1 Gb/s high throughput TCP sender core

… optimized for your needs:

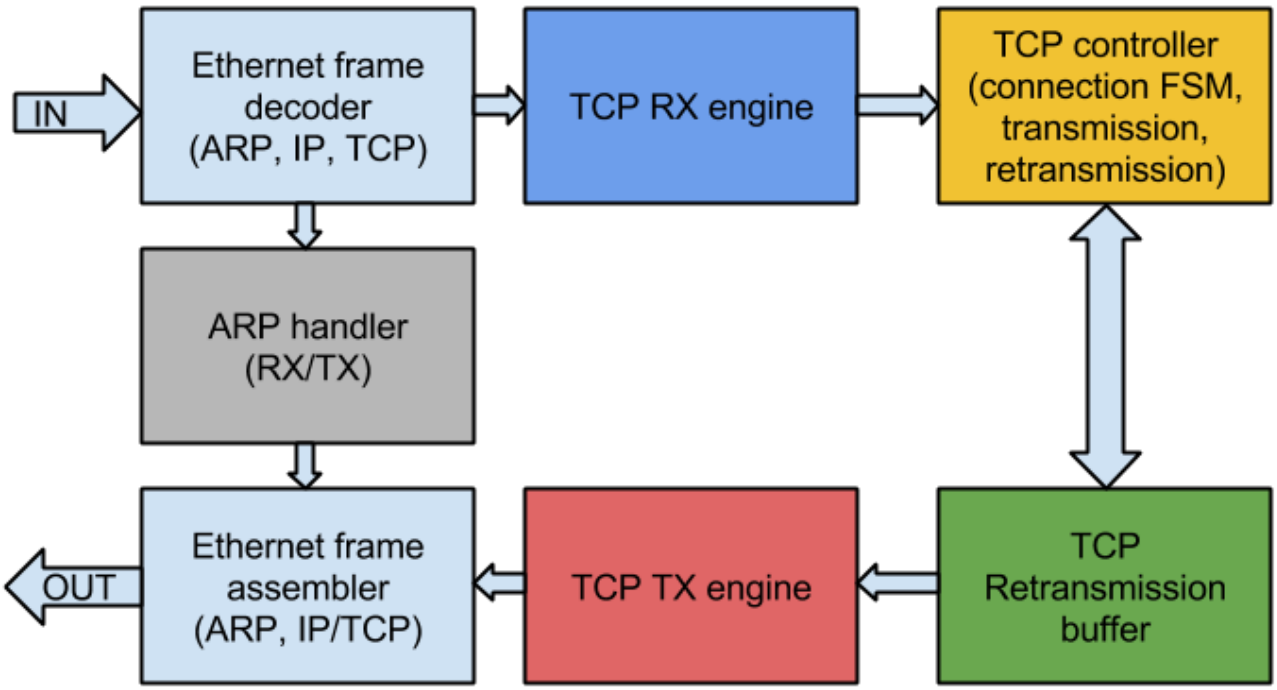
- RFC 675/RFC 793 compliant implementation
- field tested for industrial usage
- full HDL source code available for developers
- tested on CGEP-20 and CBOARD devices

AITIA's 1 Gb/s TCP sender core:

Features:

- RFC 675/RFC 793 compliant implementation
- Source code available for further development!
- Uses Xilinx Virtex 5/6 device specific GTX transceivers, can be easily ported to other devices too
- Fabric side interface is configurable for different 1Gb/s MAC modules
- 64 bit MAC data-interface running at 125 Mhz
- User side interface is configurable for different applications (64 bit@125MHz by default)
- tested on CBOARD/C-GEP 20 with SFP modules, and IBM compatible PC-s running Windos and Linux systems with standard NIC-s
- 8/16/32/64 Kbyte selectable internal buffer for retransmissions
- basic ARP protocol implementation (request,response)
- implements modified slow start, and keep alive sending
- implements hardware optimized fast retransmit, and conguestion avoidance algorithms
- Sample application available

The following figure shows the block diagram of the 1Gb/s TCP sender core



1 Gb/s TCP sender core functional description:

All major building modules of the 1 Gb/s TCP sender core are well separated according to their functions. This results in code thats easier to understand and develop, desing placement and partitioning is also much more convenient.
Because of the high interface count of the C-GEP devices we optimized the TCP sender core for minimum footprint and resource usage, and maximum application performance.

The main components are:

- TCP_sender_1G: this contains the 1 Gb/s TCP sender core, we have a common gmii for external Ethernet MAC interface connectivity outside the module. Also internal data input feed is implemented in a request-response handshake method.

    - tcp_sender_ctrl: This module controls the TCP connection state
      - handles ARP requests, and responses
      - maintains connections: SYN, ACK, RST/FIN TCP packets
      - sends keep-alive TCP packets if needed
      - handles timeout for dead connection checking
      - stores the network interface info (IP Host, and remote addresses, port numbers)
    - tcp_sender_tx:
      - handles TCP sending requests
      - controls TCP data and control frame assembly
    - tcp_sender_rx:
      - contains a simplified ethernet frame decoder for ARP and IP/TCP frames
      - signals TCP events: control flags, values (window size, ACK number, etc.)
    - tcp_retran_buffer:
      - controls read and write operations of the retransmission buffers
      - maintains TCP sequence number
      - processes received TCP ACK numbers, and window updates
      - implements TCP retransmission and conguestion avoidance algorithms
      - handles the retransmission timer

Interfacing to the core:

- Generic clock and control ports:

```
    clk                 : IN  STD_LOGIC;
    Rst                 : IN  STD_LOGIC;
    tcp_connected       : OUT STD_LOGIC;
    link_ok             : IN  STD_LOGIC;
```

The module the same clock (clk) as the output interface (usually an external 1Gb/s ethernet MAC). The Reset signal (Rst) resets the whole module, and disables its operation. The „tcp_connected" output shows if the TCP connection is up. The „link_up" signal is an optional input connected to the external MAC modules signal status indication.

```
    cfg_tcp_en_in       : IN  STD_LOGIC;
    cfg_smac_in         : IN  STD_LOGIC_VECTOR(6*8-1 downto 0 );
    cfg_sip_in          : IN  STD_LOGIC_VECTOR(4*8-1 downto 0   );
    cfg_dip_in          : IN  STD_LOGIC_VECTOR(4*8-1 downto 0   );
    cfg_gwip_in         : IN  STD_LOGIC_VECTOR(4*8-1 downto 0   );
    cfg_gwmask_in       : IN  STD_LOGIC_VECTOR(4*8-1 downto 0 );
    cfg_sport_in        : IN  STD_LOGIC_VECTOR(2*8-1 downto 0 );
    cfg_dport_in        : IN  STD_LOGIC_VECTOR(2*8-1 downto 0 );
```

The configuration ports above specify the TCP sender core's ethernet MAC address, IP address, port number, gateway IP address with netmask, and the receiver sides IP address, and port number. The new configurations take effect when the „cfg_tcp_en_in" input is driven high for a clock cycle.

```
    ts_1ms_in                           : IN  STD_LOGIC;
    ts_16ms_in                          : IN  STD_LOGIC;
    ts_1s_in                            : IN  STD_LOGIC;
```

These are the signals needed for the TCP timers (1 msec, 16msec, and 1 sec timer impulses).

- User ports (fabric):

```
    tcp_buffer_din      :  IN   STD_LOGIC_VECTOR(8*8-1  downto  0 );
    tcp_buffer_read     : OUT   STD_LOGIC;
    tcp_buffer_dav      : IN  STD_LOGIC;
```

64 bit data input interface for the TCP module. The application drivers „tcp_buffer_dav" high, if there is data to be send, and the TCP module accepts data by driving „tcp_buffer_read" high in response.

```
    rx_data_in              : IN  STD_LOGIC_VECTOR(7 downto 0 );
    rx_dav_in               : IN  STD_LOGIC;
    rx_sof_in               : IN  STD_LOGIC;
```

```
    rx_eof_in                : IN  STD_LOGIC;
    rx_crcerr_in             : IN  STD_LOGIC;
```

Receive Data from 1Gb/s Ethernet-MAC. Data is valid if „rx_dav_in" is high, start of frame, end of frame and crc error condition are signaled too. The Ethernet MAC is responsible for removing the preamble field.

```
    tx_data_out              : OUT STD_LOGIC_VECTOR(7 downto 0 );
    tx_sof_out               : OUT STD_LOGIC;
    tx_eof_out               : OUT STD_LOGIC;
    tx_dav_out               : OUT STD_LOGIC;
```

Transmit data for the 1 Gb/s Ethernet-MAC. The ARP and TCP frames are transmitted on this interface. The Ethernet MAC is responsible for adding the preamble, and the checksum fields.

Validation and performance:

except from a measurement report:

Functional tests of the TCP sender core on C-BOARD by using SGA10GD for traffic playback

Date of report: 2013.09.30 - 2013.10.01

Place: Aitia International Zrt. Budapest, Czetz János street 48-50

Testing by: Kovács László

Devices used:

- CBOARD Sn-015 (2013.09.10 firmware)
- SGA10GD card + PCAP Replayer firmware
- SGA-PCAP Replayer software v1.5.6
- CBOARD-tester program: cboard_tester v0.2
- testing PC: IBM compatible, Intel Core-I5 3GHz, 4GB RAM, 4 port Intel PRO 1000M server NIC
- LC-LC multimode optical cord for 2x10 Gb/s links
- 4 SR multimode 10 Gb/s XFP (Avago)
- 4 1 Gb/s copper SFP (Finisar) + patch cables

The following tests were performed:

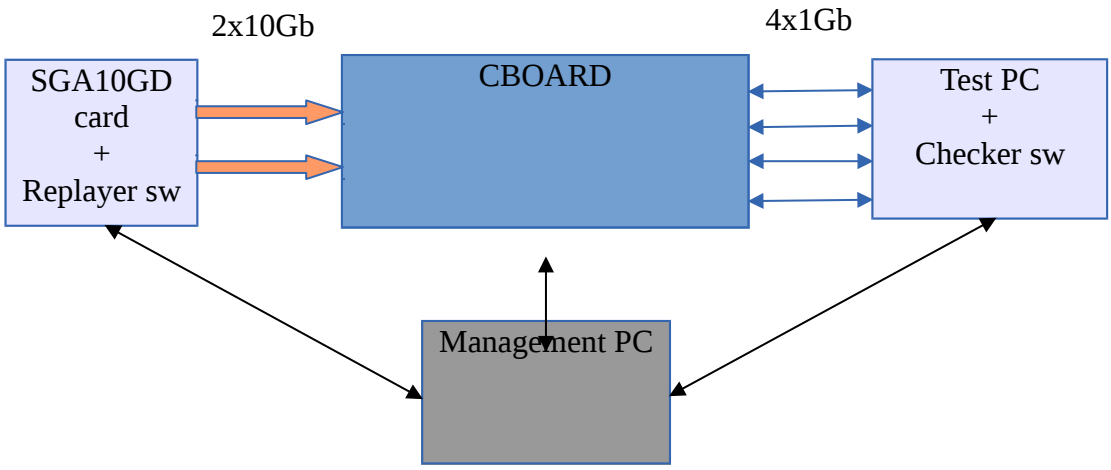
- prerecorded IP traffic playback, distributed by CBOARD (rules by IP address/protocol, even packet distribution)
- generated IP traffic, evenly distributed by CBOARD (stress-test)

Testing indicators:

- testing correct data reception by comparing the calculated and received checksum in the monitoring header

- count the incoming and sent packets, and check if the sequence number in the header increases by one for every received packet
- checking the filtering and distribution functions by sampling and by sequence number

Measure setup:



a. SGA10GD 10 Gb/s card, with pcap replayer application

b. CBOARD device

c. test-PC running the cboard_checker software

d. management PC: control CBOARD by webUI; control replayer, and monitor over remote-desktopon

Measurements 1.:

Replaying generated IP traffic, CBOARD routes all traffic to one output interface

Generator parameters:

- 2x100000 UDP packets
- Packet length: 64 - 1540 byte random value
- Inter-packet-gap: 64 - 50000 byte-time random value between ethernet frames
- Average transmission throughput: (2x)312 Mbps

Cboard_checker output:

```
Monitor Interface 0
Starting up CBOARD TCPCheck server
Connection from 192.168.10.1 Port: 7001
Contained Pack               0 Kbps

-- TCP Container stats
        Carrier TCP segs: 4697
        Sync lost count: 0
-- Source interface: 0
        Contained packs: 100000
        Contained plost: 0
        Check error cnt: 0
        Disordered pcnt: 0
-- Source interface: 1
        Contained packs: 100000
        Contained plost: 0
        Check error cnt: 0
        Disordered pcnt: 0

        Trash packets: 801
```

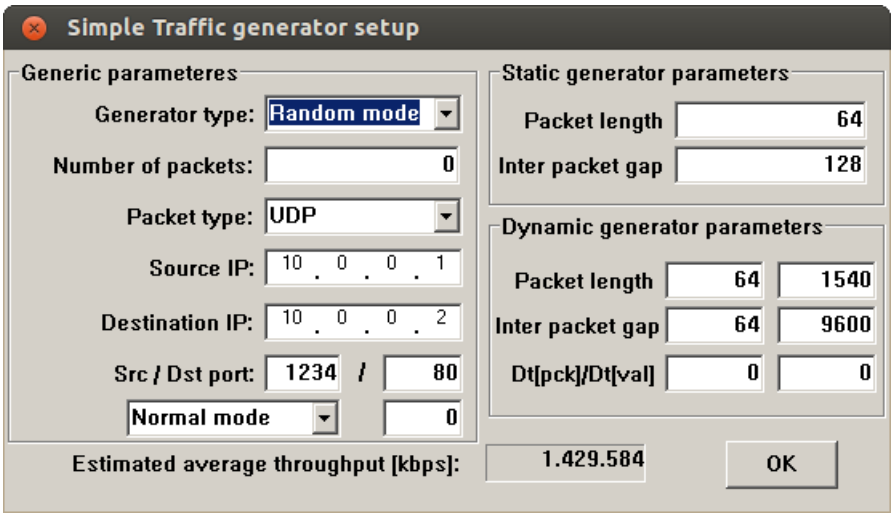
According to the programs output there was no packet loss or other error. The received frames were captured to disk too.

Measurement 2.:

Replay of generated, high speed IP traffic, and monitoring by CBOARD device without distribution. (monitor-interface stress test)

In this case the whole traffic was routed to a single interface, to see the maximal sustainable throughput of the TCP sender core.

Sending speed was set to to 2x1.4 Gb/s with random parameters:

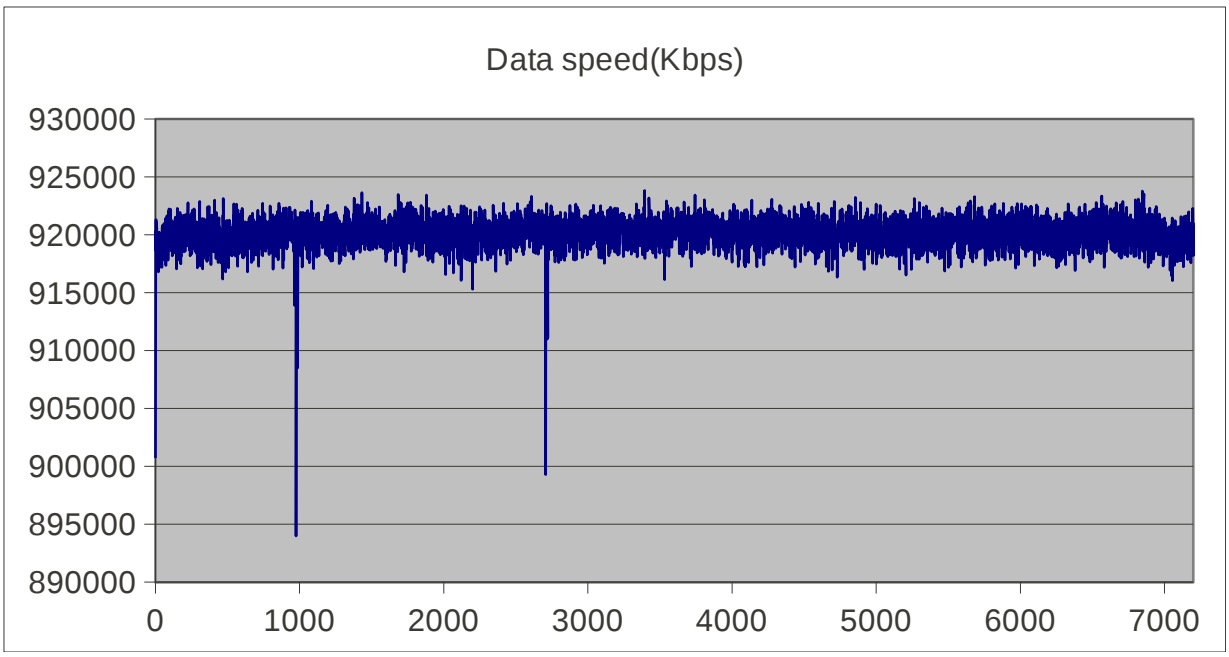


Az SGA Replayer IP csomaggenerátorának konfigurációja

The test was run for 2-2 and 1 hours on 3 different interfaces of the CBOARD.

There was no monitor synchron loss or checksum error during that time, but I observed a significant packet drop as we expected. (2x1.4Gbps in, 1x<1Gbps out)

The cboard_checker and windows network monitor showed an average network load of 900-920 Mb/s. The tester program checks the integrity of the packetstream, checks the monitoring header, so it only tests the maximal TCP throughput.



The drops on the 2 hour TCP traffic graph are caused by other processes running on the PC (windows update, remote desktop)

Cboard-tester log:

```
Monitor Interface 5: (L core)
Elapsed time: 02:15.15

Starting up CBOARD TCPCheck server
Connection from 192.168.10.104 Port: 7001
Contained Packets:        536813137

-- TCP Container stats
        Carrier TCP segs: 18930041
        Sync lost count: 0
-- Source interface: 0
        Contained packs: 260420622
        Contained plost: 27618343
        Check error cnt: 0
        Disordered pcnt: 0
-- Source interface: 1
        Contained packs: 276392515
        Contained plost: 27013254
        Check error cnt: 0
        Disordered pcnt: 0

------------------------------------------------
Monitor Interface 14: (M core)
Elapsed time: 02:00.06

Starting up CBOARD TCPCheck server
Connection from 192.168.10.112 Port: 7001
Contained Packets:        457911860

-- TCP Container stats
        Carrier TCP segs: 14188256
        Sync lost count: 0
-- Source interface: 0
        Contained packs: 262692136
        Contained plost: 28331793
```

```
        Check error cnt: 0
        Disordered pcnt: 0
-- Source interface: 1
        Contained packs: 195219724
        Contained plost: 29678065
        Check error cnt: 0
        Disordered pcnt: 0

------------------------------------------------
Monitor interface 0: (K core)
Elapsed time: 01:00.07

Starting up CBOARD TCPCheck server
Connection from 192.168.10.100 Port: 7001
Contained Packets:        187543193

-- TCP Container stats
        Carrier TCP segs: 5759629
        Sync lost count: 0
-- Source interface: 0
        Contained packs: 77565094
        Contained plost: 11510558
        Check error cnt: 0
        Disordered pcnt: 0
-- Source interface: 1
        Contained packs: 109978099
        Contained plost: 11035402
        Check error cnt: 0
        Disordered pcnt: 0
```

Many packet loss events, but no errored frames.