you need in order to use this module.

Package	Description
hcc_base_docs	This contains the two guides that will help you get started.
enc_base	The EEM base package.
enc_md5	The MD5 package described in this document.

Documents

For an overview of HCC verifiable embedded network encryption, see Product Information on the main HCC website.

Readers should note the points in the HCC Documentation Guidelines on the HCC documentation website.

HCC Firmware Quick Start Guide

This document describes how to install packages provided by HCC in the target development environment. Also follow the *Quick Start Guide* when HCC provides package updates.

HCC Source Tree Guide

This document describes the HCC source tree. It gives an overview of the system to make clear the logic behind its organization.

HCC Embedded Encryption Manager User Guide

This document describes the EEM.

HCC Message Digest Algorithm 5 User Guide

This is this document.

1.4 Change History

This section includes recent changes to this product. For a list of all the changes, refer to the file **hcc/history/enc/enc_md5.txt** in the distribution package.

Version	Changes
1.07	Made function **md5_init_fn()** stateful. MD5 HMAC and MD5 HMAC-96 are now stateful. The final digest is calculated when output data is not NULL.
1.06	Added stateful driver for MD5 hash. The digest is calculated when the output buffer of *enc_driver_hash* is not NULL, otherwise the result is accumulated.
1.05.	Added MD5 OID numbers to the API.
1.04	Added drivers for calculating MD5-HMAC and MD5-HMAC-96. (HMAC stands for Hash Message Authentication Code.) Made code more modular.
1.03	Simplified driver implementation to match SHA driver.
1.02	Simplified MD5 calculation algorithm.
1.01	Changed version number to match new **enc_base** package.

2 Source File List

This section describes all the source code files included in the system. These files follow the HCC Embedded standard source tree system, described in the *HCC Source Tree Guide*. All references to file pathnames refer to locations within this standard source tree, not within the package you initially receive.

> **Note:** Do not modify any file except the configuration file.

2.1 API Header File

The file **src/api/api_enc_sw_md5.h** is the only file that should be included by an application using this module. It defines the Application Programming Interface (API) functions.

2.2 Configuration File

The file **src/config/config_enc_sw_md5.h** contains all the configurable parameters of the system. Configure these as required. For detailed explanation of these options, see Configuration Options.

2.3 System File

The file **src/enc/software/md5/md5.c** contains the source code. **This file should only be modified by HCC**.

2.4 Version File

The file **src/version/ver_enc_sw_md5.h** contains the version number of this module. This version number is checked by all modules that use this module to ensure system consistency over upgrades.

3 Configuration Options

Set the system configuration options in the file **src/config/config_enc_sw_md5.h**.

MD5_INSTANCE_NR

The number of allowed MD5 stateful instances. The default is 1. Set this to 0 to disable the driver.

MD5_HMAC_INSTANCE_NR

The number of allowed MD5 HMAC stateful instances. The default is 1. Set this to 0 to disable the driver.

MD5_HMAC96_INSTANCE_NR

The number of allowed MD5 HMAC96 stateful instances. The default is 1. Set this to 0 to disable the driver.

4 Application Programming Interface

This section describes the single Application Programming Interface (API) functions, the hash output size, key lengths, and the error codes.

4.1 Functions

The functions are the following:

Function	Description
md5_init_fn()	Called from the EEM, this forwards the structure containing MD5 functions to the EEM.
md5_hmac_init_fn()	Called from the EEM, this forwards the structure containing MD5-HMAC functions to the EEM.
md5_hmac96_init_fn()	Called from the EEM, this forwards the structure containing MD5-HMAC-96 functions to the EEM.

md5_init_fn

Call this function from the EEM to forward the structure containing MD5 functions to it.

Note: This implementation is stateful so a hash can be calculated from multiple data buffer inputs (partial data). To calculate a hash from partial data, set the output buffer to NULL.

Format

```
t_enc_ret md5_init_fn ( t_enc_driver_fn const * * const pp_encdriver )
```

Arguments

Parameter	Description	Type
pp_encdriver	A pointer to a structure containing MD5 functions.	t_enc_driver_fn * *

Return Values

Return value	Description
ENC_SUCCESS	Successful execution.
ENC_INVALID_ERR	The module has already been initialized.

md5_hmac_init_fn

Call this function from the EEM to forward the structure containing MD5-HMAC functions to it.

This algorithm calculates the MD5-HMAC. The encryption function calculates the HMAC value. The HMAC output length is equal to MD5_HMAC_OUT_LEN.

> **Note:** This implementation is stateful so a hash can be calculated from multiple data buffer inputs (partial data). To calculate a hash from partial data, set the output buffer to NULL.

Format

```
t_enc_ret md5_hmac_init_fn ( t_enc_driver_fn const * * const pp_encdriver )
```

Arguments

Parameter	Description	Type
pp_encdriver	A pointer to a structure containing MD5-HMAC functions.	t_enc_driver_fn * *

Return Values

Return value	Description
ENC_SUCCESS	Successful execution.
ENC_INVALID_ERR	The module has already been initialized.

md5_hmac96_init_fn

Call this function from the EEM to forward the structure containing MD5-HMAC96 functions to it.

This algorithm calculates the MD5-HMAC-96. The encryption function calculates the HMAC value. The HMAC output length is equal to MD5_HMAC_96_OUT_LEN.

> **Note:** This implementation is stateful so a hash can be calculated from multiple data buffer inputs (partial data). To calculate a hash from partial data, set the output buffer to NULL.

Format

```
t_enc_ret md5_hmac96_init_fn ( t_enc_driver_fn const * * const pp_encdriver )
```

Arguments

Parameter	Description	Type
pp_encdriver	A pointer to a structure containing MD5-HMAC-96 functions.	t_enc_driver_fn * *

Return Values

Return value	Description
ENC_SUCCESS	Successful execution.
ENC_INVALID_ERR	The module has already been initialized.

4.2 Hash Output Size

The hash output size is defined in the file **src/api/api_enc_sw_md5.h**.

Name	Value	Description
MD5_OUT_LEN	16	The size of the hash output in bytes.

4.3 Key Lengths

The key lengths are as follows:

Name	Value	Description
MD5_HMAC_96_KEY_LEN	16	The length of the MD5-HMAC-96 key value.
MD5_HMAC_96_OUT_LEN	12	The length of the MD5-HMAC-96 MAC value.
MD5_HMAC_OUT_LEN	16	The length of the MD5-HMAC MAC value.

4.4 Error Codes

The table below lists the error codes that may be generated by the API calls.

Error code	Value	Meaning
ENC_SUCCESS	0	Successful execution.
ENC_INVALID_ERR	1	The module has already been initialized.

5 Integration

The MD5 module is designed to be as open and as portable as possible. No assumptions are made about the functionality, the behavior, or even the existence, of the underlying operating system. For the system to work at its best, perform the porting outlined below. This is a straightforward task for an experienced engineer.

5.1 OS Abstraction Layer

The module uses the OS Abstraction Layer (OAL) that allows it to run seamlessly with a wide variety of RTOSes, or without an RTOS.

The module uses the following OAL components:

OAL Resource	Number Required
Tasks	0
Mutexes	1
Events	0

5.2 PSP Porting

The Platform Support Package (PSP) is designed to hold all platform-specific functionality, either because it relies on specific features of a target system, or because this provides the most efficient or flexible solution for the developer. For full details of these elements, see the *HCC Base Platform Support Package User Guide*.

The module makes use of the following standard PSP functions:

Function	Package	Element	Description
psp_memcpy()	psp_base	psp_string	Copies a block of memory. The result is a binary copy of the data.
psp_memset()	psp_base	psp_string	Sets the specified area of memory to the defined value.

The module makes use of the following standard PSP macros:

Macro	Package	Element	Description
PSP_RD_LE32	psp_base	psp_endianness	Reads a 32 bit value stored as little-endian from a memory location.
PSP_WR_LE32	psp_base	psp_endianness	Writes a 32 bit value to be stored as little-endian to a memory location.