JULY 2022

5G nFAPI specifications

DOCUMENT 225.3.0

Small Cell Forum develops the technical and commercial enablers to accelerate small cell adoption and support the digital transformation of enterprises and communities.

Broad roll-out of small cells will make high-grade mobile connectivity accessible and affordable for industries, enterprises and for rural and urban communities. That, in turn, will drive new business opportunities for a widening ecosystem of service providers.

Our focus and work program reflects two key areas of diversification in the small cell ecosystem – the emergence of alternative deployment models such as neutral hosting and private networks, and Open RAN specifications enabling disaggregation of small cells at both component and network level.

We have driven the standardization of key elements of small cell technology including Iuh, FAPI, nFAPI, SON, services APIs, TR-069 evolution and the enhancement of the X2 interface. These specifications enable an open, multivendor platform and lower barriers to densification for all stakeholders.

Today our members are driving solutions that include:

- 5G components, products, networks
- Neutral host & multi-operator requirements
- Open RAN small cells & disaggregation
- Private networks & enterprise requirements
- Deployment and regulation
- Edge compute with small cell blueprint

The Small Cell Forum Release Program website can be found here: www.scf.io

If you would like more information about Small Cell Forum or would like to be included on our mailing list, please contact:

Email info@smallcellforum.org

Post Small Cell Forum, PO Box 23, GL11 5WA UK

Member Services memberservices@smallcellforum.org

Acknowledgements

Author	Company
Ganesh Shenbagaraman	Radisys
Fatih Ulupinar	Qualcomm
Andrei Radulescu	Qualcomm
Vikas Dixit	Reliance Jio
Joe Neil	Microchip

Reviewer	Company
Ravi P Sinha	Reliance Jio
Martin Lysejko	Picocom
Vicky Messer	Picocom
Ryan Husbands	BT
Rudra Kumar Shivalingaiah	Qualcomm

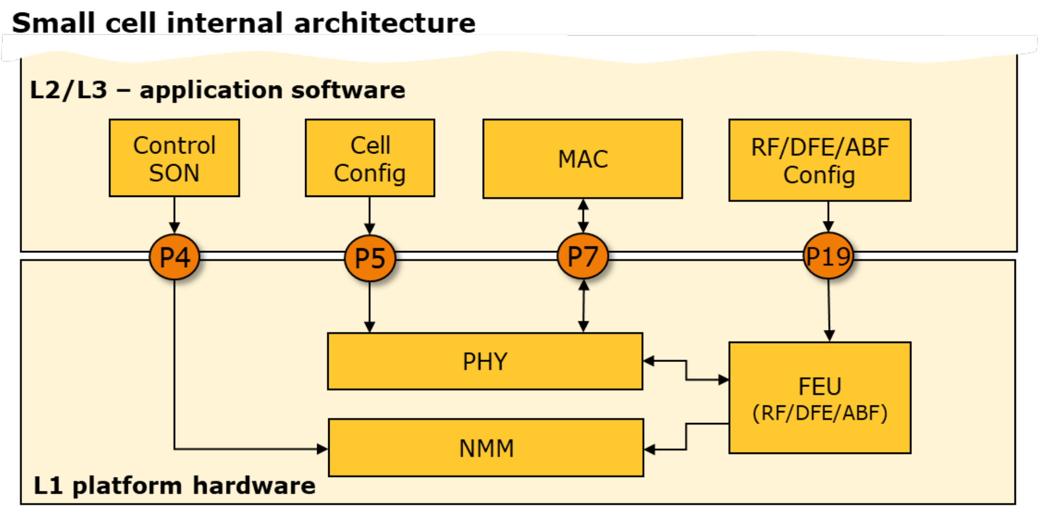
Scope

The 5G network functional application platform interface (5G nFAPI) is an initiative that extends for 5G the functional split between the MAC and PHY functions that enables virtualization of the MAC function, as identified by the study undertaken by the Small Cell Forum [SCF106][SCF159]. This virtualization is already enabled for 4G via 4G-nFAPI [SCF082]. The present document defines an equivalent transportable interface for MAC-PHY 5G function application platform interface (5G-FAPI) defined for 5G small cells [SCF222]. This RAN architecture virtualization MAC function [SCF159] is also known as the Option 6 [3GPP TR 38.816, v15.0.0] architecture split.

The 5G nFAPI interface establishes interoperability and innovation among suppliers of disaggregated RAN nodes to establish a scalable ecosystem with a converged approach to virtualization across multiple suppliers. In doing this, we support a long and distinguished engineering tradition of providing an 'interchangeability of parts' to ensure that operators deploying 5G small cells can take advantage of the latest innovations in silicon and software with minimum barriers to entry, based on an open interface. Hence the specification helps support an innovative and competitive ecosystem for vendors of 5G small cell hardware, software and equipment.

This specification belongs to the 5G-FAPI suite, which comprises four specification documents covering the following interfaces:

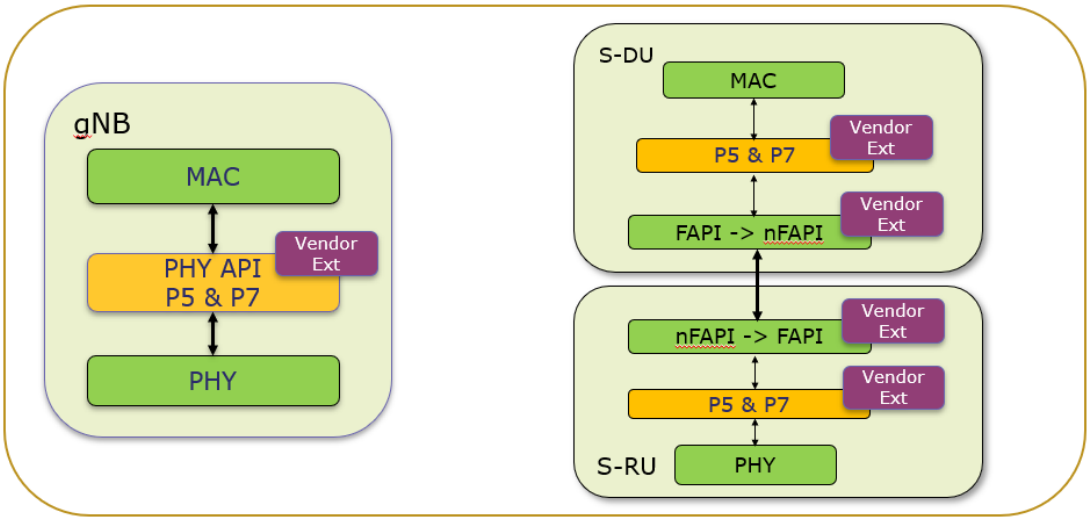
- '5G FAPI: PHY API' - PHY mode control (P5) interface and main data path (P7) [SCF222]
- '5G FAPI: RF and Digital Front End Control API' – (P19) for Frontend Unit control [SCF223]
- 'Network Monitor Mode API' - (P4) for 2G/3G/4G/5G [SCF224]
- '5G nFAPI Specification' [SCF225] (this document)



SON (Self Organising Networks), MAC (Medium Access Control), NMM (Network Monitor Mode)
FEU (Front End Unit) including DFE (Digital Front End) and ABF (Analog Beam Forming)

Executive summary

This document provides the specification for nFAPI, an extension where small cell functionality resides partially in a virtual network function (VNF) and partially in a physical network function (PNF). The relationship between FAPI and nFAPI is shown below.



This document is structured as follows:

Introduction: This section provides the high-level overview of nFAPI concepts and terminology.

nFAPI: This section details nFAPI procedures.

P5 messages: This section details P5 messages.

P7 messages: This section details P7 slot messages.

Contents

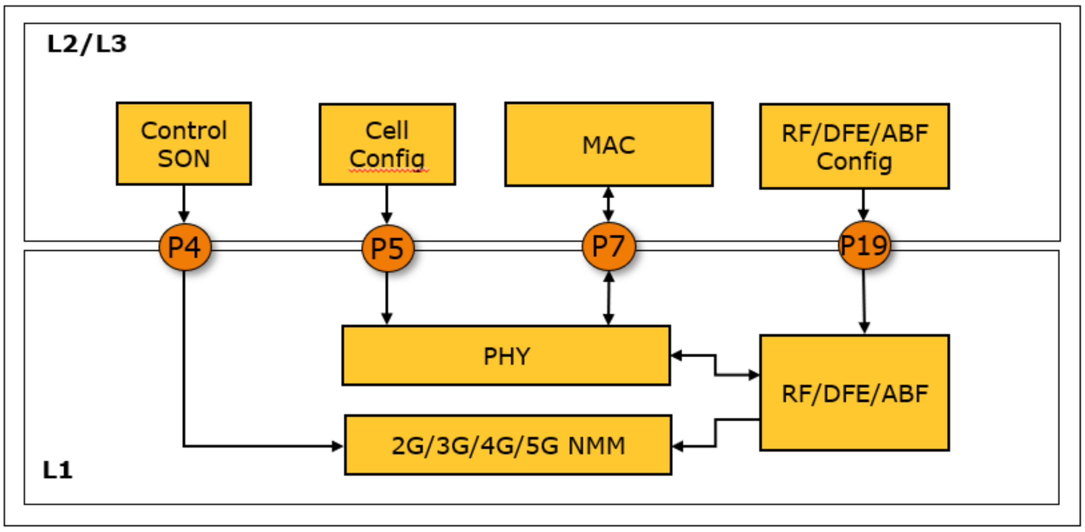
Tables

Figures

1. Introduction

This document describes the nFAPI interface for 5G NR small cells implementation.

The FAPI resides between 5G NR L2/L3 software on one hand, and the L1 PHY and FEU (Front End Unit, composed of radio and digital front ends) on the other hand. FAPI architecture and interfaces are shown in Figure 1-1.



NMM – Network Monitor Mode, DFE – Digital Front End, ABF – Analog Beam Forming

Figure 1–1 FAPI architecture and interfaces

nFAPI resides between a VNF, including 5G NR L2/L3 software, and a separate PNF including the L1 PHY and FEU. In this document, nFAPI defines the P5, P7 and P19 of the Small Cell Forum 5G NR FAPI, shown in Figure 1-2.

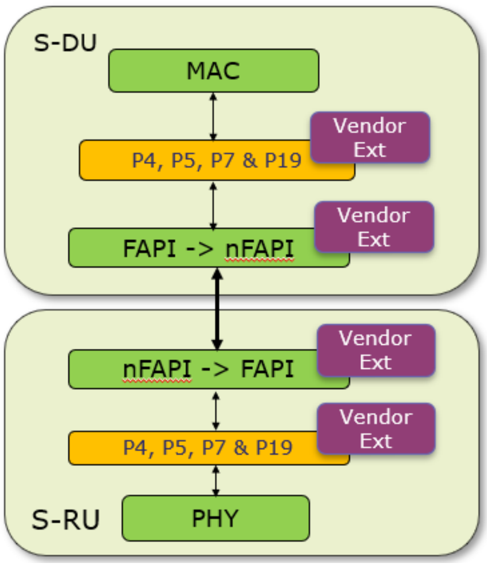


Figure 1–2 nFAPI Interfaces

Importantly, compared with FAPI which assumes a co-location of L2/L3 software and L1 PHY and FEU functionality – e.g., using a shared memory system to support the interface implementation – nFAPI has been defined assuming that a packet switched network is used to support communication between VNF and PNF components.

Disaggregated small cell products based on Option 6 split with an nFAPI interface may be realized in any of the deployment configurations described below.

1. A three-product small cell solution uses a lower layer split 6 together with the 3GPP F1 upper layer split 2.

   - S-CU: The Central Unit (CU) includes the SDAP, PDCP and RRC protocols
   - S-DU is the Distributed Unit (DU) that includes L2 layer functionalities including RLC, MAC and Scheduler.
   - S-RU is the Radio Unit (RU) that includes the PHY, baseband and RF processing.

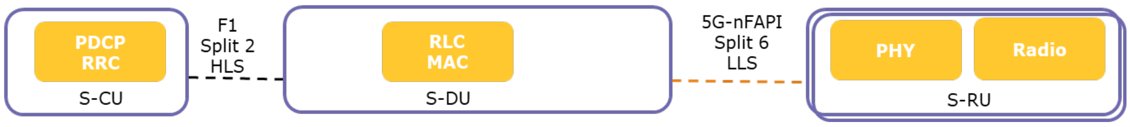


Figure 1–3 S-CU, S-DU and S-RU

2. A two-product solution just uses the lower layer split 6 over the nFAPI interface, combining the higher layer S-CU/S-DU logical functions into a single unit.

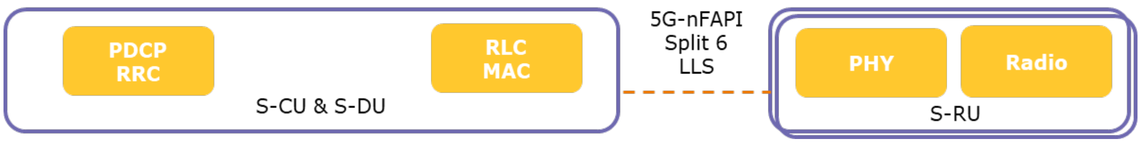


Figure 1–4 Two-unit disaggregated RAN with SCF split option 6

The following definitions are used in this document to describe the nFAPI interface-based procedures between S-DU and S-RU:

PNF: The S-RU is not virtualized and is termed a PNF (Physical Network Function). Within the context of nFAPI, the PNF device is the physical radio unit (also called S-RU), which contains a number of PHY instances with associated DFE and RF chains within the PNF device. While this document uses this context, the physical representation of the PNF device is not prescribed.

PHY Instance: Each PHY Instance supports the L1 functionality described in [4] and supports a single component carrier, as described in [8] Each PHY instance is identified by PHY Identifier (PHY ID) which is unique within a PNF.

FEU Component Instance or FEU Instance: Each FEU Component Instance (or FEU Instance, for simplicity) supports the FEU functionality described in [SCF-223] and interfaces with one or more PHY instances, as described in [SCF-222]. There are two types of FEU Component Instances: DFE Instances and RF Instances, corresponding to entities hosting Digital, respectively RF chains; RF component instances also host the ABF (Analog Beam Forming) function. Each DFE instance is identified by DFE Identifier (DFE ID) which is unique within a PNF. Each RF instance is identified by RF Identifier (RF ID) which is unique within a PNF.

S-RU Component Instance: A PHY instance or an FEU Components Instance. Each S-RU component instance is associated with an S-RU Component Type (PHY, DFE or RF) and an 8-bit S-RU Component Identifier (PHY ID, DFE ID or RF ID).

VNF: The S-DU may be virtualized and in the nFAPI specification is referred to as a VNF (Virtual Network Function). The VNF is considered to be the nFAPI peer element controlling and operating the PNF device. The structure and distribution of the VNF is considered to be the gNodeB functionality to operate the PNF device and the PHY instance(s) but is not prescribed.

The scope of this release consists of the P5, P7 and P19 interfaces of nFAPI.

2. nFAPI

2.1 nFAPI Procedures

The nFAPI P5 interface configures the PNF device at the PNF device level and at the static PHY instance level. The nFAPI P7 interface operates the PHY instance for the slot procedures. The nFAPI P19 interfaces is composed of P19 configuration and slot-oriented messages. The ensemble of P19 configuration messages constitutes the P19-C interface, and semi-statically configure the FEU instances within the PNF. The ensemble of P19 slot-oriented messages constitutes the P19-S interface, and operate the FEU instances for slot procedures.

2.1.1 PNF Procedures

The PNF device is initially configured through the OAM interface for foundation operation and connectivity with the VNF (this procedure is outside the scope of this specification). This permits the transport layer connection between the PNF and VNF, whereby further configuration of the PNF over the nFAPI interface occurs.

PNF sends PNF READY Indication message to VNF, indicating that PNF is ready. This message also carries the PNF's basic capabilities. The PNF PARAM and PNF CONFIG exchanges share the PNF capability with the VNF and permits the VNF to configure the PNF within the scope of the PNF's capability. The VNF can then instruct the PNF to start. Starting the PNF instantiates the FEU Components in the PNF's scope, enabling the FEU or PHY operations to occur or stop, disabling the FEU or PHY operations from occurring etc.

In particular, as shown in Figure 2-1, PHY Instantiation is not triggered by PNF initialization, but is rather the outcome of appropriate FEU and PHY configuration and initialization as follows:

1. PNF Initialization triggers DFE & RF Instantiation
2. Part of DFE Configuration, a DFE Profile is selected.
3. DFE instances are initialized.
4. RF instances are initialized.
5. VNF determines the list of PHY Profiles from PHY ID 0, and selects one PHY Profile ID (which must be compatible with he selected DFE profile)
6. Successful completion of 3-5 above triggers PHY Instantiation, which allows the VNF to drive the PHY State Machine.

Note: There is no prescribed order in which steps 3-5 are executed, but all of them are required for PHYs to be instantiated.

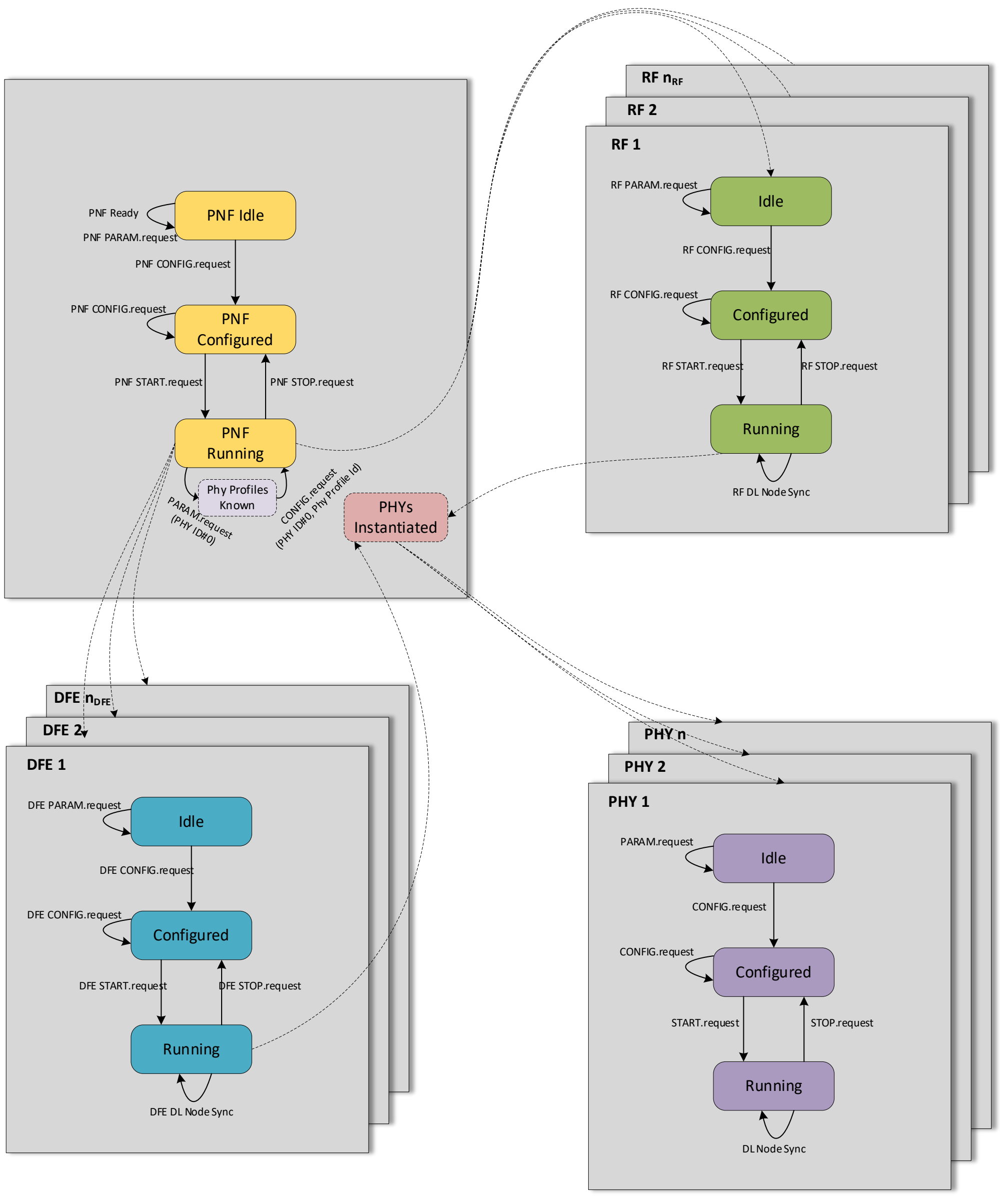


Figure 2–1 Relationship between PNF and PHY state machines

2.1.1.1 PNF Initialization

The PNF initialization procedure moves the PNF from the PNF IDLE state to the PNF RUNNING state, via the PNF CONFIGURED state. An overview of this procedure is given in Figure 2-2, the different stages are:

- The PNF READY Indication message exchange procedure
- The PNF PARAM message exchange procedure
- The PNF CONFIG message exchange procedure
- The PNF START message exchange procedure

The PNF initialization procedure is initiated by PNF by sending `PNF_READY.Indication` message to VNF as described in Figure 1-1.

The PNF initialization procedure is completed when the PNF sends the VNF a `PNF_START.response` message.

The remainder of this section describes the PNF_PARAM, PNF_CONFIG and PNF_START message exchange procedures.

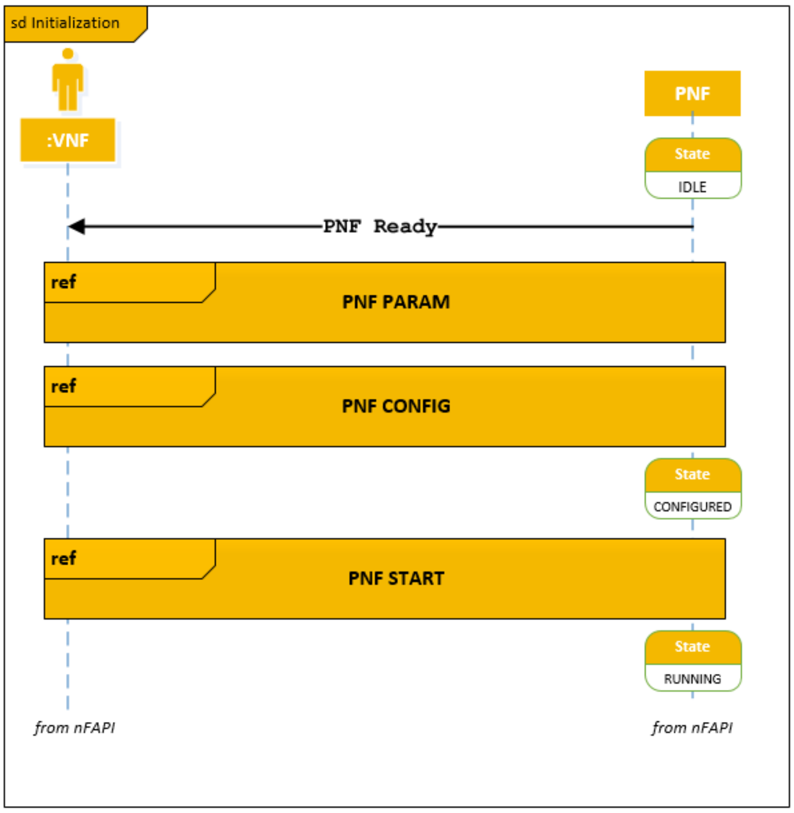


Figure 2–2   PNF Initialization procedure

The PNF READY message exchange procedure is shown in Figure 2-2. It has a dual purpose:

- This message indicates the readiness of S-RU to exchange messages with the VNF. Once SCTP connection is established with VNF and S-RU is ready for operation, the PNF starts the initialization procedure by sending `PNF READY` message.
- The PNF uses this message to inform VNF of its basic capabilities.

The PNF_PARAM message exchange procedure is shown in Figure 2-3. Its purpose is to allow the VNF software to collect information about the PNF capability.

From Figure 2-3 it can be seen that the PNF_PARAM message exchange procedure is initiated by the VNF sending a `PNF_PARAM.request` message to the PNF. If the PNF is operating correctly it will return a `PNF_PARAM.response` message. If no response is received, the action taken by the VNF is anticipated to be to trigger a management procedure and is out of scope of this document.

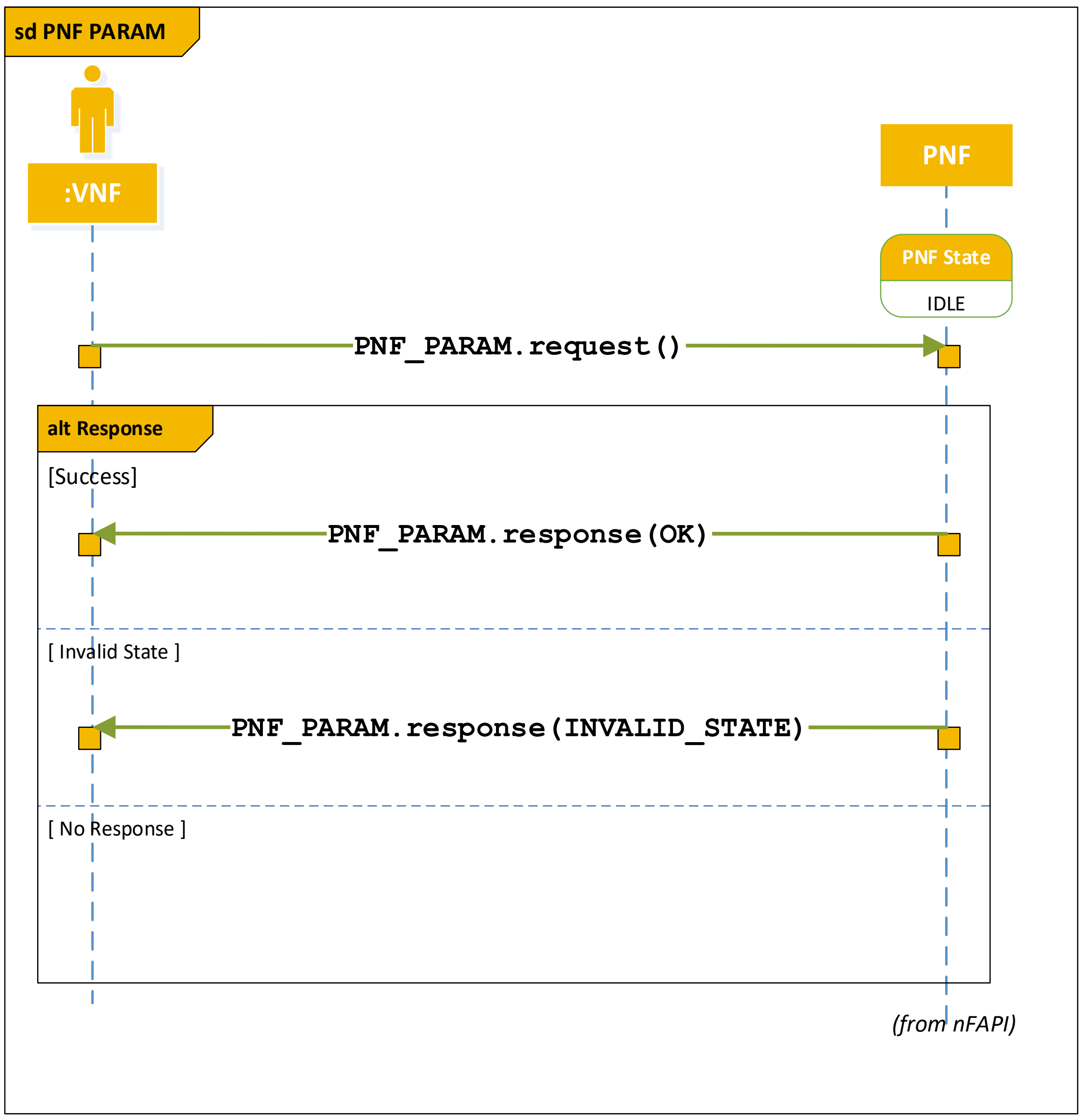


Figure 2–3 PNF_PARAM message exchange

The PNF_CONFIG message exchange procedure is shown in Figure 2-4. Its purpose is to allow the VNF to configure the PNF. The `PNF_CONFIG.request` message can be sent when the PNF is in the PNF IDLE state or the PNF CONFIGURED state. If the PNF is in the PNF IDLE state, this procedure will move the PNF into the PNF CONFIGURED state. If the PNF is in the PNF CONFIGURED state, the PNF will be re-configured with the supplied configuration and will remain in the PNF CONFIGURED state. The PNF will respond with the `PNF_CONFIG.response` message indicating success or rejection of the supplied configuration.

A rejection will result in the PNF retaining its previous configuration and state. If no response is received, the action taken by the VNF is anticipated to be to trigger a management procedure and is out of scope of this document.

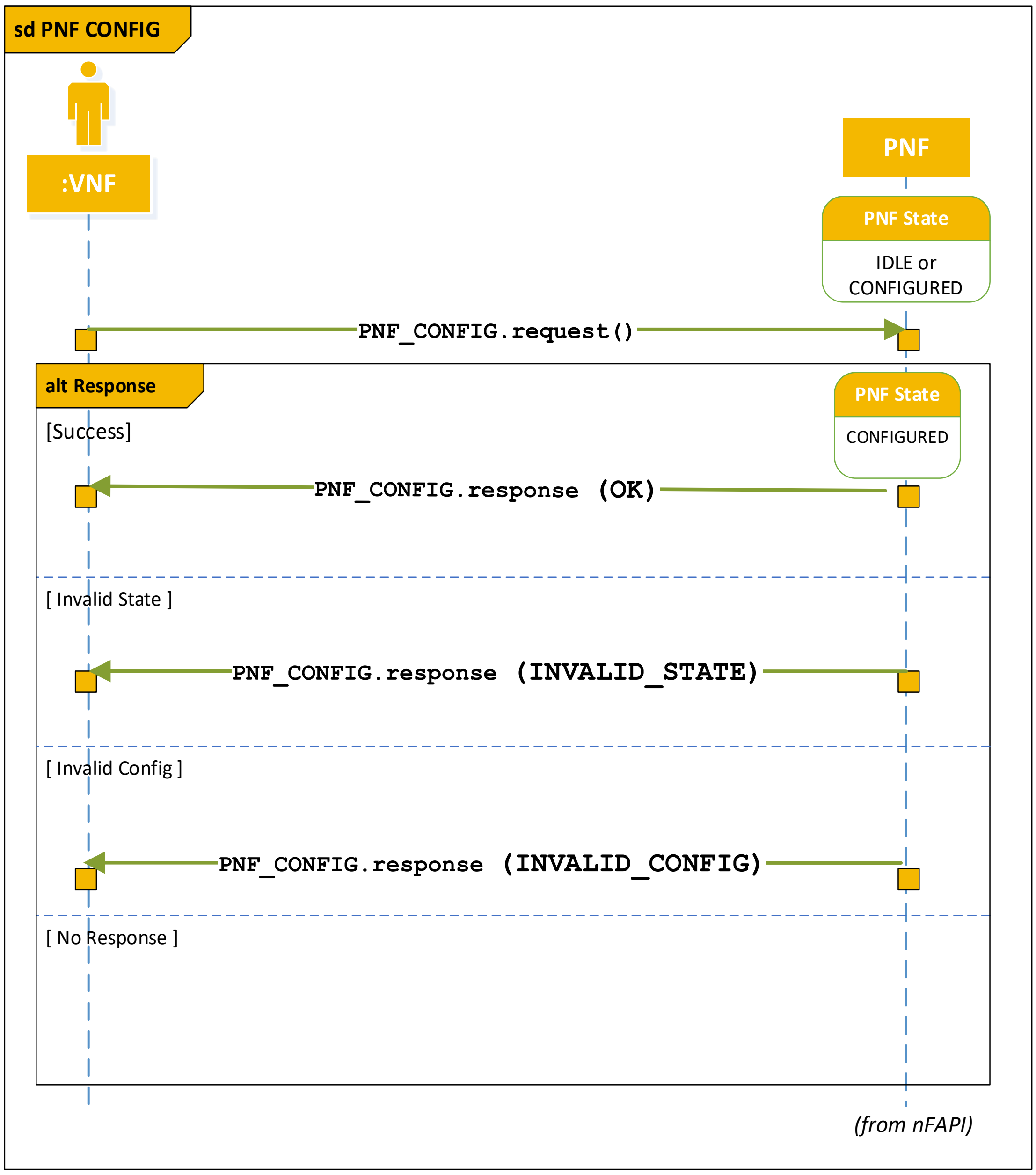


Figure 2–4 PNF_CONFIG message exchange

The PNF_START message exchange procedure is shown in Figure 2-5. Its purpose is to instruct a configured PNF to create the PHY instances. The VNF initiates this procedure by sending a `PNF_START.request` message to the PNF. If the PNF is in the PNF CONFIGURED state, it will create the PHY instances, move to the PNF RUNNING state and respond with a `PNF_START.response` message.

If the PNF receives a `PNF_START.request` message in either the PNF IDLE or PNF RUNNING state it will return a `PNF_START.response` message, including an INVALID_STATE error.

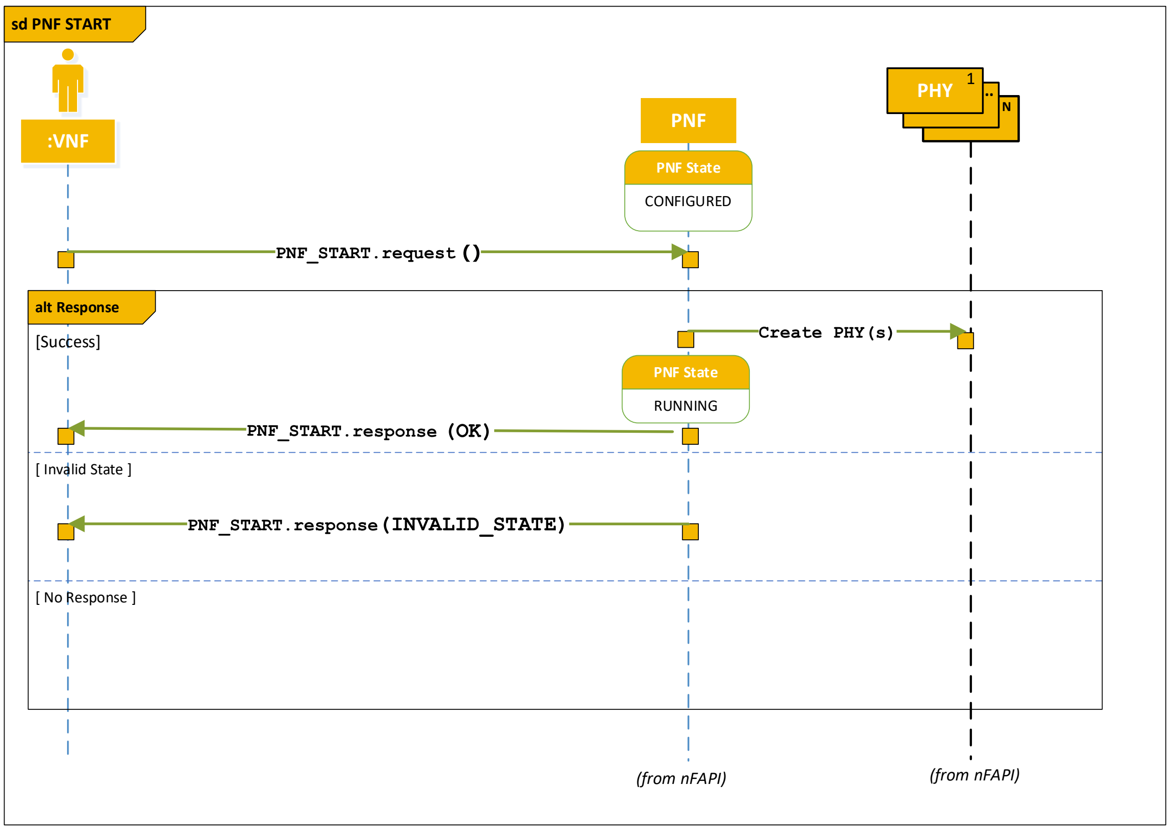


Figure 2–5 PNF_START message exchange

2.1.1.2 PNF Stop

The PNF Stop procedure is used to move the PNF from the PNF RUNNING state to the PNF CONFIGURED state. This also terminates and destroys the FEU and PHY instances. The PNF Stop procedure is shown in Figure 2-6 and initiated by the VNF sending a `PNF_STOP.request` message.

If the `PNF_STOP.request` message is received by the PNF while operating in the PNF RUNNING state, it will stop all PHY operations irrespective of the PHY states and return to the PNF CONFIGURED state. When the PNF has completed its stop procedure a `PNF_STOP.response` message is sent to the VNF.

If the `PNF_STOP.request` message was received by the PNF while not in the PNF RUNNING state, it will return a `PNF_STOP.response` message, including an INVALID_STATE error.

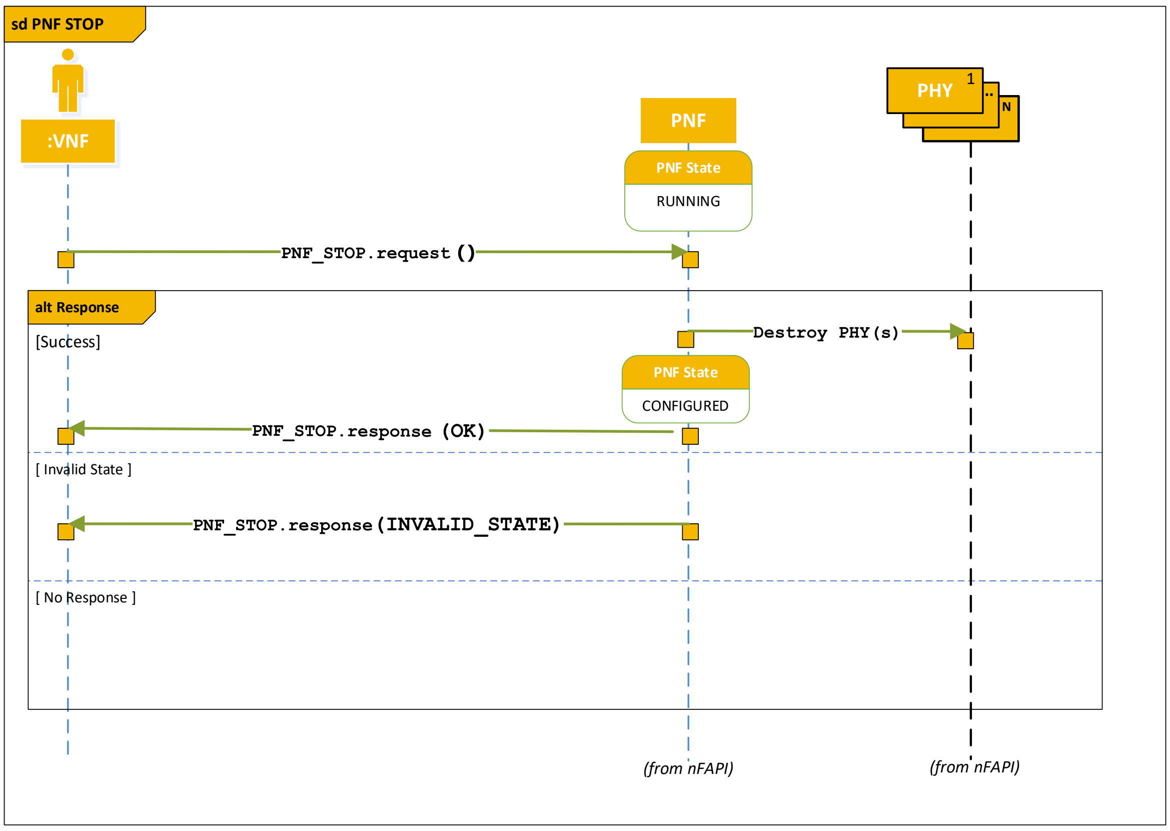


Figure 2–6 PNF_STOP message exchange

2.1.1.3 PNF Restart

The PNF Restart procedure is used to move the PNF from the PNF RUNNING state back to the PNF CONFIGURED state and then to return to the PNF RUNNING state. This procedure is defined as a sequence of the PNF_STOP (Figure 2-6) and the PNF_START (Figure 2-5) messages exchange.

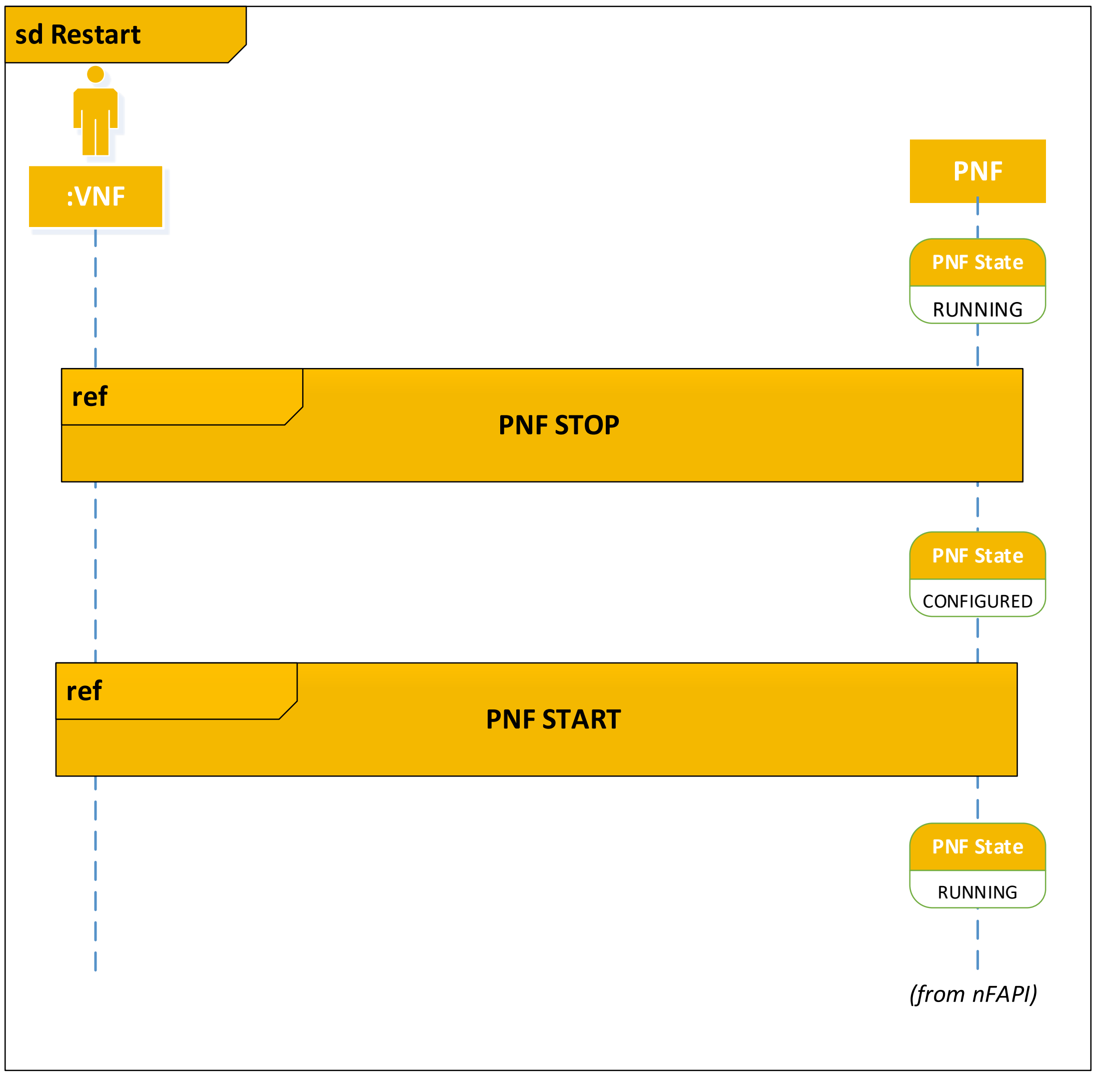


Figure 2–7 PNF Restart procedure

2.1.1.4 PNF Reconfigure

The PNF Reconfigure procedure is used to (optionally) move the PNF from the PNF RUNNING state back to the PNF CONFIGURED state, so the PNF can then be reconfigured with a new configuration, and then returned to the PNF RUNNING state. This procedure is defined as a sequence of the PNF_STOP (Figure 2-6), PNF_CONFIG (Figure 2-4) and the PNF_START (Figure 2-5) messages exchange.

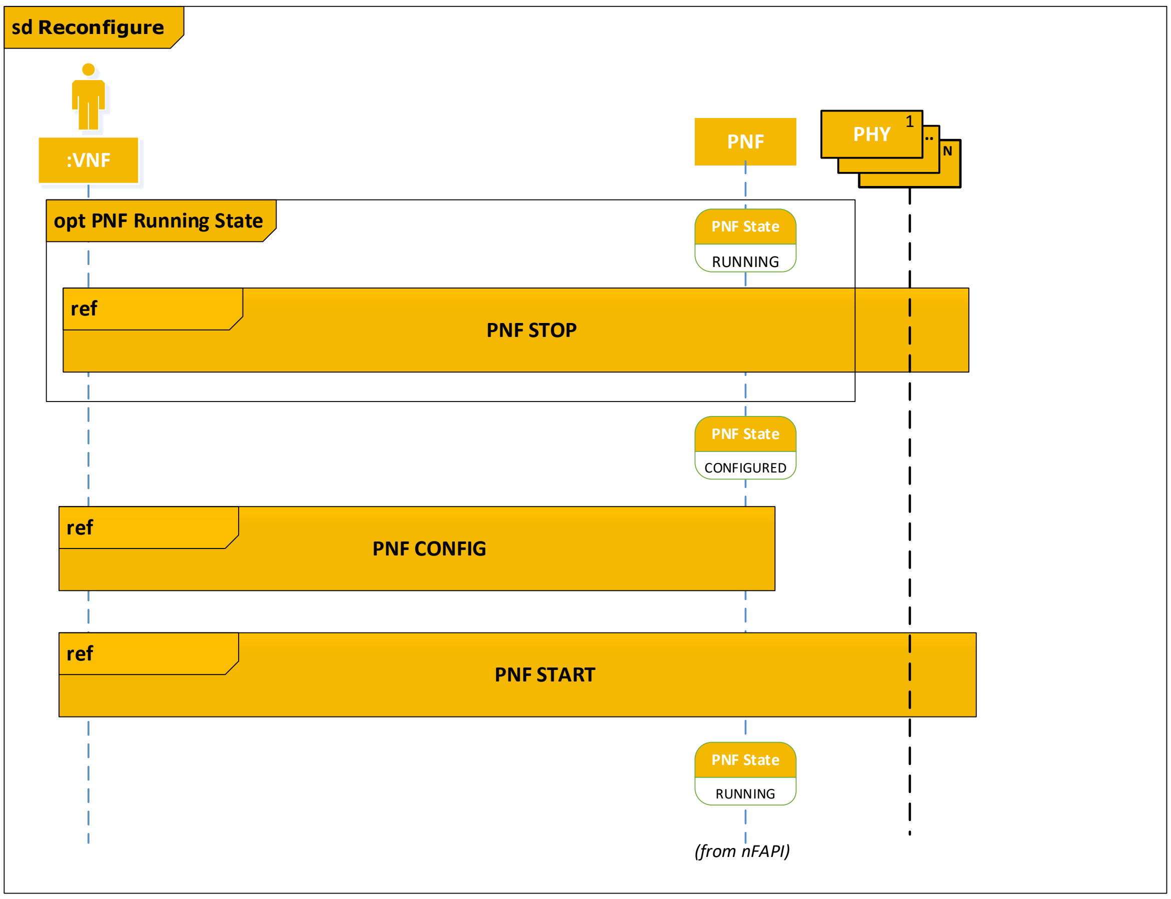


Figure 2–8 PNF reconfigure procedure

2.1.2 P5 PHY Procedures

The nFAPI P5 interface includes procedures to configure a PHY instance, and is based on the FAPI P5 interface detailed in Section 3.2.1 of [4]. This section only describes nFAPI behavior where it differs from FAPI.

It should be noted that the format of the P5 PHY messages is different for nFAPI. The definitions of nFAPI PARAM, CONFIG, START and STOP messages are specified in Sections 0 to 3.1.9.

2.1.2.1 PHY Initialization

The initialization procedure described for FAPI also applies in nFAPI with the following exceptions:

- Successful completion of the P5 PHY START procedure is indicated by a `START.response` message with Error Code set to OK, as shown in Figure 2-9.
- Instead of the `SLOT.indication` message, the Node Sync procedures follow the Start message, as shown in Figure 2-10.

    - The PHY PARAM and PHY CONFIG procedures are documented in section 2.1.1 of [4]
    - The PHY START message is documented in this section and illustrated in Figure 2-9.

- The Node Sync procedure is documented in section 2.1.3.4

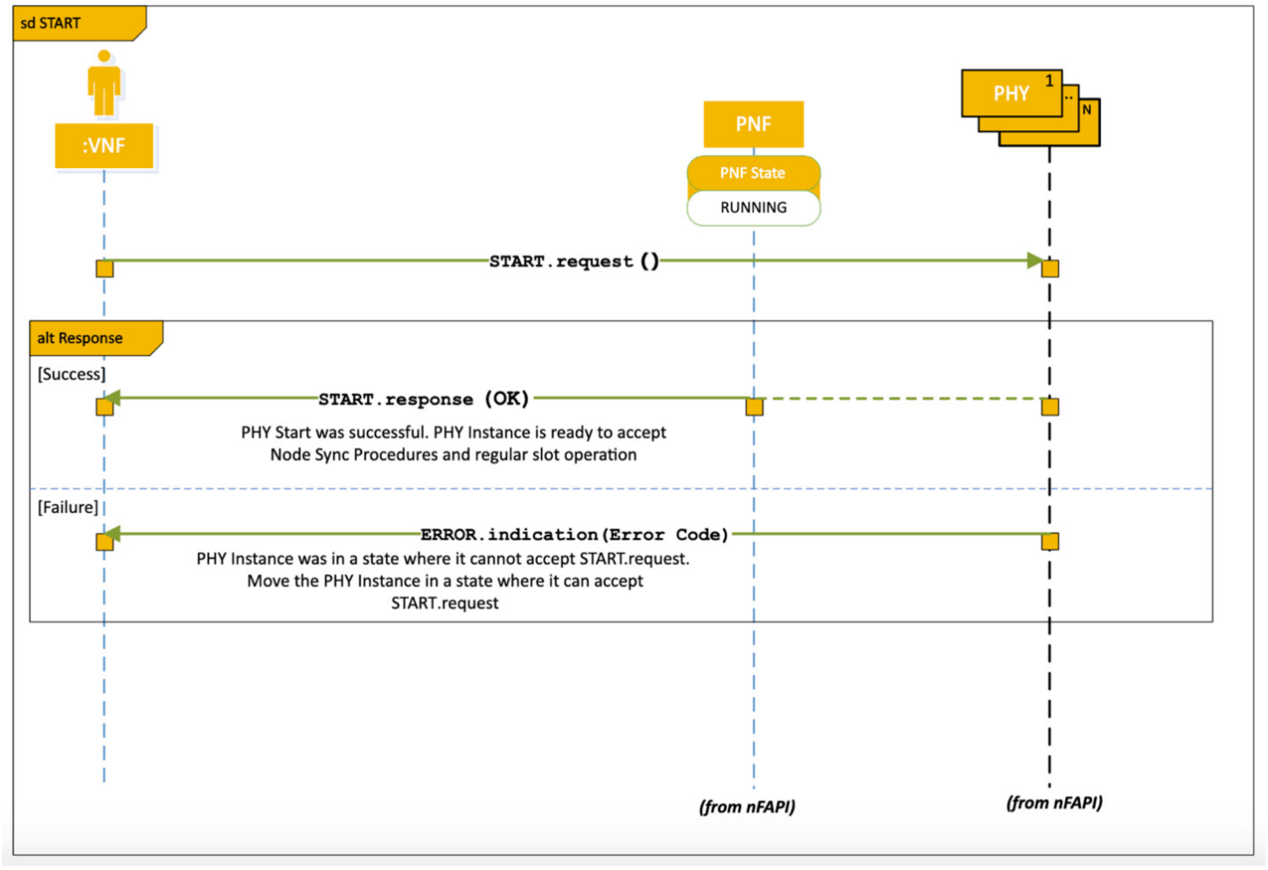

sd START
:VNF
PNF
PNF State
RUNNING
PHY
1
..
N
START.request ()
alt Response
[Success]
START.response (OK)
PHY Start was successful. PHY Instance is ready to accept Node Sync Procedures and regular slot operation
[Failure]
ERROR.indication(Error Code)
PHY Instance was in a state where it cannot accept START.request. Move the PHY Instance in a state where it can accept START.request
(from nFAPI)
(from nFAPI)

Figure 2–9 PHY START message

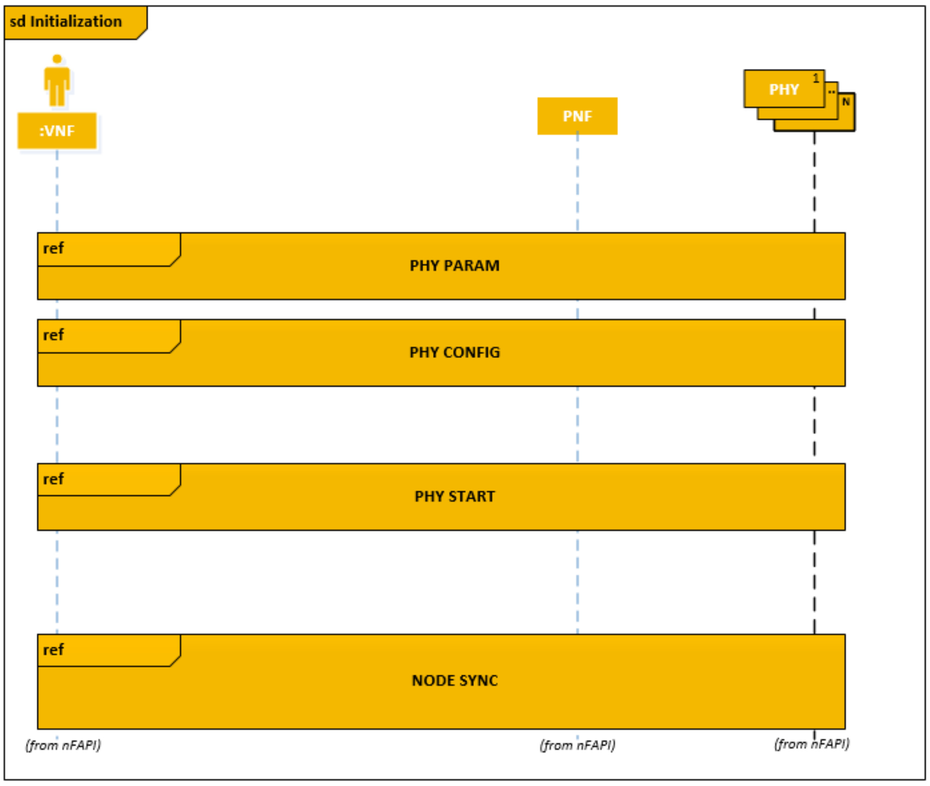


Figure 2–10 PHY Initialization procedure

2.1.2.2 PHY Termination

The termination procedure described for FAPI also applies in nFAPI.

2.1.2.3 PHY Restart

The restart procedure described for FAPI also applies in nFAPI, with the PHY START message exchange procedure defined in section 2.1.2.1.

2.1.2.4 PHY Reset

The restart procedure described for FAPI also applies in nFAPI.

2.1.2.5 PHY Reconfigure

The reconfigure procedure described for FAPI also applies in nFAPI with the PHY START message exchange procedure defined in section 2.1.2.1.

2.1.2.6 PHY Query

The query procedure described for FAPI also applies in nFAPI.

2.1.2.7 PHY Notification

The notification procedure described for FAPI also applies in nFAPI.

2.1.2.8 PHY Protocol Negotiation

The protocol negotiation procedure described for FAPI also applies in nFAPI, for negotiating the common FAPI (not nFAPI) protocol version between VNF and PNF.

2.1.2.9 PHY Instantiation

The instantiation procedure described for FAPI also applies in nFAPI.

2.1.3 P7 Slot Procedures

The nFAPI P7 interface configures the PHY instances every slot and is based on the 5G FAPI P7. This section only describes nFAPI behaviour where it differs from FAPI.

2.1.3.1 SLOT signal

In FAPI a `SLOT.indication` message may be sent from the PHY instance at the start of every 62.5µs, 125µs, 250µs, 500µs or 1ms slot, depending on the sub-carrier spacing. In nFAPI this message is not used, since the expected jitter on the fronthaul connection between VNF and PNF prevents this message from being a suitable mechanism for signalling a slot interval.

Instead, the combination of the PHY sync procedure (2.1.3.3) and API message timing procedure (2.1.3.6) ensures the alignment of MAC commands to air-interface slots.

2.1.3.2 SFN/SL synchronization

The SFN/SL synchronization mechanism described for FAPI does not apply in nFAPI. Instead, the PHY synchronization mechanism and delay management mechanisms described below is used to maintain synchronization between PHY instances and ensure timely messaging between the VNF and PHY instances.

2.1.3.3 PHY Synchronization

The PHY synchronization procedure relates to the alignment of antenna transmission times between the PHY instances. PHY synchronization generally refers to the achievement of a common timing reference between the VNF and the PHY instances. For interference management and other wireless PHY coordination techniques, it may be preferable for the PHY instances to align their time bases.

The choice of synchronization (or lack of it) between the PHY instances and VNF depends on the deployment requirements and is beyond the scope of this specification. Please refer to Timing and Synchronization Considerations [Informative] for more information.

2.1.3.4 Delay Management between VNF and PHY

The Delay Management ensures that P7 slot procedures to occur in a timely fashion. In particular, a PHY instance expects P7 slot-based messages from VNF to reach the instance in a Receive Time Window interval, that PHY maintains to buffer and process time-critical P7 messages (`DL_TTI.request`, `UL_TTI.request`, `UL_DCI.request`, `Tx_Data.request`), to apply at their targeted slots.

Delay Management is also applicable to P19-S slot procedures, where Receive Windows are maintained for P19-S messages subject to Time Management; for P19-S messages, it is FEU Instances, rather than PHY Instances, maintain the Receive Timing Windows.

Timing window management

As illustrated in Figure 2-11, each PHY maintains a Receive Timing Window characterized by:

- a size (*Timing Window*) and
- offset prior to the targeted slot (<msg> *Timing offset*, where <msg> is one of the above-listed time-critical messages)

Both of the above time parameters are specified in microseconds (µs).

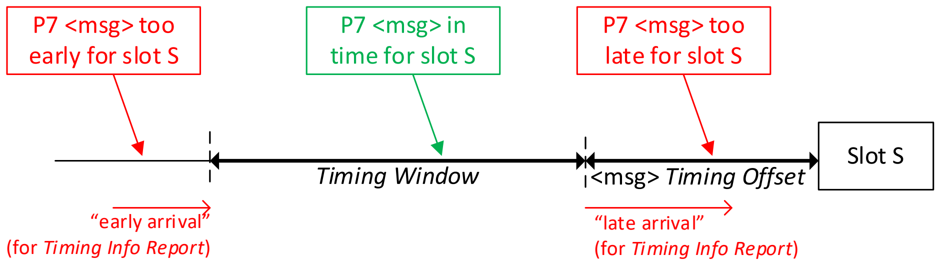

P7 <msg> too early for slot S
P7 <msg> in time for slot S
P7 <msg> too late for slot S
Slot S
Timing Window
<msg> Timing Offset
"early arrival" (for Timing Info Report)
"late arrival" (for Timing Info Report)

Figure 2–11 Receiving Window mechanism at the PHY

Relevant P7 messages for Slot S received by PHY Receive Window mechanism:

- inside the Timing Window interval: are processed for transmission, reception or configuration at Slot S;
- outside the Timing Window interval: are marked by PHY as 'too early' or 'too late', as appropriate, and used to construct a Timing Info Report to the VNF.

Delay management procedure

The delay management mechanism between the VNF and PHY Instance can be used to establish the timing reference differences between them, as well as optionally permitting the VNF to instruct the PHY Instance to update its slot number based on the offset defined by the VNF. The internal slot timing requirements for the P7 procedures are exchanged through the PNF PARAM procedure.

In the latency probe procedure (as illustrated in Figure 2-12), the VNF sends a `DL Node Sync` message to the PHY instance containing the parameter t1. Upon reception of `DL Node Sync` message, the PHY Instance shall respond with `UL Node Sync` message, indicating t2 and t3, as well as t1 which was indicated in the initiating `DL Node Sync` control message.

As configured by the VNF, the PHY Instance can send either periodic or aperiodic `Timing Info` messages informing the VNF of the P7 slot message time of arrival. If configured to aperiodic messages, the `Timing Info` message is sent when a slot completes and the set of slot messages either arrives too late or too early (depending on window configuration) - see Figure 2-12. The actions taken by the VNF in response to the `Timing Info` is out of scope for this specification.

Note: This exact timing definition associated with what constitutes too early or too late is implementation specific and is outside the scope of this document.

In this example of Figure 2-12, the delay management procedure is illustrated via a positive timing advance, but the `DL Node Sync` message also supports a negative number to indicate negative timing advance of the SFN/slot.

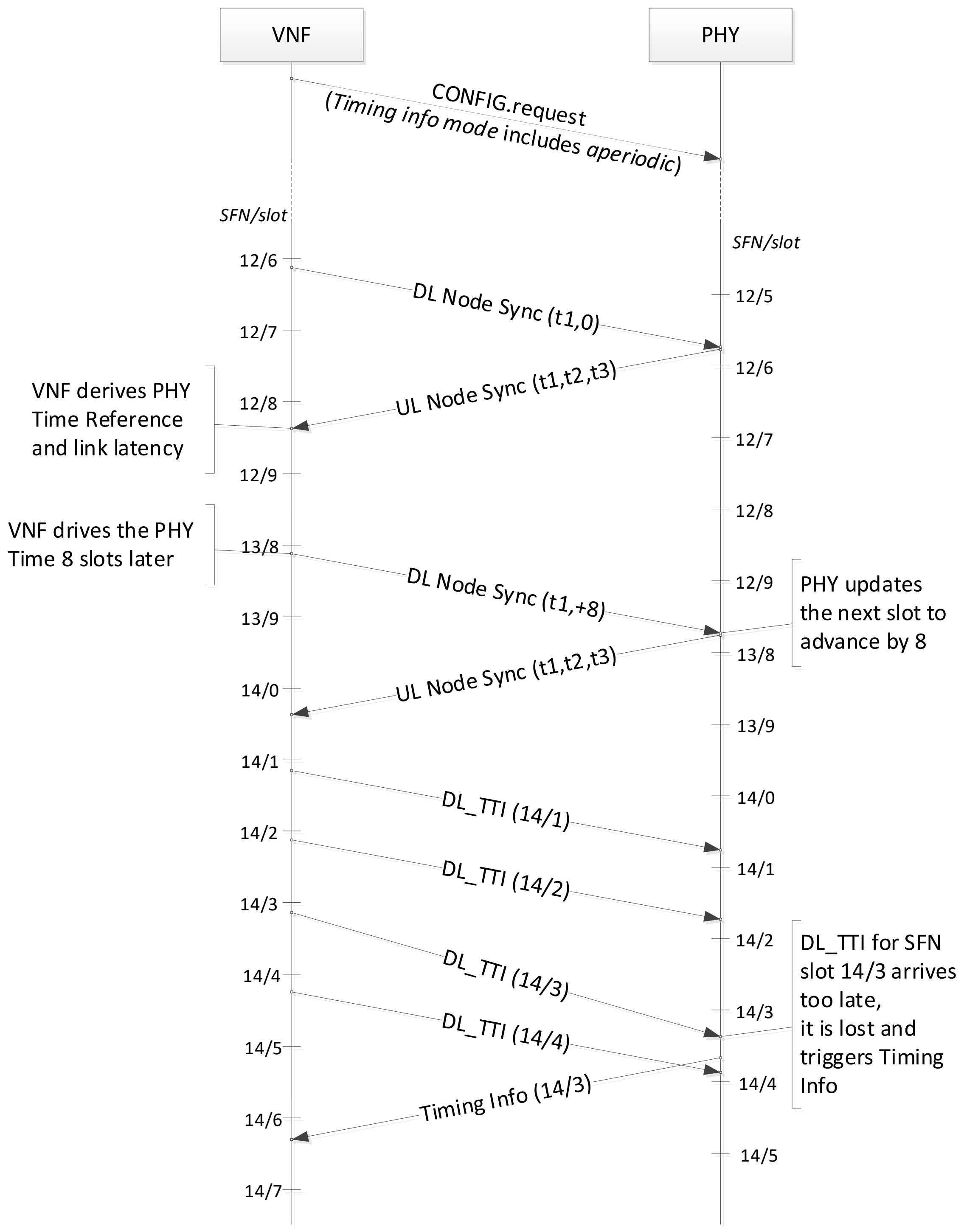


Figure 2–12 Delay management timing sequence with DL_TTI example

2.1.3.5 API message order

The API message order described for FAPI also applies in nFAPI, with the following differences:

- the role of `SLOT.indication` message is replaced by the Delay Management procedure in section 2.1.3.4. In particular, the

`DL_TTI.request` message is expected to arrive at the PHY in the Receive Window for to the SFN/SL listed in the message.

- There is no order requirement between the `DL_TTI.request` and `UL_TTI.request` messages for a particular SFN/SL; rather, the `DL_TTI.request` and `UL_TTI.request` messages are expected to arrive at the PHY in their respective Receive Windows for to the SFN/SL listed in the message.

2.1.3.6 API message timing

The nFAPI P7 messages listed in section 2.1.3.4 sent from the VNF to PNF must arrive at the PHY instance a minimum *Timing* offset before the slot they configure starts transmission on the air interface.

If the message arrives in the allowed timing window, as described in section 2.1.3.4, then the PHY instance continues normal operation for the slot.

If the message is late, the PHY instance follows the procedure shown in Figure 2-13:

- If `DL_TTI.request` is received too early or too late
- Return a `Timing Info.indication` message including the SFN/SL, the lateness of the message in μs and indicating this error applies to `DL_TTI.request`.

The same procedure is used for `DL_TTI.request`, `TX_Data.request`, `UL_TTI.request` and `UL_DCI.request`. The VNF should react to each error messages as follows:

- If the error is for `DL_TTI.request` lateness the VNF shall consider the DL slot lost
- If the error is for `TX_Data.request` lateness the VNF shall consider the DL slot lost
- If the error is for `UL_TTI.request` lateness the VNF shall consider the UL slot lost
- If the error is for `UL_DCI.request` lateness the VNF shall consider the UL slot being scheduled to be lost

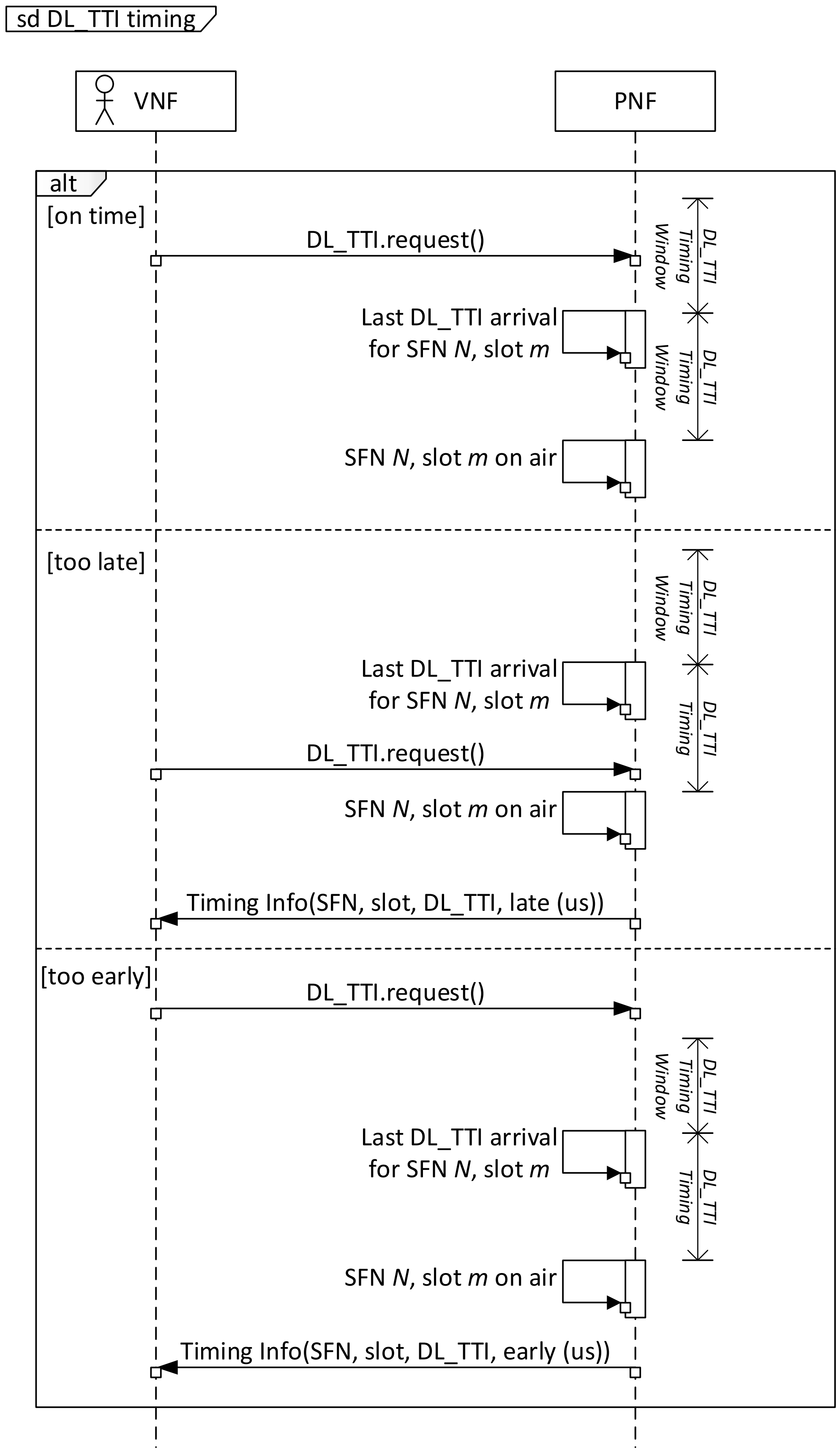


Figure 2–13 DL_CONFIG.request lateness procedure

2.1.4 nFAPI Error Procedures

2.1.4.1 General

In case of errors in processing an nFAPI message for which there is no error message or error code is defined, the receiver of the message shall discard the message and not respond further.

2.1.5 P19 Procedures

The P19 interface is described in [6] and enables interaction between VNF and FEU Instances of PNF. P19 interface procedures use P5 and P7 interfaces to configure and control DFE and RF in S-RU. For the purposes of nFAPI, P19 messages are classified as:

- P19-C messages: these messages are identified as configuration messages in section 1A.4 of [6];
- P19-S messages: these are messages identified as slot-oriented in section 1A.4 of [6];

2.1.5.1 FEU Component Initialization

The initialization procedures summarized for FAPI apply to nFAPI.

2.1.5.2 FEU Component Termination

The termination procedures summarized for FAPI also apply in nFAPI.

2.1.5.3 FEU Component Restart

The restart procedures summarized for FAPI also applies in nFAPI.

2.1.5.4 FEU Component Reset

The restart procedures summarized for FAPI also apply to nFAPI.

2.1.5.5 FEU Component Reconfigure

The reconfigure procedures summarized for FAPI also apply in nFAPI.

2.1.5.6 FEU Component Query

The query procedures summarized for FAPI also apply in nFAPI.

2.1.5.7 FEU Component Notification

The notification procedures summarized for FAPI also apply in nFAPI.

2.1.6 P19 Slot Procedures

The nFAPI P19 interface can configure FEU component instances every slot and is based on the 5G FAPI FEU slot-oriented messages. This section only describes nFAPI behavior where it differs from FAPI.

2.1.6.1 SFN/SL synchronization

The SFN/SL synchronization mechanism described for FAPI does not apply in nFAPI. Instead, only the Delay Management mechanism described in section 2.1.6.3 applies to both nFAPI and Delay-Managed FAPI, to ensure timely messaging between the VNF and FEU instances, with the differences described below:

- Delay Management for P19 is terminated in FEU Instances, rather than PHY instances;
- Delay Management applies to P19-S messages subject to Time Management, for which Receive Timing Windows are maintained, as described in section 5.2 of [6];
- P19-S specific RF/DFE Node Sync and RF/DFE Timing Info messages are used to monitor and control the Receive Timing Window operations, as described in section 5.2 of [6].

2.1.6.2 FEU Synchronization

Synchronization refers to the ability to align alignment of antenna transmission times between the FEU instances. In this release, FEU component synchronization follows the synchronization capability of PHYs that they interface with.

The choice of synchronization (or lack of it) between the FEU instances and VNF depends on the deployment requirements and is beyond the scope of this specification. Please refer to Timing and Synchronization Considerations [Informative] for more information.

2.1.6.3 Delay Management between VNF and FEU

The nFAPI and FAPI Delay Management ensure that P19-S messages subject to Time Management arrive at the FEU components in a timely fashion.

Just like for PHY Delay Management in section 2.1.3.4, a FEU component instance expects P19-S messages subject to Time Management from VNF to reach the instance in a Receive Time Window interval, that FEU components maintain to buffer and process P19-S messages.

DFE components maintain separate Receive Windows for the following P19-S messages subject to Time Management:

- DFE SCHEDULE.request

RF components maintain separate Receive Windows for the following P19-S messages subject to Time Management:

- RF SCHEDULE.request
- FEU.CONFIG.SELECT_BEAM.REQUEST
- FEU.CONFIG.SET_BEAM_SLOT_PATTERN.REQUEST

The management of the receive windows follow the same mechanism as in section 2.1.3.4, with the following changes:

- FEU components may indicate that latency and jitter computations can be derived from the PHY instances they interface with.
- If FEU components maintain independent latency and jitter metrics, they make use of the following messages to determine report these metrics:

- RF/DFE Timing Info
- RF/DFE DL and UL Node Sync

2.1.6.4 API message timing

The P19-S messages subject to Delay Management sent from the VNF to the PNF must arrive at the FEU instance(s) a minimum *Timing* offset before the slot they configure starts transmission on the air interface.

If the message arrives in the allowed timing window, as described in section 2.1.6.3, then the FEU instance continues normal operation for the slot.

If the message is late, then the FEU instance follows the procedure shown in Figure 2-13, with the following difference:

- Instead of the `DL_TTI.request` any of the P19-S messages subject to Time Management
- FEU instead of PHY, returns a `Timing Info.indication` message including the SFN/SL, the lateness of the message in μs and indicating this error applies to P19-S message.
- DFE Timing Info is used instead of Timing Info, when FEU instance is a DFE instance
- RF Timing Info is used instead of Timing Info, when FEU instance is an RF instance
- An FEU Instance may indicate that it derives timing from the controlling PHY Instances, in which VNF may choose to only probe and monitor delay performance for the controlling PHY Instances.

2.1.7 P4 Procedures

P4 interface procedures use P5 interface to configure network monitoring and reporting from S-RU. All P4 FAPI messages are carried transparently in nFAPI.

2.2 5G nFAPI co-existence with 4G nFAPI

nFAPI messages for 5G NR defined in this specification are not completely aligned with 4G nFAPI messages. The concepts of PNF and VNF, nFAPI procedures of configuration and states, sync procedures remain the same for 5G nFAPI. However, changes specific to 5G NR and optimizations needed in P5 and P7 messages for efficient transfer of signalling and data procedures have led to nFAPI message changes.

The following is the summary of changes. Detailed description of these changes is addressed in the following sections.

- nFAPI header
- Packing of P5 and P7 messages in nFAPI
- Common header for P5 and P7 messages
- Avoidance of inclusion of FAPI header

Refer [4] document for definition of fields (TLVs) of any FAPI message(s) contained in an nFAPI message.

When 4G nFAPI and 5G nFAPI interfaces are in operation on the same host, these nFAPI interfaces need to use different SCTP and UDP ports.

2.3 nFAPI transport and message formats

2.3.1 Transport Layer

2.3.1.1 nFAPI P5

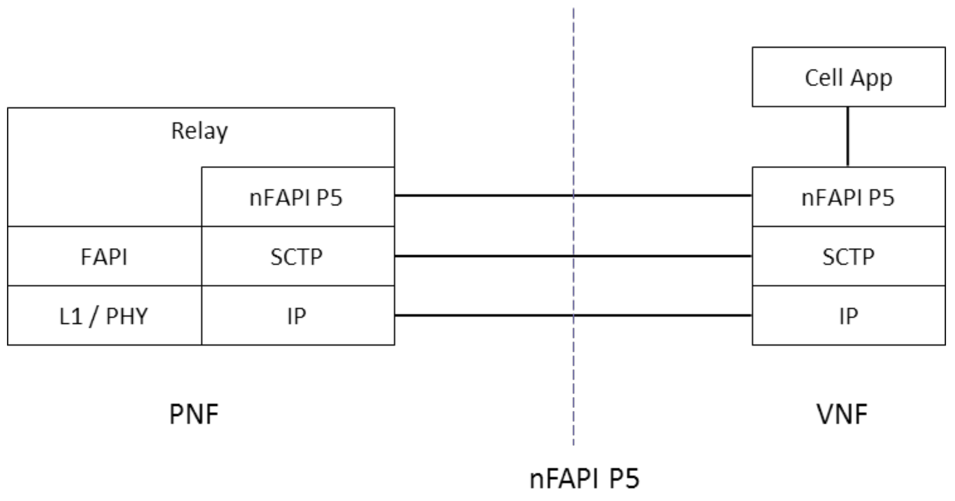


Figure 2–14 nFAPI P5 Protocol Stack

The transport layer for nFAPI P5 is required to ensure reliable message transfer, so Streaming Control Transmission Protocol (SCTP) is used as the transport protocol. Therefore there are no extra methods to ensure reliable transmission of messages at the application level (such as timers, packet validation and packet loss).

The FQDN or IP address of the P5 interface of the VNF is specified by the OA&M to the PNF. The default values for SCTP payload protocol identifier (PPI) field and the SCTP port number for 5G nFAPI P5 interface is assigned by the IANA. These values can also be changed by the OA&M. The following table lists the parameters that needs to be specified by the OA&M to the PNF. If any of these values are not explicitly specified by the OA&M, then the default value shall be used.

Parameter	Type	Default Value	Description
VNF_Address	String	N/A	FQDN or IP address of the SDU
VNF_P5_PortNumber	Integer	TBD	SCTP port number of the P5 interface
VNF_P5_PPI	Integer	TBD	SCTP PPI of the P5 interface

The payload protocol identifier (PPI) field for 5G nFAPI in SCTP is set to the value (TBA), as assigned by IANA.

There is one SCTP association between the PNF and the VNF, established by the PNF based on OAM configuration of VNF address and port. There is a single SCTP stream for each nFAPI PHY instance to transport P5 messages. Messages with the supervisory PHY (PHY ID#0 in [4]) use the same SCTP stream as the messages with the PNF.

Transport network redundancy may be achieved by SCTP multi-homing between two endpoints, of which one or both is assigned multiple IP addresses. SCTP endpoints support a multi-homed remote SCTP endpoint. For SCTP endpoint redundancy an INIT may be sent from the VNF or PNF at any time for an already established SCTP association, which shall be handled as defined in IETF RFC 4960 [10].

2.3.1.2 nFAPI P7

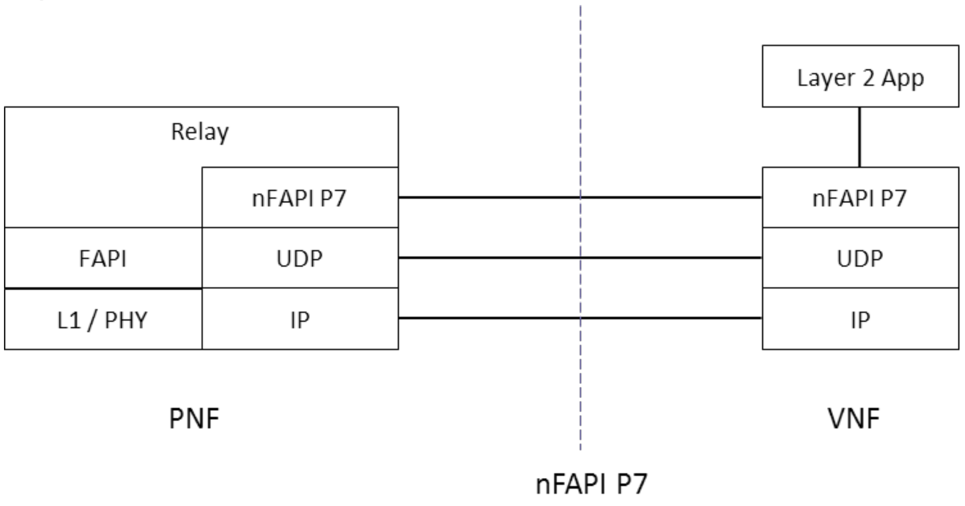


Figure 2–15 nFAPI P7 Protocol Stack

Latency and jitter constraints for nFAPI P7 preclude the use of transport protocols which include retransmission to improve reliability, allowing UDP to be used as the transport protocol. There are application-level elements (sequence numbers) in the P7 protocol to ensure that lost packets can be identified and handled accordingly.

The address and port-to-use at the VNF and PNF are exchanged over the P5 interface through PHY configuration. The PHY ID within the P7 message header defines which PHY instance/VNF interface the message is directed towards. These configuration options permit either a single UDP stream for a PNF with multiple PHY instances, or, alternatively, a UDP stream per PHY instance.

2.3.1.3 nFAPI P19

nFAPI P19-C messages use the same transport and message formats as nFAPI P5 messages.

nFAPI P19-S messages use the same transport and message formats as nFAPI P7 messages. On the S-RU side, these messages have the same termination point as the PHY instances the FEU components interface with.

2.3.1.4 Handling of IPv4/IPv6 headers

While transporting nFAPI packets over IP layer, IP header options should not be enabled at S-DU or S-RU. This restriction is applicable for both IPv4 and IPv6 based packet handling. The inclusion of an IP header option creates challenges in hardware acceleration of IP packet processing.

2.3.1.5 P7 and P19-S transport using Ethernet

Optionally Ethernet transport can be used send P7 and P19-S messages to avoid UDP/IP overhead. In this case, the nFAPI packets are transported using eCPRI protocol, with eCPRI message type set to indicate nFAPI.

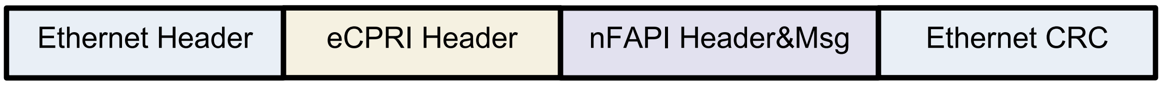
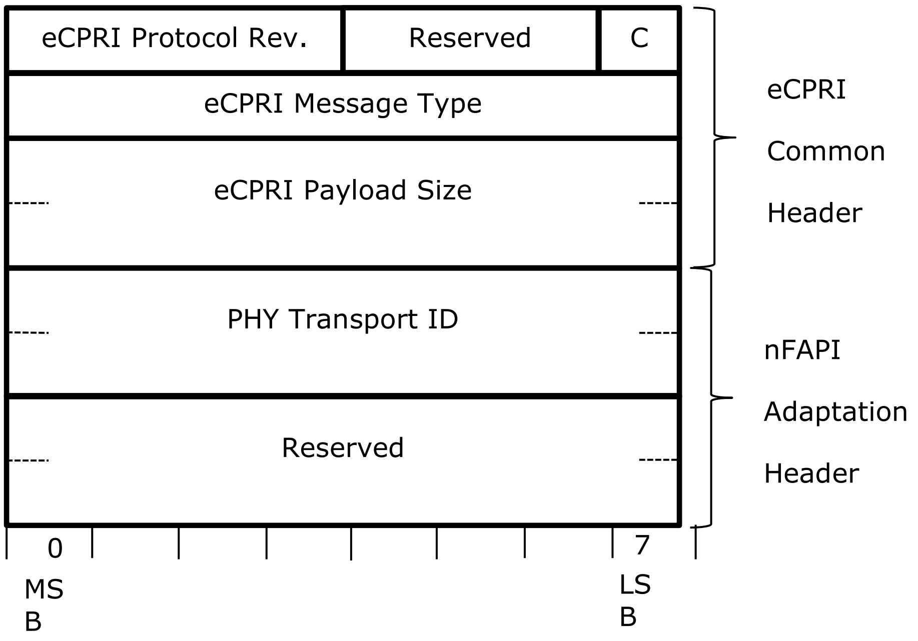


Figure 2–16 nFAPI P7/P19-S message with eCPRI

The eCPRI header fields for nFAPI P7 messages are described below.



Field	Value	Description
eCPRI Protocol Rev	0x1	Protocol revision
Reserved fields	0x0	No reserved fields are used
C	0x0	No segmentation by eCPRI
eCPRI Message Type	64 to 255	Vendor specific value (64 to 255) to be used. Indicated by PNF using P5 interface.
PHY transport ID		Transport identifier for destination PHY instance. Specified by PNF using P5 interface.

2.3.1.6 Securing the nFAPI interface

An IPSec tunnel between the PNF and VNF may be used to protect the P5/P19-C and P7/P19-S communications. The details associated with configuring the Security Gateway address in the PNF and establishing/maintaining the IPSec tunnel are out of scope of this document.

2.3.2 General Message Format

The P5 and P7 messages use different header structure as specified in the following sections.

2.3.2.1 P5 and P19-C Message Format

The message/header structure of an nFAPI PDU is shown in Figure 2-17. Each nFAPI PDU contains an nFAPI header shown as green and followed by multiple nFAPI/FAPI messages, containing an nFAPI message header and message body. The nFAPI protocol may carry nFAPI messages, FAPI messages (as specified in [4]) or combined nFAPI/FAPI messages as discussed in section 2.3.2.6.

The nFAPI header shown in Figure 2-17 contain the following fields:

Field	Length(Bits)	Description
Segment Length	16	The length in bytes of the message segment including the header
More	1	A More flag indicating there are more segments to follow to complete the entire message
Segment Number	7	The segment number starting at zero and incrementing by one between each segment
Sequence Number	8	The incrementing sequence number for all complete messages over the P5/P19-C nFAPI interface
Transmit Timestamp	32	This value is reserved (set to zero on transmission and ignored on reception) for all P5/P19-C messages.

One P5/P19-C nFAPI PDU may contain multiple P5/P19-C nFAPI messages. Each P5/P19-C nFAPI message is preceded by an nFAPI message header as shown in Figure 2-17 and followed by a padding field. The padding field ensures that each of the P5/P19-C nFAPI message start on 4-octet aligned offset within the message. The length of the padding is calculated from the length (L) of the preceding message body using the following formula:

Padding length = (4 – (L mod 4)) mod 4

Where "mod" is the mathematical modulo operation. Note that if L is a multiple of 4, then padding is 0 bytes long. Note that the last message in the P5/P19-C nFAPI PDU does not have a padding field at the end.

The P5/P19-C nFAPI message header contains the following parameters:

Field	Length (Bits)	Description
S-RU Termination Type	8	This field is shall be set to * 0x01 for P5/P7 messages. * 0x02 for P19 messages with DFE components * 0x03 for P19 messages with RF components. * 0x04 for P4 messages with Network Monitor components. For P5,this field allows easy extraction of a FAPI message when the nFAPI protocol carries a FAPI message and is included to simplify FAPI message extraction in the case one or more FAPI messages are present.
PHY ID	8	P5/P7: Within the PNF Device, it is the unique identity of the PHY instance *in the range [1..”Maximum Number Of PHYs”], where “Maximum Number of PHYs” is specified in the PNF_PARAM.response message*. For FAPI messages (FAPI_PARAM, FAPI_CONFIG) directed to PNF and PNF_XXX messages, value PHY-ID=0 is used. This includes any messages directed to the supervisory PHY with PHY ID=0 in [4]. P19: S-RU ID uniquely identifies DFE or RF instance, depending on the content of the S-RU Termination Type field
Message ID	16	The ID of the nFAPI or FAPI message
Length	32	This is the length of the message body in octets excluding the message header.

The P5/P19-C nFAPI header permits segmentation of the messages, if required by transport mechanisms. The Sequence Number increments across complete messages while the Segment Number defines the segment order within a message. The More flag defines if more segments will follow.

Note: Techniques for determining whether segmentation is required are outside the scope of this document.

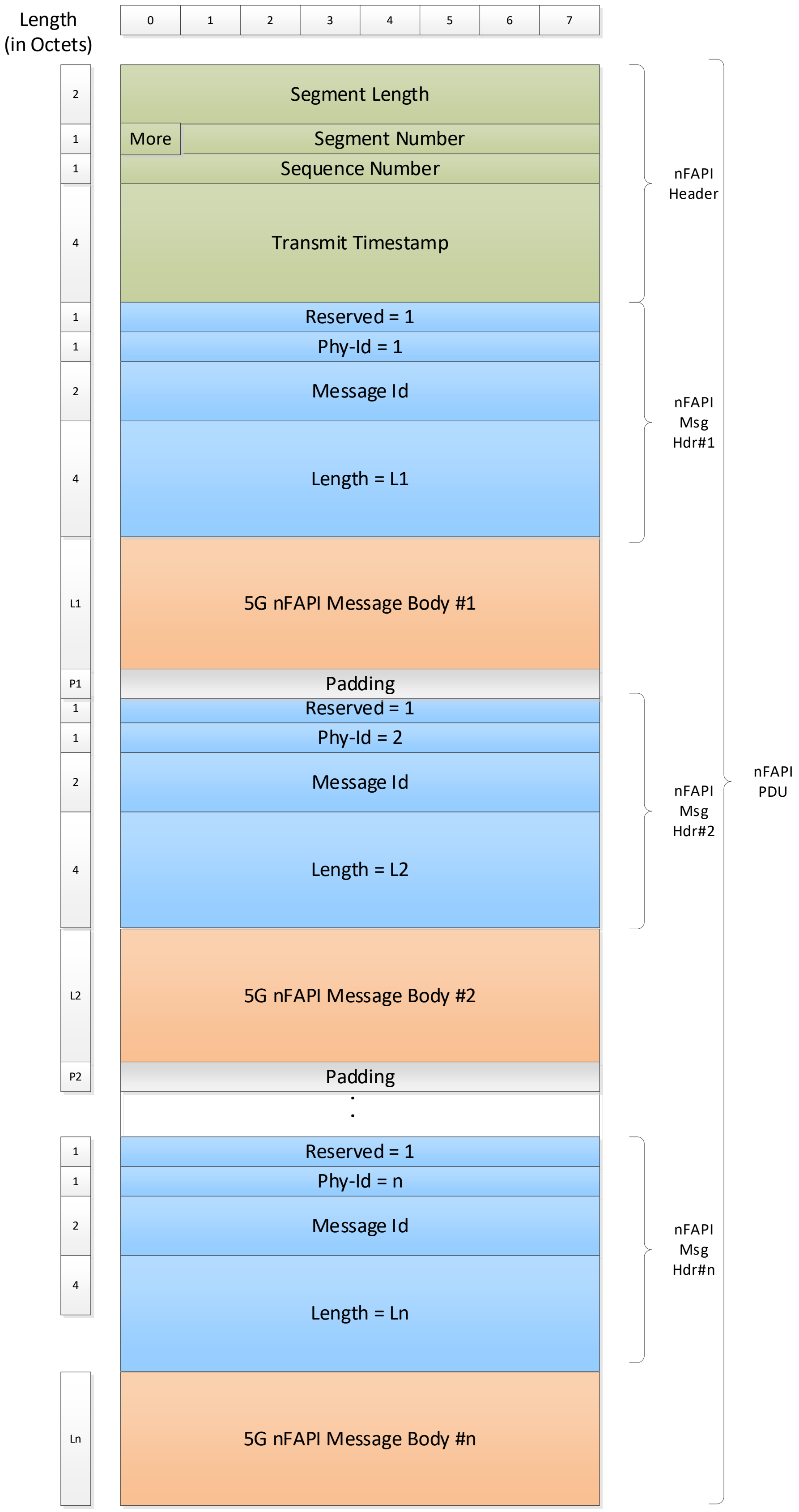


Figure 2–17 P5/P19-C Message header formats with three nFAPI messages illustrated

2.3.2.2 P7/P19-S Message Format

There are two different formatting of the P7/P19-S nFAPI messages. Both nFAPI message formatting starts with nFAPI header and each specific nFAPI message start with nFAPI message header.

The two different formats of the messages are:

- CP-UP combined format, where each nFAPI message body follows the nFAPI message header as shown in .
- CP-UP separation format, where the nFAPI message headers are collected together at the beginning of the payload and the body of the messages follows later as shown in .

P7/P19-S nFAPI messages uses a different nFAPI header as shown in . All other parts (nFAPI message header and padding) of the P7/P19-S nFAPI message are identical to P5/P19-C nFAPI messages.

The P7/P19-S nFAPI header contains the following fields:

Field	Length (Bits)	Description
Sequence Number	16	The incrementing sequence number for all complete messages over the P7/P19-S nFAPI interface. Note: When nFAPI payload is segmented due to transport restriction, all the segments shall contain the same value for this field.
Total SDU Length	24	Length of the P7/P19-S nFAPI SDU. I.e., the PDU excluding this header. Note: When nFAPI payload is segmented due to transport restriction, this field contains the total length before the segmentation.
Byte Offset	24	The offset specifying the start of this segment within the P7/P19-S nFAPI PDU.
Transmit Timestamp	32	The offset from SFN/slot 0/0 time reference of the message transmission at the transport layer, in microseconds, with a range of 0 to 10239999, or 0xFFFF This timestamp shall be used to derive the jitter values in the Timing Info message. It is mandatory for VNF to populate this field if nFAPI PDU contains a segment of at least one of the following messages: - DL_TTI.request - UL_TTI.request - UL_DCI.request - Tx_Data.request - Any P19-S message subject to time management It is also mandatory for PNF to populate this field if nFAPI PDU contains a segment of at least one of the following messages: - P7 Rx_Data.indication - P7 CRC.indication - P7 UCI.indication - P7 SRS.indication - P7 RACH.indication If an P7/P19-S nFAPI message is segmented, the timestamp field in the first segment should be set to the actual transmit time of the segment. The following

Field	Length (Bits)	Description
		segments may set this field to actual transmit time of the segment or may set it to 0xFFFFFFFF. The PNF or the VNF, respectively, shall ignore a time stamp value of 0xFFFFFFFF, if received on a message for which the VNF or PNF, respectively are expected to populate the time stamp field. This value is reserved (set to zero on transmission and ignored on reception) for all other messages.

For P7/P19-S nFAPI messages in CP-UP combined format, the nFAPI message header is identical to P5/P19-C message header described in section 2.3.2.1.

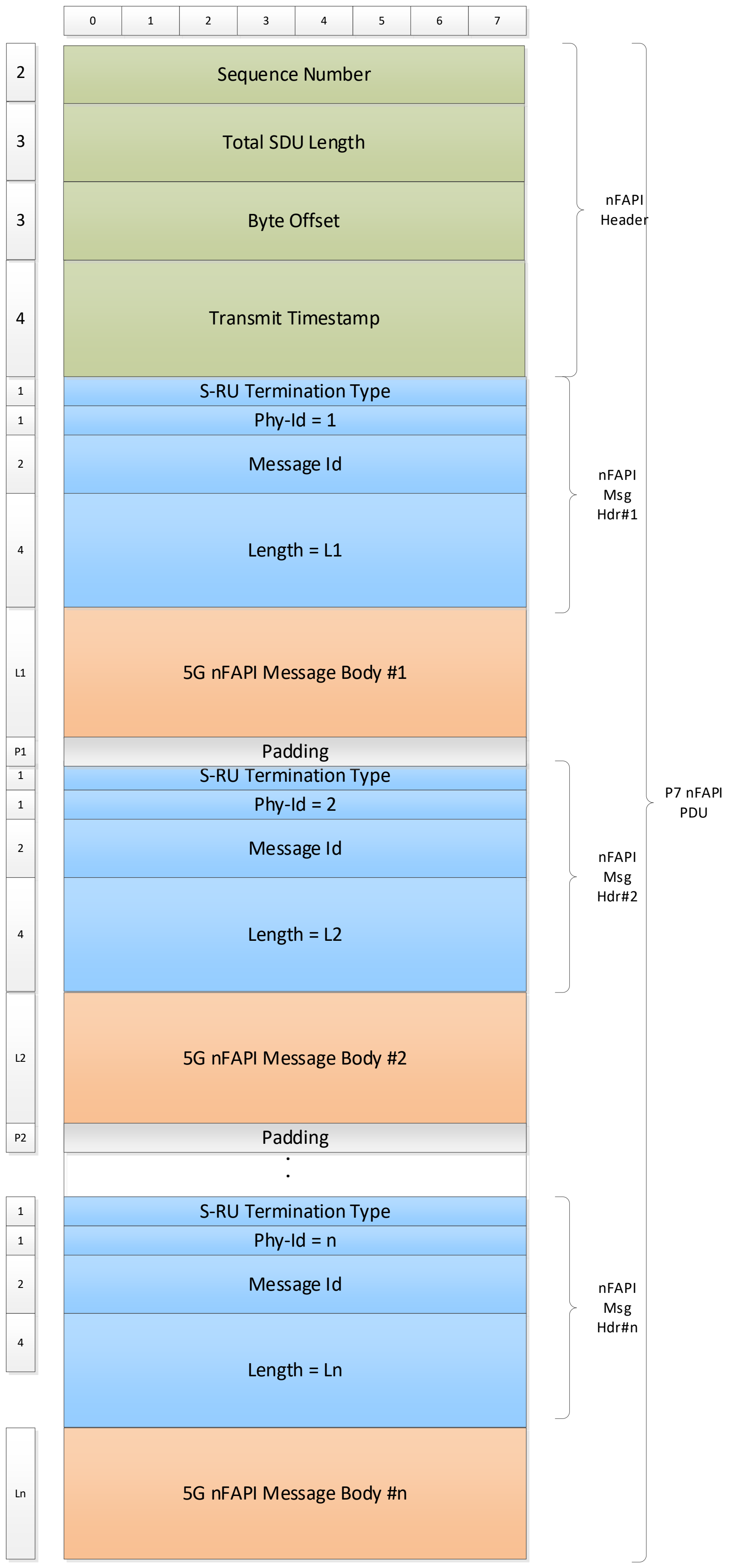


Figure 2–18 P19-S P7 nFAPI headers and message format using CP-UP combined format

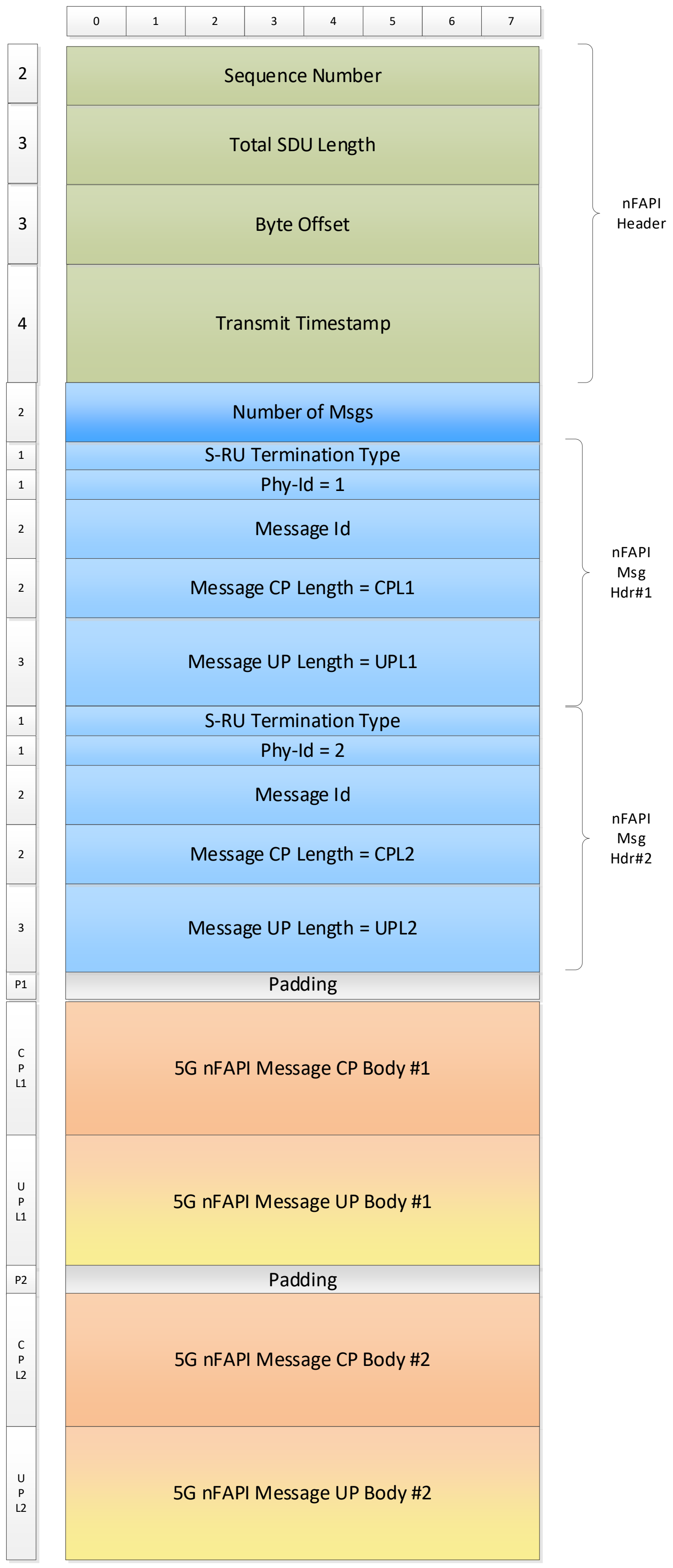


Figure 2–19 P19-S P7 nFAPI headers and messages using the CP-UP Separation formatting

For P7 and P19-S messages using CP-UP separation format, the fields of the nFAPI message header are:

Field	Length (Bits)	Description
S-RU Termination Type	8	This field shall be set to * 0x01 for P5/P7 messages. * 0x02 for P19 messages with DFE components * 0x03 for P19 messages with RF components. * 0x04 for P4 messages with Network Monitor components. For P5,this field allows easy extraction of a FAPI message when the nFAPI protocol carries a FAPI message and is included to simplify FAPI message extraction in the case one or more FAPI messages are present.
PHY ID	8	P5/P7: Within the PNF Device, it is the unique identity of the PHY instance *in the range [1..”Maximum Number Of PHYs”], where “Maximum Number of PHYs” is specified in the PNF_PARAM.response message*. For FAPI messages (FAPI_PARAM, FAPI_CONFIG) directed to PNF and PNF_XXX messages, value PHY-ID=0 is used. This includes any messages directed to the supervisory PHY with PHY ID=0 in [4]. P19: S-RU ID uniquely identifies DFE or RF instance, depending on the content of the S-RU Termination Type field
Message ID	16	The ID of the nFAPI or FAPI message
Message CP Length	16	The length of the control portion of the message. Note all TLVs defined in this specification are categorized as Control Plane.
Message UP Length	24	Then length of the User Plane portion of the message. Notes: - the messages containing User Plane data are the ones with UP/CP separation capability in SCF-222 [4], per the FAPI TLV 0x007D. - when these messages are transported using the CP-UP Separation formatting, the “Tag” field(s) associated with their payloads in those messages shall be set to 3; the “Control Length” and “TLV PDU” contents shall be set consistent with “Tag” value 3 requirements in SCF-222. - e.g. see the “Control Length” and TLV “Tag” fields in SCF-222 [4], section 3.4.6 for `TX_DATA.request`, section 3.4.7 for `RX_DATA.indication`, etc.
Padding	0-3	Number of bytes depends on the length of the previous fields, ensures the next octet starts on a 4-octet boundary

2.3.2.3 Segmentation of P7/P19-S Messages

The P7/P19-S nFAPI header permits segmentation of the messages, if required by transport mechanisms. The Sequence Number increments across completed messages. Figure 2-20 shows how the nFAPI header fields are specified for a segmented message. The SDU length (SL1 and SL2) are computed by subtracting the nFAPI header length (12 octets) from the PDU length obtained by the transport layer, either UDP length field, or for Ethernet transport, 'eCPRI Payload Size' field specified in section 2.3.1.5. Each segment shall contain 1 or more octets of SDU payload.

Note: Techniques for determining whether segmentation is required are outside the scope of this document.

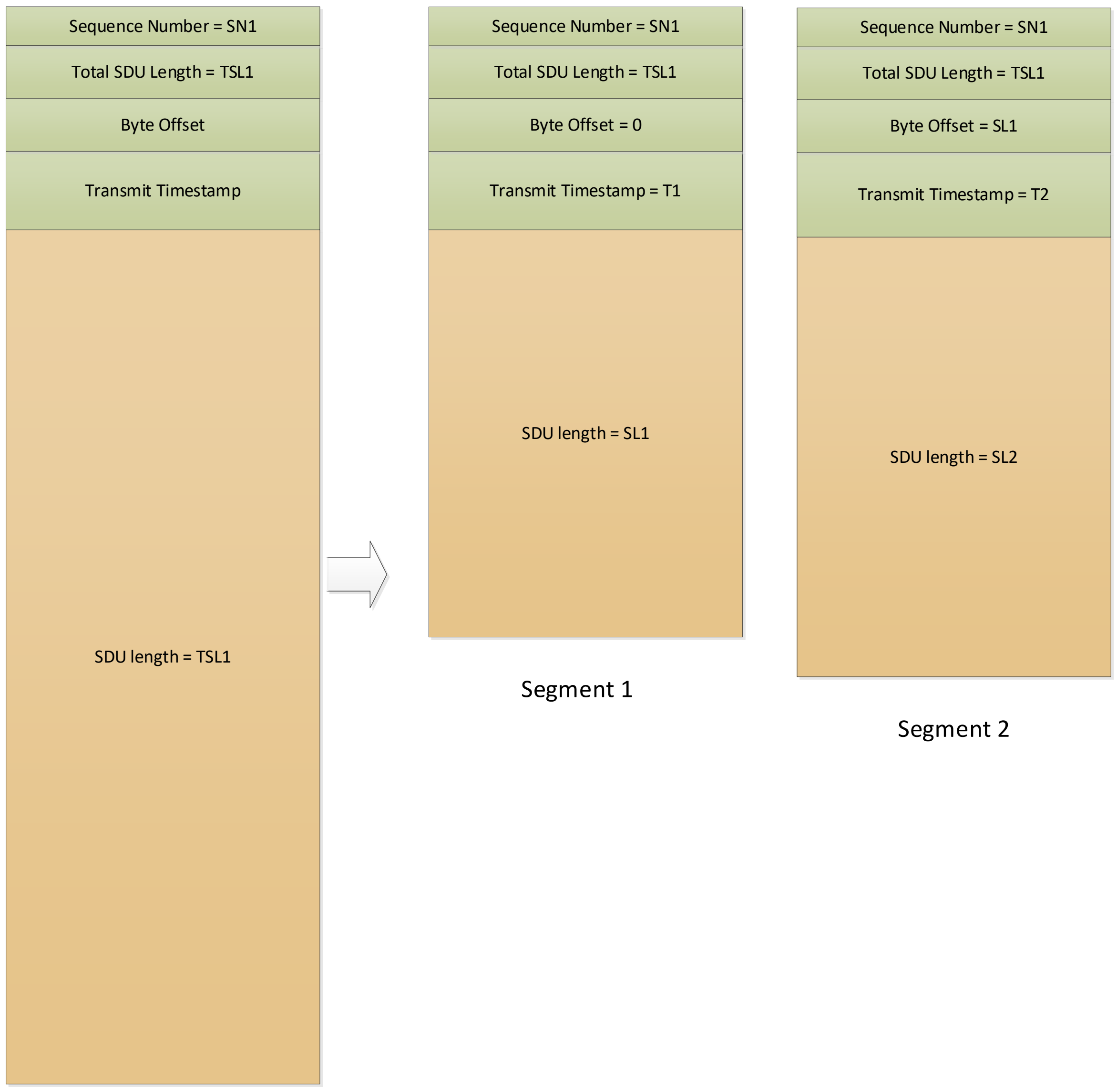


Figure 2–20 Segmentation of P7/P19-S nFAPI message total SDU length TSL1 = SL1 + SL2

2.3.2.4 Byte Order for nFAPI

All multi-octet fields representing integers are laid out in big endian order (also known as 'most significant byte first', or 'network byte order'). All fields as defined in the message structures shall be transmitted in the sequence defined and are packed with no padding, except as specified in the FAPI [4][6] specifications. Bit fields are defined by their bit number with bit 0 being the most significant bit (therefore the left most bit). This byte ordering applies to both nFAPI messages and FAPI messages carried by nFAPI.

2.3.2.5 nFAPI Message Body

The nFAPI message body consists of zero or more values and zero or more type/length/value (TLV) structures as specified in section 2.3.2.6. The TLV parameters are defined as the tag specified in the subsequent sections, the Length field, defined as the length of the subsequent value, and the value field which contains the defined value or structure.

Field	Type	Description
Tag	uint16_t	The tag value for this TLV
Length	uint16_t	The length in bytes of the value field
Value	Variable	The values included in the TLV. The length of this field is specified in the Length field of the TLV

Table 2-1 The TLV definition

2.3.2.6 nFAPI Message Types

The nFAPI protocol carries three types of messages:

- **Dedicated nFAPI messages**: These are messages defined solely in the nFAPI protocol, e.g., PNF_XXX messages
- **Transparent messages**: These are messages that are defined in the FAPI [4] specification, and carried as is by the nFAPI protocol. In other words, the nFAPI only provides a transport of these messages and does not modify them.
- **Combined messages**: These messages are defined by the FAPI specification [4], but the nFAPI specification adds new values and TLVs to these messages.

The message IDs are coordinated so that the messages defined in the FAPI specification [4] do not conflict with the dedicated messages specified in the nFAPI specification.

The transparent and combined messages are listed in the Table 3-4 of the FAPI specification [4]. When these FAPI messages are carried in nFAPI, the most significant 8 bits of the message ID is set to 0x00.

Transparent messages are defined in the nFAPI specification and carried by nFAPI using the nFAPI headers as specified in Figure 2-17. There are no nFAPI content or nFAPI TLVs in these messages. For these messages, the message body section is completely defined in the FAPI specification [4].

Combined messages are formed by encapsulating the FAPI message within the nFAPI message with a special inclusion TLV (5G_MSG_BODY). Note that this TLV only contains the body of the FAPI message, not the FAPI headers. Figure 2-21 shows how a FAPI message (PARAM.response) is augmented with nFAPI TLVs. This example message has a message body with two TLVs specified in the nFAPI spec and a third

TLV (5G_FAPI_MSG_BODY) encapsulated the body of the PARAM.response message specified in [4].

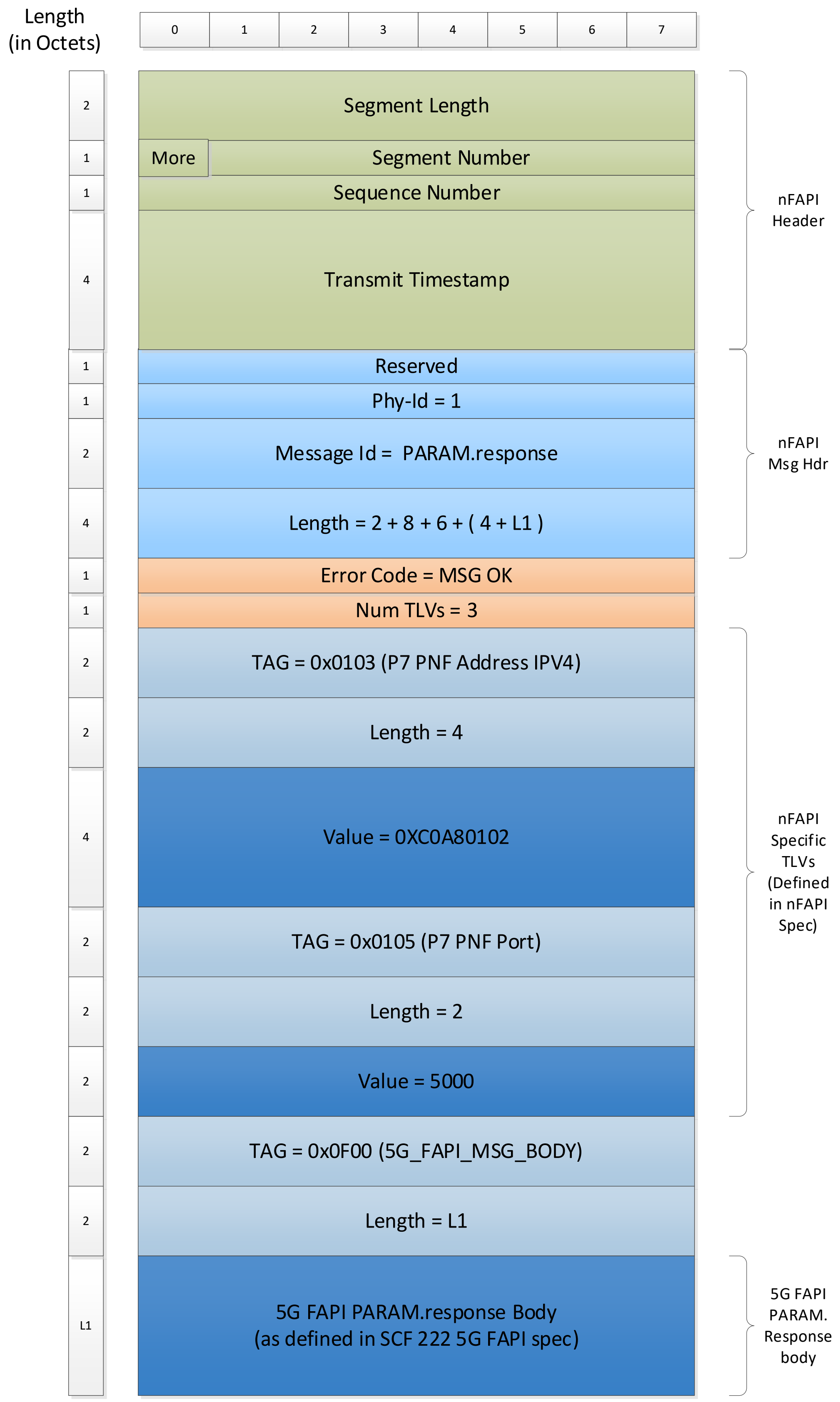


Figure 2–21 Example Combined P5 message (PARAM.response)

2.3.2.7 nFAPI Dedicated Messages

The dedicated nFAPI messages are defined in Table 2-2 below.

Message	Value	Message Body Definition
`PNF PARAM.request`	0x0100	See Section 0
`PNF PARAM.response`	0x0101	See Section 3.1.2
`PNF CONFIG.request`	0x0102	See Section 3.1.3
`PNF CONFIG.response`	0x0103	See Section 3.1.4
`PNF START.request`	0x0104	See Section 3.1.5
`PNF START.response`	0x0105	See Section 3.1.6
`PNF STOP.request`	0x0106	See Section 3.1.7
`PNF STOP.response`	0x0107	See Section 3.1.8
`START.response`	0x0108	See Section 3.1.9
PNF READY.indication	0x0109	See Section 3.1.0
RESERVED for P5 messages	0x010A-0x017F	
`DL_Node Sync`	0x0180	See Section 4.1.1
`UL_Node Sync`	0x0181	See Section 4.1.2
`Timing Info`	0x0182	See Section 4.1.3
RESERVED for P7 messages	0x0183-0x01ff	
RESERVED for Vendor Extension messages	0x0300-0x03ff	See Section 2.3.2.10

Table 2-2 Dedicated nFAPI messages

2.3.2.8 nFAPI Combined Messages

The combined messages listed in Table 2-3 are available for nFAPI.

Message	Value	Message Body Definition
`PARAM.response`	section 3.2 of [4]	See Sections 2.3.2.6 and 3.2.2
`CONFIG.request`	section 3.2 of [4]	See Sections 2.3.2.6 and 3.2.3

Table 2-3 Combined nFAPI messages

2.3.2.9 nFAPI Transparent Messages

The FAPI messages listed in section 3.2 of [4], and in section 1A.2 of [6] and not explicitly marked combined messages in section 2.3.2.8, can be transported transparently in nFAPI. As of FAPI v2.0, the FAPI P5/P7 messages listed in Table 2-4 can be transported transparently via nFAPI. The `SLOT.indication` and `TIMING.indication` FAPI messages do not serve any function in nFAPI architecture, and are thus not listed below.

Message	Value	Message Body Definition
`PARAM.request`	section 3.2 of [4]	See Section 2.3.2.6
`CONFIG.response`	section 3.2 of [4]	See Section 2.3.2.6
`START.request`	section 3.2 of [4]	See Section 2.3.2.6
`STOP.request`	section 3.2 of [4]	See Section 2.3.2.6
`STOP.indication`	section 3.2 of [4]	See Section 2.3.2.6
`ERROR.indication`	section 3.2 of [4]	See Section 2.3.2.6
`DL_TTI.request`	section 3.2 of [4]	See Section 2.3.2.6
`UL_TTI.request`	section 3.2 of [4]	See Section 2.3.2.6
`UL_DCI.request`	section 3.2 of [4]	See Section 2.3.2.6
`TX_Data.request`	section 3.2 of [4]	See Section 2.3.2.6
`Rx_Data.indication`	section 3.2 of [4]	See Section 2.3.2.6
`CRC.indication`	section 3.2 of [4]	See Section 2.3.2.6
`UCI.indication`	section 3.2 of [4]	See Section 2.3.2.6
`SRS.indication`	section 3.2 of [4]	See Section 2.3.2.6
`RACH.indication`	section 3.2 of [4]	See Section 2.3.2.6
`CONNECTIVITY.indication`	section 3.2 of [4]	See Section 2.3.2.6
`RESET.request`	section 3.2 of [4]	See Section 2.3.2.6
`RESET.indication`	section 3.2 of [4]	See Section 2.3.2.6
`Vendor messages`	section 3.1 of [4]	See Section 2.3.2.6

Table 2-4 Transparent P5/P7 nFAPI messages

The FAPI messages listed in section 1A.2 and defined in sections 2.3, 3.5 and 4.3 of [6], can be transported transparently in nFAPI. As of FAPI-FEU v2.0, the FAPI messages listed in Table 2-5 can be transported transparently via nFAPI. The `SLOT.indication` and `TIMING.indication` FAPI messages do not serve any function in nFAPI architecture, and are thus not listed below.

Message	Value	Message Body Definition
RF `PARAM.request`	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF `PARAM.response`	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF `CONFIG.request`	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF `CONFIG.response`	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF `INDICATION`	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF `SCHEDULE.request`	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF `SCHEDULE.response`	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF `START.request`	See section 1A.2 of [SCF-223]	See Section 2.3.2.6

Message	Value	Message Body Definition
RF START.response	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF STOP.request	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF STOP.indication	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF ERROR INDICATION	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF SLOT.indication	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF DL Node Sync	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF UL Node Sync	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF Timing Info	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF CONNECTIVITY.indication	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF RESET.request	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
RF RESET.indication	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.SET_BWT.REQUEST	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.SET_BWT.RESPONSE	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.GET_BWT_LIST.REQUEST	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.GET_BWT_LIST.RESPONSE	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.GET_BWT.REQUEST	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.GET_BWT.RESPONSE	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.GET_BCT.REQUEST	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.GET_BCT.RESPONSE	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.SET_BCT.REQUEST	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.SET_BCT.RESPONSE	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.UPDATE_BEAM.REQUEST	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.UPDATE_BEAM.RESPONSE	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.SELECT_BEAM.REQUEST	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.SELECT_BEAM.RESPONSE	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.SET_BEAM_SLOT_PATTERN .REQUEST	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
FEU.CONFIG.SET_BEAM_SLOT_PATTERN .RESPONSE	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE PARAM.request	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE PARAM.response	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE CONFIG.request	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE CONFIG.response	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE INDICATION	See section 1A.2 of [SCF-223]	See Section 2.3.2.6

Message	Value	Message Body Definition
DFE SCHEDULE.request	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE SCHEDULE.response	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE START.request	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE START.response	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE STOP.request	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE STOP.indication	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE ERROR INDICATION	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE SLOT.indication	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE DL Node Sync	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE UL Node Sync	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE Timing Info	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE CONNECTIVITY.indication	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE RESET.request	See section 1A.2 of [SCF-223]	See Section 2.3.2.6
DFE RESET.indication	See section 1A.2 of [SCF-223]	See Section 2.3.2.6

Table 2-5 Transparent P19 nFAPI messages

2.3.2.10 Vendor Extensions

There are optional vendor extensions within all nFAPI P5, P7 and P19 messages. These vendor extensions are contained within the message structure and shall immediately follow the standard nFAPI message content. The vendor extension shall contain a Vendor Extension Tag and Length field of the vendor extension following the same rules as the nFAPI structures, with the subsequent content to be the vendor extension payload. The nFAPI message header length shall be inclusive of the vendor extension.

There are also defined vendor extension message identifier ranges that permit entire nFAPI messages to be added for vendor extension procedures. These are defined with a Vendor Extension Message ID and shall follow the nFAPI message header content and rules as specified in section 2.3.2, with the subsequent content to be the vendor extension payload.

As an example, Figure 2-22 shows the `PARM.response` nFAPI message with the nFAPI header, two nFAPI TLVs and a single nFAPI Vendor Extension. Note: If the message contains a FAPI message using the 5G_FAPI_MSG_BODY TLV, the included FAPI message may also contain FAPI level extension TLVs.

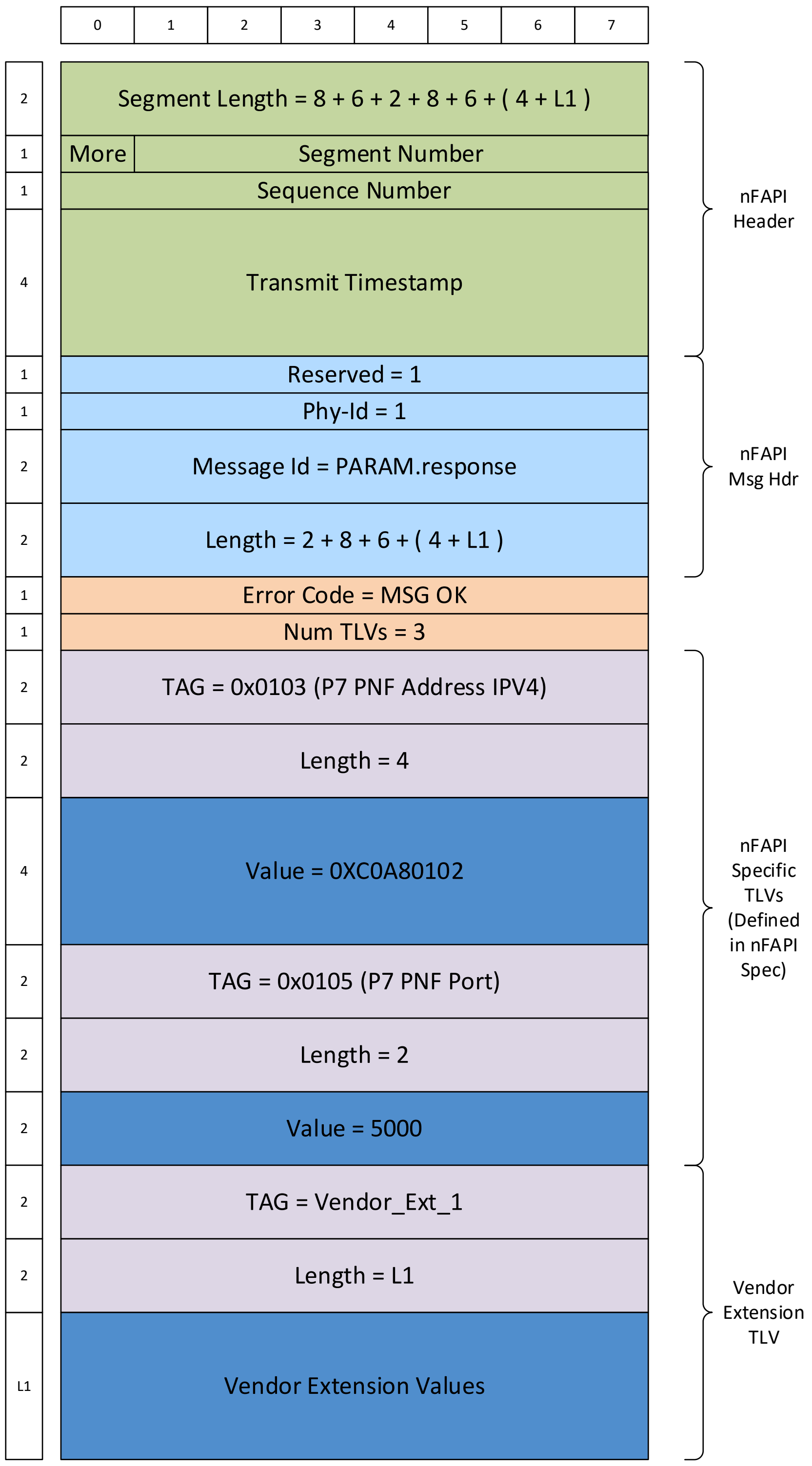


Figure 2–22 Vendor Extension Example

To remain interoperable with VNFs which do not understand the vendor extensions from a specific PNF vendor, a PNF should not rely on any change in behavior of the VNF as a result of the inclusion of vendor extensions by the PNF, and should not modify its behavior or procedures away from this specification, unless a VNF responds (in a vendor extension for that PNF vendor) in such a way that indicates it can perform vendor-specific behaviour.

The range of nFAPI VE tags available for use is defined in Table 2-6 and is a reserved set of tags that can be used by the individual vendors as required.

Tag (Hex)	Description
0xF000 – FFFF	Range of tag values available for use as vendor extension tags for use with nFAPI messages

Table 2-6 nFAPI vendor extension tag values

The vendor of the PNF may be identified from the OUI contained within the `PNF PARAM response` message. The VNF may use this OUI value in order to determine how to interpret the VE message identifiers and the VE tags within the standard messages as well as the nFAPI vendor specific messages.

2.3.2.11 Message Tags in the FAPI Suite

This section lists the message tag ranges used or reserved across the P5/P7 [4] , P4 [7], P19 [6] and nFAPI (this specification).

Message Scope	Message Tag Range	Comments
`FAPI P5` Defined	0x0000 – 0x000a	0x00 – 0x07 defined in SCF-222 [4]. SCF-225 prefixes 0x00*[nn]* for 16-bit extension
`FAPI` Reserved	0x000b – 0x007F	SCF-222 [4] does not spell out P5 or P7. SCF-225 prefixes 0x00[nn] for 16-bit extension
`FAPI P7` Defined	0x0080 – 0x008B	0x80 – 0x8A defined in SCF-222 [4]. SCF-225 prefixes 0x00[nn] for 16-bit extension
`FAPI` Reserved	0x008C-0x00FF	SCF-222 [4] does not spell out P5 or P7. SCF-225 prefixes 0x00*[nn]* for 16-bit extension
`nFAPI P5` Defined	0x0100-0x0109	In this spec, for PNF and PHY
`nFAPI P5` Reserved	0x0109-0x017F	In this spec
`nFAPI P7` Defined	0x0180-0x0183	In this spec, for P7 Delay Management
`nFAPI P7` Reserved	0x0184-0x01FF	In this spec
`FAPI P4` Defined	0x0200-0x0210	In SCF-224 [7]
`FAPI P4` Reserved	0x0211-0x02FF	
RESERVED for (n)FAPI Vendor Extension messages	0x0300-0x03FF	In this or any FAPI spec
`FAPI P19` RF-Defined	0x0400-0x040F	In SCF-223 [6], for RF non-ABF messages
`FAPI P19` RF-Reserved	0x0410-0x047F	
`FAPI P19` RF/ABF-Defined	0x0480-0x048F	In SCF-223 [6], for RF ABF messages
`FAPI P19` RF/ABF-Reserved	0x0490-0x04FF	
`FAPI P19` DFE-Defined	0x0500-0x050F	In SCF-223 [6], for DFE messages

Message Scope	Message Tag Range	Comments
`FAPI P19` DFE-Reserved	0x0510-0x05FF	
Reserved (across the entire FAPI Suite)	0x0600-0xFFFF	

2.3.3 Transport and nFAPI message loss

2.3.3.1 P5/P19-C interface messages

The SCTP association is initialized from the PNF towards the VNF using the end point configuration defined over OAM interface. There is a bidirectional (incoming plus outgoing) stream for the PNF, each PHY. Once the streams are established, the nFAPI procedures can occur between the P5/P19-C applications of the PNF and VNF.
If either side decides to perform a shutdown of the SCTP association, the messages in

flight within each stream are delivered and then all streams are closed, and the application layer must handle the recovery. This should result in a cleanup of application contexts associated with the SCTP association and return to the reconnected state. For the PNF, this should return to a re-connection attempt or raise

awareness of the status through the OAM interface. For the VNF, the resources associated with that particular PNF should be cleared and return to a pre-connected state leaving other PNFs unaffected – which is VNF implementation specific and out of scope for this document.

If either side decides to perform an abort of the SCTP association, the procedure for handling at the application layer is the same as the shutdown procedure. The only difference being that the messages pending transmission are discarded, undelivered.

If communication with the end point is lost (e.g., through time out of heartbeat) then the same procedure as abort/shutdown for handling at the application layer is required.

Further details and example application / protocol procedures are defined in RFC 4960.

2.3.3.2 P7/P19-S interface messages

UDP is a connectionless protocol. This means it relies on the application layer to handle lost and out-of-order messages. To this end, the P7 protocol header defines segments and sequence numbers on a per PHY instance to permit the nFAPI P7P19-S application to handle such occurrences. When the receiver on the P7/P19-S interface recognizes missed or out of order messages, it is the responsibility of the receiving entity to handle such situations. This means for the PNF within slot procedures, the PNF can buffer and play-out the received messages, handling any out-of-order or segmented messages that require some extra time to receive and re-constitute. Refer to sections 2.1.3.4 and 2.1.3.5 for handling of these cases.

3. P5 messages

3.1 nFAPI Dedicated Messages

This section lists the dedicated nFAPI messages.

3.1.0 PNF_READY.indication

This message is sent by PNF to VNF when PNF is configured properly and SCTP connection is established between PNF and VNF.

Tag	Field	Type	Description
N/A	Version Info	Uint32_t	Version and other information reserved for use in future. Set to 0 in this spec version.

Table 3-1 PNF_READY.indication parameters

3.1.1 PNF_PARAM.request

This message is sent by the VNF when the PNF is in the PNF IDLE state. There is no message body in the `PNF_PARAM.request`.

3.1.2 PNF_PARAM.response

The `PNF_PARAM.response` message defines the PNF's overall capability and options for the VNF to follow up with the `PNF_CONFIG.request`.

Tag	Field	Type	Description
N/A	Error Code	uint8_t	The error code
N/A	NumTLV	uint8_t	The number of TLVs
0x1000	PNF Param General	TLV	The general PNF capability parameters as defined in Table 3-3

Table 3-2 PNF_PARAM.response parameters

3.1.2.1 PNF Param General

Field	Type	Description
nFAPI Sync Mode	uint8_t	The method of nFAPI synchronization supported by the PNF. 0 = un-aligned synchronization[1] 1 = internal PNF frame alignment[2]

[1] The PHY(s) within the PNF device operate to individual frame alignment

[2] The PHY(s) within the PNF device operate to a shared frame alignment

Field	Type	Description
		2 = absolute time aligned synchronization[3] Other values are invalid
Location Mode	unit8_t	The method of location derivation supported by the PNF. 0 = none 1 = GPS 2 = GLONASS 3 = BeiDou 4 = NavIC Other values are Reserved
Location Coordinates	Array of uint8_t	The Location of the PNF. The value is formatted as the *RANAccessPointPosition* IE as defined in section 9.3.5 of [11], using BASIC-PER encoding as defined in section 9.4 of [11]. The first bit of the *NG-RANAccessPointPosition* IE is in the LSB of the first byte of the array.
Maximum Number PHYs	uint16_t	The maximum number of operational PHY instances supported by the PNF device. Each PHY instance will be assigned a PHY ID in the range [1..Maximum Number PHYs], subject to PHY Instantiation (see section 2.1.2.9). The PHY ID is the S-RU component ID for messages addressed to PHY S-RU Termination types (see section 2.3.2)
OUI	Array of uint8_t	The PNF's Organizational Unique Identifier as specified by [12]. The 24 bits of OUI are represented via a three-octet array, with octets 0, 1, respectively 2 of OUI in the array positions 0, 1, respectively 2. For each octet, bit fields are defined by their bit number with bit 0 being the most significant bit (therefore the left most bit).
Number of RF Instances	uint16_t	The number of operational RF instances supported by the PNF device. Each RF instance is assigned an RF ID in the range [1..Number of RF Instances]. The RF ID is the S-RU component ID for messages addressed to RF S-RU Termination types (see section 2.3.2)
Number of DFE Instances	uint16_t	The number of operational DFE instances supported by the PNF device. Each DFE instance is assigned an RF ID in the range [1..Number of DFE Instances]. The DFE ID is the S-RU component ID for messages addressed to DFE S-RU Termination types (see section 2.3.2)

Table 3-3 PNF Param general parameters

[3] The PHY(s) within the PNF device operate to an absolute time frame alignment

3.1.2.2 PNF PARAM.response Errors

The error codes which may be returned in `PNF_PARAM.response` are given in Table 3-4.

Error Code	Description
MSG_OK	Message is OK.
MSG_INVALID_STATE	The PNF_PARAM.request was received when the PNF was not in the PNF IDLE state.
MSG_INVALID_CONFIG	The PNF_PARAM.request was received with invalid configuration.

Table 3-4 Error codes for PNF_PARAM.response

Note: In this version of nFAPI, no additional information is appended to the invalid config cause. Vendor extensions can be added if specific error causes or details on error reasons are required.

3.1.3 PNF_CONFIG.request

The `PNF_CONFIG.request` message configures the PNF for operation and, if the PNF is in the PNF IDLE state, instructs the PNF to move into the PNF CONFIGURED state. Subsequent reception of the `PNF_CONFIG.request` in the PNF CONFIGURED state re-configures the PNF for the newly supplied configuration parameters.

Note: In the current specification version, no standardized configuration TLVs are defined, though the message is still needed to change the PNF state. Vendor extensions (see section 2.3.2.10) are still possible.

Tag	Field	Type	Description
N/A	Num TLVs	uint8_t	Number of TLVs included

Table 3-5 PNF_CONFIG.request parameters

3.1.4 PNF_CONFIG.response

The `PNF_CONFIG.response` message confirms the PNF state and acceptance or rejection of the `PNF_CONFIG.request` configuration.

Tag	Field	Type	Description
N/A	Error Code	uint8_t	The error code

Table 3-6 PNF_CONFIG.response parameters

3.1.4.1 PNF CONFIG Errors

The error codes which may be returned in `PNF_CONFIG.response` are given in Table 3-7.

Error Code	Description
MSG_OK	Message is OK.

Error Code	Description
MSG_INVALID_STATE	The PNF_CONFIG.request was received when the PNF was not in the PNF IDLE state or the PNF CONFIGURED state.
MSG_INVALID_CONFIG	The configuration provided contains parameters that are invalid or unsupported by the PNF.

Table 3-7 Error codes for PNF_CONFIG.response

3.1.5 PNF_START.request

The `PNF_START.request` message moves the PNF into PNF RUNNING state from the PNF CONFIGURED state. No message body is defined for `PNF_START.request`.

3.1.6 PNF_START.response

The `PNF_START.response` message confirms the PNF state and acceptance or rejection of the `PNF_START.request` configuration.

Tag	Field	Type	Description
N/A	Error Code	uint8_t	The error code

Table 3-8 PNF_START.response parameters

3.1.6.1 PNF_START.response Errors

The error codes which may be returned in `PNF_START.response` are given in Table 3-9.

Error Code	Description
MSG_OK	Message is OK.
MSG_INVALID_STATE	The `PNF_START.request` was received when the PNF was not in the PNF CONFIGURED state.

Table 3-9 Error codes for PNF_START.response

3.1.7 PNF_STOP.request

The `PNF_STOP.request` message moves the PNF into PNF CONFIGURED state from the PNF RUNNING state. The PHY instances are reset (if in the RUNNING state) on reception of the `PNF_STOP.request`, ready for re-configuration or re-starting as required. No message body is defined for `PNF_STOP.request`.

3.1.8 PNF_STOP.response

The `PNF_STOP.response` message confirms the PNF state and acceptance or rejection of the `PNF_STOP.request` configuration.

Tag	Field	Type	Description
N/A	Error Code	uint8_t	The error code

Table 3-10 PNF_STOP.response parameters

3.1.8.1 PNF_STOP.response Errors

The error codes which may be returned in `PNF_STOP.response` are given in Table 3-11.

Error Code	Description
MSG_OK	Message is OK.
MSG_INVALID_STATE	The `PNF_STOP.request` was received when the PNF was not in the PNF RUNNING state.

Table 3-11 Error codes for PNF_STOP.response

3.1.9 START.response

The `START.response` message confirms the PHY state and acceptance of the `START.request` configuration. FAPI specification [4] details how rejection of `START.request` configuration is signalled.

Tag	Field	Type	Description
N/A	Error Code	uint8_t	The error code

Table 3-12 START.response parameters

3.1.9.1 START.response Errors

The error codes which may be returned in `START.response` are given in Table 3-9.

Error Code	Description
MSG_OK	Message is OK.

Table 3-13 Error codes for START.response

3.2 Combined and Transparent nFAPI messages

These message IDs are defined in table 3-4 of the FAPI specification [4]. In this section, additional TLVs for each of these messages are listed.

3.2.1 PARAM.request

This message is sent by the VNF when the PNF is in the PNF RUNNING state and the PHY instance is in the IDLE state. This is a transparent message and the body of this message is specified in section 3.3.1.1 of [4] . No additional TLVs are defined for this message in this specification.

3.2.2 PARAM.response

The `PARAM.response` message defines the PHY instance's overall capability and options for the requester to follow up with the `CONFIG.request`. The message body is defined in Table 3-14.

Field	Type	Description
Error Code	uint8_t	The error code
Num TLVs	uint8_t	The number of nFAPI TLVs contained in this message body
TLVs	Variable	See Table 3-15 and Table 3-16

Table 3-14 PARAM.response message body

Description	Tag
P7 PNF Address Ipv4	0x0103 (if Ipv4 supported)
P7 PNF Address Ipv6	0x0104 (if Ipv6 supported)
P7 PNF Port	0x0105 (If UDP/IP transport is supported)
5G_FAPI_MSG_BODY	0x0F00

Table 3-15 nFAPI TLVs included in `PARAM.response` when the PHY instance is in IDLE state

Description	Tag
P7 PNF Address Ipv4	0x0103 (if Ipv4 supported)
P7 PNF Address Ipv6	0x0104 (if Ipv6 supported)
P7 PNF Port	0x0105 (If UDP/IP transport is supported)
P7 VNF Address Ipv4	0x0100 (if Ipv4 supported)
P7 VNF Address Ipv6	0x0101 (if Ipv6 supported)
P7 VNF Port	0x0102 (If UDP/IP transport is supported)
P7 Transport	0x0122
P7 PNF Ethernet Address	0x0123 (if Ethernet transport is supported)
eCPRI Message Type	0x0125 (if Ethernet transport is supported)
eCPRI Phy Transport ID	0x0126 (if Ethernet transport is supported)
P7 Version used	0x0127
DL_TTI Timing offset	0x0106
UL_TTI Timing offset	0x0107
UL_DCI Timing offset	0x0108
Tx_Data Timing offset	0x0109
Timing window	0x011E
Timing info mode	0x011F
Timing info period	0x0120

Description	Tag
P7 IP Fragmentation Allowed	0x0121
5G_FAPI_MSG_BODY	0x0F00

Table 3-16 nFAPI TLVs included in `PARAM.response` when the PHY instance is in CONFIGURED state

Note: The 5G_FAPI_MSG_BODY TLV contains the FAPI PARAM.response message specified in section 3.3.1.2 of [4].

3.2.2.1 PARAM.response Errors

The error codes which may be returned in `PARAM.response` are given in Table 3-17.

Error Code	Description
MSG_OK	Message is OK.
MSG_INVALID_STATE	The `PARAM.request` was received when the PHY instance was not in the IDLE or CONFIGURDED states.

Table 3-17 Error codes for PARAM.response

3.2.3 CONFIG.request

This message is sent by the VNF when the PNF is in the PNF RUNNING state and the PHY instance is in the IDLE state. The `CONFIG.request` message configures the PHY instance for operation and instructs the PHY instance to move into the CONFIGURED state. Subsequent reception of the `CONFIG.request` in the CONFIGURED state re-configures the PHY instance for the newly supplied configuration parameters. The body of this message is defined in Table 3-18.

Field	Type	Description
Num TLVs	uint8_t	The number of nFAPI TLVs contained in this message body
TLVs	Variable	See Table 3-19

Table 3-18 CONFIG.request message body

There is no requirement for the VNF software to provide the TLVs in the order specified in the Tables.

Description	Tag
P7 VNF Address Ipv4	0x0100 (if Ipv4 supported)
P7 VNF Address Ipv6	0x0101 (if Ipv6 supported)
P7 VNF Port	0x0102 (If UDP/IP transport is supported)
P7 Transport	0x0122
P7 VNF Ethernet Address	0x0124 (if Ethernet transport is supported)
Timing window	0x011E
Timing info mode	0x011F

Description	**Tag**
Timing info period	0x0120
5G_FAPI_MSG_BODY	0x0F00

Table 3-19 nFAPI TLVs included in `CONFIG.request` for IDLE and CONFIGURED states

Note: The 5G_FAPI_MSG_BODY TLV contains the FAPI CONFIG.request message specified in section 3.3.2.1 of [4].

3.2.4 CONFIG.response

The `CONFIG.response` message confirms the PHY instance's state and acceptance or rejection of the `CONFIG.request` configuration. This is a transparent message and the body of this message is defined in section 3.3.2.2 of [4].

3.2.5 START.request

This message is sent by the VNF when the PNF is in the PNF RUNNING state and the PHY instance is in the CONFIGURED state. The `START.request` message instructs the PHY instance to move into the RUNNING state. This is a transparent message and the body of this message is defined in section 3.3.4.1 of [4]. No additional TLVs are defined for `START.request` message.

3.2.6 ERROR.indication

This message is used to report to the VNF. The body of this message is defined in section 3.3.6.1 of [4]. This is a transparent message and no additional TLVs or values defined for this message in this specification.

3.2.7 STOP.request

The `STOP.request` message moves the PHY instance into CONFIGURED state from the RUNNING state. The body of this message is specified in section 3.3.5.1 of [4]. This is a transparent message and no additional TLVs are defined for this message in this specification.

3.2.8 STOP.indication

`STOP.indication` is a transparent message. The body of this message is specified in section 3.3.5.2 of [4]. No additional TLVs are defined for this message in this specification.

3.2.9 nFAPI Configuration TLVs

Configuration TLVs are used in the PARAM.xxx and CONFIG.xxx message exchanges with the PHY instance.

The TLV format is specified in Table 2-1.

Tag	**Description**	**Type**	**Value**
0x0100	P7 VNF Address Ipv4	Array of uint8_t	The Ipv4 address of the VNF to be used by the PNF for this P7 PHY instance

Tag	Description	Type	Value
			Note: Address is network byte order. If both Ipv4 and Ipv6 addresses supplied, dual stack implementation is required.
0x0101	P7 VNF Address Ipv6	Array of uint8_t	The Ipv6 address of the VNF to be used by the PNF for this P7 PHY instance Note: Address is network byte order. If both Ipv4 and Ipv6 addresses supplied, dual stack implementation is required.
0x0102	P7 VNF Port	uint16_t	The port of the VNF to be used by the PNF for this P7 PHY instance
0x0103	P7 PNF Address Ipv4	Array of uint8_t	The Ipv4 address of the PNF PHY instance to be used by the VNF for this PNF PHY instance Note: Address is network byte order. If both Ipv4 and Ipv6 addresses supplied, dual stack implementation is required.
0x0104	P7 PNF Address Ipv6	Array of uint8_t	The Ipv6 address of the PNF PHY instance to be used by the VNF for this PNF PHY instance Note: Address is network byte order. If both Ipv4 and Ipv6 addresses supplied, dual stack implementation is required.
0x0105	P7 PNF Port	uint16_t	The port of the PNF PHY instance to be used by the VNF for this PNF PHY instance
0x0F00	5G_FAPI_MSG_BODY	Variable	The body of the included FAPI message as specified in [9].

Table 3-20 TLV Tags used by nFAPI between VNF and PNF for transport connection setup

Tag	Description	Type	Value
0x0106	DL_TTI Timing offset	uint32_t	The timing offset before the air interface slot start that the DL_TTI.request must be received at the PHY instance. The value is in microseconds (μs).
0x0107	UL_TTI Timing offset	uint32_t	The timing offset before the air interface slot start that the UL_TTI.request must be received at the PHY instance. The value is in microseconds (μs).
0x0108	UL_DCI Timing offset	uint32_t	The timing offset before the air interface slot start that the UL_DCI.request must be received at the PHY instance. The value is in microseconds (μs).
0x0109	Tx_Data Timing offset	uint32_t	The timing offset before the air interface slot start that the Tx_Data.request must be received at the PHY instance.

Tag	Description	Type	Value
			The value is in microseconds (μs).
0x011E	Timing window	uint16_t	The window in microseconds that the PHY instance must receive and queue the P7 messages. Value: 0 → 30,000 microseconds Other values are reserved
0x011F	Timing info mode	uint8_t	The configured mode of operation for the timing info message to be sent to the VNF from the PHY instance. Value: bitX :0 = disabled, 1= enabled. Bit0: Periodic Bit1: Aperiodic Bit2 – Bit7: Reserved
0x0120	Timing info period	uint8_t	If Periodic timing mode is enabled, this defines the periodicity in slots. This field is ignored if periodic timing mode is disabled. Value: 1 → 255 Other values are reserved
0x0121	P7 IP Fragmentation Allowed	uint8_t	Specifies if PHY instance can handle IP fragmented P7 messages. • If 0x1: The PHY instance can perform IP reassembly in addition to nFAPI reassembly • If 0x0: the PHY instance can only perform nFAPI reassembly and VNF shall not send large P7 packets that would require IP fragmentation. And the sender shall set the DF (Don't fragment) bit in the IP header.
0x0122	P7 Transport	uint8_t	Specifies the transport supported by PHY instance: 0x1 : PHY supports UDP/IP transport 0x2 : PHY supports Ethernet with eCPRI 0x3 : Phy supports both transports Note: when this IE is used in the CONFIG.xxx message the value 0x3 cannot be used.
0x0123	P7 PNF PHY Ethernet Address	Array of uint8_t	6-octect IEEE Ethernet MAC address of the PHY instance
0x0124	P7 VNF Ethernet Address	Array of uint8_t	6-octect IEEE Ethernet MAC address of the VNF

Tag	Description	Type	Value
0x0125	eCPRI Message Type	uint8_t	The 'Message Type' to be used in the eCPRI header
0x0126	eCPRI Phy Transport ID	uint16_t	The value to use for 'Phy Transport ID' field in eCPRI message to refer to this PHY instance. This is used to route the message to a specific Phy instance.
0x0127	P7 Version Used	Uint8_t	Specifies the format used for P7/P19-S messages, see section 2.3.2.2. Note messages both in DL and UL direction shall be using the same formatting specified by this parameter. • 0x0 –CP-UP combined format is used • 0x1 – CP-UP separation format is used.

Table 3-21 TLV Tags used by the VNF to configure the delay management with the PHY

4. P7 Messages

P7 protocol only contains dedicated and transparent messages. It does not have any combined messages as defined in section 2.3.2.6.

4.1 nFAPI Dedicated P7 messages

4.1.1 DL Node Sync

This message is sent by the VNF to the PNF PHY instance initiating a single sync procedure.

Tag	Field	Type	Description
N/A	t1	uint32_t	Offset from VNF SFN/slot 0/0 time reference of the DL Node Sync message transmission at the transport layer, in microseconds. Value: 0 to 10239999 in units of µs Other values are reserved
N/A	Delta SFN/slot	int32_t	The delta shift in slots that the S-RU component instance must update to on the next slot boundary -163839to 163839 Notes: - The range covers [-1024*160+1, 1024*160-1] Negative numbers indicate number of slots to pull (negatively advance) the SFN/slot timeline.
N/A	Sub-carrier spacing	unit8_t	SubcarrierSpacing index as defined in TS38.211 ver 15.5.0 sec 4.2 0: 15 KHz 1: 30 KHz 2: 60 KHz 3: 120 KHz 4: 240 KHz

Table 4-1 DL Node Sync parameters

4.1.2 UL Node Sync

This message is sent by the PNF PHY instance to the VNF on receipt of the DL Node Sync.

Tag	Field	Type	Description
N/A	t1	uint32_t	The supplied t1 field in the DL Node Sync Value: 0 to 10239999 in units of μs Other values are reserved
N/A	t2	uint32_t	Offset from PNF SFN/slot 0/0 time reference of the DL Node Sync message reception at the transport layer, in microseconds. Value: 0 to 10239999 in units of μs Other values are reserved
N/A	t3	uint32_t	Offset from PNF SFN/slot 0/0 time reference of the UL Node Sync message transmission at the transport layer, in microseconds. Value: 0 to 10239999 in units of μs Other values are reserved

Table 4-2 UL Node Sync parameters

4.1.3 Timing Info

This message is be sent by the PNF PHY instance to the VNF as configured through the `CONFIG.request` message.

The content of the jitter fields in Timing Info for the various P7 messages (`DL_TTI.request, TxData.request, UL_TTI.request, UL_DCI.request`) is computed using the jitter calculation defined in RFC 3550 Section 6.4.1, with the following changes:

- the *Transmit Timestamp* from the concerned message's P7 nFAPI header is used instead of *RTP timestamp*
- *time of arrival at the PHY in microseconds* of the concerned P7 nFAPI message is used instead of *time of arrival in RTP timestamp units*

Tag	Field	Type	Description
N/A	Last SFN	uint16_t	The completed SFN at the PNF PHY instance that triggered the Timing Info message Value: 0 to 1023 Other values are reserved
N/A	Last slot	uint16_t	The completed slot (corresponding to Last SFN) at the PNF PHY instance that triggered the Timing Info message Value: 0 to 159

Tag	Field	Type	Description
			Other values are reserved
N/A	Time since last Timing Info	uint32_t	The number of ms since the last Timing Info was sent from this PNF PHY instance. Value: 0 to 4294967295
N/A	DL_TTI Jitter	uint32_t	The inter message jitter of the DL_TTI.request message reception in microseconds. Value: 0 to 4294967295
N/A	TxData Jitter	uint32_t	The inter message jitter of the TxData.request message reception in microseconds. Value: 0 to 4294967295
N/A	UL_TTI Jitter	uint32_t	The inter message jitter of the UL_TTI.request message reception in microseconds. Value: 0 to 4294967295
N/A	UL_DCI Jitter	uint32_t	The inter message jitter of the UL_DCI.request message reception in microseconds. Value: 0 to 4294967295
N/A	DL_TTI Latest Delay	int32_t	The latest delay offset in microseconds from the latest acceptable time for the DL_TTI.request as defined in the Timing Window in the PARAM.Response since the last transmission of the Timing Info Message. Note: Positive value is later than acceptable, negative value is earlier than acceptable Value: -2147483648 to 2147483647
N/A	TxData Latest Delay	int32_t	The latest delay offset in microseconds from the latest acceptable time for the TxData.request as defined in the Timing Window in the PARAM.Response since the last transmission of the Timing Info Message. Note: Positive value is later than acceptable, negative value is earlier than acceptable Value: -2147483648 to 2147483647
N/A	UL_TTI Latest Delay	int32_t	The latest delay offset in microseconds from the latest acceptable time for the UL_TTI.request as defined in the Timing Window in the PARAM.Response since the last transmission of the Timing Info Message.

Tag	Field	Type	Description
			Note: Positive value is later than acceptable, negative value is earlier than acceptable Value: -2147483648 to 2147483647
N/A	UL_DCI Latest Delay	int32_t	The latest delay offset in microseconds from the latest acceptable time for the UL_DCI.request as defined in the Timing Window in the PARAM.Response since the last transmission of the Timing Info Message. Note: Positive value is later than acceptable, negative value is earlier than acceptable Value: -2147483648 to 2147483647
N/A	DL_TTi Earliest Arrival	int32_t	The earliest arrival offset in microseconds from the latest time acceptable for the DL_TTI.request as defined in the Timing Window in the PARAM.Response since the last transmission of the Timing Info Message. Note: positive value is later than acceptable, negative value is earlier than acceptable Value: -2147483648 to 2147483647
N/A	TxData.request Earliest Arrival	int32_t	The earliest arrival offset in microseconds from the latest time acceptable for the TxData.request as defined in the Timing Window in the PARAM.Response since the last transmission of the Timing Info Message. Note: Positive value is later than acceptable, negative value is earlier than acceptable Value: -2147483648 to 2147483647
N/A	UL_TTI Earliest Arrival	int32_t	The earliest arrival offset in microseconds from the latest time acceptable for the UL_TTI.request as defined in the Timing Window in the PARAM.Response since the last transmission of the Timing Info Message. Note: Positive value is later than acceptable, negative value is earlier than acceptable Value: -2147483648 to 2147483647
N/A	UL_DCI Earliest Arrival	int32_t	The earliest arrival offset in microseconds from the latest time acceptable for the UL_DCI.request as defined in the Timing Window in the PARAM.Response since the last transmission of the Timing Info Message.

Tag	Field	Type	Description
			Note: Positive value is later than acceptable, negative value is earlier than acceptable Value: -2147483648 to 2147483647
N/A	Subcarrier Spacing	uint8_t	SubcarrierSpacing index as defined in TS38.211 ver 15.5.0 sec 4.2 0: 15 KHz 1: 30 KHz 2: 60 KHz 3: 120 KHz 4: 240 KHz

Table 4-3 Timing Info parameters

4.2 Transparent P7 messages

The P7 transparent messages are defined in section 2.3.2.9 There are no additional nFAPI TLVs or values defined for these messages. These FAPI defined messages are transported by nFAPI using the nFAPI headers specified in section 2.3.2.

4.2.1 Handling of Pointer Option in FAPI Messages

If a transparent FAPI messages contains a pointer option for specifying the payload portion of the message (for an example, see table 3-59 of [4]) then, the FAPI message, carried over nFAPI, shall not use the pointer option. The payload should be included in the FAPI message (for the same example, tag value should be 0 for the table 3-59 of [4]).

4.2.2 Extraction of FAPI Messages From an nFAPI Message

In the nFAPI protocol, the transparent FAPI message is formatted so that the FAPI message can be extracted from the nFAPI message simply by copying the message bits as shown in Figure 4-1.

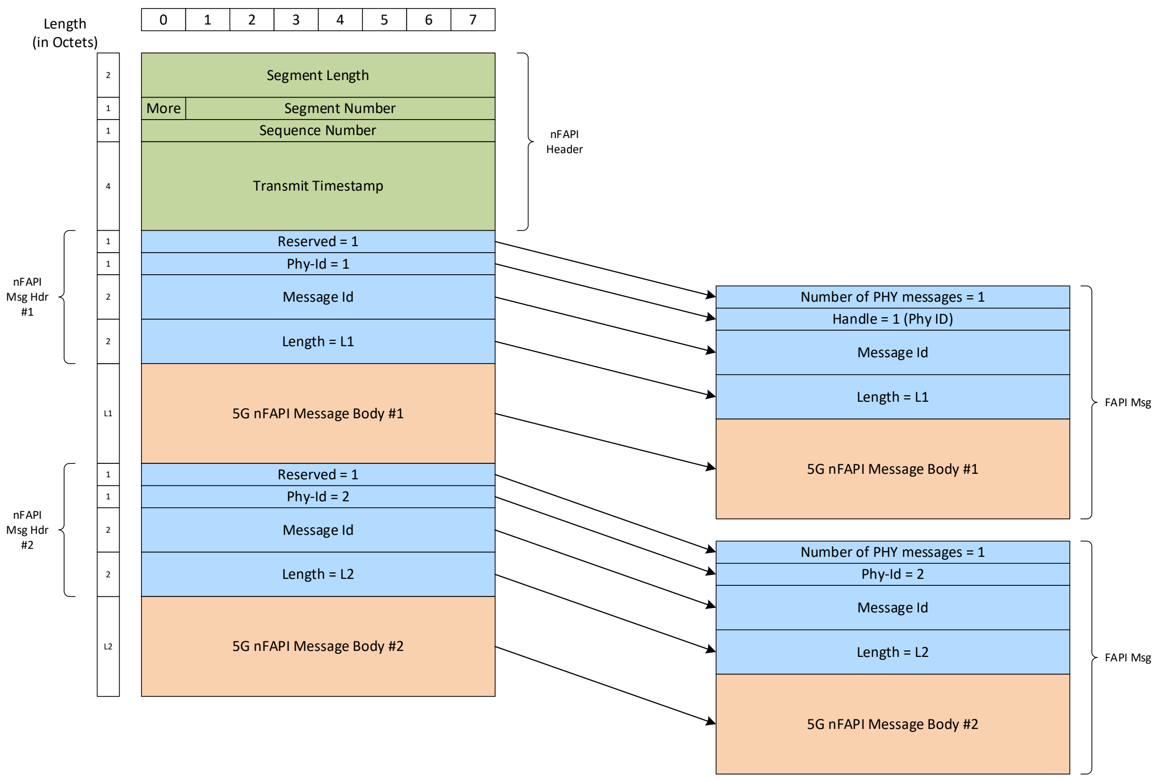


Figure 4–1 **nFAPI PDU carrying two transparent FAPI messages on the left and the two extracted FAPI messages on the right.**

5. P19 Messages

The P19 transparent messages are listed in section 2.3.2.9. There are no additional nFAPI TLVs or values defined for these messages. These FAPI defined messages are transported by nFAPI using the nFAPI headers specified in section 2.3.2.

In this version of the specification, there are no dedicated or combined nFAPI P19 messages.

Timing and Synchronization Considerations [Informative]

For 5G NR operations, a small cell requires frequency, phase and time synchronization. These requirements may vary considerably, depending on the services.

There are many methods by which such frequency and time synchronization may be provided, such as:

- GNSS (e.g., GPS)
- IEEE1588 (V2 (2008) / V2.1 (2019) Precision Time Protocol (PTP)*
- High accuracy internal oscillator

***Note**: While it is assumed that most deployments for NR will require TDD services, this is not mandatory. In the case of FDD deployment of NR small cells, frequency services (G.8265.1) is sufficient.

If a small cell is deployed outdoors, use of a local GNSS as timing source would be a solution, either with a PTP Grandmaster function in the small cell and PTP to the RU or, if highly integrated, then GNSS with a 1PPS interface into the local RU.

For indoor small cell deployment, a possible solution is to use an aggregation router with a Time Boundary Clock (T-BC) that can act as the Common clock to the small cell in the cluster. The T-BC will receive timing services over PTP from an upstream Primary Time Reference Clock (PRTC). Specific PTP profile choice is an implementation decision.

A summary of the absolute and relative Timing Alignment Error (TAE) as per 3GPP and ITU-T recommendation is given below.

Timing requirement	Maximum |TAE|, relative
MIMO and Tx Diversity	65ns
Intra-band contiguous carrier aggregation (FR2)	130ns
Intra-band non-contiguous carrier aggregation (FR2)	260ns
Intra-band contiguous carrier aggregation (FR1)	260ns
Intra-band non-contiguous carrier aggregation (FR1)	3μs
Inter band carrier aggregation	3μs

Table 0-1 5G NR timing requirements

Please refer to 3GPP TS 38.104, and ITU-T recommendations: G.703, G.803, G.8262, G.811, G.8272, G8262, G.8265.1, G.8275.1 and G.8275.2 for more information.

Small Cell Forum plans to publish a whitepaper on synchronization for 5G NR small cells with detailed recommendations.

Signalling Example [Informative]

This appendix illustrates how the normative nFAPI procedure description from this specification may be reflected in nFAPI/FAPI /internal signalling interworking for a PNF that controls multiple PHY instances.

Nothing in this informative section and shall be interpreted to constrain PNF implementation with respect to the order in which messages are sent, internal message routing, construction, triggering, or even existence of particular messages.

For the call flows in this section:

- Calls for **P5 dedicated** messages terminate between VNF and PNF
  - PHY ID is expected to be set to zero for these messages.
  - These calls may result in separate internal calls terminated between PNF and PHY instances. These are shown separately in *[italicized bracketed text]*
- Calls for **P5 Combined and Transparent** messages, as well as **P7 transparent nFAPI** messages terminate between VNF and the PHY Instance addressed by the non-zero PHY ID.
  - PNF logically routes the FAPI payload of these messages to and from the appropriate PHY Instance identified by the (PNF IP address, PNF port, PHY ID)-tuple.
  - The routing of FAPI payloads is indicated via circles ■ in the call flow
- Calls for **P7 dedicated messages** terminate between VNF and the PHY Instance addressed by the non-zero PHY ID
  - PNF triggers internal calls to/from the addressed PHY instance identified by the (PNF IP address, PNF port, PHY ID)-tuple.
  - The triggered internal calls are illustrated in *[italicized bracketed text]*. These are not normative, and only facilitate illustration in this section.
  - PNF's triggering role is indicated via diamonds ◇ in the call flow

The PNF Start procedure is illustrated in Figure 2-5, along with slot-based operation. The nFAPI PNF initialization procedure is followed. It includes:

- The nFAPI PNF PARAM procedure
- The nFAPI PNF CONFIG procedure
- The nFAPI PNF START procedure, including actions to create PHY instances

The nFAPI PNF initialization procedure completes when the VNF receives the `PNF_START.response` message, and is followed by *FEU Query, Configuration and PHY Definition*, which consists of the following procedures detailed in the 0 example:

- nFAPI FEU Profile discovery
- nFAPI PHY Profile discovery
- RF Query and Configuration
- DFE Query
- PHY Definition

PHY IDs are only defined at the completion of the above procedures, though they are not yet in any operable PHY state.

PHY Instantiation: FEU initialization procedure follows, as detailed in 0, which triggers the PHY Instantiation

- PHYs with IDs > 0 are now instantiated and in PHY IDLE mode

Following PHY instantiation is the nFAPI PHY initialization procedure, for each PHY instance:

- This includes the nFAPI PHY PARAM procedure.
- The nFAPI `PARAM.request` routes to the FAPI `PARAM.request` message
- The FAPI `PARAM.response` routes to the nFAPI `PARAM.response` message.

The nFAPI PHY initialization procedure completes when the VNF receives the `PARAM.response` message, and is followed by the nFAPI PHY CONFIG procedure, for each PHY instance:

- The nFAPI `CONFIG.request` routes to the FAPI `CONFIG.request` message
- The FAPI `CONFIG.response` routes to the nFAPI `CONFIG.response` message

The nFAPI PHY CONFIG procedure completes when the VNF receives the `CONFIG.response` message, and is followed by the nFAPI PHY START procedure, for each PHY instance:

- The nFAPI `START.request` routes the FAPI `START.request` message
- Internal PHY/PNF messaging triggers the generation of nFAPI `START.response` message, with Error Code = OK.

The nFAPI PHY START procedure completes when the VNF receives nFAPI `START.response` message, and is followed by the nFAPI Node Sync procedure:

- The nFAPI `DL Node Sync` triggers PNF & PHY internal procedures to communicate t1 to the PHY instance and determine t2, t3 from the PHY instance, as well as to communicate the slot offset to the PHY instance.
- Internal PHY & PNF procedures deliver t2, t3 to PNF to trigger the nFAPI `UL Node Sync` towards VNF

The nFAPI Node Sync procedure completes when VNF receives nFAPI `UL Node Sync` message and is followed by regular slot:

- P7 transparent slot messages (like `DL_TTI.request`, in the call flow) route transparently between nFAPI and FAPI at PNF.
- Receive window statistics for each PHY instance trigger the generation of nFAPI `Timing Info` message towards the VNF

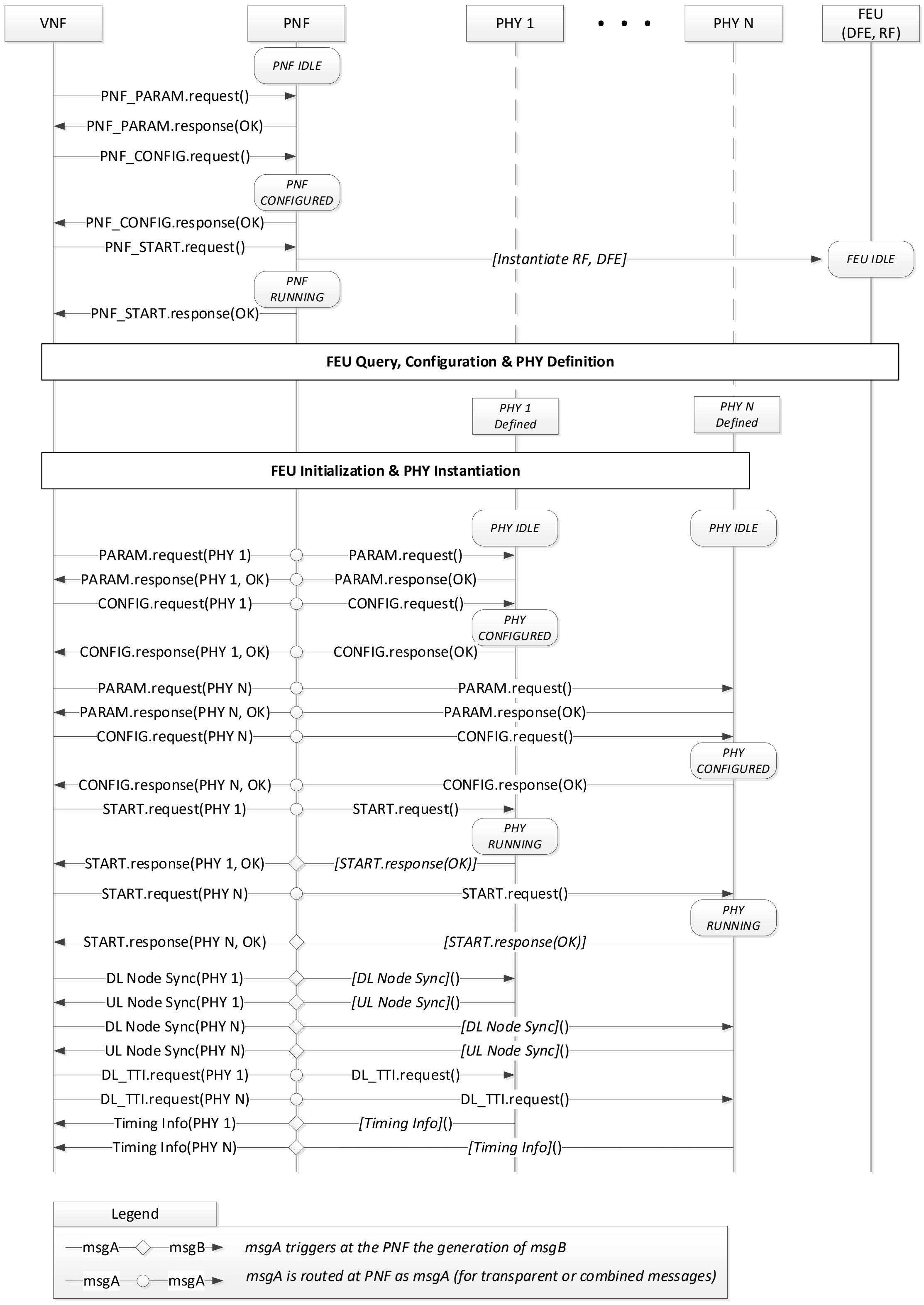


Figure 5–1 PNF Start and Slot Operation Procedure Illustration

Restart/recovery from node failure (Informative)

This section describes the handling of a few of the error scenarios that may occur while nFAPI nodes (S-DU or S-RU) are in operation.

S-DU failure

In case of a failure of S-DU due to a crash or other operational error, S-RU detects this condition from SCTP transport or through some other means. S-RU should take following actions once it detects that S-DU is not functional:

- stop RF Transmission to avoid any interference
- return to Idle state
- Re-establish SCTP connection and send `PNF_START.indication`.

S-RU failure

If S-RU crashes or stops working, S-DU detects this condition from SCTP transport or through some other means. S-DU and S-RU should take following actions once S-RU stop functioning:

- S-RU should re-establish SCTP connection and move back to Idle state after it is operational again
- S-RU should send `PNF_START.indication`
- S-DU should only initiate PNF Initialization procedure towards S-RUs that stopped functioning. Other S-RUs that are operational should not be impacted.

Front End Unit Interactions [Informative]

For the control, discovery and measurements relating to the RF and digital front end, *5G FAPI specification* has introduced the P19 interface that is dedicated to the control and the configuration of the FEU [10]. P19 defines RF front end sub-components:

- **Digital Front End (DFE) Block**: consists of multiple digital NB Narrowband (NB) and Wideband (WB) Rx and Tx chains
- **RF Block**: 'is composed of T number of RF chains that operate in the Tx direction and R number of RF chains that operate in the Rx direction' [10].
- **Analog Beam forming (ABF) Block**: responsible for Analog Beamforming, by 'adjusting the analog gains and the phases of the individual antenna elements'.

This appendix illustrates how the normative nFAPI procedure description from this specification may be reflected in nFAPI/FAPI/internal signalling interworking for a PNF that controls DFE, RF and multiple PHY instances.

Nothing in this informative section shall be interpreted to constrain PNF implementation with respect to the order in which messages are sent, internal message routing, construction, triggering, or even existence of particular messages.

For the call flows in this section, see Figure D-1. The symbols and conventions are similar to Appendix B.

The **PNF Query and Configuration** procedures are the same as in Appendix B, not illustrated here.

- The nFAPI PNF PARAM procedure
- The nFAPI PNF CONFIG procedure

The nFAPI **PNF initialization** procedure completes when the VNF receives the `PNF_START.response` message. Part of it:

- Any DFE components are instantiated
- Any RF components are instantiated

The nFAPI FEU and PHY Profile discovery helps VNF discover essential PHY components and PHY profiles

- Based on FAPI Query of PHY ID #0

RF Query and Configuration uses P19 messages to transit RF 1 into CONFIGURED state:

- P19 RF Query Procedure (PARAM exchange)
- P19 RF Configuration Procedure (CONFIG exchange)

DFE Query uses P19 messages to discover DFE Profiles:

- P19 DFE Query Procedure (PARAM exchange)

PHY Definition configures the allocation of PHY IDs and their baseband port mappings

- PHY Configuration procedure for PHY Profile Selection, towards PHY ID#0

DFE Configuration uses P19 messages to transit DFE 1 into CONFIGURED state:

- P19 DFE Configuration Procedure (CONFIG exchange)

RF and DFE Initialization use P19 messages to transit RF 1 and DFE 1 into RUNNING state:

- P19 DFE Configuration Procedure (START exchange)

PHY Instantiation follows successful initialization of all FEU components:

- PHY ID 1 enters IDLE state.

PHY Query, Configuration and Initialization follow the signaling in Appendix B, and is not further repeated here

The call flow illustrates the following slot-based procedures, for one slot

- P7 `DL_TTI.request`
- P19-S DFE `SCHEDULE.request`/`response`
- P19-S RF `SCHEDULE.request`/`response` (could equally be an `FEU.SELECT_BEAM.request/response`)

Note: The presence of P19-S `.response` messages is determined by FEU capability and VNF configuration of FEU for these messages.

P7 and P19 `Timing Info` procedures may be triggered periodically, or if Rx Windows are missed by slot-based procedures. The call flow illustrates a case where FEU is configured to trigger `Timing Info` reports separately from PHY.

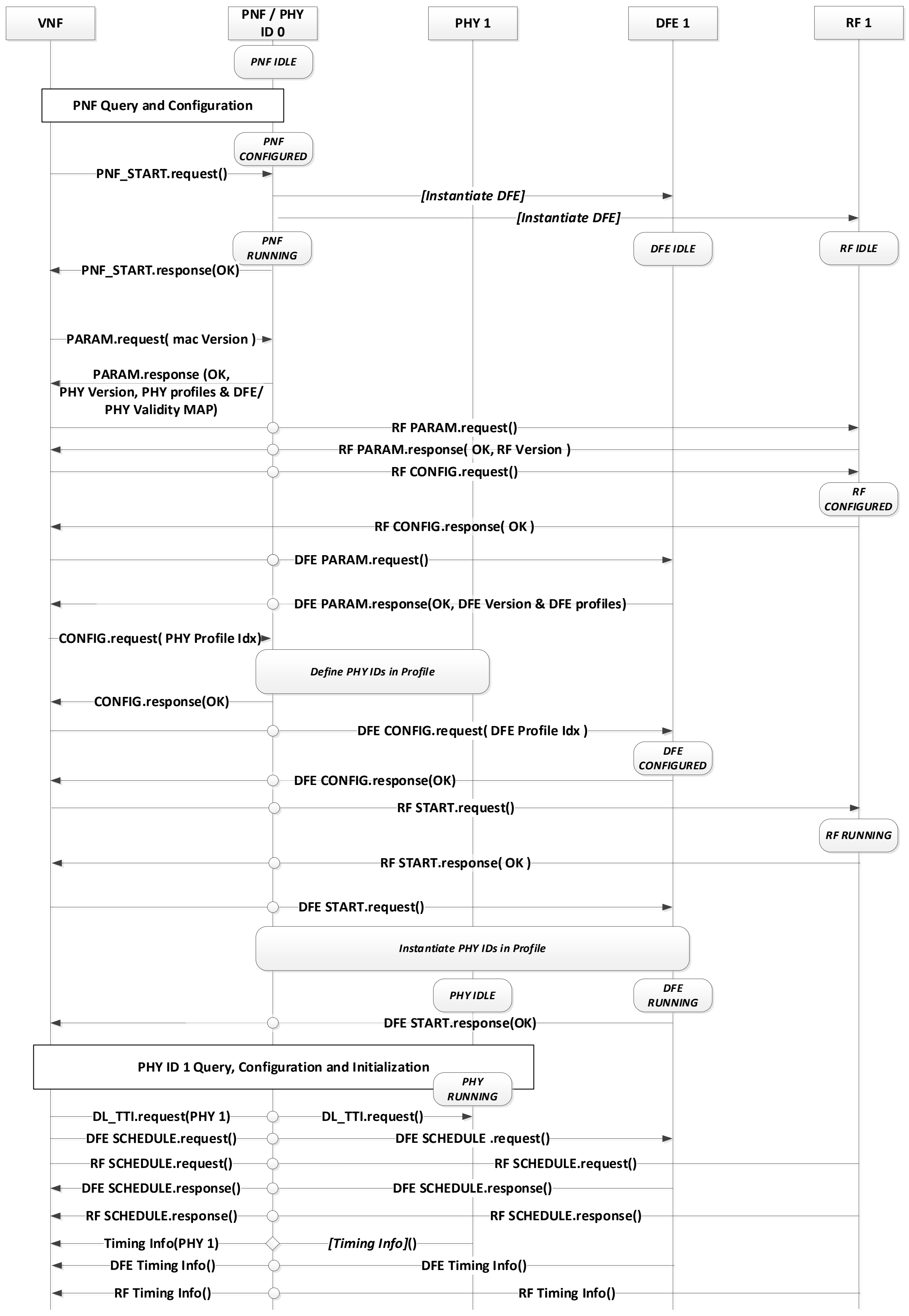


Figure D-5–2 PNF Start and Slot Operation Procedure Illustration

Non-ideal fronthaul deployment considerations [Informative]

Non-ideal fronthaul: When the nFAPI fronthaul uses transport with higher latencies (up to 30 ms), high jitter or limited throughput conditions, the fronthaul is deemed non-ideal.

Deployment of small cells with such constrained fronthaul transport may be allowed in some scenarios of low mobility and in some select indoor/outdoor locations. It is possible to achieve required throughput with relaxed KPIs by fine tuning some of the configuration parameters and implementation in S-DU and S-RU nodes.

HARQ: The HARQ processing in the S-DU can be adapted for non-ideal fronthaul scenarios by tuning the K0, K1 and K2 parameters. For more information of these parameters refer to [9].

RACH: In case of non-ideal fronthaul, 5G NR RACH timers can be configured accordingly to ensure no impacts to UEs attempting new connection or in handover conditions.

Other fine-tuning possible include link adaptation and scheduling with the known or measured latencies in uplink and downlink transmissions.

Impacts to P5 and P7 interfaces: SCTP protocol timers allow enough latencies to suit P5 over non-ideal fronthaul. Any connection failures are also auto recovered as per timers defined SCTP RFC, the values of which fall within the limits.

The delay management built in nFAPI P7 allow for the uplink and downlink latencies to be handled accordingly.

References

[1] SCF 106, "Virtualization for small cells: Overview", Small Cell Forum, June 2016, version 106.07.01

[2] SCF 159, "Small cell virtualization functional splits and use cases", Small Cell Forum, January 2016, version 159.07.02

[3] SCF 082, "nFAPI and FAPI specifications", Small Cell Forum, May 2015, version 082.09.05

[4] SCF 222. "5G FAPI: PHY API", Small Cell Forum, July 2022, version 222.04

[5] 3GPP TR 38.816, "Study on Central Unit (CU) - Distributed Unit (DU) lower layer split for NR", January 2018, version 15.0.0

[6] SCF 223. "5G FAPI: RF and Digital Frontend Control API", Small Cell Forum, May 2021, version 223.10.02

[7] SCF 224, "5G FAPI: Network Monitor Mode API", Small Cell Forum, March 2020, version 224.10.01

[8] 3GPP TS 38.104, "NR Base-station (BS) Radio Transmission and Reception", July 2020, version 16.4.0

[9] IETF RFC 4960, "Stream Control Transmission Protocol", September 2007

[10] 3GPP TS 38.455, "NG-RAN; NR Positioning Protocol A (NRPPa)", July 2020, version 16.0.0

[11] IEEE 802-2001, "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture", March 2001

[12] eCPRI 2.0 specification, http://www.cpri.info/spec.html

[13] 3GPP TS 38.214, "NR Physical layer procedures for data", July 2020, version 16.2.0