As was reported by keith (originally in #1526), with ‘gprs control-ack-type-rach’ enabled in the BSC, osmo-pcu logs:

```
DL1IF <0001> ../../git/src/osmo-bts-sysmo/sysmo_l1_if.c:251 Rx PH-RA.ind for unknown L1 SAPI PRACH
```

this is a regression introduced in I482d60a46b9d253dfe0b16140eac9fea6420b30c:

```
Parent: 17954da5 (pcuif_proto.h: extend RACH.ind with TRX / TS numbers)
Author: Vadim Yanitskiy <axilirator@gmail.com>
AuthorDate: 2019-10-05 23:45:31 +0700
Commit: Vadim Yanitskiy <axilirator@gmail.com>
CommitDate: 2019-11-23 17:42:45 +0700

PTCCH: properly handle RACH.ind for PCU_IF_SAPI_PTCCH
	Change-Id: I482d60a46b9d253dfe0b16140eac9fea6420b30c
Related: OS#1545
```

I would expect Packet Control Ack on GsmL1_Sapi_Pdtch or rather on GsmL1_Sapi_Pacch, but apparently (for some magic reason) it arrives on GsmL1_Sapi_Prach.

According to 3GPP TS 45.002, section 3.3.3.2.1:

- PRACH stands for Packet Random Access CHannel,
- used to request assignment of one or several PDTCHs,
- belongs to Packet Common Control Channel (PCCCH)!

Therefore the Packet Control Ack has nothing to do with PRACH. Neither we support PCCCH (it has been deprecated by the specifications).

```
GsmL1_LogChComb_XIII ///< PDTCH/F + PACCH/F + PTCCH/F
```

This odd behaviour seems to be caused by the following code (see #4500):

```c
static const struct sapi_dir pdtch_sapis[] = {
    { GsmL1_Sapi_Pdtch, GsmL1_Dir_TxDownlink },
    { GsmL1_Sapi_Pdtch, GsmL1_Dir_RxUplink },
    { GsmL1_Sapi_Ptcch, GsmL1_Dir_TxDownlink },
    { GsmL1_Sapi_Pracch, GsmL1_Dir_RxUplink }, // <-- ???
#if 0
    { GsmL1_Sapi_Ptcch, GsmL1_Dir_RxUplink },
    { GsmL1_Sapi_Pacch, GsmL1_Dir_TxDownlink }, // <-- ???
#endif
};
```

where we enable GsmL1_Sapi_Pracch on GsmL1_LogChComb_XIII (again, there is no PRACH in XIII). I guess we should enable GsmL1_Sapi_Pacch on direction GsmL1_Dir_RxUplink instead.

For now I submitted a blind fix: https://gerrit.osmocom.org/c/osmo-pcu/+/17669. Let's keep it WIP for now.

Related issues:
As it turns out, there is a nice (documented) feature that allows the DSP to send debug messages to /dev/rtfifo/dsp_trace. Here is what I see when trying to enable `GsmL1_Sapi_Pacch` on `GsmL1_Dir_RxUplink`:

MphActivateReq => [ hLayer1 = 0x87080020, hLayer2 = 0x600BB, u8Tn = 6, sapi = 16, dir = 2, subCh = 31, bfi = -200.00 dBm ]

    cfgParam = 00000000 00000000 00000000 00000000

[ERROR] : LogChMng_GetTypeXIIIBufAddr() => Unsupported sapi in U/L [logChCombXIII, sapi = 16 ]

[ERROR] : Device_ActivateLogCh()=> Unable to activate. [pPhyCh = 0x87089AB0, pLogChMngBuf = 0x870801FC, pi8Buf fer = 0x87089F90]

MphActivateCnf => [ hLayer3 = 0x600BB, hLayer1 = 0x87080020, TN = 6, sapi = 16, dir = 2, subCh = 31, status = -4 ]

same error when enabling on `GsmL1_Dir_TxDownlink`:

MphActivateReq => [ hLayer1 = 0x87080020, hLayer2 = 0x600BB, u8Tn = 6, sapi = 16, dir = 4, subCh = 31, bfi = -200.00 dBm ]

    cfgParam = 00000000 00000000 00000000 00000000

[ERROR] : LogChMng_GetTypeXIIIBufAddr() => Unsupported sapi in D/L [logChCombXIII, sapi = 16 ]

[ERROR] : Device_ActivateLogCh()=> Unable to activate. [pPhyCh = 0x87089AB0, pLogChMngBuf = 0x870801FC, pi8Buf fer = 0x87089F90]

MphActivateCnf => [ hLayer3 = 0x600BB, hLayer1 = 0x87080020, TN = 6, sapi = 16, dir = 4, subCh = 31, status = -4 ]

This is odd and confusing. From 3GPP TS 45.002, section 6.4.1 "Permitted channel combinations onto a basic physical channel":

  xiii)  PDTCH/F + PACCH/F + PTCCH/F