AN3429
Application note

STM32 proprietary code protection overview

Introduction

Software providers are developing complex middleware solutions (Intellectual Propriety (IP) code), which needs to be protected.

This IP code must be available as a plug-in in the end-user applications to build the complete solution. The global protection mechanism limits access to it via a dedicated application programming interface (API), while preventing any read access.

This application note provides an overview of the mechanism used to protect proprietary code from possible read out by the end-user code, debugger tools or RAM Trojan code. This mechanism provides a full API so the IP code can be easily called by the end-user application and still be protected against direct access to the IP code itself.

The proposed solution is based on the MPU features and a special memory and peripheral management mechanism from the end-user application and the IP code.

In the STM32 proprietary code protection method, two levels of protection are used:

- Global Read Out Protection (Global ROP): IP code and end user code are protected against direct reading (by debugger tools or RAM Trojan code) through STM32 ROP.
- IP code Read Out Protection (IP ROP): IP code protected against end user code (possible Trojan code) through the MPU.

As, once the global ROP is activated, the user no longer has full control of the Flash for programming, the primary application (IP code) must also embed an IAP Layer. This IAP allows loading the end-user application without compromising the protected code area.

For more details about the complete solution, please contact your local ST sales representative.

Contents

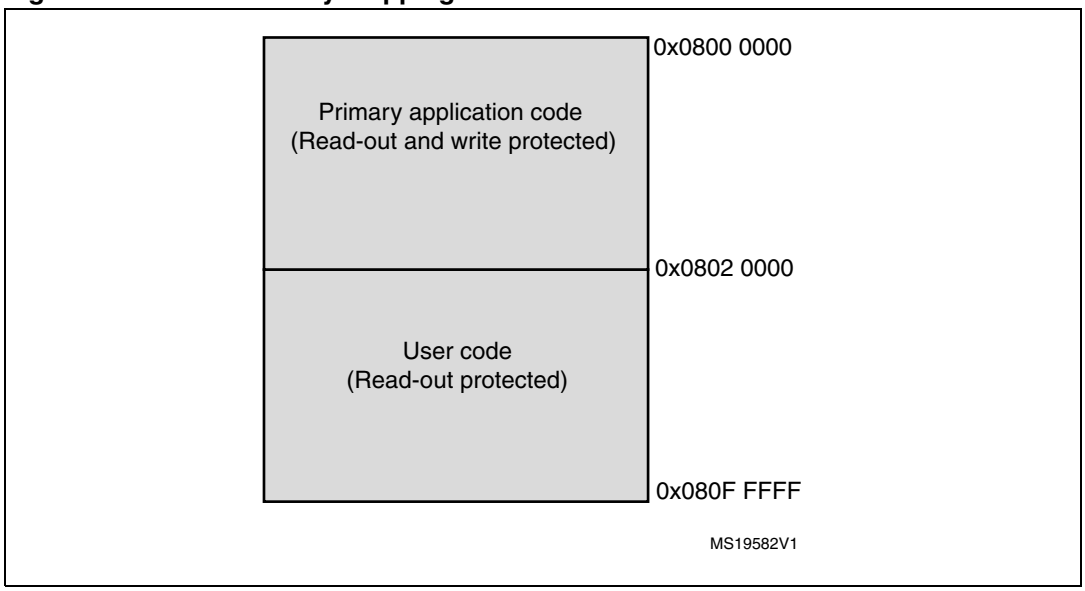
1 Code protection overview

The Flash is divided into two main areas:

- Primary application code: contains the IP memory protection code, the IP code to be protected and the IAP code that enables the loading of the end-user application.
- User code: contains the end-user application and uses the primary system application API offered by the IP memory protection code to provide access to the IP code in protected mode.

Figure 1 shows an example of Flash memory mapping (STM32F2 family)

Figure 1. Flash memory mapping



The IP Code is starts at address 0x0801 0000 defined in the linker file as follows:

```
define region IP_CODE_region = mem:[from 0x08010000 size 0x10000];
place in IP_CODE_region   { section IP_Code };
```

In this example the functions to be protected are forced-loaded in the IP code section using the following pragma with IAR:

```
#pragma location="IP_Code"
(Function definition)
```

The end user project uses the *exported_api.h* file that contains the IP code APIs of the function used by the end user application.

2 Revision history

Table 1. Document revision history

Date	Revision	Changes
19-Jul-2011	1	Initial release.