Digital Signature Standard User Guide

Version 1.10 BETA

For use with Digital Signature Standard (DSS) module versions 1.04 and above

Date: 09-Mar-2016 15:22

Table of Contents

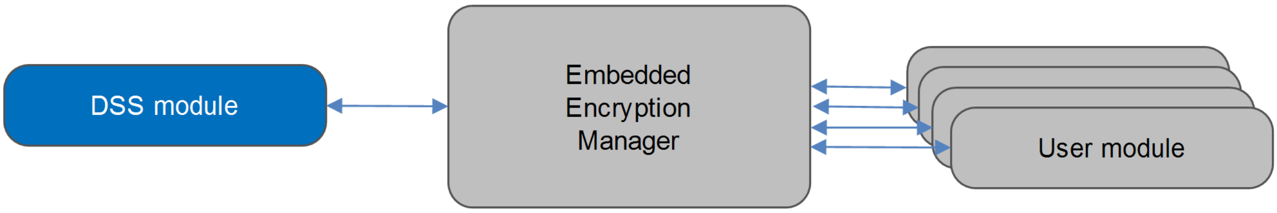
1 System Overview

1.1 Introduction

This guide is for those who want to implement encryption using the Digital Signature Standard (DSS). The DSS module implements the Digital Signature Standard, which uses the Digital Signature Algorithm (DSA).

You register the DSS module with HCC's Embedded Encryption Manager (EEM), making it usable by other applications (for example, HCC's TLS/SSL) 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 DSS module are the following:

- It conforms to the HCC Advanced Embedded Framework.
- It conforms to the HCC Coding Standard including full MISRA compliance.
- It can be used with or without an RTOS.
- It conforms to the HCC Embedded Encryption Manager (EEM) standard and is compatible with the EEM.
- It can be verified by 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_dss	The DSS 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 Big Number Arithmetic Library API Specification

This document describes the big number arithmetic functions, some of which are used by this module.

HCC Digital Signature Standard 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 **src/history/enc/enc_dss.txt** in the distribution package.

Version	Changes
1.04	Removed mutex clearing during initialization. This could cause OS compile error.
1.03	Changes for big number arithmetic.
1.02	Changed module to work with big-endian architecture.

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 files except the configuration file.

2.1 API Header File

The file **src/api/api_enc_sw_dss.h** should be included by any application using the system. This is the only file that should be included by an application using this module. For details of the functions, see Application Programming Interface.

2.2 Configuration File

The file **src/config/config_enc_sw_dss.h** contains the configurable parameters of the system. Configure these as required. This is the only file in the module that you should modify.

2.3 System File

The file **src/enc/software/dss/dss.c** contains the source code.

This file should only be modified by HCC.

2.4 Version File

The file **src/version/ver_enc_sw_dss.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_dss.h**. This section lists the available configuration options and their default values.

DSS_INSTANCE_NR

The maximum number of DSS instances. The default is 2.

DSS_SHA1_OUT_LEN

The SHA-1 hash output length. The default is SHA1_OUT_LEN.

4 Application Programming Interface

This section describes the Application Programming Interface (API) functions and the error codes.

4.1 Module Management

dss_init_fn

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

Format

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

Arguments

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

Return Values

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

dss_register_sha1

Use this function to register a Secure Hash Algorithm 1 (SHA-1) interface handler in the DSS module.

Note: Call this function just once during system initialization.

Format

```
t_enc_ret dss_register_sha1( t_enc_ifc_hdl sha1_hdl )
```

Arguments

Parameter	Description	Type
sha1_hdl	The SHA-1 interface handler.	t_enc_ifc_hdl

Return Values

Return value	Description
ENC_SUCCESS	Successful execution.

4.2 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 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 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 functions, see the *Platform Support Package (PSP) Base User Guide*.

The module makes use of the following standard PSP function:

Function	Package	Element	Description
PSP_WR_8BITARRAY_OFFSET()	psp_base	psp_array32	

The module makes use of the following standard PSP macro:

Macro	Package	Element	Description
PSP_WR_BE16	psp_base	psp_endianness	Writes a 16 bit value to be stored as big-endian to a memory location.

The module uses the following big number arithmetic functions from the Big Number Arithmetic Library API.

Note: To improve performance, you can replace these functions with optimized or hardware-supported versions. For details, see the *HCC Big Number Arithmetic Library API Specification*.

Function	Description
bn_add()	Adds two numbers.
bn_assign_be_buf()	Assigns a little-endian buffer to a big number, based on a big-endian buffer.
bn_assign_le_buf()	Assigns a buffer to a big number, based on a little-endian buffer.
bn_compare()	Compares two big numbers.
bn_get_be_buf()	Exports a big number to a big-endian buffer.
bn_get_le_buf()	Exports a big number to a little-endian buffer.
bn_modulo()	Calculates the remainder of *p_a* divided by *p_mod*.
bn_get_power_modulo()	Calculates *p_a* raised to the power of *p_e*, modulo *p_m*, and stores the result in *p_r*.
bn_inverse_modulo()	Calculates the modular multiplicative inverse of *p_a* with modulus *p_modulo*.
bn_modular_multiplication()	Counts *a*b* mod *modulo* using the Montgomery algorithm.

