嵌入式系统软件开发中的通信协议研究*)

袁 帅[1] 刘博勤[2] 陈章龙[1]

(复旦大学计算机科学与工程系微机实验室 上海200433)[1]

(西南师范大学计算机与信息科学学院计算机系 重庆400715)[2]

A Kind of Communication Protocol in Software Developing of Embedded System

YUAN Shuai[1] LIU Bo-Qin[2] CHEN Zhang-Long[1]

(Department of Computer Science&Engineering, Fudan University, Shanghai 200433)[1]

(Department of Computer & Information Science, Southwest China Normal University, Chongqing 400715)[2]

Abstract In the course of software developing of embedded system, we usually use the remote debugger, which needs connection and communication between the host and target through the serial port, parallel port or Ethernet. Thus some effective and reliable communication protocol is eagerly needed. The paper presents a new communication protocol called EDP (Embedded Debugger Protocol) in embedded software developing. This protocol is proven successful on Intel Assabet(SA-1110).

Keywords Communication protocol, Embedded system, Communication, EDP

1 引言

现今，嵌入式系统被广泛应用于各种设备中，大到车、船和卫星，小到家用电器，其应用领域日益扩大，因此，嵌入式系统上的软件开发和调试更显其重要。在嵌入式系统软件开发中，一般有3种程序调试方法。第一种是远程调试器，即通过宿主机和目标机之间的连接来下载、执行和调试嵌入式软件。前端是在宿主机上的一个基于文本或GUI(图形用户界面)的调试程序，另外还有一个运行在目标处理器上的隐藏的后端软件来负责通过某种通信链路与前端调试软件进行通信。后端一般被称作调试监控器(Debug Monitor)，它提供了对目标处理器的底层控制。第二种是在线仿真器(In-Circuit Emulator, ICE)。实际上，ICE自己就是一个嵌入式系统，它有自己的目标处理器、RAM、ROM和自己的嵌入式软件。因此在线仿真器一般非常昂贵，经常要比目标硬件还贵。第三种是模拟器。它是一个完全基于宿主机的程序，模拟了目标处理器的功能和指令集。到目前为止，模拟器最大的缺点是它仅能模拟处理器，而实际的嵌入式系统经常包含一个或更多重要的外围设备。

正因为在线仿真器价格昂贵而模拟器功能很不足之缺点，所以使用得较多的是第一种方法，即远程调试器的方法。在这种方法中，宿主机和目标机通过网络进行连接，且在嵌入式软件下载到目标机、进行调试、观察和运行时，两者之间要进行频繁的通信，为了保证通信的有效性、稳定性和容错性，必须有一套系统的通信规程，这就是我们要说的EDP (Embedded Debugger Protocol)协议。在目标机上驻留着一个调试监控程序，用来与宿主机通信完成程序下载和调试过程，而EDP协议，就是用来实现宿主机与目标机之间通信的一套协议规范。在Intel Assabet(SA-1110)目标机上经过实验证明，该协议在嵌入式系统软件开发中是可靠且高效的。

2 EDP协议内容

EDP协议用以在软件调试过程中在宿主机和目标机之间提供一种可靠连接。协议提供了从宿主机到目标机的全面而方便的相互访问。在EDP协议中，基本的通信单元就是包，在发送和接收的每个包中，至少有三层协议：数据提供层、信道层和设备层。

数据提供层协议是一种应用层的协议；信道层协议是传输层协议；设备层是数据链路层协议。图1显示了从宿主机端视角看到的通信过程。

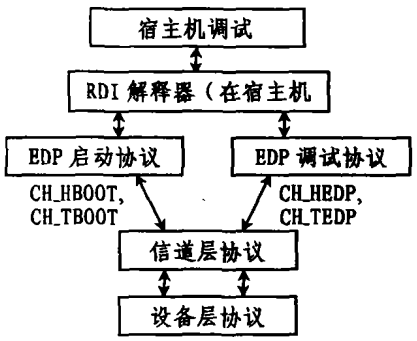

图1 宿主机端所见通信过程

2.1 数据提供层

数据提供层协议的数据包格式如图2所示。

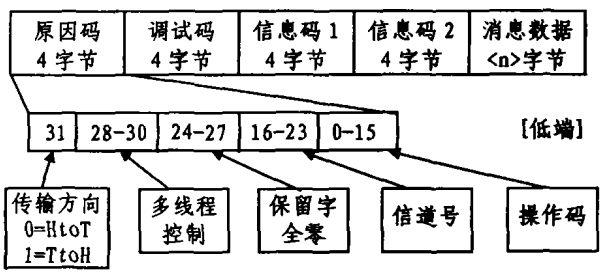

图2 数据提供层数据包格式

原因码：规定了接收者(通常是目标机)所要执行的操作，

*)本文所述工作应用于国家863项目"32位高速嵌入式CPU平台技术研究"(课题编号：2002AA1Z1009)的子系统"嵌入式集成开发环境"。袁帅 硕士研究生，主要从事嵌入式系统，嵌入式操作系统研究；刘博勤 副教授；陈章龙 教授，主要从事嵌入式系统研究。

原因码的首位用于显示该消息是由宿主机端始发的还是由目标机端始发的。由一方提出请求，另一方应答，两者的原因码其余位必须相同，而在方向位上相反。这个双字的16～23位是信道号，数据包在该信道进行传输。操作码是要求接收者进行的操作(例如读内存)。

调试码：是为了让目标机能使用宿主机软件而设置的一个字段，对于宿主机提出的请求，目标机的响应包含相同的调试码，对于无需该功能的系统(例如，单线程调试器)，它必须被设置为全1，由目标机始发的消息(例如，启动消息)也将该域设置为全1。

信息码1、信息码2：这两个字段用于目标机上有多进程操作系统时标志进程上下文信息。宿主机始发的消息和单进程目标机始发的消息，应该把这两个域设置为全1。

消息数据：根据原因码的不同，消息数据字段也有不同格式。原因码相当于指令中的操作码，而消息数据相当于指令中的操作数。操作数的数量和格式应当符合操作码的需求。

信道编号：在EDP协议中，信道号是预先定义好的，每一次通信服务都占用一个全双工信道。

2.2 信道层

信道层协议负责通过某信道将数据包从通信媒介的一端传输到另一端，信道在概念上相当于TCP/IP协议中端口的概念。信道层协议保证了数据提供层中的数据包能正确传输(如果出错，则信道层负责自动检错重发)，该协议基本上要求数据包到达的顺序与发送顺序一致，否则重排序将使效率有所降低。

信道层协议用于明确数据包在哪个信道中进行传输，该协议中包含着数据包序列号，协议头由4个字节组成，按照传输顺序依次是：信道号；宿主机发送序列号；目标机应答序列号；包类型号。数据包的格式如图3所示。

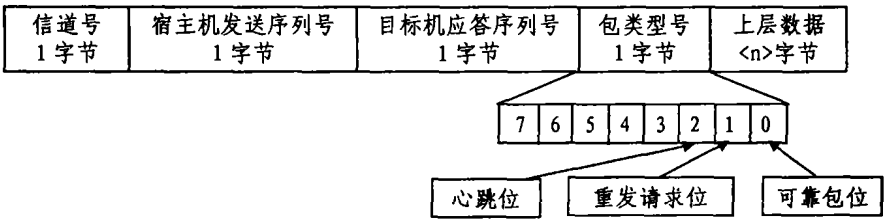

图3 信道层数据包格式

信道号：信道号是用于标志上层数据传输所使用的信道，在数据包接收到时需要检查它的正确性，它必须不超过预先设定的限制，否则即视为通信出错。

序列号：宿主机和应答序列号用来决定发送者和接收者之间的相对进程，比如判断发送者是否在接收者之前，这将决定接收者是否需要发送一个重发请求。序列号是全局性的，也就是说，不受限于信道号，也不受限于协议窗口大小。宿主机和目标机序列号在初始引导和应用程序初始化时被设置为0，每次递增1，在到达最大值255时复位至0。从单独的某个信道来看，此协议是一个简单的停止/等待的协议，这就是说，对于任何特定的信道，只有一个等待应答的数据包。

包类型号：它标志了数据包的类型，数据包有以下几种类型：

①重发请求包。不包含数据提供层的数据包，一个重发请求包将引起接收者重发一个或多个已经发送过但没有被正确接收的数据包，在重发请求包中的目标机应答序列号定义了一个重发序列的开始，重发序列的结束由“当前”的数据包所决定，在这个范围内的所有数据包将依次重发。接收者将这些重发数据包解释为普通数据包，并发回相应的应答。当最后的重发数据包被发送完，一个重发请求便完成了。

新的数据包序列号只会分配给那些可以重发的数据包，当前的序列号可以包含在别的数据包类型中(例如下面所述的心跳包)，但只有数据包是要求依次发送的时候，才会引起重发请求。

②心跳包。也不包含数据提供层的数据包，这是协议所规定的，心跳包存在的唯一目的是在长时间没有别的数据传输发生的情况下也能确定连接的目标机端是否“活着”。心跳包由宿主机始发，目标机在一定时间之内(例如30秒)给予应答，否则即视为目标机“死亡”。心跳包不会影响其他类型数据包中的序列号的变化。

③可靠数据包。包在刚发送出去之后仍然会被记录在后备队列中，直到对方收到此数据包并给出应答消息，才从后备队列中移除。如果接收到一个重发请求包，那么开始检查后备队列并找出需要被重发的数据包，并依次重发之。当对方给出应答时，数据包就从后备队列中移除。

2.3 设备层

信道层和设备层在接口上有一个特别的需求，就是在数据发送时设备层知道数据包中有多少数据要被传输，而在数据接收时能可靠地告诉上层(信道层)已经接收到了数据。

设备层数据包是用于传输信道层数据包的，在信道层数据包之上加上了包首、包尾、差错检测和字节转义。数据包格式如图4所示。

SOP	Type1	Length2	Data	CRC4	EOP1
1字节	字节	字节	(Length)字节	字节	字节

图4 设备层数据包格式

SOP(Start of Package)：数据包开始；

Type：数据类型，通常为0。如果为非0值表示接收到坏包；

Length：Data域的数据长度，单位为字节；

Data：信道层数据内容；

CRC：CRC32校验，采用IEEE802.3 32位CRC字节数据算法，对Type～Data之间的数据进行编码校验；

EOP(End of Package)：数据包结束。

为了避免一个包中的数据值被误当作控制字节值，在要传输的值之前插入一个转义字节(0x1C)，后面是数据值“或上”一个值0x40，这样一个值为0x11的数据值为避免被当作控制字节值就需要被转义为一对数据值0x1C 0x51。很明显，数据值0x1C也需要被转义，成为0x1C 0x5C。设备层数据包中除了SOP和EOP之外，其间的所有值都须被转义。

此协议可以大大改善连接的可靠性，对于以下几个错误类型能够检测到并作为坏包汇报给上层：

·数据包帧不完整。由 SOP 和 EOP 控制着数据包的完整性，不完整的数据包能由 SOP 或者 EOP 的缺失检测到。

·错误的数据包长度。Length 域中保存着 Data 域的数据长度，而其他域的长度是固定的，因此能由 Length 域中的值来判定数据包长度是否正确。

·错误的 CRC32校验。CRC32校验能够检测出丢失中间数据的包，使用的算法是 IEEE802.3 32位 CRC 字节数据算法。当一个数据包 CRC32校验失败的时候，在该包的 Type 域中将被置上"坏包"码并传输到上层（信道层）协议去。当接收到坏包的时候，信道层协议会要求当前等待的数据包重发一次。

3 通信效率

我们以串行通信（8位数据位，1位停止位，无奇偶校验）为例，评估 EDP 协议的传输效率，因为在程序代码从宿主机端下载到目标机端的过程是整个调试过程中数据传输量较大较为耗时的一部分，所以下面以这个过程为代表评估 EDP 协议的传输效率更有普遍性和客观性。

以下是一些预定义：

LoadSpeed：有效传输速率，即程序代码下载到目标机的速率（bytes/second）。

MaxData：数据提供层中消息数据的最大字节数（一次能传输的有效数据最大长度），在 EDP 协议中被定义为4k bytes。

EscapeRate：转义率，表示一个设备层数据包中出现控制码的概率。

Baud：串行口传输的波特率。

TargetDelay：目标机处理一个设备层数据包所需用的时间给通信造成的延迟（单位：秒）。

ErrorRate：包误码率，设备层中出现的错误包在所有数据包中的概率。

根据以上定义和 EDP 通信协议格式，我们得出的有效数据传输率（bytes/second）为：

$$LoadSpeed=\frac{MaxData-8}{(1+ErrorRate)\times[\frac{(MaxData+62)\times(1+EscapeRate)}{Baud/8}+T\arg etDelay]}$$

根据在 Intel Assabet（SA-1110）目标机上进行的实验，串行口波特率设置为115200，8位数据位，1位停止位，无奇偶校验，从宿主机下载一个大小为128k bytes 的程序源代码到目标机，耗时21.86秒，测试和计算得出的结果如表1所示。误差大约为 2.7%。

表1 实验测试和计算结果

Max Data	Error Rate	Escape Rate	Baud	Target Delay	Load Speed 理论值	Load Speed 实际值
4K bytes	0.31×10^{-8}	5.22%	115200	0.36 s	6158.27	5994.61

4 协议应用简单实例

现举例说明协议使用方法，该例子在目标机的 RAM（地址为0x00008765）写入一个数据值0x46。

宿主机提出写内存数据请求：

1. 在数据提供层

原因码字段设置为 MemWrite|CH_HEDP|HtoT，因为要进行的是内存写操作，使用的信道是 CH_HEDP，数据传输方向是从宿主机到目标机。

在单线程无需使用宿主机软件的目标机调试中，调试码字段，信息码1，信息码2字段都设置为0xFFFFFFFF

根据原因码，消息数据字段依次为：地址值（32位），写入数据的字节数 n（32位），写入的数据值（n 字节），在此例子中依次为：0x00008765，1，0x46。

2. 在信道层

信道号设置为 CH_HEDP。

宿主机发送序列号和目标机应答序列号分别填充当前宿主机发送序列号和当前目标机应答序列号。

发送序列号增1，同时作为可能的重发数据存储在后备列队中。

包类型号为0x01，表示是可靠数据包。

上层数据域的内容是数据提供层的数据包。

这样就形成了信道层数据包，接着发给下层（设备层）。

3. 在设备层

数据类型 Type = 0x00，表示无错误数据。

Data 域是信道层数据包的内容。

Length 字段是 Data 的长度（单位为字节）。

CRC 字段是根据 IEEE802.3 32位 CRC 字节数据算法求得的校验码，对 Type、Length 和 Data 进行正确性校验。

上述步骤完成之后，对数据进行转义过程，以避免和一些控制字节值相冲突。最后一步是在数据头尾各加上一个 SOP、EOP，形成设备层数据包，将数据包通过硬件发送到目标机去。

目标机应答：目标机接收到宿主机发送来的消息之后，进行校验和解码，若消息校验无误，目标机在 RAM 的 0x00008765地址写入数据0x46。然后给宿主机一个应答，应答格式和数据如下：

在数据提供层，原因码为 MemWrite|CH_HEDP|TtoH，其中操作码和信道号不变，而数据传输方向相反，这时候是从目标机到宿主机方向。而消息数据域为：状态码（32位，为 Edp_ok=0表示成功），因此在此例子中消息数据为：0x00。数据提供层其他各项与宿主机请求时的数据提供层相同。

在信道层和设备层，目标机应答格式与宿主机请求时相同层数据格式类似，而应答内容依上层数据作出变化。

展望 此协议最初的设计目的是为了在嵌入式系统的软件开发过程中保证宿主机和目标机之间通信的有效性、稳定性和容错性，协议采用多信道全双工通信方式，对波特率没有特定要求，因此通信速率仅受硬件环境限制。本协议不仅可用于串行通信，也可用于并行通信以及以太网通信等多种通信方式。协议的高、中、低三层分别承担给出指令、保证通信完整性和数据包正确性的责任，有很强的检错纠错能力，因此具有广泛的通用性和高可靠性，不仅可以用于嵌入式系统软件的开发过程，也可以用于其它诸多需要连接和通信过程的系统研发和应用。展望未来，相信 EDP 协议及其类似协议将越来越多地被人们普遍接受和广泛使用。

参 考 文 献

1 Michael Barr. Programming Embedded Systems in C and C++. O'Reilly & Associates, Inc

2 嵌入式系统——Intel StrongARM 结构与开发. 复旦大学计算机科学与工程系

3 Wright G R, Stevens W R. TCP/IP Illustrated, Volume 2: The Implementation

4 IEEE802.3以太网标准规范