We currently use a fn-advance default of 20 frames, and a rts-advance of 5, resulting in a total of 25 frames (equalling 115ms) of downlink frame number advance.

This will cause

- significantly increased RTT for GPRS user plane data
- increase latency of RLC/MAC signaling, specifically
tbf establishment
  - potentially cause window stalls if we don't poll for ACK/NACK a lot sooner than our window filling up.
- probably mess with LAPDm timing

I would guess that on modern hardware, particularly with SCHED_RR on TRX + BTS, we can reduce the fn_advance drastically. The rts_advance likely needs to remain in place without too many changes, as this is the amount of time the PCU has to prepare downlink data (i.e. schedule DL).

As a second step, we could possibly even think of something like a dynamically sized fn-advance, similar to dynamic jitter buffers work in RTP.

**Related issues:**

Related to libosmocore - Bug #4074: LAPD timers completely broken

**Associated revisions**

Revision bb0c5368 - 07/10/2020 03:32 PM - pespin

Transceiver: Fix race condition obtaining DI burst from Upper layer

The queue was being accessed sequentially obtaining and releasing the mutual exclusion zone twice. First in getStaleBurst() dropping all FN=currTime, then in getCurrentBurst() trying to obtain FN=currTime.

However, since in between the mutex is released, it could happen that for instance upper layer would introduce currTime-1 in the queue, which would make then getCurrentBurst() detect that one instead of potential currTime in the queue and return NULL.

By holding the mutex during the call to both functions we make sure the state is kept during the whole transaction.

Related: OS#4487 (comment #7)
Change-Id: ff1fd8d7c5f21ee2894192ef1ac2a3cdda6bbb98

Revision 1d0c6fe7 - 07/10/2020 03:32 PM - pespin

Add rate counter for missing Txbursts when scheduled towards the radioInterface

Related: OS#4487
Change-Id: lbb2c492b3c67cbab11fbb936ae3a090b5756aa8

Revision 624b5cdc - 08/17/2020 05:11 PM - daniel

osmo-bts-trx: Use much lower clock advance values towards PCU and TRX

osmotrx fn-advance (which is the clock_advance variable here) and osmotrx rts-advance together make up the minimum delay the BTS can react
to a channel request, etc.

The default of 20 are around 92ms which is clearly too much. With modern hardware and using SCHED_RR a lower value should not be an issue.

See OS#4487 for some related measurements on more CPU-limited devices like a LimeNet-micro3.

Fixes: OS#4487
Fixes: SYS#4885
Related: SYS#4881
Change-Id: I7da3d0948f38e12342fb714b29f8edc5e9d0933d

Revision e91bbfc3 - 12/14/2021 12:30 AM - fixeria
bts/osmo-bts.cfg: use default {fn,rts}-advance values

It's better if we run ttcn3-bts-test with the default values, given that they were significantly reduced some time ago.

Change-Id: ld97e848e5df0b6c504d06f62642511cf5df066
Related: I7da3d0948f38e12342fb714b29f8edc5e9d0933d (osmo-bts)
Related: OS#4487

Revision 7fc251e5 - 12/14/2021 01:01 PM - fixeria
ttcn3-bts-test/osmo-bts.cfg: use default {fn,rts}-advance values

It's better if we run ttcn3-bts-test with the default values, given that they were significantly reduced some time ago.

Change-Id: If8438adf06d2b6b7858ea89a659ba246a1
Related: I7da3d0948f38e12342fb714b29f8edc5e9d0933d (osmo-bts)
Related: OS#4487

History

#1 - 04/08/2020 04:47 PM - daniel
- File Screenshot_20200408-183109.png added
- File Screenshot_20200408-183218.png added
- Status changed from New to In Progress

So far on my laptop I reduced fn-advance to 3 and pings look a lot better.

#2 - 04/09/2020 06:21 PM - daniel

https://gerrit.osmocom.org/c/osmo-bts/+/17766

Please test with those or even lower values and report back what still works.

#3 - 05/12/2020 12:01 PM - laforge
- Assignee changed from daniel to pespin

pespin, please take over

#4 - 06/16/2020 05:11 PM - pespin
- Status changed from In Progress to Feedback
- % Done changed from 0 to 80

I updated the gerrit patch and put some updated comments in there.

So in summary:

- I tested with B200 + osmo-trx-uhd + multi-arfcn with 2 TRX
- I tested with LimeSDR-USB + osmo-trx-lms + 1 TRX
- I had to run osmo-pcu also with SCHED_RR (-r 1) to avoid having issues with PDTCH DI blocks not enqueued quickly enough in BTS (related to rts-advance value)
- I also noticed that using a more conservative logging levels (I was using a quite verbose and compute intensive one for RLCMAC category) also helps in getting more stable.

03/09/2022
- "fn-advance" can be decreased to 2 by default, it worked fine. "rts-advance is on the edge already, so I wouldn't touch that one.

I also submitted patches improving some related scheduler code to provide more information. I also added rate counters in order to display issues related to fn-advance and rts-advance ("show rate-counters" in osmo-bts).

---

**#5 - 06/29/2020 03:14 PM - pespin**

I did some testing with a LimeNET-micro and so far it looks good from osmo-bts-trx side, but it's not working properly on osmo-trx-lms side due to Tx downlink bursts arriving too late when using fn-advance 2 or 3. I get lots of messages like this from time to time:

DTRXDDL <0003> Transceiver.cpp:430 [tid=1404242023869184][chan=0] dumping STALE burst in TRX->SDR interface (0:2005343 vs 1:2005343), retrans=0

I'm running all through systemd services and they have realtime scheduling set in the service files.

I added some rate counters to monitor that kind of issue in osmo-trx, and provide also some VTY command to establish a threshold at which osmo-trx will exit to flag the BTS that something's wrong, like we do for other counters (overruns, underruns, dropped packets, etc.):

remote:  [https://gerrit.osmocom.org/c/osmo-trx/+/19050](https://gerrit.osmocom.org/c/osmo-trx/+/19050) Rename device specific rate counter multi-thread helpers
remote:  [https://gerrit.osmocom.org/c/osmo-trx/+/19051](https://gerrit.osmocom.org/c/osmo-trx/+/19051) Introduce rate counter tx_stale_bursts

While at it, I also fixed some bug in the rate counter thresholds I observed.

---

**#6 - 07/08/2020 04:55 PM - pespin**

Using current default fn values (20 and so), I have been running osmo-bts-trx-osmo-trx-lms in LimeNet-Micro3 for a few hours with 1 phone attached and pinging some IP addr. Then check over time the related rate counters:

in osmo-bts-trx:

```
trx_clk:sched_dl_miss_fn: 0 (0/s 0/m 0/h 0/d) Downlink frames scheduled later than expected due to missed timerfd event (due to high system load)
```

This one didn't change over time, which is good.

But then in osmo-trx-lms:

```
trx:tx_stale_bursts: 232 (0/s 22/m 232/h 0/d) Number of Tx burts dropped by TRX due to arriving too late
trx:tx_stale_bursts: 1849 (0/s 0/m 1849/h 1793/d) Number of Tx burts dropped by TRX due to arriving too late
trx:tx_stale_bursts: 5067 (0/s 0/m 1890/h 5031/d) Number of Tx burts dropped by TRX due to arriving too late
trx:tx_stale_bursts: 5201 (0/s 0/m 2024/h 5031/d) Number of Tx burts dropped by TRX due to arriving too late
trx:tx_stale_bursts: 6156 (0/s 0/m 1125/h 5998/d) Number of Tx burts dropped by TRX due to arriving too late
```

So we are dropping around 2k bursts per hour aprox, with current settings. I still need to figure out what fn param relates to that.

---

**#7 - 07/08/2020 09:38 PM - pespin**

After a few more hours with the test running with same environment, I continue to have trx_clk:sched_dl_miss_fn at 0 and trx:tx_stale_bursts at around 2425/h

Regarding that counter, I just found out that there may be a race condition between getStaleBurst() and getCurrentBurst(), where a burst is fed in the queue in between and then getCurrentBurst() fails because the required one may be not the first one even if it was queued and getStaleBurst() will later drop it. So we potentially need to refactor that code to avoid those issues., and repeat the test (adding an extra counter for the case where getCurrentBurst fails, because then by comparing with the other one it can be known if bursts arrived late or never arrived).

If that's not introducing an issue, then we need to investigate whether lowering fn_advance further degrades the current situation or not.

---

**#8 - 07/09/2020 04:28 PM - pespin**

I submitted a bunch of more patches fixing the potential race condition as well as adding more counters useful to gasp timing issues:

remote:  [https://gerrit.osmocom.org/c/osmo-trx/+/19206](https://gerrit.osmocom.org/c/osmo-trx/+/19206) Transceiver: Fix race condition obtaining DI burst from Upper layer
remote:  [https://gerrit.osmocom.org/c/osmo-trx/+/19207](https://gerrit.osmocom.org/c/osmo-trx/+/19207) Add rate counter for missing Tx bursts when scheduled towards the radiointerface
remote:  [https://gerrit.osmocom.org/c/osmo-trx/+/19205](https://gerrit.osmocom.org/c/osmo-trx/+/19205) Introduce rate counters to detect issues in received DI bursts from TRXD
A few days ago I implemented the transmit queue in fake_trx.py, so now it behaves just like a normal transceiver and reflects the negative impact of high fn-advance values, see https://osmocom.org/issues/4658#note-8. TL;DR several test cases from ttcn3-bts-test started to fail with queuing enabled. I also believe, that high fn-advance value could most likely be the reason of permanent BTS_Tests.TC_rsl_modify_encr failures. The test case expects a new encryption key to be applied immediately, while on practice it’s applied 20 TDMA frames later (5 xCCH frames).

#10 - 07/22/2020 03:37 PM - pespin

After a few hours running same test I did a few days ago (limenet-micro, 1 MS pinging for a few hours, default fn-advance 20):

OsmoTRX# show rate-counters
osmo-trx statistics 0:
  device:rx_overruns:  0 (0/s 0/m 0/h 0/d) Number of Rx overruns in FIFO queue
  device:tx_underruns:  0 (0/s 0/m 0/h 0/d) Number of Tx underruns in FIFO queue
  device:tx_drop_events:  0 (0/s 0/m 0/h 0/d) Number of times Tx samples were dropped by HW
  device:tx_drop_samples:  0 (0/s 0/m 0/h 0/d) Number of Tx samples dropped by HW
  device:tx_drop_events:  3 (0/s 0/m 0/h 3/d) Number of times Tx samples were dropped by HW
  device:tx_drop_samples:  3 (0/s 0/m 0/h 3/d) Number of Tx samples dropped by HW
  trx:tx_stale_bursts:  13274 (0/s 0/m 7556/h 10075/d) Number of Tx bursts dropped by TRX due to arriving too late
  trx:tx_unavailable_bursts:  13599 (0/s 0/m 7660/h 10377/d) Number of Tx bursts unavailable (not enqueued) at the time they should be transmitted
  trx:tx_trxd_fn_repeated:  0 (0/s 0/m 0/h 0/d) Number of Tx bursts received from TRXD with repeated FN
  trx:tx_trxd_fn_outoforder:  0 (0/s 0/m 0/h 0/d) Number of Tx bursts received from TRXD with a past FN
  trx:tx_trxd_fn_skipped:  141 (0/s 0/m 104/h 118/d) Number of Tx bursts potentially skipped due to FN jumps

For reference, after few startup seconds, all were 0 and trx:tx_unavailable_bursts=184 (expected during startup to have some until it stabilizes).

And osmo-bts-trx:

OsmoBTS# show rate-counters
L1 scheduler timeslot 17:
  l1sched_tsdl late:  3 (0/s 0/m 0/h 3/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  4 (0/s 0/m 0/h 4/d) Downlink frames not found while scheduling
L1 scheduler timeslot 16:
  l1sched_tsdl late:  3 (0/s 0/m 0/h 3/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  4 (0/s 0/m 0/h 4/d) Downlink frames not found while scheduling
L1 scheduler timeslot 15:
  l1sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 14:
  l1sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 13:
  l1sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 12:
  l1sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 11:
  l1sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 10:
  l1sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 9:
  l1sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 8:
  l1sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 7:
  l1 sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1 sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 6:
  l1 sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1 sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 5:
  l1 sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1 sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 4:
  l1 sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1 sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 3:
  l1 sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1 sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 2:
  l1 sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1 sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling
L1 scheduler timeslot 1:
  l1 sched_tsdl late:  0 (0/s 0/m 0/h 0/d) Downlink frames arrived too late to submit to lower layers
  l1 sched_tsdl_not_found:  0 (0/s 0/m 0/h 0/d) Downlink frames not found while scheduling

03/09/2022
So from those numbers, my opinion is that on the BTS side, everything looks quite good. I think those 3-4 bursts lost during around 4 hours on TS0 and TS1 we can ignore.

On the TRX side, it looks much worse. More or less tx_unavailable_bursts - 184 (startup drop) - tx_trxd_fn_skipped = tx_stale_bursts. So 99% of the issues are basically bursts arriving too late to the Lower Layer TX thread pushing stuff into the SDR. My observation is that these usually happen in big bursts of hundreds of thousands of FNs at the same time (the counter increments a lot in a few seconds).

Finally, we get 144 skipped FN, which cross-checking against BTS, means either
1- The Kernel dropped UDP packets between BTS->TRX
2- There is some bug in BTS sometimes not sending DL packets and not being counted in the counter.

I switched to fn-dvance 6 on the same setup, I get similar (dropped/too-late) values to the ones I got with fn-advance 20. I have the feeling is going to be in timing/locking between threads in osmo-trx rather than between fn-advance between osmo-trx and osmo-bts-trx. For instance one candidate may be the radioClock Time being locked/unlocked several times with different threads operating on its values and creating some race conditions.

That's only a blind guess so far.

I also created ticket #4679 to get rid of some bursty log line about rach clipping which may also help getting rid of some unstabilities. So my opinion right now on the ticket's topic is: I need to do more testing, but looks like we'll be able to decrease fn-advance since the situation is not worse than with a higher value.

BTW, I forgot to say that while running the long test, I see from time to time this from the kernel on journalctl, which may also cause unstabilities:

```
    Jul 24 17:44:01 limenet-micro3 rngd[381]: stats: bits received from HRNG source: 120064
    Jul 24 17:44:01 limenet-micro3 rngd[381]: stats: bits sent to kernel pool: 64096
    Jul 24 17:44:01 limenet-micro3 rngd[381]: stats: entropy added to kernel pool: 64096
```
I did some more tests with LimeNet-micro this time applying ultra-low values proposed by Daniel in https://gerrit.osmocom.org/c/osmo-bts/+/17766, and I get similar results to those with current values (high timing ones), so to me it’s a green light towards going ahead setting the new lower default values. It also confirms that the issue with dropped DL bursts is within osmo-trx threads themselves, and should be investigated separately.

OsmoTRX# show rate-counters

<table>
<thead>
<tr>
<th>device</th>
<th>rx_overruns: 0 (0/s 0/m 0/h 0/d)</th>
<th>tx_underruns: 0 (0/s 0/m 0/h 0/d)</th>
<th>rx_drop_events: 0 (0/s 0/m 0/h 0/d)</th>
<th>tx_drop_samples: 0 (0/s 0/m 0/h 0/d)</th>
<th>tx_drop_events: 3 (0/s 0/m 3/h 3/d)</th>
<th>tx_drop_samples: 3 (0/s 0/m 3/h 3/d)</th>
<th>tx_stale_bursts: 10054 (0/s 0/m 10054/h 8489/d)</th>
<th>tx_unavailable_bursts: 10094 (0/s 0/m 10094/h 8529/d)</th>
<th>tx_trxd_fn_repeated: 0 (0/s 0/m 0/h 0/d)</th>
<th>tx_trxd_fn_outoforder: 0 (0/s 0/m 0/h 0/d)</th>
<th>tx_trxd_fn_skipped: 0 (0/s 0/m 0/h 0/d)</th>
<th>rx_empty_burst: 0 (0/s 0/m 0/h 0/d)</th>
<th>rx_clipping: 40086 (0/s 3944/m 39599/h 13817/d)</th>
<th>rx_no_burst_detected: 0 (0/s 0/m 0/h 0/d)</th>
</tr>
</thead>
</table>

(trhe high clipping here is unrelated and happens because I have the MS at 30cm from the LimeNetMicro pinging some internet host).

#13 - 08/17/2020 05:05 PM - pespin

I did some more tests with LimeNet-micro this time applying ultra-low values proposed by Daniel in https://gerrit.osmocom.org/c/osmo-bts/+/17766, and I get similar results to those with current values (high timing ones), so to me it’s a green light towards going ahead setting the new lower default values. It also confirms that the issue with dropped DL bursts is within osmo-trx threads themselves, and should be investigated separately.

OsmoTRX# show rate-counters

| device          | rx_overruns: 0 (0/s 0/m 0/h 0/d) | tx_underruns: 0 (0/s 0/m 0/h 0/d) | rx_drop_events: 0 (0/s 0/m 0/h 0/d) | tx_drop_samples: 0 (0/s 0/m 0/h 0/d) | tx_drop_events: 3 (0/s 0/m 3/h 3/d) | tx_stop_events: 3 (0/s 0/m 3/h 3/d) | tx_trxd_fn_repeated: 0 (0/s 0/m 0/h 0/d) | tx_trxd_fn_outoforder: 0 (0/s 0/m 0/h 0/d) | tx_trxd_fn_skipped: 0 (0/s 0/m 0/h 0/d) | rx_empty_burst: 0 (0/s 0/m 0/h 0/d) | rx_clipping: 40086 (0/s 3944/m 39599/h 13817/d) | rx_no_burst_detected: 0 (0/s 0/m 0/h 0/d) |
|----------------|----------------------------------|-----------------------------------|------------------------------------|-------------------------------------|-------------------------------------|-------------------------------------|--------------------------------------------|---------------------------------------------|---------------------------------------------|---------------------------------------------|---------------------------------------------|---------------------------------------------|---------------------------------------------|

(trhe high clipping here is unrelated and happens because I have the MS at 30cm from the LimeNetMicro pinging some internet host).

#14 - 08/18/2020 08:26 AM - pespin

- Status changed from Feedback to Resolved
- % Done changed from 80 to 100

https://gerrit.osmocom.org/c/osmo-bts/+/17766 was merged, closing.

#15 - 09/17/2020 03:50 PM - laforge

- Related to Bug #4074: LAPD timers completely broken added

Files

| Screenshot_20200408-183109.png | 196 KB | 04/08/2020 | daniel |
| Screenshot_20200408-183218.png | 276 KB | 04/08/2020 | daniel |