基于CAN总线多节点并行升级协议设计

侯文杰
经纬纺织机械股份有限公司　北京　102600

摘　要：为了解决频繁的在现场升级多节点嵌入式设备固件问题，介绍基于CAN总线多节点并行升级固件方法。通过利用CAN总线的广播特性，向所有节点传输固件，大幅减少固件传输时间。同时，兼顾了不同功能节点的固件升级。

关键词：CAN；广播；并行；IAP

引言

IAP是In Application Programming的首字母缩写，IAP是用户自己的程序在运行过程中对User Flash的部分区域进行烧写，目的是为了在产品发布后可以方便地通过预留的通信口对产品中的固件程序进行更新升级。

CAN是控制器域网 (Controller Area Network, CAN) 的简称，是由研发和生产汽车电子产品著称的德国BOSCH公司开发的，并最终成为国际标准（ISO11898）。是国际上应用最广泛的现场总线之一。

传统的IAP都是点对点升级，通过CAN-ID来确定要升级的节点，然后将固件传输给该节点，完成升级，升级过程中，不能有其他节点向该节点发送数据，否则会导致升级失败。该过程一般会持续几分钟，如果设备CAN总线上有几十个节点，依次升级，完成升级需要的时间就难以忍受了。

本文介绍一种利用CAN总线的广播特性，并行升级固件的方法。同时，该方法也兼容了不同功能节点的固件更新。

并行升级原理

按照硬件来划分，系统包括上位机和嵌入式控制板。按照软件功能来划分，系统包括升级程序，bootloader，嵌入式应用程序。其中，升级程序运行在上位机中，bootloader和嵌入式应用程序，都运行在嵌入式控制板中。系统架构图如图1所示。

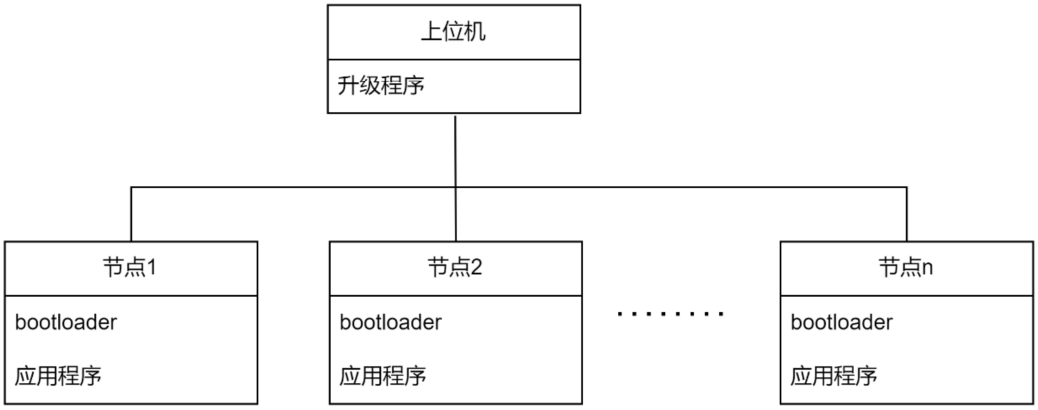

图1　系统架构图

并行升级协议主要分为三种命令控制命令，查询命令，广播命令。控制命令和查询命令是点对点通讯，控制命令发送完成后，查询命令查询状态，检查控制命令是否执行成功。广播命令用来发送固件程序，所有节点同时接收。

所有节点设置为全接收状态，通过软件来过滤和自身相关的数据。节点CAN-ID划分，比如n=100，上位机CAN-ID为0x65，节点1-节点n的CAN-ID为0x01-0x64，广播CAN-ID为0x780。节点1只接收CAN-ID为0x01或0x780的数据帧，节点n只接收CAN-ID为n或者0x780的数据帧，即每个节点只接收自身CAN-ID和广播CAN-ID。由于上位机需要控制和查询所有节点，上位机接收的CAN-ID范围为0x1-0x64。

对于一种场景，所有的节点固件程序都是相同的，在固件升级传输过程中，固件数据帧占有非常大的比例。将固件数据采用广播来传输，所有节点并行接收固

件数据，将原本需要重复传输100次的固件数据缩减到1次，大幅降低了数据传输量，从而缩短升级时间，节点数越多收益越大。升级单个节点和多个节点的时间差异，主要在控制命令和查询命令的时间，这两种命令在传统IAP升级过程中也无法省去。由于控制命令和查询命令数据帧占比很小，因此，升级100个节点和升级1个节点时间相差不多，能够大幅提高升级效率。

命令介绍

• 应用跳转bootloader命令

升级跳转　IDH, IDL, 0x55, 0x55, 0x01, 0x01

不升级跳转　IDH, IDL, 0x55, 0x55, 0x01, 0x02

该命令为点对点通讯，控制节点进入bootloader程序，使总线所有节点都不能主动发送数据，将控制权交给上位机，节点向上位机回复自身CAN-ID，上位机根据回复来决定是否对其进行下一步操作。其中，IDH和IDL表示CAN-ID的高低字节。“升级跳转”，表示该节点需要升级。“不升级跳转”，表示该节点不需要升级。由此命令来兼容，不同类型节点升级过程。每次升级固件时，选定升级的节点必须是同类型节点。

• 擦除命令

IDH, IDL, 0x55, 0x55, 0x03, 0x00, pageCntH, pageCntL

该命令为点对点通讯，控制节点擦除Flash。上位机依次对跳转成功的节点发送该命令，节点向上位机回复自身CAN-ID，并开始擦除Flash，上位机根据回复来决定是否对其进行下一步操作。pageCntH和pageCntL分别表示页数的高低字节，该信息由固件大小和节点Flash页大小计算得出。页数信息在固件传输结束后，用于页数据的校验，判断是否存在页缺失的情况。

• 擦除状态查询

IDH, IDL, 0x55, 0x55, 0x04, 0x01

该命令为点对点命令，查询擦除命令的执行状态。上位机依次对擦除命令有回复的节点发送该命令，节点向上位机回复自身CAN-ID。上位机根据回复来决定是否对其进行下一步操作。由于不同类型的嵌入式设备的Flash擦除时间不同，为保证嵌入式设备能成功擦除，该命令需要和上一条命令保持时间间隔，根据实际情况进行调整。

• 页传输命令

起始命令　0x07, 0x80, 0x55, 0x55, 0x55, 0x55

页信息　0x07, 0x80, lenH, lenL, addrH, addrL, pageNoH, pageNoL

数据帧　0x07, 0x80, 0xXX, 0xXX, …

结束命令　0x07, 0x80, 0xAA, 0xAA

该命令为广播命令，向所有并行升级的节点广播固件数据。上位机对总线按照顺序广播，起始命令，页信息，数据帧，结束命令。节点根据起始命令和结束命令，判断页数据是否接收完成。根据数据长度信息lenH、lenL，对数据帧字节数进行校验，判断页数据是否完整无误，是否存在丢帧情况，如果数据完整无误，根据起始地址信息addrH、addrL，将数据写入到Flash的对应地址。通过当前页信息的页号pageNoH、pageNoL和上一次页信息的页号pageNoH、pageNoL，判断当前页号是否与上一次页号加1相等，如果相等，则页数据是连续的，否则存在页丢失的情况。通过擦除命令中的页数pageCntH、pageCntL和页信息的页号pageNoH、pageNoL对比，判断固件是否烧写结束。命令中lenH和lenL表示数据字节数高低字节，addrH和addrL表示起始地址，pageNoH和pageNoL表示页号。

• 页查询命令

查询命令　IDH, IDL, 0x55, 0x55, 0x04, 0x01

该命令为点对点命令，查询页传输命令的执行状态。上位机对上一次查询有回复的节点依次发送该命令，节点向上位机回复自身CAN-ID，表示页烧写成功。上位机根据回复的节点，继续发送页传输命令，发送完成后，发送页查询命令，重复该过程，直到全部页发送完成。

• bootloader跳转应用命令

跳转命令　IDH, IDL, 0x55, 0x55, 0x01, 0x02

该命令为点对点命令，控制节点从bootloader跳转到应用程序。上位机依次对所有节点发送该命令，节点向上位机回复自身CAN-ID，升级成功节点会跳转到应用程序；不成功的节点保持在bootloader状态，等待重新升级；不升级的节点会跳转到应用程序。

具体实现

上述命令介绍基本说明了升级过程的流程动作，流程图如图2。在实现过程中，还需要注意以下几点：

• bin文件数据处理过程中，将文件内容按照节点Flash页大小，将固件程序分割成页大小，为每一页数据附加长度，起始地址，页号信息用于数据传输和校验，按照FLASH页大小分块传输，能够让嵌入式设备更高效的接收和存储数据；

• 升级程序可以选择需要升级的节点，跳转时对需要升级的节点采用升级跳转，对不升级的节点采用不升级跳转。节点需要建立bootloader和应用之间共享的存储区域，用于记录是否需要升级的标志，该功能在并行升级时，可以屏蔽不同功能的节点，对设备起到保护作

用，防止固件烧写到不同的设备中；

• 升级程序需要记录每一步操作节点的回复信息，用来显示具体步骤的成功和失败，以及节点号，成功的节点才会进行下一步操作，不成功的节点，上位机在后续步骤中，不再与该节点进行控制或查询交互，减少没必要的操作，例如节点断线、断电、程序异常、通讯异常等情况下，节点是不会有回复的；

• 凡是涉及烧写Flash的操作，接下来的查询命令都需要等待一段时间再发送，确保Flash烧写操作完成，等待时间根据具体情况调整；

• 任意步骤所有节点都没有回复，流程结束。

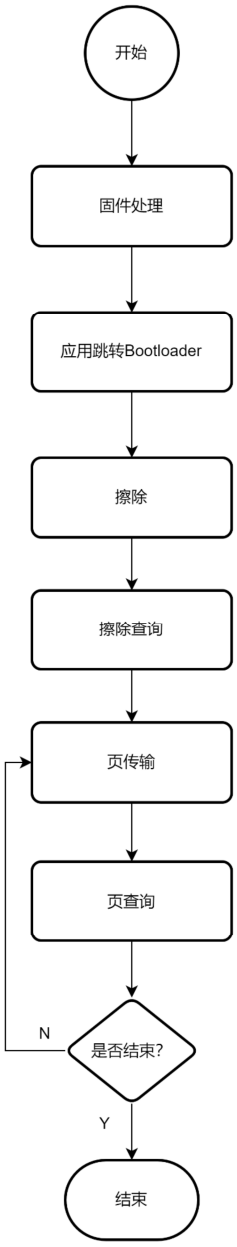

图2 升级流程图

结束语

本文设计了一种CAN总线多节点并行升级协议，利用CAN总线的广播特性，对传统的IAP进行了优化和改进，解决了传统IAP在多节点设备升级场景中的时长问题。经过实测，70KB的应用程序在单节点和多节点升级实验中，时间所差无几，完全符合协议设计原理。

协议可以应用于服务人员所携带的笔记本或便携设备，现场对设备进行固件升级，提升设备调试效率，提升客户服务效率；协议可以应用于现场上位机，用户通过操作上位机，使用U盘将固件导入到上位机中，通过上位机对设备进行固件升级，可以避免服务人员不必要的出差；协议可以应用于局域网或云服务器场景，通过CAN网关与CAN总线连接，实现远程固件升级。

考虑到安全问题，可以采用压缩和加密等方式来保护固件文件，在上位机和设备端需要增加解压和解密等算法来处理数据。

参考文献

[1]国际标准化组织。CAN总线国际标准。ISO11898-1.