CAN / UART / MODBUS Quick Reference

Industrial Interfaces on Linux on One Page

Concrete examples using **can0**, **/dev/beiis-uart-i-a** and **/dev/beiis-modbus-ii-a**.
Interface and permission changes normally require **sudo**.

1. SocketCAN

Find and Configure the Interface

`ip -br link show type can`	List CAN interfaces
`sudo ip link set can0 down`	Stop before changing settings
`sudo ip link set can0 type can bitrate 500000`	Classic CAN at 500 kbit/s
`sudo ip link set can0 type can bitrate 1000000 dbitrate 8000000 fd on`	CAN FD: 1 / 8 Mbit/s
`sudo ip link set can0 up`	Bring the interface up

Receive, Send and Generate Traffic

`candump -tz -x can0`	Show frames with timestamps
`candump can0,123:7FF`	Only CAN ID 0x123
`cansend can0 123#DEADBEEF`	Send a Classical CAN frame
`cansend can0 123##1DEADBEEF`	CAN FD frame with BRS
`cangen can0 -g 10 -I i -L 8 -D i`	Send a test frame every 10 ms

Status and Troubleshooting

`ip -details -statistics link show can0`	Bitrate, state and counters
`candump -e can0`	Show decoded error frames
`sudo ip link set can0 type can restart-ms 100`	Automatic bus-off recovery
`journalctl -k -g can0`	Kernel messages for the interface

No frames? Check bitrate, termination, common ground and CAN-H/CAN-L.

2. UART / TTY

Find the Device

`ls -l /dev/serial/by-id/`	Stable USB serial names
`ls -l /dev/beiis-uart-i-a`	Check alias target and permissions
`journalctl -k -g tty`	Detected TTY devices
`udevadm info -q property -n /dev/beiis-uart-i-a`	Show udev properties

115200 Baud, 8N1, No Flow Control

`stty -F /dev/beiis-uart-i-a 115200 cs8 -cstopb -parenb -ixon -ixoff -crtscts raw -echo`	Configure the port
`stty -F /dev/beiis-uart-i-a -a`	Show current settings
`picocom -b 115200 /dev/beiis-uart-i-a`	Interactive terminal
`printf 'AT\r\n' > /dev/beiis-uart-i-a`	Send text with CR/LF
`timeout 5 hexdump -C < /dev/beiis-uart-i-a`	Read binary data for five seconds

Permissions and Busy Ports

`groups`	Show current groups
`sudo usermod -aG dialout $USER`	Allow serial access; log in again
`fuser -v /dev/beiis-uart-i-a`	Find the process using the port
`sudo systemctl stop serial-getty@ttyAMA0.service`	Stop a console getty when required

Unreadable characters usually mean mismatched baud rate, parity, data bits or stop bits.

3. Modbus mit `mbpoll`

Read and Write Modbus RTU

`mbpoll -m rtu -a 1 -b 19200 -P even -r 1 -c 4 -t 4 /dev/beiis-modbus-ii-a`	Read holding registers 1–4
`mbpoll -m rtu -a 1 -b 19200 -P even -r 1 -c 4 -t 3 -1 /dev/beiis-modbus-ii-a`	Read input registers once
`mbpoll -m rtu -a 1 -b 19200 -P even -r 10 -t 4 /dev/beiis-modbus-ii-a 123`	Write 123 to holding register 10

Modbus TCP

`mbpoll -m tcp -a 1 -r 1 -c 4 -t 4 192.168.1.50`	Read four holding registers
`nc -vz 192.168.1.50 502`	Check TCP port 502
`sudo tcpdump -ni eth0 tcp port 502`	Observe Modbus TCP traffic

Data Types and Addressing

`-t 0`	Read/write coils
`-t 1`	Read discrete inputs
`-t 3`	Read input registers
`-t 4`	Read/write holding registers
`-1`	Poll only once
`-0`	Use zero-based PDU addressing
`-v`	Verbose diagnostics

Typical Modbus Errors

Timeout: check slave ID, baud rate, parity, A/B, ground and timeout.

CRC: check serial settings, termination, shielding and noise.

Exception: check function, register range and write permission.

Important: Four Common Causes of Communication Problems

SocketCAN

All nodes need the same bitrate. Always set the interface **down** before changing it.

UART

Baud rate, data bits, parity, stop bits and flow control must match on both sides.

Modbus

Documentation often mixes 40001 notation, one-based references and zero-based PDU addresses. Use **-0** deliberately.

RS-485

Terminate only both ends of the bus. A/B naming is not consistent across vendors.

BE-IIS
BRECHEL ELECTRONIC
Industrial Interface Solutions

Local help: `man ip`, `man candump`, `man stty`, `man mbpoll`
Official SocketCAN reference:
https://docs.kernel.org/networking/can.html