TTCANopen

轻奢车规级 CANFD 转 USB

使用说明书(Av6.0)



TTCANopen 工作室

目 录

. 产品升级说明 4
1. 与消费类产品的差异 5
2. 接口和主要指标 9
3. 设备接口说明 11
4. 安装驱动 13
5. CANFD 参数配置 14
6. 普通调试模块 18
7. 普通工作模块 25
8. 使用指令配置模块参数 30
9. TTCANopen 调试模块 42
10. TTCANopen 工作模块 47
11. TTCANopen 过滤器 50
12. TTCANopen 心跳器模块 52
13. CANFD 数据域不加速发送 56
14. 模块 TTCANopen 特性 58

附录 1. TTCANopen 协议 CAN 标识的分配方法 59

附录 2. TTCANopen 协议设备内部多条指令的发送规则.... 65

免责声明..... 67

风险提示..... 67

产品升级说明

1）2026 年 10 月 1 日，对原来的消费类 CANFD 转 USB 模块 V5.0 进行了全面的升级，不仅采用了车规级器件，并对电路原理图进行了改进，同时对指令格式进行完善。加强了软件对错误处理的能力，整体提升了系统的可靠性。当前模块升级为轻奢车规级 CANFD 转 USB 模块 AV6.0 版本。

（注: 在不影响用户理解的前提下，新版本的使用说明书沿用了旧版本的部分插图）

1. 与消费类产品的差异

为了帮助前期使用过消费类产品的用户可以快速转型到本轻奢车规级模块，本章主要阐述两者的主要区别。

1.1 元器件配料升级为车规级

轻奢车规级 CANFD 转 USB 模块采用了车规级器件，配料表见图 1 所示。

1.2 修改了部分电路逻辑图

1）轻奢车规级电路逻辑图中，新增了一条从 MCU 的 PB4 管腿到 CAN 接口芯片第 8 管腿的控制线。当设备上电和更改 CAN 配置时，MCU 将该管腿电平拉高，禁止 CAN 接口芯片的输出，避免了在此期间模块对 CAN 总线的干扰。

2）轻奢车规级模块采用了供电电压为 5V，参考电压为 3.3V 的 CAN 接口芯片。

3）ISP 接口的 3V 和 5V 引腿，功能进行了互换。

4）SPI 接口原来的 NC 空闲引脚，接入了 5V，给对接板提供了两种工作电压的选择。

1.3 修改指令帧格式提升其鲁棒性

修改指令帧格式的原因见图 2 所示，即在收发指令的结尾新增了一个字节“数据域长度”。由此提升指令格式的鲁棒性，但也造成了

轻奢车规级模块与消费类模块指令不兼容，用户需适应新的指令格式，并使用新开发的上位机软件 MY_CANFD_AUTO。

1.4 MY_CANFD_AUTO 轻奢车规级上位机新增功能

1）为了加强帧格式的鲁棒性，无论是发送帧还是接收帧的帧尾都增加了一个字节的“数据域长度”。

2）新增了将配置信息写入 flash 的功能，设备上电直接使用 flash 中的信息启动模组或模块 。

3）新增了参数配置寄存器观察按钮。

4）新增了 TTL 串口波特率设置。

5）新增了 ID 过滤器配置窗口内容，由原来的 3 个，增加到 6 个。

1.5 模块的 CANFD 转 TTL 串口和 CANFD 转 SPI 功能

由于后续将要推出“轻奢车规级 CANFD 邮票模组”，这两项功能也将在其上进行体现和描述。为了提高 TTL 串口和 SPI 通信的可靠性，其通信协议较消费类模块新增了 32 位 CRC 校验和出错重传机制，由于其内容篇幅较多，具体请参阅“轻奢车规级 CANFD 邮票模组使用说明书”，本说明书只针对 CANFD 转 USB 部分进行阐述。

1.6 模块调试

模块可以使用普通串口工具进行调试，也可以使用为其新开发的 MY_CANFD_AUTO 上位机软件。在消费类模块使用说明书中是用普通串口工具进行描述的，本说明书将改为用 MY_CANFD_AUTO

上位机软件进行描述。

轻奢车规级模组(模块)配料表

序号	元器件名称	品名	规格	个数	备注
1	MCU 芯片(GD)	GD32A103TBU7	QNF36	1	车规
2	CANFD 接口(SIT)	SIT1051AQT/3	SOP8	1	车规
3	三端稳压(LRC)	S-LR1117DS33X	SOT223	1	车规
4	12M 晶体(杨兴)	YSX321SC	SMD3225-4P	1	车规
5	10k 电阻(国巨)	AC0805FR-0710KL	0805	3	车规
6	60.4 电阻（国巨）	AC0805FR-0760R4L	0805	3	车规
7	27.4 电阻（国巨）	AC0805FR-0727R4	0805	2	车规
8	100n 电容（国巨）	AC0805KRX7R9BB104	0805	8	车规
9	12P 电容（国巨）	AC0402FRNPO9BN120	0402	2	车规
10	10u 钽电容（湘江）	CA55-A016M106T	A 型	2	聚合物
11	印制板（生益）	FR-4	44 x36 x1.6mm	1	TG155
				以上为模组配套	
				以下为模块配套另加	
12	USB-A 头			1	普通
13	3p 弯针			1	普通

注 1：本车规级产品由于生产规模和运营成本等客观条件限制，并未以整体模组形式向相关部门申办认证，故在车规级前标注“轻奢”以示区别。本品未涂三防漆，需嵌入用户系统后，由用户统一完成。本品在出厂时做了 100 摄氏度高温 72 小时抽检，和外观检测。其它相关环境测试也需在嵌入用户系统后，由用户随系统统一测试完成。友情提示：车规级 GD32A103TBU7 芯片的工作温度范围为：-40 至+105C 摄氏度

注 2: 轻奢车规级产品较消费类产品有着更为广泛的应用适配场景，注意需当遵守 GD 公司对其芯片及衍生品的管理规范如下：

同意遵守原厂产品及其衍生品的出口、再出口、进口、转让、分销及使用等所有适用的出口管制与经济制裁法律法规；并保证取得所适用的出口管制法律及法规要求的一切国际及国家出口许可或类似许可；且保证不会将原厂产品及其衍生品直接或间接用于军事用途及被禁止的最终用途等；并且不会直接、间接或在知情的情况下出售、运送或提供原厂产品及其技术信息给军事最终用户及任何列于中国、美国或联合国等有权政府机构禁止或限制清单上的实体。

图 1　轻奢车规级 CANFD 转 USB 模块配料表

轻奢车规级帧格式鲁棒性的讨论

车规级通讯模块对其工作可靠性的提升是多纬度的，不仅体现在对硬件元器件的苛求上，对通讯帧格式的鲁棒性也有更高的要求，

下面我们从两个方面进行扼要的阐述，1）虚警概率；2）断帧误判。

1）虚警概率：一个随机序列，恰好匹配预设的帧格式，从而产生错误判决的概率。

如果只设置了一个字节的帧标识，即：“帧头”，此时的虚警概率为：1/256 ≈ 3.9×10^{-3}；当我们增加一个字节的帧头，或添加一个帧尾字节，虚警概率为：1/256 × 1/256 ≈ 1.5×10^{-5}；当我们设置的帧标识字节越多虚警概率就会越低。在一个高可靠的系统中其虚警概率至少要达到 10^{-10} 以下。

下面计算一下在消费类产品中，我们 CANFD 转换帧格式的虚警概率，其帧标识字节主要包括：帧头、帧类型、数据域长度、帧尾。

其虚警概率 = 1/256 × 5/256 × 16/256 × 1/256 ≈ 1.8×10^{-8}；未能达标。

显然我们需要增一个帧标识字节，再 × 1/256 ≈ 7.2×10^{-11}；**使其达标。**

那么这个字节应添加在哪里呢？就是我们下面需要探讨的。

2）断帧误判：发送时，断帧本身是一个小概率事件，但在我们实操过程中的确会发生，比如我们通过文件方式批量发送 CANFD 指令，中途停止发送文件，可能会在结尾出现一个不完整的断帧，此时我们又启动发送一个新帧，新帧的某个字节正处在断帧预判的帧尾位置，如果这个字节刚好与帧尾标识符相同时，断帧就会产生误判为完整帧，其概率是 1/256，于是我们得出结论新添加的帧标识字节应该在帧尾最为合理。其可将该误判概率降低到 1/256 × 1/256。**那么问题来了，新添到帧尾的标识字节应该是什么呢？好像问题并不简单。**

假如断帧是一个长帧，新发帧是一个短帧，短帧的结尾和长帧预判的结尾刚好重合，此时无论我们添加多少个不变的帧尾字节，都无法避免该误判。聪明的你，一定想到了解决的妙法，对的，添加到帧尾的帧标识是“数据域长度”，这个标识曾经在帧格式中出现过一次，二次出现意义非凡。

图 2　轻奢车规级帧格式结尾新增“数据域长度”字节

2. 接口和主要指标

模块遵循 CAN 总线协议 2.0A、2.0B 和 ISO11891-1:2015 CANFD 规范，与上位机通讯形式为(USB)虚拟串口，可以在支持 CDC 类的操作系统上使用，如：WIN、LINUX、UNIX、MAC、安卓和鸿蒙等（本模块测试是在 WIN10 上进行的）。

本品可作为普通 CANFD 转 USB 模块使用，也可作为 CAN 转 USB 使用。

本品 USB 端口能够监听 5M CANFD 总线 100%满载，不丢帧。

本品主要接口指标如下：

USB 端口通讯: USB2.0 全速(12M)， 虚拟串口，A 型链接器

供电: 采用 USB 端口+5V 供电

CAN 端口链接器: 单排 3P 插针（2.54mm）

CAN 收发器: SIT1051AQT/3

终端电阻: 平衡式终端电阻

CAN 通讯速率 bps: 10k 至 1M （分档）

CANFD 数据速率 bps: 10k 至 6M （分档）

采样点分配：75%，80%，85%，自动，（可选）

ID 过滤器：6 个（只准许指定 ID 通过）

核心 ARM 芯片: GD32A103TBU7

外观尺寸: 60×30×10 mm （含壳）

模块集成: 本品集成有五种子模块

1).普通调试模块、

2).普通工作模块、

3).TTCANopen 调试模块、

4).TTCANopen 工作模块、

5).TTCANopen 心跳器模块

3. 设备接口说明

3.1 ISP 固件加载更新接口

本品为用户提供了 ISP 固件加载更新接口，见图 3 所示，用户可以从 www.ttcanopen.com.cn 网站获取产品的最新测试用固件，使用 ISP 下载器将其安装到设备上（当然用户也可以将自己研发的固件安装到设备上）。注意测试用固件包含了测试用例和时间限制，出厂正式固件已经将其去除。（注意：带外壳的出厂产品，外壳是一次性压接的，开启外壳时，对接销容易损坏，非必要不建议用户对出厂成品更新固件。）

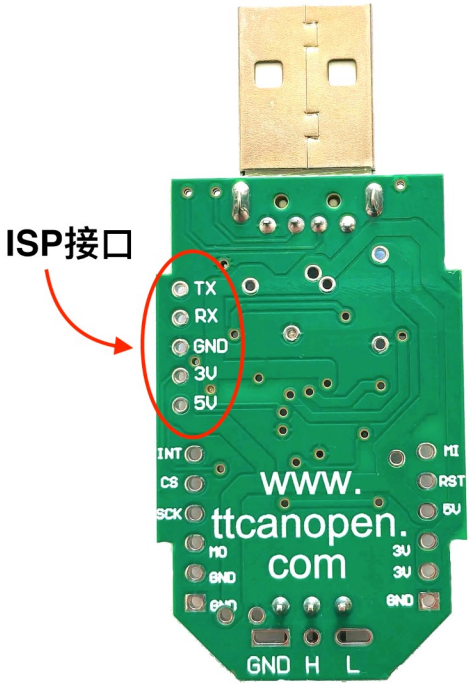


图 3　设备 ISP 固件更新接口

ISP 接点从上至下分别为：TX，RX，GND，BOOT，5V。注意：上电时 BOOT 为 3V 时，芯片进入烧录状态。

用户需通过 GD 公司提供的 GD32AllinOneProgrammer 软件，为模块安装固件。

用户可以使用市面上多数 ISP 下载器安装设备固件，使用时请注意插针的排序，连接不当可能会损毁设备。

用户也可以到 TTCANopen 淘宝网店购买设备专用的 ISP 下载器，其使用非常方便，只需将专用 ISP 下载器的 5 根插针直接从背面插入到设备的对应 ISP 接口上即可，见图 4 所示。

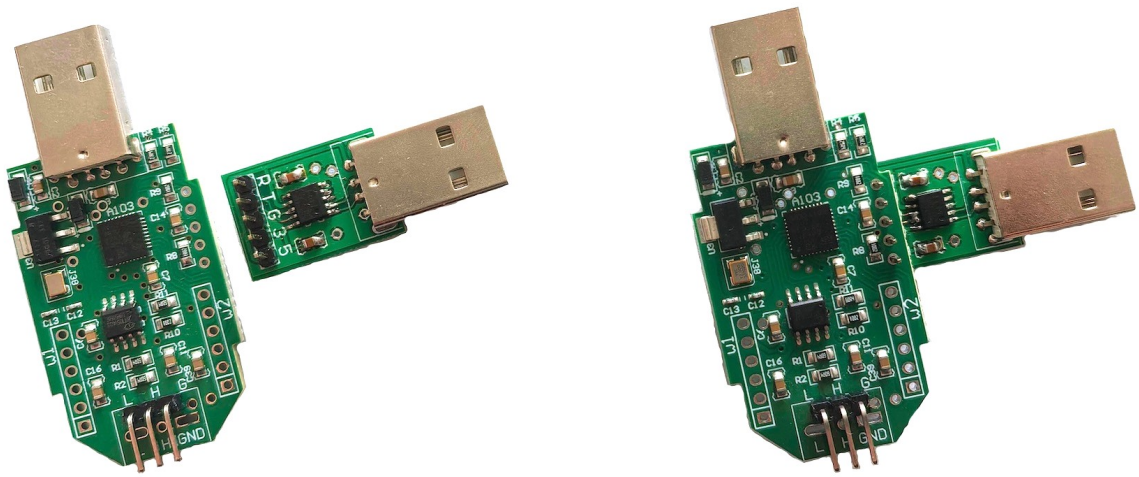


图 4 专用 ISP 下载器的连接

3.2 CAN(FD)端口

模块 CAN(FD)接口有三个接线端子，见图 5 所示，分别是 CAN_L（左）、CAN_H（中）总线信号和 GND（右）地线，通常总线信号线为双绞线，地线可选（多为屏蔽地）。

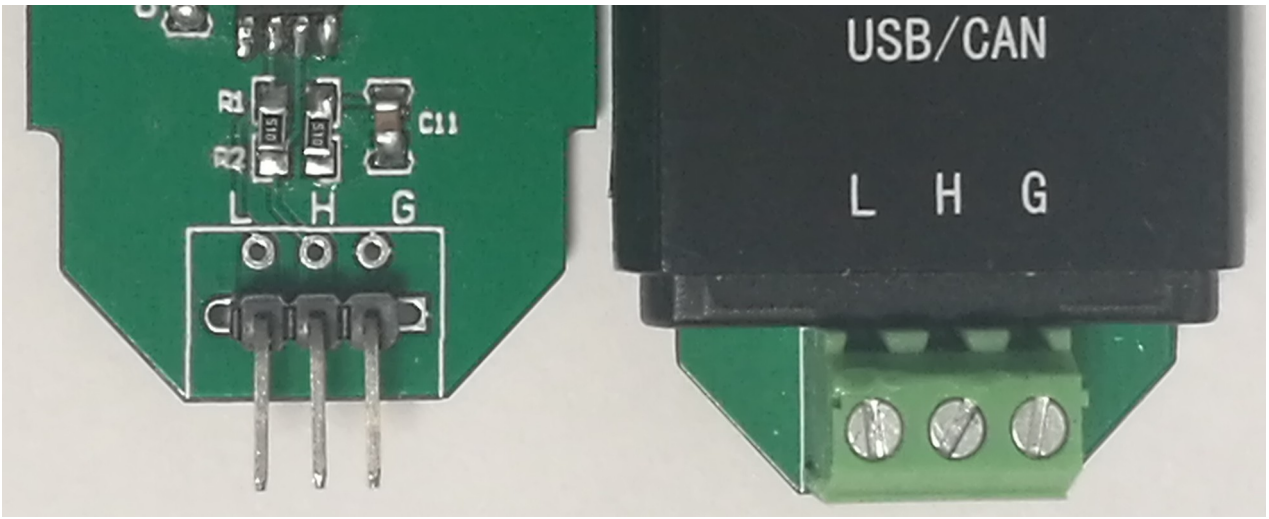


图 5 CAN 端口接线端子

4. 安装驱动（Windows）

WIN10 免驱动（LINUX、UNIX、MAC、安卓和鸿蒙免驱），在 WIN7 和 XP 上，PC 机需安装由 GD 公司提供的虚拟串口驱动。（注意：请不要同时安装其它公司的虚拟串口驱动）

5. CANFD 参数配置

将模块插入 PC 机的 USB 接口，运行 MY_CANFD_AUTO,

点击“配置参数”按钮，出现 CANFD 配置参数对话框，用户可以选择“CANFD 参数设置（串口）”，或“用指令设置 CANFD 参数”。其后者比前者可配置的参数更为齐全，是重点推荐用户使用的方式。

5.1 “CANFD 参数设置（串口）”选项配置 CANFD 参数（见图 7）

在 USB 虚拟串口状态，模块将串口参数映射成 CANFD 参数，具体如下：

波特率：设置 CAN 总线通讯速（单位 kbps），可输入 10、20、50、100、125、250、500、800、和 1000。

校验位： 选择功能模块（注：产品集成了 5 种功能模块）

NONE： 普通调试模块

ODD： 普通工作模块

EVEN： TTCANopen 调试模块

MARK： TTCANopen 工作模块

SPACE：TTCANopen 心跳器模块

数据位： 选择 CANFD 数据域发送倍率（CANFD 状态有效）

8: 2 倍， 7: 4 倍， 6: 6 倍

停止位：设置模块状态（每种模块都包含三种工作状态）

1: CAN 状态， 1.5: 静听状态 2: CANFD 状态

配置操作如下：

1）点击“配置参数”按钮，出现“CANFD 参数配置等....”对话框。

2）选择“CANFD 参数设置（串口）”子对话框，在这里使用 4 个下拉菜单选择 CANFD 参数。

3）关闭配置对话框，点击“⊙串口”按钮，打开串口，完成配置。

4）接收窗口会出现帮助信息和当前配置参数等。



图 7 “CANFD 参数设置（串口）”选项配置 CANFD 参数

5.2 “用指令设置 CANFD 参数”选项配置 CANFD 参数（见图 8）

该方式是重点推荐给用户的 CANFD 参数配置方法，配置前需先打开串口，在各下拉菜单中选择好 CANFD 参数后，点击下面的大长按钮“在串口打开或网络链接后，点击本按钮发送配置指令”，当鼠标箭头移动到该按钮上时，按钮上会显示即将发送的“配置指令”。点击指令发出后接收窗口会出现帮助信息和当前模块的配置状态。

配置操作如下：

1）点击“⊙串口”按钮，打开串口。

2）点击“配置参数”按钮，出现“CANFD 参数配置等….”对话框。

3）选择“用指令设置 CANFD 参数”子对话框，在这里使用 6 个下拉菜单选择 CANFD 参数。

4）将鼠标移动到“在串口打开或网络链接后，点击本按钮发送配置指令”按钮上，此时按钮上的文字会变为即将发送的配置指令，点击后，配置指令发出。

5）接收窗口会出现帮助信息和当前配置参数等。

6）关闭“CANFD 参数配置等….”对话框。

7）如果需要将当前配置信息写入 flash 存储，左键点击“ W ”按钮，此后设备上电启动，模块会首先读取 flash 中的配置信息设置模块，并屏蔽“CANFD 参数设置（串口）”功能，打开串口，模块直接使用 flash 中的配置运行。无需像消费类模块那样每次上电启动，都

要重新配置 CANFD 参数。

（注：鼠标右键点击“ W ”按钮，可清除 flash 中的配置信息）

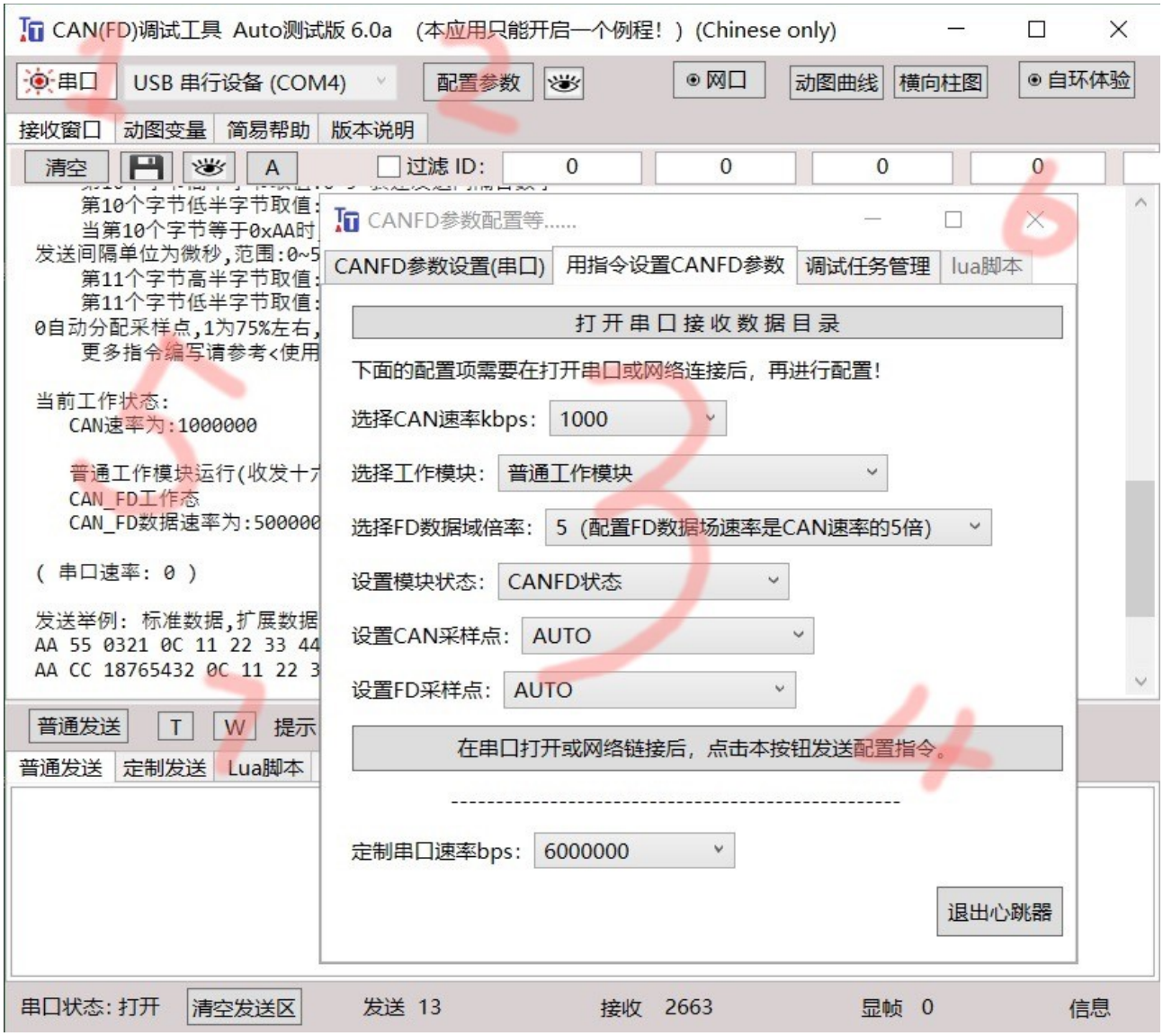


图 8 “用指令设置 CANFD 参数”选项配置 CANFD 参数

6. 普通调试模块

普通调试模块，是用户在调试已有 CAN 或 CANFD 网络时，临时加入的调试设备，待调试完成后脱离原网络。其不是原 CAN 网络的组成部分。

用户可以将模块配置为 CAN 状态或 CANFD 状态，使其与被调试的 CAN 网络状态相匹配，当用户只需对原网络进行监听时，可将模块配置为静听态，这样对原网络的影响最小，并能同时静听原网络中的 CAN 指令和 CANFD 指令。

为了方便用户观察原网络中指令的运行情况，普通调试模块虚拟串口输出为字符串格式，每帧指令独占一行，用户可使用各种串口调试工具显示调试，也可以使用 MY_CANFD_AUTO 上位机软件进行调试。

普通调试模块支持 4 种指令帧的显示，其格式如下：

1）显示标准遥控帧指令格式：

帧计数	接收时标	帧标识	ID 字	DLC	帧标识
00000000 至 FFFFFFFF	时:分:秒:毫秒.微妙	<	0000 至 07EF	0 至 8	>

2）显示标准数据帧指令格式：

帧计数	接收时标	帧标识	ID 字	DLC	帧标识	数据场 DLC 个字节
00000000 至 FFFFFFFF	时:分:秒:毫秒.微妙	[	0000 至 07EF	0 至 8	]	xx ~ xx

3）显示扩展遥控帧指令格式：

帧计数	接收时标	帧标识	ID 字	DLC	帧标识
00000000 至 FFFFFFFF	时:分:秒:毫秒.微妙	<	00000000 至 1FBFFFFF	0 至 8	>

4）显示扩展数据帧指令格式：

帧计数	接收时标	帧标识	ID 字	DLC	帧标识	数据场 DLC 个字节
00000000 至 FFFFFFFF	时:分:秒:毫秒.微妙	[	00000000 至 1FBFFFFF	0 至 8	]	xx ~ xx

注：上述格式是按普通 CAN 状态表述的，CANFD 状态不支持遥控帧，CANFD 指令中 DLC 字段和其所对应的数据场长度，遵守 CANFD 协议相关规定。DLC 字段用 10 进制显示。

普通调试模块接收指令显示举例：

1）标准遥控帧：

00000001 00:00:04:727.279 < 0123 8 >

2）标准数据帧：

00000002 00:00:04:727.390 [0321 4] 11 22 33 44

3）扩展遥控帧：

00000003 00:00:04:727.490 < 12345678 6 >

4）扩展数据帧：

00000004 00:00:04:727.615 [18765432 3] 11 22 33

上例中:（举例:扩展数据帧）

第一个字符串“00000004”：

为接收帧计数，计数范围 0～0xFFFFFFFF；

第二个字符串“00:00:04:727.615”：

为接收时标，时：分：秒：毫秒.微妙，分辨率为 1 微秒；

第三个[扩起来]的字符串“[18765432 3]”：

为帧 ID 和 DLC，(ID 为 32 位十六进制数,低 29 位有效。DLC 为十进制数，CAN 为 0 至 8；CANFD 为 0～8、12、16、24、32、

48、64，遵守 CANFD 协议相关规定)；<尖括号>则表述遥控帧；

第四个字符串“11 22 33 ”：

为数据帧中 DLC 个字节数据内容。

（注：对于标准帧，ID 为 16 位十六进制数，低 11 位有效）

图 9 为在 MY_CANFD_AUTO 上位机接收窗口观察到的四种帧举例，分别为：标准遥控帧、标准数据帧、扩展遥控帧和扩展数据帧。



图 9 普通调试模块接收显示举例

普通调试模块支持 4 种指令帧的发送，并对发送帧进行“回显”，发送指令使用十六进制方式，其格式如下：

1）发送标准遥控帧指令格式：

帧头	帧标识	ID 字	DLC	帧尾	DLC
AA	77	0000 至 07EF	00 至 08	DD	00 至 08

2）发送标准数据帧指令格式：

帧头	帧标识	ID 字	DLC	数据场 DLC 个字节	帧尾	DLC
AA	55	0000 至 07EF	00 至 08	xx ~ xx	DD	00 至 08

3）发送扩展遥控帧指令格式：

帧头	帧标识	ID 字	DLC	帧尾	DLC
AA	FF	00000000 至 1FBFFFFF	00 至 08	DD	00 至 08

4）发送扩展数据帧指令格式：

帧头	帧标识	ID 字	DLC	数据场 DLC 个字节	帧尾	DLC
AA	CC	00000000 至 1FBFFFFF	00 至 08	xx ~ xx	DD	00 至 08

注：CANFD 不支持遥控帧，CANFD 指令中 DLC 字段和其所对应的数据场长度，遵守 CANFD 协议相关规定，其有效值为：0x00～0x08、0x0C、0x10、0x18、0x20、0x30、0x40。

普通调试模块发送指令举例（十六进制方式）：

1）发送标准遥控帧：

AA 77 0123 08 DD 08

2）发送标准数据帧：

AA 55 0321 08 11 22 33 44 55 66 77 88 DD 08

3）发送扩展遥控帧：

AA FF 12345678 06 DD 06

4）发送扩展数据帧：

AA CC 18765432 08 11 22 33 44 55 66 77 88 DD 08

图 10 为 MY_CANFD_AUTO 上位机中普通调试模块发送指令和回显示例，分别为：标准遥控帧、标准数据帧、扩展遥控帧和扩展数据帧。



图 10 普通调试模块发送指令和回显示例

7. 普通工作模块

普通工作模块，是已有 CAN 或 CANFD 网络原生的组成部分，其主要承担 CAN（FD）网络与系统上位机的通讯连接和协议转换，是中心监控系统的交通枢纽。

用户可以将模块配置为 CAN 状态或 CANFD 状态，使其与固有 CAN 网络状态保持一致。

为了提高指令效率，模块虚拟串口输入输出皆为 16 进制格式，帧与帧之间使用帧头帧尾标识。

普通工作模块支持 4 种指令帧的输入输出，其中，发令格式与普通调试模块相同，但普通工作模块不“回显”发送指令。

普通工作模块接收指令格式如下：

1）接收标准遥控帧指令格式：

帧头	接收时标	帧标识	ID 字	DLC	帧尾	DLC
BB	tt tt tt tt	77	0000 至 07EF	00 至 08	EE	00 至 08

2）接收标准数据帧指令格式：

帧头	接收时标	帧标识	ID 字	DLC	数据场 DLC 个字节	帧尾	DLC
BB	tt tt tt tt	55	0000 至 07EF	00 至 08	xx ~ xx	EE	00 至 08

3）接收扩展遥控帧指令格式：

帧头	接收时标	帧标识	ID 字	DLC	帧尾	DLC
BB	tt tt tt tt	FF	00000000 至 1FBFFFFF	00 至 08	EE	00 至 08

4）接收扩展数据帧指令格式：

帧头	接收时标	帧标识	ID 字	DLC	数据场 DLC 个字节	帧尾	DLC
BB	tt tt tt tt	CC	00000000 至 1FBFFFFF	00 至 08	xx ~ xx	EE	00 至 08

注 1：tt tt tt tt 为 32 位接收时间计数（计数单位为 0.1ms），其低字节在前，高字节在后；而 ID 字是高字节在前，低字节在后，解帧时需特别留意。

注 2：CANFD 不支持遥控帧，CANFD 指令中 DLC 字段和其所对应的数据场长度，遵守 CANFD 协议相关规定，其有效值为：0x00～0x08、0x0C、0x10、0x18、0x20、0x30、0x40。

普通工作模块接收指令举例：

1）标准遥控帧：

BB B5 3C 02 00 77 01 23 08 EE 08

2）标准数据帧：

BB B6 3C 02 00 55 03 21 08 11 22 33 44 55 66 77 88 EE 08

3）扩展遥控帧：

BB B7 3C 02 00 FF 12 34 56 78 06 EE 06

4）扩展数据帧：

BB B9 3C 02 00 CC 18 76 54 32 08 11 22 33 44 55 66 77 88 EE 08

图 11 为普通工作模块，在普通串口调试工具指令接收示例，分别为：标准遥控帧、标准数据帧、扩展遥控帧和扩展数据帧，在这里每帧指令并没有独立成行，而是用帧头 BB 和帧尾 EE+DLC 进行分割的。

由于 MY_CANFD_AUTO 上位机软件中对接收帧进行了帧同步

操作，同步后在 MY_CANFD_AUTO 上位机软件接收窗口中，每一帧都是独立成行显示的，更加便于观察见图 12。

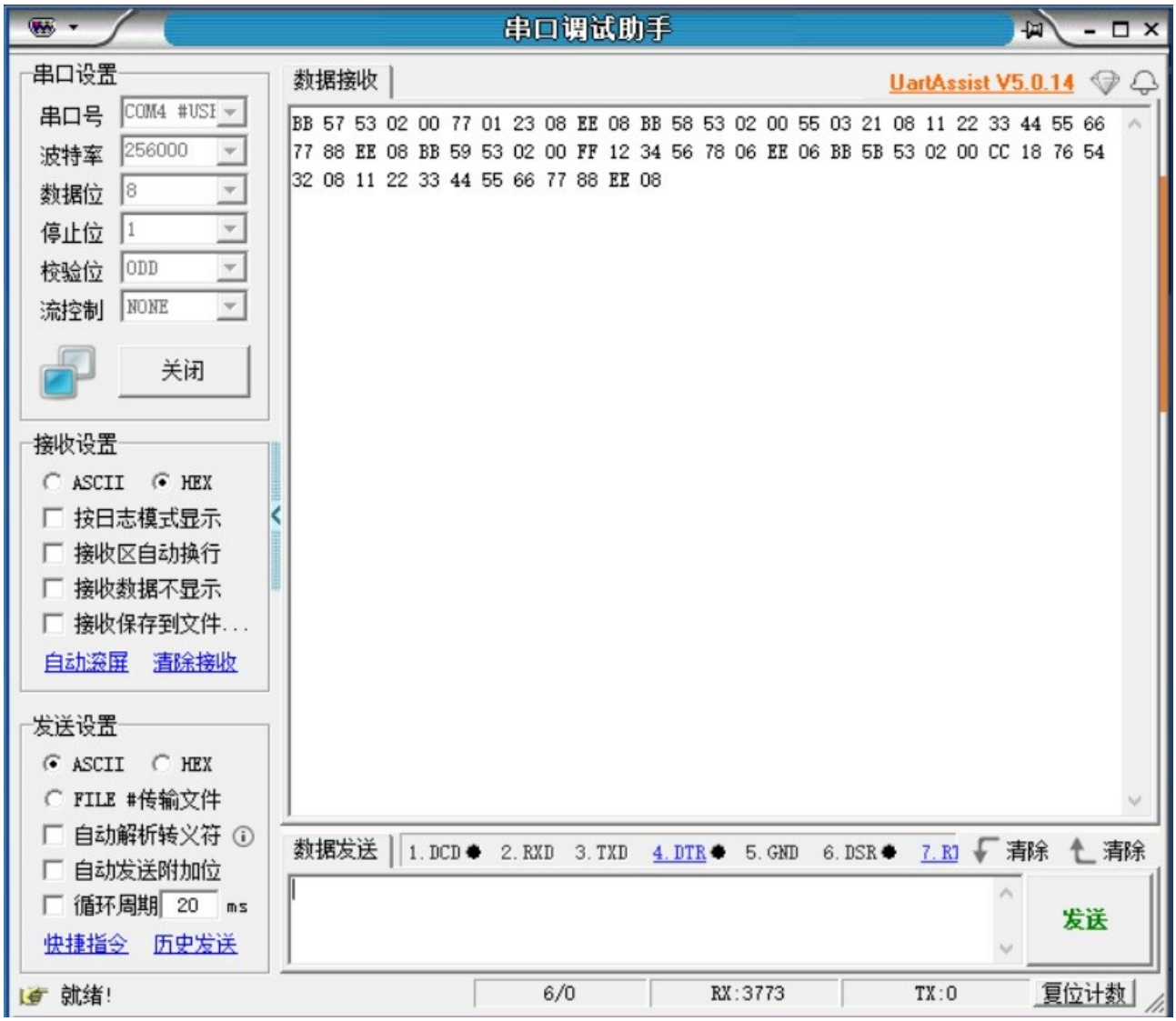


图 11 普通工作模块在串口调试工具接收指令示例



图 12 普通工作模块在 MY_CANFD_AUTO 上位机接收指令示例

8. 使用“指令”配置模块参数

模块除了可以使用串口配置项定义模块的 CAN（FD）配置参数和工作状态，也可以使用专用指令进行上述操作。

在 MY_CANFD_AUTO 的“用指令设置 CANFD 参数”中，当我们将鼠标箭头移动到大长按钮“在串口打开或网络链接后，点击本按钮发送配置指令”上时，按钮上会显示即将发送的该配置指令序列。

实际上，在模块内部分配了一个固有的寄存器空间来存放模块的配置参数，用户可以通过修改该寄存器中的内容，完成配置参数和设置工作状态。表 1 为该寄存器中相关参数的分配状况。

表 1 模块固有的配置参数寄存器空间的分配

序号	寄存器地址	字节数	参数描述
1	FF00	1	当该值置 1 时，启动配置，然后自动归 0.
2	FF01	1	选择工作模块： 0:为普通调试模块 1:为普通工作模块 2:为 TTCANopen 调试模块 3:为 TTCANopen 工作模块 4:为 TTCANopen 心跳模块
3	FF02	1	配置工作状态：

			0:为 CAN 状态 1:为静听态 2:为 CANFD 态
4	FF03	1	配置 CAN 和 CANFD 速率： 低半字节配置 CAN 速率 1～9 分别对应 CAN 速率的 10、20、50、 100、125、250、500、800、1000 kbps 高半字节配置 CANFD 的倍率 1～A 分别对应 CANFD 为 CAN 速率的 1～10 倍
5	FF04	1	设置两帧发送间隔（微妙） 高半字节为发送间隔首数据， 低半字节为 0 的个数。 注意：当该字节被设置为 0xAA 时 将由模块根据通讯速率自动生成发送间隔
6	FF05	1	选择配置采样点 低半字节配置 CAN 采样点 高半字节配置 FD 采样点 0:为自动配置采样点 1:为 75%左右 2:为 80%左右

			3:为 85%左右
7	FF06～	1	保留

为了能够设置和修改该寄存器内容，设计了专用指令格式如下：

设置模块配置参数寄存器指令格式：

帧头	帧标识	寄存器地址	DLC	数据内容	帧尾	DLC
AA	66	FF00 至 FF06	01 至 06	xx ～ xx	DD	01 至 06

用户配置完参数后，需将 0xFF00 寄存器置 1，新配置方才生效，生效后该寄存器内容自动归零。

配置模块参数举例：

例 1；修改 CAN 发送速率为 1000000bps

AA 66 FF03 01 19 DD 01

AA 66 FF00 01 01 DD 01

例 2:修改 CAN 工作状态为 CANFD

AA 66 FF02 01 02 DD 01

AA 66 FF00 01 01 DD 01

当然用户也可以使用一条指令将配置参数一次完成（推荐）。

例 3:一次配置全部参数，包括配置立即生效

AA 66 FF00 06 01 01 02 59 12 00 DD 06

该指令配置模块为“普通工作模块”、“CANFD 态”、“CAN 速率 1000000bps”、“CAN FD 速率 5000000bps”、“发送间隔 100 微妙”、“自动配置采样点”配置立即生效。

该一次完成指令配置的操作，在 MY_CANFD_AUTO 上位机软件中是以选择下拉菜单和点击按钮的形式完成的，无需用户死记硬背指令，见第 5 章内容。

但在使用普通串口调试工具中，或用户的二次开发中，用户需要使用该指令去配置模块的 CAN（FD）参数，图 13 是在普通串口调试工具中使用配置指令成功反馈显示信息举例。

1）在发送窗口编辑发送指令（HEX）；

2）点击发送按钮；

3）查看接收窗口的帮助信息和配置信息；

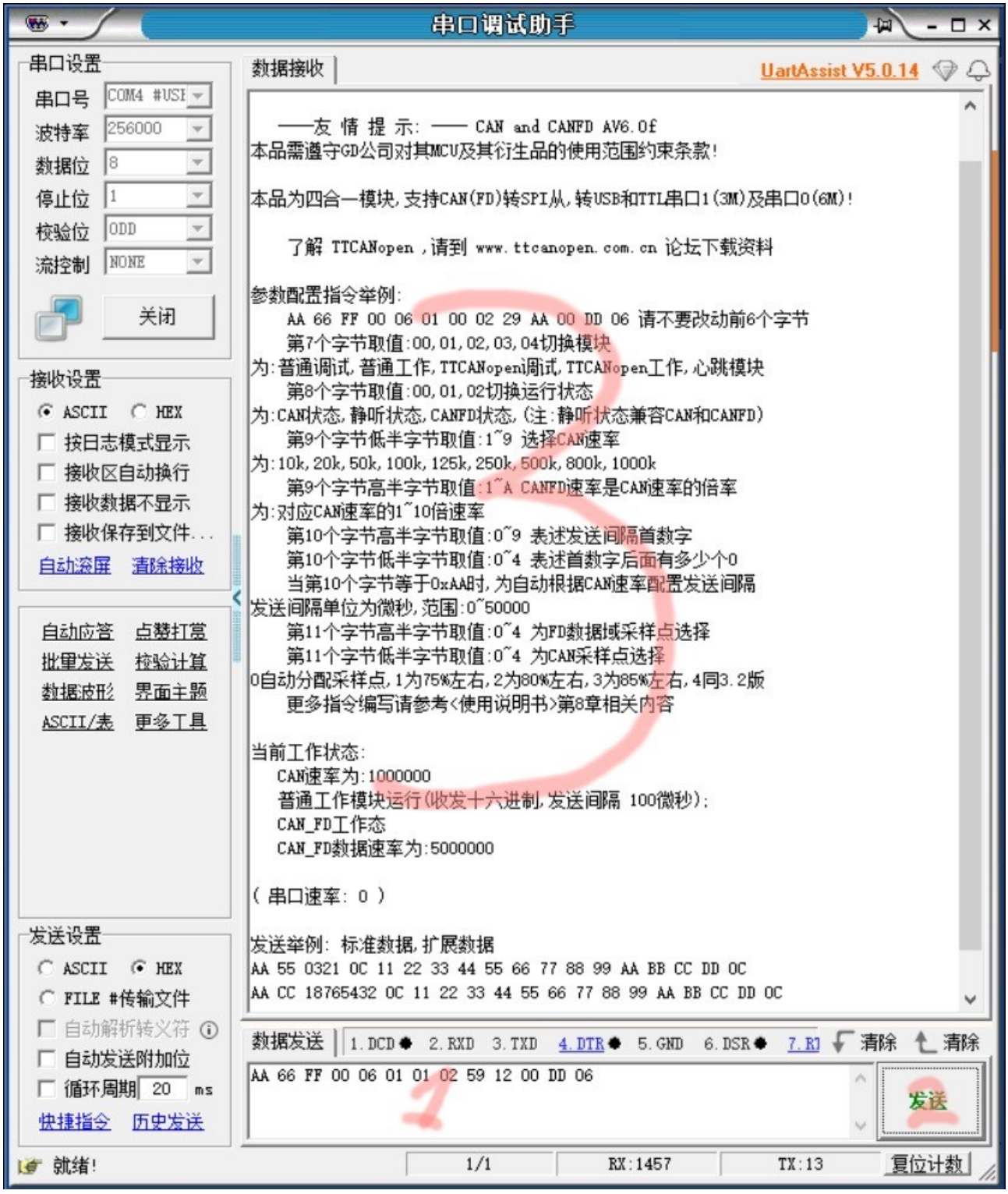


图 13 普通串口调试工具使用配置指令成功反馈显示信息举例

除了上述配置 CANFD 参数的 6 个 1 字节寄存器外，还有 6 个 32 位 ID 过滤寄存器，用户在需要时进行配置。

当需要只显示观查某个或某几个特定 ID 的 CAN（FD）指令时，可以通过配置 ID 过滤寄存器的方式实现，只有与 ID 过滤寄存器中 ID 相同的 CAN（FD）指令才能被挑选出来，其它则被过滤掉。表 2 为 ID 过滤寄存器空间的分配状况。

表 2 模块 ID 过滤寄存器空间的分配

序号	寄存器地址	字节数	参数描述
1	FF10	4	第一个 32 位 ID 过滤寄存器
2	FF14	4	第二个 32 位 ID 过滤寄存器
3	FF18	4	第三个 32 位 ID 过滤寄存器
4	FF1C	4	第四个 32 位 ID 过滤寄存器
5	FF20	4	第五个 32 位 ID 过滤寄存器
6	FF24	4	第六个 32 位 ID 过滤寄存器

每一个 32 位 ID 过滤寄存器中存放一个需要显示观察的 CAN（FD）指令的 ID 值。

ID 过滤寄存器的配置指令与设置模块配置参数寄存器指令格式基本相同，只不过其起始地址是：0xFF10。

用户可以用指令一次性写入这 6 个 32 位 ID 过滤寄存器如下：

AA 66 FF 10 18 xx xx xx xx yy yy yy yy zz zz zz zz mm mm mm mm nn nn nn nn kk kk kk kk DD 18。

也可以配置少于 6 个 ID 过滤，如：只配置 2 个 ID 过滤，

AA 66 FF 10 08 xx xx xx xx yy yy yy yy DD 08

其中，xx xx xx xx 和 yy yy yy yy 等，都是低字节在前高字节在后的无符号 32 位数，代表需要选取的指定 ID 值，这样模块就会只传递这些指定 ID 的帧，其它被忽略。

如：只想看 ID 字等于 0123 的帧，就需要发送配置指令：

AA 66 FF 10 04 23 01 00 00 DD 04

注意：无论是标准帧还是扩展帧都使用 32 位过滤字，在上面的指令发出后，可以看到 ID 字是 0123 的标准帧，也可以看到 ID 字是 00000123 的扩展帧。

另外，需要注意的是过滤 ID 字必须从地址 FF 10 开始配置起，而且第一个过滤 ID 字不能是 0（即 xx xx xx xx 不能是 0）。

如果第一个过滤 ID 字是 0，模块将忽略 ID 过滤功能，将显示全部所收到的帧。

这些 32 位 ID 过滤寄存器是可以通过写配置参数到 flash 一同写入到 flash 中的，每次开机启动其和 CANFD 参数一同被读出，配置到模块中。

为了方便用户查看这些配置寄存器中的内容，在上位机软件

MY_CANFD_AUTO 中，特地在“配置参数”按钮右侧，设置了一个查看按钮“ő”见图 14 所示。

在图 14 中，可以看到“串口速率：0”的内容，这个和 CANFD 转 TTL 串口相关，这里略过。

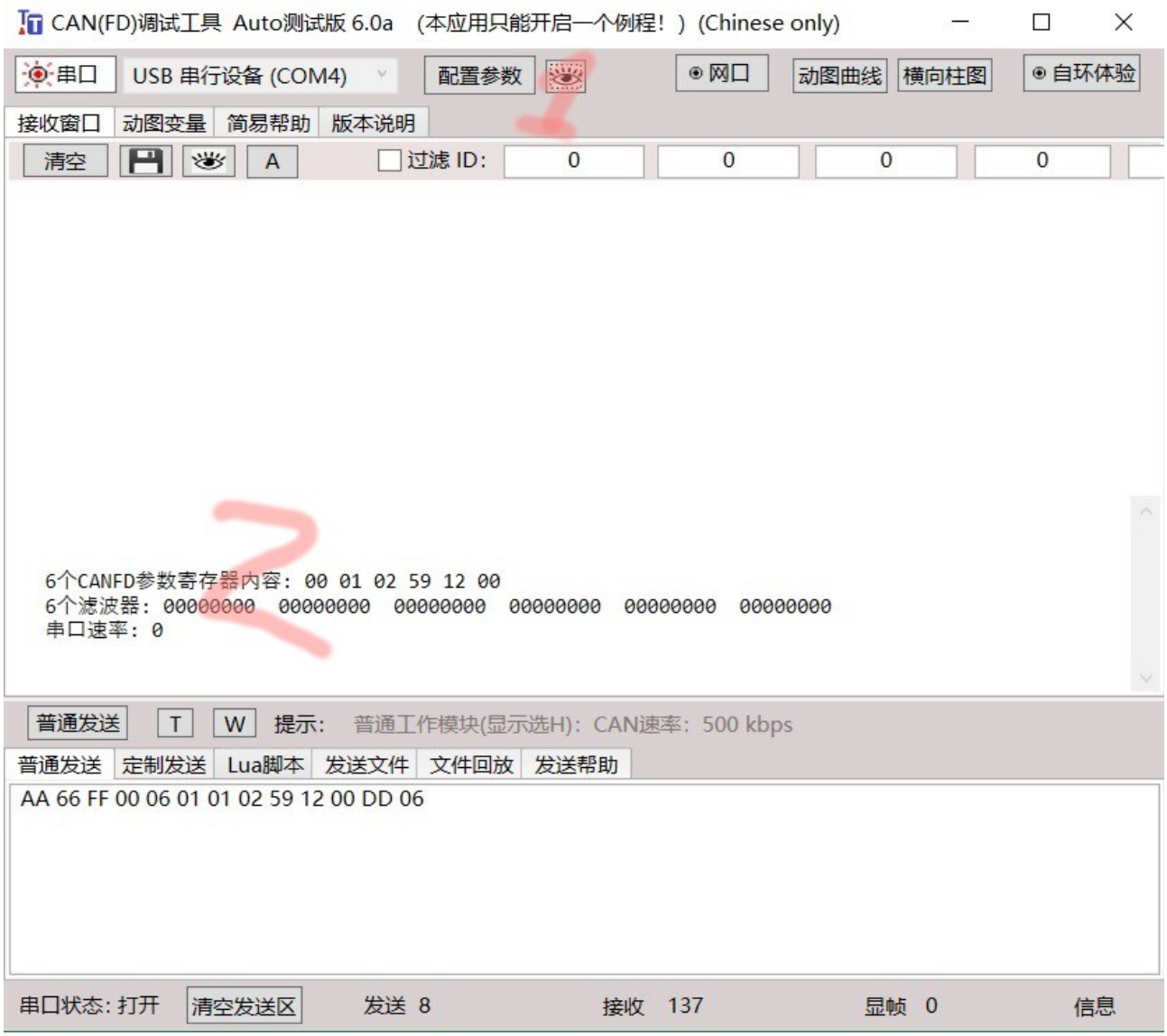


图 14 查看配置参数寄存器内容

在 MY_CANFD_AUTO 中，配置 ID 过滤寄存器是非常方便的，在软件的上端右侧有 6 个 ID 输入窗口，每个输入窗口可以输入一个 16 进制的 32 位无符号数，如我们可以输入 00000123，或 00AB17CD 等，然后点击勾选前面的“过滤 ID”选择框。如果想清除过滤 ID，直接去掉前面的勾选就可以了。（注意：这里过滤 ID 窗口是直接输入一个十六进制的 32 位数，类似 0x00000123，0x00AB17CD，省略了前面的 0x，关注一下它和指令中的差异）。

应用户提出的需要，模块配置了额外的两条指令如下：

1）UID 读取指令，读取到单片机序列号中的一个 32 位数，用户可以通过这个 UID 区分单片机。

指令格式：AA 66 FF 00 01 02 DD 01

发送该指令后，模块会以字符串形式返回一个 32 位数，

如：53535B28

2）配置模块时间指令，在 TTCANopen 应用层协议中，模块时间是由心跳器每秒发送一次主心跳完成总线时间同步的。但对于没有使用 TTCANopen 应用层协议的用户模块，时间是从上电时刻 0 开始计数的。有用户提出需求希望模块时间能够与 PC 机基本同步，为此新增了该指令。

指令格式：AA 66 FF 07 05 01 tt tt tt tt DD 05

其中，tt tt tt tt 是一个 32 位无符号数，低字节在前高字节在后，是一天中以毫秒为计时单位的时刻值。

用户可以先读取 PC 机当前的时间，并将其转换为毫秒计数，发送该指令，这样就可以将模块的时间配置为与 PC 机基本同步。用户可以通过周期性的发送该指令，去修正同步时间的漂移。

在上位机软件 MY_CANFD_AUTO 中，“普通发送”按钮右侧，有一个“ T ”按钮，左键点击将 PC 机时间同步到模块上，右键点击将模块时间置零。

说明书以下部分与 TTCANopen 协议有关，有需求的用户可以继续阅读。

本品为业界首个直接支持 TTCANopen 应用层协议的轻奢车规级 USB 转 CAN_FD 模块，用户可以访问 www.ttcanopen.com.cn 网站获得更多的 TTCANopen 相关知识。

TTCANopen 是由国人打造的一款拥有自主知识产权的 CAN 应用层协议，与以往的应用层协议不同， TTCANopen 实际上是一个开放的协议框架。它是由众多的“基本子协议”和“应用子协议”构成，用户可以根据自己的专业应用特点，去选择“应用子协议”;也可以使用“基本子协议”去生成新的“应用子协议”;
TTCANopen 协议在应用层界面直接兼容最新的 CAN_FD。

TTCANopen 应用层协议以连续寄存器空间为信息载体，使用了两个不同的 CAN 标识符划分的方式分别传递过程变量和非过程变量，巧妙的解决了寄存器空间分层管理与其指令优先级分配不平衡的矛盾。

TTCANopen 还特别规范了设备内部多条指令的发送原则，这是其它应用层协议所不具备的。

TTCANopen 引入了系统心跳的概念，用以同步总线上的设备，使系统拥有统一的时钟，并使用时间相位矩阵取代了传统的指令查询矩阵，提高了总线带宽的利用率，大大的减小了浪涌出现的概率，使应用系统实现了由传统的指令驱动到现代的时间驱动的跨越。

TTCANopen 应用层协议对 29 位 CAN 标识符的划分及其物理含义的分配，如:优先级段、寄存器基地址段、设备编号段、功能码段都具有明确的应用层协议指令含义，据此设计的应用层协议具有很好的直读性，易于解析，大大降低

了学习入门的难度。是一款非常适合推广和普及的应用层协议。

另外，TTCANopen 有着类 modbus 特性，使其能够非常容易的和市面上大多数组态软件完美对接。为用户构建应用系统提供了极大的便利。

9. TTCANopen 调试模块

TTCANopen 调试模块，是用户在调试已有 TTCANopen 网络时，临时加入的调试设备，待调试完成后脱离原网络。其不是原 TTCANopen 网络的组成部分。

用户可以将模块配置为 CAN 状态或 CANFD 状态，使其与被调试的 TTCANopen 网络状态相匹配，当用户只需对原网络进行监听时，可将模块配置为静听态，并能同时静听原网络中的 CAN 和 CANFD 指令。

推荐用户使用本说明书第 8 章所介绍的指令方式设置模块参数（详见第 8 章）。

为了方便用户观察原网络中指令的运行情况，TTCANopen 调试模块虚拟串口输出为字符串格式，每帧指令独占一行，用户可使用各种串口调试工具显示调试，也可以使用 MY_CANFD_AUTO 上位机软件调试。

与普通模块不同，TTCANopen 模块将 29 位 ID 字拆分为四个具有实际物理含义的子段，分别为：优先级段、寄存器基地址段、设备编号段和功能码段，使其能够贴近实际应用构成一套完整的应用层指令系统和管理系统。

TTCANopen 仅使用了 29 位的扩展数据帧。　其格式如下：

1）显示 TTCANopen 扩展数据帧指令格式：

帧计数	接收时标	帧标识	优先级	寄存器基地址	设备地址	指令	DLC	帧标识	数据场 DLC 个字节
00000000 至 FFFFFFFF	时:分:秒:毫秒.微妙	[	0 或 1	0000 至 FBFF	00 至 7F	00 至 1F	00 至 08	]	xx ~ xx

注 1：上述格式是按普通 CAN 状态表述的，CANFD 指令中 DLC 字段和其所对应的数据场长度，遵守 CANFD 协议相关规定。DLC 字段用 10 进制显示。

TTCANopen 调试模块显示帧举例：

1）TTCANopen 扩展数据帧：

00000008 00:09:29:546.220 [1 7100 32 19 2] 11 22

上例中:

第一个字符串“00000012”：

为接收帧计数，计数范围 0～0xFFFFFFFF；

第二个字符串“00:09:29:546.220”：

为接收时标，分辨率为 1 微秒；

第三个[扩起来]的字符串“[1 7100 32 19 2]”：

分别为 1 位优先级、16 位寄存器基地址、7 位设备地址、5 位指

令功能码和 DLC；（DLC 为十进制数，CAN 状态其为 0 至 8，CANFD 状态为 0 至 8、12、16、24、32、48、64，遵守 CANFD 协议相关规定）。

第四个字符串“11 22”：

为数据帧中 DLC 个字节数据内容。

图 15 为在 MY_CANFD_AUTO 上位机接收窗口观察到的接收帧举例。



图 15 TTCANopen 调试模块接收显示举例

TTCANopen 仅使用了 29 位的扩展数据帧的发送，并对发送帧进行“回显”，发送指令使用十六进制方式，其格式如下：

1）发送 TTCANopen 扩展数据帧指令格式：

帧头	帧标识	优先级	寄存器基地址	设备地址	指令	DLC	数据场 DLC 个字节	帧尾	DLC
AA	CC	00 或 01	0000 至 FBFF	00 至 7F	00 至 1F	00 至 08	xx ~ xx	DD	00 至 08

注 1：寄存器基地址为 16 位十六进制数，高地址在前，低地址在后。

注 2：CANFD 指令中 DLC 字段和其所对应的数据场长度，遵守 CANFD 协议相关规定。其有效值为：0x00～0x08、0x0C、0x10、0x18、0x20、0x30、0x40。

TTCANopen 发送帧指令举例：

1）扩展数据帧：

AA CC 01 7100 32 19 08 11 22 33 44 55 66 77 88 DD 08

TTCANopen 发送帧解析：AA（帧头）、CC（扩展帧）、01（优先级）、7100（寄存器基地址）、32（设备地址）、19（功能码）、08（DLC）、11 22 33 44 55 66 77 88（数据场数据）、DD（帧尾）08（DLC）。

图 16 为 MY_CANFD_AUTO 上位机中 TTCANopen 调试模块发送指令和回显示例。（注意与接收帧的区别：回显帧是没有帧计数的）

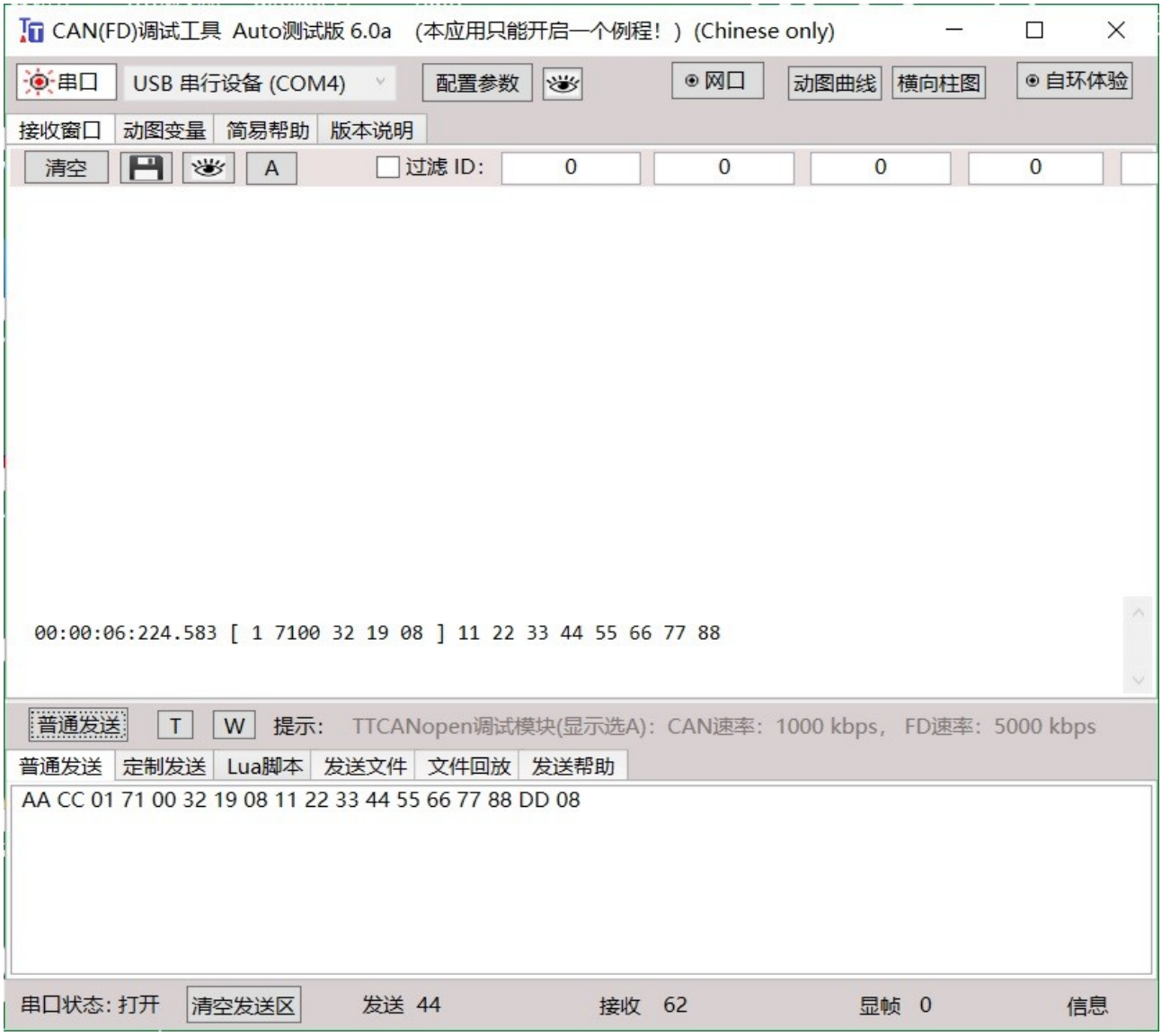


图 16 TTCANopen 调试模块发送指令和回显示例

10. TTCANopen 工作模块

TTCANopen 工作模块，是已有 TTCANopen 网络原生的组成部分，其主要承担 TTCANopen 网络与系统上位机的通讯连接和协议转换，是中心监控系统的交通枢纽。利用 TTCANopen 协议“组态王”驱动，用户可快速建立起自己的中心监控系统。

用户可以将模块配置为 CAN 状态或 CANFD 状态，使其与固有 TTCANopen 网络状态保持一致。

用户可以使用本说明书第 8 章所介绍的指令方式设置模块参数（详见第 8 章）。

为了提高指令效率，TTCANopen 工作模块虚拟串口输入输出皆为 16 进制格式，帧与帧之间使用帧头帧尾标识。

TTCANopen 工作模块仅支持 29 位的扩展数据帧输入输出，其中，发令格式与 TTCANopen 调试模块相同，但 TTCANopen 工作模块不“回显”发送指令。

TTCANopen 工作模块指令接收格式如下：

1）接收 TTCANopen 扩展数据帧指令格式：

帧头	接收时标	帧标识	优先级	寄存器基地址	设备地址	指令	DLC	数据场 DLC 个字节	帧尾	DLC
BB	tt tt tt tt	CC	00 或 01	0000 至 FBFF	00 至 7F	00 至 1F	00 至 08	xx ~ xx	EE	00 至 08

注 1：tt tt tt tt 为 32 位接收时间计数（0.1ms），其低字节在前，高字节在后；而寄存器基地址是高字节在前，低字节在后，解帧时需特别留意。

注 2：CANFD 指令中 DLC 字段和其所对应的数据场长度，遵守 CAN_FD 协议相关规定。其有效值为：0x00～0x08、0x0C、0x10、0x18、0x20、0x30、0x40。

TTCANopen 扩展数据帧接收指令举例：

1）TTCANopen 扩展数据帧：

BB 15 23 00 00 CC 01 71 00 32 19 03 11 22 33 EE 03

TTCANopen 接收帧解析：BB（帧头）、15 23 00 00（时标）、CC

（扩展帧）、01（优先级）、71 00（寄存器基地址）、32（设备地址）、19（功能码）、03（DLC）、11 22 33 （数据场数据）、EE（帧尾）、03（DLC）。

图 17 为 TTCANopen 工作模块，指令接收示例。

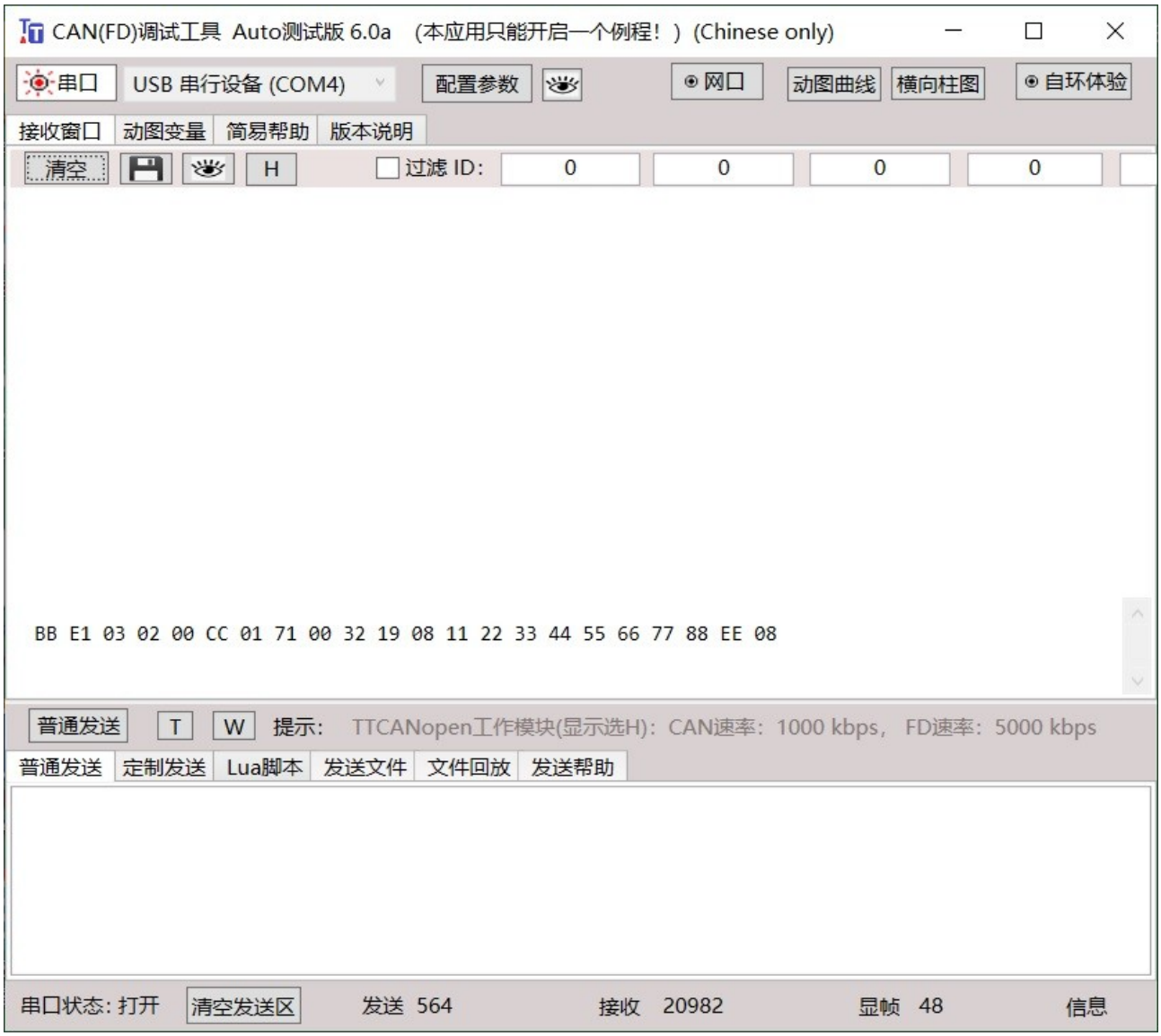


图 17 TTCANopen 工作模块接收指令演示

11. TTCANopen 过滤器

在本说明书第 8 章中介绍了模块 ID 过滤寄存器空间的分配，在 TTCANopen 模块的应用中，这 6 组 ID 过滤器的使用规则发生了变化，其更为灵活易用。

TTCANopen 协议，将 29 位 ID 字，合理的划分为 5 个有实际物理含义的子段，即：优先级段（1 位）、寄存器基地址段（16 位）、设备地址段（7 位）和功能码段（5 位）。（ TTCANopen 协议 ID 字的详细划分内容请参考网站内教程或本说明书的附录 1 。 ）

TTCANopen 协议的过滤字（32 位四个字节）只针对寄存器基地址段、设备地址段和功能码段，不包含优先级段。

TTCANopen 协议的过滤字（从高到低）前两个字节指向寄存器基地址段，第 3 个字节指向设备地址段，第 4 个字节指向功能码段。

如：我们配置过滤字为 0x71003209，这里“0x7100”是寄存器基地址，“0x32”是设备地址，“0x09”是功能码（读）。这个就能够显示针对 0x32 设备其寄存器地址为 0x7100 的读指令（0x09），比如：BB 23 56 00 00 CC 01 71 00 32 09 03 11 22 33 EE 03。

如果我们将对应字段用“0xFFFF”或“0xFF”取代，就表明不对该字段进行筛选过滤操作，如：0x710032FF，这表明我们需要显示总线上对设备 0x32 上，其寄存器基地址为 0x7100 的所有操作，而不仅仅是上面“0x09”这个“读”指令。又如：“0xFFFF32FF”就是要关注对

设备 0x32 上的所有操作，而不仅仅是关于其“0x7100”寄存器的操作。

由于有了 0xFF 屏蔽字的作用，使得 TTCANopen 协议的过滤字在实际应用中更加灵活方便。

（注意：需要提醒的是在配置过滤字指令中，过滤字低字节在前高字节在后，如上指令为：AA 66 FF 10 04 09 32 00 71 DD）

图 18 是在 MY_CANFD_AUTO 上位机配置 TTCANopen 过滤器。



图 18 在 MY_CANFD_AUTO 上配置过滤器举例

12. TTCANopen 心跳器模块

心跳器模块，是为 TTCANopen 网络提供系统心跳的设备，网络上的设备收到系统心跳会将其本地时间与心跳时间同步，完成总线时间统一。

用户可以使用本说明书第 8 章所介绍的指令方式设置模块参数（详见第 8 章）。

心跳器，完成一次心跳分为两帧发送，第一帧为时间预报帧，第二帧为心跳同步帧，两帧通常间隔 4～5 帧时间。

（注意：CANFD 状态下的心跳，使用数据域不加速的指令格式）

预报帧：　00 0A 02 01 19 08 tt tt tt tt re te TT TT

心跳帧：　00 0A 03 01 19 00

其中：tt tt tt tt 为“预报时间”是紧跟着的“心跳同步帧”的结束时刻，以 0.1ms 为记时单位，0～0x337F97FF 是一天中 0.1ms 的计数个数（最大 863999999），即：所谓 TTCANopen 时间。

re 和 te 是“CAN 接收错误计数”和“CAN 发送错误计数”是 CAN 端口收发错误寄存器中的计数值，是心跳器 CAN 端口的通讯质量的表征。

TT TT 是“心跳周期”以 ms 为计数单位。默认为 1（E8 03）秒。

心跳同步帧为固定帧，帧发送结束时刻为时标，网络上的设备，接收此帧后立即将其本地时间值同步到预报时间上，完成与系统时间的同步。

在同一网络中只能有一个心跳器工作，其它心跳器处于备机状态，并将本地时间自动同步到主心跳器上，当检查到主心跳器十次没有发出正常心跳时，备机会自动启动替换主心跳器。

用户可以通过 USB 仿真串口的发送端，控制心跳器的启动和停止，也可以设置心跳器的起始时间和心跳周期。

设置串口调试工具的发送窗口为字符方式，然后输入？问号并发送，这时在接收窗口会出现帮助信息如图 19 所示。

用户可以根据该提示在发送窗口进行所需操作，出错时，会有相关提示。

如果用户需要将心跳周期改为 2 秒，就可以在发送窗口输入：L02000#，然后点“发送键”；

如果用户需要将起始时间改为 12:12:12:000.0， 可输入：T12:12:12:000.0#，然后发送。

如果用户需求启动或停止心跳，可输入：Q， 然后发送。

用户也可以发送：N 来结束心跳器工作，恢复设备到调试模块。图 20 为 TTCANopen 调试模块接收心跳同步信息。而 TTCANopen 工作模块接收心跳同步信息是以十六进制显示的。

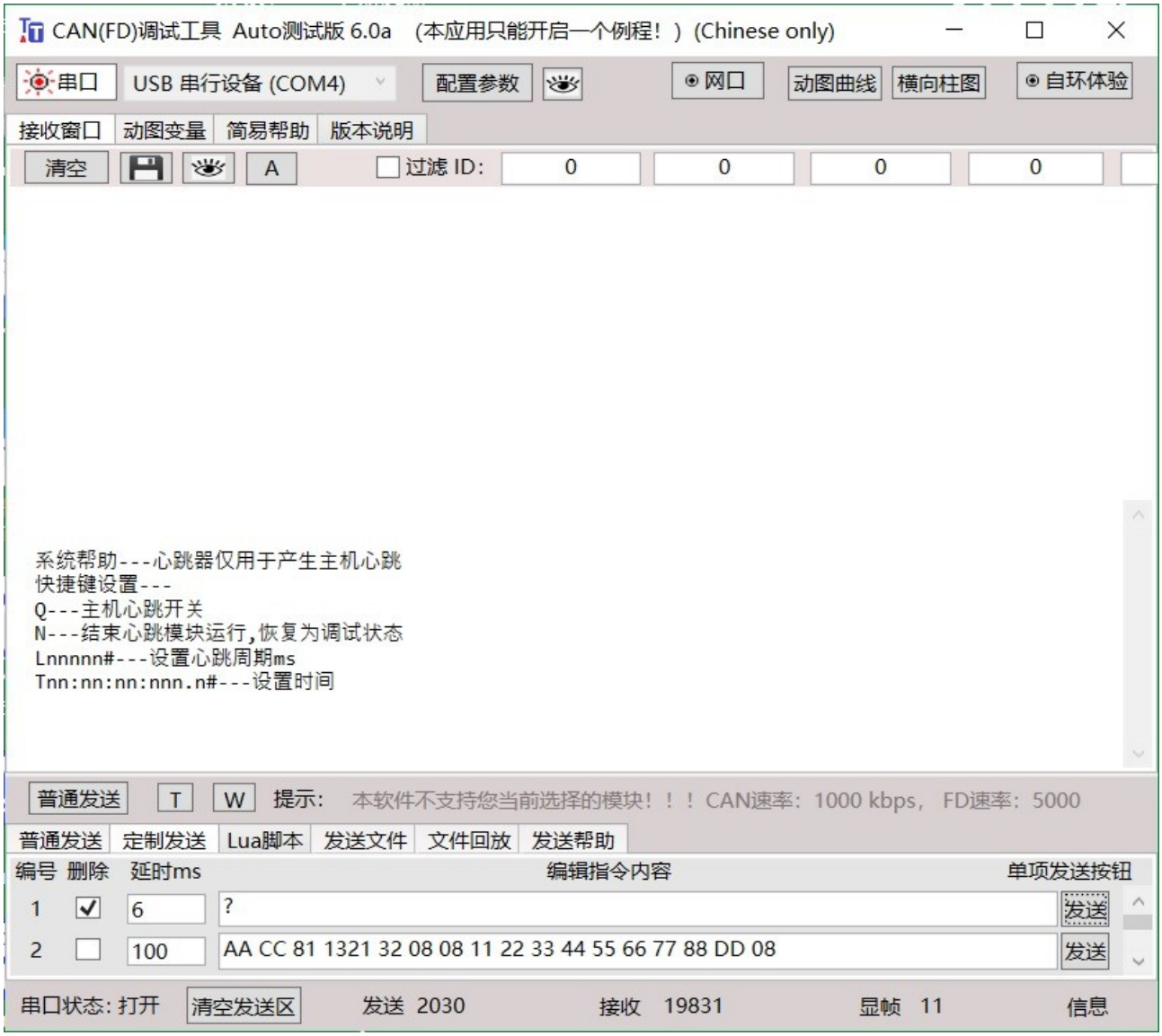

CAN(FD)调试工具 Auto测试版 6.0a (本应用只能开启一个例程！) (Chinese only)
串口
USB 串行设备 (COM4)
配置参数
网口
动图曲线
横向柱图
自环体验
接收窗口
动图变量
简易帮助
版本说明
清空
A
过滤 ID:
0
0
0
0
系统帮助---心跳器仅用于产生主机心跳
快捷键设置---
Q---主机心跳开关
N---结束心跳模块运行,恢复为调试状态
Lnnnnn#---设置心跳周期ms
Tnn:nn:nn:nnn.n#---设置时间
普通发送
T
W
提示:
本软件不支持您当前选择的模块！！！CAN速率：1000 kbps， FD速率：5000
普通发送
定制发送
Lua脚本
发送文件
文件回放
发送帮助
编号
删除
延时ms
编辑指令内容
单项发送按钮
1
6
?
发送
2
100
AA CC 81 1321 32 08 08 11 22 33 44 55 66 77 88 DD 08
发送
串口状态: 打开
清空发送区
发送 2030
接收 19831
显帧 11
信息

图 19 心跳器帮助信息

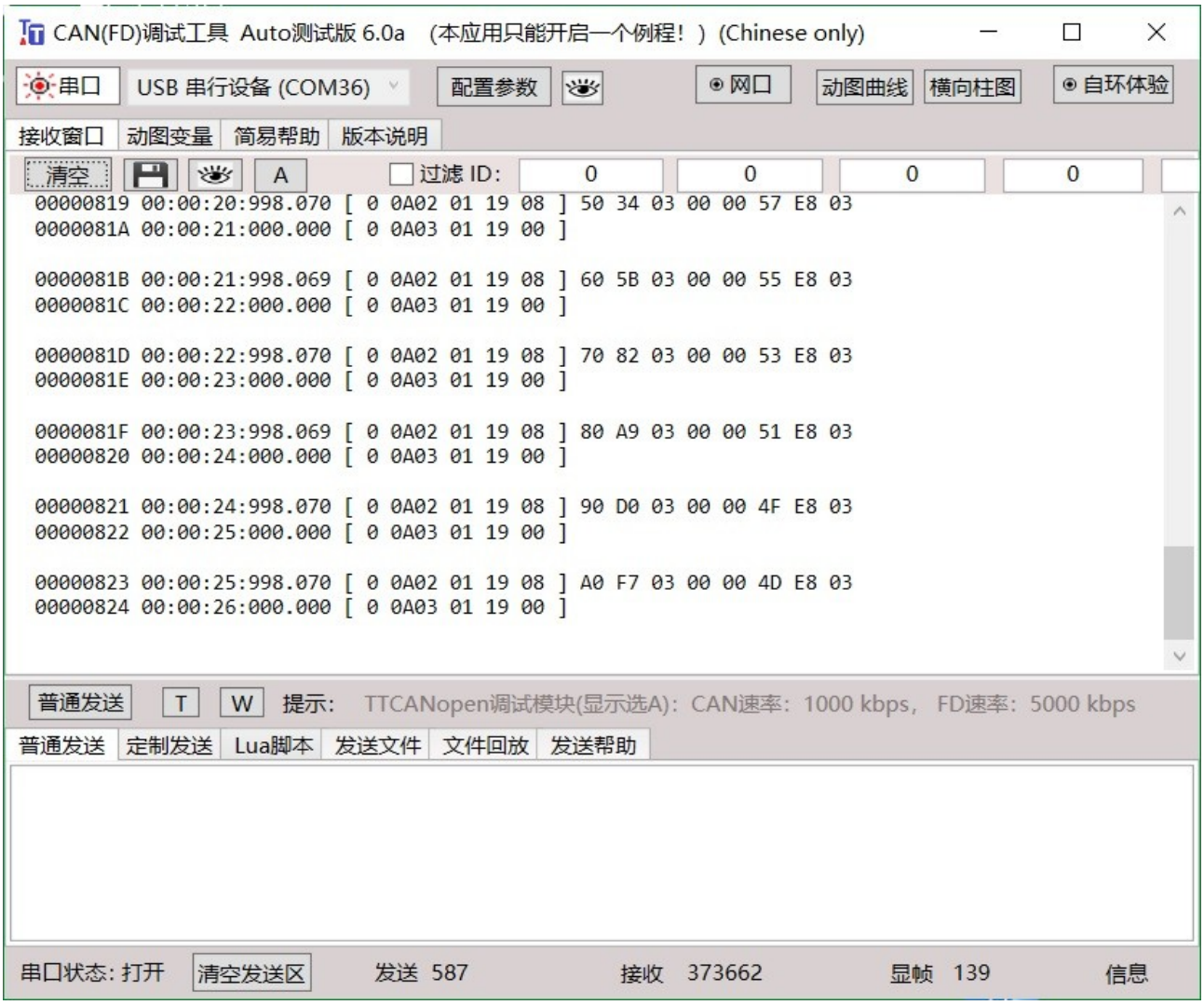


图 20 TTCANopen 调试模块接收心跳同步信息

13. CANFD 数据域不加速发送

在消费类模块 v3.7 版本时，默认 CANFD 发送数据域是加速的。但在实际应用中，有少数用户在使用数据域不加速的方式发送 CANFD 帧，为了使模块功能更加完备，满足更多用户的应用需求，在消费类模块 v4.0 版以后添加了该功能。轻奢车规级模块 VA6.0 继承了该功能。

发送方法：在发送指令序列中，将表述 CANFD ID 字的字节序列的最高位置 1.

举例：常规发送指令如：（默认数据域加速）

AA 55 01 23 08 11 22 33 44 55 66 77 88 DD 08

指令中，01 23 是 ID 字，将 ID 字最高位置 1 后指令为：

AA 55 81 23 08 11 22 33 44 55 66 77 88 DD 08

再例：

AA CC 12 34 56 78 04 11 22 33 44 DD 04

指令中，12 34 56 78 是 ID 字，将 ID 字最高位置 1 后指令为：

AA CC 92 34 56 78 04 11 22 33 44 DD 04

实际发送的 CANFD ID 字，标准帧是 11 位，扩展帧是 29 位的，都没有用到 16 位或 32 位字节 ID 字的最高位，这里最高位置 1，只是用于标识发送不加速，不会影响真实发送的 ID 字。

另外，鉴于不加速帧传输的可靠性更好，模块作为 TTCAN

open 心跳器时，发送心跳帧，使用 CANFD 数据域不加速方式发送。

预报帧： AA CC 80 0A 02 01 19 08 tt tt tt tt re te TT TT DD 08

心跳帧： AA CC 80 0A 03 01 19 00 DD 00

14. 模块 TTCANopen 特性（只针对 TTCANopen 状态）

1）模块遵从“TTCANopen 应用层协议 CAN 标识的分配方法”，见附录 1；

2）模块遵从“TTCANopen 应用层协议设备内部多条指令的发送规则”，见附录 2；

3）模块接受系统心跳，并与之同步，图 21 为 TTCANopen 调试模块的时间同步过程。

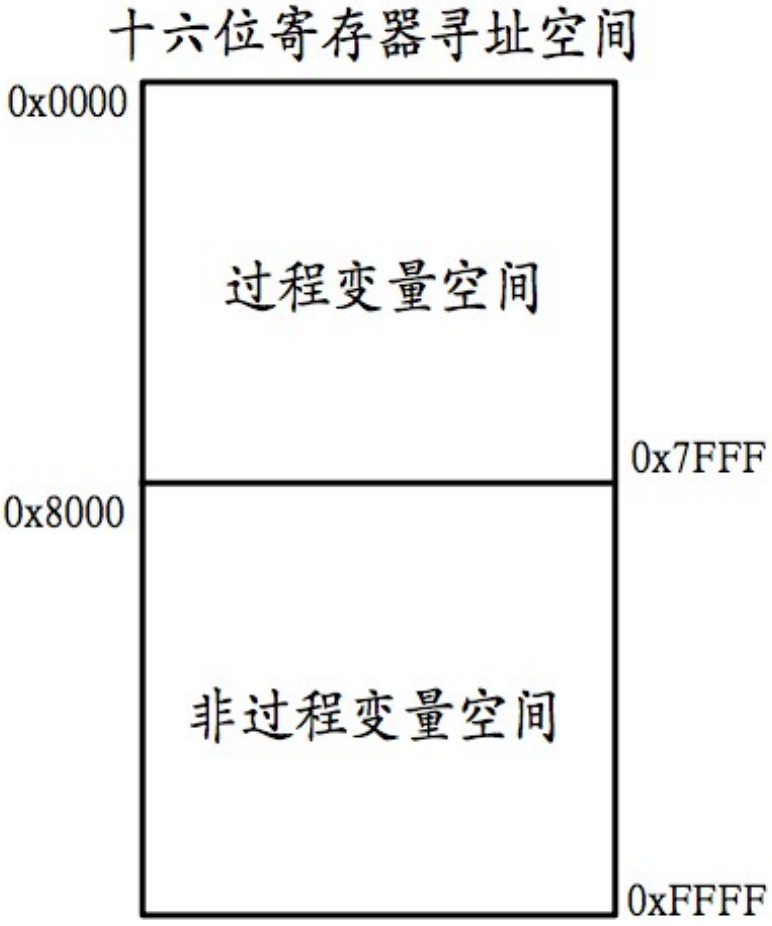


图 21 TTCANopen 调试模块与系统时间同步

附录 1:　TTCANopen 应用层协议

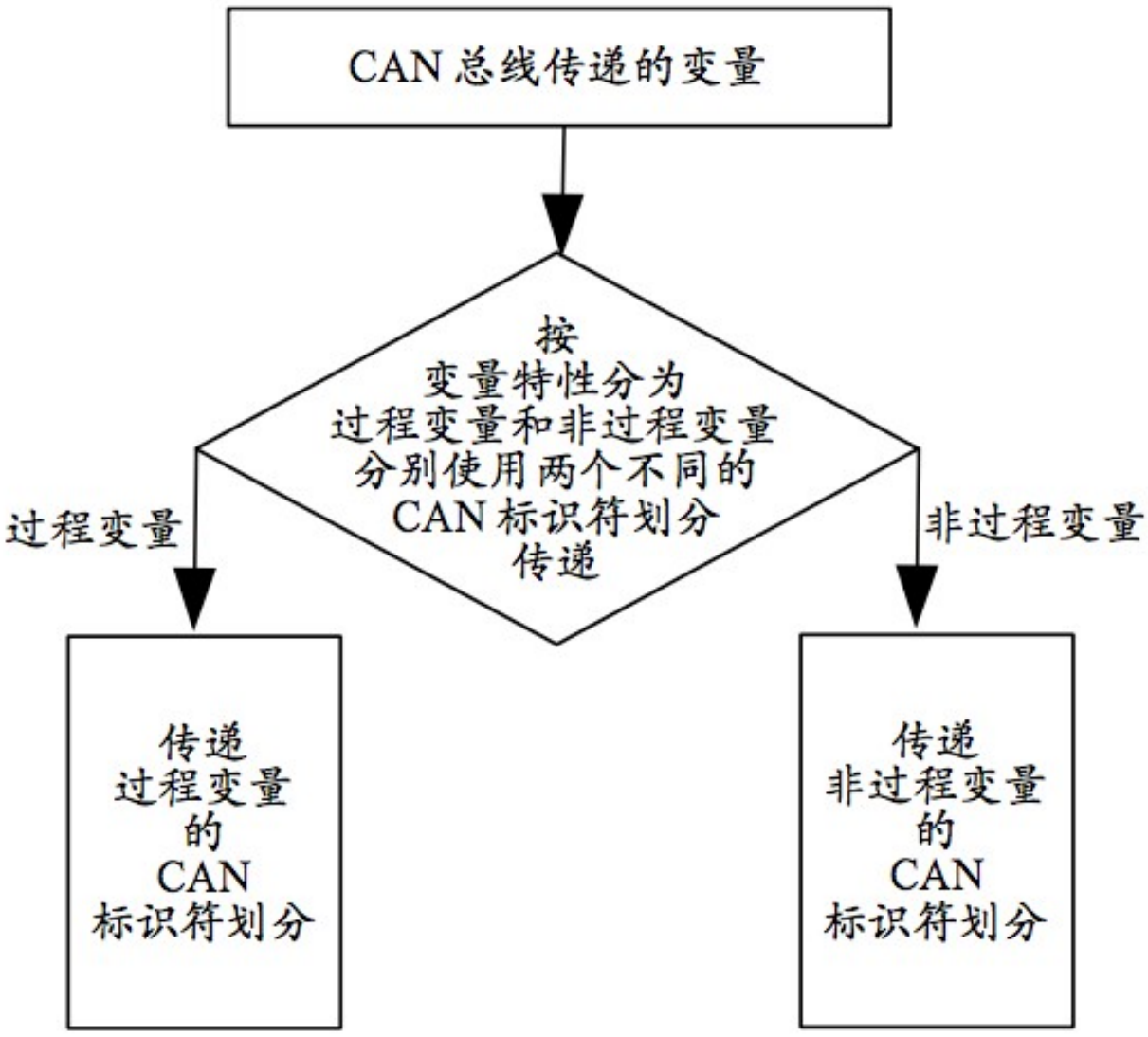
CAN 标识的分配方法

TTCANopen 应用层协议以连续寄存器空间为信息载体，并将其十六位寄存器寻址空间平分为两部分，见附图 1 所示。 其中: 0x0000～0x7FFF 用于存放、传递过程变量；0x8000～0xFFFF 用于存放、传递非过程变量。



附图 1 寄存器空间的划分

TTCANopen 应用层协议根据 CAN 总线传递变量特性的不同，即过程变量或非过程变量，使用两个不同的 CAN 标识符划分分别传递上述两种变量，见附图 2 所示，从而为两种变量提供了不同的总线竞争途径，并为应用层协议实施生产者 -消费者模型和主机-从机模型提供了明确的对象指向。



附图 2 使用不同的 CAN 标识符划分传递两种不同的变量

传递过程变量的 CAN 标识符划分，采用了在 CAN 标识符中抽取多个位片段，并重新排列拼接组合的方式，构成应用层传递过程变量的寄存器基地址段，从而可以在过程变量寄存器基地址空间构成多个等价竞争层，很好的解决了变量分层管理与总线竞争分配不平衡的矛盾。

传递过程变量的 CAN 标识符划分，见附图 3 所示，其将 CAN2.0b 或 CAN_FD 协议扩展帧格式中 29 位 CAN 标识符划分为六段，即:优先级段、子段 1、子段 3、设备编号段、功能码段和子段 2，其中:子段 1、子段 2 和子段 3 排列拼接组合构成寄存器基地址段，各段组成说明如下:

优先级段:　由 CAN 标识符的最高位第 28 位组成;

子段 1:　　由 CAN 标识符的第 27 位组成;

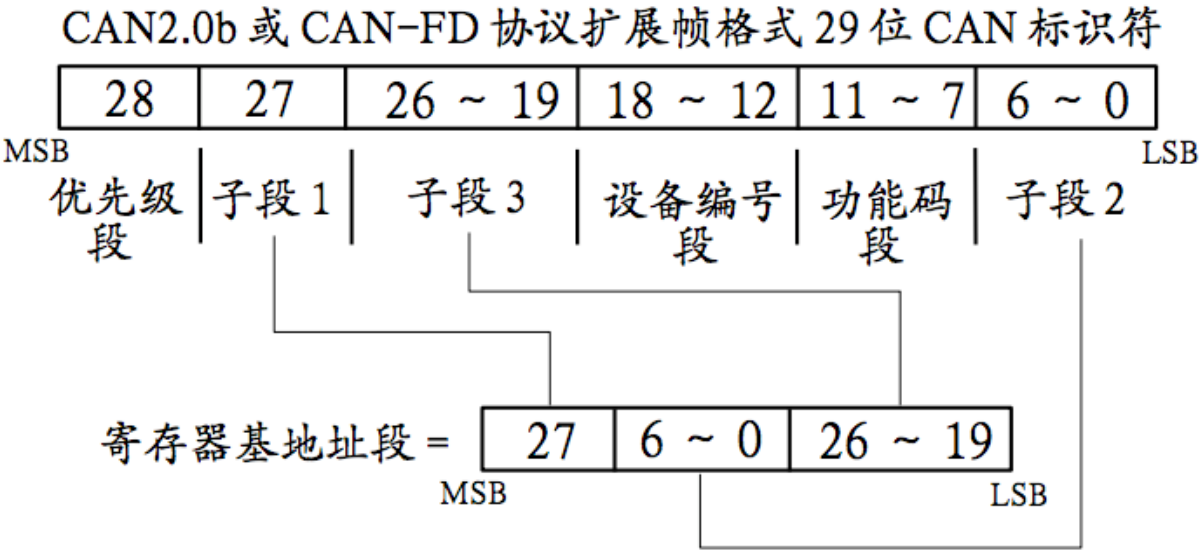
子段 3:　　由 CAN 标识符的第 26～19 位组成;

设备编号段: 由 CAN 标识符的第 18～12 位组成;

功能码段:　由 CAN 标识符的第 11～7 位组成;

子段 2:　　由 CAN 标识符的第 6～0 位组成;

寄存器基地址段:由子段 1、子段 2 和子段 3 排列拼接组合而成，即:由 CAN 标识符的第 27 位、6～0 位和 26～19 位组成十六位的寄存器基地址段。



附图 3 传递过程变量的 CAN 标识符划分

当 CAN 标识符的第 27 位(即:寄存器基地址段的最高位)为 0 时，寄存器基地址空间指向过程变量空间(即:0x0000～0x7 FFF)，由于十六位寄存器基地址段的低八位(26～19 位)处于 CAN 标识符的较高端 MSB，而其次高七位(6～0 位)处于 CAN 标识符的低端 LSB，由此过程变量寄存器空间变量的竞争能力由其十六位寄存器基地址段的低八位决定，从而在过程变量寄存器地址空间形成了 128 个等价竞争层，每层包含 256 个地址空间，在每层相同位置上的过程变量具有相同的总线竞争能力，在同层中，低地址变量的竞争能力优于高地址变量。

传递非过程变量的 CAN 标识符划分，见附图 4 所示，其将 CAN2.0b 或 CAN_FD 协议扩展帧格式中 29 位 CAN 标识符划分为四段，即:优先级段、寄存器基地址段、设备编号段和功能码段，各段

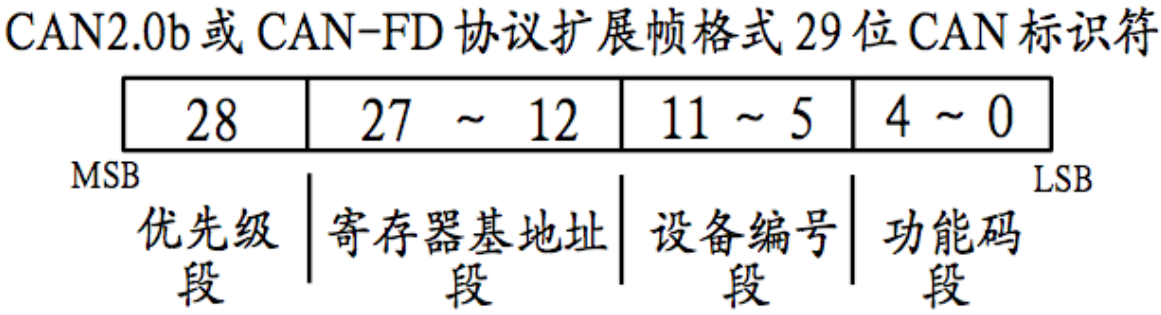
组成说明如下:

优先级段:　由 CAN 标识符的最高位第 28 位组成;

寄存器基地址段:由 CAN 标识符第 27～12 位组成;

设备编号段:　由 CAN 标识符的第 11～5 位组成;

功能码段:　由 CAN 标识符的第 4～0 位组成;



附图 4 传递非过程变量的 CAN 标识符划分

一般非过程变量的实时性要求不高，在应用层通常适用主机-从机模型，主机和从机间采用问答方式通讯，变量分段管理与其总线竞争能力没有突出的矛盾，因此从 CAN 标识符的高端直接截取十六位(27～12 位)构成寄存器基地址段，当 CAN 标识符的第 27 位(即: 寄存器基地址段的最高位)为 1 时，寄存器基地址空间指向非过程变量空间(即:0x8000～0xFFFF)，非过程变量的总线竞争能力整体低于过程变量，这是应用层所期望的，由于 CAN 标识符的高七位不能连续出现高电平 1，当优先级位(28 位)为 1 时，非过程变量实际有效的地

址空间为 0x8000～0xFBFF，也就是在 0xF000 段的尾部集中有 1024 个地址空间(0xFC00～0xFFFF)不能使用，如果对非过程变量采用和过程变量一样的寄存器拼接组合的方法构成寄存器基地址段，势必将这 1024 个不可用地址空间分散到各等价层中，这对非过程变量寄存器空间的使用和管理无益，因此应用层对过程变量和非过程变量分别采用不同的 CAN 标识符划分进行总线传递是非常必要的。

TTCANopen 应用层协议对 CAN 标识符的划分及其物理含义的分配，如:优先级段、寄存器基地址段、设备编号段、功能码段都具有明确的应用层协议指令含义，据此设计的应用层协议具有很好的直读性和开放性，使应用层协议易于解析，便于用户在框架内构建适合其使用环境的应用层子协议。

TTCANopen 应用层协议以连续寄存器空间为信息载体，其应用层协议指令是对该连续寄存器空间的读写访问，由此使其能够天然的兼容 CAN-FD 协议对数据场长度的扩充。

附录 2:　TTCANopen 应用层协议

设备内部多条指令的发送规则

通常 CAN 通讯协议只规范了多个设备在总线上同时发送指令时的竞争顺序，而对于单个设备内部产生的多条指令的发送顺序却很少涉及。

在纯主从模式下，设备内部产生多条指令的可能性并不大，相关问题很难暴露，而当系统发展演绎到全触发方式后情况就不同了，一个设备可能具有多个触发变量，它们可以是周期触发、变化触发、越界触发等等，而总线的发送速率是有限的，且同时还存在其他设备对总线的竞争，因此在同一设备中就有可能积累多条指令等待发送。

通常设备会默认以“先生优势”规则处理这些指令，即先产生的指令优先发送，这样处理的最大好处是指令动作先后逻辑规范，这对那些具有先后关联的系列指令尤为重要，但事物总是两方面的，这种“先生优势”规则，会延迟和阻碍后产生的高优先级指令的发送，如：告警指令，尤其是当总线上存在竞争时，情况尤为严重，如:当设备指令发送端口被一条先生成的低优先级指令占据时，其在总线上的竞争处于劣势，从而使该设备中后产生的告警指令无法参与竞争而被推迟发送，这种延迟有时可能是致命的。 因此我们需要重新规范设备内部多条指令的发送规则，最粗暴简单的方法就是对这多条指令进行

优先级排队，将 ID 字小的指令排在前面，每当设备产生一条新指令时都要重新进行一次指令排序(注意:这种重新排序应当包括那条已经处在发送端口的未发指令)，这样设备便具有了最优的发送竞争力，但同时可能会破坏前面所说的某些指令的先后逻辑关联，可能使系统运转出现偏差。

为此 TTCANopen 应用层协议做出如下规范，当一设备同时具有多条指令等待发送时：

1)优先级位为 0 的指令优先于优先级位为 1 的指令发送;

2)优先级位相同的指令按其产生的先后顺序发送。

虽然只有短短的两条规定，但它却完成了其它应用层协议很难规范完成的内容，而这对于 TTCANopen 协议来说却是自然天成的，它的第一条规定确保了低优先级(1)的指令不能阻碍高优先级(0)指令的发送，使设备中诸如告警指令得以顺利抢先通过总线发送；第二条规定保障了具有先后逻辑关系的系列指令的顺利完成，因为我们一般不会将这些相关指令定义在不同的优先级位上。

免责声明

本着对用户负责的精神，说明书尽可能翔实正确的对产品功能和指标进行描述，但不能保证没有遗漏和错误。另外，随着产品不断升级和完善，说明书内容也具有一定的时效性和适用性，可能会在没有特别通知的情况下，进行修改和补充。

请用户及时关注 www.ttcanopen.com.cn 网站中相关信息。感谢您的包容与支持。

风险提示

1）GD32A103TBU7 的工作温度范围是-40～105 摄氏度，因此该芯片及其衍生品，不能用于发动机舱及重大安全和可靠性应用场景。

2）虽然本轻奢车规级模组和模块，较消费类产品可靠性有重大的提升，但其并没有提交相关部门通过车规级认证，因此本品仍旧仅限于提供给初学者和爱好者学习交流体验。

3）凡将本品用于车载及工业现场和类似场景的用户，需当自行完成系统的冗余设计，可靠性论证，风险评估。本品厂家不对用户行为后果进行担责。

4）售出产品固件升级和定制固件修改，需当将模组寄回厂家完成，厂家不提供现场服务。

5）由于本品极易被复制和衍生，厂家只对原厂出售的正版产品

提供售后服务。

更多内容请关注：www.ttcanopen.com.cn