NanoVNA Console Commands:

LR - July 2 2019 - Original
LR - July 5 2019 - Added better command output descriptions
LR – Aug 6 2019 – Cleaned up formatting

According to edy555, the original designer of the NanoVNA, the following commands were originally developed to assist in the development and debugging of the device.

help – prints most of the following commands – some are not enabled but appear in the source-code

exit – exits and then restarts console mode
outputs: ChibiOS/RT Shell
info – prints firmware info on LCD:
Kernel: 4.0.0
Compiler: GCC 5.4.1 20160919
Architecture: ARMv6-M
Core Variant: Cortex-M0
Port Info: Preemption through NMI
Platform: STM32F072xB Entry Level Medium Density devices
Board: NanoVNA
Build time: May 5 2019 - 08:54:38
echo – Simple echo routine to echo input back to terminal output.
Usage: echo "message"
systime – outputs binary time from 1/1/1980:
Output example: 81225068
threads – prints running threads:

```
stklimit    stack     addr refs prio     state         name
20000200 2000039C 20001BF8    0  128    WTEXIT         main
20001AF8 20001B3C 20001B90    0    1     READY         idle
20002910 20002BA4 20002C98    0  128     READY        sweep
20002CD0 20002D4C 20002F18    0  129   CURRENT       noname
```

reset - `Performing reset`
freq – Usage: freq {`frequency(Hz)`}
offset - `Usage: offset {frequency offset(Hz)}`
time – get the time in:
`timespec.year+1980, timespec.month, timespec.day, timespec.millisecond`
outputs: 1980/1/1 8203499

How to set the time??

dac - `usage: dac {value(0-4095)}`
saveconfig - `Config saved.`
clearconfig - `usage: clearconfig {protection key}`
`Where key is hardcoded as "1234"`
`A wrong input gives: "Key unmatched" error.`
`Correct key gives: "Config and all cal data cleared"`
data - `usage: data [array]`
`example:` **`data [100 200]`**
`produces output of:`

```
0.999902069 -0.003914059
0.999517679 -0.003805434
0.999603867 -0.003855689
0.999514162 -0.004347831
```

...up to…

```
0.998815953 -0.006629021
0.998574435 -0.006352890
0.997710704 -0.006602317
0.998088240 -0.006423361
0.998069703 -0.006385779
0.998046994 -0.006417039
```

NOTE: need to determine what the output parameters above represent.

dump - (if enabled) outputs dump buffer:

```
FB8D FF8D F84C FF8A F824 FF85 FA9B FF80 FEE5 FF7A 03C6 FF74 075A FF74 0783 FF79
03FF FF81 FE9C FF88 F9C5 FF89 F7C2 FF8C F8C7 FF85 FC41 FF81 00ED FF7C 058C FF75
07F6 FF74 0672 FF7C 01B5 FF86 FC35 FF8E F8D2 FF8B F803 FF88 FA0C FF84 FE31 FF7C
02F2 FF76 06DE FF75 07D1 FF78 04DB FF7E FF8A FF87 FA7C FF8A F7FF FF8B F868 FF83
FB74 FF7F 0028 FF79 04E6 FF73 07B7 FF75 06FC FF7C 02CA FF85 FD45 FF8A F90C FF8D
F7C2 FF8E F9A3 FF84 FD5A FF7B 021A FF78 0677 FF74 07EE FF75 056B FF7F 0066 FF85
```

gamma – if enabled in F/W, Prints `gamma[0]& gamma[1]`

frequencies – prints a long list of all the sweep points from start to stop

```
50000
9049500
18049000
27048500
36048000
45047500
…up to…
846003000
855002500
864002000
873001500
882001000
891000500
900000000
```

port – Sets port of DSP `tlv320aic3204`

`usage: port {0:TX 1:RX}`

stat – outputs processor status

```
average: -138 -137
rms: 3597 8
callback count: 9126700
awd: 27
```

gain – sets gain of `DSP tlv320aic3204`

`usage: gain {lgain(0-95)} [rgain(0-95)]`

power - `usage: power {0-3}`

sweep - `usage: sweep {start(Hz)} [stop] [points]`

`if no inputs, then prints current setup eg:300000000 500000000 101`

NOTE: Unconfirmed:

otherwise, can force new sweep settings using the following commands:

start
stop
center
span
cw
set – sets sweep points
start
stop

test – **unknown how to use** – appears to set freq span from 10MHz to 90MHz and involves one of the LEDs and touchscreen input

touchcal – produces calibration touch points in the upper left and lower right corners

outputs: `"first touch upper left, then lower right..."`
`"done"`
`"touch cal params: A B C D (upper left and lower right x-y co-ords)`

touchtest – test touch accuracy – touch, hold and drag pointer to draw a line on the display

pause – pause LCD display output

resume – resume LCD display output

cal - `usage: cal [load|open|short|thru|isoln|done|reset|on|off|in]`

save - `save {id} where 'id' is from 0 to 4`

recall - `recall {id} where 'id' is from 0 to 4`

trace – prints status of either 2 or 4 traces depending on F/W version used:

```
0 SWR        CH0 1.000000000 0.000000000
1 LOGMAG     CH1 1.000000000 7.000000000
2 SMITH      CH0 1.000000000 0.000000000
3 PHASE      CH1 1.000000000 4.000000000
```

marker – Usage: `marker [n] [off|{index}]`

`Example Output with no parameters supplied: 1 0 100000000`

edelay – electrical delay in **picoseconds**

Usage: `edelay [n]  where n is in picoseconds`

`Output with no parameters supplied shows current setting.`

`Eg: 2`0.000000000

Units are unknown but most likely picoseconds and the decimal points are probably due to the Firmware library routine used.