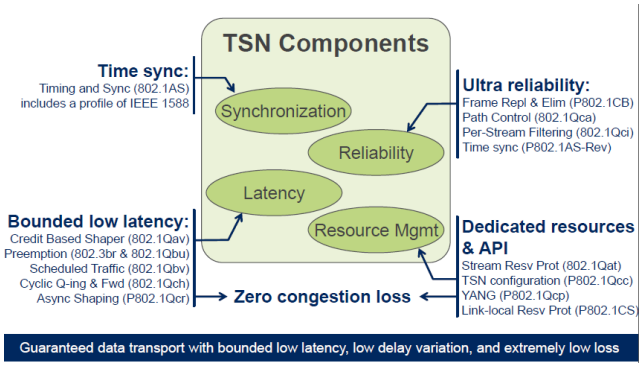
Network Traffic Rate Limiting Solution on Layerscape TSN Platform

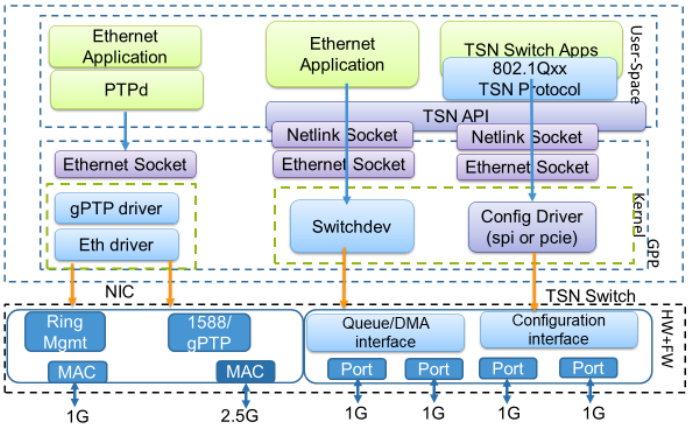
TSN(Time Sensitive Networking) is all about Layer 2 of the OSI model and is an extension to IEEE 802.1 to make Ethernet deterministic and more robust and reliable. The following describes TSN standards.



This document describes rate limiting solution implementation with QoS features of SJA1105 switch to handle the congestion of competing traffic flows, including the software architecture on TSN platforms, ingress traffic policer, prioritizing configuration, Time-Aware Scheduler in 802.1Qbv engine through SJA1105 switch and using IEEE 1588(Precision Time Protocol) to synchronize the SJA1105 PTP clocks.

1. Software Solution Architecture on TSN Platform

Please refer to the following figure for software architecture on TSN platforms.



2. Ingress Traffic Policer

The L2 Ingress Policer inside the SJA1105 is implemented as a Token Bucket:
Bucket max size (also known as burst size) is called SMAX (maximum is 0xFFFF).

Bucket refill speed is RATE bytes per second (up to a maximum of 64000).
Each ingress packet removes from the bucket a number of tokens equal to its length in bytes, can also police traffic based on maximum frame size.
The Policing table has 45 entries:
One for each Ingress Port x VLAN PRIO (5 x 8).
One for Broadcast Traffic coming from each Ingress Port (5).

L2 policing table_entry is described as the following in file config-modify.c.

```
const char *options[] = {
                "sharindx",
                "smax",
                "rate",
                "maxlen",
                "partition",
};
```

Please use the attached policer-limit script to generate the XML file in package sja1105-tool with the following command.

```
$ policer-limit --port 2 --prio 5 --mtu 1518 --rate-mbps 600
Configuration saved as ./src/helpers/bin/policer-limit.xml.
View with: "sja1105-tool config load ./src/helpers/bin/policer-limit.xml; sja1105-tool config show | less"
```

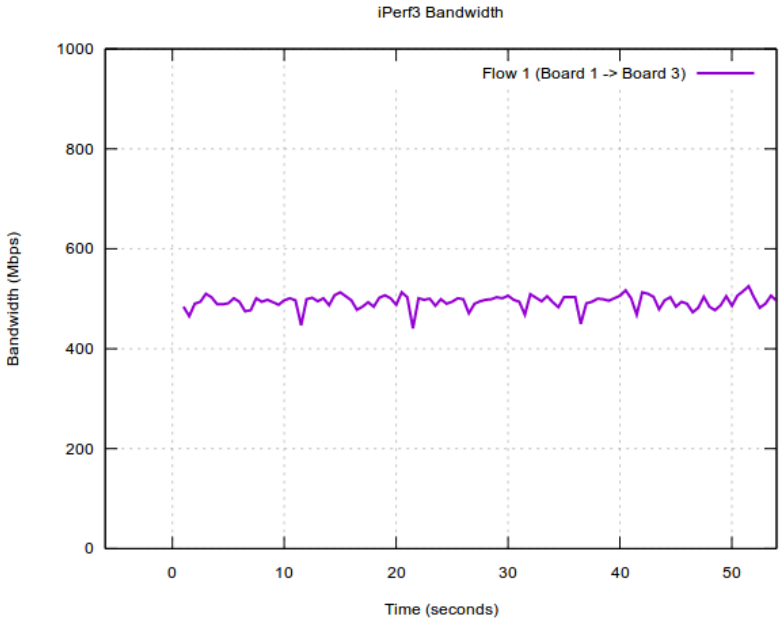
The generated XML file policer-limit.xml is similar as the following.

```
<l2-policing-table>
        <entry>
                <index>0</index>
                <sharindx>0x0</sharindx>
                <smax>0xFFFF</smax>
                <rate>0xFA00</rate>
                <maxlen>0x5EE</maxlen>
                <partition>0x0</partition>
        </entry>
          …
</l2-policing-table>
```

Load the specific configuration on the target board.

```
[root@openil] $ sja1105-tool config load policer-limit.xml
[root@openil] $ sja1105-tool config upload
```

The following is the test result with iperf3 on LS1021ATSN.



3. Prioritizing Configuration

Configure the SJA1105 switch of the testing board to assign VLAN priorities for the traffic, the flow from target board 1 is tagged with VLAN PCP 5, the flow from target board 2 is tagged with VLAN PCP 3. On the test board, the egress traffic following the strict priority queuing discipline of the switch.

L2 mac-configuration-table is described as the following in file config-modify.c.

```
const char *options[] = {
        "top",
        "base",
        "enabled",
        "ifg",
        "speed",
        "tp_delin",
        "tp_delout",
        "maxage",
        "vlanprio",
        "vlanid",
        "ing_mirr",
        "egr_mirr",
        "drpnona664",
        "drpdtag",
        "drpsotag", /* P/Q/R/S only */
        "drpsitag", /* P/Q/R/S only */
        "drpuntag",
        "retag",
        "dyn_learn",
```

```
        "egress",
        "ingress",
        "mirrcie", /* P/Q/R/S only */
        "mirrcetag", /* P/Q/R/S only */
        "ingmirrvid", /* P/Q/R/S only */
        "ingmirrpcp", /* P/Q/R/S only */
        "ingmirrdei", /* P/Q/R/S only */
    };
```

Please use the attached prioritizing.sh script to generate the XML file in package sja1105-tool with the following command.

$ prioritizing.sh --flow1-prio 5 --flow2-prio 3

Configuration saved as ./src/helpers/configs/rate-limiting/prioritizing.xml.

View with: "sja1105-tool config load ./src/helpers/configs/rate-limiting/prioritizing.xml; sja1105-tool config show | less"

The generated XML file prioritizing.xml is similar as the following.

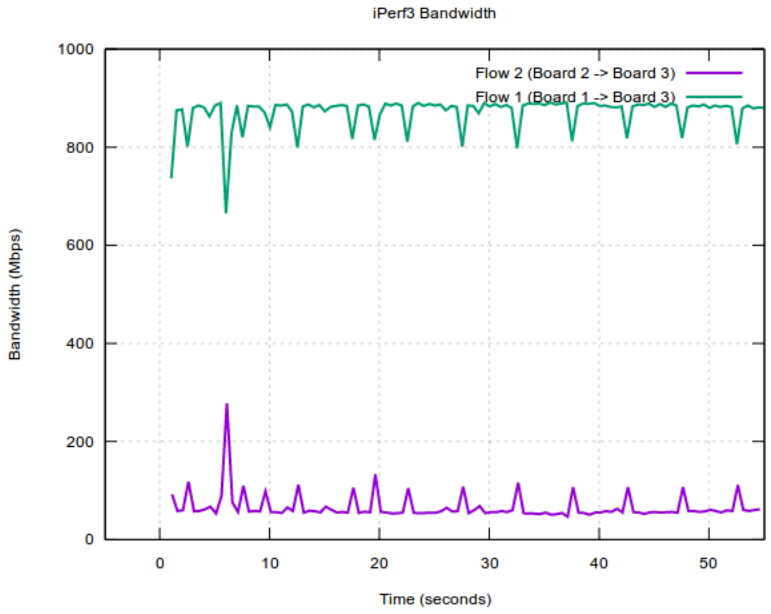
```
<mac-configuration-table>
…
  <entry>
        <index>2</index>
        <top>[0x3F 0x7F 0xBF 0xFF 0x13F 0x17F 0x1BF 0x1FF ]</top>
        <base>[0x0 0x40 0x80 0xC0 0x100 0x140 0x180 0x1C0 ]</base>
              <enabled>[0x1 0x1 0x1 0x1 0x1 0x1 0x1 0x1 ]</enabled>
              <ifg>0x0</ifg>
              <speed>0x1</speed>
              <tp_delin>0x0</tp_delin>
              <tp_delout>0x0</tp_delout>
              <maxage>0xFF</maxage>
              <vlanprio>0x5</vlanprio>
              <vlanid>0x0</vlanid>
              <ing_mirr>0x0</ing_mirr>
              <egr_mirr>0x0</egr_mirr>
              <drpnona664>0x0</drpnona664>
              <drpdtag>0x0</drpdtag>
              <drpsotag>0x0</drpsotag>
              <drpsitag>0x0</drpsitag>
              <drpuntag>0x0</drpuntag>
              <retag>0x0</retag>
              <dyn_learn>0x1</dyn_learn>
              <egress>0x1</egress>
              <ingress>0x1</ingress>
              <mirrcie>0x0</mirrcie>
              <mirrcetag>0x0</mirrcetag>
              <ingmirrvid>0x0</ingmirrvid>
              <ingmirrpcp>0x0</ingmirrpcp>
```

```
                    <ingmirrdei>0x0</ingmirrdei>
            </entry>
                <entry>
                        <index>4</index>
                        <top>[0x3F 0x7F 0xBF 0xFF 0x13F 0x17F 0x1BF 0x1FF ]</top>
                        <base>[0x0 0x40 0x80 0xC0 0x100 0x140 0x180 0x1C0 ]</base>
                        <enabled>[0x1 0x1 0x1 0x1 0x1 0x1 0x1 0x1 ]</enabled>
                            <ifg>0x0</ifg>
                            <speed>0x1</speed>
                            <tp_delin>0x0</tp_delin>
                            <tp_delout>0x0</tp_delout>
                            <maxage>0xFF</maxage>
                            <vlanprio>0x3</vlanprio>
                            <vlanid>0x0</vlanid>
                            <ing_mirr>0x0</ing_mirr>
                            <egr_mirr>0x0</egr_mirr>
                            <drpnona664>0x0</drpnona664>
                            <drpdtag>0x0</drpdtag>
                            <drpsotag>0x0</drpsotag>
                            <drpsitag>0x0</drpsitag>
                            <drpuntag>0x0</drpuntag>
                            <retag>0x0</retag>
                            <dyn_learn>0x1</dyn_learn>
                            <egress>0x1</egress>
                            <ingress>0x1</ingress>
                            <mirrcie>0x0</mirrcie>
                            <mirrcetag>0x0</mirrcetag>
                            <ingmirrvid>0x0</ingmirrvid>
                            <ingmirrpcp>0x0</ingmirrpcp>
                            <ingmirrdei>0x0</ingmirrdei>
                 </entry>
                …
        </mac-configuration-table>
```

Load the specific configuration on the target board.

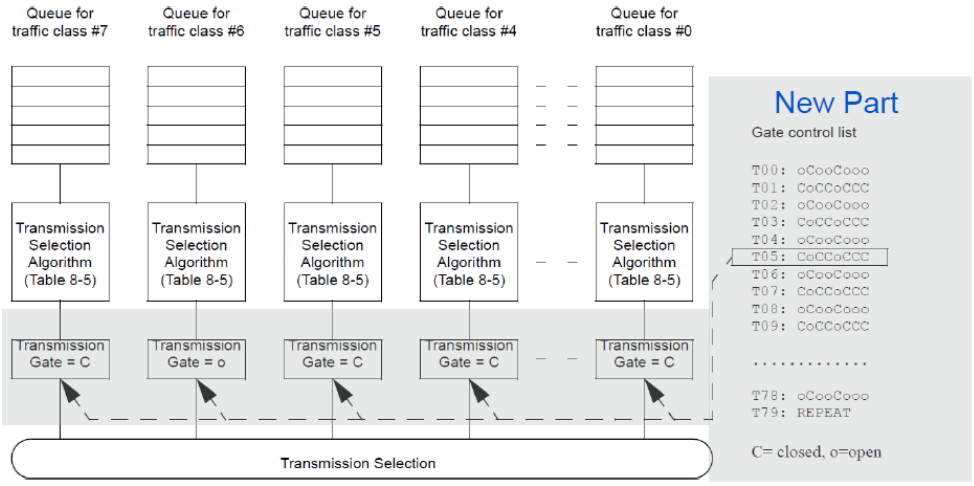
```
[root@openil] $ sja1105-tool config load prioritizing.xml
[root@openil] $ sja1105-tool config upload
```

The following is the priority test result with iperf3 on LS1021ATSN.



4. Time-Aware Scheduler

The Time-Aware Scheduler, defined in IEEE 802.1Qbv, associates "gates" with each of the 8 priority queues connected to the egress ports. A gate is said to be "open" if frames are allowed to be selected for transmission from that gate's associated queue, or "closed" otherwise. A cyclic schedule is kept, where multiple timeslots define what is the state of every one of the gates (open or closed), and for how long.



The Time-Aware Scheduler of the SJA1105 switch works by following the guidelines in 802.1Qbv:

Its 5 Ethernet ports each have 8 gates on egress, which can be open or closed.

Each gate controls its associated queue (there are 8 queues, one per traffic class priority).

Whenever a gate is open, packets from its respective queue can be sent out the wire.

The Time Aware Scheduler (or Qbv engine) functions based on a clock ticking with a period of 200ns.

Time slots can be created, where some gates can be opened (allow certain traffic classes) and some can be closed.

Each time slot's action applies to some specified egress ports. A time slot has a defined period of time for which it is active, and is chained together with other time slots in a periodic cycle.

The structure of the schedule entry table is defined as the following.

```
const char *options[] = {
        "winstindex",
        "winend",
        "winst",
        "destports",
        "setvalid",
        "txen",
        "resmedia_en",
        "resmedia",
        "vlindex",
        "delta",
};
gate_mask=$((0xff))
for gate in ${gates_open[@]+"${gates_open[@]}"}; do
        gate_mask=$((${gate_mask} & ~(1<<${gate})))
done
resmedia= gate_mask
delta=$(echo "(${duration_ms}*5000)/1" | bc)
$ scheduling.sh --flow1-prio 5 --flow2-prio 3 --flow1-time-ms 6 --flow2-time-ms 4
Configuration saved as ./src/helpers/configs/rate-limiting/scheduling.xml.
View with: "sja1105-tool config load ./src/helpers/configs/rate-limiting/scheduling.xml; sja1105-tool config show | less"
```

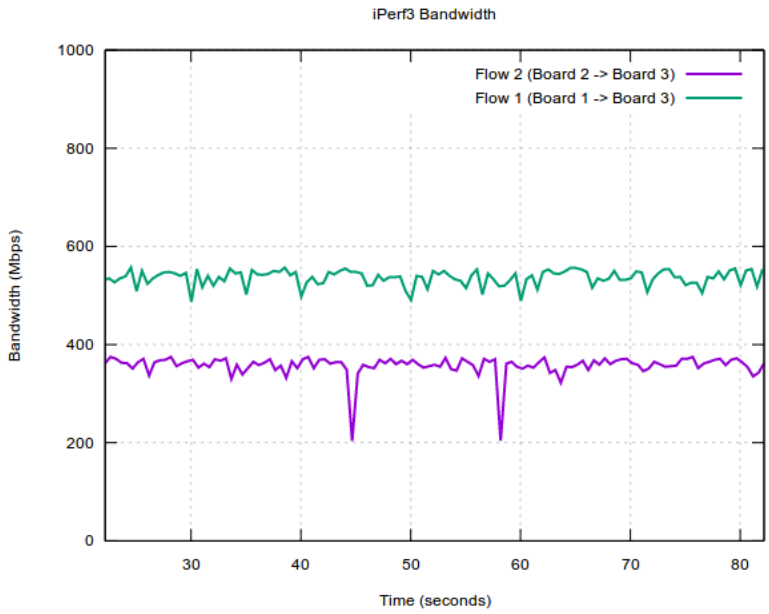
The generated XML file scheduling.xml is as the following.

```
<schedule-table>
    <entry>
        <index>0</index>
        <winstindex>0x0</winstindex>
        <winend>0x0</winend>
        <winst>0x0</winst>
        <destports>0x2</destports>
        <setvalid>0x0</setvalid>
        <txen>0x0</txen>
        <resmedia_en>0x1</resmedia_en>
        <resmedia>0xDF</resmedia>
        <vlindex>0x0</vlindex>
        <delta>0x7530</delta>
    </entry>
    <entry>
```

```
                <index>1</index>
                <winstindex>0x0</winstindex>
                <winend>0x0</winend>
                <winst>0x0</winst>
                <destports>0x2</destports>
                <setvalid>0x0</setvalid>
                <txen>0x0</txen>
                <resmedia_en>0x1</resmedia_en>
                <resmedia>0xF7</resmedia>
                <vlindex>0x0</vlindex>
                <delta>0x4E20</delta>
        </entry>
</schedule-table>
```

Load the specific configuration on the target board.
[root@openil] $ sja1105-tool config load scheduling.xml
[root@openil] $ sja1105-tool config upload

The following is the schedule test result with iperf3 on LS1021ATSN.



5. Using IEEE 1588 to Synchronize the SJA1105 PTP Clocks

If there are multiple Time-Aware Schedulers in the same L2 network, they need to have a common notion of time. By synchronizing the clocks of all Time-Aware Schedulers using IEEE 1588 (PTP), frames can be forwarded in a coordinated manner, similar to synchronized traffic lights.
In this application, synchronizing the PTP clocks of the switches with ptp4l daemon in Linux, configure the Qbv engine to use the PTP as the clock source.

SJA1105 PTP related commands set is defined as the following.

```
struct sja1105_ptp_cmd {
        uint64_t ptpstrtsch;    /* start schedule */
        uint64_t ptpstopsch;    /* stop schedule */
        uint64_t startptpcp;    /* start pin toggle   */
        uint64_t stopptpcp;     /* stop pin toggle */
        uint64_t cassync;       /* if cascaded master, trigger a toggle of the
                                   PTP_CLK pin, and store the timestamp of its
                                   1588 clock (ptpclk or ptptsclk, depending on
                                   corrclk4ts), in ptpsyncts.
                                   only for P/Q/R/S series */
        uint64_t resptp;        /* reset */
        uint64_t corrclk4ts;    /* if (1) timestamps are based on ptpclk,
                                   if (0) timestamps are based on ptptsclk */
        uint64_t ptpclksub;     /* only for P/Q/R/S series */
        uint64_t ptpclkadd;     /* enum sja1105_ptp_clk_add_mode */
};
```

SJA1105 PTP command commit function wraps around sja1105_spi_send_packed_buf.

```
sja1105_spi_send_packed_buf(spi_setup,
                            SPI_WRITE,
                            CORE_ADDR + ptp_control_addr,
                            packed_buf,
                            BUF_LEN);
```

Generate the XML configuration to configure qbv engine to use PTP clock.

```
<schedule-entry-points-parameters-table>
    <entry>
        <index>0</index>
        <clksrc>0x3</clksrc>
        <actsubsch>0x0</actsubsch>
    </entry>
</schedule-entry-points-parameters-table>
```

The clock source definition for sja1105-tool Qbv engine is as the following.

```
Disabled      clksrc="0b00"
Standalone    clksrc="0b01"
as6802        clksrc="0b10"
ptp           clksrc="0b11"
```

On the target board, load the XML configuration into the sja1105.

$ sja1105-tool config load -f qbv-ptp.xml

The XML configurations for SJA1105 use the PTP clock source for Qbv engine, it is needed to run the PTP synchronization daemon as the following.

$ ptp4l -i eth0 -p /dev/ptp0 -m -l 7 -t 1000

The PTP daemon will begin to keep in sync the SJA1105 PTP clock with the eTSEC clock of

the LS1021 eth0 port.

Synchronization offsets can be followed by examining the following output.

ptp4l[46335.657]: sja1105: offset 202 ns, delay 94627 ns