CANopen 对象字典的分析与具体实现

金超，夏继强，满庆丰

（北京航空航天大学 机械工程及自动化学院，北京 100191）

摘要：CANopen 是基于 CAN 总线的开放的、标准化的应用层协议，对象字典是其核心概念。本文在对 CANopen 对象字典进行介绍与分析后，给出了一种数组形式的对象字典具体实现方法；对不同保存属性对象的存储策略进行了讨论，并给出了对象字典的存储与读写访问方法。

关键词：CANopen；对象字典；静态数组

中图分类号：TP336　　**文献标识码**：A

Analysis and Concrete Implementation of CANopen Object Dictionary

Jin Chao, Xia Jiqiang, Man Qingfeng

(Department of Mechanical Engineering and Automation, Beihang University, Beijing 100191, China)

Abstract: CANopen is an open and standardized application layer protocol based on CAN bus, and object dictionary is the core concept. This paper firstly introduces and analyzes CANopen object dictionary, then realizes an object dictionary in array form. Lastly different attributes of object storage strategies are discussed, and storage and read/write access methods of object dictionary are given.

Key words: CANopen; object dictionary; Static arrays

引　言

CANopen 协议是在 CAN 总线的基础上定义的应用层协议，是具有高度灵活配置能力的标准化嵌入式网络协议。因其协议精练、透明、便于理解，又具有实时性和可靠性高、数据传输速率高、组网成本低等优点，在多个领域中得到了广泛的应用。

对象字典是 CANopen 协议的重要组成部分，CANopen 网络中每个节点都有一个对象字典。对象字典包含了描述这个设备和它的网络行为的所有参数，以标准化的方式描述了 CANopen 设备，是通信网络与应用程序之间的接口。

在 CANopen 协议栈的实现中，对象字典的构建与访问是其中的重点与难点。良好的对象字典实现是 CANopen 协议栈高效稳定运行的基础。但在 CANopen 的协议规范中，只有 eds 与 dcf 文件用来描述对象字典的构成与具体的对象数据，而对具体程序中的对象字典实现没有规范。目前国内学者对 CANopen 协议的研究和使用中，较少涉及到对象字典的具体实现方法。

本文的对象字典实现采用了静态数组的形式，在国外开源协议栈 CanFestival 的对象字典实现基础上，增加了对象字典的存储方法；并分析了不同对象的存储与访问策略，给出了对象字典的具体实现方法。

1　对象字典的协议规范分析

1.1　对象字典的概念

对象字典（Object Dictionary，OD）是一个有序的对象组，用来提供对设备所有重要数据、参数和功能的访问（无论远程或本地）。这种访问是基于一种索引和子索引的逻辑地址体系实现的。对于每一个 CANopen 设备，对象字典结构都是相同的，这为通信参数、制造商定义对象和设备对象提供了一套标准化的地址空间。

1.2　对象字典中对象的结构

对象字典中的元素，即各种通信对象和应用对象，使用标准的方式描述，对象的结构如下所示：

索引	对象类型	对象名称	数据类型	访问属性	必选/可选

可以看出，描述对象字典中对象的结构有 6 个属性。“索引”表示对象在对象字典中的位置；“对象类型”的定义如表 1 所列，在对象字典中，“VAR”、“ARRAY”和“RECORD”为常用的对象类型，子索引主要用来访问“ARRAY”和“RECORD”中的子项，“VAR”类型的子索引为 0；

"对象名称"以简单的文字描述了对象的功能;"数据类型"给出了对象所属的数据的类型;"访问属性"给出了对象的被访问权限;"必选/可选"定义了对象在对象字典中的实现要求。

表 1 对象类型的定义

对象类型	注 释
NULL	没有数据
DOMAIN	具有大量变量的数据,例如可执行程序代码
DEFTYPE	类型的定义,如 Boolean、UNSIGNED16 等
DEFSTRUCT	定义新的 RECORD 类型
VAR	一个单一值
ARRAY	数组类型的数据
RECORD	由多种单一数据组成的结构类型

1.3 对象字典的布局

对象字典的布局如表 2 所列。其中,0x0001～0x009F 是对数据类型的定义,不必在程序中具体实现。0x1000～0x1FFF 是通信子协议区域,描述设备在 CANopen 网络中通信及交换数据所必须具备的基本功能。其内容包括设备类型、制造商信息、PDO 通信参数及映射参数、SDO 服务器及客户端的参数等。这个范围内定义的通信参数适用于所有的 CANopen 设备。0x2000～0x5FFF 是预留给制造商定义的区域,用来描述在标准设备子协议中没有规定的设备对象。0x6000～0x9FFF 为标准设备子协议区域,描述了如 I/O 设备、传感器、驱动和运动控制等设备接入 CANopen 网络的标准化规范。目前,CANopen 标准设备子协议还在不断丰富和完善中。

表 2 对象字典布局

索引(hex)	对 象
0000	保留
0001～001F	基本数据类型
0020～003F	复杂数据类型
0040～005F	制造商规定的数据类型
0060～007F	设备子协议的基本数据类型
0080～009F	设备子协议的复杂数据类型
00A0～0FFF	保留
1000～1FFF	通信子协议区域
2000～5FFF	制造商特定子协议区域
6000～9FFF	标准设备子协议区域
A000～BFFF	标准接口子协议区域
C000～FFFF	保留

具体的 CANopen 节点设备开发中,只需实现自己应用相关的对象字典条目,而不必实现全部内容。即便是在通信子协议区域,强制实现的对象也仅有几个。

2 对象字典的具体实现

对象字典是 CANopen 设备模型的核心,是连接通信与应用的接口。但在具体开发中,对象字典的实现有很大的自由度。对于一些简单的节点设备,甚至不需要在程序中实现对象字典的实体,只把对象字典的概念留在逻辑层面。良好的对象字典的实现,有利于发挥 CANopen 协议便于配置、灵活高效的优势,有利于节点设备的开发与升级。

对象字典的逻辑结构为线性结构,若以数组形式建立完整的对象字典表格,则可以通过索引快速地访问对象条目。但在具体的节点设备中建立完整的数据字典表格是不太现实的,因为一般嵌入式系统存储资源有限,而且具体的 CANopen 设备也只用到有限的对象字典条目。本文参考了国外开源协议栈 CanFestival 的对象字典实现思想,即将使用的对象字典条目按照静态数组结构存储,再建立一个索引表,通过查找索引表定位到要访问的对象字典条目。

对于 CANopen 从站设备,对象字典结构在运行前已确定,运行时无需动态改变对象字典结构,所以采用静态数组结构的对象字典实现可以满足大多数 CANopen 应用开发的要求。

2.1 对象字典的建立

对象字典的实体采用静态数组加对象条目索引表的方式,描述语言为 C 语言。首先给出相关结构体的定义。

对象字典中一个子索引对应的数据条目的结构体定义如下:

```
typedef struct s_ODEntry{
    uint8_t    AccessType;
    uint8_t    DataType;
    uint32_t   size;
    storeAttribute * storeAttri;
    void * pObject;
}ODEntry;
```

其中 AccessType 为对象的访问权限;DataType 为对象条目的数据类型;size 为数据大小;storeAttri 为对象条目的存储属性;pObject 为指向对象数据的指针。本文所使用的外部非易失性存储器为 AT45DB 系列 Flash 存储器,存储属性有 4 个,分别为对象存储类型、外部 Flash 的页地址、页内偏移地址和存储空间大小。

对象字典中一个对象索引条目的结构体定义如下:

```
typedef struct s_indexTable{
    ODEntry * pEntry;
```

```
    uint8_t    subCount;
    uint16_t   Index;
} indextable;
```

其中 pEntry 指向数据字典条目；subCount 为子索引个数；Index 为条目的索引号。

利用“ODEntry”类型可以构建索引对象的条目，以建立索引号为 0x1000 的设备类型对象为例，如下所示：

```
/* index 0x1000 : 设备类型 */
const uint32_t DeviceType = 0x06000000;
storeAttribute store_obj1000 = {0, 0, 0, 0};
ODEntry ObjDict_Index1000[] ={
    {RO, uint32, sizeof(uint32_t), &store_obj1000, (void *)
     &DeviceType}
};
```

利用“indextable”类型可以构建整个对象字典的数组形式的实体，如下所示：

```
indextable ObjDict_objdict[] ={
    { ObjDict_Index1000, 1, 0x1000}, //0
    { ObjDict_Index1001, 1, 0x1001}, //1
    { ObjDict_Index1008, 1, 0x1008}, //2
    ......
};
```

然后建立一个对象字典条目索引和在对象字典实体中实际存储位置相对应的索引表。通过条目索引查找整个索引表，根据实际的存储位置，定位到要访问的对象条目。

2.2 对象字典的存储与访问

上一小节中构建了对象字典的数组存储结构。对象字典条目所属的数据对象有着不同的存储与访问要求，读写属性上有只读、只写和可读写 3 种类型，存储属性上分为掉电丢失和掉电保存 2 种类型。

在上述定义的对象字典对象条目的结构体中，有一项为数据保存属性，具体定义数据的存储类型以及在外部非易失性存储器中的存储位置及存储空间的大小。存储类型指明了对象字典条目的保存类别，存储位置则只对需要掉电丢失的对象数据起作用，表明了数据在非易失性存储器中的存储位置。

下面给出不同存储类型对象的存储方法：

① 只读属性的数据，如设备类型、制造商标识等对象数据，变量直接定义为 const 类型，随程序存储在片内 Flash 区域。

② 不需要掉电保存的过程数据，以普通变量的形式存储在 RAM 中。

③ 简单的系统配置数据，如 SDO 客户端的通信参数，存储于片外 Flash 中，并在 RAM 中建立变量，便于应用程序与协议程序的使用。这类变量在系统初次运行时将默认的参数载入到片外 Flash 中，以后每次上电从片外 Flash 中读取，覆盖掉 RAM 中的初始默认值。

④ 复杂的系统配置数据。可以直接存储到片外 Flash 中，不在 RAM 中建立相应变量。应用程序可直接从片外 Flash 中读取相应配置数据。

对象字典的访问通过建立一个读取与写入对象字典接口进行。首先根据索引定位到要访问的对象条目，再根据数据类型与访问属性等信息对对象数据进行读写访问。但不是所有的数据访问都要经过这个接口，在应用程序端，对象字典数据的访问可以直接通过数据对象所指向的内存中的变量进行，或者直接读写对象在外部存储器中的数据。在通信端，对象字典的访问主要通过服务数据对象(SDO)进行。其他网络节点通过被访问节点的 SDO 服务器读写被访问节点的对象字典。SDO 的数据传输方式有 3 种：加速传输(用于传输少于 4 个字节的数据)、分段传输(用于传输大于 4 个字节的数据)和分块传输(比分段传输更高的传输效率)。对传输少量数据的加速传输和分段传输来说，可以建立通信数据的缓冲区，这样就只需访问对象字典一次；而对分块传输来说，因为数据量较大，不能在 RAM 中缓冲所有块的数据，所以块传送需要多次访问对象字典。

结 语

本文阐明了对象字典的概念，采用静态数组形式的对象字典的实现方法可满足大多数 CANopen 设备的应用开发的要求。讨论了对象字典存储与访问方法，对 CANopen 协议的开发使用有一定参考意义。

参考文献

[1] CiA. CiA Draft Standard 301 CANopen Application Layer and Communication Profile, 2002.

[2] 蔡豪格. 现场总线 CANopen 设计与应用[M]. 周立功，黄晓清，译. 北京：北京航空航天大学出版社，2011：62－64.

[3] 徐喆，闫士珍，宋威. 基于散列表的 CANopen 对象字典的设计[J]. 计算机工程，2009，35(8)：44－46.

[4] CANopen 开发中的常见问题[EB/OL]. (2009－10－04)[2012－04－10]. http://www.picavr.com/news/2009－10/1439.htm.

金超(硕士研究生)，主要研究领域为现场总线技术与嵌入式系统；
夏继强(副教授)，主要研究领域为工业测控网络及现场总线技术；
满庆丰(教授)，主要研究领域为工业测控网络及嵌入式系统。

(责任编辑：梅栗芳　收稿日期：2012-04-13)