PDCP

I would say PDCP is a kind of interface between inside world and outside world. By "Inside World", I roughly mean "Network Access Network(mainly eNodeB) and Controller for eNodeB", "Network Emulator Hardware and internal firmware", and "Mobile Device (UE)".

- Overview
- PDCP Data Structure
- Integrity Protection Process
- Examples
  - Example : PDCP <-- DL DCCH (RRC) : RRC Connection Reconfiguration
  - Example : UL DCCH (RLC) --> PDCP : RRC Connection Reconfiguration Complete
  - Example : DL DTCH -> PDCP : IP Data
  - Example : ROHC Feedback

Overview

The data coming into the box (or UE) first go through PDCP and then gets into RLC. Data waiting in RLC trying to go out to the outside world has to go through PDCP to outer world.

In the case when everything goes OK, you don't see any problems in protocol stack and just check protocol log you may think the function of PDCP looks simple and all it does is just to add/remove a small header to each packet and send/recieve it from/to RLC. But if you list up all the detailed functions of PDCP, you would get pretty long list as shown below.

- Transfer of Data (C-Plane and U-Plane) between RLC and Higher U-Plane interface
- Maintenance of PDCP SN(Sequence Number)
- Transfer of SN Status (for use Upon Handover)
- ROHC (Robust Header Compression)
- In-Sequence delivery of Upper Layer PDUs at re-establishment of lower layer
- Elimination of duplicate of lower layer SDUs at re-establishment of lower layer for RLC AM
- Ciphering and Deciphering of C-Plane and U-Plane data
- Integrity Protection and Integrity verification of C-Plane Data
- Timer based Discard
- Duplicate Discard

As in other layers, let's start looking into the diagram from 3GPP specification (TS 36.323). The first diagram is as follows.
What are we supposed to do ?
Yes, Verbalize it (describe it).

In the following diagram, you see that PDCP is directly connected to RLC Layer (RLC UM and RLC AM).
Do you find anything strange or missing ?
Yes, PDCP has no connection to RLC TM mode, meaning RLC TM mode data does not go through PDCP.

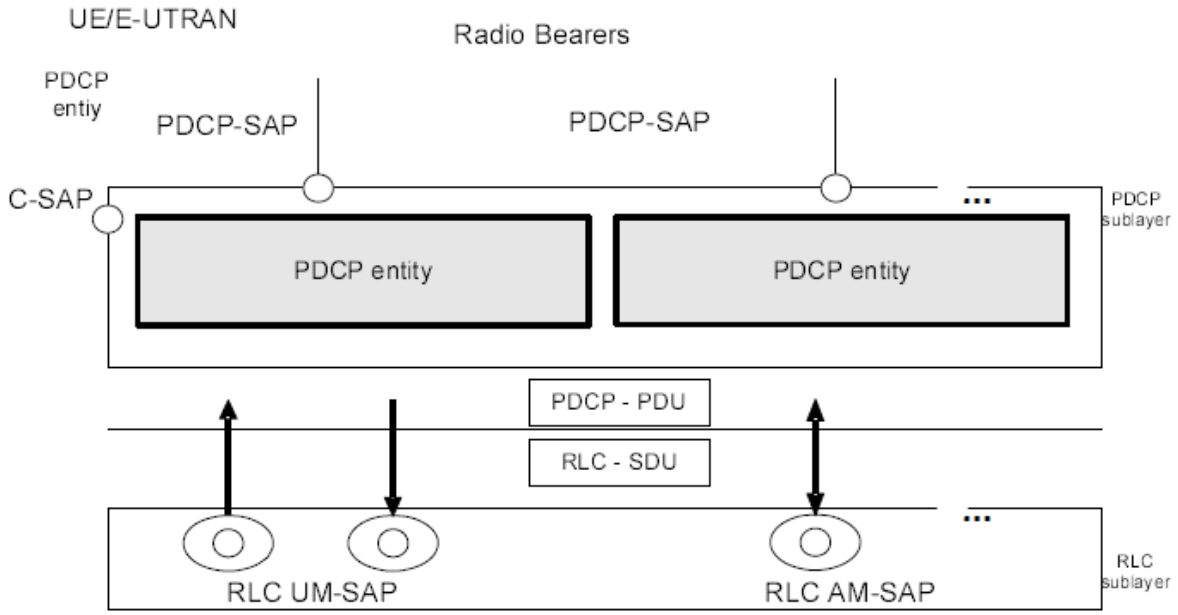


Figure 4.2.1.1 - PDCP layer, structure view

Following Diagram would give you more detailed information about PDCP Operation. Of course, all of the these functionality is listed in the 3GPP specification, but it would not become yours unless you combine these diagrams and the descriptions in the specification.

Now let's follow through the diagram from left side.

i) Data coming into PDCP first go through "Sequence Numbering" Procedure. It means that PDCP add "Sequence Number" to each of incoming data block. Once it add 'Sequence Number', it has to manage the number. On reciever side, we can figure out many things like "Is the data getting delivered in order ? Is there any duplicate data ? How can I combine the multiple chunks of data block into an original big chunk data ?"
ii) Then it goes through Header Compression. But it says "this applies only to U-plane data". It means that Signaling Message does not go through this Header Compression. Even though not shown in this diagram, we can disable Header Compression even for U-plane data (e.g, IP Packet data).
iii) From here we see two paths, one through "Integrity/Ciphering" and the other one directly goes to the last step. Integrity Protection applies only to C-Plane data (C-Plane data means RRC/NAS message, i.e DCCH data, not DTCH data). Again you can disable "Integrity Protection" setp by applying IEA0 to this process.
iv) Then it goes to Ciphering process. Ciphering applies both C-Plane and U-Plane Data. Ciphering process can also be disabled by applying EEA0.
v) Eventually at the last step of transmission PDCP, a header is added and get out of PDCP layer.

The other side (right side, receiving side) is simply reverse process of transmission process. So I will leave this to the readers to interpret the diagram.

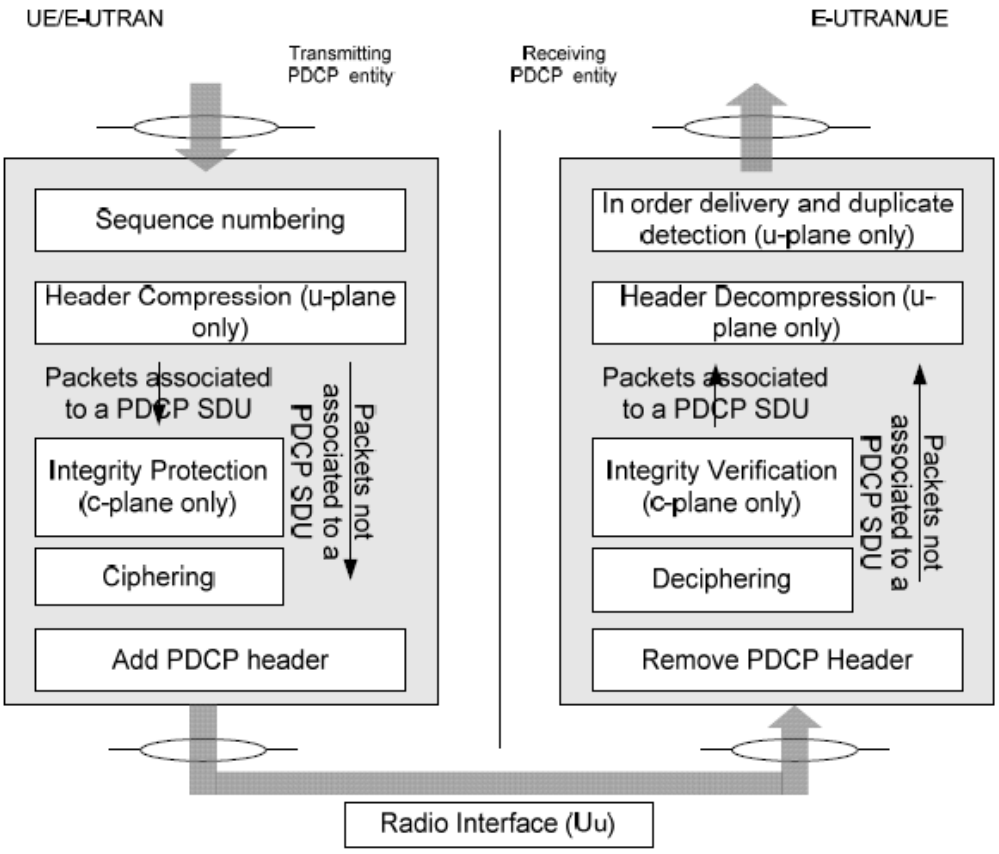


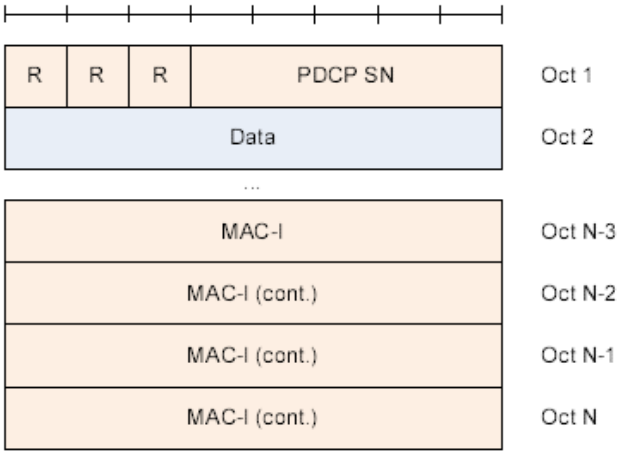
Figure 4.2.2.1 - PDCP layer, functional view

PDCP Data Structure

As in MAC PDU, RLC PDU.. PDCP also has it's own data structure and again you would never understand this structure in detail unless you get some real data and anlayze the data manually. But putting all together the following diagram (from 3GPP 36.323) would guide you to decode PDCP Data.

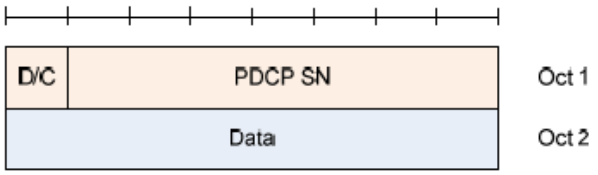
The only key question when you try to decode real PDCP data would be "Which diagram I have to use to decode this data ?". Only practice would give you the quick answer.

First Let's briefly look at the data structure. Following is the data structure for SRB. It means all the C plane data (RRC/NAS message) use this data structure. As you see, the first three bits are reserved. and next comes 5 bits PDCP SN(Sequence Number) and at the end of the block 4 bytes of MAC-I data is attached.

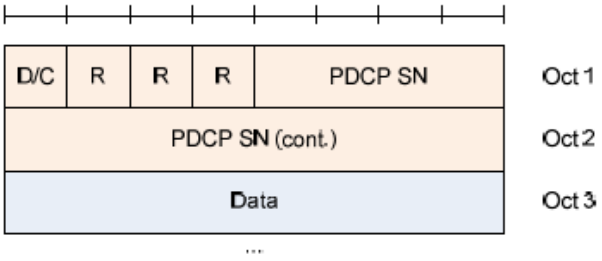


< 36.323 Figure 6.2.2.1 : PDCP Data PDU Format for SRBs >

Following two diagram shows the PDCP structure for DRB. It means U-Plane data, DTCH data is using this kind of data structure. The only differences between these two data type is the size of SN (Sequence Number). One is using 7 bit and the other is 12 bits.
Comparing to SRB PDCP structure, you would find two differences. D/C bit at the first bit of the structure. This tells you whether the PDCP data is for user data or PDCP control data (This control data does not mean C-Plane data. It is control data being used in PDCP layer). And the other difference is that you do not have the MAC-I at the end of the data. MAC-I is the special information for Integrity protection. Do you remember Integrity Protection applies only to C-Plane data, not to U-Plane data ?

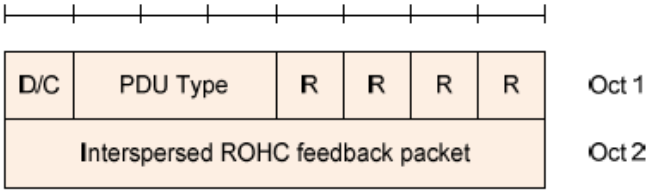


< 36.323 Figure 6.2.4.1 : PDCP Data PDU Format for DRBs using a 7 bit SN >

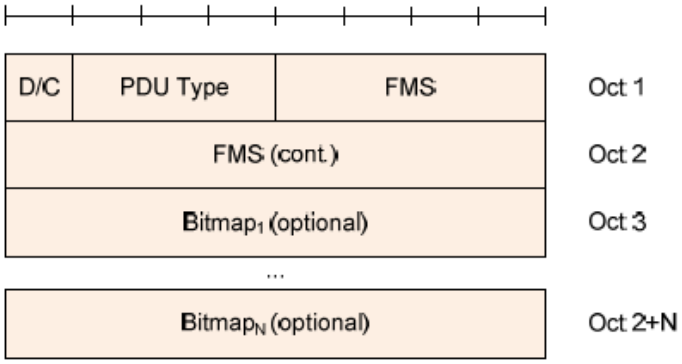


< 36.323 Figure 6.2.3.1 : PDCP Data PDU Format for DRBs using a 12 bit SN >

Following two data structure is a kind of PDCP layer control packet. Most of the LTE PDCP PDU you woud see would be the Data PDU. However, you may see the PDCP packet with D/C field = 0 (i.e, Control PDU) when PDCP PDU(s) get lost during the communication or you will often see these packets when ROHC is in action. In the case of missing PDCP packets, you would see Figure 6.2.6.1 structure and in ROHC case you would see Figure 6.2.5.1 structure as shown below.



< 36.323 Figure 6.2.5.1: PDCP Control PDU format for interspersed ROHC feedback packet >



< 36.323 Figure 6.2.6.1: PDCP Control PDU format for PDCP status report using a 12 bit SN >

All of the following tables shows the meaning of each data field shown in the diagrams we saw above. You will refer to this table a lot when you try to decode PDCP data yourself. Otherwise, you would read now and forget as your click to another page.

< 36.323 Table 6.3.2.1: PDCP SN length >

Length	Description
5	SRBs
7	DRBs, if configured by upper layers (*pdcp-SN-Size* [3])
12	DRBs, if configured by upper layers (*pdcp-SN-Size* [3])
15	DRBs, if configured by upper layers (*pdcp-SN-Size* [3])
16	SLRBs

< 36.323 Table 6.3.7.1: D/C field >

Bit	Description
0	Control PDU
1	Data PDU

< 36.323 Table 6.3.8.1: PDU type >

Bit	Description
000	PDCP status report
001	Interspersed ROHC feedback packet
010-111	reserved

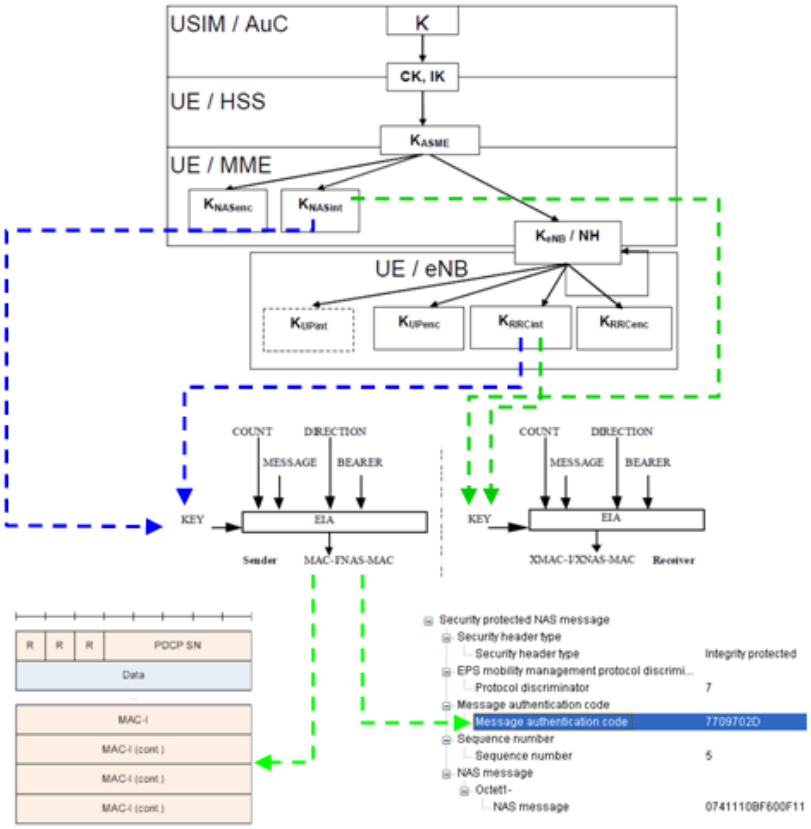
< 36.323 Table 6.3.10.1 Bitmap >

Bit	Description
0	PDCP SDU with PDCP SN = (FMS + bit position) modulo (Maximum_PDCP_SN + 1) is missing in the receiver. The bit position of N^{th} bit in the Bitmap is N, i.e., the bit position of the first bit in the Bitmap is 1.
1	PDCP SDU with PDCP SN = (FMS + bit position) modulo (Maximum_PDCP_SN + 1) does not need to be retransmitted. The bit position of N^{th} bit in the Bitmap is N, i.e., the bit position of the first bit in the Bitmap is 1.

Integrity Protection Process

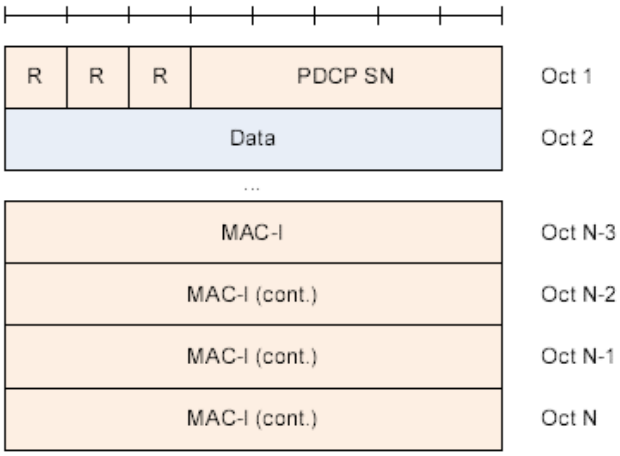
My perception of any process is "a black box that convert a set of inputs into a set of output". So for any process, I usually tries to define what is the inputs and out is the final outputs.. and then try to find the all of the steps one by one to convert the input into the outputs.

If I define LTE Integrity Protection Procedure with this logic, it would be "a process that convert 'K' value from USIM into PDCP MAC-I and NAS MAC(Message Authentication Code)". Describing this process with illustration would be as follows. Just try to go through this illustration whenever you have chance and try to have your own idea first.
The process described here apply only to C-Plane data. (Refer to 33.401 for the detailed description of each of the steps).



< Example : PDCP <-- DL DCCH : RRC Connection Reconfiguration >

This example PDCP PDU has following structure. The SDU/PDU is from eNB PDCP.



DL DCCH RRC : 20 06 03 59 3D 11 4D E1 B8 18 3A 10 17 00 30 00 07 88 80 00 08 01 B2 91 26 08 08 48 68 23 4B 73 2B A0 23 13 2B 63 60 13 1B 08 28 0E 05 40 08 09 38 DC 04 01 08 80 18 00 00 84 08 36 05 40 08 14 18 36 05 40 08 10 00 68 26 05 40 08 12 80 5F B0 07 88 84 00 08 00 00 00 00 08 98 07 88 80 00 09 18 2F A0 00 00 00 08 BF 02 CF 03 20 08 1F 93 53 E0 FA 81 BC 0F A8 E0 60 78 26 3E 9D 52 D2 E8 00 FB 50 65 03 FF 04 07 E8 10

PDCP PDU : **06** 20 06 03 59 3D 11 4D E1 B8 18 3A 10 17 00 30 00 07 88 80 00 08 01 B2 91 26 08 08 48 68 23 4B 73 2B A0 23 13 2B 63 60 13 1B 08 28 0E 05 40 08 09 38 DC 04 01 08 80 18 00 00 84 08 36 05 40 08 14 18 36 05 40 08 10 00 68 26 05 40 08 12 80 5F B0 07 88 84 00 08 00 00 00 00 08 98 07 88 80 00 09 18 2F A0 00 00 00 08 BF 02 CF 03 20 08 1F 93 53 E0 FA 81 BC 0F A8 E0 60 78 26 3E 9D 52 D2 E8 00 FB 50 65 03 FF 04 07 E8 10 **0C F3 6E 17**

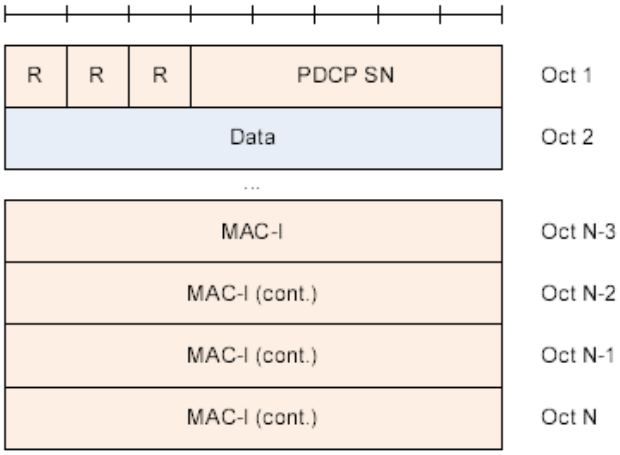
Header : 06 (Hex), 00000110 (Bin)

R - 0
R - 0
R - 0
SN - 00110 (Bin), 6 (Dec)

MAC-I : 0C F3 6E 17 // This is added by eNB PDCP Layer. UE is comparing this with the one it calculated internally.

< Example : UL DCCH --> PDCP : RRC Connection Reconfiguration Complete >

This example PDCP PDU has following structure. The SDU/PDU is from eNB PDCP.



UL DCCH RLC : A0 07 **07** 10 00 **C4 8E 9A A7**
PDCP PDU : **07** 10 00 **C4 8E 9A A7**

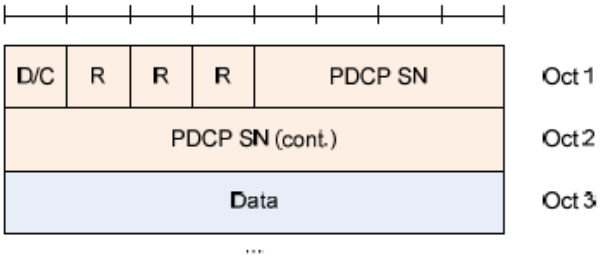
Header : 07 (Hex), 00000111 (Bin)

R - 0
R - 0
R - 0
SN - 00111 (Bin), 7 (Dec)

MAC-I : C4 8E 9A A7 // This is added by UEPDCP Layer. eNB is comparing this with the one it calculated internally.

< Example : DL DTCH -> PDCP : IP Data >

This example PDCP PDU has following structure. The SDU/PDU is from eNB PDCP.



DL DTCH (from IP Interface) : 45 00 00 C8 04 12 40 00 80 11 72 BF C0 A8 01 02 C0 A8 01 01 EA 60 4A 12 00 B4 AE 8B 80 08 32 60 69 14 46 F2 18 69 D6 05 D5 57 57 D7 D3 55 57 5C 41 45 5E D6 DC D4 53 5E 5F 53 D6 D0 D2 DE D8 D0 55 56 D6 DE D0 D1 57 D4 54 54 57 5D 58 5C D5 D0 D2 55 D5 D6 54 51 55 D5 55 D4 D2 D3 D1 D5 51 5C D4 D9 D2 D5 51 5D 58 5E 56 DB DB 55 54 57 52 53 52 D7 D0 D0 D2 D6 55 D5 D1 D1 D3 D3 D7 D5 D4 53 53 50 55 D7 D5 55 57 57 52 54 56 56 50 56 54 D5 54 56 55 D1 D2 D1 DD D3 D3 DD D5 D3 D7 5E 5D 51 D6 DC D6 51 5D 58 5C D2 D2 D0 51 50 D4 52 54 D6 57 51 54 55 D4 D4 D6 50 D7 DF D7 D7 54 D2 DD D3 50 5B 5D 55 D7 D1 D7 D3

PDCP PDU : **80 48** 45 00 00 C8 04 12 40 00 80 11 72 BF C0 A8 01 02 C0 A8 01 01 EA 60 4A 12 00 B4 AE 8B 80 08 32 60 69 14 46 F2 18 69 D6 05 D5 57 57 D7 D3 55 57 5C 41 45 5E D6 DC D4 53 5E 5F 53 D6 D0 D2 DE D8 D0 55 56 D6 DE D0 D1 57 D4 54 54 57 5D 58 5C D5 D0 D2 55 D5 D6 54 51 55 D5 55 D4 D2 D3 D1 D5 51 5C D4 D9 D2 D5 51 5D 58 5E 56 DB DB 55 54 57 52 53 52 D7 D0 D0 D2 D6 55 D5 D1 D1 D3 D3 D7 D5 D4 53 53 50 55 D7 D5 55 57 57 52 54 56 56 50 56 54 D5 54 56 55 D1 D2 D1 DD D3 D3 DD D5 D3 D7 5E 5D 51 D6 DC D6 51 5D 58 5C D2 D2 D0 51 50 D4 52 54 D6 57 51 54 55 D4 D4 D6 50 D7 DF D7 D7 54 D2 DD D3 50 5B 5D 55 D7 D1 D7 D3

Header : 80 48 (Hex) , 10000000 01001000 (Bin)

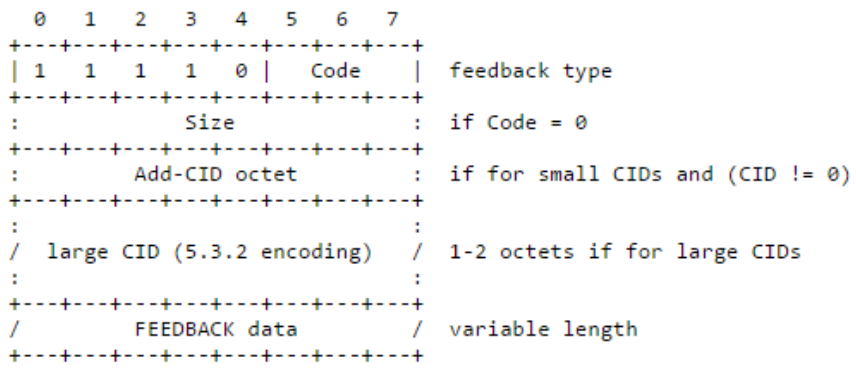
D/C - 1 // indicate that this is Data PDU
R - 0
R - 0
R - 0
SN - 0000 01001000 (Bin), 72 (Dec)

Following is the decoded IP Packet for your reference : (Blue and Pink part in the above PDU indicate Source and Destination IP address)

```
Internet Protocol Version 4, Src: 192.168.1.2, Dst: 192.168.1.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
        0000 00.. = Differentiated Services Codepoint: Default (0)
        .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 200
    Identification: 0x0412 (1042)
    Flags: 0x02 (Don't Fragment)
        0... .... = Reserved bit: Not set
        .1.. .... = Don't fragment: Set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 128
    Protocol: UDP (17)
    Header checksum: 0x72bf [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 192.168.1.2
    Destination: 192.168.1.1
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
User Datagram Protocol, Src Port: 60000 (60000), Dst Port: 18962 (18962)
    Source Port: 60000
    Destination Port: 18962
    Length: 180
    Checksum: 0xae8b [validation disabled]
        [Good Checksum: False]
        [Bad Checksum: False]
    [Stream index: 0]
Data (172 bytes)
```

< Example : ROHC Feedback >

To decode the PDU in this example, 3GPP specification is not enough because this PDCP PDU contains ROHC feedback defined in RFC 4495 as shown below. (If you are interested only in PDCP header, 3GPP is enough .. but if you want to decode it a little bit further into the palyload of the PDU, you need to look into RFC 4495 in this case)



Code: 0 indicates that a Size octet is present.
1-7 indicates the size of the feedback data field, in octets.

Size: Indicates the size of the feedback data field, in octets.

FEEDBACK data: FEEDBACK-1 or FEEDBACK-2 (see below).

CID information in a feedback element indicates the context for which feedback is sent. The LARGE_CIDS parameter that controls whether a large CID is present is taken from the channel state of the receiving compressor's channel, not from the state of the channel carrying the feedback.

Following is an example of PDCP Control PDU captured during ROHC Operation.

PDCP PDU : **10 F6** 20 00 41 16 11 39 (HEX)
PDCP PDU : 00010000 11110110 00100000 00000000 01000001 00010110 00010001 00111001 (Bin)

Header : 10 F6 (Hex) , 00010000 11110110(Bin)

D/C - 0 // indicate that this is Control PDU
PDU Type - 001 // Indicate that this is Interspersed ROHC Feedback
R - 0
R - 0
R - 0
R - 0
Interspersed ROHC Feedback - 11110110 (Bin) // If you further decode this based on RFC, it goes as follows
11110 - Fixed in RFC4995
110 - Size of the feedback Data Field in Octets