# YANG model for management of Network Tester

- IETF119 Hackathon
- March 16-17, 2023
- Online

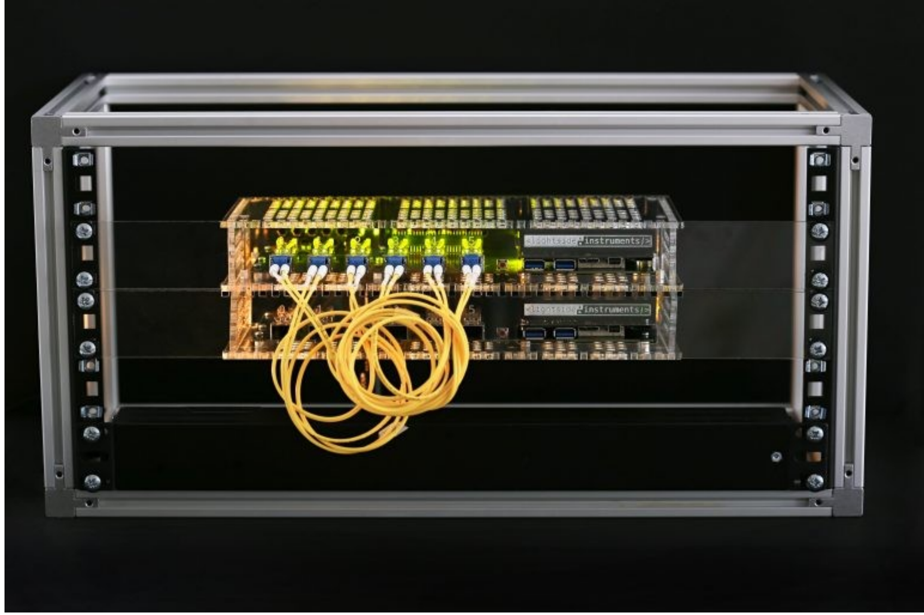
```
      +-----------------+
 eth0 |                 | eth1
   +-<|TG   tester0   TA|<-+
   |  |                 |  |
   |  +-----------------+  |
   |        +-----+        |
   +------->| DUT |>-------+
            +-----+

      +-----------------+
 eth0 |                 | eth1
   +-<|TG   tester0     |x
   |  |                 |
   |  +-----------------+
+-----+
| DUT |
+-----+
   |  +-----------------+
   |  |                 |
   +->|TA   tester1     |
      |                 |
      +-----------------+
```

## Network Tester Management Solutions

* Command line (SCPI)
* Cisco TRex
* Keysight Open Traffic Generator APIs & Data Models (2023)
* Other

* YANG model

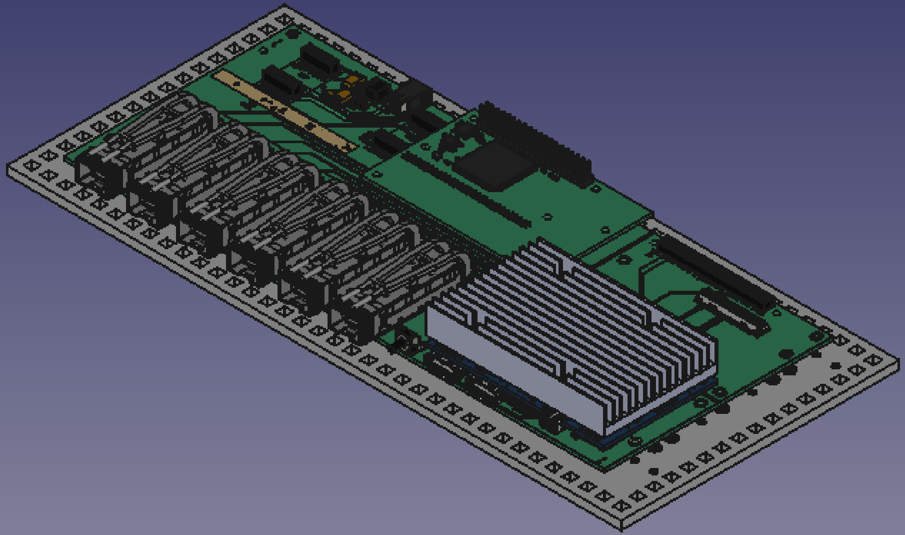
## The project



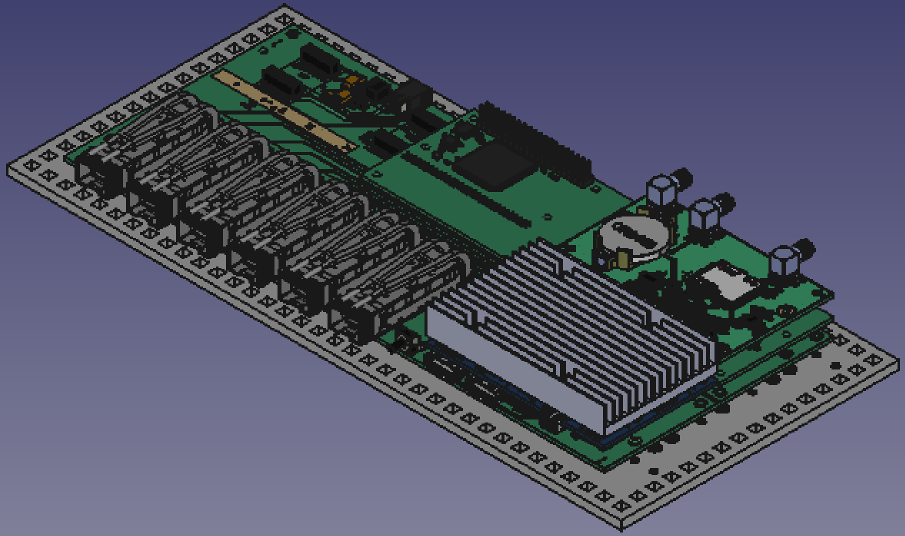
Specification:
* draft-ietf-bmwg-network-tester-cfg-04

Client side:
* Test script – rfc2544-benchmark.py (Python)

Device side:
* Software - YANG/NETCONF server instrumentation code (C)
* Firmware - (Verilog)
* Hardware – off-the-shelf FPGA module Ultra96 + 6x SFP+ network programmability kit shield ( KiCAD, Walk-through, OSHWA UIDs NO000005, NO000006)
* Pre-silicon gate level simulation with cocotb/iverilog as alternative to target hardware

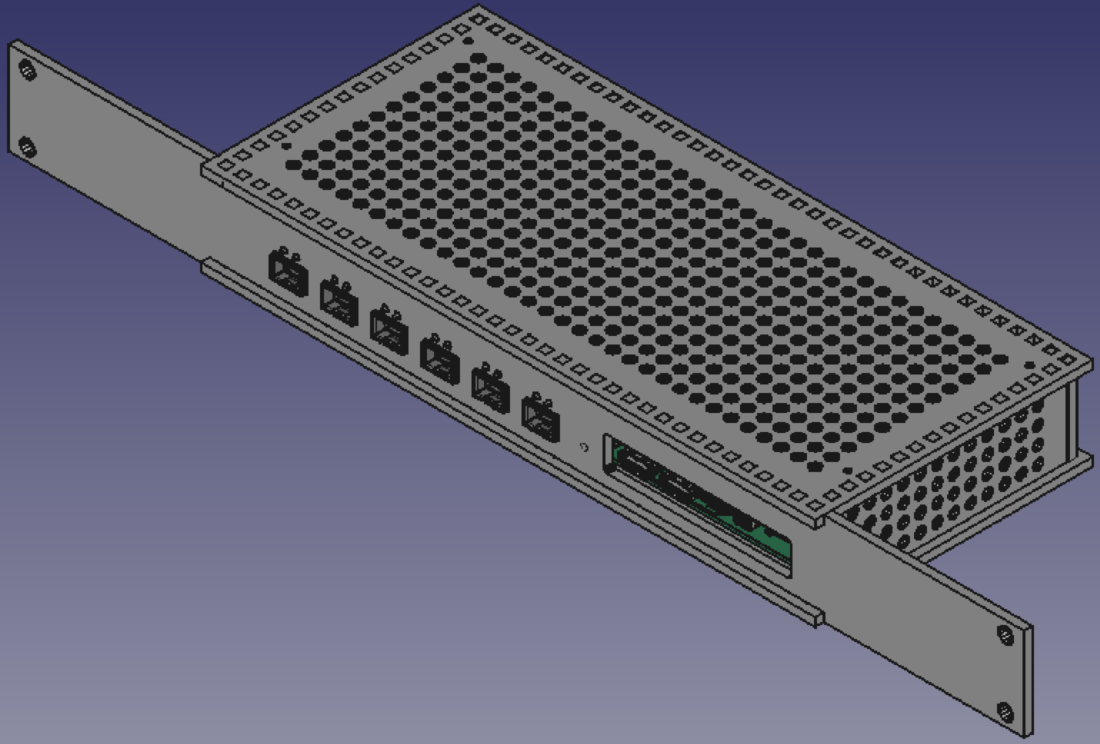


OSHW
NO000005



OSHW
NO000005

OSHW
NO000006



OSHW
NO000005

OSHW
NO000006

## YANG tree diagram of the models

The following slide contains the complete YANG tree diagram
of the ietf-traffic-generator.yang and ietf-traffic-analyzer.yang modules

```
module: ietf-traffic-analyzer
  augment /if:interfaces/if:interface:
    +--rw traffic-analyzer!
       +--rw testframe-filter! {testframe-filter}?
       |  +--rw type    identityref
       |  +--rw mask?   string
       |  +--rw data?   string
       +--rw capture {capture}?
       |  +--rw start-trigger
       |  |  +--rw (start-trigger)?
       |  |     +--:(frame-index)
       |  |     |  +--rw frame-index?       uint64
       |  |     +--:(testframe-index)
       |  |        +--rw testframe-index?   uint64
       |  +--rw stop-trigger
       |     +--rw (stop-trigger)?
       |        +--:(when-full)
       |           +--rw when-full?   empty
       +--ro state
          +--ro pkts?              yang:counter64
          +--ro octets?            yang:counter64
          +--ro idle-octets?       yang:counter64 {idle-octets-counter}?
          +--ro errors?            yang:counter64
          +--ro testframe-stats
          |  +--ro testframe-pkts?    yang:counter64
          |  +--ro sequence-errors?   yang:counter64
          |  +--ro payload-errors?    yang:counter64
          |  +--ro latency
          |     +--ro samples?   uint64
          |     +--ro min?       uint64
          |     +--ro max?       uint64
          |     +--ro average?   uint64
          |     +--ro latest?    uint64
          +--ro capture {capture}?
             +--ro frame* [sequence-number]
                +--ro sequence-number    uint64
                +--ro timestamp?         yang:date-and-time
                +--ro length?            uint32
                +--ro data?              string
```

```
module: ietf-traffic-generator
  augment /if:interfaces/if:interface:
    +--rw traffic-generator
       +--rw (type)?
       |  +--:(single-stream)
       |  |  +--rw testframe-type?     identityref
       |  |  +--rw frame-size          uint32
       |  |  +--rw frame-data?         string
       |  |  +--rw interframe-gap      uint32
       |  |  +--rw interburst-gap?     uint32
       |  |  +--rw frames-per-burst?   uint32
       |  |  +--rw modifiers
       |  |     +--rw modifier* [id]
       |  |        +--rw id             uint32
       |  |        +--rw action         identityref
       |  |        +--rw offset         uint32
       |  |        +--rw mask           string
       |  |        +--rw repetitions    uint32
       |  +--:(multi-stream)
       |     +--rw streams
       |        +--rw stream* [id]
       |           +--rw id                   uint32
       |           +--rw testframe-type?      identityref
       |           +--rw frame-size           uint32
       |           +--rw frame-data?          string
       |           +--rw interframe-gap       uint32
       |           +--rw interburst-gap?      uint32
       |           +--rw frames-per-burst?    uint32
       |           +--rw frames-per-stream    uint32
       |           +--rw interstream-gap      uint32
       |           +--rw modifiers
       |              +--rw modifier* [id]
       |                 +--rw id             uint32
       |                 +--rw action         identityref
       |                 +--rw offset         uint32
       |                 +--rw mask           string
       |                 +--rw repetitions    uint32
       +--rw realtime-epoch?
       |        yang:date-and-time {realtime-epoch}?
       +--rw total-frames?
```

## Design and implementation

```
NETCONF Server (Model (YANG), Implementation Generator module (C), Analyzer module (C))

TRAFFIC-GENERATOR-SW (C)              TRAFFIC-ANALYZER-SW (C)
  |                                      |
Socket API                             Socket API
  |               \   |   /              |
Kernel      Sync ->{RTCLOCK}(Verilog)  Kernel
  |          PPS  /   |   \              |
 DMA                                    DMA
  | [AXI]                                | [AXI]
 MAC  TRAFFIC-GENERATOR-HW (C,Verilog)  MAC    TRAFFIC-ANALYZER-HW (C,Verilog)
    \   /                                 |       |
     \ /                                  |       |
   GMII_MUX                               +---+---+
      | [GMII]                                | [GMII]
     PHY                                     PHY
      |                                       |
    SFP+ TX                                 SFP+ RX
      |                                       |
      +->----------------------------------->-+
```

* - underlined text has links to repositories

## Some management transaction examples follow:

1. Configuration of 64 octet packet stream with dynamic timestamps
   with minial interframe gap on a traffic generator

2. Configuration of testframe filter with bitfield matching

3. Get counters and status information from the traffic anazlizer

* Notice the use of automated command line serialization with **yangcli**

# 1. Configure traffic generation:

```
yangcli user@192.168.4.145> create /interfaces/interface[name='eth0']/traffic-generator -- frame-size=64 interframe-gap=20 \
testframe-type=dynamic \
frame-data=123456789ABCDEF012345678080045000002E000000000A112CBCC0000201C0000202C0200007001A00000\
00102030405060708090A0B0C0D0E0F10119CD50E0F

<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <candidate/>
  </target>
  <default-operation>merge</default-operation>
  <test-option>set</test-option>
  <config>
    <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
      <interface>
        <name>eth0</name>
        <traffic-generator
          xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
          nc:operation="create"
          xmlns="urn:ietf:params:xml:ns:yang:ietf-traffic-generator">
          <testframe-type
            xmlns:nttg="urn:ietf:params:xml:ns:yang:ietf-traffic-generator">nttg:dynamic</testframe-type>
          <frame-size>64</frame-size>
          <frame-data>123456789ABCDEF012345678080045000002E000000000A112CBCC0000201C
0000202C0200007001A0000000102030405060708090A0B0C0D0E0F10119CD50E0F</frame-data>
          <interframe-gap>20</interframe-gap>
        </traffic-generator>
      </interface>
    </interfaces>
  </config>
</edit-config>
```

# 2. Configure test frame filter:

```
yangcli user@192.168.4.145> create /interfaces/interface[name='eth1']/traffic-analyzer/testframe-
filter
 -- type=bit-field-match data="123456789ABCDEF012345678"
mask="000000000000FFFFFFFFFFFF"

<edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <target>
    <candidate/>
  </target>
  <default-operation>merge</default-operation>
  <test-option>set</test-option>
  <config>
    <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
      <interface>
        <name>eth1</name>
        <traffic-analyzer xmlns="urn:ietf:params:xml:ns:yang:ietf-traffic-analyzer">
          <testframe-filter
            xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
            nc:operation="create">
            <type
              xmlns:ntta="urn:ietf:params:xml:ns:yang:ietf-traffic-analyzer">ntta:bit-field-match</type>
            <mask>000000000000FFFFFFFFFFFF</mask>
            <data>123456789ABCDEF012345678</data>
          </testframe-filter>
        </traffic-analyzer>
      </interface>
    </interfaces>
  </config>
</edit-config>
```

# 3. Get status information:

```
yangcli user@192.168.4.145>
xget /interfaces/interface/traffic-analyzer/state

<get xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <filter type="xpath"
    select="/interfaces/interface/traffic-analyzer/state"/>
</get>
```

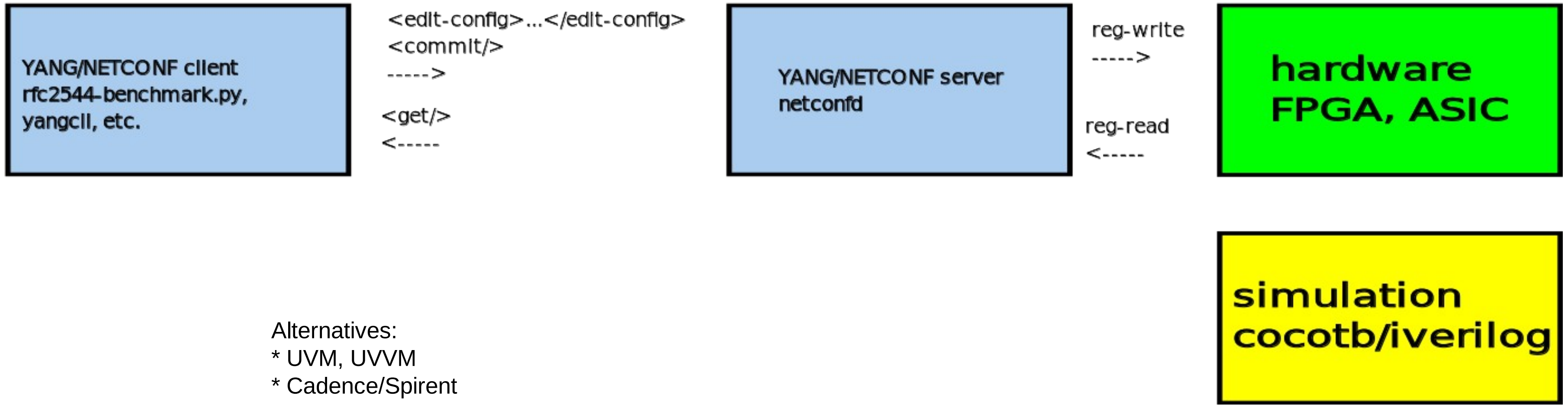
```
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <data>
    <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
      <interface>
        <name>eth1</name>
        <traffic-analyzer xmlns="urn:ietf:params:xml:ns:yang:ietf-traffic-analyzer">
          <state>
            <pkts>43200851</pkts>
            <octets>2764854464</octets>
            <octets-idle>562950384</octets-idle>
            <bad-crc-octets>0</bad-crc-octets>
            <bad-crc-pkts>0</bad-crc-pkts>
            <bad-preamble-octets>0</bad-preamble-octets>
            <bad-preamble-pkts>0</bad-preamble-pkts>
            <octets-total>3630210805</octets-total>
            <testframe-stats>
              <pkts>43200851</pkts>
              <sequence-errors>0</sequence-errors>
              <latency>
                <samples>43200851</samples>
                <min-sec>0</min-sec>
                <min>832</min>
                <max-sec>0</max-sec>
                <max>864</max>
                <last-sec>0</last-sec>
                <last>864</last>
              </latency>
            </testframe-stats>
            <capture>
              <timestamp>
                <nsec>902536272</nsec>
              </timestamp>
              <sequence-number>43200851</sequence-number>
              <data>555555555555D5123456789ABCDEF012345678080045000002E000000000A112CBCC0000201
C0000202C0200007001A0000000000000029331520000000005E735CB98F0345964C7</data>
            </capture>
          </state>
        </traffic-analyzer>
      </interface>
    </interfaces>
  </data>
</rpc-reply>
```

## Tasks

* Implement -04 updates. Bitwise mask filter for testframes.

* Validate with pre-silicon test environment.

## Model defined pre-silicon testing environment



Alternatives:
* UVM, UVVM
* Cadence/Spirent

...
sim-run 1000 ns
sim-finish

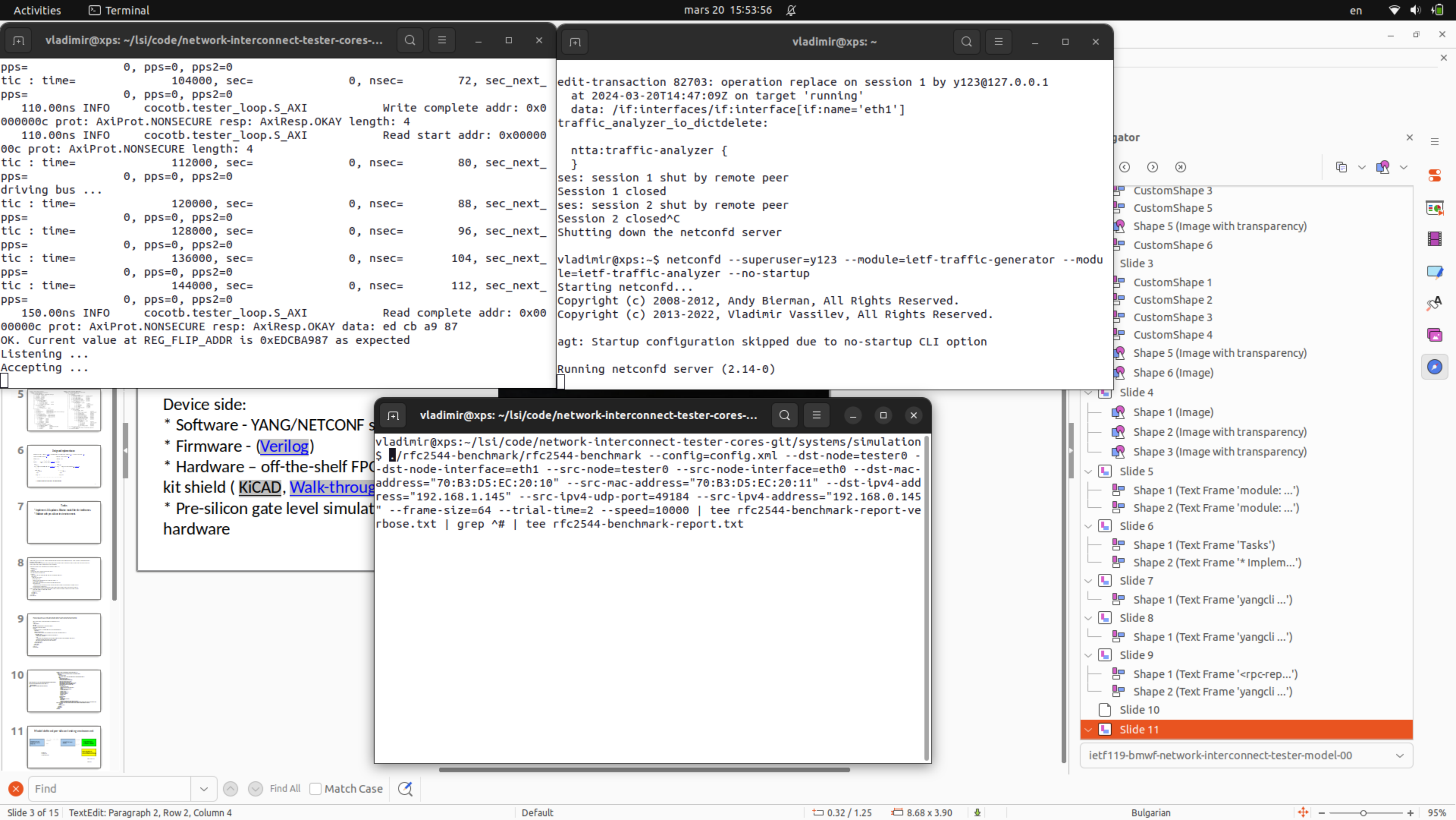
## Results:

We managed to run a RFC2544 benchmark against a **netconfd** server implementing the model by controling a gate-level simulation of the synthesizble traffic-generator-gmii and traffic-analyzer-gmii cores in **cocotb** sim_time_ns=1565330 (!!! Notice that we used bogus 10 Kb Ethernet speed to actually simulate the dataplane in realtime).

We published the results in a branch :

* Waveform trace (cocotb/iverilog gate-level generated)
* Report (with verbose NETCONF transaction)

Short RFC2544 report and some random screenshots complete this presentation.



Activities Terminal mars 20 15:53:56 en

vladimir@xps: ~/lsi/code/network-interconnect-tester-cores-...

```
pps=              0, pps=0, pps2=0
tic : time=          104000, sec=            0, nsec=        72, sec_next_
pps=              0, pps=0, pps2=0
   110.00ns INFO     cocotb.tester_loop.S_AXI           Write complete addr: 0x0
000000c prot: AxiProt.NONSECURE resp: AxiResp.OKAY length: 4
   110.00ns INFO     cocotb.tester_loop.S_AXI           Read start addr: 0x00000
00c prot: AxiProt.NONSECURE length: 4
tic : time=          112000, sec=            0, nsec=        80, sec_next_
pps=              0, pps=0, pps2=0
driving bus ...
tic : time=          120000, sec=            0, nsec=        88, sec_next_
pps=              0, pps=0, pps2=0
tic : time=          128000, sec=            0, nsec=        96, sec_next_
pps=              0, pps=0, pps2=0
tic : time=          136000, sec=            0, nsec=       104, sec_next_
pps=              0, pps=0, pps2=0
tic : time=          144000, sec=            0, nsec=       112, sec_next_
pps=              0, pps=0, pps2=0
   150.00ns INFO     cocotb.tester_loop.S_AXI           Read complete addr: 0x00
00000c prot: AxiProt.NONSECURE resp: AxiResp.OKAY data: ed cb a9 87
OK. Current value at REG_FLIP_ADDR is 0xEDCBA987 as expected
Listening ...
Accepting ...
```

vladimir@xps: ~

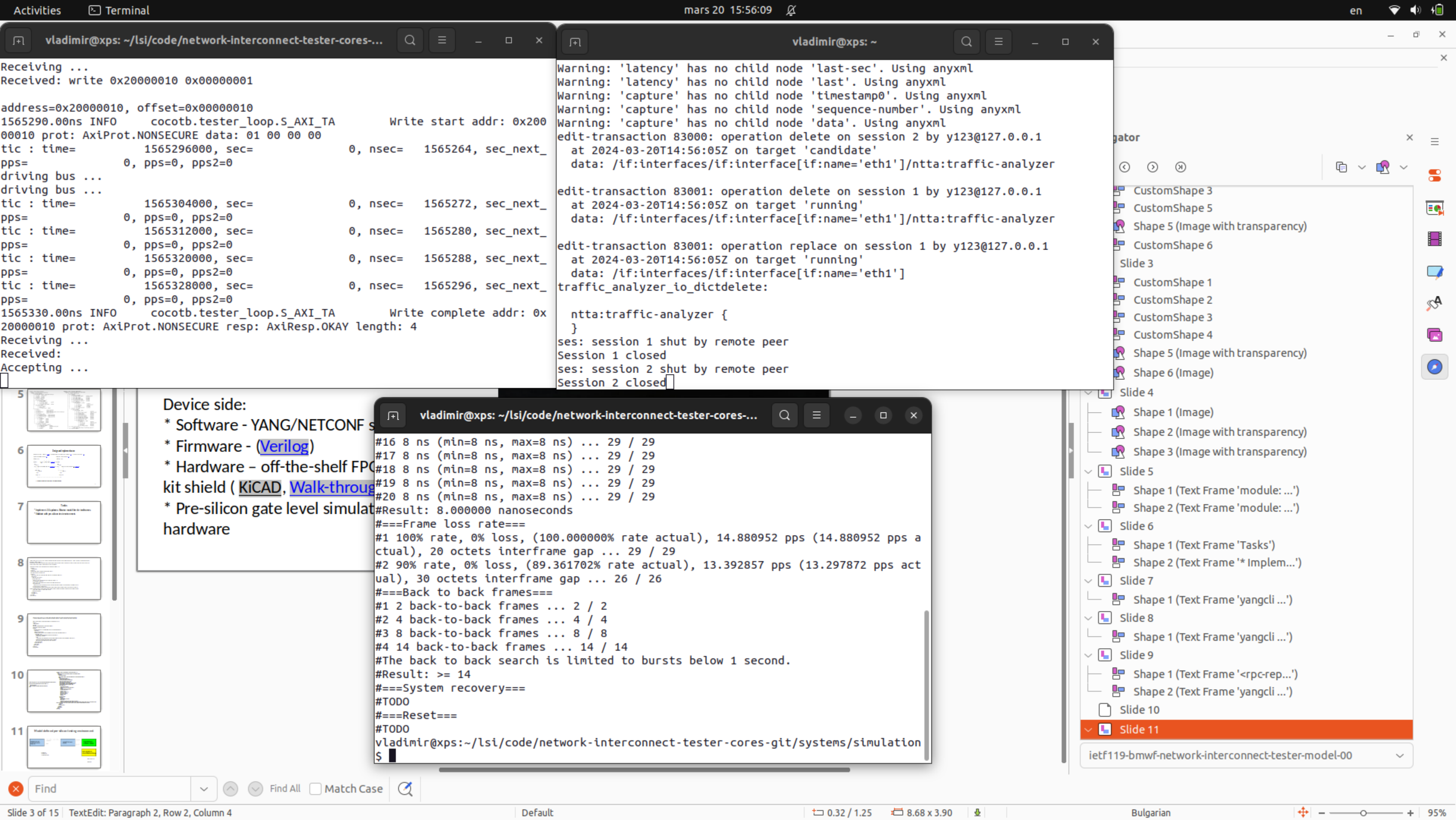
```
edit-transaction 82703: operation replace on session 1 by y123@127.0.0.1
  at 2024-03-20T14:47:09Z on target 'running'
  data: /if:interfaces/if:interface[if:name='eth1']
traffic_analyzer_io_dictdelete:

  ntta:traffic-analyzer {
  }
ses: session 1 shut by remote peer
Session 1 closed
ses: session 2 shut by remote peer
Session 2 closed^C
Shutting down the netconfd server

vladimir@xps:~$ netconfd --superuser=y123 --module=ietf-traffic-generator --modu
le=ietf-traffic-analyzer --no-startup
Starting netconfd...
Copyright (c) 2008-2012, Andy Bierman, All Rights Reserved.
Copyright (c) 2013-2022, Vladimir Vassilev, All Rights Reserved.

agt: Startup configuration skipped due to no-startup CLI option

Running netconfd server (2.14-0)
```

vladimir@xps: ~/lsi/code/network-interconnect-tester-cores-...

```
vladimir@xps:~/lsi/code/network-interconnect-tester-cores-git/systems/simulation
$ ./rfc2544-benchmark/rfc2544-benchmark --config=config.xml --dst-node=tester0 -
-dst-node-interface=eth1 --src-node=tester0 --src-node-interface=eth0 --dst-mac-
address="70:B3:D5:EC:20:10" --src-mac-address="70:B3:D5:EC:20:11" --dst-ipv4-add
ress="192.168.1.145" --src-ipv4-udp-port=49184 --src-ipv4-address="192.168.0.145
" --frame-size=64 --trial-time=2 --speed=10000 | tee rfc2544-benchmark-report-ve
rbose.txt | grep ^# | tee rfc2544-benchmark-report.txt
```

Device side:
* Software - YANG/NETCONF s
* Firmware - (Verilog)
* Hardware – off-the-shelf FPG
kit shield ( KiCAD, Walk-throug
* Pre-silicon gate level simulat
hardware

gator
CustomShape 3
CustomShape 5
Shape 5 (Image with transparency)
CustomShape 6
Slide 3
CustomShape 1
CustomShape 2
CustomShape 3
CustomShape 4
Shape 5 (Image with transparency)
Shape 6 (Image)
Slide 4
Shape 1 (Image)
Shape 2 (Image with transparency)
Shape 3 (Image with transparency)
Slide 5
Shape 1 (Text Frame 'module: ...')
Shape 2 (Text Frame 'module: ...')
Slide 6
Shape 1 (Text Frame 'Tasks')
Shape 2 (Text Frame '* Implem...')
Slide 7
Shape 1 (Text Frame 'yangcli ...')
Slide 8
Shape 1 (Text Frame 'yangcli ...')
Slide 9
Shape 1 (Text Frame '<rpc-rep...')
Shape 2 (Text Frame 'yangcli ...')
Slide 10
Slide 11
ietf119-bmwf-network-interconnect-tester-model-00
Find
Find All
Match Case
Slide 3 of 15 TextEdit: Paragraph 2, Row 2, Column 4 Default 0.32 / 1.25 8.68 x 3.90 Bulgarian 95%

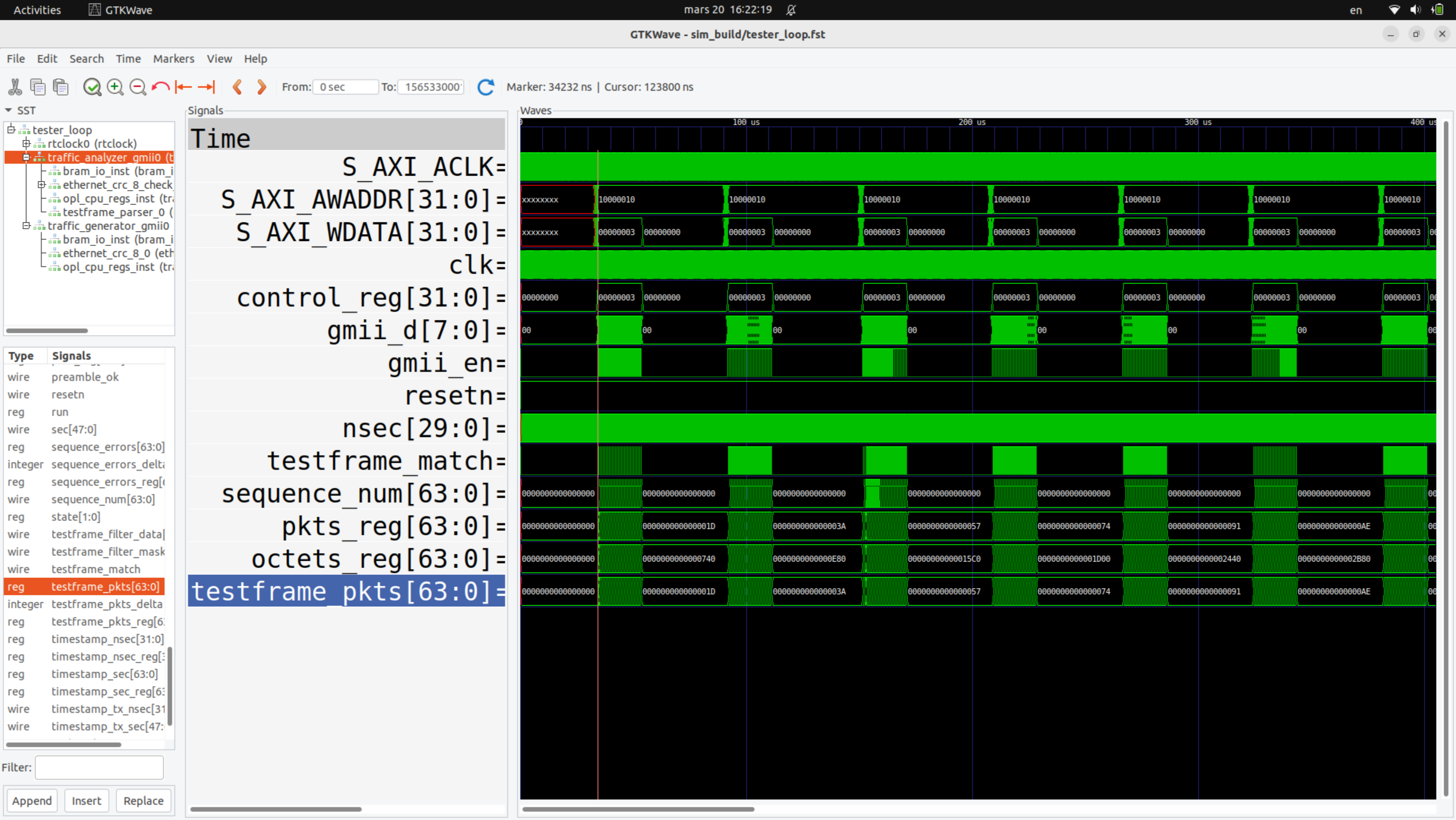

Activities
Terminal
mars 20 15:56:09
en
vladimir@xps: ~/lsi/code/network-interconnect-tester-cores-...
vladimir@xps: ~
gator
CustomShape 3
CustomShape 5
Shape 5 (Image with transparency)
CustomShape 6
Slide 3
CustomShape 1
CustomShape 2
CustomShape 3
CustomShape 4
Shape 5 (Image with transparency)
Shape 6 (Image)
Slide 4
Shape 1 (Image)
Shape 2 (Image with transparency)
Shape 3 (Image with transparency)
Slide 5
Shape 1 (Text Frame 'module: ...')
Shape 2 (Text Frame 'module: ...')
Slide 6
Shape 1 (Text Frame 'Tasks')
Shape 2 (Text Frame '* Implem...')
Slide 7
Shape 1 (Text Frame 'yangcli ...')
Slide 8
Shape 1 (Text Frame 'yangcli ...')
Slide 9
Shape 1 (Text Frame '<rpc-rep...')
Shape 2 (Text Frame 'yangcli ...')
Slide 10
Slide 11
ietf119-bmwf-network-interconnect-tester-model-00
5
6
7
8
9
10
11
Find
Find All
Match Case
Slide 3 of 15
TextEdit: Paragraph 2, Row 2, Column 4
Default
0.32 / 1.25
8.68 x 3.90
Bulgarian
95%

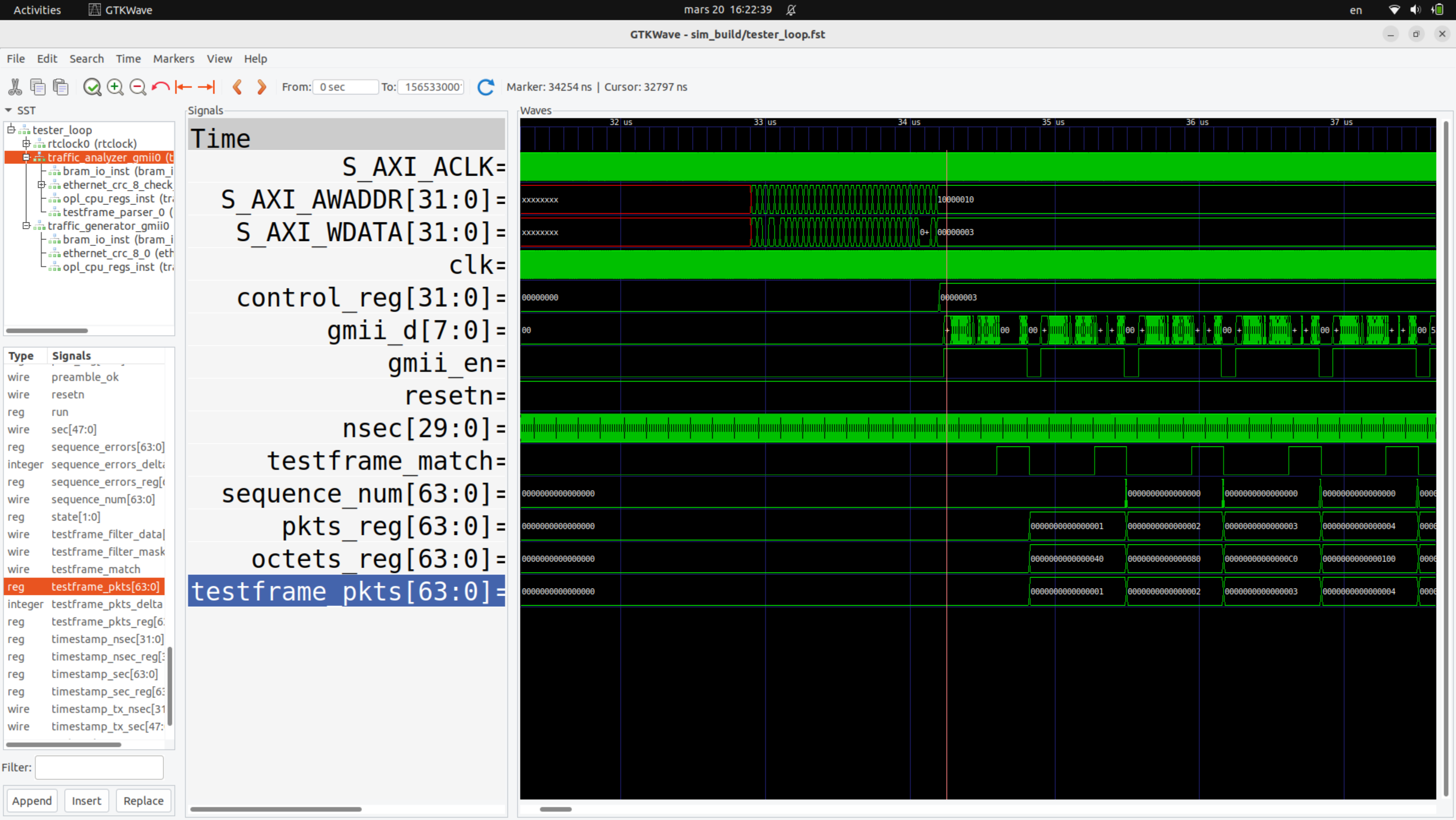
```
Receiving ...
Received: write 0x20000010 0x00000001

address=0x20000010, offset=0x00000010
1565290.00ns INFO     cocotb.tester_loop.S_AXI_TA       Write start addr: 0x200
00010 prot: AxiProt.NONSECURE data: 01 00 00 00
tic : time=         1565296000, sec=             0, nsec=   1565264, sec_next_
pps=            0, pps=0, pps2=0
driving bus ...
driving bus ...
tic : time=         1565304000, sec=             0, nsec=   1565272, sec_next_
pps=            0, pps=0, pps2=0
tic : time=         1565312000, sec=             0, nsec=   1565280, sec_next_
pps=            0, pps=0, pps2=0
tic : time=         1565320000, sec=             0, nsec=   1565288, sec_next_
pps=            0, pps=0, pps2=0
tic : time=         1565328000, sec=             0, nsec=   1565296, sec_next_
pps=            0, pps=0, pps2=0
1565330.00ns INFO     cocotb.tester_loop.S_AXI_TA       Write complete addr: 0x
20000010 prot: AxiProt.NONSECURE resp: AxiResp.OKAY length: 4
Receiving ...
Received:
Accepting ...
```

```
Warning: 'latency' has no child node 'last-sec'. Using anyxml
Warning: 'latency' has no child node 'last'. Using anyxml
Warning: 'capture' has no child node 'timestamp0'. Using anyxml
Warning: 'capture' has no child node 'sequence-number'. Using anyxml
Warning: 'capture' has no child node 'data'. Using anyxml
edit-transaction 83000: operation delete on session 2 by y123@127.0.0.1
  at 2024-03-20T14:56:05Z on target 'candidate'
  data: /if:interfaces/if:interface[if:name='eth1']/ntta:traffic-analyzer

edit-transaction 83001: operation delete on session 1 by y123@127.0.0.1
  at 2024-03-20T14:56:05Z on target 'running'
  data: /if:interfaces/if:interface[if:name='eth1']/ntta:traffic-analyzer

edit-transaction 83001: operation replace on session 1 by y123@127.0.0.1
  at 2024-03-20T14:56:05Z on target 'running'
  data: /if:interfaces/if:interface[if:name='eth1']
traffic_analyzer_io_dictdelete:

  ntta:traffic-analyzer {
  }
ses: session 1 shut by remote peer
Session 1 closed
ses: session 2 shut by remote peer
Session 2 closed
```

Device side:
* Software - YANG/NETCONF s
* Firmware - (Verilog)
* Hardware – off-the-shelf FP
kit shield ( KiCAD, Walk-throug
* Pre-silicon gate level simulat
hardware

```
#16 8 ns (min=8 ns, max=8 ns) ... 29 / 29
#17 8 ns (min=8 ns, max=8 ns) ... 29 / 29
#18 8 ns (min=8 ns, max=8 ns) ... 29 / 29
#19 8 ns (min=8 ns, max=8 ns) ... 29 / 29
#20 8 ns (min=8 ns, max=8 ns) ... 29 / 29
#Result: 8.000000 nanoseconds
#===Frame loss rate===
#1 100% rate, 0% loss, (100.000000% rate actual), 14.880952 pps (14.880952 pps a
ctual), 20 octets interframe gap ... 29 / 29
#2 90% rate, 0% loss, (89.361702% rate actual), 13.392857 pps (13.297872 pps act
ual), 30 octets interframe gap ... 26 / 26
#===Back to back frames===
#1 2 back-to-back frames ... 2 / 2
#2 4 back-to-back frames ... 4 / 4
#3 8 back-to-back frames ... 8 / 8
#4 14 back-to-back frames ... 14 / 14
#The back to back search is limited to bursts below 1 second.
#Result: >= 14
#===System recovery===
#TODO
#===Reset===
#TODO
vladimir@xps:~/lsi/code/network-interconnect-tester-cores-git/systems/simulation
$
```


Activities
GTKWave
mars 20 16:22:19
en
GTKWave - sim_build/tester_loop.fst
File Edit Search Time Markers View Help
From: 0 sec
To: 156533000
Marker: 34232 ns | Cursor: 123800 ns
SST
tester_loop
rtclock0 (rtclock)
traffic_analyzer_gmii0 (t
bram_io_inst (bram_i
ethernet_crc_8_check
opl_cpu_regs_inst (tr
testframe_parser_0 (
traffic_generator_gmii0
bram_io_inst (bram_i
ethernet_crc_8_0 (eth
opl_cpu_regs_inst (tr
Type Signals
wire preamble_ok
wire resetn
reg run
wire sec[47:0]
reg sequence_errors[63:0]
integer sequence_errors_delta
reg sequence_errors_reg[
wire sequence_num[63:0]
reg state[1:0]
wire testframe_filter_data[
wire testframe_filter_mask
wire testframe_match
reg testframe_pkts[63:0]
integer testframe_pkts_delta
reg testframe_pkts_reg[6
reg timestamp_nsec[31:0]
reg timestamp_nsec_reg[
reg timestamp_sec[63:0]
reg timestamp_sec_reg[6
wire timestamp_tx_nsec[31
wire timestamp_tx_sec[47:
Filter:
Append Insert Replace
Signals
Time
S_AXI_ACLK=
S_AXI_AWADDR[31:0]=
S_AXI_WDATA[31:0]=
clk=
control_reg[31:0]=
gmii_d[7:0]=
gmii_en=
resetn=
nsec[29:0]=
testframe_match=
sequence_num[63:0]=
pkts_reg[63:0]=
octets_reg[63:0]=
testframe_pkts[63:0]=
Waves
100 us
200 us
300 us
400 us
xxxxxxxx 10000010 10000010 10000010 10000010 10000010 10000010 10000010
xxxxxxxx 00000003 00000000 00000003 00000000 00000003 00000000 00000003 00000000 00000003 00000000 00000003 00000000 00000003
00000000 00000003 00000000 00000003 00000000 00000003 00000000 00000003 00000000 00000003 00000000 00000003 00000000 00000003
00 00 00 00 00 00 00
0000000000000000 0000000000000000 0000000000000000 0000000000000000 0000000000000000 0000000000000000 0000000000000000
0000000000000000 000000000000001D 000000000000003A 0000000000000057 0000000000000074 0000000000000091 00000000000000AE
0000000000000000 0000000000000740 0000000000000E80 00000000000015C0 0000000000001D00 0000000000002440 0000000000002B80
0000000000000000 000000000000001D 000000000000003A 0000000000000057 0000000000000074 0000000000000091 00000000000000AE



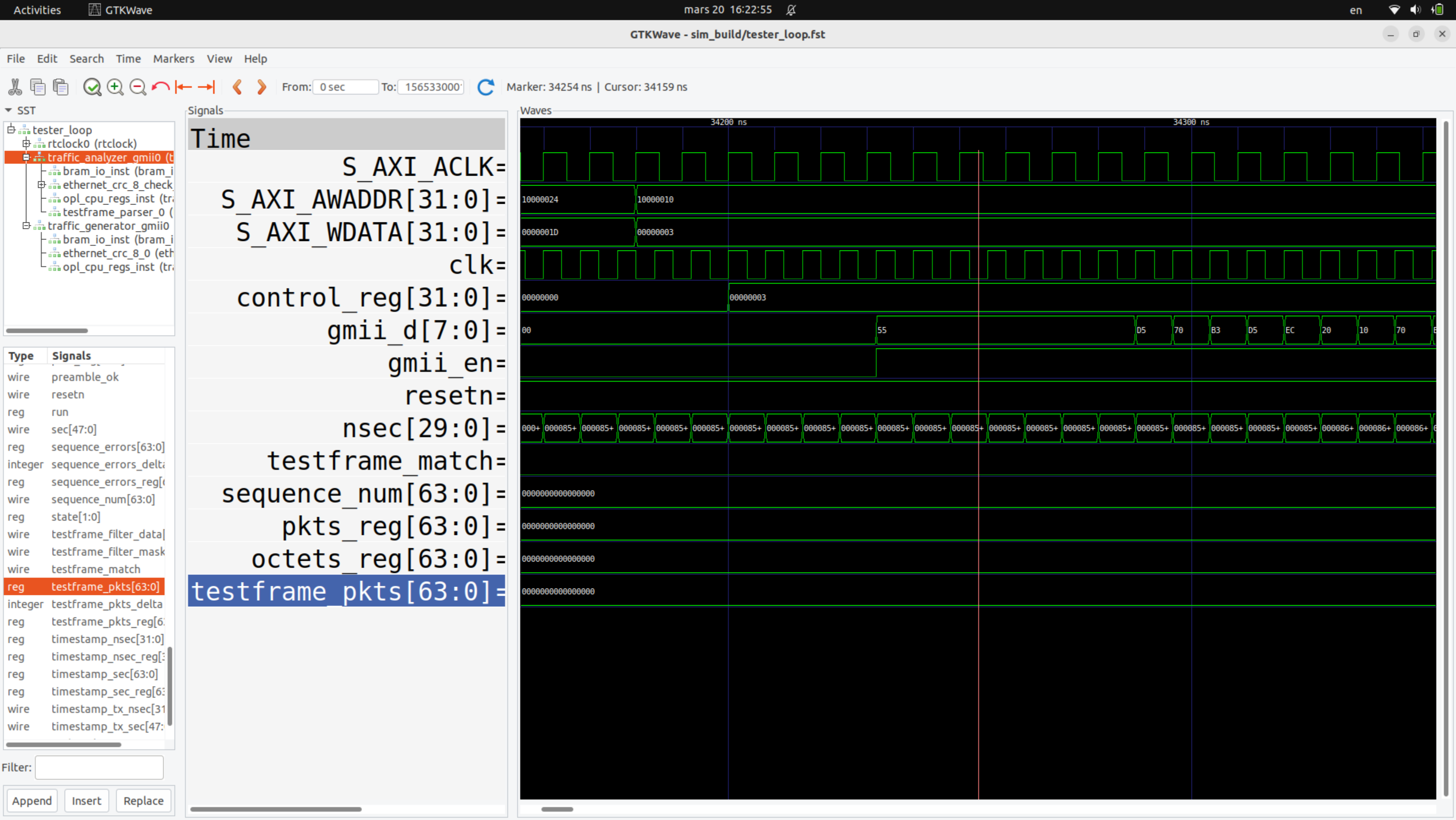
GTKWave - sim_build/tester_loop.fst

Marker: 34254 ns | Cursor: 32797 ns

```
tester_loop
  rtclock0 (rtclock)
  traffic_analyzer_gmii0 (t
    bram_io_inst (bram_i
    ethernet_crc_8_check
    opl_cpu_regs_inst (tr
    testframe_parser_0 (
  traffic_generator_gmii0
    bram_io_inst (bram_i
    ethernet_crc_8_0 (eth
    opl_cpu_regs_inst (tr
```

| Type | Signals |
|---|---|
| wire | preamble_ok |
| wire | resetn |
| reg | run |
| wire | sec[47:0] |
| reg | sequence_errors[63:0] |
| integer | sequence_errors_delta |
| reg | sequence_errors_reg[ |
| wire | sequence_num[63:0] |
| reg | state[1:0] |
| wire | testframe_filter_data[ |
| wire | testframe_filter_mask |
| wire | testframe_match |
| reg | testframe_pkts[63:0] |
| integer | testframe_pkts_delta |
| reg | testframe_pkts_reg[6 |
| reg | timestamp_nsec[31:0] |
| reg | timestamp_nsec_reg[3 |
| reg | timestamp_sec[63:0] |
| reg | timestamp_sec_reg[6 |
| wire | timestamp_tx_nsec[31 |
| wire | timestamp_tx_sec[47: |

```
Time
S_AXI_ACLK=
S_AXI_AWADDR[31:0]=    xxxxxxxx                    10000010
S_AXI_WDATA[31:0]=     xxxxxxxx                 0+ 00000003
clk=
control_reg[31:0]=     00000000                 00000003
gmii_d[7:0]=           00                       + 00 + 00 + 00 + 00 + 00 5
gmii_en=
resetn=
nsec[29:0]=
testframe_match=
sequence_num[63:0]=    0000000000000000    0000000000000000  0000000000000000  0000000000000000  000
pkts_reg[63:0]=        0000000000000000    0000000000000001  0000000000000002  0000000000000003  0000000000000004  000
octets_reg[63:0]=      0000000000000000    0000000000000040  0000000000000080  00000000000000C0  0000000000000100  000
testframe_pkts[63:0]=  0000000000000000    0000000000000001  0000000000000002  0000000000000003  0000000000000004  000
```



GTKWave - sim_build/tester_loop.fst

From: 0 sec To: 156533000 Marker: 34254 ns | Cursor: 34159 ns

```
tester_loop
  rtclock0 (rtclock)
  traffic_analyzer_gmii0 (t
    bram_io_inst (bram_i
    ethernet_crc_8_check
    opl_cpu_regs_inst (tr
    testframe_parser_0 (
  traffic_generator_gmii0
    bram_io_inst (bram_i
    ethernet_crc_8_0 (eth
    opl_cpu_regs_inst (tr
```

```
Type     Signals
wire     preamble_ok
wire     resetn
reg      run
wire     sec[47:0]
reg      sequence_errors[63:0]
integer  sequence_errors_delt
reg      sequence_errors_reg[
wire     sequence_num[63:0]
reg      state[1:0]
wire     testframe_filter_data[
wire     testframe_filter_mask
wire     testframe_match
reg      testframe_pkts[63:0]
integer  testframe_pkts_delta
reg      testframe_pkts_reg[6
reg      timestamp_nsec[31:0]
reg      timestamp_nsec_reg[
reg      timestamp_sec[63:0]
reg      timestamp_sec_reg[6
wire     timestamp_tx_nsec[31
wire     timestamp_tx_sec[47:
```

```
Time                      34200 ns                                  34300 ns
S_AXI_ACLK=
S_AXI_AWADDR[31:0]=       10000024   10000010
S_AXI_WDATA[31:0]=        0000001D   00000003
clk=
control_reg[31:0]=        00000000          00000003
gmii_d[7:0]=              00                      55                 D5 70 B3 D5 EC 20 10 70
gmii_en=
resetn=
nsec[29:0]=               000+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000085+ 000086+ 000086+ 000086+
testframe_match=
sequence_num[63:0]=       0000000000000000
pkts_reg[63:0]=           0000000000000000
octets_reg[63:0]=         0000000000000000
testframe_pkts[63:0]=     0000000000000000
```

## The End