Technical Note

CANopen 轻松入门

入门教程

TN01010101 V1.00 Date:2015/10/01 工程技术笔记

类别	内容
关键词	CANopen DS301
摘　要	用于初次接触 CANopen 的用户，主要以 CANopen DS301 为主要教授方向

广州致远电子股份有限公司

修订历史

版本	日期	原因
V1.00	2015/10/01	创建文档

目 录

1. 前言

本教程适用于 CIA CANopen 协议 DS301 又名 CIA301 标准。**用户须已经掌握 CAN2.0A 协议的基本知识。即基本的帧结构、ID、数据、DLC 等知识，本文不再从 CAN 底层开始叙述**。如果读者需要了解 CAN 底层，推荐北京航空航天出版社出版的《项目驱动——CAN-bus 现场总线基础教程》。

本文由广州致远电子股份有限公司周立功、黄敏思等整理和编撰。文章引用 CANopen 协会 CiA 组织的蔡豪格主席肖像与多篇示意图，再次表示非常感谢！

2. CANopen 在 ISO 层级中的位置

从 OSI 的 7 层网络模型的角度来看同，CAN（Controller Area Network）现场总线仅仅定义了第 1 层（物理层，见 ISO11898-2 标准）、第 2 层（数据链路层，见 ISO11898-1 标准）；而在实际设计中，这两层完全由硬件实现，设计人员无需再为此开发相关软件（Software）或固件（Firmware），只要了解如何调用相关的接口和寄存器，即可完成对 CAN 的控制。如图 2.1 所示。

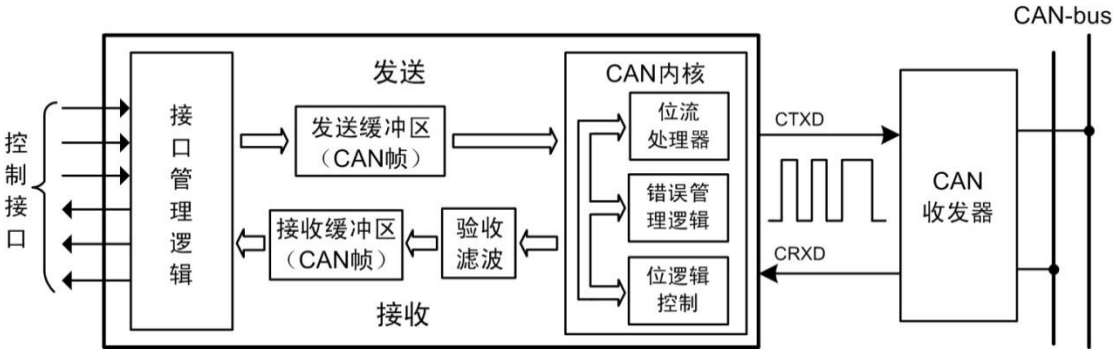


图 2.1 CAN 控制器结构

但 CAN 没有规定应用层。也就是没有规定与实际应用相关的逻辑，比如开关量输入输出，模拟量输入输出。所以本身对于应用来说，是不完整的。这就像铁矿石（物理层）冶炼成铁锭（数据链路层），然后针对具体应用，再加工做成汽车、轮船、钢筋、坦克、钢结构建筑等等。如图 2.2 所示。

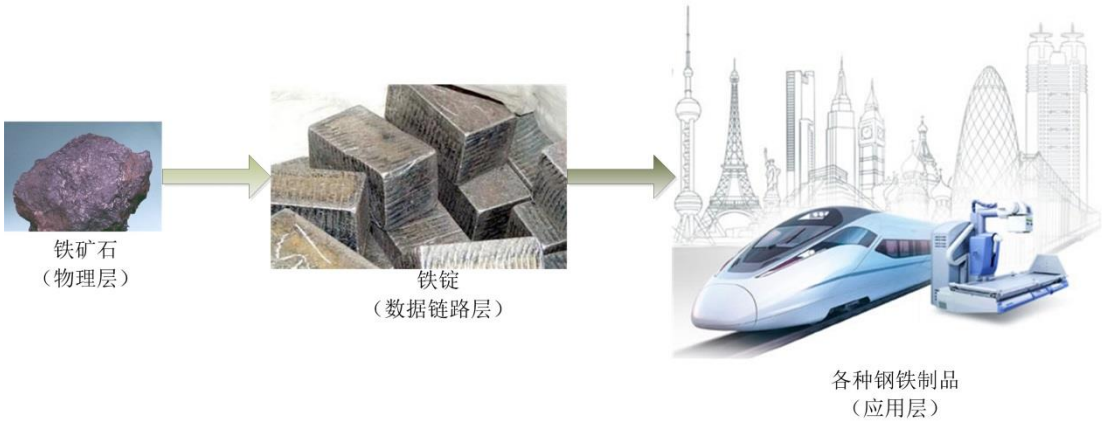


图 2.2 从物理层到应用层

因此，基本每个行业的 CAN 应用，都需要一个高层协议来定义 CAN 报文中的 11/29 位标识符、8 字节数据的使用。但在 CAN 总线的工业自动化应用中，由于设备的互通互联的需求越来越多，所以需要一个开放的、标准化的高层协议：这个协议支持各种 CAN 厂商设备的互用性、互换性，能够实现在 CAN 网络中提供标准的、统一的系统通讯模式，提供设备功能描述方式，执行网络管理功能。其中包括：

- 应用层(Application layer)：为网络中每一个有效设备都能够提供一组有用的服务与协议。
- 通讯描述(Communication profile)：提供配置设备、通讯数据的含义，定义数据通讯方式。
- 设备描述(Device proflile)：为设备（类）增加符合规范的行为。

3. CANopen 协议诞生和发展

CANopen 协议是在 20 世纪 90 年代末，由总部位于德国纽伦堡的 CiA 组织——CAN-in-Automation，（http://www.can-cia.org ）在 CAL（CAN Application Layer）的基础上发展而来。

图 3.1 CANopen 与 CiA

由于 CANopen 协议的创始人团队也是 CAN-bus 的创始人团队，此协议充分发挥了 CAN-bus 所具备的所有优势，特别是 CiA 组织的主席蔡豪格（Holger Zeltwanger）先生对于 CANopen 协议坚持开放、免费、非盈利的原则。一经推出便在欧洲得到了广泛的认可与应用。虽然 CiA 组织背后没有强大的财阀支撑，但时至今日已经成为全世界最为流行的 CAN 应用层协议。让我们记住这位可爱的德国老人，如图 3.2 所示。

图 3.2 CiA 组织蔡豪格主席

经过对 CANopen 协议规范文本的多次修改，使得 CANopen 协议的稳定性、实时性、抗干扰性都得到了进一步的提高。并且 CiA 在 CANopen 基础协议——CiA 301 之上，对各个行业不断推出设备子协议，使 CANopen 协议在各个行业得到更快的发展与推广。所谓的**子协议**，就是**针对不同行业的应用对象，对 CANopen 内部的数据含义进行重新定义，或者添加新的控制逻辑**。

目前 CANopen 协议已经在运动控制、车辆工业、轨道交通、电机驱动、工程机械、船舶海运等行业得到广泛的应用。比如轨道交通中的城市轻轨车辆（低地板车）中，CiA 联合西门子、庞巴迪等轨道交通厂商，共同制定了以下轨道交通相关的 CANopen 子协议：

- CiA 421 series: Train vehicle control system 列车车辆控制系统
- CiA 423 series: Diesel engine control system 柴油机控制系统
- CiA 424 series: Door control system 门控制系统
- CiA 426 series: Exterior light control system 外部灯控制系统
- CiA 430 series: Auxiliary equipment control system 辅助设备控制系统

◆ CiA 433 series: Interior light control system 内部灯控制系统

如图 3.3 所示，为 CANopen 在轨道列车中的地位，主干网为列车总线（WTB），每接车厢采用车辆总线（MVB）来连接与列车行驶相关，对实时性要求高的部件。而 CANopen 主要是连接各种非高安全性的部件。

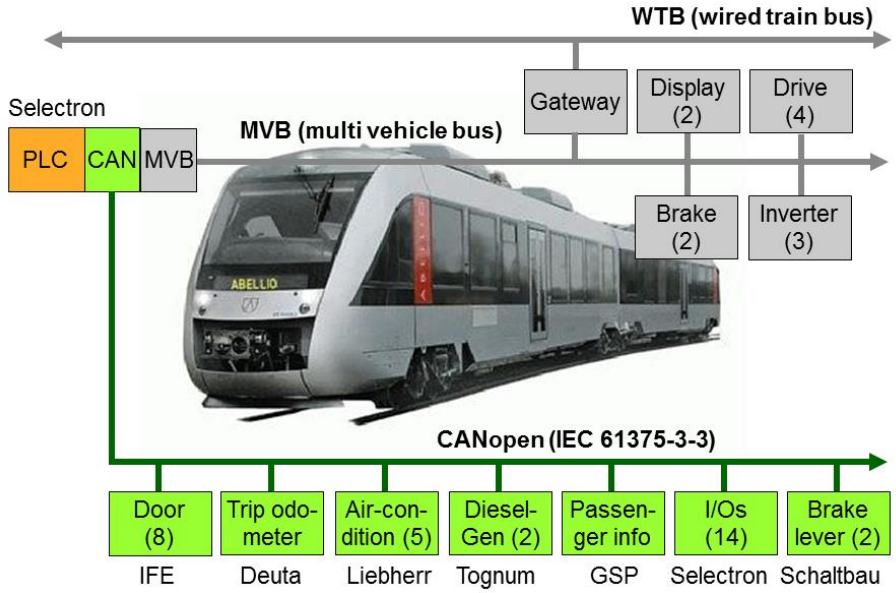


图 3.3 CANopen 在轨道交通中的应用

图 3.4 所示为 CANopen 设备结构。CANopen 协议通常分为用户应用层、对象字典以及通信三个部分。

其中**最为核心的是对象字典**，描述了应用对象和 CANopen 报文之间的关系。

CANopen 通信是本文关键部分，其定义了 CANopen 协议通信规则以及与 CAN 控制器驱动之间对应关系，熟悉这部分对全面掌握 CANopen 协议至关重要。

用户应用层是用户根据实际的需求编写的应用对象，这部分本入门教程将不作详细。

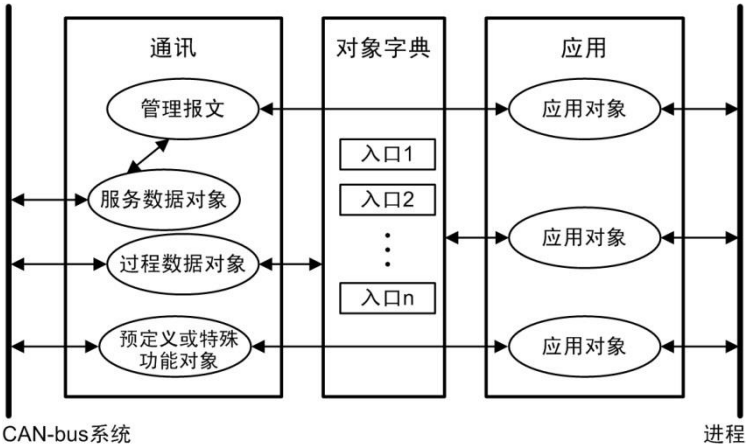


图 3.4 CANopen 设备结构

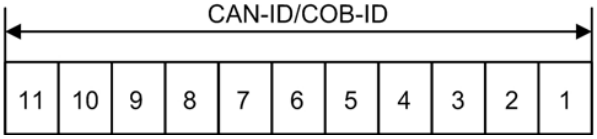
4. CANopen 的预定义报文 ID 分类

在 CANopen 创立之初，即使在 CAN 总线应用最广泛的汽车电子行业，网络中的 CAN 节点数量和需要通讯的信息都是比较少的。人们使用 CAN 取代 RS485，**主要是看重其可以突发发送的实时性优势**，而在多节点、长距离应用中，CAN 总线和 RS485 比起来并无优势，比如同样的波特率下，CAN 的通信距离只能达到 RS485 的 0.6-0.8 倍，而多节点通信 CAN 无法进行任意的突发发送，不得不遵循 RS485 那样的轮询通信机制，否则会导致拥堵，如图 4.1 所示。就像这个十字路口的汽车，如果车只有 10 辆，即使没有交通灯，根本不会拥堵。而如果有 100 辆，如果任意行驶，就会发生严重拥堵。

图 4.1 CAN 的突发优势和多节点拥堵

CANopen 的创始人是非常了解 CAN 总线这个特征，所以在设计 CANopen 时，对其定义为**小网络、控制信号**的**实时通讯**：

- 报文传输采用 **CAN 标准帧格式**。即 11bit 的 ID 域，以尽量减小传输时间；



- 网络控制报文均采用数据最小字节数。比如心跳报文，只有 1 个字节数据；
- 实时更新的**过程数据无需接收方报文应答**。即采用生产消费模型，降低总线负载；
- 需要接收方确认的配置参数一般都是采用快速单字传输。即 1 个报文最多传送 1 个 32 字节的参数变量，避免了分帧引起的实时性降低。

以上这些定义都是为了节约时间开销，最大限度保证实时性。同时为了减小简单网络的组态工作量，CANopen 定义了强制性的缺省标识符（CAN 帧 ID）分配表，以减少使用者与维护者的学习时间，快速上手。

4.1 网络管理（NMT）与特殊协议（Special protocols）报文 ID 分类

虽然 CANopen 的通讯发挥了 CAN 的特色，所有节点通信地位平等，运行时允许自行发送报文，但 CANopen 网络为了稳定可靠可控，都需要设置一个**网络管理主机** NMT-Master（Network Management-Master），就像一个交响乐团的指挥家，所有节点的启动、停止都是有他进行指挥，如图 4.2 所示。

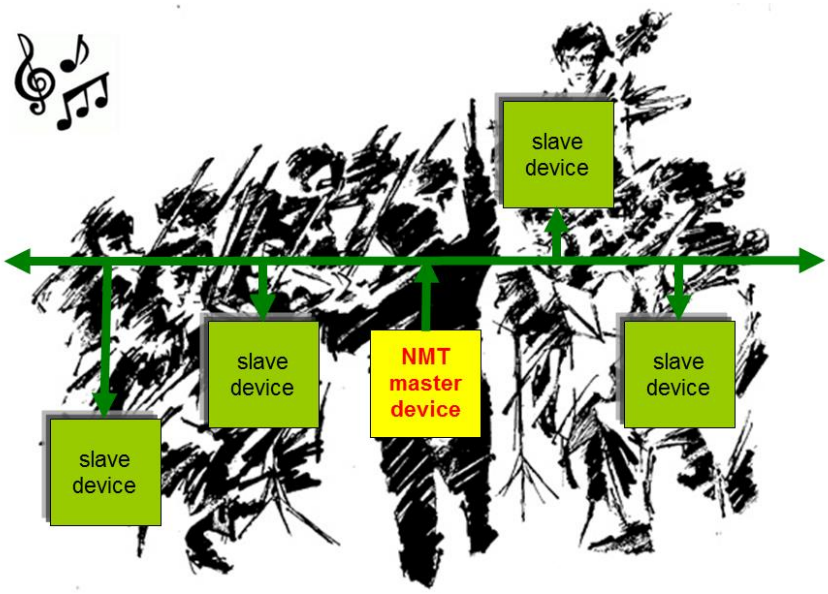


图 4.2 NMT-Master 就像交响乐指挥家

NMT 主机一般是 CANopen 网络中具备监控的 PLC 或者 PC（当然也可以是一般的功能节点），所以也成为**CANopen 主站**。相对应的其他 CANopen 节点就是 NMT 从机(NMT-slaves)。

NMT 主机和 NMT 从机之间通讯的报文就称为 **NMT 网络管理报文**。管理报文负责层管理、网络管理和 ID 分配服务。例如，初始化、配置和网络管理（其中包括节点保护）。网络管理中，同一个网络中只允许有一个主节点、一个或多个从节点，并遵循主从模式。

另外，为了**协调各个节点的同步、心跳、时间、错误提示**等通讯控制，CANopen 还定义了一系列**特殊协议**（Special protocols）报文。如表 4.1 所示，为 CANopen 预定义报文（Pre-defined CAN-IDs）的 NMT 报文和特殊协议报文。

表 4.1 NMT 与特殊协议的 CAN-ID 定义

Object 对象	Specification 规范	CAN-ID
NMT 网络管理命令	CiA301	000_h
Global failsafe command 全局故障安全命令	CiA304	001_h
Flying master 动态主站	CiA302-2	071_h to 076_h
Indicate active interface 标示活动接口	CiA302-6	$07F_h$
Sync 同步报文	CiA301	080_h
Emergency 紧急报文	CiA301	081_h to $0FF_h$（080_h +node-ID）
Time stamp 时间戳报文	CiA301	100_h
Safety-relevant data objects 安全相关数据对象	CiA301	101_h to 180_h

CAN-ID 就是这类报文的 COB-ID，其中**读者必须需要记住的是绿色底纹的这些常用的 CAN-ID 含义**，在研发和应用 CANopen 中，这三类是最为常用的 NMT 与特殊协议报文。

4.2 过程数据对象（PDO）和服务数据对象（SDO）的报文 ID 分类

用户应用 CANopen 时，需要传递的配置信息和应用信息都是放在**过程数据对象 PDO（Process data object）**和**服务数据对象 SDO (Service data object)**里面。这些对象就和市场上卖水果的箩筐，大小是一样的，只是装的东西（应用数据）不一样，如图 4.3 所示。这就是 CiA301 协议所规定的基础协议——“箩筐”，而 CiA4xx 的子协议或者用户自定义的对象就是“箩筐”里面的东西。

图 4.3 PDO 和 SDO 就像水果箩筐

PDO 和 SDO 的通讯区别在于，**PDO 属于过程数据，即单向传输，无需接收节点回应 CAN 报文来确认，从通讯术语上来说是属于“生产消费”模型**。如图 4.4 所示。

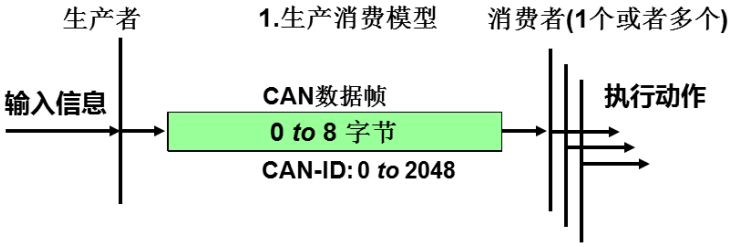


图 4.4 生产消费模型

而 **SDO 属于服务数据，有指定被接收节点的地址（Node-ID），并且需要指定的接收节点回应 CAN 报文来确认已经接收，如果超时没有确认，则发送节点将会重新发送原报文。这种通讯方式属于常见的“服务器客户端”的通信模型**，即我们通常所说的轮询式。如图 4.5 所示。

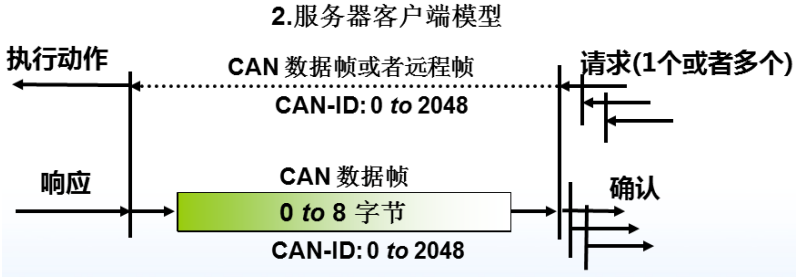


图 4.5 服务器客户端模型

对于 PDO 和 SDO 的报文 ID 分配，为了减少网络的组态工作量，CANopen 预定义了强制性的缺省标识符（CAN-ID）分配表，该分配表是基于 11 位 CAN-ID 的标准帧格式。**将其划分为 4 位的功能码（Function-ID）和 7 位的节点号（Node-ID）**。如图 4.6 所示.

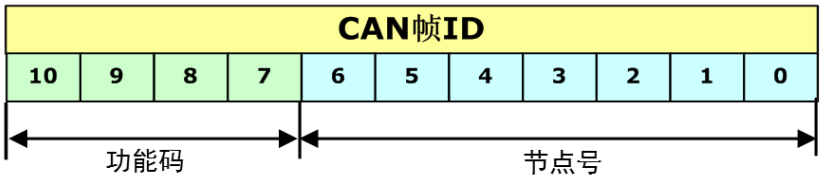


图 4.6 PDO 和 SDO 的预定义连接 ID 分配

在 CANopen 里也通常把 CAN-ID 称为 COB-ID（通信对象编号）。所以**我们可以分清楚两个易于混淆的名称：**

- COB-ID：Communication Object Identifier,即 CANopen 中对某种通讯对象的报文帧 ID，即 CAN 报文的 11 位 ID。代表了一种通讯含义。
- Node-ID：节点 ID 号，即 CANopen 网络中的节点地址，CANopen 规定了逻辑上最大 128 个节点，所以 Node-ID 最大为 128（7 位）。

COB-ID 和 Node-ID 无必然联系，但在过程数据对象（PDO）和服务数据对象（SDO）中，COB-ID 中包含了 Node-ID。

由于需要区分每个 CANopen 节点的输入和输出，所以 **PDO 分为 TPDO（发送 PDO）和(接收 RPDO)，发送和接收是以 CANopen 从站节点为参考**（如果 CAN 主站就相反）。**TPDO 和 RPDO 分别有 4 个数据对象，每种数据对象就是 1 条 CAN 报文封装**，如表 4.2 所示，这些都是数据收发的容器，就像图 4.3 所示，水果箩筐为使用者准备好，就看使用者在里面放什么水果了。

而 SDO 就相对比较简单固定，**发起通讯的“问”SDO 的 CAN 帧 ID 就是 600_h +node-ID，这里的 Node-ID 是被问的节点地址，而被问的节点应“答”SDO 的 CAN 帧 ID 就是 580_h +node-ID**。一般在 CANopen 网络中，只有 NMT 主机能发起 SDO 通讯，进行节点参数配置或者关键性参数的传递。当然从节点也可以对其他从节点发起 SDO 通讯。

如表 4.2 所示。为 CANopen 预定义报文的 PDO 报文和 SDO 报文中的 ID 分类。**使用者务必牢记！**

表 4.2 PDO 与 SDO 的 CAN-ID 定义

Object 对象	Specification 规范	CAN-ID（COB-ID）
TPDO1 发送过程数据对象 1	CiA301	181_h to $1FF_h$（180_h +node-ID）
RPDO1 接收过程数据对象 1	CiA301	201_h to $27F_h$（200_h +node-ID）
TPDO2 发送过程数据对象 2	CiA301	281_h to $2FF_h$（280_h +node-ID）
RPDO2 接收过程数据对象 2	CiA301	301_h to $37F_h$（300_h +node-ID）
TPDO3 发送过程数据对象 3	CiA301	381_h to $3FF_h$（380_h +node-ID）
RPDO3 接收过程数据对象 3	CiA301	401_h to $47F_h$（400_h +node-ID）
TPDO4 发送过程数据对象 4	CiA301	481_h to $4FF_h$（480_h +node-ID）
RPDO4 接收过程数据对象 4	CiA301	501_h to $57F_h$（500_h +node-ID）
Default SDO server-to-client 服务数据对象“答”	CiA301	581_h to $5FF_h$（580_h +node-ID）
Default SDO client-to-server 服务数据对象“问”	CiA301	601_h to $67F_h$（600_h +node-ID）

5. 对象字典 OD（Object dictionary)

CANopen 对象字典（OD: Object Dictionary）是 CANopen 协议最为核心的概念。所谓的对象字典就是一个有序的对象组，描述了对应 CANopen 节点的所有参数，包括通讯数据的存放位置也列入其索引，**这个表变成可以传递形式就叫做 EDS 文件（电子数据文档 Electronic Data Sheet）**。对象字典，就像体检表，具备这个人每个功能的参数，便于用人单位（主站）进行合理分配工作。如图 5.1 所示。

姓 名		性 别	女	年 龄	30	部 门	
血 压	98/66mmHg（正常<140/90）			体重指数（体重÷身高²）		19 （正常值18-25）	
一般查体	正常			眼 科	轻度干眼症		
	检查项目	测定值	正常值	口 腔	牙结石		
尿液分析	PH 值(PH)	7	5--7	尿液分析	蛋白质(PRO)	—	（—）
	亚硝酸盐(NIT)	—	（—）		胆红素(BIL)	—	（—）
	葡萄糖(GLU)	—	（—）		尿胆原(URO)	—	空白或±
	尿比重(SG)	1.015	1.0-1.2		酮体(KET)	—	（—）
	隐血(BLD)	—	（—）		白细胞(WBC)	—	（—）

血液检查	项 目	测定值	正常值	血液检查	项 目	测定值	正常值
	血 糖	4.44	3.89-6.10mmol/L		总蛋白	67.3	60-85g/L
	甘油三脂	0.33	0.23-1.78mmol/L		白蛋白	43.2	35-55g/L
	总胆固醇	2.83	2.8-5.85 mmol/L		转氨酶 AST	23.7	0-40 U/L
	尿素氮	3.11	1.81-7.14mmol/L		总胆红素	8.5	6.0-23umol/L
	肌 酐	71.6	44-133umol/L		直接胆红素	2.1	1.2-6.0umo/L
	尿 酸	211.1	160-400umol/L		甲胎蛋白	—	（—）

参 数	结 果	正常值	参 数	结 果	正常值
白细胞数目	6×10^9/L	4.0-10.0	平均红细胞体积	87.9 FL	80.0-97.0
淋巴细胞数目	1.9×10^9/L	0.6-4.1	平均血红蛋白含量	29.3 Pg	26.5-33.5

图 5.1 对象字典与体检表

每个对象采用一个 16 位的索引值来寻址，这个索引值通常被称为**索引，其范围在 0x0000 到 0xFFFF 之间。为了避免数据大量时无索引可分配，所以在某些索引下也定义了一个 8 位的索引值，这个索引值通常被称为子索引**，其范围是 0x00 到 0xFF 之间。

每个索引内具体的参数，最大用 32 位的变量来表示，即 Unsigned32，四个字节。

每个 CANopen 设备都有一个对象字典，使用电子数据文档（EDS 文件）来记录这些参数，而不需要把这些参数记录在纸上。对于 CANopen 网络中的主节点来说，不需要对 CANopen 从节点的每个对象字典项都访问。

CANopen 对象字典中的项由一系列子协议来描述。子协议为对象字典中的每个对象都描述了它的功能、名字、索引、子索引、数据类型，以及这个对象是否必需、读写属性等等，这样可保证不同厂商的同类型设备兼容。

CANopen 协议的核心描述子协议是 DS301，其包括了 CANopen 协议应用层及通信结构描述，其它的协议子协议都是对 DS301 协议描述文本的补充与扩展。在不同的应用行业都会起草一份 CANopen 设备子协议，子协议编号一般是 DS4xx 。

5.1 对象字典概述

如表 5.1 所示，为对象字典索引区域定义，其中标绿色底纹的**通讯对象子协议区**和**制造商特定子协议区**是用户需要关注的区域。

表 5.1 对象字典概述

Index range 索引范围	Description 描述
0000_h	Reserved 保留
0001_h to $025F_h$	Data types 数据类型
0260_h to $0FFF_h$	Reserved 保留
1000_h to $1FFF_h$	Communication profile area 通讯对象子协议区
2000_h to $5FFF_h$	Manufacturer-specific profile area 制造商特定子协议区
6000_h to $9FFF_h$	Standardized profile area 标准化设备子协议区
$A000_h$ to $AFFF_h$	Network variables 网络变量（符合 IEC61131-3）
$B000_h$ to $BFFF_h$	System variables 用于路由网关的系统变量
$C000_h$ to $FFFF_h$	Reserved 保留

5.2 通讯对象子协议区（Communication profile area）

通讯对象子协议区（Communication profile area）定义了所有和通信有关的对象参数，如表 5.2 所示，**标绿色底纹的索引范围 1000_h to 1029_h 为通用通讯对象，所有 CANopen 节点都必须具备这些索引，否则将无法加入 CANopen 网络**。其他索引根据实际情况进行分配与定义。

表 5.2 通讯对象子协议区

Index range 索引范围	Description 描述
1000_h to 1029_h	General communication objects 通用通讯对象
1200_h to $12FF_h$	SDO parameter objects SDO 参数对象
1300_h to $13FF_h$	CANopen safety objects 安全对象
1400_h to $1BFF_h$	PDO parameter objects PDO 参数对象
$1F00_h$ to $1F11_h$	SDO manager objects SDO 管理对象
$1F20_h$ to $1F27_h$	Configuration manager objects 配置管理对象
$1F50_h$ to $1F54_h$	Program control object 程序控制对象
$1F80_h$ to $1F89_h$	NMT master objects 网络管理主机对象

5.3 通用通讯对象（General communication objects）

由于**通用通讯对象十分重要，NMT 主站（CANopen 主站）在启动时，通常都全部或者部分读取所有从站中通用通讯对象中的索引，所以所有的通用通讯对象都必须在 CANopen 从站中实现**，使用者也必须熟知这些索引地址与其含义。如表 5.3 所示。

表 5.3 通用通讯对象

Index 索引	Object 对象	Name 名字
1000_h	VAR 变量	Device type 设备类型
1001_h	VAR 变量	Error register 错误寄存器
1002_h	VAR 变量	Manufacturer status register 制造商状态寄存器

1003_h	ARRAY 数组	Pre-defined error field 预定义错误场
1005_h	VAR 变量	COB-ID Sync message 同步报文 COB 标识符
1006_h	VAR 变量	Communication cycle period 同步通信循环周期（单位 us）
1007_h	VAR 变量	Synchronous windows length 同步窗口长度(单位 us)
1008_h	VAR 变量	Manufacturer device name 制造商设备名称
1009_h	VAR 变量	Manufacturer hardware version 制造商硬件版本
$100A_h$	VAR 变量	Manufacturer software version 制造商软件版本
$100C_h$	VAR 变量	Guard time 守护时间（单位 ms）
$100D_h$	VAR 变量	Life time factor 寿命因子（单位 ms）
1010_h	VAR 变量	Store parameters 保存参数
1011_h	VAR 变量	Restore default parameters 恢复默认参数
1012_h	VAR 变量	COB-ID time stamp 时间报文 COB 标识符（发送网络时间）
1013_h	VAR 变量	High resolution time stamp 高分辨率时间标识
1014_h	VAR 变量	COB-ID emergency 紧急报文 COB 标识符
1015_h	VAR 变量	Inhibit time emergency 紧急报文禁止时间（单位 100us）
1016_h	ARRAY 数组	Consumer heartbeat time 消费者心跳时间间隔(单位 ms)
1017_h	VAR 变量	Producer heartbeat time 生产者心跳时间间隔（单位 ms）
1018_h	RECORD 记录	Identity object 厂商 ID 标识对象
1019_h	VAR 变量	Sync.counter overflow value 同步计数溢出值
1020_h	ARRAY 数组	Verify configuration 验证配置
1021_h	VAR 变量	Store EDS 存储 EDS
1022_h	VAR 变量	Storage format 存储格式
1023_h	RECORD 记录	OS command 操作系统命令
1024_h	VAR 变量	OS command mode 操作系统命令模式
1025_h	RECORD 记录	OS debugger interface 操作系统调试接口
1026_h	ARRAY 数组	OS prompt 操作系统提示
1027_h	ARRAY 数组	Module list 模块列表
1028_h	ARRAY 数组	Emergency consumer 紧急报文消费者
1029_h	ARRAY 数组	Error behavior 错误行为

5.4 制造商特定子协议（Manufacturer-specific Profile）

对象字典索引 2000_h to $5FFF_h$ 为制造商特定子协议，**通常是存放所应用子协议的应用数据**。而上文所描述的通讯对象子协议区（Communication profile area）是存放这些应用数据的通信参数。比如广州致远电子的 XGate-COP10 从站模块规定了：

- RPDO 的通讯参数存放在 1400_h to $15FF_h$ 映射参数存放在 1600_h to $17FF_h$ 数据存放为 2000_h 之后厂商自定义区；
- TPDO 的通讯参数存放在 1800_h to $19FF_h$ 映射参数存放在 $1A00_h$ to $1BFF_h$ 数据存放为 2000_h 之后厂商自定义区。

具体讲解将在后面 PDO（过程数据对象）章节进行详细叙述。

对于在设备子协议中未定义的特殊功能，制造商也可以在此区域根据需求定义对象字典对象。因此这个区域对于不同的厂商来说，相同的对象字典项其定义不一定相同。

5.5 标准化设备子协议(Standardized profile area)

标准化设备子协议，为各种行业不同类型的标准设备定义对象字典中的对象。目前已有十几种为不同类型的设备定义的子协议，例如 DS401、DS402、DS406 等，其索引值范围为 0x6000～0x9FFF。同样，这个区域对于不同的标准化设备子协议来说，相同的对象字典项其定义不一定相同。

这部分在本文中不做叙述。

5.6 对象字典和 EDS 文件实例

对于对象字典和 EDS 文件的实现，需要使用专用的 EDS 生成工具，并且能通过 CiA 的 EDS 测试工具进行一致性测试。

我们可以通过广州致远电子的 CANopen 从站协议栈模块 XGate-COP10 模块的对象字典和 EDS 文件，来真实感受一下。

如图 5.2 所示，为 XGate-COP10 的 EDS 文件导入到 USBCAN-E-P 主站卡管理软件 CANManager for CANopen 中。配置从站框中可以观察到 XGate-COP10 的对象字典内容，1008_h 的索引是这个设备的名称 XGate-COP10，1009_h 是硬件版本，$100A_h$ 是软件版本，1018h 的索引为标示对象，其下有若干个子索引，其中 1008.01h 的子索引为厂商代码 0x2B6，这是广州致远电子股份有限公司在 CiA 协会申请的厂商代码，任何一个生产 CANopen 的厂家虽然不强制加入 CiA 协会，但必须申请唯一的厂商代码。

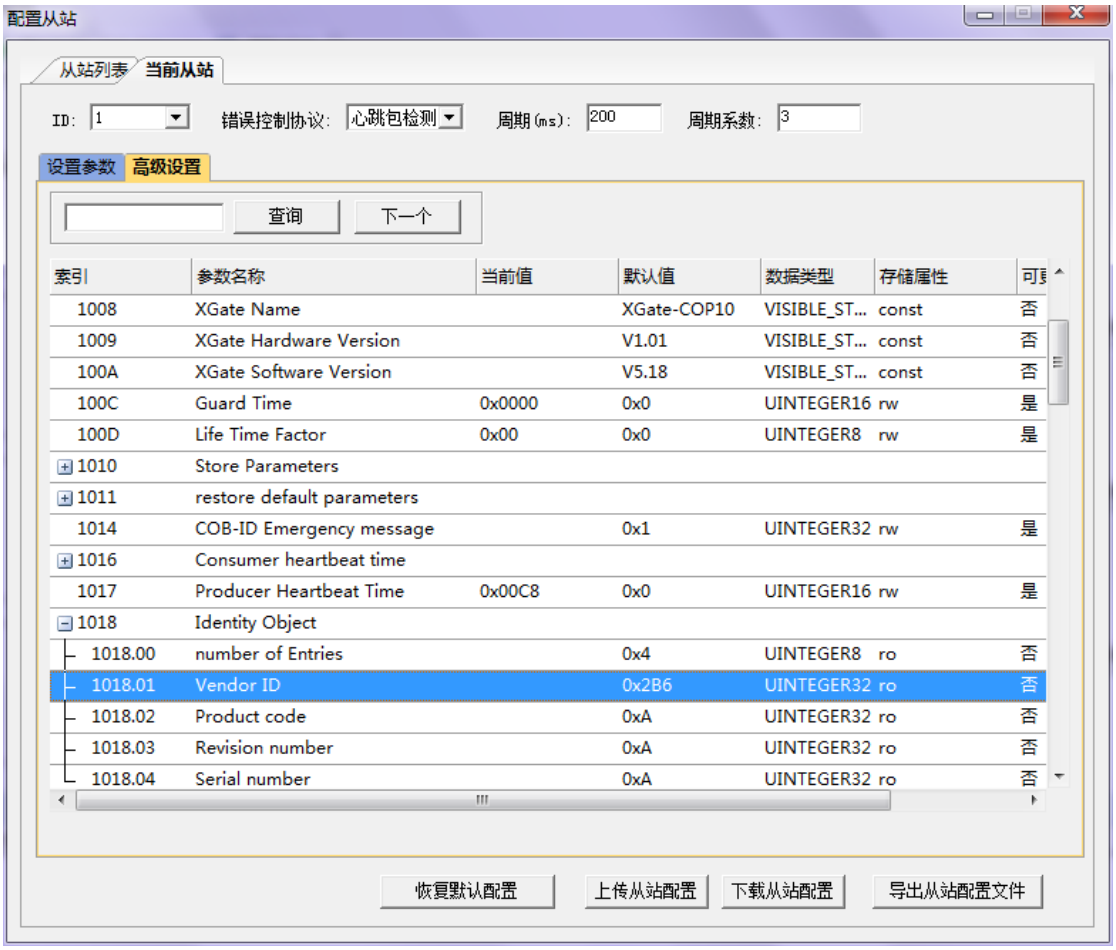


图 5.2 对象字典配置

对象字典导出后就是所谓的 EDS 文件，用于随着产品组态时使用。如程序清单所示，

为 EDS 文件中到 1018h 索引的内容。

```
[FileInfo]
FileName=XGate-COP10.eds
FileVersion=1
FileRevision=100
EDSVersion=100
Description=EDS for XGate-COP10
CreationTime=10:22AM
CreationDate=06-20-2009
CreatedBy=
ModificationTime=10:22AM
ModificationDate=06-20-2009
ModifiedBy=xiangjl, GUANGZHOU ZHIYUAN electronic co.,ltd

[DeviceInfo]
VendorName=GUANGZHOU ZHIYUAN electronic co.,ltd
VendorNumber=694
ProductName=CANopen Comm_moudle
ProductNumber=0
RevisionNumber=100
OrderCode=XGate-COP10
Baudrate_10=1
Baudrate_20=1
Baudrate_50=1
Baudrate_125=1
Baudrate_250=1
Baudrate_500=1
Baudrate_800=1
Baudrate_1000=1
SimpleBootUpMaster=0
SimpleBootUpSlave=1
Granularity=8
DynamicChannelsSupported=0
GroupMessaging=0
NrOfRXPDO=12
NrOfTXPDO=12
LSS_Supported=1

[DummyUsage]
Dummy0001=0
Dummy0002=0
Dummy0003=0
Dummy0004=0
Dummy0005=1
```

```
Dummy0006=1
Dummy0007=1

[Comments]
Lines=1
Line1=XGate-COP10

[MandatoryObjects]
SupportedObjects=3
1=0x1000
2=0x1001
3=0x1018

[1000]
ParameterName=Device Type
ObjectType=0x7
DataType=0x0007
AccessType=ro
DefaultValue=0x0
PDOMapping=0x0

[1001]
ParameterName=Error Register
ObjectType=0x7
DataType=0x0005
AccessType=ro
DefaultValue=0x0
PDOMapping=0x0
LowLimit=0x0
HighLimit=0xFF

[1018]
SubNumber=0x5
ParameterName=Identity Object
ObjectType=0x9

[1018sub0]
ParameterName=number of Entries
ObjectType=0x7
DataType=0x0005
AccessType=ro
DefaultValue=0x4
PDOMapping=0x0
```

```
[1018sub1]
ParameterName=Vendor ID
ObjectType=0x7
DataType=0x0007
AccessType=ro
DefaultValue=0x2B6
PDOMapping=0x0
LowLimit=0x0
HighLimit=0xFFFFFFFF

[1018sub2]
ParameterName=Product code
ObjectType=0x7
DataType=0x0007
AccessType=ro
DefaultValue=0xA
PDOMapping=0x0
LowLimit=0x0
HighLimit=0xFFFFFFFF

[1018sub3]
ParameterName=Revision number
ObjectType=0x7
DataType=0x0007
AccessType=ro
DefaultValue=0xA
PDOMapping=0x0
LowLimit=0x0
HighLimit=0xFFFFFFFF

[1018sub4]
ParameterName=Serial number
ObjectType=0x7
DataType=0x0007
AccessType=ro
DefaultValue=0xA
PDOMapping=0x0
LowLimit=0x0
HighLimit=0xFFFFFFFF
```

6. 网络管理 NMT（Network management）与 CANopen 主站

前文所述，一个 CANopen 网络中为了保证可靠、可控，必须要 NMT 网络管理，就像一个军队一样，要令行禁止，才能达到稳定、高效的目标。如图 6.1 所示。指挥员（NMT 主机）通过发号施令，士兵（NMT 从机）进行自由俯卧训练，这样整个训练都是有序的。

图 6.1 网络管理与部队管理

所以**每个 CANopen 从节点的 CANopen 协议栈中，必须具备 NMT 管理的相应代码，这是节点具备 CANopen 协议的最基本的要素**。

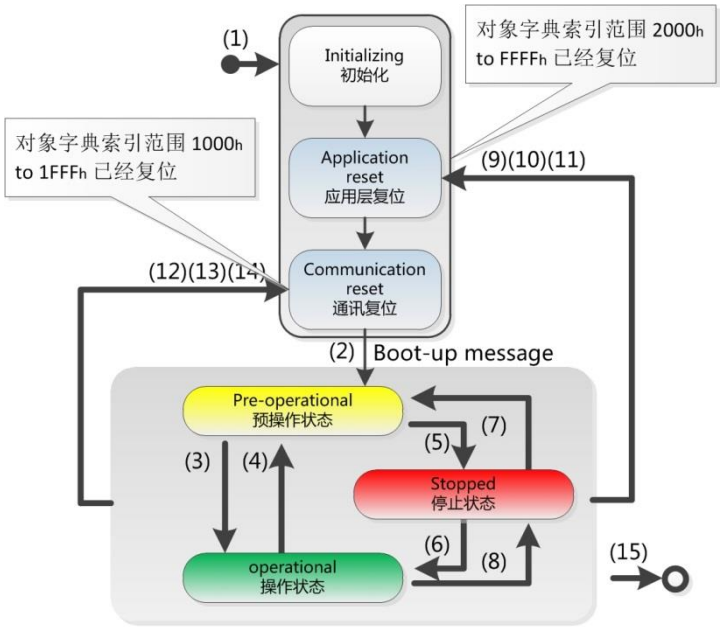
6.1 NMT 节点状态

NMT 管理涉及到一个 CANopen 节点从上电开始的 6 钟状态，包括：

- **初始化**（Initializing）：节点上电后对功能部件包括 CAN 控制器进行初始化；
- **应用层复**位（Application Reset）：节点中的应用程序复位（开始），比如开关量输出、模拟量输出的初始值；
- **通讯复位**（Communication reset）：节点中的CANopen通讯复位（开始），从这个时刻起，此节点就可以进行CANopen通讯了。
- **预操作状态**（Pre-operational）：节点的CANopen通讯处于操作就绪状态，此时此节点不能进行PDO通信，而可以进行SDO进行参数配置和NMT网络管理的操作；
- **操作状态**（operational）：节点收到NMT主机发来的启动命令后，CANopen通讯被激活，PDO通信启动后，按照对象字典里面规定的规则进行传输，同样SDO也可以对节点进行数据传输和参数修改；
- **停止状态**（Stopped）：节点收到NMT主机发来的停止命令后，节点的PDO通信被停止，但SDO和NMT网络管理依然可以对节点进行操作；

除了初始化状态，NMT主机通过NMT命令可以让网络中任意一个的CANopen节点进行其他5种状态的切换。如图 6.2所示。

当然CANopen节点也可以程序自动完成这些状态的切换。



(1)Power on上电
(2)Automatic switch to Pre-operational
自动切换到预操作状态
(3)and(6)NMT Switch to Operational
网络管理切换到操作状态
(4)and(7)NMT Switch to Pre-Operational
网络管理切换到预操作状态
(5)and(8)NMT Switch to Stopped
网络管理切换到停止状态
(9),(10)and(11)NMT Switch to Application reset
网络管理切换到应用层复位状态
(12),(13)and(14)NMT Switch to Communication reset
网络管理切换到通讯复位状态
(15)Power-off or hardware reset
掉电或者硬件复位

图 6.2 NMT 管理状态转换图

6.2 NMT 节点上线报文

任何一个 CANopen 从站上线后，为了提示主站它已经加入网络（便于热插拔），或者避免与其他从站 Node-ID 冲突。这个从站必须发出**节点上线报文（boot-up）**，如图 6.3 所示，节点上线报文的 **ID 为 700_h+Node-ID，数据为 1 个字节 0**。生产者为 CANopen 从站。

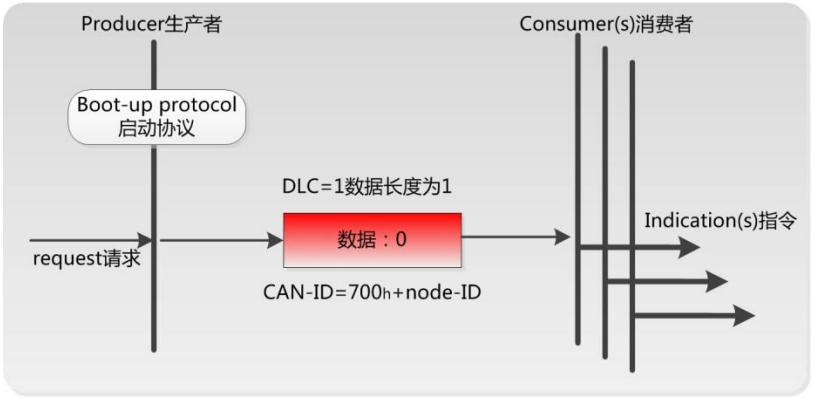


图 6.3 节点上线报文

6.3 NMT 节点状态与心跳报文

为了监控 CANopen 节点是否在线与目前的节点状态。CANopen 应用中通常都要求在线上电的从站定时发送状态报文（心跳报文），以便于主站确认从站是否故障、是否脱离网络。

如图 6.4 所示，为心跳报文发送的格式，CANID 与节点上线报文相同**为 700_h+Node-ID，数据为 1 个字节，代表节点目前的状态，04_h 为停止状态，05_h 为操作状态，$7F_h$ 为预操作状态。**

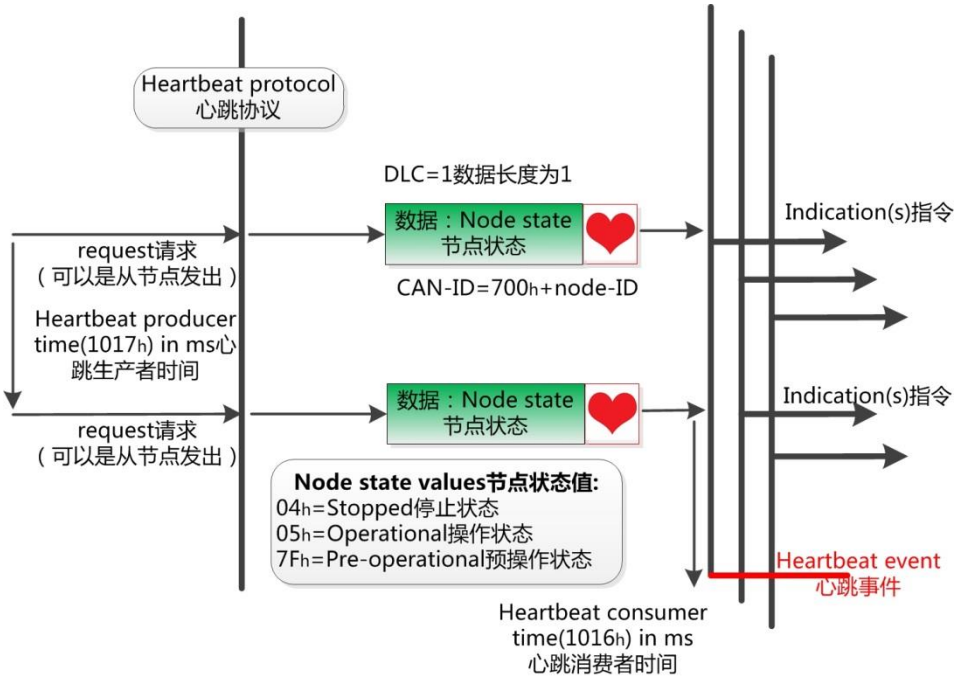


图 6.4 节点状态与心跳报文

CANopen 从站按其对象字典中 1017_h 中填写的心跳生产时间（ms）进行心跳报文的发送，而 CANopen 主站（NMT 主站）则会按其 1016h 中填写的心跳消费时间进行检查，假设超过诺干次心跳消费时间没有收到从站的心跳报文，则认为从站已经离线或者损坏。

6.4 NMT 节点守护

在早期 CANopen 应用中，还有一种可以**通过轮询模式监视从站状态的节点守护模式，它与心跳报文模式二者不能并存**。通过节点守护，MNT 主机可以检查每个节点的当前状态。

NMT-Master 节点发送**标准远程帧**（无数据）如下：

NMT-Master ➔ NMT-Slave

COB-ID
0x700+Node_ID

NMT-Slave 节点**应答发送数据帧，数据为 1 个字节**：

NMT-Master ➔ NMT-Slave

COB-ID	Byte0
0x700 + Node_ID	Bit 7 : toggle Bit6-0 : 状态

数据部分包括一个触发位（bit7），**触发位必须在每次节点保护应答中交替置“0”或者“1”**。触发位在第一次节点保护请求时置为“0”。位 0 到位 6（bits0～6）表示节点状态，可为表 6.1 中的数值。

表 6.1 节点守护状态

Value	状态
4	Stopped 停止
5	Operational 操作
127	Pre-operational 预操作

由于远程帧在 CAN 发展中逐渐被淘汰，而节点守护由于需要更多的主站开销与增加网络负载，CiA 协会已经不建议使用，被心跳报文所取代。

6.5 NMT 节点状态切换命令

NMT 网络管理中，最核心的就是 NMT 节点状态切换命令，这是 NMT 主站所进行网络管理的“命令”报文。**使用者必须牢记这些命令。**

CANID 均为 000_h，具备最高的 CAN 优先级。数据为 2 个字节：

◆ **第 1 个字节代表命令类型：**

01_h 为启动命令（让节点进入操作状态）；

02_h 为停止命令（让节点进入停止状态）；

80_h 为进入预操作状态（让节点进入预操作状态）；

81_h 为复位节点应用层（让节点的应用恢复初始状态，比如列车门都恢复打开状态）；

82_h 为复位节点通讯（让节点的 CAN 和 CANopen 通讯重新初始化，一般用于总线收到干扰，导致节点总线错误被动，或者总线关闭时）。

◆ **第二个字节代表被控制的节点 Node-ID，**

如果要对整个网络所有节点同时进行控制，则这个数值为 0 即可。

如图 6.5 所示，

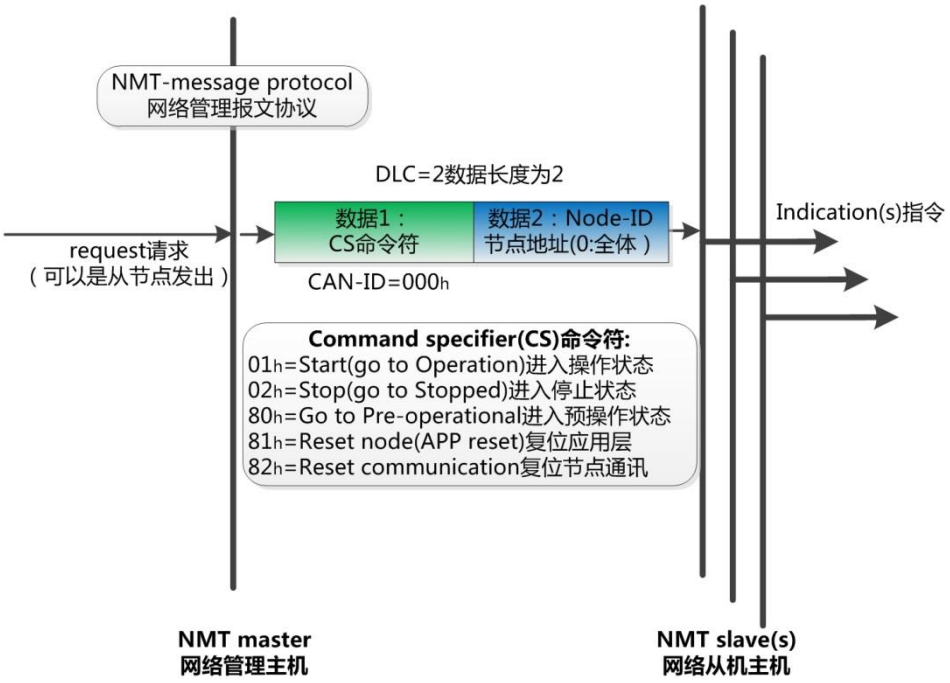


图 6.5 节点状态切换命令

6.6 CANopen 主站设备

通常 NMT 主站也称为 CANoepn 主站，上文所述为 CANopen 最基本的 NMT 操作，而作为一个完整的 CANopen 主站设备，设备模型如图 6.6 所示。为了满足管理整个 CANopen 网络的从站设备，需要具备以下功能：

- 支持 PDO、SDO 发送与接收；
- 支持 NMT 网络管理；
- 支持 PDO 通信类型并能够支持监控每一个 PDO 目标；
- LSS 层设置功能：从站波特率设置、从站节点编号设置；
- 支持从站管理功能：类型与名称读取、对象字典读写；
- 紧急报文发送功能；
- 扩展 CANopen 标准指示灯功能。

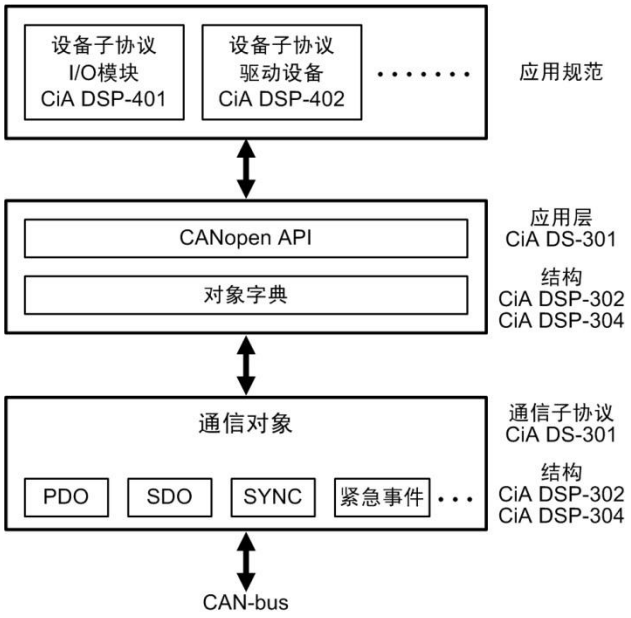


图 6.6 CANopen 设备模型

目前有二种形式的主站，一种是可编程控制器（PLC）中的一个单元，它的内部集成了 CANopen 的主站功能，这个单元能连接到 CANopen 总线，同时因为它是 PLC 中的一个单元，它能与 PLC 的 CPU 交换数据，因此通过编写 PLC 程序对它所连接的 CANopen 从站进行管理和控制。

图 6.7 带 CANopen 的 PLC

另一种是通过 PC 扩展一个 CANopen 主站通信卡，从而令 PC 具有管理 CANopen 通信网络的能力。推荐使用 PCI 总线或 USB 总线来扩展 CANopen 通信卡，比如广州致远电子股份有限公司的 PCI-5010-P 或 USBCAN-E-P 主站卡，如图 6.8 所示。使用它们不仅可以令 PC 成为一个 CANopen 网络的管理节点，还可以开发或测试 CANopen 网络、拓展连接其他网络。

图 6.8 推荐的 CANopen 主站通信卡

PCI-5010-P 通信卡内嵌 1 路隔离 CAN 接口，常用于工控场合，通过 PCI 总线连接工控 PC 机；USBCAN-E-P 通信卡也带 1 路全隔离 CAN 接口，常用于便携测试领域，通过 USB 总线连接测试 PC 机。这两款设备的内嵌 CAN 接口都设计有增强隔离、ESD、EFT、EMI 等多种保护措施，保障设备在干扰恶劣环境中的可靠通信。同时，配套各种 CANopen 支持软件，有 CANopen 函数库、编程示例、监控与测试软件、OPC 服务器、协议分析等。

另外，作为通用的 CANopen 通信接口卡，这两款设备还具有硬件自动存储报文、通用 CAN 报文收发、总线参数诊断等增强功能，方便进行复杂网络的二次开发。

7. 过程数据对象 PDO（Process data object)

如前文所述 PDO 属于过程数据用来传输实时数据，**即单向传输，无需接收节点回应 CAN 报文来确认，从通讯术语上来说是属于“生产消费”模型** PDO，如图 7.1 所示，就像食品销售柜台，**生产者摆出“食品”，但只有“需要”的消费者才会来买，没有指向性**。

图 7.1 生产者消费者模型

数据长度被限制为 1~8 字节。最多只要 1 帧就可以把一条信息或者一个变量传递结束。

7.1 PDO 的 CAN-ID 定义

PDO 通信比较灵活，广义上只要符合 PDO 范围内的所有 CANID 都可以作为节点自身的 TPDO 或者 RPDO 使用，也称为 COB-ID，不受功能码和 Node-ID 限制，如图 7.2 所示。

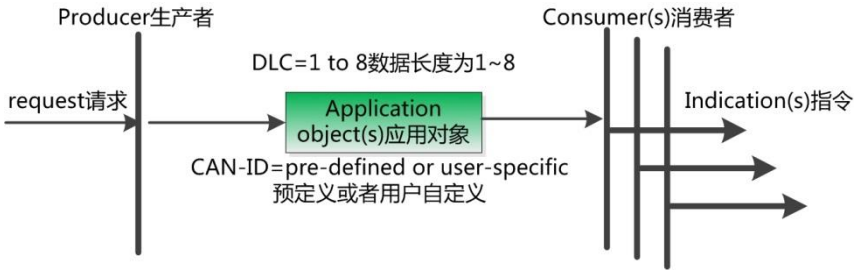


图 7.2 PDO 的 CANID 规则

而在 PDO 预定义中，人为规定了 TPDO 和 RPDO，规定了 Node-ID 在 PDO 中的位置，规定了 PDO 的编号，如表 7.1 所示。

表 7.1 PDO 的 CAN-ID 定义

Object 对象	Specification 规范	CAN-ID（COB-ID）
TPDO1 发送过程数据对象 1	CiA301	181_h to $1FF_h$（180_h +node-ID）
RPDO1 接收过程数据对象 1	CiA301	201_h to $27F_h$（200_h +node-ID）
TPDO2 发送过程数据对象 2	CiA301	281_h to $2FF_h$（280_h +node-ID）
RPDO2 接收过程数据对象 2	CiA301	301_h to $37F_h$（300_h +node-ID）
TPDO3 发送过程数据对象 3	CiA301	381_h to $3FF_h$（380_h +node-ID）
RPDO3 接收过程数据对象 3	CiA301	401_h to $47F_h$（400_h +node-ID）
TPDO4 发送过程数据对象 4	CiA301	481_h to $4FF_h$（480_h +node-ID）
RPDO4 接收过程数据对象 4	CiA301	501_h to $57F_h$（500_h +node-ID）

PDO 分为 TPDO（发送 PDO）和(接收 RPDO)，发送和接收是以 CANopen 节点自身为参考（如果 CAN 主站或者其他从站就相反）。**TPDO 和 RPDO 分别有 4 个数据对象，每种数据对象就是 1 条 CAN 报文封装**，这些都是数据收发的容器，就像图 4.3 所示，水果箩筐为使用者准备好，就看使用者在里面放什么水果了。

当然，如果某个节点需要传递的资源特别多，则有出现例如 TPDO5 之类的数据对象，而它们的 CAN-ID 定义就需要打破预定义的规则，比如**我们可以定义 Node-ID 为 1 的节点中 TPDO5 是 182_h**，这里的 PDO 的 COB-ID 中的低 7 位不再是表示 Node-ID。其实所有的 PDO 的 COB-ID 与 Node-ID 无必然规则上的联系。

7.2 PDO 的传输形式

PDO 的两种传输方式：同步传输和异步传输。 如图 7.3 所示，1、2 为异步传输，3、4 为同步传输。

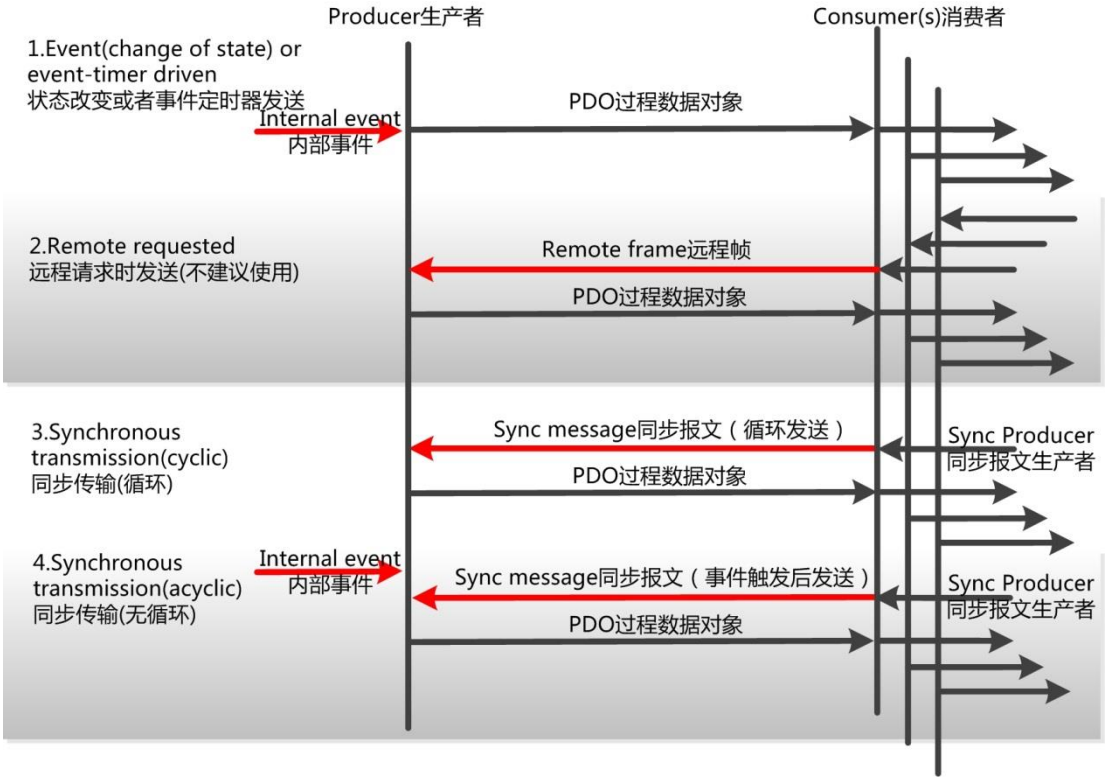


图 7.3 PDO 的传输形式

- **异步传输（由特定事件触发）**

其触发方式可有两种，第一种是由设备子协议中规定的对象特定事件来触发（例如，定时传输，数据变化传输等）。第二种是通过发送与 PDO 的 COB-ID 相同的远程帧来触发 PDO 的发送。**目前应用中的异步传输基本都采用第一种**。

- **同步传输（通过接收同步对象实现同步）**

同步传输就是通过同步报文让所有节点能在同一时刻进行上传数据或者执行下达的应用指令，可以有效避免异步传输导致的应用逻辑混乱和总线负载不平衡的问题。一般发送同步报文的节点是 NMT 主机。

同步传输又可分为周期传输（循环）和非周期传输（无循环）。周期传输则是通过接收

同步对象（SYNC）来实现，可以设置 1~240 个同步对象触发。 非周期传输是由远程帧预触发或者由设备子协议中规定的对象特定事件预触发传送。

7.3 PDO 的通信参数

PDO 通信参数，定义了该设备所使用的 COB-ID、传输类型、定时周期等。RPDO 通讯参数位于对象字典索引的 1400_h to $15FF_h$，TPDO 通讯参数位于对象字典索引的 1800_h to $19FF_h$。每条索引代表一个 PDO 的通信参数集，其中的子索引分别指向具体的各种参数。如表 7.2 所示。

表 7.2 PDO 的通信参数

Index 索引	Sub-index 子索引	Description 描述	Data type 数据类型
RPDO: 1400_h to $15FF_h$ TPDO: 1800_h to $19FF_h$	00_h	Number of entries 参数条目数量	Unsigned8
	01_h	COB-ID：发送/接收这个 PDO 的帧 ID	Unsigned32
	02_h	Transmission type 发送类型	Unsigned8
		00_h：非循环同步	
		01_h：循环同步	
		FC_h：远程同步	
		FD_h：远程异步	
		FE_h：异步，制造商特定事件	
		FF_h：异步，设备子协议特定事件	
	03_h	Inhibit time 生产禁止约束时间(1/10ms)	Unsigned16
	05_h	Event timer 事件定时器触发的时间(单位 ms)	Unsigned16
	06_h	SYNC start value 同步起始值	Unsigned8

- **Number of entries 参数条目数量**：即本索引中有几条参数；
- **COB-ID**：即这个 PDO 发出或者接收的对应 CAN 帧 ID；
- **发送类型**：即这个 PDO 发送或者接收的传输形式，通常使用循环同步和异步制造商特定事件较多；
- **Inhibit time 生产禁止约束时间**(1/10ms)：约束 PDO 发送的最小间隔，避免导致总线负载剧烈增加，比如数字量输入过快，导致状态改变发送的 TPDO 频繁发送，总线负载加大，所以需要一个约束时间来进行“滤波”，这个时间单位为 0.1ms；
- **Event timer 事件定时器触发的时间**(单位 ms)：定时发送的 PDO，它的定时时间，如果这个时间为 0，则这个 PDO 为事件改变发送。
- **SYNC start value 同步起始值**：同步传输的 PDO，收到诺干个同步包后，才进行发送，这个同步起始值就是同步包数量。比如设置为 2，即收到 2 个同步包后才进行发送。

7.4 PDO 的映射参数

PDO 映射参数是初学者学习 CANopen 时的一个难点，它包含了一个对象字典中的对象列表，这些对象映射到相应的 PDO，其中包括数据的长度（单位，位），对于生产者和消费者都必须要知道这个映射参数，才能够正确的解释 PDO 内容。就是**将通信参数、应用数据和具体 CAN 报文中数据联系起来。**

RPDO 通讯参数 1400_h to $15FF_h$，映射参数 1600_h to $17FF_h$，数据存放为 2000_h 之后厂商自定义区域；TPDO 通讯参数 1800_h to $19FF_h$，映射参数 $1A00_h$ to $1BFF_h$，数据存放为 2000_h 之后厂商自定义区域。

为了更加直观地表现映射，表 7.3 模拟 TPDO1，将参数、应用数据、CAN 报文数据联合起来展示，不同的映射采用不同的颜色。

表 7.3 PDO 在对象字典中的映射关系

Index 索引	Sub 子索引	Object contents 对象内容
1800_h 通信参数	01_h	COB-ID：值为 181_h
	02_h	发送类型: **FE_h**
	03_h	生产禁止约束时间(1/10ms)：200
	05_h	Event timer 事件定时器触发的时间(单位 ms)：0
	06_h	SYNC start value 同步起始值：无
$1A00_h$ 映射参数	01_h	值 20000108_h 为映射到索引 2000_h 的子索引 01_h，对象是 8 位
	02_h	值 20030310_h 为映射到索引 2003_h 的子索引 03_h，对象是 16 位
	03_h	值 20030108_h 为映射到索引 2003_h 的子索引 01_h，对象是 8 位
以下为厂商自定义区域：		
2000_h	01_h	值 01_h
2000_h	02_h	值 02_h
2001_h	00_h	值 00_h
2002_h	00_h	值 00_h
2003_h	01_h	值 12_h
2003_h	02_h	值 34_h
2003_h	03_h	值 5678_h

CAN transmission(CAN 发送报文)

TPDO1（CAN-ID = 181_h） Data field：数据域 4 个字节

Data1	Data2	Data3	Data4
01_h	78_h	56_h	12_h

请读者阅读 3 遍以上，如果还是无法理解。请打开广州致远电子的 CANopen 主站卡管理软件 CANManager for CANopen，导入 XGate-COP10 模块从站协议栈模块的 EDS 文件，来真实感受一下。点击配置从站，“高级”中进行感性认识。

8. 服务数据对象 SDO（Service data object）

SDO 主要用于 CANopen 主站对从节点的参数配置。**服务确认是 SDO 的最大的特点，为每个消息都生成一个应答，确保数据传输的准确性**。如图 8.1 所示，这就像快递，需要收方签收后，给寄方发送一个已经签收的确认才算完成一次投递。

图 8.1 SDO 与快递签收

在一个 CANopen 系统中，通常 **CANopen 从节点作为 SDO 服务器，CANopen 主节点作为客户端**（称为 CS 通讯）。SDO 客户端通过索引和子索引，能够访问 SDO 服务器上的对象字典。这样 CANopen 主节点可以访问从节点的任意对象字典项的参数，并且 SDO 也可以传输任何长度的数据（当数据长度超过 4 个字节时就拆分成多个报文来传输）。

8.1 通讯原则（communication principle）

SDO 的通讯原则非常单一，**发送方（客户端）发送 CAN-ID 为 600_h+Node-ID 的报文，其中 Node-ID 为接收方（服务器）的节点地址，数据长度均为 8 字节；**

接收方（服务器）成功接收后，回应 CAN-ID 为 580_h+Node-ID 的报文。这里的 Node-ID 依然是接收方（服务器）的节点地址，数据长度均为 8 字节。如图 8.2 所示。

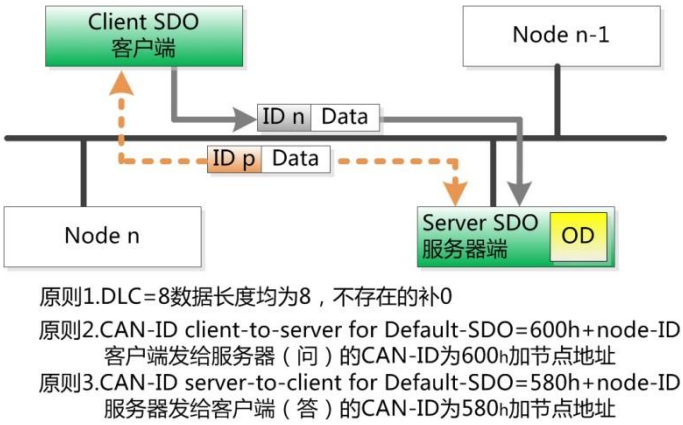


图 8.2 SDO 通讯原则

8.2 快速 SDO 协议（Expedited SDO protocol）

最常用最常见的 SDO 协议是**快速 SDO，所谓快速，就是 1 次来回就搞定。前提是读取和写入的值不能大于 32 位**。如图 8.3 所示，为快速 SDO 协议的示意图。**命令中直接包含了要读写的索引、子索引、数据**。可谓直接命中。

快速 SDO 的难点在于 CS 命令符的记忆，需要读者收藏这个示意图。

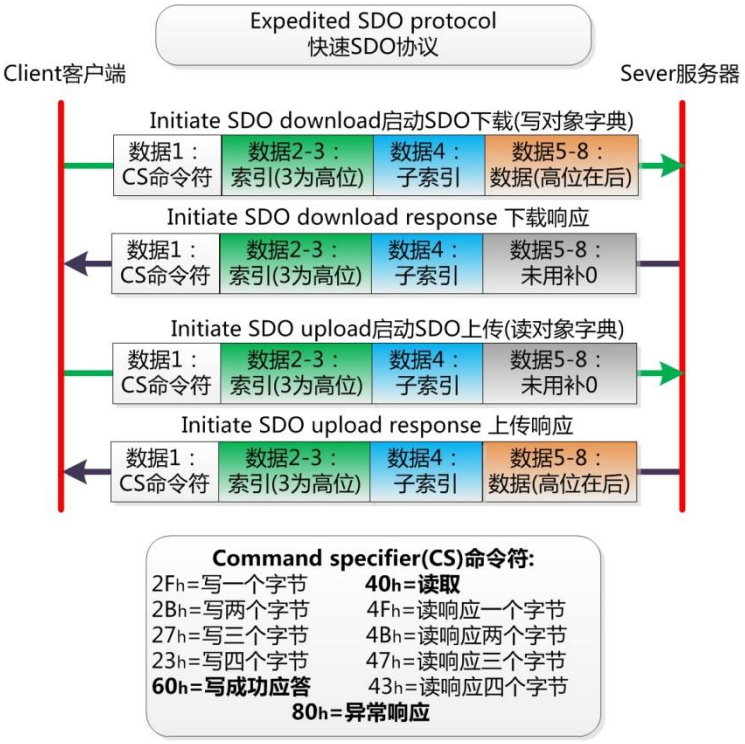


图 8.3 快速 SDO 示意图

通过快速 SDO，可以直接对 CANopen 节点的对象字典中的值进行读取和修改，所以在做参数配置之外，也经常作为关键性数据传输之用。比如 CANopen 控制机器人的电机转动角度时，就使用 SDO 来传输，保证可靠到达。

8.3 普通 SDO 协议（Normal SDO protocol）

当需要传输的值超过 32 位时，就不能使用快速 SDO 传输。必须使用普通 SDO 进行分帧传输。在应用中较少用到，一般用于 CANopen 节点的程序固件升级，或者做网关转换 MVB 总线之类数据最大可达 256 位的应用。

普通 SDO 协议难点在于分包逻辑与 CS 命令符的变化。依然难以记忆，需要读者将以下示意图进行收藏。

当然普通 SDO 的 CAN 帧 ID 与快速 SDO 相同，依然**发送方（客户端）发送的报文 CAN-ID 为 600_h+Node-ID，接收方（服务器）成功接收后，回应 CAN-ID 为 580_h+Node-ID 的报文。**

1. 下载协议 download protocol 如图 8.4 所示。

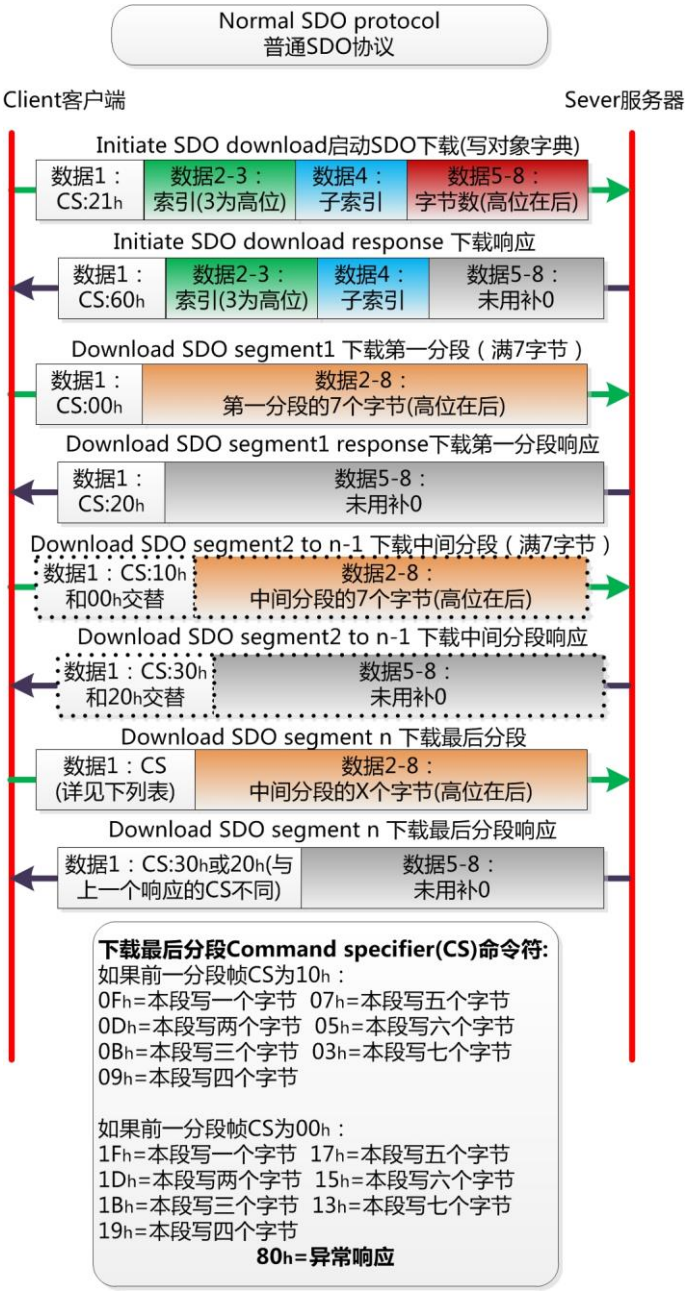


图 8.4 普通 SDO 下载协议

2. 上传协议 upload protocol 如图 8.5 所示。

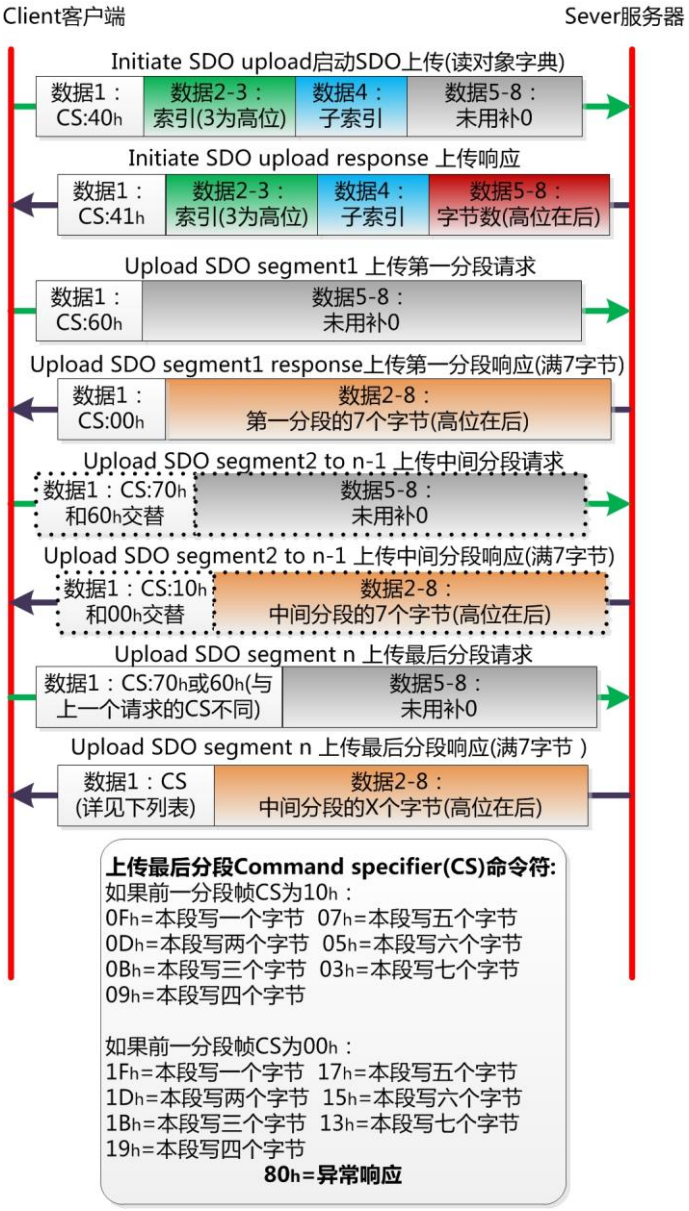


图 8.5 普通 SDO 上传协议

9. 特殊协议（Special protocols）

为了方便 CANopen 主站对从站管理。在 CANopen 协议中，已经为特殊的功能预定义了 COB-ID，其主要有以下几种特殊报文。

9.1 同步协议（Sync protocol）

同步（SYNC），**该报文对象主要实现整个网络的同步传输**，如图 9.1 所示，就像阅兵分列式上的方阵，所有士兵迈着整齐的步伐行进。

图 9.1 同步协议与阅兵分列式

每个节点都以该同步报文作为 PDO 触发参数，因此该同步报文的 COB-ID 具有比较高的优先级以及最短的传输时间。 **一般选用 80_h 作为同步报文的 CAN-ID**，如图 9.2 所示。

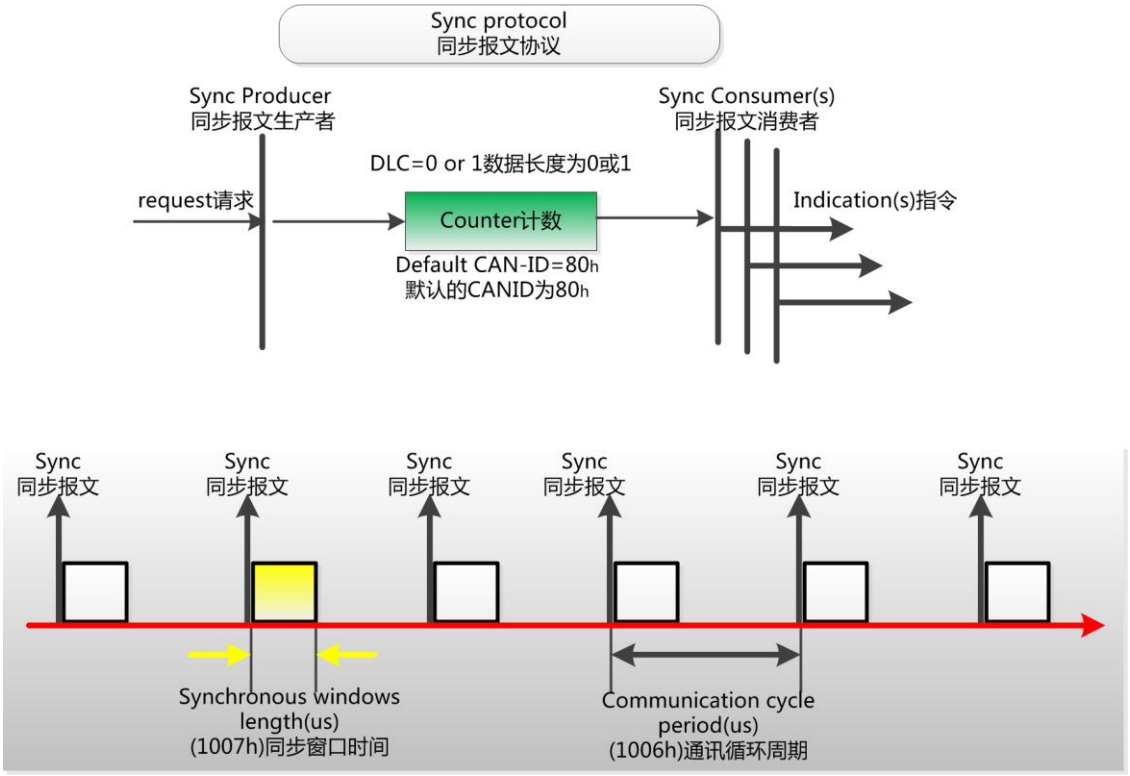


图 9.2 同步报文

一般同步报文由 NMT 主机发出，CAN 报文的数据为 0 字节。但如果一个网络内有 2 个同步机制，就需要设置不同的同步节拍，比如某些节点按 1 个同步帧发送 1 次 PDO，其他的节点收到 2 个同步帧才发送 1 此 PDO，所以这里 PDO 参数中的同步起始值就起了作用。

在同步协议中，有 2 个约束条件：

- **同步窗口时间**：索引 1007_h 约束了同步帧发送后，从节点发送 PDO 的时效，即在这个时间内发送的 PDO 才有效，超过时间的 PDO 将被丢弃；
- **通讯循环周期**：索引 1006_h 规定了同步帧的循环周期。

9.2 时间戳协议（Time-stamp protocol）

时间标记对象（Time Stamp），NMT 主机发送自身的时钟，为网络各个节点提供公共的时间参考，即网络对时，如图 9.3 所示。。这在故障诊断中非常需要，比如列车中火灾报警，检修人员需要准确获知报警的时刻，然后关联查看其它设备在这个时刻的工作状态。

图 9.3 时间戳协议与校时

时间戳协议采用广播方式，无需节点应答，**CAN-ID 为 100_h，数据长度为 6，数据为当前时刻与 1984 年 1 月 1 日 0 时的时间差**。如图 9.4 所示。节点将此时间存储在对象字典 1012_h 的索引中。

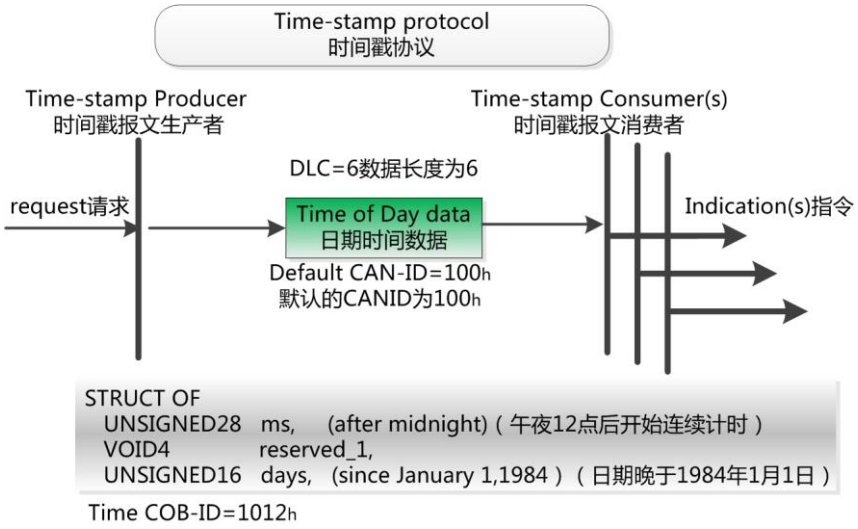


图 9.4 时间戳协议

由于时间换算起来非常费劲，这里特地准备好了换算函数，方便读者使用。

```
void DataTime(void)
{
    uint32 n,i,da;
  canopen_data;//定义网络的日期;
  canopen_msecond;//定义网络的时间（ms）
    /*计算年*/
    n= (canopen_data+671)/1461;//求有多少个 2 月 29 日
    Year=(canopen_data-n)/365+1984;//得到年
    if((canopen_data-n)%365==0){
          Year=Year-1;
    }
    /*计算月日*/
    if((Year%400==0)||(Year%4==0&&Year%100!=0))//判断闰年
    {
       m[2]=29;
    }else{
       m[2]=28;
    }
    da=canopen_data-((canopen_data-n)/365)*365-n;//减去该年前的前天，得到改年的第几天
    for(i=0;i<12;i++)
    {
          if(da>m[i])
          {
                da=da-m[i];//让天数减去每个月的天数
          }
          if(da==0)
          {
                Month=i+1;
                Day=m[i+1];
          }
          if(da<=m[i+1]&&da!=0)//如果得到小于或等于该月的天数，则可以确定为哪一天
          {
                Month=i+1;
                Day=da;
                break;
          }
    }
    /*计算时分秒*/
    canopen_msecond=canopen_msecond/1000;//把毫秒转换为秒
    Hour=canopen_msecond%(3600*24)/3600;
    Minute=canopen_msecond%3600/60;
    Second=canopen_msecond%60;
}
```

9.3 紧急报文协议（Emergency protocol）

紧急事件对象（Emergency），是当设备内部发生错误，触发该对象，**发送设备内部错误代码，提示 NMT 主站**。紧急报文属于诊断性报文，一般不会影响 CANopen 通讯，**其 CAN-ID 存储在 1014_h 的索引中，一般会定义为 080_h +node-ID，数据包含 8 个字节，**如图 9.5 所示。

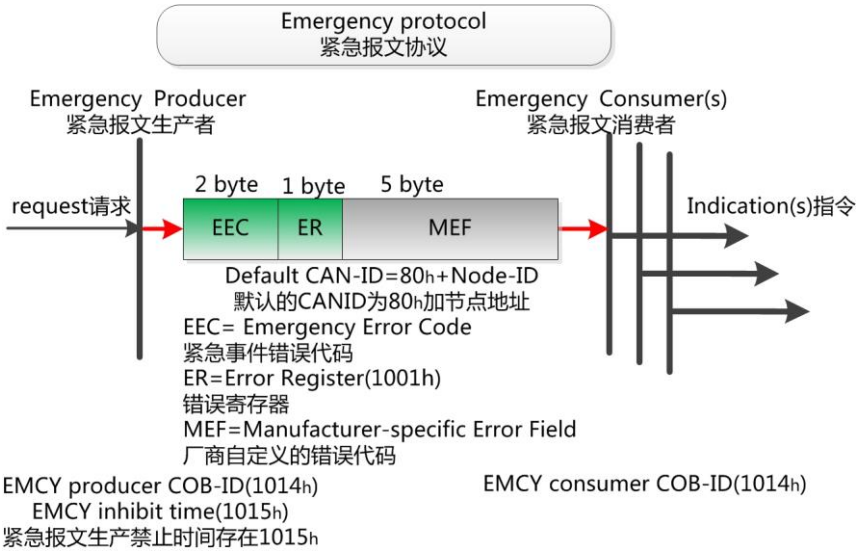


图 9.5 紧急报文

其中包括 EEC：紧急时间错误代码，ER：错误寄存器，MEF：厂商自定义的错误代码。当然这些都需要查表才能获知，进行诊断。

表 9.1 Emergency error codes（紧急报文错误代码）

$00xx_h$	**Error reset or no error 错误复位或没有错误**	$60xx_h$	**Device software 软件错误**
$10xx_h$	**Generic error 一般错误**	$61xx_h$	Internal 内部
$20xx_h$	**Current 电流错误**	$62xx_h$	User 用户
$21xx_h$	Device input side 设备输入端	$63xx_h$	Data set 数据设置
$22xx_h$	Inside of device 设备内部	$70xx_h$	**Additional modules 辅助设备错误**
$23xx_h$	Device output side 设备输出端	$80xx_h$	**Monitoring 监视错误**
$30xx_h$	**Voltage 电压错误**	$81xx_h$	Communication 通讯
$31xx_h$	Main 主供电	8110_h	CAN overrun CAN 通讯超载
$32xx_h$	Inside of device 设备内部	8120_h	Error Passive 错误被动
$33xx_h$	Output 输出	8130_h	Life Guard Error 节点守护错误
$40xx_h$	**Temperature 温度错误**	8140_h	Recovered from Bus-off 总线关闭恢复
$41xx_h$	Ambient 环境	$82xx_h$	**Protocol error 协议错误**
$42xx_h$	Device 设备	8210_h	PDO not processed PDO 没有处理
$50xx_h$	**Device hardware 硬件错误**	8220_h	Length exceeded 长度越界
		$90xx_h$	**External error 外部错误**
		$F0xx_h$	**Additional functions 附加功能错误**
		$FFxx_h$	**Device-specific 设备特定的错误**

与 PDO 的生产禁止时间类似，紧急报文也有生产禁止时间，存储在对象字典的 1015_h 中，为了限制节点不断发送紧急报文，导致总线负载过大。

10. 免责声明

本文档相关资料版权均属广州致远电子股份有限公司所有，其产权受国家法律绝对保护，未经本公司授权，其它公司、单位、代理商及个人不得非法使用和拷贝，否则将受到国家法律的严厉制裁。

本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其它方式授予任何知识产权许可。

本文档可能包含某些错误，一经发现将收入勘误表，并因此可能导致产品与已出版的规格有所差异。如客户索取，可提供最新的勘误表。

本文档中提及的含有订购号的文档以及其它致远电子文献可通过访问广州致远电子股份有限公司的万维网站点获得，网址是：www.zlg.cn。

广州致远电子股份有限公司保留在任何时候修订本手册且不需通知的权利。

销售与服务网络

广州致远电子股份有限公司

地址：广州市天河区车陂路黄洲工业区 7 栋 2 楼
邮编：510660
网址：www.zlg.cn

全国销售与服务电话：400-888-4005

全国服务电话：400-888-4005

销售与服务网络：

广州总公司

广州市天河区车陂路黄洲工业区 7 栋 2 楼
电话：(020)28267985　22644261

北京分公司

北京市海淀区知春路 108 号豪景大厦 A 座 19 层
电话：(010)62536178　62635573

深圳分公司

深圳市福田区深南中路 2072 号电子大厦 12 楼
电话：(0755)83640169　83783155

武汉分公司

武汉市洪山区广埠屯珞瑜路 158 号 12128 室（华中电脑数码市场）
电话：(027)87168497　87168397

成都分公司

成都市一环路南二段 1 号数码科技大厦 403 室
电话：(028)85439836　85432683

上海分公司：上海

上海市北京东路 668 号科技京城东楼 12E 室
电话：(021)53865521　53083451

上海分公司：南京

南京市珠江路 280 号珠江大厦 1501 室
电话：(025)68123923　68123920

上海分公司：杭州

杭州市天目山路 217 号江南电子大厦 502 室
电话：(0571)89719491　89719493

重庆分公司

重庆市九龙坡区石桥铺科园一路二号大西洋国际大厦（赛格电子市场）2705 室
电话：(023)68796438　68797619

西安办事处

西安市长安北路 54 号太平洋大厦 1201 室
电话：(029)87881295　87881296

请您用以上方式联系我们，我们会为您安排样机现场演示，感谢您对我公司产品的关注！