UARTCAN Command Line Reference

UARTCAN内置一个命令解释器，可以通过超级终端（或者Putty、SecrueCRT）等软件来连接。

默认串口参数：`波特率115200、8位数据、1位停止、无校验、无流控`。

本文档基于UARTCAN固件v20.8.1，其余固件版本仅供参考。

can

查看、设置CAN接口参数。

命令格式：`can [baud|mode|speed|clear] CAN port setup.`

不带参数的can命令显示当前CAN接口参数和状态。包括命令帮助、波特率、CAN模式、收发帧计数。

```
can
can [baud|mode|speed|clear] CAN port setup.
 BAUD=1000k MODE=0 Normal
 RX OK=0 LOST=0
 TX OK=0 LOST=0
```

can baud

查看、设置CAN接口波特率。波特率单位为kbps，支持波特率范围：1000kbps-40kbps。

```
can baud
 can baud [baud in kbps] set CAN baudrate.
 BAUD=1000k
can baud 500
 BAUD=500k
```

CAN接口目前支持的波特率为：`1000、900、800、666、600、500、400、300、250、200、150、125、100、90、80、60、50、40(kbps)`

波特率设置完成立即生效，保存参数需要使用`param save`命令。

can mode

设置CAN接口工作模式。

```
can mode
 can mode [0|1|2|3] set CAN work mode.
 MODE=0 Normal
can mode 0
 MODE=0 Normal
can mode 1
 MODE=1 LoopBack
can mode 2
 MODE=2 Silent
can mode 3
 MODE=3 Silent_LoopBack
```

命令解释如下：

- `can mode`命令查看当前CAN接口工作模式，
- `can mode 0`设置CAN为Normal模式。
- `can mode 1`设置CAN为LoopBack模式。
- `can mode 2`设置CAN为Silent模式。

- `can mode 3`设置CAN为Silent_LoopBack模式。

四种工作模式的意义见下表，更详细解释请参考STM32用户手册的bxCAN部分。

工作模式	取值	备注
Normal	0	正常模式（常用）
Loopback	1	环回模式
Silent	2	静默模式
Silent_Loopback	3	环回静默模式

can speed

查看过去64秒每秒内接收到的CAN帧数，即接收帧速率了。单位`帧/秒`或者`fps`。

```
can speed
Oldest
     0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0     46
  2892   2853   2910   1299      0      0      0      0
                                                   Newest
CAN RX speed at last 64 seconds. MIN=0fps MAX=2910fps
```

can clear

清理CAN收发计数器。

```
can clear
 Clear RX/TX Counters.
 RX OK=0 LOST=0
 TX OK=0 LOST=0
```

std

发送标准帧。

命令格式：`std [hexID] [hexDATA|remote] Send standard data/remote message.`

命令举例如下：

```
std 44 12345678
 TX:00000000 DATA MSG ID=0x0044 DATA=12 34 56 78
std 44 remote
 TX:00000001 REMOTE MSG ID=0x0044
std 33 0011223344556677
 TX:00000002 DATA MSG ID=0x0033 DATA=00 11 22 33 44 55 66 77
```

可以带两个参数：

- 第一个参数为帧 ID，16位，hex表示。
- 第二个参数如果为`remote`表示远程帧，如果为hex字符串，表示实际数据，最大8个字节。
- 环境变量`SHOW`控制是否显示发送数据。
- 环境变量`CNT`控制发送数量。
- 环境变量`PRD`控制发送周期。

- 环境变量`ADD`控制重复发送时帧ID是否自增。

ext

发送扩展帧。

命令格式：`ext [hexID] [hexDATA|remote] Send extended data/remote message.`

命令举例如下：

```
ext 44 12345678
 TX:00000003 DATA MSG ID=0x00000044 DATA=12 34 56 78
ext 44 remote
 TX:00000004 REMOTE MSG ID=0x00000044
ext 33 0011223344556677
 TX:00000005 DATA MSG ID=0x00000033 DATA=00 11 22 33 44 55 66 77
```

可以带两个参数：

- 第一个参数为帧 ID，32位，hex表示。
- 第二个参数如果为`remote`表示远程帧，如果为hex字符串，表示实际数据，最大8个字节。
- 环境变量`SHOW`控制是否显示发送数据。
- 环境变量`CNT`控制发送数量。
- 环境变量`PRD`控制发送周期。
- 环境变量`ADD`控制重复发送时帧ID是否自增。

env

查看、设置环境变量：

命令格式：`env [show|add|cnt|prd] Set/Show environment variables.`

不带参数的`env`命令显示当前所有环境变量：

```
env
env [show|add|cnt|prd] Set/Show environment variables.
 SHOW=1 ADD=1 SINGLE=0 CNT=1 PRD=0us
```

带一个参数的`env`命令查看需要的环境变量：

```
env show
 SHOW=1
env cnt
 CNT=1
env prd
 PRD=0us
```

带两个参数的`env`命令设置环境变量：

```
env show 1
 SHOW=1
env cnt 100
 CNT=100
env prd 200
 PRD=200us
```

目前`UARTCAN`支持的环境变量如下表所示：

变量名	意义	备注
SHOW	是否显示收发数据	0:不显示 1:显示 按ctrl+p切换
ADD	发送时帧ID是否自增	0:不自增 1:自增
SINGLE	是否自动重传	0:不重传，1:重传
CNT	发送数据帧数量	大于0的整数
PRD	发送周期	单位us

注意：

- 连续接收数据时可以按下ctrl+p来切换是否显示数据，即改变SHOW环境变量。
- 设置SHOW=0关闭显示数据可以达到最大的CAN收发速度。
- 设置SINGLE=1可以防止CANH和CANL悬空时发送报错。
- 发送周期设置为0以最大速度发送。

按下`ctrl+p`组合键，可暂停CAN帧接收显示，再次按下`ctrl+p`继续显示接收到的数据包。 暂停显示以后，设备在后台继续接收CAN帧，可以使用can命令查看接收到的数据帧数量。

uart

查看、设置UART异步串口参数。

命令格式：`uart [baud|mode|addr] UART port setup.`

不带参数的`uart`命令显示当前UART接口状态，执行结果如下：

```
uart
uart [baud|mode|addr] UART port setup.
 BAUD=115200
 MODE=0 TERM
 ADDR=1
```

uart baud

查看、设置UART接口波特率。

- `uart baud`命令查看当前UART接口波特率
- `uart baud 57600`将UART接口波特率设置为57600bps

```
uart baud
 UART BAUD = 115200
uart baud 57600
 set UART BAUD to 57600, save & reboot to apply new baud.
```

修改UART接口波特率以后，需要执行`param save`命令保存参数，重启生效。

uart mode

查看UART接口工作模式。

```
uart mode
 UART MODE = 0 TERM
```

短按按键查看UART工作模式，长按按键切换UART接口工作模式。

uart addr

设置UART接口地址。

- 使用`uart addr`命令查看当前UART接口地址
- 使用`uart addr 2`将UART接口地址设置为2

```
uart addr
 UART ADDR = 1
uart addr 2
 set UART ADDR to 2, save & reboot to apply new addr.
```

filter

显示、设置CAN接口滤波参数。**该命令使用复杂，如无特殊需要，建议使用默认设置。**

命令格式：`filter [0-7] [mode|scale|act|r0|r1] [param] CAN filter setup.`

UARTCAN支持最多8个滤波器设置，编号0-7，不带参数的filter命令显示当前CAN接口滤波器设置:

- mode取值为`idlist`或者`idmask`；
- scale取值32或者16；
- act取值1或者0；
- R0和R1为两个寄存器，具体意义由mode和scale取值决定。

```
filter
filter [0-7] [mode|scale|act|r0|r1] [param] CAN filter setup.
 filter 0 MODE=idmask SCALE=32 ACT=1 R0=0x00000000 R1=0x00000000
 filter 1 MODE=idmask SCALE=32 ACT=0 R0=0x00000000 R1=0x00000000
 filter 2 MODE=idmask SCALE=32 ACT=0 R0=0x00000000 R1=0x00000000
 filter 3 MODE=idmask SCALE=32 ACT=0 R0=0x00000000 R1=0x00000000
 filter 4 MODE=idmask SCALE=32 ACT=0 R0=0x00000000 R1=0x00000000
 filter 5 MODE=idmask SCALE=32 ACT=0 R0=0x00000000 R1=0x00000000
 filter 6 MODE=idmask SCALE=32 ACT=0 R0=0x00000000 R1=0x00000000
 filter 7 MODE=idmask SCALE=32 ACT=0 R0=0x00000000 R1=0x00000000
```

第一个参数为滤波器编号，带一个参数的filter命令输出当前滤波器设置，如：

- `filter 0`命令输出当前滤波器0的设置；
- `filter 1`命令输出当前滤波器1的设置。

可以使用该命令依次查看7个滤波器设置。

```
filter 0
 filter 0 MODE=idmask SCALE=32 ACT=1 R0=0x00000000 R1=0x00000000
filter 1
 filter 1 MODE=idmask SCALE=32 ACT=0 R0=0x00000000 R1=0x00000000
```

所有命令第一个参数为滤波器编号，UARTCAN支持编号0-7共8个滤波器，为了便于叙述，本节以0号滤波器为例描述。

filter 0 mode

设置滤波器滤波模式，滤波模式会影响其它参数的设置。

- `filter 0 mode 0`设置0号滤波器为`idmask`模式
- `filter 0 mode 1`设置0号滤波器为`idlist`模式

具体过程如下：

```
filter 0
 filter 0 MODE=idmask SCALE=32 ACT=1 R0=0x00000000 R1=0x00000000
filter 0 mode 1
 filter 0 MODE=idlist SCALE=32 ACT=1 R0=0x00000000 R1=0x00000000
filter 0 mode 0
 filter 0 MODE=idmask SCALE=32 ACT=1 R0=0x00000000 R1=0x00000000
```

filter 0 scale

设置滤波器位宽(16位或32位)。

- `filter 0 scale 0`设置0号滤波器为16位
- `filter 0 scale 1`设置0号滤波器为32位

```
filter 0 scale 0
 filter 0 MODE=idmask SCALE=16 ACT=1 R0=0x00000000 R1=0x00000000
filter 0 scale 1
 filter 0 MODE=idmask SCALE=32 ACT=1 R0=0x00000000 R1=0x00000000
filter 0 scale 0
 filter 0 MODE=idmask SCALE=16 ACT=1 R0=0x00000000 R1=0x00000000
```

filter 0 act

使能或者禁止滤波器。

- `filter 0 act 0`禁止0号滤波器
- `filter 0 act 1`使能0号滤波器

```
filter 0 act 0
 filter 0 MODE=idmask SCALE=16 ACT=0 R0=0x00000000 R1=0x00000000
filter 0 act 1
 filter 0 MODE=idmask SCALE=16 ACT=1 R0=0x00000000 R1=0x00000000
```

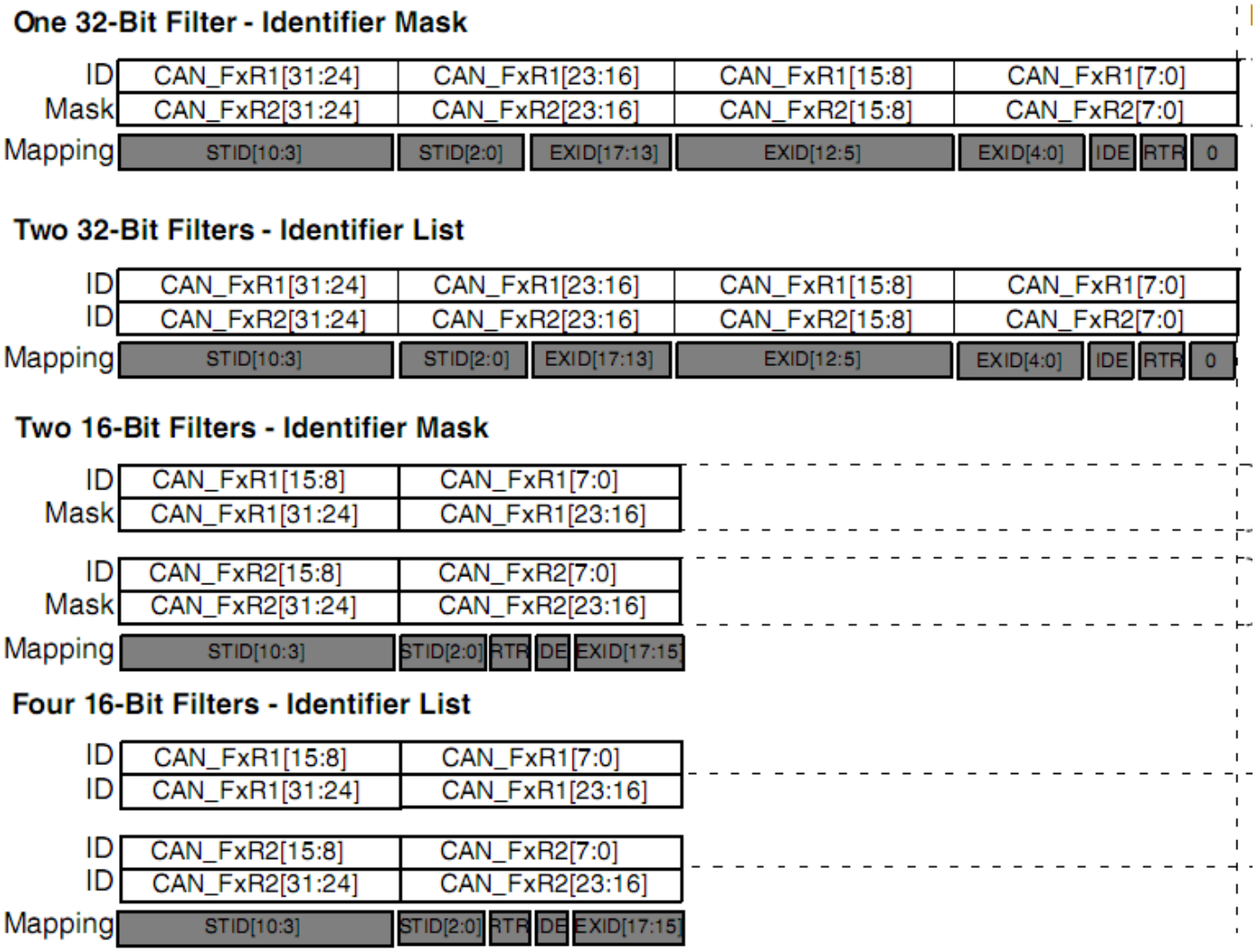
注意:CAN**接口必须使能至少一个滤波器才能顺利收到数据。**

filter 0 r0/r1

每一组滤波器工作模式分为`idmask`和`idlist`两种，每一组滤波器位宽可以为32位或者16位，因此存在4种组合方式：

- 1组32位idmask滤波器
- 2组32位idlist滤波器
- 2组16位idmask滤波器
- 4组16位idlist滤波器

四种组合方式及其ID映射见下图：



用户需要按照实际滤波需求计算出r0和r1，然后分别使用`filter 0 r0`和`filter 0 r1`命令设置r0和r1寄存器。

```
filter 0 r0 11
 filter 0 MODE=idmask SCALE=16 ACT=1 R0=0x00000011 R1=0x00000000
filter 0 r1 22
 filter 0 MODE=idmask SCALE=16 ACT=1 R0=0x00000011 R1=0x00000022
```

param

操作设备参数。

命令格式：`param [load|save|restore] Operate parameters.`

param命令带三个子命令：load、save、restore。

- `param load`从内置EEPROM加载保存的参数。
- `param save`保存参数到内置EEPROM。
- `param restore`恢复默认参数。

reboot

重启系统。

可以带一个延时参数，单位ms，如`reboot 900`延时900ms以后重启。

命令输出如下：

```
reboot
 UARTCAN v20.7.26 SN:FF570679804957573951871 7
 ECHO Studio <echo.xjtu@gmail.com>. All Rights Reserved.
reboot 900
 UARTCAN v20.7.26 SN:FF570679804957573951871 7
 ECHO Studio <echo.xjtu@gmail.com>. All Rights Reserved.
```

help

获取在线帮助。

命令输出如下：

```
help
 can -> can [baud|mode|speed|clear] CAN port setup.
 std -> std [hexID] [hexDATA|remote] Send standard data/remote message.
 ext -> ext [hexID] [hexDATA|remote] Send extended data/remote message.
 env -> env [show|add|cnt|prd] Set/Show environment variables.
 uart -> uart [baud|mode|addr] UART port setup.
 filter -> filter [0-7] [mode|scale|act|r0|r1] [param] CAN filter setup.
 param -> param [load|save|restore] Operate parameters.
 reboot -> reboot [delay ms] Restart system.
 help -> help Info.
 version -> display SW version and SN.
```

version

获取固件和设备序列号等信息。

命令输出如下：

```
version
 UARTCAN v20.7.26 SN:FF570679804957573951871 7
 ECHO Studio <echo.xjtu@gmail.com>. All Rights Reserved.
```