应用笔记

Application Note

文档编号：**AN1101**

APM32F4xx_CRC 应用笔记

版本：**V 1.0**

1 引言

本应用笔记提供如何在 APM32F4xx 系列上配置和应用 CRC 外设，并介绍了 CRC 参数模型以及 CRC 算法。

目录

2 CRC 简介

CRC，全称是 Cyclic Redundancy Check ，即循环冗余校验码。该单元可将输入数据经过固定的多项式计算得到 32 位的 CRC 计算结果，主要用来检测或校验数据传输或者保存后的正确性与完整性。

2.1 CRC 参数模型

通常需要知道一个参数模型，才能计算出准确的 CRC 值。一个完整的 CRC 参数模型应该包含以下信息：POLY，WIDTH，INIT，REFIN，REFOUT，XOROUT。通常情况下，如果只提供了一个多项式而未说明其他参数，则初始化值 INIT 为 0x00，输入反转 REFIN 为 false，输出反转 REFOUT 为 false，异或输出 XOROUT 为 0x00。输入需要校验的数后，根据参数模型通过 CRC 算法进行计算，就可以得到校验值。其中：

1）POLY：十六进制多项式，但省略最高位 1。如 x8 + x4 + x3 + x + 1，二进制为 1 0001 1011，省略最高位 1，转换为十六进制为 0x1B。

2）WIDTH：生成的 CRC 数据位宽，例如 CRC-16，生成的 CRC 为 16 位。

3）NAME：参数模型名称。

4）INIT：代表 CRC 的初始值，和 WIDTH 位宽保持相同。

5）REFIN：值为 false 或者 true。false 表示在进行计算之前，原始数据不需要翻转，反之，true 表示需要翻转。例如原始数据：0x5D = 0101 1101，如果 REFIN 为 true，进行翻转之后为 1011 1010 = 0xBA。

6）REFOUT：true 或 false，是规定运算完成后得到的 CRC 值是否进行翻转的参数。

7）XOROUT：与计算结果进行异或运算，得到最终的 CRC 值，保证它的位宽和 WIDTH 相同。

2.1.1 常用 CRC 参数模型

关于 CRC 常用参数模型及其应用场合如表 1 所示。

表 1 常用 CRC 介绍

CRC 算法名称	多项式公式/16 进制多项式	16 进制多项式	输入值反转	输出值反转
CRC-5/ITU	X^5+X^3+X+1	15	true	true
CRC-5/ITU	X^5+X^3+X+1	15	true	true
CRC-5/USB	X^5+X^2+1	05	true	true
CRC-7/MMC	X^7+X^3+1	09	false	false
CRC-8/ITU	$X^8+X^7+X^3+X^2+1$	07	false	false
CRC-16/USB	$X^{16}+X^{12}+X^5+1$	8005	true	True
CRC-16/MODBUS	$X^{16}+X^{12}+X^5+1$	8005	true	true

CRC-32/ MPEG-2	$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$	04C11DB7	false	false
CRC-32	$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$	04C11DB7	true	true

2.2 CRC 算法

CRC 的概念是在需要发送的数据后面添加校验码，由此生成一个新的发送帧，发送给接收端。假设需要发送的数据为 K bits,末尾添加的校验码为 R bits,生成的新帧为 K+R bits。接收端接收到数据后，通过对校验码的比对进行数据传输正确与否的验证。这样的比对方式要求接收端和发送端确定一个共同的除数。在数据帧发送前，通过添加一个数进行去余处理。当数据传输正确时，CRC 计算结果应该没有余数，反之，存在余数则说明传输过程发生了错误。

具体步骤如下：

1）在通信前，发送端和接收端应先约定多项式的值，除数 P。除数 P 的位数应为校验码位数加 1。

2）发送端通过在原始的 K 位数据后面添加 R 个零的方式将原始数据向左移动 R 位。

3）进行模 2 除法运算。将长度为 K+R 的数据除以 P，进行循环计算，当余数的阶数小于 R 时停止计算。其中，得到余数为附加的校验码，当长度小于 R 位时，应在前面补齐零。

4）发送端将 R 位的校验码附加在原始数据后面，并将整个带有校验码的数据发送给接收方。

5）接收方在接收到数据后，以模 2 除法的方式使用除数 P 对数据进行除法运算。当出现余数时，证明传输过程中出现了错误；反之，传输正常。

下面以 CRC-8 多项式 0x07 给出应用实例。

1）CRC-8 多项式 0x07，源数据为 10101110；根据约定的多项式，我们初始化 CRC 校验值为 0x00。

2）在数据末尾添加 8 个零，即数据左移 8 位，得到：1010111000000000

3）取前面 8 位数据（10101110）与多项式 0x07 进行异或运算，得到结果

11001010。

4） 将结果向左移动 1 位，丢弃最高位，得到结果 10010100。 重复以上步骤，依次从结果中取 8 位数据与 0x07 异或，再进行左移，直到余下的数据位数小于 8。最后的余数是 00101011。这个 00101011 就是计算得到的 CRC 校验码。

5）将它附加在原始数据的末尾，得到发送的数据：1010111000101011。接收端接收到数据后，同样进行 CRC 校验。

6）按照相同的过程，对接收到的数据进行模 2 除法运算，直到最后得到的余数为 0，表示没有错误发生。如果余数不为 0，则说明存在错误。

3 APM32F4XX CRC 介绍

3.1 APM32F4XX CRC 参数模型

APM32 自带的硬件 CRC 计算模块使用的是 CRC-32 模型，其多项式为 0x4C11DB7——X32 + X26 + X23 + X22 + X16 + X12 + X11 + X10 + X8 + X7 + X4 + X2 + X +1。

按照前文内容 2.1提及的模型参数分析说明：APM32 CRC 数据位宽为 32 位，十六进制多项式为 0x4C11DB7，INIT=0xFFFFFFFF，REFIN=false，REFOUT=false，XOROUT=0x00000000。

3.2 硬件实现优势

APM32 自带的硬件 CRC 计算模块，使用硬件实现 CRC 带来以下一些优势：

1）高速计算：硬件实现 CRC 可以利用专门的电路和并行计算来实现快速的 CRC 计算。相比于软件实现，硬件可以在几个时钟周期内完成 CRC 计算，提供更高的处理速度。

2）低功耗：由于使用了专门的电路来执行 CRC 计算，硬件实现可以通过电路设计的优化来降低功耗。相比于在软件中使用循环来计算 CRC，硬件实现可以节省大量的能量。

3）占用资源少：软件实现 CRC 可能需要较大的存储器和处理器资源来执行复杂的计算。而硬件实现通常只需要一些逻辑门和寄存器，占用的资源较少，尤其适用于资源有限的嵌入式系统。

4）实时性：硬件实现的 CRC 可以在实时数据流中进行连续计算，没有延迟或中断。这对于需要快速处理数据流的应用非常重要，如实时传输、通信协议等。

5）硬件抗干扰能力强：硬件实现的 CRC 通常能够更好地抵御噪声和干扰，因为它可以采用差分信号和物理层技术来提高抗干扰能力。这对于数据传输的可靠性和错误检测非常重要。

综上，CRC 的硬件实现具有高速计算、低功耗、占用资源少、实时性好以及较强的抗干扰能力等优势。这使得硬件实现在需要高效率、可靠性和实时性的应用中更加有优势。

4 APM32F4XX CRC 应用例程

4.1 软件设计流程

APM32F4XX 例程中已根据数据算出标准校验值，网页计算过程如图 3 所示，计算得标准校验值为 0x379E9F06。将待校验数据串存放在数组中，进行 CRC 校验，当输出校验值与标准校验值相等时，证明传输正常；反之，二者不同时证明数据传输出错。

图 3 标准校验值计算

CRC 计算单元包含一个 32 位的数据寄存器（CRC_DATA），用于进行数据输入和保存 CRC 计算结果。其计算时间为 4 个 AHB 时钟周期。每写入一次新数据，其结果是上一次的计算结果和新的计算结果的组合，（对整个字进行运算）。在计算期间，会暂停 CPU 的写操作，因此可以对寄存器 CRC_DATA 进行“背靠背”写入或连续地“读—写”操作。要计算支持数据的 CRC，需要按照以下步骤进行操作:

1) 通过 RCC 外设启用 CRC 外设时钟。

2) 配置初始 CRC 值寄存器（CRC_DATA）将 CRC 数据寄存器设置为初始 CRC 值。

3) 通过 CRC 控制寄存器（CRC_CTRL）中的 Reset 位将 CRC 外设重置。

4) 将数据设置到 CRC 数据寄存器。

5) 读取 CRC 数据寄存器的内容，即为计算得到的 CRC 值。

软件编写流程图如下：

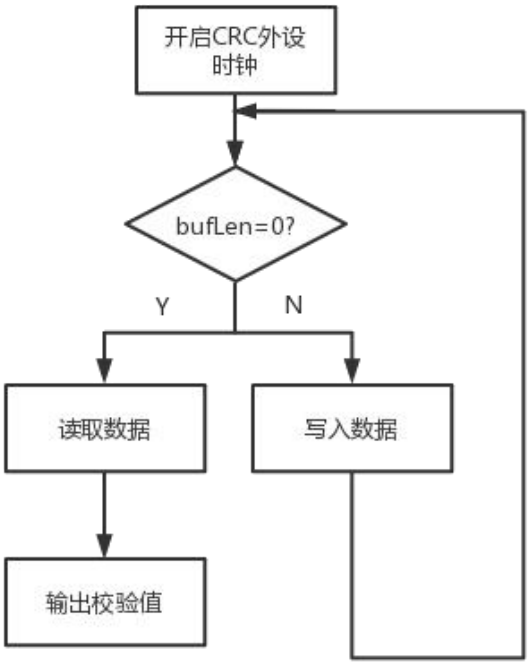

图 4　CRC 软件编写流程图

4.2　软件实现

4.2.1　CRC 参数定义

定义数组存放待校验数据，定义 32 位整型变量用于存储 CRC 输出校验值；同时完成串口初始化及相关参数设定：将波特率设置为 115200，校验位为 0，停止位为 1 等。相关代码如下：

```
static const uint32_t aDataBuffer[BUFFER_SIZE] =

  {

    0x00001021, 0x20423063, 0x408450a5, 0x60c670e7, 0x9129a14a,
    0xb16bc18c, 0xd1ade1ce, 0xf1ef1231, 0x32732252, 0x52b54294,
    0x72f762d6, 0x93398318, 0xa35ad3bd, 0xc39cf3ff, 0xe3de2462,
    0x34430420, 0x64e674c7, 0x44a45485, 0xa56ab54b, 0x85289509,
    0xf5cfc5ac, 0xd58d3653, 0x26721611, 0x063076d7, 0x569546b4,
    0xb75ba77a, 0x97198738, 0xf7dfe7fe, 0xc7bc48c4, 0x58e56886,
    0x78a70840, 0x18612802, 0xc9ccd9ed, 0xe98ef9af, 0x89489969,
    0xa90ab92b, 0x4ad47ab7, 0x6a961a71, 0x0a503a33, 0x2a12dbfd,
    0xfbbfeb9e, 0x9b798b58, 0xbb3bab1a, 0x6ca67c87, 0x5cc52c22,
    0x3c030c60, 0x1c41edae, 0xfd8fcdec, 0xad2abd0b, 0x8d689d49,
    0x7e976eb6, 0x5ed54ef4, 0x2e321e51, 0x0e70ff9f, 0xefbedfdd,
    0xcffcbf1b, 0x9f598f78, 0x918881a9, 0xb1caa1eb, 0xd10cc12d,
    0xe16f1080, 0x00a130c2, 0x20e35004, 0x40257046, 0x83b99398,
    0xa3fbb3da, 0xc33dd31c, 0xe37ff35e, 0x129022f3, 0x32d24235,
    0x52146277, 0x7256b5ea, 0x95a88589, 0xf56ee54f, 0xd52cc50d,
    0x34e224c3, 0x04817466, 0x64475424, 0x4405a7db, 0xb7fa8799,
    0xe75ff77e, 0xc71dd73c, 0x26d336f2, 0x069116b0, 0x76764615,
    0x5634d94c, 0xc96df90e, 0xe92f99c8, 0xb98aa9ab, 0x58444865,
    0x78066827, 0x18c008e1, 0x28a3cb7d, 0xdb5ceb3f, 0xfb1e8bf9,
    0x9bd8abbb, 0x4a755a54, 0x6a377a16, 0x0af11ad0, 0x2ab33a92,
    0xed0fdd6c, 0xcd4dbdaa, 0xad8b9de8, 0x8dc97c26, 0x5c644c45,
    0x3ca22c83, 0x1ce00cc1, 0xef1fff3e, 0xdf7caf9b, 0xbfba8fd9,
    0x9ff86e17, 0x7e364e55, 0x2e933eb2, 0x0ed11ef0

  };

uint32_t uCRCValue = 0;


USART_Config_T usartConfigStruct; /* USART configuration */
USART_ConfigStructInit(&usartConfigStruct);
usartConfigStruct.baudRate = 115200;
usartConfigStruct.mode = USART_MODE_TX_RX;
usartConfigStruct.parity = USART_PARITY_NONE;
usartConfigStruct.stopBits = USART_STOP_BIT_1;
usartConfigStruct.wordLength = USART_WORD_LEN_8B;
usartConfigStruct.hardwareFlow = USART_HARDWARE_FLOW_NONE;
```

4.2.2 启用 CRC 外设时钟

在开始使用 CRC 前，需要先通过 RCM 外设启用 CRC 外设时钟。

```
RCM_EnableAHB1PeriphClock(RCM_AHB1_PERIPH_CRC);
```

4.2.3 配置初始 CRC 值

将 CRC_CTRL 寄存器 RST 位置 1，配置初始 CRC 值寄存器（CRC_DATA）将 CRC 数据寄存器设置为初始 CRC 值 0xFFFFFFF。

```
CRC_ResetDATA();
```

4.2.4 CRC 数据校验

完成了开启时钟，初始值的配置后，调用 CRC 计算函数，利用 while 循环往数据寄存器里以 32 位为单位写入数据。当 BUFFER_SIZE 等于零时，意味着数据写入完毕，读取数据寄存器的值，得到的即为输出校验值。可以通过串口助手观察到标准校验值和输出校验值，当输出校验值与标准校验值相等时，证明数据传输没有出错。

其中 32 位数据（32bit Data）作为输入寄存器是写入时存储 CRC 计算器的新数据。作为输出寄存器是读取时返回 CRC 计算的结果。

```
CRC_ResetDATA();
uCRCValue=CRC_CalculateBlockCRC((uint32_t*)aDataBuffer,BUFFER_SIZE);
printf("BlockCRC = 0x379E9F06 \r\n");
printf("CalculateBlockCRC = 0x%08X \r\n", uCRCValue);
```

CRC_CalcBlockCRC 函数代码如下：

```
uint32_t CRC_CalculateBlockCRC(uint32_t *buf, uint32_t bufLen)
{
  while(bufLen--)
  {
    CRC->DATA = *buf++;
  }
  return (CRC->DATA);
}
```

5 版本历史

表 2 文档版本历史记录

日期	版本	变更历史
2023.07.05	1.0	初稿

声明

本手册由珠海极海半导体有限公司（以下简称“极海”）制订并发布，所列内容均受商标、著作权、软件著作权相关法律法规保护，极海保留随时更正、修改本手册的权利。使用极海产品前请仔细阅读本手册，一旦使用产品则表明您（以下称“用户”）已知悉并接受本手册的所有内容。用户必须按照相关法律法规和本手册的要求使用极海产品。

1、权利所有

本手册仅应当被用于与极海所提供的对应型号的芯片产品、软件产品搭配使用，未经极海许可，任何单位或个人均不得以任何理由或方式对本手册的全部或部分内容进行复制、抄录、修改、编辑或传播。

本手册中所列带有“®”或“TM”的“极海”或“Geehy”字样或图形均为极海的商标，其他在极海产品上显示的产品或服务名称均为其各自所有者的财产。

2、无知识产权许可

极海拥有本手册所涉及的全部权利、所有权及知识产权。

极海不应因销售、分发极海产品及本手册而被视为将任何知识产权的许可或权利明示或默示地授予用户。

如果本手册中涉及任何第三方的产品、服务或知识产权，不应被视为极海授权用户使用前述第三方产品、服务或知识产权，除非在极海销售订单或销售合同中另有约定。

3、版本更新

用户在下单购买极海产品时可获取相应产品的最新版的手册。

如果本手册中所述的内容与极海产品不一致的，应以极海销售订单或销售合同中的约定为准。

4、信息可靠性

本手册相关数据经极海实验室或合作的第三方测试机构批量测试获得，但本手册相关数据难免会出现校正笔误或因测试环境差异所导致的误差，因此用户应当理解，极海对本手册中可能出现的该等错误无需承担任何责任。本手册相关数据仅用于指导用户作为性能参数参照，不构成极海对任何产品性能方面的保证。

用户应根据自身需求选择合适的极海产品，并对极海产品的应用适用性进行有效验证和测试，以确认极海产品满足用户自身的需求、相应标准、安全或其它可靠性要求；若因用户未充分对极海产品进行有效验证和测试而致使用户损失的，极海不承担任何责任。

5、合规要求

用户在使用本手册及所搭配的极海产品时，应遵守当地所适用的所有法律法规。用户应了解产品可能受到产品供应商、极海、极海经销商及用户所在地等各国有关出口、再出口或其它法律的限制，用户（代表其本身、子公司及关联企业）应同意并保证遵守所有关于取得极海产品及／或技术与直接产品的出口和再出口适用法律与法规。

6、免责声明

本手册由极海“按原样”（as is）提供，在适用法律所允许的范围内，极海不提供任何形式的明示或暗示担保，包括但不限于对产品适销性和特定用途适用性的担保。

对于用户后续在针对极海产品进行设计、使用的过程中所引起的任何纠纷，极海概不承担责任。

7、责任限制

在任何情况下，除非适用法律要求或书面同意，否则极海和/或以“按原样”形式提供本手册的任何第三方均不承担损害赔偿责任，包括任何一般、特殊因使用或无法使用本手册相关信息而产生的直接、间接或附带损害（包括但不限于数据丢失或数据不准确，或用户或第三方遭受的损失）。

8、适用范围

本手册的信息用以取代本手册所有早期版本所提供的信息。