ANALOG DEVICES

AN-724
应用笔记

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A. • Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com

ADuC70xx串行下载协议

简介

微转换器(MicroConverter®)系列产品的一个关键特性是可以在线下载代码到片内FLASH/EE程序存储器。在线代码下载是通过器件UART串行端口进行的，因此一般被称为串行下载。利用串行下载功能，开发人员可以在将器件直接焊接到目标系统的同时对其重新编程，从而不需要外部器件编程器。此外，只需一个能访问微转换器的串行端口，就可以在现场通过串行下载特性执行 系统升级。这意味着制造商可以在现场升级系统固件，而不必换出器件。

在上电时或施加外部复位信号期间，通过特定引脚配置可以将任何微转换器配置为串行下载模式。对于ADuC70xx系列微转换器，BM输入引脚可以通过一个电阻(1 kΩ)拉低。在上电时或施加硬复位输入信号期间，如果器件检测到这一状况，器件将进入串行下载模式。在此模式下，片内驻留的加载器程序会启动，配置器件UART，并通过特定串行下载协议与任何主机通信，以管理下载的数据，将其存入Flash/EE存储器空间。要下载的程序数据必须是从小到大顺序格式。

注意，串行下载模式工作在器件的标准电源额定值范围(2.7 V至3.6 V)，无需特别高的编程电压，因为它是在片内产生的。图1显示了在评估板上如何进入串行下载模式。

用户可以使用ADI公司提供的一个Windows®程序(ARMWSD.exe，一款QuickStart™开发工具)，通过PC串行端口COM1至COM31下载代码到微转换器。不过请注意，任何支持串行下载协议的主机(PC、微控制器或DSP等)都可以将代码下载到微转换器，本应用笔记将详细介绍该串行下载协议。

本应用笔记详细描述了微转换器串行下载协议，以便最终用户能够理解该协议(嵌入式主机至嵌入式微转换器)，并且能将该协议成功应用到目标系统中。

为明确起见，这里使用的术语“主机”(Host)指的是尝试向微转换器下载数据的宿主机器(PC、微控制器或DSP)；术语“加载器”(loader)指的是微转换器内置的片内串行下载固件。

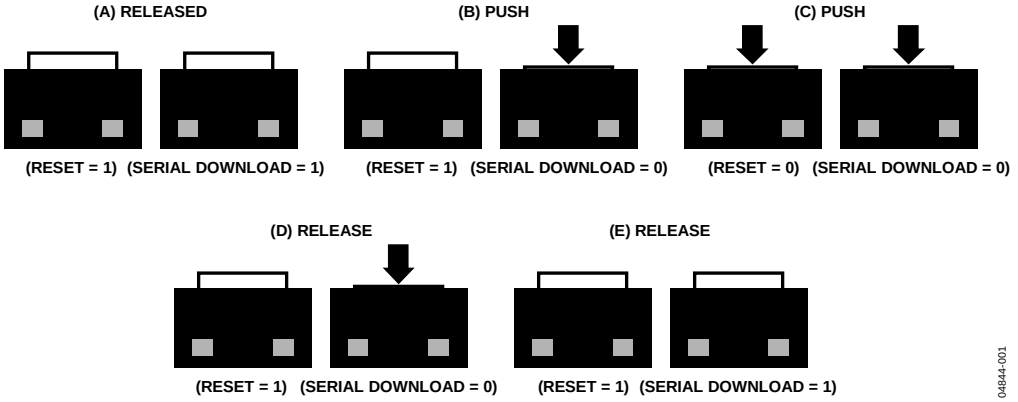

图 1.串行下载模式下的微转换器

目录

运行微转换器加载器

为了运行ADuC70xx微转换器上的加载器，必须通过一个电阻(通常为1 kΩ下拉电阻)拉低串行下载$\overline{BM}$引脚，并且复位器件。切换器件本身的RESET输入引脚或者周期供电可复位器件。

物理接口

一旦触发，加载器就等待主机发送退格(BS = 0x08)字符进行同步。加载器测量此字符的时序，并相应地配置微转换器UART串行端口用无奇偶性的8个数据位开始，以主机的波特率进行发送或接收。波特率必须在600 bps至115,200 bps之间(含本数)。注意，ADuC7060/ADuC7061的最高波特率为38400 bps。收到退格字符后，加载器即发送如下24字节ID数据包：

15字节 = 产品标识符
3字节 = 硬件和固件的版本号
4字节 = 保留，以备后用
2字节 = 换行和回车

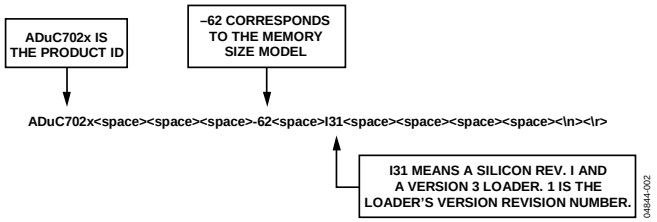

图 2.ID数据包示例

定义数据传输包格式

一旦UART配置完成，数据传输即可开始。通用通信数据传输包格式如表1所示。

数据包起始ID字段

第一个字段是数据包起始ID字段，它包括两个起始字符(0x07和0x0E)。这些字节为常数，用于加载器检测一个数据传输包的有效起始。

数据字节数字段

接下来的字段是数据字节的总数，包括数据1(命令功能)。数据字节最小数目是5，对应命令功能和地址。数据字节最大数目是255：一个命令功能、4字节的地址和250字节的数据。

命令功能字段(数据1)

命令功能字段描述数据包的功能。允许使用5个命令功能中的一个。这5个命令功能分别由一个ASCII字符表示：E、W、V、P或R。数据包命令功能如表2所示。

地址字段(数据2至数据5)

地址字段包含一个32位地址(h、u、m、l)，MSB位于h中，LSB位于l中。

数据字节字段(数据6至数据255)

用户代码是按字节下载/验证的。数据字节字段最多包含250个数据字节。

数据一般是Intel® Hex扩展16字节记录格式(参见Intel扩展十六进制部分)，并且在传输到加载器之前作为表1所述数据包的一部分由主机重新编译。

校验和字段

数据包校验和写入校验和字段。该二进制补码校验和是通过对字节数字段的十六进制值和数据1至数据255字段(以实际存在的数据字段计算)的十六进制值求和而算得的。校验和即是该和的二进制补码值。因此，从数据字节数到校验和的所有字节之和的LSB应当为0。这可以用数学公式表示为：

$$CS = 0x00 - (Number\ of\ Data\ Bytes + \sum_{1}^{N=255} Data\ Byte_N)$$

换言之，除起始ID外的所有字节的8位和必须等于0x00。

命令应答

加载器程序对每个数据包都会发出一个应答：否定应答BEL (0x07)或肯定应答ACK (0x06)。

如果接收到的校验和不正确或地址无效，加载器就会发送一个BEL信号。如果下载数据覆盖旧数据(没有擦除)，加载器不会给出警告。PC接口必须确保代码下载的所有位置都被擦除。

表1. 数据传输包格式

起始ID						
ID0	ID1	数据字节数	数据1 CMD	数据2至数据5	数据x (x = 6至255)	校验和
0x07	0x0E	5至255	E, W, V, P, or R	h, u, m, l	xx	CS

表2. 数据包的命令功能

命令功能	数据1字段中的命令字节	加载器肯定应答	加载器否定应答
擦除页	E (0x45)	ACK (0x06)	BEL (0x07)
写	(0x57)	ACK (0x06)	BEL (0x07)
验证	V (0x56)	ACK (0x06)	BEL (0x07)
保护	P (0x50)	ACK (0x06)	BEL (0x07)
运行(跳转至用户代码)	R (0x52)	ACK (0x06)	BEL (0x07)

表3. 擦除Flash/EE存储器命令

起始ID						
ID0	ID1	数据字节数	数据1 CMD	数据2至数据5	数据6（页）	校验和
0x07	0x0E	6	E (0x45)	h, u, m, l	x pages (1 to 124)	CS

表4. Flash/EE存储器编程命令

起始ID						
ID0	ID1	数据字节数	数据1 CMD	数据2至数据5	数据x (x = 1至250)	校验和
0x07	0x0E	5 + x (6 to 255)	W (0x57)	h, u, m, l	Data bytes	CS

表5. 验证命令、位修改

原始位	发送位	恢复位
7	4	7
6	3	6
5	2	5
4	1	4
3	0	3
2	7	2
1	6	1
0	5	0

擦除命令

擦除命令允许用户从数据2至数据5决定的特定页开始擦除Flash/EE。擦除地址下调为页面起始位置。该命令还包括要擦除的页数。如果地址为0x00000000，页数为0x00，加载器将认为是批量擦除命令，即会擦除整个用户代码空间和Flash/EE保护。擦除数据包命令如表3所示。

写入命令

写入命令包括数据字节数(5 + x)、命令、要编程的第一个数据字节地址和要编程的数据字节。数据下载后就被编程到Flash/EE中。如果校验和不正确或者接收地址超出范围，加载器将发送一个BEL信号。如果主机从加载器接收到一个BEL信号，主机应中止下载过程，并重新开始整个下载过程。

验证命令

验证命令与写入命令几乎一致，如表5所示。命令字段是V (0x56)，但为了提高检测错误的机率，数据字节作了改变：低5位被移至高5位，高3位被移至低3位。

加载器恢复正确位序列，并同Flash存储器内容相比较。如果位序列和校验和均正确，则返回ACK(0x06)，否则返回BEL(0x07)。

Flash/EE存储器保护命令

使用此命令必须遵循如下三个步骤：

1. 初始化命令。类型必须是0x00，h、u、m、l可以是任意值。
2. 发送要保护页的组地址。对每组页面重复执行这一步。类型必须是0x0F。
3. 发送h、u、m、l中的密钥；类型必须是0x01。FEEADR取hu的值，FEEDAT取ml的值。如果不需要密钥，h、u、m、l必须是0xFFFFFFFF。

例如，为了保护0至7页不被写入，应设置读保护并使用密钥0x12345678。必须发送下列命令：

- 起始序列：
  0x07 0x0E 0x06 0x50 0xXXXXXXXX 0x00 CS
- 保护：
  0x07 0x0E 0x06 0x50 0x00000000 0x0F CS
  (0到3页)
  0x07 0x0E 0x06 0x50 0x00000200 0x0F CS
  (4到7页)
  0x07 0x0E 0x06 0x50 0x0000F800 0x0F CS
  (读保护)
- 密钥和结束序列：
  0x07 0x0E 0x06 0x50 0x12345678 0x01 CS

注意：保护命令仅适用于版本0及后续版本的加载器。对于版本0，FEEADR = ml，FEEDAT = ml。而在后续版本中，FEEADR = hu。

本协议不允许取消对Flash/EE存储器的保护。要解除保护，可使用批量擦除命令。

远程运行命令

一旦主机将所有数据包都发送到加载器，主机便可发送最后一个包以指示加载器开始执行代码。

本协议实现了两类远程运行：

- 软件复位(h, u, m, l = 0x1)。
- 跳转至用户代码(h, u, m, l = 0x0)。

表8给出了一个远程运行或复位的实例。建议采用软件复位，因为它可以重置所有外设。

表6. 验证Flash/EE存储器命令

起始ID						
ID0	**ID1**	**数据字节数**	**数据1 CMD**	**数据2至数据5**	**数据x (x = 1至250)**	**校验和**
0x07	0x0E	5 + x (6至255)	V (0x56)	h, u, m, l	修改的数据字节	CS

表7. Flash/EE存储器保护命令

起始ID						
ID0	**ID1**	**数据字节数**	**数据1 CMD**	**数据2至数据5**	**数据6**	**校验和**
0x07	0x0E	0x06	P (0x50)	h, u, m, l	类型	CS

表8. 远程运行命令

包ID					
ID0	**ID1**	**数据字节数**	**数据1 CMD**	**数据2至数据5**	**校验和**
0x07	0x0E	0x05	R (0x52)	h, u, m, l = 0x1	0xA8

INTEL扩展十六进制格式

d是一种以可显示的ASCII字符或可打印格式存储机器语言的标准。它与Hex 8格式相似，但它所输出的Intel扩展线性地址记录也用于建立高16位数据地址。每个数据记录都以冒号开始，然后是一个8字符前缀，最后以2字符校验和结束。每个记录都有如下格式：

:BBAAAATTHHHH....HHHCC

其中：
BB是2位数十六进制字节计数，代表该行出现的数据字节数。

AAAA是4位数十六进制地址，代表数据记录的起始地址。

TT是2位数记录类型：
- 00–数据记录
- 01–文件结束记录
- 02–扩展段地址记录
- 03–起始段地址记录
- 04–扩展线性地址记录
- 05–起始线性地址记录

HH是2位数十六进制数据字节。

CC是2位数十六进制校验和，它是记录中前面所有字节之和的二进制补码，包括前缀(所有字节之和 + 校验和 = 00)。

记录类型

数据记录

记录类型00代表数据记录，它包含文件的数据。数据记录以冒号起始符(:)开始，然后是字节计数(10)、第一个字节的地址(0000)和记录类型(00)。记录类型之后是数据字节。数据字节之后是校验和，它是记录中前面所有字节(不包括起始字符）之和的二进制补码。下面是数据记录的实例(其中的空格仅为清楚起见而插入，并不存在于实际的目标文件中)：

:10 0000 00 FFFEFDFCFBFAF9F8F7F6F5F4F3F2F1F0 78

:05 0010 00 0102030405 DC

结束记录

记录类型01代表结束记录，表示数据文件的末尾。结束记录以冒号起始符(:)开始，然后是字节计数(00)、地址(0000)、记录类型(01)和校验和(FF)。例如：

:00 0000 01 FF

扩展段地址记录

记录类型02代表扩展段地址记录，定义段基址的位4至位19。它可以出现在目标文件中的任何地方，并且会影响其后所有数据记录的绝对存储器地址，直到它发生改变。扩展段地址记录以冒号起始符(:)开始，然后是字节计数(02)、地址(0000)、记录类型(02)、由段基址的位4至位19代表的4字符十六进制数(1000)和2字符校验和(FB)。例如：

:02 0000 02 1000 FB

起始段地址记录

记录类型03代表起始段地址记录，定义目标文件的执行起始段基址的位4至位19。例如：

:02 0000 03 0000 FB

扩展线性地址记录

记录类型04代表扩展线性地址记录，定义目的地址的位16至位31。它可以出现在目标文件中的任何地方，并且会影响其后所有数据记录的绝对存储器地址，直到它发生改变。扩展线性地址记录以冒号起始符(:)开始，然后是字节计数(02)、地址(0000)、记录类型(04)、由目的地址的位16至位31代表的4字符十六进制数(FFFF)和2字符校验和(FC)。例如：

:02 0000 04 FFFF FC

起始线性地址记录

记录类型05代表起始线性地址记录，定义目标文件的执行起始地址的位16至位31。例如：

:02 0000 05 0000 F9

Intel十六进制目标文件示例

下面是一个Intel十六进制目标文件的示例，包含下列记录：扩展线性地址记录、扩展段地址记录、数据记录和结束记录。

:020000040108F1
:0200000212FFEB
:0401000090FFAA556D
:00000001FF

1. 确定数据记录的扩展线性地址偏移（本例中为0108）。
   :02 0000 04 0108 F1
2. 确定数据记录的扩展段地址（本例中为12FF）。
   :02 0000 02 12FF EB
3. 确定数据记录中的数据的地址偏移（本例中为0100）。
   :04 0100 00 90FFAA55 6D
4. 计算数据记录第一个字节的绝对地址。
   + 0108 0000(线性地址偏移左移16位)
   + 0001 2FF0(段地址偏移左移4位)
   + 0000 0100(相对于数据记录的地址偏移)
   = 0109 30F0(第一个数据字节的32位地址)
5. 计算：
   010930F0 90
   010930F1 FF
   010930F2 AA
   010930F3 55

限制

未实现记录类型02、03、04和05。不支持的记录会被忽略。仅低16位地址对于访问内部Flash存储器有意义，因此可以放心地忽略改变高16位地址的记录类型。

注释

AN04844-0-6/09(C)