OsmoBTS - Feature #2977

OsmoBTS measurment processing at L1SAP too complex / pass measurements along with data

02/21/2018 11:22 PM - laforge

Status:	In Progress	**Start date:**	02/21/2018
Priority:	Normal	**Due date:**	
Assignee:	dexter	**% Done:**	70%
Category:			
Target version:			
Spec Reference:			

Description

The entire dualism of PH_DATA.ind / PH_TCH.ind containg (unsued) measurement data, but then having a separate PRIM_INFO_MEAS is odd to begin with. The measurements should always accompany the PH-DATA.ind / PH-TCH.ind and PRIM_INFO_MEAS should be abandoned.

Related issues:

Related to OsmoBTS - Bug #3032: (Wrong?) measurement of both RSSI and ToA on ...	**New**	**03/05/2018**
Related to OsmoBTS - Bug #2975: OsmoBTS doesn't generate measurement indicati...	**Stalled**	**02/21/2018**
Related to OsmoBTS - Feature #3530: merge PRIM_INFO_MEAS into PRIM_PH_DATA an...	**Rejected**	**09/06/2018**
Related to OsmoBTS - Feature #3428: Implement handling of NOPE / IDLE indicat...	**Stalled**	**07/28/2018**
Related to OsmoPCU - Bug #3395: Uplink CS/MCS control is broken osmo-pcu is u...	**Feedback**	**07/14/2018**
Related to OsmoPCU - Feature #1526: Acquire/update timing advance (TA)	**Stalled**	**02/22/2016**
Related to OsmoBTS - Bug #3803: fix frame number calculation in scheduler_trx	**In Progress**	**02/15/2019**

History

#1 - 03/05/2018 08:43 AM - laforge

- Related to Bug #3032: (Wrong?) measurement of both RSSI and ToA on TCH channels added

#2 - 03/05/2018 08:43 AM - laforge

Ideally, we should attach all measurement information in the best resolution (ber10k, toa256, rssi) to the PH-DATA.ind and remove the somwhat strange INFO-IND that we use separately at the moment. After all, we have the following cases:

- measurements only, no valid data: Use PH-DATA.ind with zero-length payload (bad frame)
- measurements + data: Use fully filled-in PH-DATA.ind
- data without measurements: doesn't exist, doesn't make sense to support (but theoretically possible now).

#3 - 03/05/2018 03:06 PM - fixeria

- Assignee set to fixeria

#4 - 09/25/2018 07:56 PM - fixeria

- Tracker changed from Bug to Feature

- Assignee deleted (fixeria)

#5 - 09/25/2018 07:57 PM - fixeria

- Related to Bug #2975: OsmoBTS doesn't generate measurement indications in absence of uplink bursts added

#6 - 09/25/2018 08:19 PM - fixeria

- Related to Feature #3530: merge PRIM_INFO_MEAS into PRIM_PH_DATA and PRIM_TCH added

#7 - 09/30/2018 11:02 AM - laforge

- Subject changed from OsmoBTS measurment processing at L1SAP too complex to OsmoBTS measurment processing at L1SAP too complex / pass measurements along with data

- Assignee set to dexter

#8 - 10/01/2018 07:56 AM - dexter

This looks like a duplicate of #3530, probably we should reject #3530 then.

#9 - 10/25/2018 09:41 AM - pespin

- Related to Feature #3428: Implement handling of NOPE / IDLE indications from Transceiver added

#10 - 01/21/2019 03:17 PM - dexter

- File notes.txt added

#11 - 01/21/2019 03:18 PM - dexter

- File deleted (notes.txt)

#12 - 01/21/2019 03:25 PM - dexter

- File notes.txt added

I had another look into the code to pinpoint the locations where a change is needed. We will have to touch l1sap.h in libosmocore but I think when we just append struct members to struct ph_data_param and struct ph_tch_param we should be ok.

Attached one finds my notes. To me the process of changing the flow of the measurement results seems to be straight forward, but in some cases the old code did send a measurement indication up, but skipped the related data part because it has checked the received data bad. When we put both together we can either choose to skip the measurement as well or not to skip it but hand up empty data/tch into the common code. The latter one looks more correct to me. Its fine if we get empty tch/data, if required we can skip it anyway.

#13 - 01/23/2019 12:56 PM - dexter

- File notes.txt added

I now further inspected the code. Its possible to merge meas_ind and data/tch, but before we can do that we need to resolve a few open questions. Especially the fn and the toa256 values are computed differently for meas_ind and tch/data. This is a bit confusing. I am not sure if it is the right way to add separate meas_ members to the tch/data struct. I would prefer not to add redundancy to the the tch/data strucht.

I have attached the current state of my notes and I have marked all open questions with a (!).

#14 - 01/26/2019 12:44 PM - laforge

adding tnt as a watcher here as he's working on #1851 which is somewhat related.

#15 - 01/26/2019 01:26 PM - laforge

Responding / quoting here would have been much easier if the "notes.txt" was simply copy+pasted. Please do that in the future, thanks!

> scheduler_trx.c:tx_data_fn()
> -> l1_if.c->l1if_process_meas_res()->l1sap_up() (sends meas_ind)
> common/scheduler.c:_sched_compose_ph_data_ind()->l1sap_up() (sends data)
> (directly one after another in the middle)
>
> (!) Why do we use hardcoded values here?

Please see the comment "handle loss detection of SACCH". So basically we're sending substitute measurements in case there were lost frames. I'm not sure if this should still be done. I guess you are the person understanding the requirements of the higher-level code best in terms of whether and where we must substitute something. The hardcoded values indicate 100% BER and -110 dBm receive level, which are the "worst case" which we must assume in case a block was lost.

> (!) Why do we use hardcoded frame number for data (0) and the real fn for meas?

I don't know.

> scheduler_trx.c:rx_data_fn()
> -> l1_if.c->l1if_process_meas_res()->l1sap_up() (sends meas_ind)
> common/scheduler.c:_sched_compose_ph_data_ind()->l1sap_up() (sends data)
> (directly one after another at the end)
>
> (!) 4 * (*toa256_sum) / *toa_num != *toa256_sum / *toa_num -- Why that?

it's computing the toa256 average and then multiplying it to four, which may be a remainder of having previously used quarter-bits in the ph_data_ind. This seems wrong, as the function _sched_compose_ph_data_ind() has the argument "ta_offs_256bits". The factor-of-four is wrong here, IMHO.

Commit acefd0586e5d463b2e7a6a039131994bc12573fc introduced the "toa256" resolution change. Before the change, _sched_compose_ph_data_ind() used quarter-bits as units, now it does 256th bits. l1if_fill_meas_res() was changed, but somehow the other callers weren't changed. This means that the measurement reports are all wrong now in terms of TOA :(

> (!) Cosmetic: link_qual_cb should be NULL instead of 0

ACK.

> scheduler_trx.c:rx_pdtch_fn()
> -> l1_if.c->l1if_process_meas_res()->l1sap_up() (sends meas_ind)
> (may return 0 when PDTCH is bad)
> common/scheduler.c:_sched_compose_ph_data_ind()->l1sap_up() (sends data)
> (in case of bad PDTCH we will not get any measurement info anymore
> or we allow to hand up bad pdtch data --> problem?)
>
> (!) 4 * (*toa256_sum) / *toa_num != *toa256_sum / *toa_num -- Why that?

Same bug as above.

> (!) fn != (fn + GSM_HYPERFRAME - 3) % GSM_HYPERFRAME

I guess this is some poor man's approach at converting a "last burst of block" frame number to "first burst of block" ? Might be a bug or some really odd difference how frame numbers are treated on he PDTCH/PCU side? But as it's only done inside osmo-bts-trx and not osmo-bts-{sysmo,...} I'd guess it's probably rather a bug? Maybe it doesn't matter as the frame number isn't used for anythign important? Needs more investingation!

> scheduler_trx.c:rx_tchf_fn()
> -> l1_if.c->l1if_process_meas_res()->l1sap_up() (sends meas_ind)
> (complex decision logic in between, decides between TCH and FACCH)
> common/scheduler.c:_sched_compose_ph_data_ind()->l1sap_up() (sends facch)
> common/scheduler.c:_sched_compose_tch_ind()->l1sap_up() (sends tch, located at the end)
>
> (!) FACCH: tn != (fn + GSM_HYPERFRAME - 7) % GSM_HYPERFRAME ?
> (!) TCH: fn != (fn + GSM_HYPERFRAME - 7) % GSM_HYPERFRAME

It's probably again converting the frame number of the last burst of the block to the frame number of the first burst of the block. As TCH/F are spread over 8 bursts, it seems correct?

> (!) FACCH: toa256 != 4 * toa256

Same bug as above.

> scheduler_trx.c:rx_tchh_fn()
>
> (!) FACCH: *first_fn != fn + GSM_HYPERFRAME - 10 - ((fn % 26) >= 19)) % GSM_HYPERFRAME?
> (!) TCH: *first_fn != fn + GSM_HYPERFRAME - 10 - ((fn%26)==19) - ((fn%26)==20)) % GSM_HYPERFRAME

Again, likely the conversion from last-fn to first-fn, which is a bit more complicated in case of TCH/H

> (!) FACCH: toa256 != toa256/64

Same bug as above.

#16 - 01/26/2019 01:33 PM - laforge

> This is measurement related members are already present in data or tch indication:
>
> - int8_t rssi; (same as inv_rssi?)
> - uint32_t fn; /*!< GSM Frame Number */
> - uint16_t ber10k; /*!< BER in units of 0.01% */

inv_rssi is the inverse of RSSI. It's the same value just expressed differently.

(!) How do we handle the is_sub field? octphy does not set it.
==> who sets the flag? l1sap.c seems to read it only, but nobody seems
to set it!

The "is_sub" flag expresses if a given measurement/block is part of the RX*SUB (true), or only part of the RX*FULL measurements. This is important in the context of DTX. For HR/FR/EFR it's rather easy. I think it was even you who implemented the ts45008_83_is_sub() function which computes this flag. It is set in common/measurement.c:

```
        /* We expect the lower layers to mark AMR SID_UPDATE frames already as such.
         * In this function, we only deal with the comon logic as per the TS 45.008 tables */
        if (!ulm->is_sub)
                ulm->is_sub = ts45008_83_is_sub(lchan, fn, false);
```

So basically for the AMR SID UPDATE blocks the lower layer is responsible to mark them, for all other blocks the common layer takes care of it.

#17 - 01/26/2019 04:05 PM - laforge

- Status changed from New to In Progress

- % Done changed from 0 to 10

#18 - 01/26/2019 04:06 PM - laforge

I just pushed https://gerrit.osmocom.org/#/c/osmo-bts/+/12699 as an attempt to fix the "toa *4" error in osmo-bts-trx. The patch is not tested yet.

#19 - 01/26/2019 06:06 PM - fixeria

Just my two cents:

PDTCH: (!) fn != (fn + GSM_HYPERFRAME - 3) % GSM_HYPERFRAME
FACCH/F: (!) fn != (fn + GSM_HYPERFRAME - 7) % GSM_HYPERFRAME
TCH/F: fn != (fn + GSM_HYPERFRAME - 7) % GSM_HYPERFRAME

This is not only poor, but also a wrong approach. There is no warranty that
(fn + GSM_HYPERFRAME - X) % GSM_HYPERFRAME would always point to the first
burst (bid == 0) of a block. Almost every multiframe layout has IDLE frames,
TCH multiframes also do contain SACCH frames, while TCH/H is a pure nightmare.

A proper approach to calculate the first frame number using the last is
to introduce a few functions, which would rely on the existing multiframe
layouts. This is how it's done in trxcon for TCH/H.

#20 - 01/28/2019 09:25 AM - msuraev

- Related to Bug #3395: Uplink CS/MCS control is broken osmo-pcu is used with osmo-bts-trx/osmo-trx added

#21 - 01/28/2019 09:26 AM - msuraev

- Related to Feature #1526: Acquire/update timing advance (TA) added

#22 - 01/28/2019 04:07 PM - dexter

#23 - 02/01/2019 05:39 PM - dexter

- % Done changed from 10 to 20

I have had a closer look to those wired fn calculations ((fn + GSM_HYPERFRAME - x) % GSM_HYPERFRAME). To me it sounds logical that the intend behind those formulas is to calculate the fn that belongs to the beginning of the block. The measurement reports are also connected to the beginning of the block. However, the measurement indications use a pointer *first_fn, which always stores the fn of the beginning of the block but the value there is not calculated, it is just stored when the first burst of the block arrives. This is much simpler and I have exchanged the formulas with the *first_fn pointers. Then it is also coherent with the measurement indications.

A patch can be found here:
https://gerrit.osmocom.org/#/c/osmo-bts/+/12779 scheduler_trx: use stored block fn instead of calculating it.

To make sure this change does not mess up anything I have checked through all possible PDCH locations and all possible TCH/F and TCH/H locations. I noticed no malfunctions or any other suspicious behavior. I also did a testrun with ttcn3 and docker. This also showed no problems.

====

Also for the sysmo-bts I have the same experiment running as I already have for osmo-bts-trx. The measurement indication is no attached to tch and data indications. However, this still needs some cleanups. However, once the problems with toa256 and the fn are resolved things should fall in place.

#24 - 02/15/2019 04:47 PM - dexter

- Related to Bug #3803: fix frame number calculation in scheduler_trx added

#25 - 02/15/2019 04:48 PM - dexter

I think before we can move on here we need to fix the formulas in scheduler_trx.c. This is a dependency, before tch indication and measurement indication do not use coherent frame numbers we can not merge the primitives. See also #3803

#26 - 02/15/2019 04:48 PM - dexter

- Status changed from In Progress to Stalled

#27 - 10/29/2019 08:39 AM - dexter

- Status changed from Stalled to In Progress

Unfortunately this seems to be a duplicate of #3530. I will keep this ticket and close #3530

I have now investigated the relevant places where changes must be made. I also made sure that scheduler_trx.c now adds the values that are relevant for measuremet. I have commented out the sections that send the info indication up to the higher layers and made a test to confirm that no info indication with measurement data is arriving at higher layers anymore. The next step is to make use of the measurement data in the TCH/DATA indications and to confirm that measurement reports are working as before.

For the various phy based BTSs things are a little bit difficult. I have already checked how things should be changed for osmo-bts-sysmo. For all other phy based BTSs its basically the same, but we have to be very careful not to break anything here as we probably can not test each model manually. Especially osmo-bts-octphy has to be done blindly. However, I am confident that that the phy-based BTSs will not pose much of a problem. Osmo-bts-trx is by far the most complicated candidate.

#28 - 10/31/2019 01:09 PM - dexter

- % Done changed from 20 to 70

The changes are now done for osmo-bts-trx, but not for the phy based bts models. However, l1sap.c can handle both ways, so in theory we could update the phy based bts models whenever there is time for that. Legacy models like octbts can be left unchanged.

The patches are in gerrit and ready for review:
https://gerrit.osmocom.org/c/libosmocore/+/15888
https://gerrit.osmocom.org/c/osmo-bts/+/15918

#29 - 11/11/2019 01:27 PM - dexter

The patches above are still in review.

There were concerns about the auto-detection which of the two ways (separate measurement indications vs. included measurement indications) are used. A flag could be introduced into the structs, however this is a problem because then we need to ensure that the primitives are initialized before use. As far as I can see there is no initialization yet. An alternative way would be a function in the higher layers that is called to switch between the two methods. Apart from that I think the auto-detection we have is sufficient but I am also happy to replace it with something different as long as it does not require to touch lgacy-bts code.

Files

notes.txt	5.18 KB	01/21/2019	dexter
notes.txt	7.12 KB	01/23/2019	dexter