IO switching example with UID advanced protection

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	Example code for determining whether IO is switched by checking a custom UID
BSP Version	M051 Series BSP CMSIS v3.01.001
Hardware	NuTiny–EVB–M051_V3.0

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1 Function Description

1.1 Introduction

The Nuvoton M051 series microprocessor provides a UID (Unique Identifier). The developer can determine whether the application continues to execute by identifying the correctness of the UID and ensuring that the product is not copied. Simple UID recognition security seems to be insufficient, so it will provide an example of custom recognition complexity. Developers can perform custom operations based on the UID of the microprocessor and use the calculated UID to identify better security.

1.2 Principle

In order to facilitate the development, the wafer UID can be directly read and calculated, and then the application is merged and then written into the wafer. This example is divided into three blocks. Please refer to Figure 1-1. The following sections will explain each block:

1. NuLink.exe: A simple tool provided by Nuvoton, which must be used with NuLink ICE, where NuLink.exe is used for UID reading and burning into modified applications.

2. TCDEF.exe: Compiled by Visual C++, provides a UID that is read back by NuLink after a custom operation, and merges with the application bin file generated by Keil.

3. APROM.bin: The developer application project must contain the UID area after the operation is reserved, and how to identify the UID judgment program after the operation.

In order to facilitate the operation, regarding the above process, the batch file will be used to integrate the process. If the developer has completed the UID custom operation and how to identify the UID custom operation program, the batch file can be directly operated.

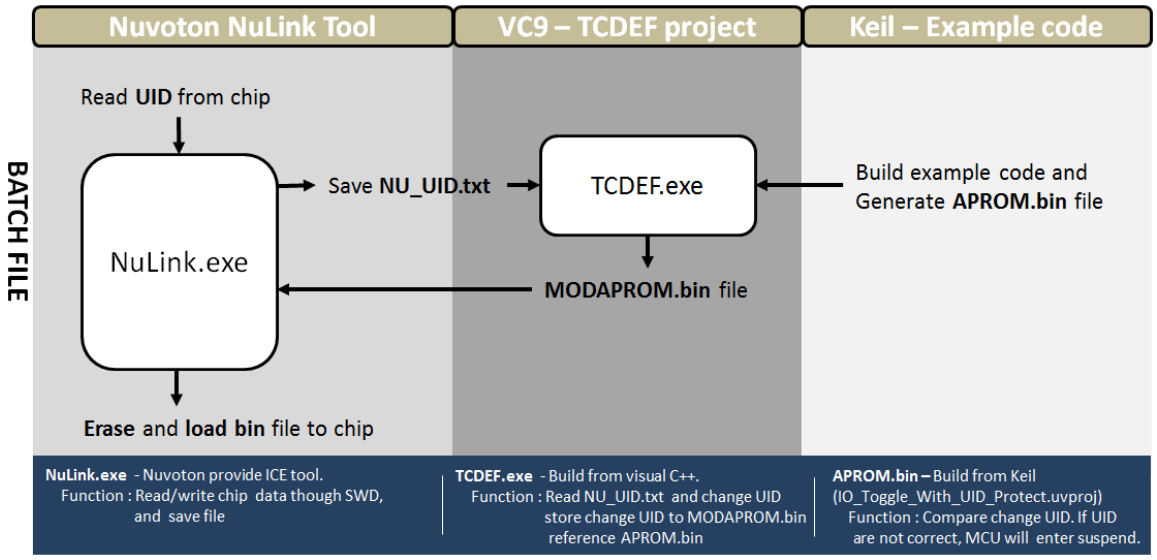

Figure 1-1 Introduction of the example flow

The UID read from the chip is used to perform the custom operation. Then it will be stored in the APROM reserved position. The APROM reserved position is the same as the definition of TCDEF.exe in the example code, as shown in Figure 1-2.

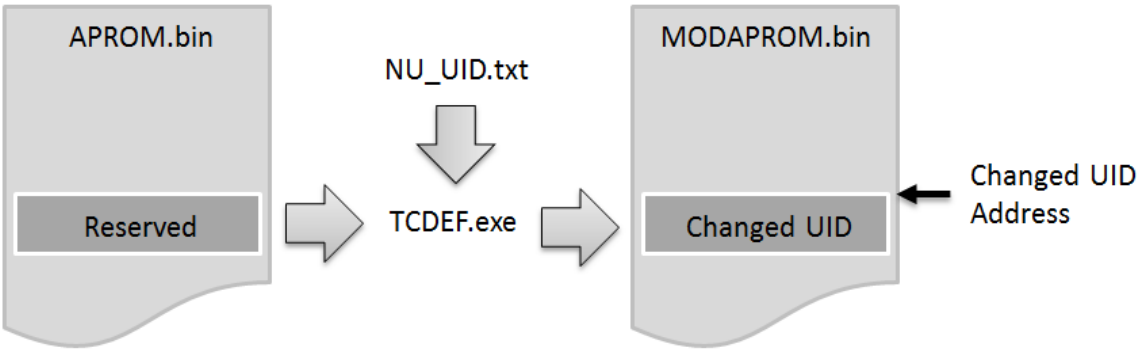

Figure 1-2 Changed UID storage address after custom operation

This example customizes the partial reservation of the operation, provides the entire process method, and finally compares the wafer UID and the UID stored in the APROM.

1.3 Demo Result

1. Execute IO_Toggle_With_UID_Protect.bat and display the command line window as shown in Figure 1-3 below. After completing the burn-in, press any key to end the command line window, and the chip will re-execute the custom UID application.

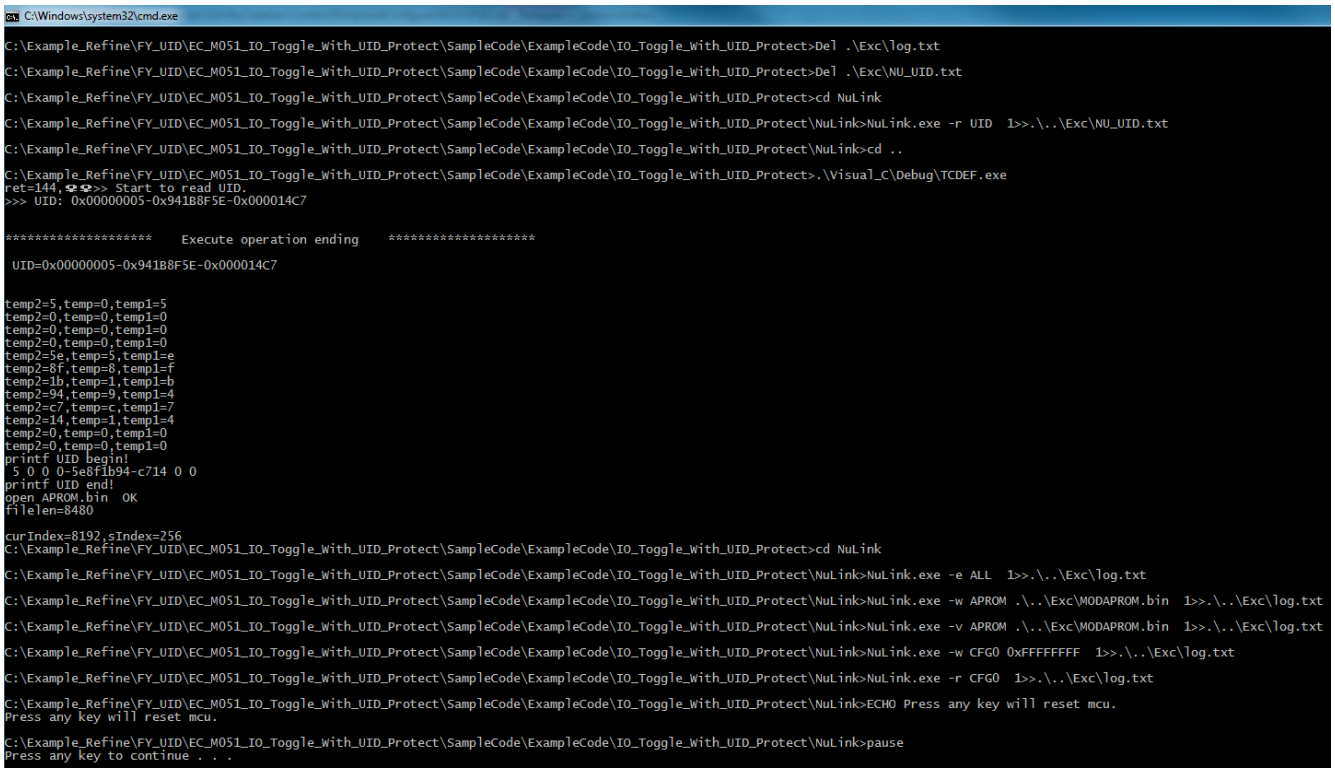

Figure 1-3 Batch file execution result

2. The comparison wafer UID is the same as the UID after the custom operation, and the wafer will continue to be toggled by PB13.

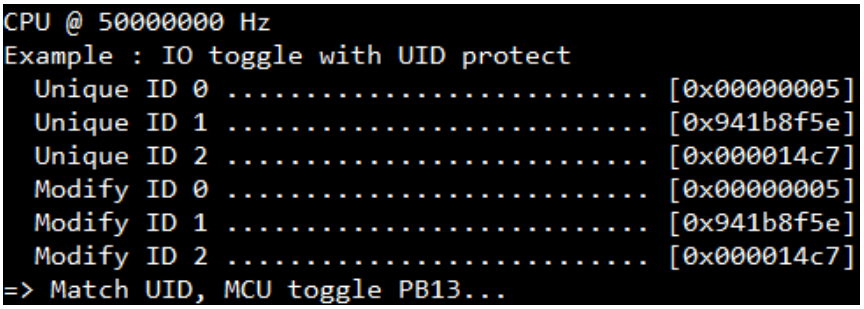

Figure 1-4 Example output by UART output

2 Code Description

2.1 Batch file - IO_Toggle_With_UID_Protect.bat

Delete log.txt and NU_UID.txt file

```
Del .\Exc\log.txt
Del .\Exc\NU_UID.txt
```

Read chip UID and save to NU_UID.txt file

```
NuLink.exe -r UID >> .\..\Exc\NU_UID.txt
```

Execute TCDEF.exe.

```
.\Visual_C\Debug\TCDEF.exe
```

Chip erase.

```
NuLink.exe -e ALL >> .\..\Exc\log.txt
```

Programming MODAPROM.bin and verification.

```
NuLink.exe -w APROM .\..\Exc\MODAPROM.bin >> .\..\Exc\log.txt
NuLink.exe -v APROM .\..\Exc\MODAPROM.bin >> .\..\Exc\log.txt
```

Write Config0 and save to log.txt

```
NuLink.exe -w CFG0 0xFFFFFFFF >> .\..\Exc\log.txt
NuLink.exe -r CFG0 >> .\..\Exc\log.txt
```

Chip reset.

```
NuLink.exe -reset
```

2.2 TCDEF.exe

Start address and offset of changed UID stored.

```
#define APROMUIDADD                 0x2000  //Store UID change data start address of APROM
#define UIDOFFSET                   0x100   //Store UID change data offset of APROM
```

Custom operation UID function.

```
void changUIDData()
{
    int i = 0;
    for (i = 0; i < 12; i++);
    /************* User change UID implement ***************/
}
```

2.3 KEIL – Example code

Custom UID storage location retention and definition must be the same as TCDEF.exe definition.

```
#define UID_START_ADDR              0x00002000   // UID start address of APROM
#define UID_ADDR_OFFSET             0x100        // UID address offset of APROM

__attribute__((at(UID_START_ADDR + UID_ADDR_OFFSET))) static const uint32_t
gau32UIDAddr[4]= {2,2,2};
```

Determine whether the UID of the chip is the same as the UID after the custom operation. If there is a custom operation, you need to add how to interpret the custom UID.

```
/************* User change UID compare implement ***************/
/* Compare original UID and change UID */
if((au32mcuUID[0] == au32modfiyUID[0]) && (au32mcuUID[1] == au32modfiyUID[1]) &&
    (au32mcuUID[2] == au32modfiyUID[2]))
    {
        printf("=> Match UID, MCU toggle PB13...\n");
        while(1)
        {
            Delay_count(500);
            P35 = 0;
            Delay_count(500);
            P35 = 1;
        }
    }
```

3 Software and Hardware Environment

Software Environment

- BSP version
  - M051 Series BSP CMSIS v3.01.001
- IDE version
  - Keil uVersion 5.24
- Visual C++ version
  - Microsoft Visual C++ 2008
- NuLink.exe version
  - Nuvoton NuLink Command Tool v2.06.6871

Hardware Environment

- Circuit components
  - NuTiny–EVB–M051_V3.0

4 Directory Information

EC_M051_IO_Toggle_With_UID_Protect	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	Source file of example code
IO_Toggle_With_UID_Protect	
KEIL	Example files of KEIL project.
Exc	Temporary and log files.
NuLink	Nuvoton NuLink tools.
Visual_C	Visual C++ project for UID changed.

5 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click IO_Toggle_With_UID_Protect.uvproj.
2. Enter Keil compile mode
   a. Build
3. Execute IO_Toggle_With_UID_Protect.bat
   a. Checking that the MCU and NuLink are properly connected to the PC.
   b. UID custom operate and program application.
   c. Chip reset to execute application.

6 Revision History

Date	Revision	Description
Jul. 16, 2019	1.00	1. Initially issued.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.