Inhaltsverzeichnis

PEAK Traces in SocketCan umwandeln

Jedes CAN- Programm hat ja scheinbar sein eigenes Trace Format.

Um die Traces der PEAK CAN Programme unter Linux über die CAN- Tools abspielen zu können, hilft ein kleines Python Script:

trc2socketcanlog.py

```
import sys
import re
import argparse



if __name__=="__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "-i",
        "--input",
        help="file to handle",
        required=True
    )
    parser.add_argument(
        "-m",
        "--multiple",
        help="use this flag if the trace contains multiple busses",
        action='store_true'
    )

    args = parser.parse_args()
    regex = re.compile(r"^0+", re.IGNORECASE)
    with open(args.input, encoding="utf8") as fin:
        line="foo"
        while line:
            line = fin.readline()
            if line and line[:1]!=";": # remove comments
                line=line.rstrip() # remove CR
                f = filter(None, line.split(' ')) # suppress empty
fields
                elements=list(f)
                time=float(elements[1]) / 1000
                if args.multiple: # extended format
                    id=elements[4]
                    bus=elements[2]

                    while len(id)>1 and id[0]=="0" and id[1] in
"0123456789":
                        id=id[1:] # remove leading "0"s, if the next
```

```
char is not a number. Otherways canplayer would throw an error
                    print(f"({time:0.6f}) can{bus}
{id}#{''.join(elements[7:])}")
                else:
                    id=elements[3]
                    id = regex.sub("", id) # remove leading "0"s
                    while len(id)>1 and id[0]=="0" and id[1] in
"0123456789":
                        id=id[1:] # remove leading "0"s, if the next
char is not a number. Otherways canplayer would throw an error
                    print(f"({time:0.6f}) can0
{id}#{''.join(elements[5:])}")
```

Damit konvertiert man dann entweder den PEAK Trace in das Format, was vom `canplayer` abgespielt werden kann

```
python3 trc2socketcanlog.py peak_trace.trc  > playback.log
canplayer playback.log
```

oder wenn's nur eine einmalige Ausgabe bleiben soll, kann man die Konvertierung auch ohne Umweg über eine Datei direkt auf den CAN-Bus schieben:

```
python3 trc2socketcanlog.py peak_trace.trc  | canplayer
```

Falls im Trace mehrere Busse sind, kann der canplayer die Kanäle mappen (Ziel=Quelle)

```
canplayer -v -l i -I trace.log can0=can1 can1=can2 can1=can3
```