The idea of baseband frequency hopping is to "commutate" the bursts among the available TRX.

So we have a BTS with e.g. 4 TRX. TRX0 carries the CCCH and hence cannot do hopping. However, TRX1 to TRX3 can implement baseband hopping.

This works by having each of the TRX on a fixed ARFCN (no tuning or actual hopping of the carrier frequency). The trick is to then assign dedicated channels (TCH + SDCCH) not on one ARFCN but on a hopping sequence that “hops” between those three ARFCN, basically jumping to a different ARFCN for every burst.

This feature would be implemented in osmo-bts. It has four separate streams of bursts to/from the four TRX in the example above, and it needs to re-order them according to the hopping sequences before passing them into the convolutional decoder.
This explains all the mappings that we need to add to osmo-bts scheduler as a new abstraction layer level.

### 6.2.3 "Hopping sequence generation"

This section explains how to get the ARFCN based on input params (MAI, N=len(MA), FN, MAIO, HSN. (see "Figure 6" for graphical input/output params).

### 6.2.4 Specific cases

Frequency hopping is not permitted on any timeslot supporting a BCCH according to table 3 of clause 7. A non-hopping radio frequency channel sequence is characterized by a mobile allocation consisting of only one radio frequency channel, i.e. with N=1, MAIO=0. In this instance sequence generation is unaffected by the value of the value HSN.

"Figure 1" is quite useful to understand the mapping in different layer levels. "Figure 4", "Figure 5", "Figure 6" shows hoping in action.

### Current Status Summary:

- Configuration of hopping params supported in osmo-bsc, as well as sending values over OML (afaui because we supported that in the past with an older non-osmo-bts BTS)
- Only initial minimal retrieving implemented in osmo-bts (oml.c), but hopping not supported in general.

### Steps:

- osmo-bts:
  - It lacks most of the hopping support. OML messages take the params but do nothing (eg oml_c oml_rx_set_radio_attr and oml_rx_set_chan_attr). Right now osmo-bts expects arfcn-list to be len()=1.
  - Towards osmo-trx, we probably need to enable HANDOVER on all ARFCNs in MA group (ARFCN list) for that TS? it also means we need some ref counting structure since several channels may require it to be set.
  - Also in osmo-trx, we specifically set each TS with its type. TODO: checked what needs to be done there regarding this.
  - Update in lots of placed (grep "hopping") that hopping is not implemented in osmo-bts.

- scheduler:
  - Downlink (Tx): trx_sched Fn in scheduler_trx.c calls _sched_rts() and _sched_dl_burst() for each timeslot, but when using frequency hoping then this would be wrong, we need to somehow convert arfcn + fn => ts, and so far AFAIU the spec only provides functions to go the opposite path...
  - We could perhaps go this way and simply end up sending on a different arfcn for which it was polled?

### TODO:

- Channel assignment parameters (RSL? 08.58? already done by BSC?)
- Find out how to infer TS from arfcn + fn (UL or RTS), and other scheduler required changes.
- Pass information to PCU

#2 - 05/12/2020 01:26 PM - pespin

### Regarding operational sample configuration:

The idea is to set, for the same TS number on all TRXs [len(TRXs)=N], the same HSN (Hopping sequence Number) and a different MAIO (0..N-1) So for instance this would be possible:
phys_chan_config TCH/H
hopping enabled 1
hopping sequence-number 3
hopping maio 1
hopping arfcn add 870
hopping arfcn add 872
timeslot 3
phys_chan_config TCH/F
hopping enabled 1
hopping sequence-number 4
hopping maio 1
hopping arfcn add 870
hopping arfcn add 872
timeslot 4
phys_chan_config TCH/F
hopping enabled 1
hopping sequence-number 5
hopping maio 1
hopping arfcn add 870
hopping arfcn add 872
timeslot 5
phys_chan_config TCH/F
hopping enabled 1
hopping sequence-number 6
hopping maio 1
hopping arfcn add 870
hopping arfcn add 872
timeslot 6
phys_chan_config PDCCH
hopping enabled 1
hopping sequence-number 7
hopping maio 1
hopping arfcn add 870
hopping arfcn add 872
timeslot 7
phys_chan_config PDCCH
hopping enabled 1
hopping sequence-number 8
hopping maio 1
hopping arfcn add 870
hopping arfcn add 872
trx 1
timeslot 0
phys_chan_config TCH/H
hopping enabled 1
hopping sequence-number 1
hopping maio 2
hopping arfcn add 870
hopping arfcn add 872
timeslot 1
phys_chan_config TCH/H
hopping enabled 1
hopping sequence-number 2
hopping maio 2
hopping arfcn add 870
hopping arfcn add 872
timeslot 2
phys_chan_config TCH/H
hopping enabled 1
hopping sequence-number 3
hopping maio 2
hopping arfcn add 870
hopping arfcn add 872
timeslot 3
phys_chan_config TCH/F
hopping enabled 1
hopping sequence-number 4
hopping maio 2
hopping arfcn add 870
hopping arfcn add 872
timeslot 4
phys_chan_config TCH/F
hopping enabled 1
hopping sequence-number 5
hopping maio 2
hopping arfcn add 870
hopping arfcn add 872
Regarding RSL CHan ACT

Check TS 08.58 sec "9.3.5 Channel Identification" IE, where it points to TS 0408:

04.08 "Channel Description" *
04.08 "Mobile Allocation" *

A * denotes that the whole of the 04.08 element including the element identifier and length should be included. The 04.08 "Mobile Allocation" shall for compatibility reasons be included but empty, i.e. the length shall be zero.

From there, TS 04.08 sec "10.5.2.5 Channel Description" explains how to either encode the ARFCN or the MAIO + HSN.

Similarly, "10.5.2.21 Mobile Allocation" IE encodes an ARFCN List as a bitmap of 1 and 0. It, however, must be always empty according to spec TS 08.58.

This is already implemented in osmo-bsc.git abis_rsl.c rsl_tx_chan_activ() as pointed out by Harald. On the other hand, in osmo-bts.git, we are not even parsing RSL_IE_CHAN_IDENT in rsl.c rsl_rx_chan_activ(), which means nowadays we are not even verifying that the ARFCN in there fine (we probably assume the ARFCN for the trx associated to the rsl connection is used).

We also need to make sure we send the CA (list of arfcns) in BCCH in the BTS:

If frequency hopping is applied, the cell allocation if present in the message is used to decode the mobile allocation. If the cell allocation is not included, the mobile station uses its current cell allocation, the current CA is the last CA received on the BCCH. Afterward, the current CA may be changed by some messages sent on the main signalling link containing a CA (the possible messages are: ASSIGNMENT COMMAND, HANDOVER COMMAND and FREQUENCY REDEFINITION). Note that there are cases in which the current CA is undefined, see sub-clause 3.4.3.3.

Regarding PCU

Taking over from what Harald did look at, on osmo-pcu.git we need to:

- pass Mobile Allocation bitmap over the PCU socket (only if it turns out we need to implement "Frequency Parameters" for indirect mode in "Immediate assignment", see further below). We actually don't need to add this anyway because it's already being sent to BSC-BTS-PCU through System Information 13 element "GPRS Mobile Allocation : GPRS Mobile Allocation IE >" -- Defined in 3GPP TS 04.6012.10a GPRS Mobile Allocation"
- Pass per-pdch TS (MAIO, HSN) parameters over the PCU socket so the PCU knows about them: MAIO, HSN. We could try encoding it inside 16bit arfcn (we need 6 bit for MAIO and 6 bit for HSN, so 12 of 16 total, plus some way to differentiate between hopping enabled or disabled). It needs to be done at least for info_ind, when in osmo-pcu arfcn is copied and later used. Other usages look more like gsmtap related.
- encode the hopping parameters in any kind of RLC/MAC messages, specifically
  - Encoding::write_immediate_assignment()
  - Encoding::write_packet_uplink_assignment()
  - Encoding::write_packet_downlink_assignment()

TS 04.08:

- "10.5.2.16 IA Rest Octet" Included in "IMMEDIATE ASSIGNMENT". Contains struct "Frequency Parameters", which in turn contains MAIO and Mobile Allocation. We don't seem to need this though, since we only send Downlink and Uplink options which doesn't contain those.
"10.5.2.25a Packet Channel Description": Included in "IMMEDIATE ASSIGNMENT". It has a bit to encode either ARFCN or MAIO + HSN. In there, there seems to be "Direct" vs "Indirect" "encoding of hopping RF channel configuration", the first providing MAIO+HSN and the other one MAIO+MA_NUMBER_IND. We probably want to use direct since we have MAIO + HSN from the BTS anyway).

Summary of steps:
- Move osmocom-bb hopping functions to libosmocore
- osmo-bts.git: Rework scheduler-trx (plus generic scheduler structures) to decouple 1 trx = 1 specific rfch. Use libosmocore rfch functions in in schedueler-trxto convert rfch<->trx and pass logical trx to generic scheduler. (we may want to add some generic code in the scheduler to do so).
- pcuif: Find way to include MAIO+HSN (12bit) into uint16_t to avoid changing protocol. Otherwise, change structs to accomodate MAIO+HSN. TRX in pcuinf continues to work the same since they are considered logical TRXs.
- osmo-pcu.git: Write MAIO+HSN in write_assignment if hopping is enabled
- faketrx: We may have to add support for hopping in faketrx to test with TTCN3

#3 - 05/12/2020 01:26 PM - laforge

pespin: Let me illustrate what happens in the following tables.

Assumptions:
- 5-TRX BTS (C0..C4 each represent one ARFCN)
- BCCH on TRX0/TS0
- sequential hopping sequence (for illustration, works for any other sequence)
- b = BCCH/CCCH
- A..E = logical timeslots (think of TCH/F or PDCH)

**TS0**

<table>
<thead>
<tr>
<th>TRX</th>
<th>C0</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn=0</td>
<td>b</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>fn=1</td>
<td>b</td>
<td>D</td>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>fn=2</td>
<td>b</td>
<td>C</td>
<td>D</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>fn=3</td>
<td>b</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>A</td>
</tr>
<tr>
<td>fn=4</td>
<td>b</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
</tbody>
</table>

=> BCCH + 4 dedicated timeslots (A..D)

**TS1..7**

<table>
<thead>
<tr>
<th>TRX</th>
<th>C0</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
</tr>
</thead>
<tbody>
<tr>
<td>fn=0</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>fn=1</td>
<td>E</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>fn=2</td>
<td>D</td>
<td>E</td>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>fn=3</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>fn=4</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>A</td>
</tr>
<tr>
<td>fn=5</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
</tr>
</tbody>
</table>

=> 5 dedicated timeslots (A..E)

#4 - 05/12/2020 01:29 PM - laforge

- Precedes Feature #4547: support for frequency hopping in osmo-pcu added

#5 - 05/12/2020 07:50 PM - fixeria

- Status changed from New to In Progress

I would like to follow test-driven development approach here, so my current plan is:

1. Implement freq. hopping support for the MS side of ttcn3-bts-test:
   - trxcon remains unchanged and still maintains a single TRXC/TRXD connection
     - on TRXC we already have an MS specific command: `SETFH <HSN> <MAIO> <CH1> <CH2> [...] <CHN>`
   - fake_trx.py gets the actual implementation of freq. hopping

05/15/2020
hopping sequence is generated from parameters indicated by SETFH
in Transceiver/FakeTRX every burst coming from the BurstForwarder triggers freq. change
unit tests!

- BTS_Tests.f_est_dchan() must be extended:
  - to be able to send RSL MT_IMMEDIATE_ASSIGN_CMD with hopping parameters
  - to be able to send L1CTL DM_EST_req with hopping parameters

2. Implement test cases as a part of ttcn3-bts-test:
   - 2.0. Investigate what needs to be changed or refactored in order to indicate freq. hopping parameters to osmo-bts-trx (IUT)
     - osmo-bts.cfg in docker-playground needs to be changed, so osmo-bts-trx knows about all additional transceivers
     - osmo-bsc.cfg in docker-playground needs to be changed, so osmo-bsc can OML-bootstrap all transceivers for us
     - fake_trx.py needs to know about all additional transceivers too (by default we have 1 MS + 1 BTS only)
   - 2.1. Make sure that L2 frames on different kinds of logical channel (SDCCH, TCH, PDCH) can be sent and received when freq. hopping is in use
   - 2.2. Make sure that several channels with and without freq. hopping can co-exist together, not interfering with each other
   - 2.3. Make sure that BCCH on C0/TRX0 can still be decoded when at least one dedicated channel is hopping
   - 2.4. Make sure that CBCH on SDCCH/0 can still be decoded when SI4 indicates that it
   - FIXME: record GSM_RR_Types.MaioHsn is empty o_O

3. Finally, implement baseband freq. hopping support in osmo-bts-trx:
   - 3.0. Illustrate the current bts/trx/ts/lchan dependency diagram, investigate what needs to be changed
     - obviously we need to decouple both logical channel state and trx/ts state
     - logical channel handlers (rx_data_fn, rx_tchf_fn, etc.) shall have no access to anything else than their associated state
     - this is how it's (almost) done in trxcon, but still: *trx is needed to know TSC for Tx bursts, *ts provides multiframe layout info...
     - we can just store what's needed in the logical channel state
   - 3.2. Implement handling of the hopping related parameters in the OML messages
   - 3.3. Implement the actual code for dispatching bursts to and from the transceivers
   - 3.4. Indicate support of freq. hopping to the BSC among with the other features

#6 - 05/12/2020 08:07 PM - fixeria

Some more additional notes:

- 1. Implement freq. hopping support for the MS side of ttcn3-bts-test:
  - Calypso PHY does not require any changes and can be also used in the testing setup

- 3. Finally, implement baseband freq. hopping support in osmo-bts-trx:
  - 3.5. We already have hopping sequence generation code in OsmocomBB, makes sense to move it to libosmocore

#7 - 05/13/2020 08:55 PM - fixeria

3.5. We already have hopping sequence generation code in OsmocomBB, makes sense to move it to libosmocore

https://gerrit.osmocom.org/c/libosmocore/+/18244 libosmogsm: import hopping sequence generation code
https://gerrit.osmocom.org/c/libosmocore/+/18245 libosmogsm: add Doxygen docs for gsm0502_hop_seq_gen()