IAR Systems (/) > Support (/support/) > Tech Notes (/support/tech-notes/) > ..

Calculate CRC32 as in STM32 hardware

Technical Note 64424

Targets:	ARM
Component:	General
Updated:	14.05.2013 09:10

Introduction

STM32 devices from STMicroelectronics have a built-in hardware CRC32 calculator. (This means that using CRC32 in an application can be done without using code space for the algorithm).

Discussion

The CRC32 algorithm used in STM32 devices can be reproduced by the IELFTOOL utility in IAR Embedded Workbench for ARM version 5.50 and newer.

Solution for IAR Embedded Workbench for ARM version 6.40 and later

Options for 6.40 and later

Configure options in Project > Options > Linker > Checksum to enable IELFTOOL to make CRC32 calculation in the same way as the hardware in the STM32 devices.

The options should be set as follows:

```
Size: 4 bytes
 Alignment: 4
 Algorithm: CRC32(0x4C11DB7)
 Complement: As is
 Bit order: MSB first
 [ ]Reverse byte order within word [unchecked]
 Initial Value: 0xFFFFFFFF
 [ ]Use as input [unchecked]
 Checksum unit size: 32-bit
```

Example for STM32F10x

This example shows source code, the linker configuration file and the corresponding settings in the linker options dialog.

- Source code (snippet)

RELATED TECH NOTES

Checksum calculation with IAR ILINK Li...
Technical Note 11927

(/support/tech-notes/general/checksum-calculation-with-ielftool-after-linking-with-ilink/)

Debugging checksum calculations
Technical Note 35170

(/support/tech-notes/general/debugging-checksum-calculations/)

Calculate CRC with MSP430 hardware
Technical Note 41610

(/support/tech-notes/general/calculate-crc-with-msp430-hardware/)

Checksum generation
Technical Note 26457

(/support/tech-notes/general/checksum-generation/)

Calculate CRC32 as in K60 hardware
Technical Note 85753

(/support/tech-notes/general/calculate-crc32-as-in-k60-hardware/)

Incorrect checksum when debugging
Technical Note 48021

(/support/tech-notes/debugger/the-checksum-is-incorrect-when-debugging/)

Calculate CRC32 as in STM32 hardware
Technical Note 64424

(/support/tech-notes/general/calculate-crc32-as-in-stm32-hardware-v.5.50-and-later/)

IELFTOOL Checksum - Placing the chec...
Technical Note 62709

(/support/tech-notes/general/ielftool-checksum---placing-the-checksum/)

IELFTOOL Checksum - When using NXP...
Technical Note 52619

(/support/tech-notes/general/ielftool-checksum---when-using-lpc-device/)

IELFTOOL Checksum - Basic actions
Technical Note 65473

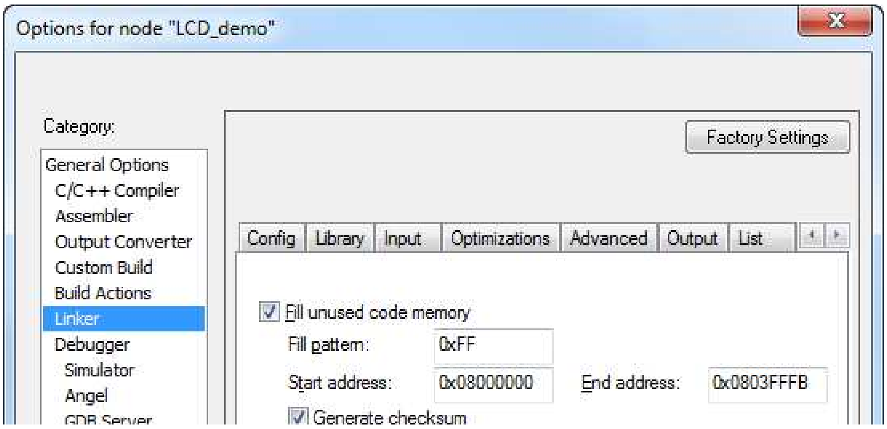
(/support/tech-notes/general/ielftool-checksum---basic-actions/)

```
#include "stm32f10x_crc.h"

extern uint32_t __checksum;

uint32_t calcCrc32(uint8_t* data, uint32_t len)
{
   uint32_t* pBuffer = (uint32_t*) data;
   uint32_t BufferLength = len/4;
   uint32_t index = 0;

   RCC_AHBPeriphClockCmd(RCC_AHBPeriph_CRC, ENABLE);

   CRC_ResetDR();

   for(index = 0; index < BufferLength; index++)
   {
   CRC->DR = pBuffer[index];
   }

   return CRC->DR;
}

void main(void)
{
  ...

  uint32_t valCrc32 =
   calcCrc32((uint8_t*)0x08000000, 0x040000 - 4);

  if (valCrc32 == __checksum)
  {
    // TBD
  }
  else
  {
    // TBD
  }
  ...
}
```

- Linker configuration file (snippet)

```
define symbol __ICFEDIT_region_ROM_start__ = 0x08000000;
define symbol __ICFEDIT_region_ROM_end__ = 0x0803FFFB;

place at address mem:0x0803FFFC { readonly section .checksum };
```

- Linker options dialog

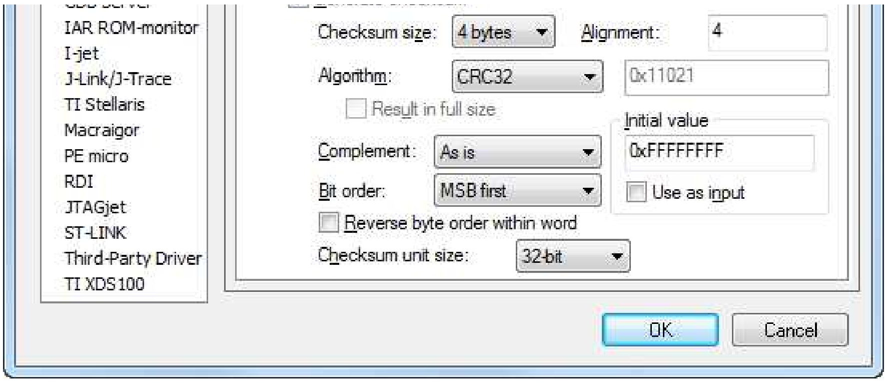


☑ Fill unused code memory

Fill pattern: 0xFF

Start address: 0x08000000 End address: 0x0803FFFB

☑ Generate checksum



Checksum size: 4 bytes
Alignment: 4
Algorithm: CRC32
0x11021
□ Result in full size
Initial value
0xFFFFFFFF
Complement: As is
Bit order: MSB first
□ Use as input
□ Reverse byte order within word
Checksum unit size: 32-bit

- Example Project:
  Example (6.40).zip (/contentassets/d8bbf6cb53c44ac39f10798d3c086d67/example-6.40.zip)

Solutions for older IAR Embedded Workbench for ARM versions

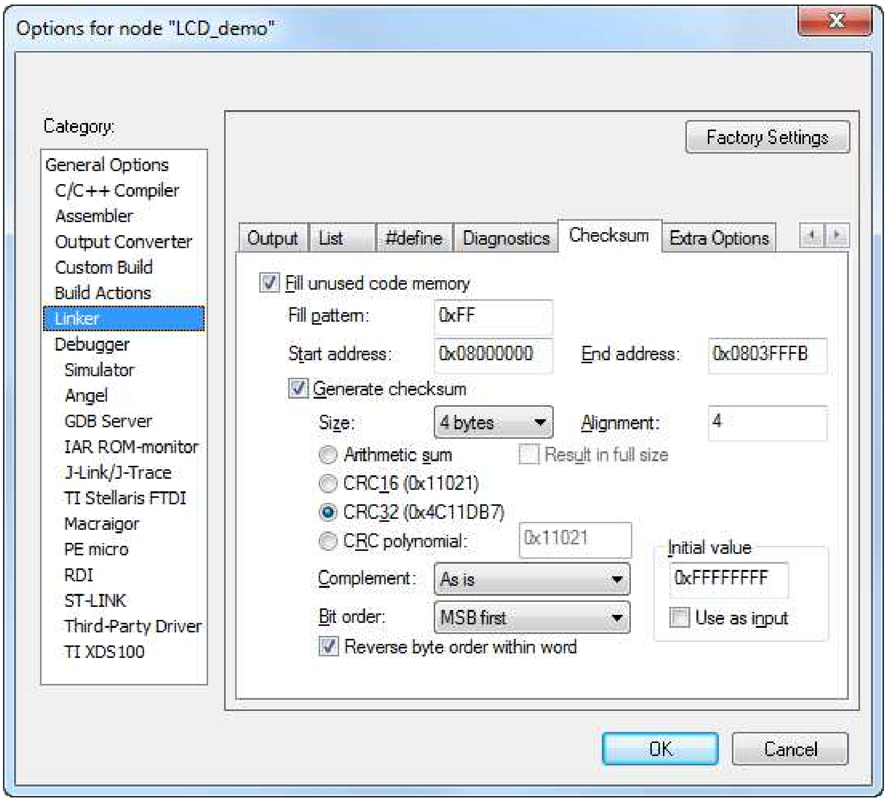
Options for 6.10 to 6.30

Configure options in Project > Options > Linker > Checksum to enable IELFTOOL to make CRC32 calculation in the same way as the hardware in the STM32 devices.

The options should be set as follows:

```
Size: 4 bytes
 Alignment: 4
 Algorithm: CRC32(0x4C11DB7)
 Complement: As is
 Bit order: MSB first
 [*]Reverse byte order within word [CHECKED]
 Initial Value: 0xFFFFFFFF
 [ ]Use as input [unchecked]
```

- Linker options dialog



Options for node "LCD_demo"

☑ Fill unused code memory
Fill pattern: 0xFF
Start address: 0x08000000
End address: 0x0803FFFB
☑ Generate checksum
Size: 4 bytes
Alignment: 4
○ Arithmetic sum
□ Result in full size
○ CRC16 (0x11021)
● CRC32 (0x4C11DB7)
○ CRC polynomial: 0x11021
Initial value
0xFFFFFFFF
Complement: As is
Bit order: MSB first
□ Use as input
☑ Reverse byte order within word

- Example Project:

Example (6.21).zip (/contentassets/d8bbf6cb53c44ac39f10798d3c086d67/example-6.21.zip)

Solution for IAR Embedded Workbench for ARM version 5.50

In version 5.50, IELFTOOL can be configured to calculate CRC32 in the same manner as the calculation of CRC32 using the STM32 hardware. This ability is not documented and it can only be invoked as a command line option.

The details that concern the STM32 CRC32 algorithm are

- 0xFFFFFFFF as initial value.
- Utilize the two new ielftool algorithm flags i and r.

The string to enter in Project > Options > Build Actions > Post-build command line is...

```
ielftool --fill 0xFF;__checksum_begin-__checksum_end
--checksum __checksum:4,crc32:ir,0xFFFFFFFF;__checksum_begin-__checks
--verbose $TARGET_PATH$ $TARGET_PATH$
```

...where these substrings are placeholders that needs to changed for the project in use:

__checksum
The name of the symbol where the checksum value should be stored. Note that it must exist in the symbol table in the input ELF file.

__checksum_begin
The first address on which the checksum should be calculated.

__checksum_end
The last address on which the checksum should be calculated.

The preceding command line should be combined with all options reset (unchecked) in Project > Options > Linker > Checksum

IAR EMBEDDED WORKBENCH (/IAR-EMBEDDED-WORKBENCH/)

Tools for ARM (/iar-embedded-workbench/tools-for-arm/)
Tools for 8051 (/iar-embedded-workbench/tools-for-8051/)
Tools for MSP430 (/iar-embedded-workbench/tools-for-msp430/)
Tools for AVR (/iar-embedded-workbench/tools-for-avr/)

SUPPORT (/SUPPORT/)

Resources (/support/resources/)
Customer Care (/support/customer-care/)
User Guides (/support/user-guides/)

Technical Support (/support/technical-support/)

INVESTORS (/INVESTORS/)

Investment Case (/investors/investment-case/)
Investor Press Archive (/investors/press-archive/)
About IAR Systems Group (/investors/about-iar-systems-group/)
Corporate Governance (/investors/corporate-governance/)

SOCIAL MEDIA

(https://www.facebook.com/iarsystems) (https://twitter.com/iarsystems)

(https://www.linkedin.com/company/iar-systems)

(https://www.youtube.com/channel/UCxxoUOKedI8_s3ZdJFuu6Yg)

Cookies (/metapages/cookies/) | Privacy Policy (/metapages/privacy-policy/)

| Trademarks (/metapages/trademarks/) | Terms of Use (/metapages/terms-of-use/)