```
================================================================
------------------------------------------------------------
   FILA 10G Command Reference   (15/05/2012)
------------------------------------------------------------
================================================================
```

The FILA10G serial console runs at 19200,N,8,1. You may
use the <TAB> key to display a list of commands. While
typing a command, the <TAB> key will display the syntax
of that command.

```
FiLa10G% <TAB>
  ... => list of commands

FiLa10G% command <TAB>
  ... => syntax of command
```

Below is a list of commands and their arguments.
Note that <arg> arguments are obligatory and
arguments marked [<arg>] are optional. Alternative
values for one argument are denoted by <opt1|opt2|opt3>.

Most commands accept values and addresses specified in
decimal or in hexadecimal. Note that hex is entered also as
"xFF" instead of "0xFF"!

```
================================================================
```

Syntax: arp <on|off>

   Enables/disables automatic ARP queries.

```
------------------------------------------------------------
```

Syntax: dataselect <5B|5C|VDIF>

   Data format selection between 5B/5C/VDIF. Case insensitive.

   At present 5C is not implemented

```
------------------------------------------------------------
```

Syntax: listdev

  Lists the devices available in the design.

```
  This is a list of device names, their type and their
  memory address. Most devices are mapped into software
  address space and their memory can be accessed with
  the low-level commands 'read', 'write'.

  Most devices however have an own set of configuration
  commands and read/write are not recommended.

  The most useful output of 'listdev' are the names
  of the devices visible to the software.

-------------------------------------------------------------

Syntax: read <start_address> [<end_address>]

  Reads and prints words from memory.
  Addresses can be decimal or hex.

-------------------------------------------------------------

Syntax: regwrite <bank name> <register nr> <value>

   Writes the value into a 32-bit register in a register bank.

   Bank name:
      name of register bank (use command 'listdev' for names)

   Register nr:
      number of register in the bank, starts from 0

   Value:
      unsigned decimal or hex

   The list of registers and their meaning is documented
   other documentation. The most important are:

    write
     0 = VDIF word 0 header content, bit30=0 std, =1 legacy
     1 = VDIF word1 header content bit 31=0 header,=1 no header, bit30=0 1pps wait, =1 no wait,
            Bit1=0 TVG data rate 2Gbps, =1 4Gbps
     2 = VDIF word2 header content bit 31-24 (VV) + packet size-1 in 8-bytes unit (SSSSSS),
         as 0xVVSSSSSS
     3 = VDIF word3 header content bit 31-26 (Vvv) + packet number-1 in 1 second,
         as 0xVvvxxNNNNNN
     4 = VDIF word 4 header content
    5 = VDIF word 5,6,7 header content

    6 = UDP/IP data stream eth1 destination IP address
    7 = UDP/IP data stream eth1 destination port
    read/write
     8  = Configuration bits
         {0=arm, 2,1=inputsrc, 4,3=outformat, 5=rst, 9,8=tvgmode}
     9  = UDP/IP data stream eth0 destination port
     10 = two-letter station ID in the low bits 15-0
     11 = UDP/IP data stream eth0 destination IP address
     13 = RTC base second in decimal (automatically used by 'timesync')
     14 = years since 2000
     15 = RTC base MJD in decimal (automatically used by 'timesync')
    read-only
     26 = BCD time code version of the current RTC value
     28 = current RTC seconds count
     29 = growing count of 64-bit words read from input FIFO
     30 = status bits

------------------------------------------------------------

Syntax: regread <bank name> <register nr>

  Reads and prints a 32-bit value from the specified register
  in a register bank.

  Bank name:
    name of register bank (use command 'listdev' for names)

  Register nr:
    number of register in the bank, starts from 0

------------------------------------------------------------

Syntax: reset

  Resets the FIFO and RTC.

  In practice this clears all data from the FIFOs and restarts
  the real-time clock on the next 1PPS. Correct time
  sync is attempted (RTC value of previous second +1) but
  is not really guaranteed.

------------------------------------------------------------

Syntax: sysstat
```

Shows system status bits.

This includes FIFO status, 1PPS sync status, the selected
data source and test vector format, amongst much other information.

Syntax: tengbinfo <10Gb core name>

Retreives the current parameters of a core. Prints
the ARP table of the core (table of IP=>MAC mapping).

Syntax: tick

Toggles 1PPS tick display.

If toggled on, the current RTC time value is printed
every time it changes. This should happen roughly
at each 1PPS boundary, but due to software and serial
console lag it is not an exact indication.

----

syntax: time

Prints current value of the RTC

Syntax: timesync <sec> <mjd>

Initiates time synchronization of User Logic RTC to next 1PPS.

Sec:
  second of day
MJD:
  integer days in modified julian date

The source of 1PPS depends in the 'inputselect'ed data source.
Note that if the input is the TVG test vector generator, it
operates standalone from onboard oscillators, your system will
not be in sync with any GPS or station 1PPS time reference.

If the input is any of the VSI connectors, the clock is synced to the 1PPS coming from there. True sync of course requires that this external VSI data source itself be synchronized before synchronizing the FILA10G to it.

Syntax: tvectmode <All-0|All-1|VSI-H|cnt32>

Change test vector generation mode. Case insensitive.

The VSI-H mode is a pseudorandom noise sequence that restarts on internal 1PPS. It is an implementation of http://www.haystack.mit.edu/tech/vlbi/mark5/mark5_memos/016.pdf

----

Syntax: tengbarp <10Gb core name> <ip last byte> <MAC xx:xx:xx:xx:xx:xx>

Two 10Gb core are implemented, named eth0 and eth1.

Sets one ARP entry in a 10G core.

Only the last byte of an IP address is specified, for example 15 instead of 192.168.1.15.

The MAC address is case insensitive.

This command is useful for entering a Multicast MAC address into the table.

For Mark5C recording, a fake MAC needs to be entered for its IP address.

In case of the Mark5C the command is needed to enter a fake MAC for its IP address, since the 5C card is not able to respond to network queries.

Syntax: tengbcfg <10Gb core name>

Sets the parameters of a 10G core.
Somewhat similar to Linux 'ifconfig'.

Examples:

```
  tengbcfg eth0 mac=ba:dc:af:e4:be:e1
```

```
tengbcfg eth0 ip=192.168.1.10 gateway=192.168.1.1 nm=27
tengbcfg eth0 port=46227
```

mac: interface MAC address
Note that changing the MAC will also clear the ARP table!

ip: interface IP address
It is simultaneously the subnet address prefix so e.g. 192.168.1.10 would mean IP is 10 and subnet is 192.168.1.x

nm: netmask of subnet in CIDR notation

http://en.wikipedia.org/wiki/CIDR_notation
For example 192.168.1.0/24 means nm=24

The ARP table of the 10G core has 255 entries, limiting the subnet to 255 addresses. For this reason you must have nm>=24.

gateway: gateway IP through which to reach devices outside own subnet. Naturally the gateway IP must be on own subnet to be reachable itself!

port: incoming UDP/IP port
Data arriving to this port number is forwarded to FPGA user logic instead of the control software.

Currently not really used. A future implementation of a "10GbE UDP/IP => Mark5B/VDIF de-framer => VSI output" processing system would use this port.

Syntax: version

Displays info about current design

Syntax: write <access_type (b|s|l)> <address> <data>

Writes data to memory.

```
 Access type:
   b = write a byte  (8-bit)
   s = write a short (16-bit)
   l = write a long  (32-bit)

 Address:
   decimal or hex

 Data:
   unsigned decimal or hex

------------------------------------------------------------

------------------------------------------------------------

Syntax: inputselect <vsi1|vsi2|vsi1-2|tvg>

  Data format selection between vsi1|vsi2|vsi1-2|tvg. Case insensitive.

------------------------------------------------------------
```