

**GigaDevice Semiconductor Inc.**

**Introduction of library invocation scheme  
based on MDK implementation**

**Application Note  
AN075**

# Table of Contents

<b>Table of Contents .....</b>	<b>2</b>
<b>List of Figures .....</b>	<b>3</b>
<b>List of Tables .....</b>	<b>4</b>
<b>1. Introduction.....</b>	<b>5</b>
<b>2. Application scenario analysis .....</b>	<b>6</b>
<b>2.1. Application scenario .....</b>	<b>6</b>
<b>3. The method implemented by the demo .....</b>	<b>6</b>
<b>3.1. Overview .....</b>	<b>6</b>
<b>3.2. MPEG-2 software algorithm project creation.....</b>	<b>7</b>
<b>3.3. Separate CRC calculation code.....</b>	<b>11</b>
<b>3.4. Calling the CRC algorithm function in a new project.....</b>	<b>13</b>
<b>4. Revision history.....</b>	<b>16</b>

# List of Figures

Figure 3-1. modify of the Template.....	7
Figure 3-2. The content of the Mpeg2 directory.....	7
Figure 3-3. CRC test code calculation result .....	10
Figure 3-4. Third-party software MPEG-2 CRC calculation result .....	11
Figure 3-5. Linker set.....	12
Figure 3-6. the map file .....	13
Figure 3-7. crc library file .....	14
Figure 3-8. directory structure of the new .....	14
Figure 3-9. basic framework for cooperative development.....	15

## List of Tables

<b>Table 3-1. crc.h code cell.....</b>	<b>8</b>
<b>Table 3-2. crc32_mpeg2_init code cell.....</b>	<b>8</b>
<b>Table 3-3. crc32_mpeg2_calc code cell .....</b>	<b>9</b>
<b>Table 3-4. CRC test code cell .....</b>	<b>9</b>
<b>Table 3-5. scatter-loading file code cell .....</b>	<b>13</b>
<b>Table 4-1. Revision history.....</b>	<b>16</b>

## 1. Introduction

This application note is dedicated to software code design, which aims to achieve collaborative development and reduce code downloads by decentralizing image files.

This application note will describe the method of dispersing an image by implementing a software algorithm to calculate CRC MPEG-2, of course, the implementation method is not limited to the method described in this application note, and customers can also use other methods to achieve the same purpose.

This application note is theoretically applicable to all GD32 MCU products developed using MDK.

## 2. Application scenario analysis

### 2.1. Application scenario

In the process of product application development, the following scenarios are often encountered:

Scenario 1, in the development of relatively large application code, it is often found that most of the code functions are normal, and a small part of the code needs to be adjusted continuously. When adjusting this part of the code, the entire application code will be recompiled and downloaded. When the entire application is relatively large (for example, the bin file is larger than 1MB), the download time of the entire code will become very long, often exceeding 3 minutes. At this time, if only the modified part of the code is updated without modifying other codes, It can greatly improve the development efficiency and improve the development experience.

Scenario 2, some algorithm developers will solidify their developed algorithms to a fixed position on the chip, and solution providers will develop their own applications based on the solidified algorithms.

This application note describes a collaboratively developed approach to a software-based CRC algorithm.

## 3. The method implemented by the demo

### 3.1. Overview

CRC (Cyclic Redundancy Check): It is the most commonly used error checking code in the field of data communication. It is characterized in that the length of the information field and the check field can be arbitrarily selected. Cyclic redundancy check is a data transmission error detection function, which performs polynomial calculation on data and attaches the obtained result to the back of the frame. The receiving device also performs a similar algorithm to ensure the correctness and integrity of data transmission.

This section will implement the algorithm for calculating CRC MPEG-2 by software, and separate the execution file of the algorithm from the execution file of the main calling code to realize cooperative development. Since this article mainly describes the method of realizing separation and development, it will not introduce the implementation of CRC algorithm in detail.

### 3.2. MPEG-2 software algorithm project creation

The software project used in this section will be based on the firmware library project of GD32W51x, and will be implemented based on GD32W515P-EVAL-V1.1.

First, obtain the firmware library project of GD32W51x from the official website. The firmware library project used in this article is version V1.0.0. After the acquisition is successful, you need to confirm whether the software and hardware functions are normal.

Create a new directory and source code file, as shown in [\*\*Figure 3-1. modify of the Template\*\*](#)  
Create the mpeg2 directory under the Template directory to save the CRC code file, [\*\*Figure 3-2. The content of the Mpeg2 directory\*\*](#) shows the contents of the folder, There are only 2 files in this folder, crc.c and crc.h.

**Figure 3-1. modify of the Template**

GD32W51x_Firmware_Library > Template >			
名称	修改日期	类型	大小
IAR_Project	2022/7/4 10:29	文件夹	
Keil_Project	2022/7/4 10:29	文件夹	
<b>mpeg2</b>	2022/7/4 10:29	文件夹	
gd32w51x_it.c	2022/6/30 14:45	C 文件	4 KB
gd32w51x_it.h	2022/6/30 14:45	C/C++ Header	3 KB
gd32w51x_libopt.h	2022/6/30 14:45	C/C++ Header	3 KB
main.c	2022/6/30 16:12	C 文件	4 KB
main.h	2022/6/30 14:45	C/C++ Header	2 KB
readme.txt	2022/6/30 14:45	文本文档	2 KB
systick.c	2022/6/30 14:45	C 文件	3 KB
systick.h	2022/6/30 14:45	C/C++ Header	2 KB

**Figure 3-2. The content of the Mpeg2 directory**

GD32W51x_Firmware_Library > Template > mpeg2			
名称	修改日期	类型	大小
crc.c	2022/6/30 16:13	C 文件	2 KB
crc.h	2022/6/30 16:13	C/C++ Header	1 KB

[\*\*Table 3-1. crc.h code cell\*\*](#) is the code in crc.h, there are two function declarations in crc.h, these two functions are implemented in crc.c for external calls.

**Table 3-1. crc.h code cell**

```
#ifndef _CRC_H
#define _CRC_H

void crc32_mpeg2_init(unsigned int poly, unsigned int *crc_table);
unsigned int crc32_mpeg2_calc(unsigned int crc, unsigned int *crc_table, void* input, int len);

#endif
```

The function `crc32_mpeg2_init` is used to initialize the CRC calculation table, [Table 3-2.](#) [`crc32\_mpeg2\_init` code cell](#) shows its implementation.

**Table 3-2. crc32\_mpeg2\_init code cell**

```
/*
 \brief      generate crc MPEG-2 table
 \param[in]  poly: crc polynomial, for MPEG-2 it is 0x4C11DB7
 \param[in]  crc_table: a pointer to a polynomial table, at least size is 256
 \param[out] crc_table: a pointer to generated polynomial table
 \retval     none
 */

void crc32_mpeg2_init(unsigned int poly, unsigned int *crc_table)
{
    int i;
    int j;
    unsigned long c;

    for (i = 0; i < 256; i++) {
        c = i;
        c = i << 24;

        for (j = 0; j<8; j++){
            if (c & 0x80000000){
                c = (c << 1) ^ poly;
            } else {
                c = ((c << 1));
            }
        }
        crc_table[i] = c;
    }
}
```

The function `crc32_mpeg2_calc` is used for CRC calculation, [Table 3-3. `crc32\_mpeg2\_calc`](#)

code cell shows its implementation.

**Table 3-3. crc32\_mpeg2\_calc code cell**

```

/*!
 \brief      calculate crc MPEG-2
 \param[in]  crc: initial value, for MPEG-2 it is 0xFFFFFFFF
 \param[in]  crc_table: a pointer to a polynomial table, at least size is 256
 \param[in]  input: a pointer to need calculate data
 \param[in]  len: length of need calculate data
 \param[out] none
 \retval     calculate result
 */

unsigned int crc32_mpeg2_calc(unsigned int crc, unsigned int *crc_table, void* input, int len)
{
    int i;
    unsigned char index;
    unsigned char* pch;
    pch = (unsigned char*)input;
    for (i = 0; i < len; i++) {
        index = (unsigned char)((crc >> 24) ^ *pch);
        crc = (crc << 8) ^ crc_table[index];
        pch++;
    }
    return crc;
}

```

Add CRC test code cell in main.c, and call this function in the main function, the test function will calculate its MPEG-2 CRC for the input data "123456789" check value, [Table 3-4. CRC test code cell](#) shows the test code cell.

**Table 3-4. CRC test code cell**

```

unsigned int crc_table[256] = {0};

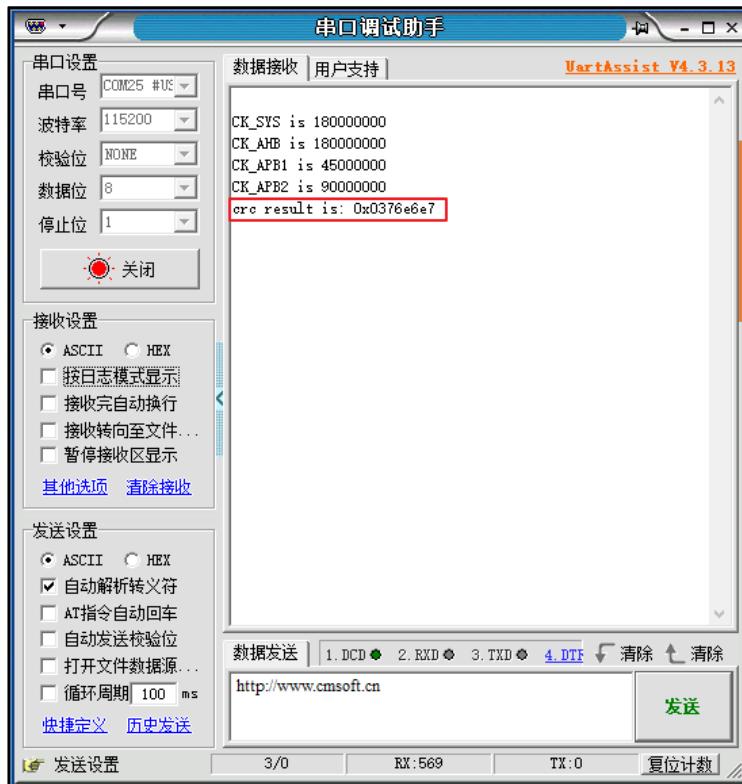
void crc_test(void)
{
    unsigned int crc;
    crc32_mpeg2_init(0x4C11DB7, crc_table);

    crc = 0xFFFFFFFF;
    crc = crc32_mpeg2_calc(crc, crc_table, "123456789", 9);
    printf("\r\nCRC result is: 0x%08x\r\n", crc);
}

```

**Figure 3-3. CRC test code calculation result**, shows the calculation results in the serial port software.

**Figure 3-3. CRC test code calculation result**



The calculation results of the third-party software are shown in [\*\*Figure 3-4. Third-party software MPEG-2 CRC calculation result\*\*](#), which are consistent with the calculation results of the program we designed, so we can know that the designed software MPEG-2 CRC calculation algorithm is correct.

**Figure 3-4. Third-party software MPEG-2 CRC calculation result**

**CRC ( 循环冗余校验 ) 在线计算**

○Hex     Ascii    **校验文件**

123456789

需要校验的数据 :

输入的数据为ascii字符串 , 例如 : 1234

参数模型 NAME : CRC-32/MPEG-2     $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+$

宽度 WIDTH : 32

多项式 POLY ( Hex ) : 04C11DB7    例如 : 3D65

初始值 INIT ( Hex ) : FFFFFFFF    例如 : FFFF

结果异或值 XOROUT ( Hex ) : 00000000    例如 : 0000

输入数据反转 ( REFIN )     输出数据反转 ( REFOUT )

**计算**    **清空**

校验计算结果 ( Hex ) : 0376E6E7    **复制**

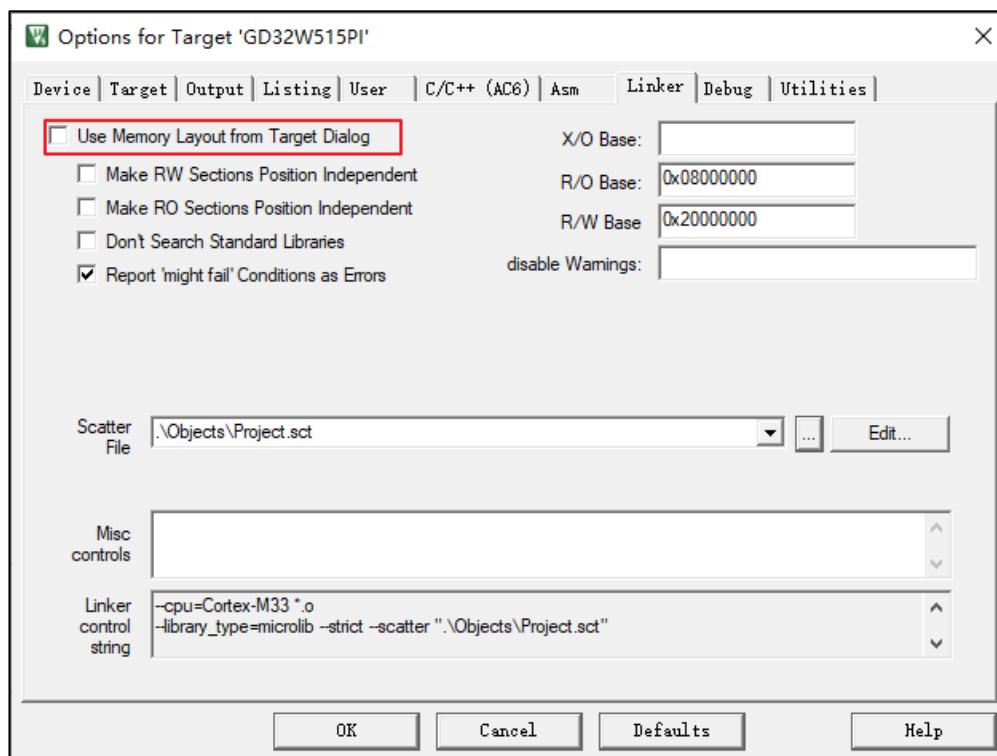
高位在左低位在右 , 使用时请注意高低位顺序 !!!

校验计算结果 ( Bin ) : 00000011011101101110011011100111    **复制**

### 3.3. Separate CRC calculation code

The content of this section will be based on the content of the previous section, scatter and load the code of MPEG-2 CRC calculation to the specified location, and other projects can directly call the code of the specified location to realize the CRC calculation.

Open the project's options page, remove the default option under Linker, and use the scatter-loading file to allocate the project's memory, [\*\*Figure 3-5. Linker set\*\*](#) shows the linker set.

**Figure 3-5. Linker set**


Make the following changes in the default scatter-loading file, change the size of the first load area LR\_IROM1 to 0x0000FFFF, change the size of the first execution area ER\_IROM1 to 0x0000FFFF, and create a new load area for LR\_CODE\_2, whose starting address is next to LR\_IROM1, and the size is also 0x0000FFFF, defined an execution area CRC\_TEST, which start address is consistent with LR\_CODE\_2, and all the codes of crc.o are placed in this area.

[Table 3-5. scatter-loading file code cell](#) shows those changes.

**Table 3-5. scatter-loading file code cell**

```

; ****
; *** Scatter-Loading Description File generated by uVision ***
; *****

LR_IROM1 0x08000000 0x0000FFFF { ; load region size_region
    ER_IROM1 0x08000000 0x0000FFFF { ; load address = execution address
        *.o (RESET, +First)
        *(InRoot$$Sections)
        .ANY (+RO)
        .ANY (+XO)
    }
    RW_IRAM1 0x20000000 0x00070000 { ; RW data
        .ANY (+RW +ZI)
    }
}

LR_CODE_2 0x08010000 0xFFFF {
    CRC_TEST 0x08010000 {
        crc.o
    }
}

```

Compile the project again. The map file after the project is compiled, shows that the two functions `crc32_mpeg2_calc` and `crc32_mpeg2_init` defined in `crc.o` are located at `0x08010000` and `0x08010054`, consistent with what we defined in the scatter-loading file, [Figure 3-6. the map file](#) shows the address location of `crc32_mpeg2_calc` and `crc32_mpeg2_init`. After downloading the program to the development board, you can see that the printed result is the same as the result of the unmodified project. At this time, the location where the chip `0x08010000` starts is stored with executable code that can perform mpeg-2 calculations. If it is not erased, the code will always be stored here, and other programs can call this function.

**Figure 3-6. the map file**

Project.map								
Exec Addr	Load Addr	Size	Type	Attr	Idx	E	Section Name	Object
0x08010000	0x08010000	0x00000054	Code	RO	2418		.text.crc32_mpeg2_calc	crc.o
0x08010054	0x08010054	0x0000006e	Code	RO	2416		.text.crc32_mpeg2_init	crc.o

### 3.4.

### Calling the CRC algorithm function in a new project

Now we can create other projects and implement the CRC calculation by calling the code at

the beginning of 0x08010000, but for the convenience of calling we will create a library file to participate in the compilation of the new project, instead of calling directly through the address, the content of this section will be focus on describing the method of implementation.

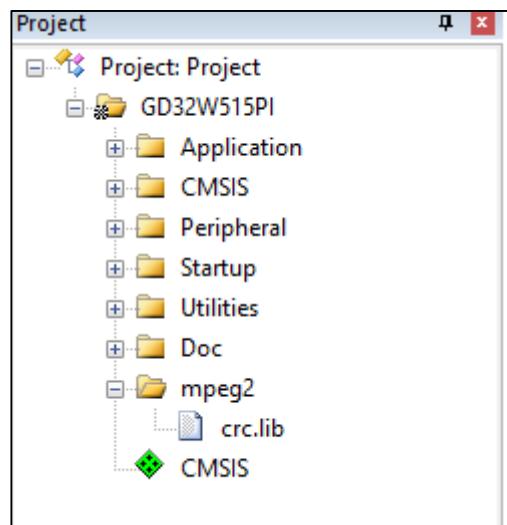
First, reopen a firmware library project, create an mpeg2 folder in the same way, copy the crc.h file from the previous summary to this directory in the project, and create a crc.lib library file, add content which show in [\*\*Figure 3-7. crc library file\*\*](#).

**Figure 3-7. crc library file**

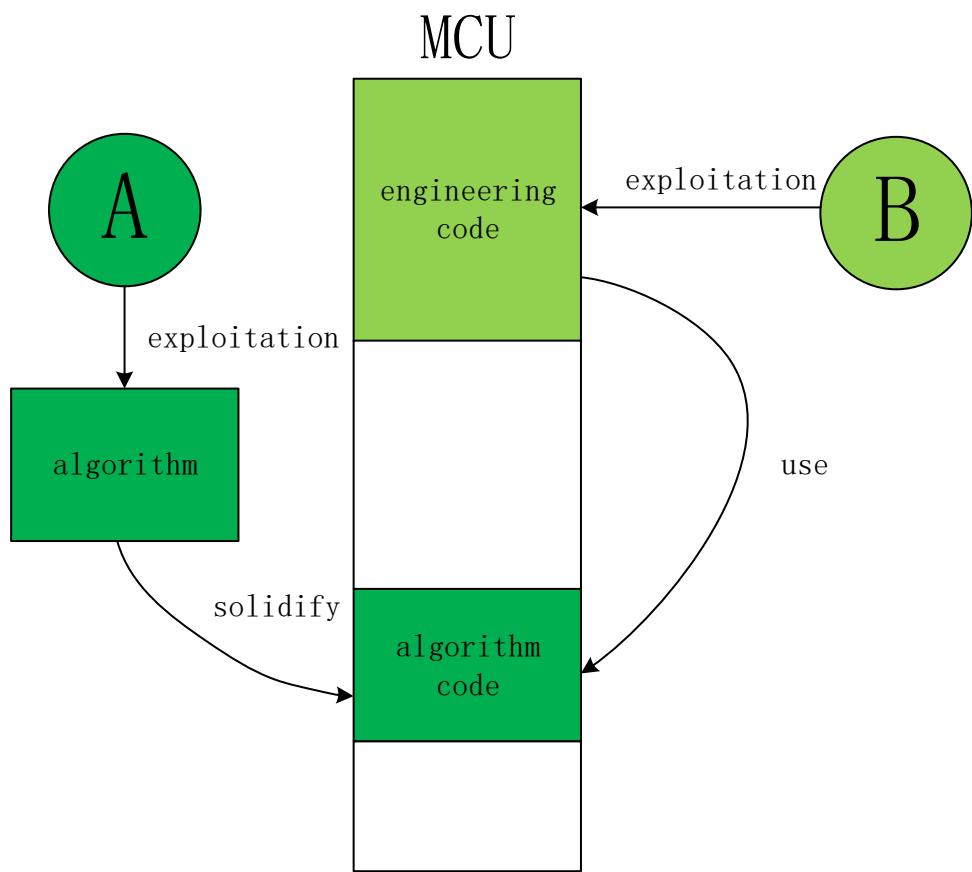
```
0x08010000 T crc32_mpeg2_calc
0x08010054 T crc32_mpeg2_init
```

Add the library to the project, [\*\*Figure 3-8. directory structure of the new\*\*](#) shows directory structure of the new project, add crc\_test test function in main.c, [\*\*Table 3-4. CRC test code cell\*\*](#) shows the test function, compile and download to the development board, will find that the crc value calculated by the serial port output is still correct.

**Figure 3-8. directory structure of the new**



If we regard the code development of software crc calculation as the function code or algorithm code developed by A, A can download and solidify it into the chip after completing, B can use the code to develop his own functions. This constitutes the most basic framework for cooperative development like [\*\*Figure 3-9. basic framework for cooperative development\*\*](#).

**Figure 3-9. basic framework for cooperative development**


**Note:** There are many ways to solidify the code into the chip. For example, you can use the image tool to generate the hex file from the algorithm and solidify it into the chip from B, which is not limited to the method described above.

## 4. Revision history

**Table 4-1. Revision history**

Revision No.	Description	Date
1.0	Initial Release	Jul.18 2022

## Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company under the intellectual property laws and treaties of the People's Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it's suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it's suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.