OsmoPCU - Feature #1823

Ericsson PCU TRAU frame encoding/decoding

10/15/2016 03:42 PM - laforge

Status:	Stalled	**Start date:**	10/15/2016
Priority:	Low	**Due date:**	
Assignee:		**% Done:**	70%
Category:			
Target version:	RBS2000 with BSC-located PCU		
Spec Reference:			

Description

It is not yet clear if this code will be part of OsmoPCU internally, or if it is better kept in an external process.

The general idea is as follows:

- open the E1 time-slots used for PCU TRAU frames
  - try to use only full 64k slots. If 16k sub-slots required, the subchan_mux code of libosmo-abis should be reused.
- synchonize on the PCU-TRAU frames (frame_sync.c which I already wrote but never tested)
- receive + decode the TRAU frames. We need at least
  - PCU_TRAU_ER_FT_CCU_SYNC_IND
  - PCU_TRAU_ER_FT_DATA_IND for CS1, CS2, CS3, CS4
- encode + transmit the TRAU frames. We need at least
  - PCU_TRAU_ER_FT_PCU_SYNC_IND
  - PCU_TRAU_ER_FT_DATA_IND for CS1, CS2, CS3, CS4

The SYNC procedure is the starting point. The *-SYNC-IND messages are always sent in case the link is idle and nothing else is to be sent.

See Ericsson_RBS2000_GPRS for more information.

History

#1 - 10/21/2016 04:16 PM - dexter

Did some experimentation on this:

In our openbsc.cfg we already have some configuration for timeslot 7 as PDCH:

```
   timeslot 7
    phys_chan_config TCH/F_PDCH
    hopping enabled 0
    e1 line 0 timeslot 4 sub-slot 0
```

Also set "gprs mode gprs" To my understanding this should be enough for the configuration. I am not sure how the BTS is told that timeslot 7 actually is a PDCH, in the meeting we said that the PDCH configuration looks like a TCH configuration at least from the OM2000 perspective. I checked the configuration TS Configuration Request, it is indeed the same as on all other timeslots - I am a bit confused, as there must be some difference somewehere. Somehow the BTS must know the channel type. How can it know if it shall generate SYNC-IND requests on the T1 link or not if there is no difference?

There is also a problem with the dahdi devices itsself. Osmo-nitb opens all configured T1 timeslots, including the one that is expected to be related to PDCH (or P-GSL?). Since we want to use only full 64K T1 timeslots I changed the config so that it occopies only timeslot 4 (probably something is missing here). In my experiment I forced osmo-nitb not to open /dev/dhadi/4.

Then I started osmo-nitb and watched out for something that could be a SYNC-IND on /dev/dhadi/4:

```
root@sysmocom:~# dd if=/dev/dahdi/4 | hexdump -C
00000000  7e 7e 7e 7e 7e 7e 7e 7e  7e 7e 7e 7e 7e 7e 7e 7e  |~~~~~~~~~~~~~~~~|
*
003ea600  fe fe fe fe fe fe fe fe  fe fe fe fe fe fe fe fe  |................|
*
00ec5db0  fe fe fe fe fe fe fe fe  fe fe 7e 7e 7e 7e 7e 7e  |..........~~~~~~|
00ec5dc0  7e 7e 7e 7e 7e 7e 7e 7e  7e 7e 7e 7e 7e 7e 7e 7e  |~~~~~~~~~~~~~~~~|
*
00f0da00  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
```

According to https://osmocom.org/projects/openbsc/wiki/Ericsson_RBS2000_GPRS the SYNC-IND should be something more complex than that what I get. It definitely would be something that is recognizable.

I am not sure how the development state is here? Did we ever observe SYNC-IND messages in an experiment yet? Or is this something that we just assume to happen, but never checked?

#2 - 10/29/2016 05:15 PM - laforge

On Fri, Oct 21, 2016 at 04:16:38PM +0000, dexter [REDMINE] wrote:

> e1 line 0 timeslot 4 sub-slot 0

by using this, you are asking the BTS to only use one 16k sub-slot for PCU TRAU frames, not an entire 64k slot. This way, only GPRS is possible, no EGPRS. Still, it might be intresting to get at least that working? When using sub-slots, you need the sub-channel de-multiplexer that we already use for the regular voice TRAU frames.

Also, I expect you must be using the VTY command for switching the dynamic tch/pdch on ts7 to PDCH mode. When I tried that last, OsmoNITB was starting to repotr "unknown TRAU frame 0x15" to me. Which indeed is PCU_TRAU_ER_FT_CCU_SYNC_IND.

It is not clear to me if using "sub-slot full" in the config file will configure the IS in a way to use a full 64k slot for the given air-interface timeslot. Most likely this is not implemented yet.

#3 - 11/09/2016 05:26 PM - dexter

- Status changed from New to In Progress

- % Done changed from 0 to 40

I wrote a prototype (a dummy for the l2tp daemon side and code to be placed inside the PCU later) to handle the PGSL trau frames. It currently only has a dummy-pcu that consits of dummy functions with the exact same name and parameter list like the real functions in the pcu. I can receive ULDATA.ind and DLDATA.req. DLDATA.req triggers the mechanism of sending a DLDATA.ind. Unfortunately the encoding of the pgsl message fails for me (i am sure its something minor).

#4 - 11/10/2016 04:57 PM - dexter

- % Done changed from 40 to 50

The encoding of the PGSL frames now seems to work fine. The frame type (CS1-CS4) is determined in l1if_pdch_req() by looking at the length of the data that comes from the PCU. It looks like that MCS frames need some rework before we can hand them off into the PGSL encoder.

There are a few questions left concerning l1if_pdch_req():

int l1if_pdch_req(void *obj, uint8_t ts, int is_ptcch, uint32_t fn, uint16_t arfcn, uint8_t block_nr, uint8_t *data, uint8_t len)

- What is the *obj pointer for, can it I ignore it?
- What is the is_ptcch for? Does it affect the way I handle the frames (probably it affects frame.u.dldata_ind.ucm?)
- Do we ignore the arfcn field?

#5 - 11/11/2016 06:45 PM - dexter

- % Done changed from 50 to 70

The unix domain socket now accepts pgsl frames with 2 byte LAPD header. The SAPI is expected to be 12, the TEI is used to reference the transceiver. The arfcn in l1if_pdch_reg() is used to determine the TRX/TEI, which is then included in the LAPD header. So I think we are complete now and we can integrate it in the PCU now.

#6 - 11/14/2016 05:00 PM - dexter

Integrated my lapd code into the PCU, did not test it very much, bit it seems to be ok on the first look. The pcu now connects towards the l2tpd. If the l2tpd is going down it retries periodically until it gets the l2tpd connection agin.

#7 - 11/15/2016 06:06 PM - dexter

After some tests and debugging I get plausible log messages from the PCU when I send handcrafted ULDATA.ind/DLDATA.req. It looks to me like if it could work, it now needs to be tried out in a more realistic setup

#8 - 01/19/2017 11:58 AM - dexter

- Status changed from In Progress to Stalled

#9 - 05/17/2018 02:02 PM - laforge

- Priority changed from Normal to Low

#10 - 09/04/2019 09:26 AM - laforge

- Assignee deleted (dexter)