Data I/O Checksum Calculation Methods
Application Note

(Chinese translation follows this English text.)

General Description and Scope

This document describes common checksum calculation methods for devices that we support in TaskLink.

When Would I Use 8/16/32-bit Checksum?

The 8-bit, 16-bit and 32-bit checksum methods just add up the data—they do not consider data location like the CRC checksum method does. Therefore, with the 8-bit, 16-bit and 32-bit checksum methods, whole block swaps don't affect the checksum. And with the 8-bit checksum method, even byte swaps don't affect the checksum.

On the **File I/O** tab, byte swap can be selected by checking the Odd/Even Byte Swap checkbox. It transposes odd and even order bytes within the data file, but it will only affect devices with 16-bit and 32-bit data width. The byte swap will have no effect on 8-bit devices. For 16-bit devices it will transpose odd and even order WORDs within the data file. For 32-bit devices it will transpose odd and even order DWORDs within the data file.

What Data Is the Checksum Calculated on?

TaskLink firstly translates the selected 'PC File' into absolutely binary format and then expands/shrinks it to match the sector table(s) of 'Sectors' tab. The generated data file is continuous from address 0 and all gap(s) is filled with the value specified by "Automatic RAM Fill". If there is special feature(s) to be stored into data file, TaskLink does it also. The checksum then be calculated based on this generated file.

Note: The options 'Special Data Sectors' and 'Sector Protect Information' under 'Data' tab could introduce difference.

Description of Checksum Methods

8-Bit Checksum

The 8-bit checksum method simply adds all the BYTEs in the image.bin file after handling by TaskLink, to get the checksum. The overflow (carry) is omitted.

For example, for the data shown in Figure 1 below:
8-bit checksum is 0x37C (= 0x12 + 0x 34 + . . .)

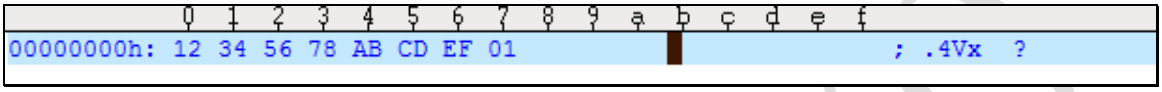

Figure 1: Sample data.

16-Bit Checksum

The 16-bit checksum method simply adds all the WORDs in the image.bin file after handling by TaskLink, to get the checksum. The overflow (carry) is omitted.

For example, for the data shown in Figure 1 above:
16-bit checksum is 0x17C02 (= 0x3412 + 0x7856 + . . .) This sample shows Little Endian byteswap.

32-Bit Checksum

The 32-bit checksum method simply adds all the DWORDs in the image.bin file after handling by TaskLink, to get the checksum. The overflow (carry) is omitted.

For example, for the data shown in Figure 1 above:
32-bit checksum is 0x7A4601BD (= 0x78563412 + 0x01EFCDAB)

32-Bit CRC Checksum

32-bit CRC checksum method uses this polynomial to calculate the checksum:
x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1.

For example, for the data shown in Figure 1 above:
32-bit CRC checksum is 0xB9196C38 (Use polynomial above for calculation.)

Selecting Checksum Methods in TaskLink

To select a checksum method in TaskLink, in the **Edit Task** dialog, click the **Data** tab. Then use the dropdown arrow to view the selections. Refer to Figure 2 below.

The method labeled Device Width Checksum autoselects the checksum method based on the device width. For example: an 8-bit device will use the 8-bit checksum method, and a 32-bit device will use the 32-bit checksum method.

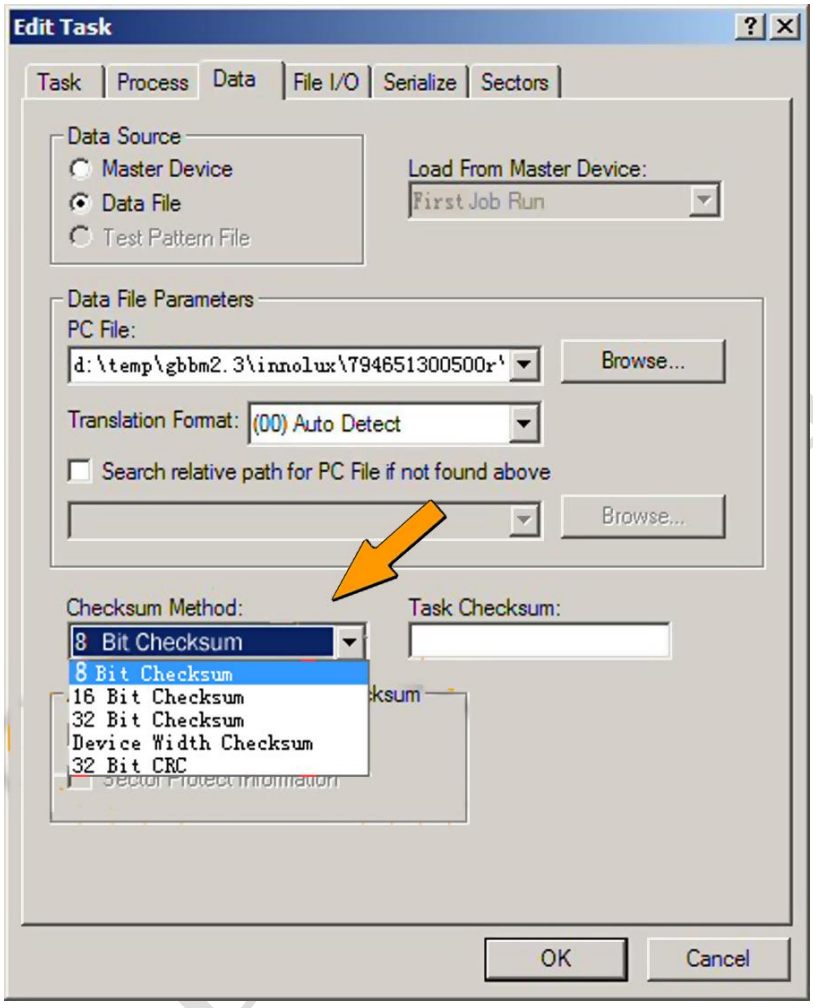

Figure 2: The *Checksum Method* dropdown expands to show the list of methods available.

Special Data and Sector Protection

If your target device has special data sectors and/or sector-protection features, this information can be included in the checksum by checking the boxes as available. Selected items will be added to the checksum. See Figure 3 below.

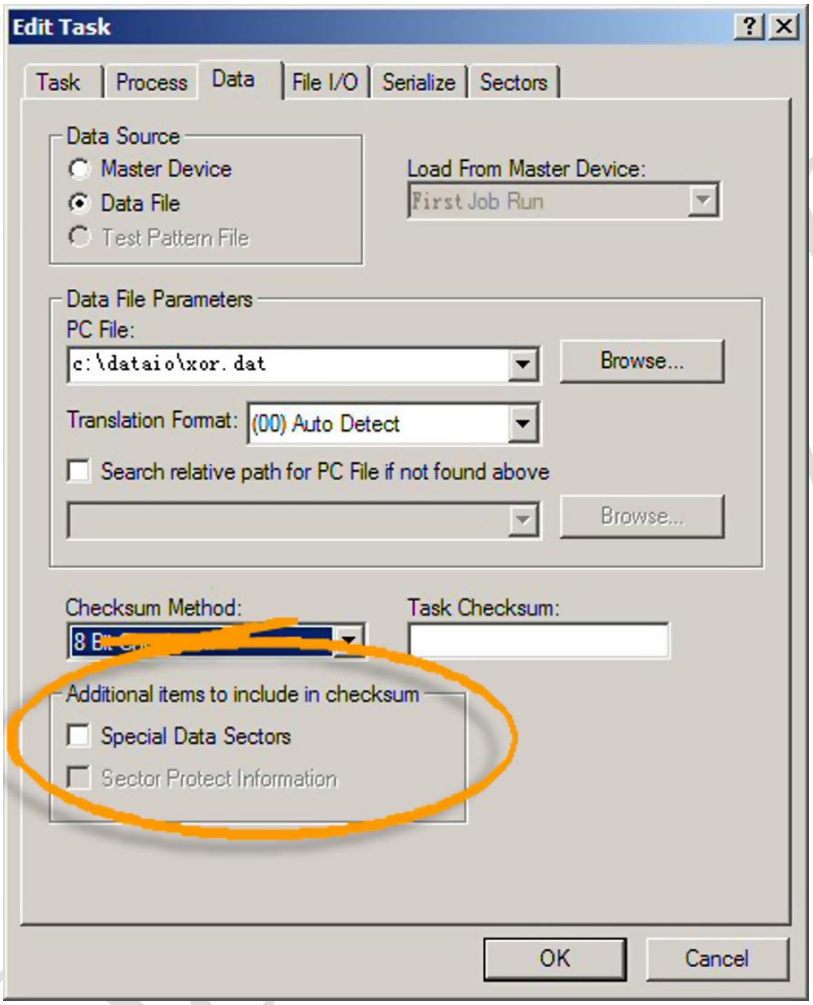

Figure 3: Additional items will be included in the checksum when the available boxes are checked.

Data I/O Checksum 计算方法

简介和使用范围:

本文描述了 TaskLink 计算 Checksum(以下简称 CS)的方法。

8/16/32-bit CS 应用场合：

8-bit,16-bit 和 32-bit CS 的计算只是简单的累加数据，与数据的位置无关，这一点与 CRC 计算不同。因此，整个区块数据的交换并不会影响 8-bit,16-bit 和 32-bit CS 的值。同样的，奇偶字节的交换也不会影响 8-bit CS 的值。

在 **File I/O** 这个选项卡中,您可以使用 Odd/Even Byte Swap 选项来交换奇偶字节中的数据，注意这个选项对 8-bit 的芯片无效。而对于 32-bit 的芯片，它也是只交换奇偶字节中数据，而不是奇偶字中的数据。

CS 计算方法

8-Bit CS

将 TaskLink 处理过的数据文件中数据按照每一个字节（BYTE）进行累加。超过 32-bit 的进位舍去。

例如，对于图 1 中的数据:
8-bit CS 为 0x37C (= 0x12 + 0x 34 + . . .)

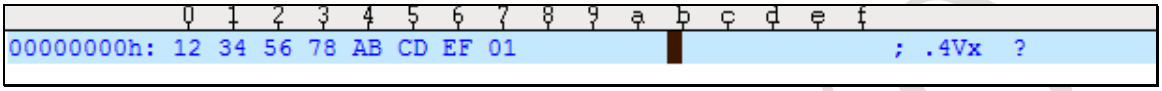

图4:示例数据。

16-Bit Checksum

将 TaskLink 处理过的数据文件中数据按照每一个字（WORD）进行累加. 超过 32-bit 的进位舍去。

例如，对于图 1 中的数据:
16-bit CS 为 0x17C02 (= 0x3412 + 0x7856 + . . .)

32-Bit Checksum

将 TaskLink 处理过的数据文件中数据按照每一个双字（DWORD）进行累加. 超过 32-bit 的进位舍去。

例如，对于图 1 中的数据:
32-bit CS 为 0x7A4601BD (= 0x78563412 + 0x01EFCDAB)

32-Bit CRC Checksum

公式为:
x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1.

例如，对于图 1 中的数据:
32-bit CRC 为 0xB9196C38

在 TaskLink 中选择计算方法：

您可以在 TaskLink 的 **Edit Task** 对话框中的 **Data** 选项卡中的 Checksum Method 下拉框中选择您需要的计算方式，如图 2 所示。

Device Width Checksum 这个选项将自动选择和芯片位数一致的 CS 计算方法。也就是说，8-bit 芯片使用 8-bit CS，16-bit 芯片使用 16-bitCS，32-bit 芯片使用 32-bit CS。

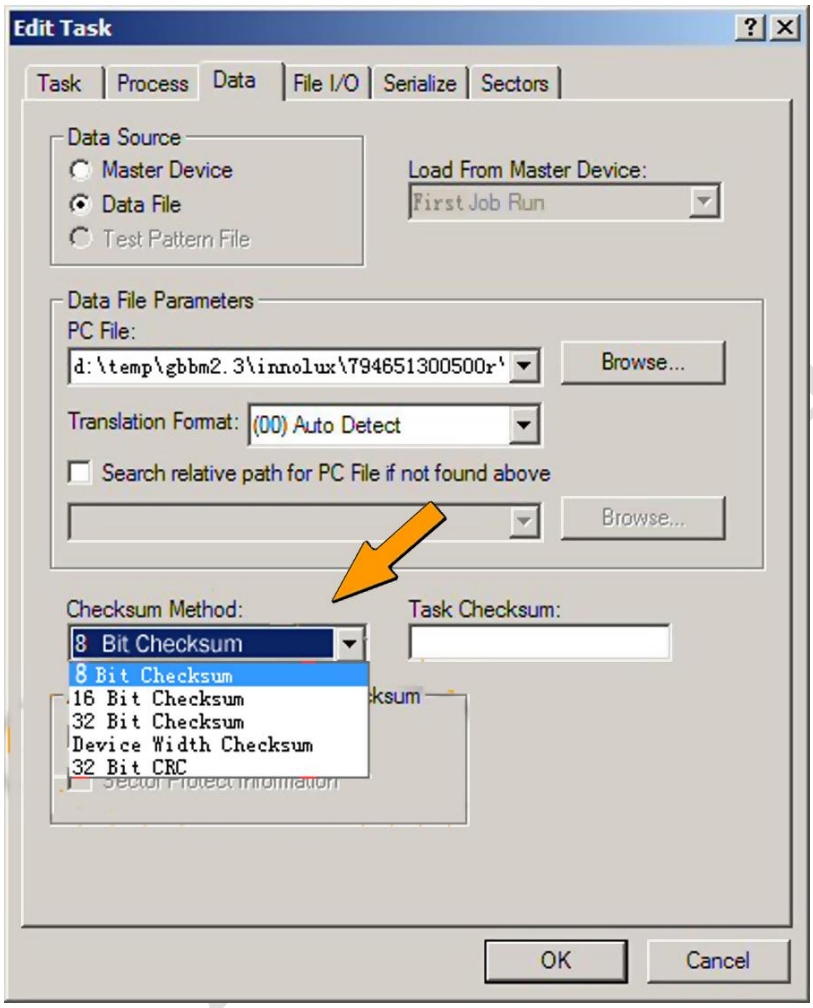

图 5: CS 计算方法下拉列表。

特殊数据和扇区保护：

如果您使用的芯片有特别数据扇区（如 OTP）和扇区保护功能，在您选中了图三中的单选框后，这些信息也会计算到 CS 中。

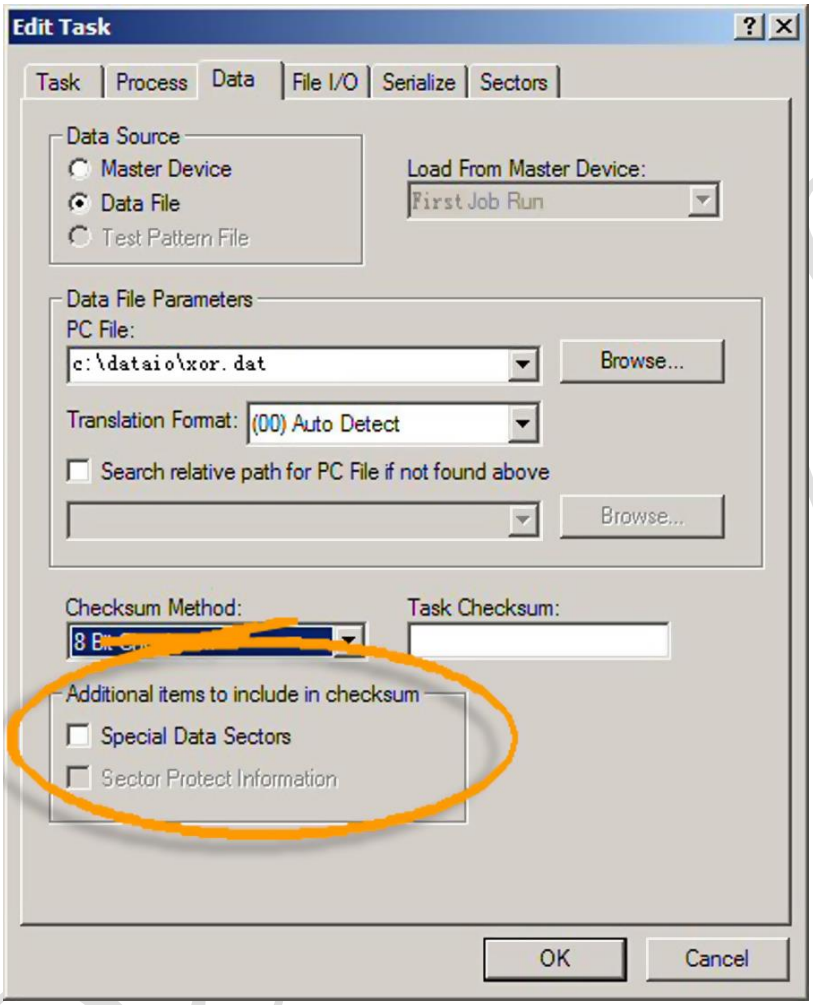

图 6:加入其他信息到 CS 的单选框.

Revision History:

V1.0	July 7, 2010	First Version
V1.1	Aug 13, 2010	Revised
V1.2	July 14, 2023	Added data explanation