第33卷 第18期 计 算 机 工 程 2007年9月
Vol.33 No.18 Computer Engineering September 2007
·软件技术与数据库· 文章编号：1000—3428(2007)18—0064—03 文献标识码：A 中图分类号：TP311

基于TETRA协议的SDL模型与C语言映射规则

汪 浩[1,2]，权进国[2]，林孝康[1,2]
(1. 清华大学电子工程系，北京 100084；2. 清华大学深圳研究生院，深圳 518055)

摘 要：介绍了SDL与C语言之间映射的意义，基于TETRA协议，从数据类型、信号传递、信号保存和进程调度4个方面分析了二者之间映射的若干规则。讨论了在实际应用中，提高映射效率须注意的问题。该文提出的映射规则提高了映射代码的效率，适用于自动映射工具不支持的特定操作系统。

关键词：陆上集群无线电协议；SDL；映射；信号

Mapping Rules of SDL Model into C Language Based on TETRA Protocol

WANG Hao[1,2], QUAN Jin-guo[2], LIN Xiao-kang[1,2]
(1. Department of Electronic Engineering, Tsinghua University, Beijing 100084;
2. Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055)

【Abstract】The significance of the study on mapping rules of SDL model into C language is introduced. According to TETRA protocol, this paper analyzes some mapping rules of the two languages, which is presented in four aspects: data type, signal transferring, signal reservation and process schedule. It discusses the issue about enhancing the efficiency of mapping in the practical application. The rules can enhance the efficiency of the generated code. It is suitable for the particular operation system that the automatic mapping tool can not employ.

【Key words】terrestrial trunked radio (TETRA) protocol; SDL; mapping; signal

陆上集群无线电(terrestrial trunked radio, TETRA)协议是欧洲电信标准协会(ETSI)制定的数字集群通信(digital trunked radio)标准。ETSI用SDL(specification and description language)模型描述了TETRA空中接口(air interface)物理层以上(不包括下层媒体接入控制层)的协议[1]，而在协议实现上，一般使用C语言。

Telelogic Tau公司提供的SDL Suite软件可以将SDL的描述和设计直接转换成标准的C代码，所生成的代码能够和多种嵌入式或非嵌入式实时多任务操作系统(RTOS)集成，如Solaris, Nuclues, pSOS等[2]。但是这种利用工具生成C代码的方法存在2个问题：

(1)用工具生成C代码往往效率(时间和空间上)很低，不能保证硬件资源的有效利用；

(2)这种方法生成的代码并不能和某些特殊的嵌入式操作系统集成，如uClinux等。

uClinux是一种开放源代码的、小核心的多任务嵌入式操作系统，有着良好的移植性。它沿袭了主流Linux绝大部分的特性，广泛应用于嵌入式系统中，如IP电话、PDA设备、通信终端等。对于TETRA移动终端的操作系统，本文选用了uClinux，同时，前面提到的方法不适合提高C代码的效率，因此，需要人工把SDL模型映射成高效的C代码。

1 SDL概述

SDL是原CCITT推荐的规范描述语言，经过ITU-T发展和标准化，定义见文献[3]。SDL一般用来描述实时多任务通信系统的行为。在整个协议软件的开发过程中，SDL处于系统规范描述阶段，因此，可以完全独立于硬件与操作系统。SDL所描述的系统有分明的等级结构，包括系统(system)、功能块(block)和进程(process)等。一个系统可以包含多个块，每个块可由一个或者多个进程组成，进程是并行执行的扩展有限状态机，它是SDL系统中的最小处理单元。当收到信号(或者称为消息)时，进程做出响应，状态发生迁移(也可能不迁移)，同时执行特定的操作(如向其他进程发送信号)。一个进程可能会调用多个过程(procedure)，所谓过程就是由进程调用的子程序。

2 SDL与C语言之间的映射规则

2.1 基本数据类型

SDL中的数据定义以预定义数据类为基础，通过数据类构造器来定义数据类型。

SDL中预定义的数据类型包括：Boolean(布尔型)，Character(字符型)，Charstring(字符串型)，Integer(整数型)，Natural(自然数型)，Real(实数型)，Pid(进程标识型)等。数据类型构造器有：Array(数组)，Literals(字面量)，Struct(结构)，Choice(选择)等。以上数据类型和C语言之间的映射规则如表1所示。其中，Integer, Natural和Real可以根据实际情况选择不同的数据类型。例如，TETRA协议的SDL模型中，TM_SDL_LengthType是Natural类型。它表示从逻辑链路控制层(LLC)发往媒体接入控制层(MAC)的服务数据单元(SDU)的长度(b)，协议里规定的最大长度是2 632b(十进制)，因此，笔者在C语言里把它映射为unsigned short int(16b)。

作者简介：汪 浩(1981－)，男，博士研究生，主研方向：集群通信，无线通信；权进国，高级工程师；林孝康，教授、博士生导师
收稿日期：2006-10-11 **E-mail：**hao-wang04@mails.tsinghua.edu.cn

表 1　数据类型映射规则

SDL 中的数据类型	C 中的数据类型
Boolean	enum(True 和 Flase 两个元素)
Character	char
Charstring	string
Integer	char/ short int/ long int(根据数据的范围选择)
Natural	unsigned char/ unsigned short int/ unsigned long int(根据数据的范围选择)
Real	float/ double(根据实际的精度要求选择)
Pid	unsigned char
Array	[ ](数组)
Literals	enum(枚举)
Struct	struct(结构体)
Choice	union(共用体)

2.2　信号传递

SDL 中的进程是基于信号触发机制的，一旦进程收到某个有效信号，就会做出相应的响应(状态迁移、向其他进程发送信号等)。因此，在 C 语言中需要定义和信号相关的数据结构以及处理函数。

在 TETRA 移动终端的协议实现上，本文把信号定义成如下的数据结构：

```
typedef struct
{
  Message_Type Msgtype;
  Message_Para Message_Body;
  ID next;
}Message;
```

每个信号都是一个结构体，它包括 3 部分：信号类型 Msgtype，信号内容 Message_Body 和指针 next。其中，信号内容表示信号所携带的信息，它在 C 语言中的数据类型是一个共用体。由于在进程处理信号的过程中，存在多个信号先后发往进程的情况，因此信号需要排队，并依次等待处理。next 就是用来指向信号队列中下一个信号的指针。

本文还定义了一些和信号相关的处理函数，如：AllocMessageID(在信号缓冲区给信号分配存储空间)，MsgFirst(返回队头信号的指针)，MsgQlast(在信号队尾加入新信号)，MsgTail(删除信号列头的信号)，MoveMsg(把源队头的信号插入到目的队列的头部)等。

信号传递包括信号的接收和信号的发送，基于 TETRA 协议的 SDL 模型，本文用 C 语言实现信号接收和发送的流程，见图 1。

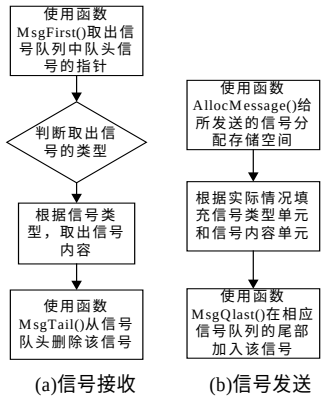

图 1　信号传递的处理流程

2.3　信号保存

保存是 SDL 中的一种对信号的特殊操作。一般情况下，进程中的状态机在某一状态下只接收一部分信号，如果到达的信号不属于这部分，就会直接被丢弃。有时，虽然信号在当前状态不能处理，但是在下一个状态是有效的，这时就需要把信号保存下来，等待下一状态处理，这就是保存的功能。

把 SDL 中的保存映射成 C 语言，本文采用的处理流程见图 2。Queue_Current 是当前进程的输入信号队列，Queue_Save 用来存储需要保存信号的队列。

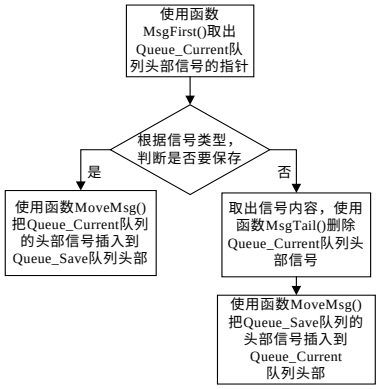

图 2　信号保存的处理流程

2.4　进程调度

SDL 模型中的不同进程是并发执行的，而 C 是一种基于顺序执行的高级语言。为了使得 C 语言产生并发执行的效果，可以采用下面的方法：(1)用顺序执行近似替代并发执行：用一个外部函数统一调度需要并发执行的函数，这些函数在外部函数中轮流顺序地执行。这种方法在实时性要求不高的场合，能够达到和并发执行相近的效果。(2)采用操作系统调度：把需要并发执行的函数分别作为操作系统的一个进程，由操作系统统一调度。(3)使用多处理器(CPU)：把需要并发执行的函数分别交给不同的处理器处理。显然这种方法的并发性能最好。

TETRA 移动终端一般对发送和接收的并发性要求较高，而对协议中层与层之间信号交互的实时性要求不高，另外 TETRA 移动终端一般使用的是单处理器。基于以上实际情况，本文采用了一种混合的并发执行方案，即：把 SDL 模型中进程映射成的 C 函数分成 2 个函数组，一个和发送操作相关，一个和接收操作相关，每个函数组都用(1)中提到的方法，分别在一个外部函数的调度下顺序执行。然后使用(2)中的方法，把这 2 个外部函数作为操作系统的 2 个进程，并发执行。

另一方面，在 TETRA 协议的 SDL 模型中，存在着某个进程创建和释放其他进程的情况。这里用一个具体的例子来说明进程创建和释放的映射规则。

TETRA 协议逻辑链路控制层中，Basic_Link 功能块的进程 Basic_Link_Acknowledged(表示有确认的基本链路)是由进程 Link_Control 创建的，并且最多可以创建 4 个这样的进程，表示同时可以有 4 条确认的基本链路存在。进程 Link_Control 和 Basic_Link_Acknowledged 分别被映射成 C 函数 BL_Link_Control()和 BL_Ack()。同时，本文在 C 语言中定义全局变量 BL_ACK_Flag，用来指示确认基本链路的创建情况。BL_Link_Control()函数在创建和释放进程时，都会修改 BL_ACK_Flag 的值；而外部函数(方法(1)中提到的用来调度的函数)则会根据此全局变量的值，顺序调用若干次 BL_Ack()函数，但是每次调用传递的参数不同，用来区分不同的确认基本链路。

3　提高映射效率应注意的问题

把 SDL 映射成 C 语言的过程中，效率问题需要重点关注，主要包括空间效率(程序大小)和时间效率(执行时间)。但空间效率和时间效率往往是一对矛盾，这需要根据具体的硬件资

源和程序实时性的要求在空间和时间上寻找一个折衷点。

(1)特殊数据类型的处理

在 TETRA 协议的 SDL 模型中，有的数据类型的长度是几个比特，如 AL_DISC_Type，它是逻辑链路控制层中用来拆除高级链路的协议数据单元(PDU)的数据类型，在 SDL 模型中它被定义为

```
NEWTYPE AL_DISC_Type
STRUCT
PDU_Type LLC_PDU_Type;
AdvancedLinkService LinkServiceType;
AdvancedLinkNumber LinkNumberType;
Report DisconnectionReportType;
ENDNEWTYPE;
```

其中，结构(struct)的 4 个成员的长度分别为 4b, 1b, 2b, 3b，如果把它们都映射成 unsigned char 类型，就需要 4B，比较浪费存储空间。这里使用了 C 语言中位域的结构，上面的 SDL 数据类型被映射为

```
typedef struct
{
unsigned short int   PDU_Type : 4 ;
unsigned short int   AdvancedLinkService : 1 ;
unsigned short int   AdvancedLinkNumber : 2 ;
unsigned short int   Report : 3 ;
}AL_DISC_Type;
```

这种实现方法，只占用 2B 的空间，提高了映射效率。

(2)远端变量的处理

SDL 中的远端变量(remote variable)是指可以提供不同进程共享的变量。SDL 对远端变量的操作分为出口(export)和进口(import)，在“服务器”进程中，通过出口操作把远端变量的值复制到一个隐含的变量中，其他进程通过进口操作可以得到该复制的值。

在 C 语言中，本文把远端变量映射成全局变量(global variable)，这样就避免了复杂的进口、出口操作，提高了代码的映射效率，而最终达到的效果是不变的。

(3)进程、过程之间的参数传递

在 TETRA 协议 SDL 模型的映射过程中，进程、过程之间的参数传递主要依靠全局变量。但程序中如果定义过多的全局变量，将导致程序的数据段空间增大。由于硬件的存储空间是有限的，因此需要把占用程序数据段空间较多的全局变量定义成全局变量指针，并给该指针动态分配存储空间。即：在需要使用某个变量的时候，使用存储分配函数(由操作系统提供)分配空间，对应的全局变量指针指向这段空间；使用结束后，用存储释放函数释放空间。

这种方法提高了空间利用率，但是会增加一定的时间开销(即分配空间和释放空间的时间开销)。因此，在实际应用中就要根据具体情况在时间和空间上作权衡。

4 结束语

SDL 是一种标准化的协议描述语言，而 C 是协议实现上广泛采用的语言。本文基于 TETRA 协议，研究了这 2 种语言的映射规则，并总结了在实际应用中，提高映射效率应该注意的问题。本文所讨论的内容，对通信协议的特定实现(如使用转换工具不支持的嵌入式操作系统、提高映射效率)有借鉴意义。

参考文献

1 ETSI ETS 300 392-13-1997 Terrestrial Trunked Radio (TETRA), Voice Plus Data (V+D), Part 13: SDL Model of Air Interface[S]. 1997.

2 宋茂强. 通信软件设计基础[M]. 北京: 北京邮电大学出版社, 2001.

3 ITU-T Z 100-1999 Specification and Description Language (SDL)[S]. 1999.

4 ETSI EN 300 392-2 V2.5.2-2005 Terrestrial Trunked Radio (TETRA), Voice Plus Data (V+D), Part 2: Air Interface (AI) [S]. 2005.

(上接第 63 页)

4.2 性能分析

在 XML 文档元素总数一定的情况下，当$|A|=|D|=n$ 时，循环嵌套算法的时间性能最差，为 $O(n^2)$；当$|A|=m, |D|=n$ 时，循环嵌套算法的时间性能为 $O(m \cdot n)$。

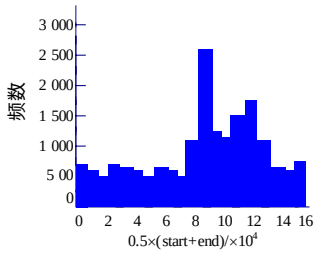

图 3 元素编码期望分布

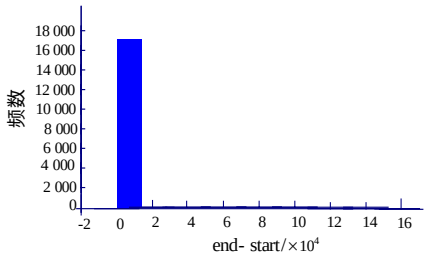

图 4 编码长度分布

二分法可以有效解决这个问题，假设第 i 次划分后递归结束，则$|A1|=|A2|=\ldots|A\,2^i|=m/\,2^i$，$|D1|=|D2|=\ldots|D\,2^i|=n/\,2^i$，此时性能为$=O(mn/\,2^i)$。因此划分次数 i 的取值非常关键，i 过小表示元素的编码分布比较集中，时间性能无法得到明显改进，i 过大表示元素编码的长度(end-start)几乎覆盖了整个编码区间，时间性能虽然得到明显改进，但是划分带来的代价也是无法接受的。

通过对大量实验文档的分析表明，元素的编码分布是比较松散和平均的，见图 3。而元素编码的长度几乎覆盖了整个编码区间的情况很少出现，见图 4。每次划分都可以得到一个相对合适的 i 值，所以二分法有一定的利用价值。

5 结语

本文针对路径表达式计算中的核心操作——结构连接问题的处理进行了初步研究，在区域编码基础上，提出了基于二分法的连接算法，该算法不要求数据有序或者索引的存在。实验结果显示，该算法在输入数据无序的情况下优于循环嵌套算法，而且在各种数据分布下具有良好的稳定性。

参考文献

1 Zhang C, Naughton J, Dewitt D, et al. On Supporting Containment Queries in Relational Database Management Systems[C]//Timos S. Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2001: 425.

2 Chien S Y, Vagena Z, Zhang D H, et al. Efficient Structural Joins on Index XML Document[C]//Bernstein P A. Proc. of the 28th Int'l Conf. on Very Large Databases. San Francisco: Morgan Kaufmann Publishers, 2002: 263-274.

3 Wang W, Jiang H F, Lu H J, et al. Pbitree Coding and Efficient Processing of Containment Joins[C]//Dayal U, Ramamritham K, Vijayaraman T M. Proc. of the 19th Int'l Conf. on Data Engineering. Los Alamitos: IEEE Press, 2003: 391-402.

4 王 静, 孟小峰, 王 珊. 基于区域划分的结构连接[J]. 软件学报, 2004, 15(5): 720-728.