AN12533

Add Criterion for Valid User Code into HEX file

Rev. 0 — 07/09/2019

Application Note

1 Introduction

When developing LPC800 parts in MDK, and LPC8N04 part in MCUXpresso, customer may face a problem. After compilation, the HEX file may not contain the Criterion for Valid User Code, It may result in the user code burned into chips cannot be executed.

The Criterion for Valid User Code is used to judge whether the user code is valid or invalid. The reserved CPU exception vector location 7 (offset 0x0000 001C in the vector table) should contain the 2's complement of the checksum of table entries 0 through 6. It causes the checksum of the first 8 table entries to be 0.

During startup, the bootloader code calculates checksum for the first 8 locations in sector 0 of the flash. If the checksum is not zero; indicating valid user code is not found, the bootloader enters USART ISP mode automatically while do not execute the code from address 0x0.

If developing LPC800 parts using IAR, after compiling, the Criterion for Valid User Code will be added into HEX file automatically.

While using MDK and MCUXpresso, after compiling, the Criterion for Valid User Code will not be added into HEX file automatically, it is needed to do something manually to add the criteria.

Below methods introduce how to add Criterion for Valid User Code into HEX file automatically during compile procedure under MDK and MCUXpresso.

2 Solutions to add Criterion for Valid User Code into HEX file

2.1 Under MDK

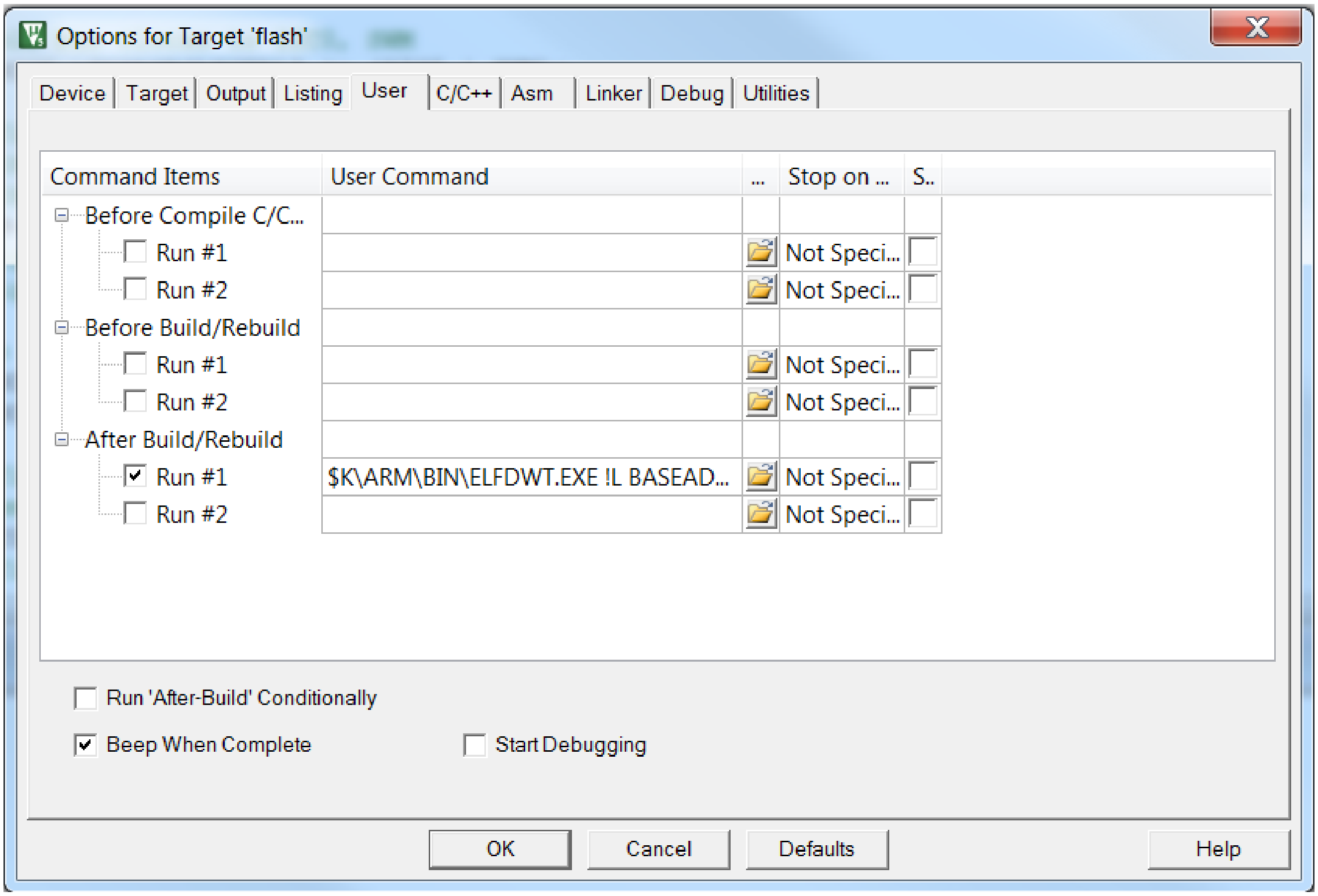
There are two solutions to add the Criterion for Valid User Code during MDK compile procedure.

2.1.1 Solution1

Enter “Options for target” setting interface, click “User” tab,

Add user command #1: $K\ARM\BIN\ELFDWT.EXE !L BASEADDRESS(0x00000000)

See Figure 1:



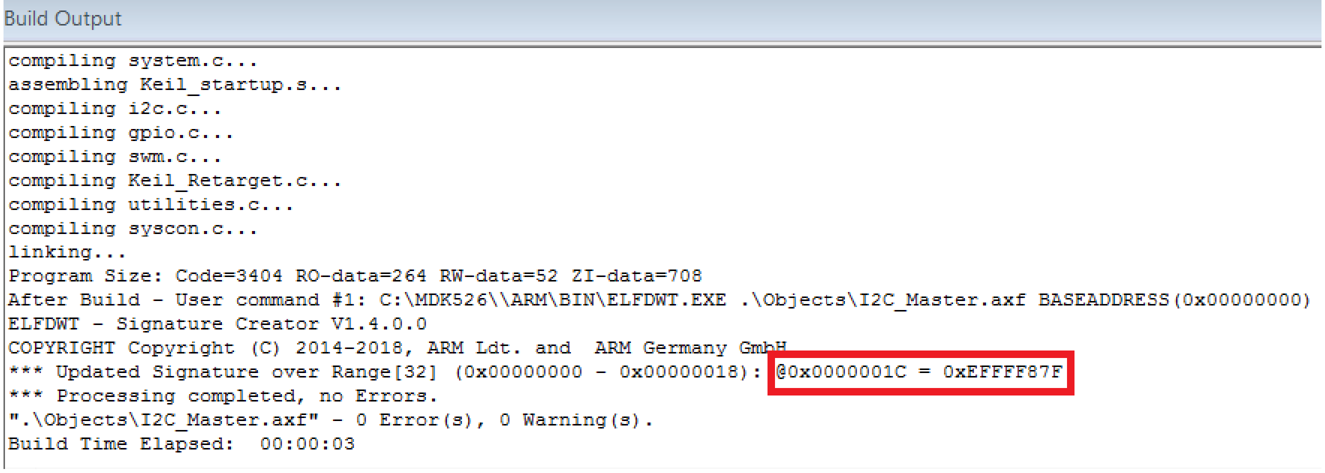
Command Items	User Command	...	Stop on ...	S..
Before Compile C/C...				
□ Run #1			Not Speci...	□
□ Run #2			Not Speci...	□
Before Build/Rebuild				
□ Run #1			Not Speci...	□
□ Run #2			Not Speci...	□
After Build/Rebuild				
☑ Run #1	$K\ARM\BIN\ELFDWT.EXE !L BASEAD...		Not Speci...	□
□ Run #2			Not Speci...	□

□ Run 'After-Build' Conditionally

☑ Beep When Complete □ Start Debugging

Figure 1. Add user command #1

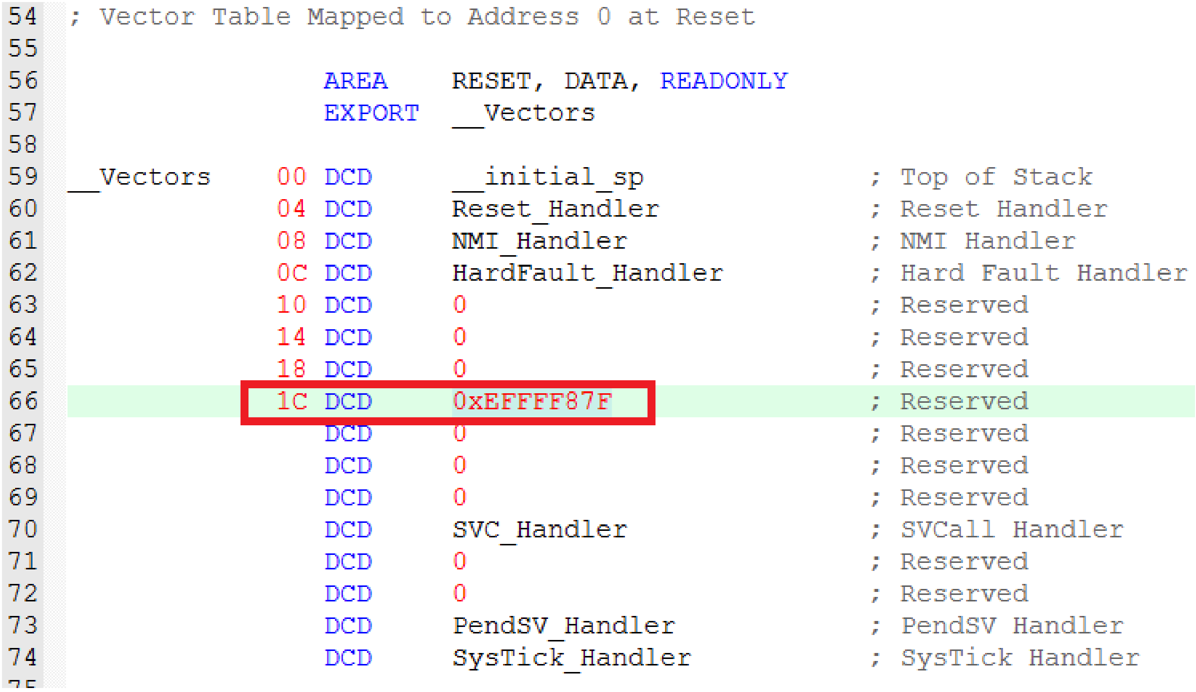
The Criterion for valid user code is shown in Figure 2:


Build Output

```
compiling system.c...
assembling Keil_startup.s...
compiling i2c.c...
compiling gpio.c...
compiling swm.c...
compiling Keil_Retarget.c...
compiling utilities.c...
compiling syscon.c...
linking...
Program Size: Code=3404 RO-data=264 RW-data=52 ZI-data=708
After Build - User command #1: C:\MDK526\\ARM\BIN\ELFDWT.EXE .\Objects\I2C_Master.axf BASEADDRESS(0x00000000)
ELFDWT - Signature Creator V1.4.0.0
COPYRIGHT Copyright (C) 2014-2018, ARM Ldt. and  ARM Germany GmbH
*** Updated Signature over Range[32] (0x00000000 - 0x00000018): @0x0000001C = 0xEFFFF87F
*** Processing completed, no Errors.
".\Objects\I2C_Master.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:03
```

Figure 2. Get the Criterion for valid user code

When get the Criterion for valid user code, add it into Keil_startup.s file, at offset 0x0000 001C in the vector table as Figure 3.



```
; Vector Table Mapped to Address 0 at Reset

                AREA    RESET, DATA, READONLY
                EXPORT  __Vectors

__Vectors    00 DCD     __initial_sp              ; Top of Stack
             04 DCD     Reset_Handler             ; Reset Handler
             08 DCD     NMI_Handler               ; NMI Handler
             0C DCD     HardFault_Handler         ; Hard Fault Handler
             10 DCD     0                         ; Reserved
             14 DCD     0                         ; Reserved
             18 DCD     0                         ; Reserved
             1C DCD     0xEFFFF87F                ; Reserved
                DCD     0                         ; Reserved
                DCD     0                         ; Reserved
                DCD     0                         ; Reserved
                DCD     SVC_Handler               ; SVCall Handler
                DCD     0                         ; Reserved
                DCD     0                         ; Reserved
                DCD     PendSV_Handler            ; PendSV Handler
                DCD     SysTick_Handler           ; SysTick Handler
```

Figure 3. Add the Criterion for valid user code into Keil_startup.s file

2.1.2 Solution2

Another solution to put the Criterion for Valid User Code into HEX file is running user command totally.

First, add user command #1:

$K\ARM\BIN\ELFDWT.EXE !L BASEADDRESS(0x00000000)

Second, add user command #2:

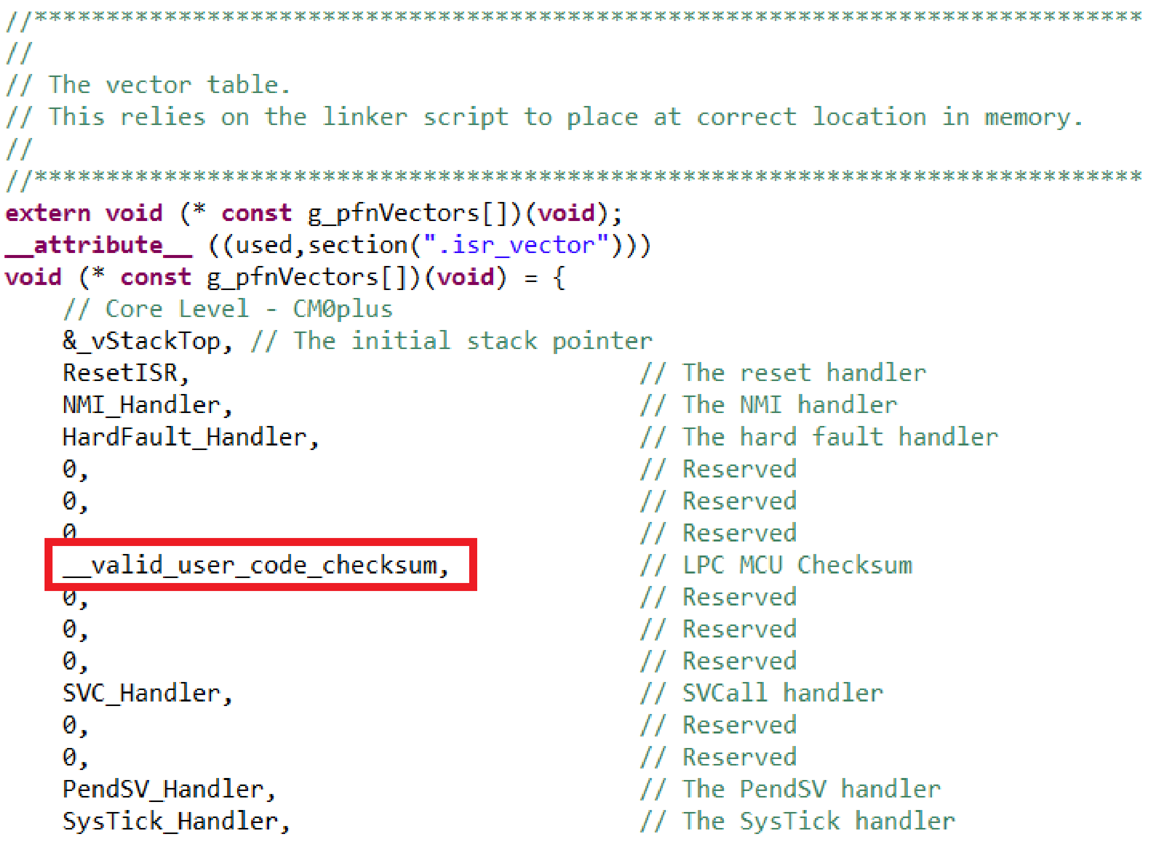
$K\ARM\ARMCC\BIN\fromelf.exe --i32 --output=xxx.hex .\Objects\xxx.axf

> **NOTE**
>
> Solution2 works for NXP LPC800 Code Bundle MDK. For NXP LPC800 SDK MDK, SDK uses different compiling config, the solution2 cannot be used, only the solution1 can work.

2.2 Under MCUXpresso

MCUXpresso IDE provides an API, named: __valid_user_code_checksum

User is required to add this API into Keil_startup.s file, at offset 0x0000 001C in the vector table. Then, the Criterion for Valid User Code is added into HEX file after compilation.



```
//*****************************************************************************
//
// The vector table.
// This relies on the linker script to place at correct location in memory.
//
//*****************************************************************************
extern void (* const g_pfnVectors[])(void);
__attribute__ ((used,section(".isr_vector")))
void (* const g_pfnVectors[])(void) = {
    // Core Level - CM0plus
    &_vStackTop, // The initial stack pointer
    ResetISR,                               // The reset handler
    NMI_Handler,                            // The NMI handler
    HardFault_Handler,                      // The hard fault handler
    0,                                      // Reserved
    0,                                      // Reserved
    0                                       // Reserved
    __valid_user_code_checksum,             // LPC MCU Checksum
    0,                                      // Reserved
    0,                                      // Reserved
    0,                                      // Reserved
    SVC_Handler,                            // SVCall handler
    0,                                      // Reserved
    0,                                      // Reserved
    PendSV_Handler,                         // The PendSV handler
    SysTick_Handler,                        // The SysTick handler
```

Figure 4. Add the Criterion for valid user code into Keil_startup.s file

For those LPC800 parts with available SDK code, the API is already added at 0x1C.

For LPC8N04, without available SDK code, maybe user uses different version code to do development. And for LPC8N04, the chip has different silicon versions with different bootloader, special attention is required to the vector checksum verification.

There are two bootloader versions (version 0.12 and version 0.14) available for the LPC8N04. The difference between the two Boot ROM versions is that version 0.14 supports application vector checksum verification before booting the image while version 0.12 does not support.

If the code for bootloader version 0.12 is used directly for bootloader version 0.14 chips, the vector checksum verification missing issue maybe happened. User need check whether the API exists on 0x1C in the vector table. If not, need add the API manually to 0x1C. The code for bootloader version 0.14 do not have this issue.

3 Revision history

Table 1. Revision history

Revision history		
Rev	**Date**	**Description**
0	20190709	Initial release

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 07/09/2019
Document identifier: AN12533