# webwxgetmsgimg (7).jpegO-RAN.WG6.AAL-GAnP-R004-v11.00

*Technical Specification*

1

2

**O-RAN Working Group 6 (Cloudification and Orchestration Work Group)**

**O-RAN Acceleration Abstraction Layer General Aspects and Principles**

Copyright © 2025 by the O-RAN ALLIANCE e.V.

The copying or incorporation into any other work of part or all of the material available in this specification in any form without the prior written permission of O-RAN ALLIANCE e.V. is prohibited, save that you may print or download extracts of the material of this specification for your personal use, or copy the material of this specification for the purpose of sending to individual third parties for their information provided that you acknowledge O-RAN ALLIANCE as the source of the material and that you inform the third party that these conditions apply to them and that they must comply with them.

O-RAN ALLIANCE e.V., Buschkauler Weg 27, 53347 Alfter, Germany

Register of Associations, Bonn VR 11238, VAT ID DE321720189

# 1 Contents

2

1. [List of Figures 4](#_bookmark0)
2. [Foreword 5](#_bookmark1)
3. [Modal verbs terminology 5](#_bookmark2)
4. [1. Scope 6](#_bookmark3)
5. [2. References 7](#_bookmark4)
6. [2.1 Normative References 7](#_bookmark5)
7. [2.2 Informative references 7](#_bookmark16)
8. [3. Definitions of terms, symbols and abbreviations 9](#_bookmark18)
9. [3.1 Terms 9](#_bookmark19)
10. [3.2 Symbols 10](#_bookmark20)
11. [3.3 Abbreviations 10](#_bookmark21)
12. [4. General Aspects 11](#_bookmark22)
13. [4.1 Hardware Acceleration 11](#_bookmark23)
14. [4.2 AAL Architecture and concepts 12](#_bookmark25)
15. [4.2.1 Overview 12](#_bookmark26)
16. [4.2.2 AAL Architecture 13](#_bookmark27)
17. [4.2.3 HW Accelerator Manager 15](#_bookmark30)
18. [4.2.4 AAL Interfaces 16](#_bookmark32)
19. [4.2.5 AAL Deployment in O-Cloud environments 18](#_bookmark34)
20. [4.3 Relationship with Standards 20](#_bookmark39)
21. [4.3.1 Relationship with ETSI 20](#_bookmark40)
22. [5. AAL Interface definition General Principles and Requirements 22](#_bookmark41)
23. [5.1 General Principles 22](#_bookmark42)
24. [5.1.1 Extensibility 22](#_bookmark43)
25. [5.1.2 Interrupt and Poll Mode 22](#_bookmark44)
26. [5.1.3 HW Independence 22](#_bookmark45)
27. [5.1.4 Discovery and Configuration 22](#_bookmark46)
28. [5.1.5 Multiple AAL-LPU Support 22](#_bookmark47)
29. [5.1.6 AAL offload capabilities 23](#_bookmark49)
30. [5.1.7 Look-aside Acceleration Model 23](#_bookmark50)
31. [5.1.8 Inline Acceleration Model 24](#_bookmark52)
32. [5.1.9 AAL Application interface Concurrency and Parallelism 25](#_bookmark55)
33. [5.1.10 Separation of Control and User Plane AAL Application interface APIs 25](#_bookmark56)
34. [5.1.11 Support of Versatile Acceleration Payload 25](#_bookmark57)
35. [5.1.12 Support of Different Transport Mechanisms 25](#_bookmark58)
36. [5.1.13 AAL API namespace 26](#_bookmark59)
37. [5.1.14 Chaining of AAL Profiles 26](#_bookmark60)
38. [5.2 High-PHY Profile Specific Principles 28](#_bookmark65)
39. [5.2.1 Separation of Cell and Slot Level Parameter Configurations 28](#_bookmark66)
40. [5.2.2 SFN/slot-based Synchronization 28](#_bookmark67)
41. [5.2.3 Compatibility with O-RAN FH interface 29](#_bookmark68)
42. [5.2.4 Inline Profile for High-PHY Stack 29](#_bookmark69)
43. [6. AAL-LPU Principles 30](#_bookmark71)
44. [6.1 Overview 30](#_bookmark72)
45. [6.2 LPU Deployment and Operation 31](#_bookmark73)
46. [6.2.1 Example AAL-LPU Mapping 31](#_bookmark74)
47. [6.2.2 Statistics 34](#_bookmark81)
48. [6.2.3 Memory Management 34](#_bookmark82)
49. [6.2.4 Run Time Configurations 34](#_bookmark83)

© 2025 by the O-RAN ALLIANCE e.V. Your use is subject to the copyright statement on the cover page of this specification. 2

1. [6.2.5 AAL Profile(s) offload, processing status query and processed data retrieval 35](#_bookmark84)
2. [6.2.6 AAL-LPU Exposure 35](#_bookmark86)
3. [6.2.7 Accelerator configuration options between IMS & FOCOM 37](#_bookmark91)
4. [7. AAL Profiles 39](#_bookmark92)
5. [7.1 O-DU AAL Profiles 39](#_bookmark93)
6. [7.1.1 O-DU Protocol Stack Reference 39](#_bookmark94)
7. [7.1.2 O-DU Protocol Stack Reference for mMTC 43](#_bookmark97)
8. [7.1.3 O-DU AAL Profile Definitions 47](#_bookmark100)
9. [7.2 O-CU AAL Profiles 70](#_bookmark139)
10. [Annex (informative): Change History 71](#_bookmark140)

11

12

# 1 List of Figures

1. [Figure 4.1-1: Example illustration of the effect of hardware acceleration on functional compute performance 12](#_bookmark24)
2. [Figure 4.2.2-1: High Level AAL Architecture Diagram 14](#_bookmark28)
3. [Figure 4.2.2-2: AAL Resource Relationship and Cardinality 15](#_bookmark29)
4. [Figure 4.2.3-1 Example HAM software deployment scenarios 16](#_bookmark31)
5. [Figure 4.2.4-1: AAL Application Common and profile APIs 17](#_bookmark33)
6. [Figure 4.2.5-1: Accelerator APIs/Libraries in Container (left) and Virtual Machine Implementations (right) 18](#_bookmark35)
7. [Figure 4.2.5.1-1: Example AAL Implementation Software deployment contained within O-Cloud Platform Software](#_bookmark36)
8. [only 19](#_bookmark36)
9. [Figure 4.2.5.1-2: Example AAL Implementation Software Deployment split between O-Cloud Platform Software and](#_bookmark37)
10. [NF Deployment 19](#_bookmark37)
11. [Figure 4.2.5.1-3: Example AAL Implementation Software Deployment split between O-Cloud Platform Software and](#_bookmark38)
12. [NF Deployment in a virtual machine environment 20](#_bookmark38)
13. [Figure 5.1.5-1: Logical Representation of AAL Application interface support for multiple AAL-LPUs 23](#_bookmark48)
14. [Figure 5.1.7-1: AAL Application interface look-aside acceleration model - Data flow 23](#_bookmark51)
15. [Figure 5.1.8-1: AAL Application interface inline acceleration model - Data flow 24](#_bookmark53)
16. [Figure 5.1.8-2: User plane dataflow paths in look-aside and inline acceleration architectures. 25](#_bookmark54)
17. [Figure 5.1.14-1: Data flow through unchained AAL Profiles 26](#_bookmark61)
18. [Figure 5.1.14-2: Data flow through chained AAL Profiles 27](#_bookmark62)
19. [Figure 5.1.14-3: Dataflow through chained lookaside HW Accelerator for consecutive Hi-PHY functions 28](#_bookmark63)
20. [Figure 5.1.14-4: Dataflow through chained lookaside HW Accelerator for consecutive and non-consecutive PHY. 28](#_bookmark64)
21. [Figure 5.2.4-1: Partial Inline Model for AAL Hi-PHY Profile 29](#_bookmark70)
22. [Figure 6.2.1-1: Scenario 1: A single AAL LPU exposes a single AAL Profile Queue used by a single AAL Application.](#_bookmark75)

24 [31](#_bookmark75)

1. [Figure 6.2.1-2: A single HW Accelerator supporting two LPU's each assigned to individual AAL Applications 32](#_bookmark76)
2. [Figure 6.2.1-3: Two HW Accelerators each supporting a single AAL-LPU assigned to a single AAL Application 32](#_bookmark77)
3. [Figure 6.2.1-4: A single AAL LPU exposes two AAL Profile Queues used by a single AAL Application 33](#_bookmark78)
4. [Figure 6.2.1-5: A single AAL LPU supporting two AAL Profile Instances exposes two AAL Profile Queues used by a](#_bookmark79)
5. [single AAL Application 33](#_bookmark79)
6. [Figure 6.2.1-6: AAL-LPU Mapping example showing chained profile support 34](#_bookmark80)
7. [Figure 6.2.5.1- 1 Various AAL Bypass scenarios 35](#_bookmark85)
8. [Figure 6.2.6.2.1-1: Example AAL-LPU and profile supported 36](#_bookmark87)
9. [Figure 6.2.6.2.1-2: Example HW Accelerator configuration 36](#_bookmark88)
10. [Figure 6.2.6.2.1-3: Example assignment of AAL-LPUs and supported profiles to POD 36](#_bookmark89)
11. [Figure 6.2.6.2.2-1: Example AAL Application configured AAL-Profile-Instances 37](#_bookmark90)
12. [Figure 7.1.1-1: O-DU PHY processing blocks for 5G NR Downlink 39](#_bookmark95)
13. [Figure 7.1.1-2: O-DU PHY processing blocks for 5G NR Uplink 41](#_bookmark96)
14. [Figure 7.1.2-1: O-DU PHY processing blocks for mMTC Downlink 44](#_bookmark98)
15. [Figure 7.1.2-2: O-DU PHY processing blocks for mMTC Uplink 46](#_bookmark99)
16. [Figure 7.1.3.2.1-1: AAL\_MU-MIMO\_PRECODER\_WEIGHTS\_CALC 48](#_bookmark101)
17. [Figure 7.1.3.2.1-2: Example AAL\_MU-MIMO\_PRECODER\_WEIGHTS\_CALC use. 48](#_bookmark102)
18. [Figure 7.1.3.2.2-1: AAL\_FFT 49](#_bookmark103)
19. [Figure 7.1.3.2.2-2: AAL\_FFT example for SRS Processing 50](#_bookmark104)
20. [Figure 7.1.3.3.1-1: AAL\_PDSCH\_FEC Profile 51](#_bookmark105)
21. [Figure 7.1.3.3.2-1: AAL\_PDSCH\_HIGH-PHY Profile 52](#_bookmark109)
22. [Figure 7.1.3.3.3-1: AAL\_PDCCH\_HIGH-PHY Profile 53](#_bookmark111)
23. [Figure 7.1.3.3.4-1: AAL\_PBCH\_HIGH-PHY Profile 54](#_bookmark113)
24. [Figure 7.1.3.3.5-1: AAL\_CSI-RS\_HIGH-PHY Profile 55](#_bookmark115)
25. [Figure 7.1.3.3.6-1: AAL\_PT-RS-DL\_HIGH-PHY Profile 56](#_bookmark117)
26. [Figure 7.1.3.3.7-1AAL\_DOWNLINK\_ HIGH-PHY Profile 57](#_bookmark118)
27. [Figure 7.1.3.4.1-1: AAL\_PUSCH\_FEC Profile 58](#_bookmark119)
28. [Figure 7.1.3.4.2-1 : AAL\_PUSCH\_HIGH-PHY Profile 59](#_bookmark121)
29. [Figure 7.1.3.4.3.1-1: AAL\_PUCCH\_HIGH-PHY Profile (PUCCH format 0) 60](#_bookmark123)
30. [Figure 7.1.3.4.3.2-1: AAL\_PUCCH\_HIGH-PHY Profile (PUCCH format 1) 61](#_bookmark125)
31. [Figure 7.1.3.4.3.3-1: AAL\_PUCCH\_HIGH-PHY Profile (PUCCH format 2/3/4) 62](#_bookmark127)
32. [Figure 7.1.3.4.4-1: AAL\_PRACH\_HIGH-PHY Profile 63](#_bookmark129)
33. [Figure 7.1.3.4.5-1: AAL\_SRS\_HIGH-PHY Profile 64](#_bookmark131)
34. [Figure 7.1.3.4.6-1: AAL\_PT-RS-UL\_HIGH-PHY profile 65](#_bookmark133)
35. [Figure 7.1.3.4.7-1: AAL\_UPLINK\_ HIGH-PHY Profile 66](#_bookmark134)
36. [Figure 7.1.3.5.1-1: AAL\_NPDSCH\_FEC Profile 67](#_bookmark135)
37. [Figure 7.1.3.5.2-1: AAL\_NPDCCH\_FEC Profile 68](#_bookmark136)
38. [Figure 7.1.3.5.3-1: AAL\_NPBCH\_FEC Profile 69](#_bookmark137)
39. [Figure 7.1.3.5.4-1: AAL\_NPUSCH\_FEC Profile 70](#_bookmark138)

6

# 7 Foreword

1. This Technical Specification (TS) has been produced by W6 of the O-RAN Alliance.
2. The content of the present document is subject to continuing work within O-RAN and may change following formal O-
3. RAN approval. Should the O-RAN Alliance modify the contents of the present document, it will be re-released by O-
4. RAN with an identifying change of version date and an increase in version number as follows:
5. version xx.yy.zz
6. where:
7. xx: the first digit-group is incremented for all changes of substance, i.e. technical enhancements, corrections,
8. updates, etc. (the initial approved document will have xx=01). Always 2 digits with leading zero if needed.
9. yy: the second digit-group is incremented when editorial only changes have been incorporated in the document.
10. Always 2 digits with leading zero if needed.
11. zz: the third digit-group included only in working versions of the document indicating incremental changes during
12. the editing process. External versions never include the third digit-group. Always 2 digits with leading zero if
13. needed.

# 21 Modal verbs terminology

1. In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and
2. "**cannot**" are to be interpreted as described in clause 3.2 of the O-RAN Drafting Rules (Verbal forms for the expression
3. of provisions).
4. "**must**" and "**must not**" are **NOT** allowed in O-RAN deliverables except when used in direct citation. 26

# 1 1. Scope

1. This document defines O-RAN O-Cloud hardware accelerator interface functions and protocols for the O-RAN AAL
2. interface. The document studies the functions conveyed over the interface, including configuration and management
3. functions, procedures, operations and corresponding solutions, and identifies existing standards and industry work that
4. can serve as a basis for O-RAN work.

# 1 2. References

## 2.1 Normative References

1. References are either specific (identified by date of publication and/or edition number or version number) or
2. non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the
3. referenced document (including any amendments) applies.
4. NOTE: While any hyperlinks included in this clause were valid at the time of publication, O-RAN cannot
5. guarantee their long-term validity.
6. The following referenced documents are necessary for the application of the present document.

|  |  |  |
| --- | --- | --- |
| 9 |  | |
| 10 | [1] | O-RAN WG1 Architecture Description |
| 11 | [2] | O-RAN WG1 OAM Architecture |
| 12 | [3] | Void |
| 13 | [4] | ETSI GS NFV-IFA 002: "Network Functions Virtualization (NFV) Release 2; Acceleration Technologies; |
| 14 |  | VNF Interfaces Specification" |
| 15 | [5] | [ETSI](https://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/019/03.01.01_60/gs_NFV-IFA019v030101p.pdf) GS NFV-IFA 019: "Network Function Virtualization (NFV); Acceleration Technologies; Acceleration |
| 16 |  | Resource Management Interface Specification; Release 3" |
| 17 | [6] | [5G; NR; Physical Channels and Modulation 3GPP TS 38.211 v15.2.0 Release 15](https://www.etsi.org/deliver/etsi_ts/138200_138299/138211/15.02.00_60/ts_138211v150200p.pdf) |
| 18 | [7] | [5G; NR; Multiplexing and Channel Coding 3GPP TS 38.212 v15.2.0 Release 15](https://www.etsi.org/deliver/etsi_ts/138200_138299/138211/15.02.00_60/ts_138211v150200p.pdf) |
| 19 | [8] | [LTE; E-UTRA Physical Channels and Modulation 3GPP TS 36.211 v15.2.0 Release 15](https://www.etsi.org/deliver/etsi_TS/136200_136299/136211/15.02.00_60/ts_136211v150200p.pdf) |
| 20 | [9] | [LTE; E-UTRA Multiplexing and Channel Coding 3GPP TS36.212 v15.2.1 Release 15](https://www.etsi.org/deliver/etsi_ts/136200_136299/136212/15.02.01_60/ts_136212v150201p.pdf) |
| 21 | [10] | ETSI GS [NFV-IFA 001:](https://docbox.etsi.org/ISG/NFV/Open/Publications_pdf/Specs-Reports/NFV-IFA%20001v1.1.1%20-%20GS%20-%20Acceleration%20-%20UCs%20report.pdf) "Network Functions Virtualisation (NFV); Acceleration Technologies; Report on |
| 22 |  | Acceleration Technologies & Use Cases" |
| 23 | [11] | O-RAN WG6 O2 General Aspects and Principles |
| 24 | [12] | O-RAN WG6 Cloudification and Orchestration Use Cases and Requirements for O-RAN Virtualized RAN |
| 25 | [13] | ETSI GS NFV-IFA [011:](https://docbox.etsi.org/ISG/NFV/Open/Publications_pdf/Specs-Reports/NFV-IFA%20011v4.3.1%20-%20GS%20-%20VNF%20Packaging%20Spec.pdf) "Network Functions Virtualisation (NFV) Release 4; Management and Orchestration; |
| 26 |  | VNF Descriptor and Packaging Specification" |
| 27 | [14] | ETSI GR NFV-IFA 046: "Network Functions Virtualisation (NFV) Release 5; Architectural Framework; |
| 28 |  | Report on NFV support for virtualisation of RAN" |
| 29 | [15] | O-RAN WG6 AAL Common API R003 |
| 30 | [16] | O-RAN WG4 Control, User and Synchronization (CUS) Plane Specification |
| 31 |  |  |
| 32 | 2.2 | Informative references |

1. References are either specific (identified by date of publication and/or edition number or version number) or
2. non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the
3. referenced document (including any amendments) applies.
4. NOTE: While any hyperlinks included in this clause were valid at the time of publication, O-RAN cannot
5. guarantee their long-term validity.
6. The following referenced documents are not necessary for the application of the present document, but they assist the
7. user with regard to a particular subject area.
   1. [i.1] [Vocabulary for 3GPP Specifications (TR21.905)](https://www.etsi.org/deliver/etsi_tr/121900_121999/121905/04.05.00_60/tr_121905v040500p.pdf) 2

3

# 1 3. Definitions of terms, symbols and abbreviations

## 3.1 Terms

* 1. For the purpose of this document the terms and definitions given in O-RAN WG6 Cloudification and Orchestration Use
  2. Cases (UC) and Requirements for O-RAN Virtualized RAN [13], ETSI GS NFV-IFA 002 [[4],](#_bookmark7) and the following apply:
  3. **Hardware (HW) Accelerator (HWA)** is a specialized HW implementation that can offload processing from
  4. application(s) running on a General-Purpose Processor. The Hardware (HW) Accelerator is a physical Managed Element
  5. as defined in [13].
  6. NOTE: Examples of Hardware Accelerators include ASIC, FPGA, DSP and GPU.
  7. NOTE: Throughout this document, the term “Accelerator” and “Hardware (HW) accelerator” are used
  8. interchangeably.
  9. **Acceleration Abstraction Layer (AAL)** specifies a common and consistent set of interfaces used by different entities to
  10. enable interaction with different types of HW Accelerators within an O-Cloud instance.
  11. **AAL Implementation** is a realization of an AAL including but not limited to the software libraries, drivers and the
  12. Hardware Accelerator
  13. **Accelerated Function (AF)** is a representation of a workload building block that an accelerator processes on behalf of
  14. an AAL Application within an O-RAN Network Function in an O-Cloud [[1].](#_bookmark6)
  15. **AAL Application (AAL-App)** is defined as a workload that can offload Accelerated Functions to AAL-LPU(s).
  16. NOTE: Unless explicitly noted, the term Application refers to an AAL Application in AAL specifications.
  17. NOTE: Unless explicitly noted, the terms Application, NF Application, L2 Application, VNF/CNF, or NF workload
  18. accessing the AAL in figures of this present document refers to an AAL Application.
  19. **AAL Profile(s) (AAL-Profile)** specify one or more Accelerated Functions that an accelerator processes on behalf of an
  20. AAL Application within an O-RAN Network Function in an O-Cloud [[1].](#_bookmark6)
  21. **AAL Operations** actions supported by the AAL interface.
  22. **AAL Profile Instance (AAL-Profile-Instance)** is an executing instance of an AAL profile that can be used by an AAL
  23. Application via the AAL interface. The AAL-Profile-Instance executes within an AAL-LPU execution environment.
  24. **AAL Logical Processing Unit (AAL-LPU)** is a logical representation of resources within an instance of a HW
  25. Accelerator (example: there can be multiple processing units or subsystems on a hardware accelerator, or resource
  26. partitioning (hard – dedicated resources, soft – soft resources) and these can be logically represented as a AAL Logical
  27. Processing Unit)
  28. **AAL Profile Queue** is a logical grouping construct and may be used by the AAL Application to group operations together.
  29. For example, AAL Profile Queues may access specific resources (compute, I/O) of an AAL-LPU executing specific AAL
  30. Profile Instances(s).
  31. **AAL Profile Queue ID** is a unique index used to designate the AAL Profile Queue in function exposed by specific AAL
  32. profile.
  33. NOTE: An AAL Profile Queue or an AAL Profile Queue ID does not reflect a HW design or an AAL Implementation
  34. specification
  35. **HW Accelerator Manager (HAM)** is an acceleration management function, that provides management capabilities for
  36. the HW Accelerator(s) and the AAL-LPU's in the O-Cloud Node. Management capabilities include but not limited to
  37. lifecycle management, configuration, updates/upgrades and failure handling of the HWA and the LPUs. The HAM may
  38. use the services of an external software repository to securely transfer approved software artifacts, for example FW/SW
  39. images, binaries, libraries and configuration data. The specification of a software repository and the communication with
  40. the HAM are outside the scope of O-RAN specifications. The HW Accelerator Manager is considered an O-Cloud
  41. Platform Software.

44

## 3.2 Symbols

1. Void

## 3.3 Abbreviations

1. For the purposes of the present document, the abbreviations given in [[i.1]](#_bookmark17) and the following apply. An abbreviation
2. defined in the present document takes precedence over the definition of the same abbreviation, if any, in [[i.1] .](#_bookmark17)

|  |  |  |
| --- | --- | --- |
| 6 | AF | Accelerated Function |
| 7 | AAL | Acceleration Abstraction Layer |
| 8 | AAL-App | Acceleration Abstraction Layer - Application |
| 9 | AAL-LPU | Acceleration Abstraction Layer - LPU |
| 10 | AALI-C | Acceleration Abstraction Layer Interface-Common |
| 11 | AALI-C-Mgmt | Acceleration Abstraction Layer Interface-Common-Management |
| 12 | AALI-C-App | Acceleration Abstraction Layer Interface-Common-Application |
| 13 | AALI-P | Acceleration Abstraction Layer Interface-Profile |
| 14 | BF | Beam-forming |
| 15 | CNF | Cloudified Network Function |
| 16 | DFT/iDFT | Discrete Fourier Transform / Inverse Discrete Fourier Transform |
| 17 | DMS | Deployment Management Services |
| 18 | FCAPS | Fault, Configuration, Accounting, Performance, Security |
| 19 | FEC | Forward Error Correction |
| 20 | FFT/iFFT | Fast Fourier Transform / inverse Fast Fourier Transform |
| 21 | HWA | Hardware Accelerator |
| 22 | IMS | Infrastructure Management Services |
| 23 | LPU | Logical Processing Unit |
| 24 | RB | Resource Block |
| 25 | SMO | Service Management and Orchestration |
| 26 | UC | Use Case |
| 27 | VNF | Virtual Network Function |

# 1 4. General Aspects

## 4.1 Hardware Acceleration

1. In the design of digital computing systems, ranging from general-purpose processors to fully customized hardware,
2. there is a tradeoff between flexibility and efficiency, with efficiency increasing by orders of magnitude when any given
3. application is implemented in hardware. The range of implementation options includes general-purpose processors
4. (GPPs) such as CPUs, more specialized processors such as Graphics Processing Units (GPUs), functions implemented
5. on field-programmable gate arrays (FPGAs), and fixed-functions implemented on application-specific integrated
6. circuits (ASICs). Hardware accelerator is a specialized HW implementation that can offload processing from
7. application(s) running on the General-Purpose Processor (GPP). Any transformation of data or computation can be
8. implemented purely in software running on a generic CPU, or purely in a specialized hardware accelerator, or using a
9. combination of both. The implementation of computing tasks in hardware to improve performance is known as
10. hardware acceleration. The hardware acceleration can be implemented in the form of lookaside or inline mode where in
11. the former case, the host CPU invokes an accelerator for data processing and receives the result after processing is
12. complete, while in the latter case, the accelerator, after being invoked by the host CPU with the request for data
13. processing, completes the processing request of data received from a source node and directly transfers the post-
14. processed data to a destination node, where the source or destination node can be different than host CPU (e.g., an
15. Ethernet Interface). The principle of hardware acceleration and functional offloading in lookaside mode and inline is
16. illustrated in Figure 4.1-1, allowing the application to offload workload to a hardware accelerator and to continue
17. performing other work in parallel- this could be to continue to execute other software tasks in parallel or to sleep and
18. wait for the accelerator hardware to complete. The hardware acceleration boosts application performance in
19. environments with compute-intensive, deeply pipelined, massively parallel operations as shown in Figure 4.1-1. This
20. model requires the support of two operations, one for initiating the offload and another for retrieving the operation once
21. complete.

Without hardware acceleration – serial execution

CPU

Func Func Func

3 4 5

Func 2

Func 1

With hardware acceleration – massively parallel execution within and across functions Lookaside

Func

5

Func 1

Func 2

CPU

Accelerator

|  |  |  |
| --- | --- | --- |
| Func 3 |  | Func 4 |
|  |

Inline

Accelerator handles compute-intensive, deeply pipelined, massively parallel operations. The CPU handles other operations

CPU

Func

5

Func 1

Func 2

Accelerator

|  |  |  |
| --- | --- | --- |
| Func 3 |  | Func 4 |
|  |

1

* 1. **Figure 4.1-1: Example illustration of the effect of hardware acceleration on functional compute**
  2. **performance.**

## 4.2 AAL Architecture and concepts

### 4.2.1 Overview

* 1. The goal of the acceleration abstraction layer (AAL) is to specify a common and consistent set of interfaces to enable
  2. interactions with the HW Accelerators and facilitates decoupling of an AAL Application, from a specific HW
  3. implementation. To accommodate the many different combinations of HW and SW implementation and many different
  4. network deployment scenarios, the AAL introduces the concept of an AAL profile which is used to distinguish between
  5. the different combinations of accelerated functions to be offloaded.
  6. The AAL specifications shall define the AAL interface between the AAL Application and AAL Implementation in the
  7. O-Cloud instance. This includes the APIs, information models, operations and input/outputs used by the AAL
  8. Application to interface with the AAL Implementation. When the AAL APIs are utilized:
     1.  AAL Applications that utilize the AAL APIs render the same functionality as described in the O-RAN
     2. specifications describing the AAL Application [1].
     3.  AAL Applications that utilize the AAL APIs shall preserve the interface definitions and behave the same way
     4. as described in the O-RAN specifications describing those interfaces.
     5.  O-RAN NFs that utilize the AAL APIs shall preserve the synchronization topologies as defined in [[16].](#_bookmark15)
     6. In addition, the AAL Specification shall define the requirements for managing the hardware accelerator in the O-Cloud
     7. instance. The AAL Implementation itself shall not be defined by the AAL GAnP specification. ETSI GS NFV-IFA 002
     8. [5] defines several abstraction models including pass through and abstracted models that can be used to realize an AAL
     9. Implementation.
     10. The AAL specification facilitates the following:
     11. An O-cloud provides the flexibility of deploying multiple software implementations from different software vendors on
     12. a common CPU-based (e.g., x86/ARM) platform with hardware accelerators (e.g., FPGA/DSP/ASIC/GPU) for specific
     13. functions, and conversely, also allows multiple physical deployment scenarios in terms of centralizing or distributing
     14. each network element with the same software implementation.
     15. A disaggregated and cloudified multi-vendor RAN requires common vendor-neutral APIs for managed element
     16. discovery, lifecycle management, FM/PM, and orchestration across both PNFs and VNFs in order to function as a
     17. cohesive unit that supports key lifecycle use cases such as scale-out, slice management, fault tolerance, and hitless
     18. software upgrades.

### 4.2.2 AAL Architecture

* + 1. The end-to-end high-level AAL architecture block diagram is shown in Figure 4.2.5.1-1. For AAL Applications O1
    2. interface usage is optional. Interaction between DMS and AAL Application is out of the scope of this document. For
    3. further understanding of the deployment and operation of AAL-LPUs please refer to section 6.2.

DMS

1



Planning, Manifest,

artifacts etc..

**Service Management and Orchestration (SMO)**

Federated O-Cloud

Orchestration and Management

Network Function Orchestrator (NFO)

O2ims

O2dms

O1

Vendor

Specific

IMS

DMS

AAL-C-Mgmt

Leverage Open source and existing K8s, ETSI-NFV etc..

AAL

Application

AAL-C-App

AAL-P

AAL Implementation

AAL

Profile

Queue

AAL

Profile

Queue

AAL

Profile

Queue

AAL Profile

Instance

AAL Profile

Instance

HW Accelerator

Manager

AAL-LPU N

Vendor Specific

O-Cloud Compute Node

HW Accelerator (Physical)

AAL-LPU 1

AAL-LPU 0

AAL Profile

Instance

Common operations/ actions/events toward AAL Application

AAL-Profile Specific API that provides control of an instance of AAL-Profile

Common administrative operations/ actions/events from the HW Accelerator Manager toward the O- Cloud IMS

RAN Network Function FCAPS

|  |  |  |
| --- | --- | --- |
| AAL  Application |  | AAL  Application |
|  |

In scope for WG6 AAL Subgroup

AAL Profile

Instance

AAL Profile

Instance

AAL Profile

Instance

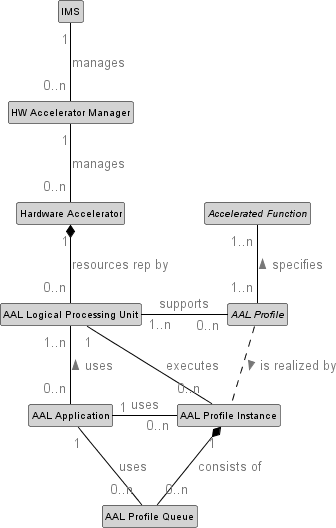
Standardized by WG6 O2 Subgroup Vendor Specific

2

3 **Figure 4.2.2-1: High Level AAL Architecture Diagram**

4

1. Figure 4.2.2-2: shows a pictorial view of the entity relationship and cardinality between AAL entities that constitute the
2. AAL architecture. It is not meant to be the basis of a class diagram or object model which would normally be the start
3. of an information model. Its purpose is just to help the reader mentally conceptualize the concepts depicted. See clause
4. [4.2.4](#_bookmark32) of the present document for a high-level description of the AAL-C-Mgmt interface and the AAL-C-App interface.
5. See [[15]](#_bookmark14) for a detailed description of the specification of the AAL-C-Mgmt and AAL-C-App interfaces and the
6. operations supported.
7. AAL Managed Elements and Managed Functions are managed by SMO via the O2 interface exposed by the IMS and
8. DMS using the AALI-C-MGMT interface terminated by the HW Accelerator Manager. Specifically, AAL Managed
9. Elements and Managed Functions do not utilize the O1 interface for management of AAL Managed Elements and
10. Managed Functions.
11. The HW Accelerator, the HW Accelerator Manager, AAL-LPU and AAL drivers are defined as O-Cloud Platform
12. Software for the purposes of management and orchestration [[1].](#_bookmark6) The O-Cloud Infrastructure and O-Cloud Platform
13. Software use case flows in [[12]](#_bookmark13) are applicable for these AAL elements. 18

1

* 1. **Figure 4.2.2-2: AAL Resource Relationship and Cardinality**

### 4.2.3 HW Accelerator Manager

* 1. The HW Accelerator Manager (HAM) is responsible for exposing a consistent mechanism to the O-Cloud platform for
  2. the discovery, lifecycle management, fault, state/status, performance, configuration, updates/upgrades, and error
  3. handling of the HW Accelerator(s) that are part of the O-Cloud Platform Hardware. The HW Accelerator Manager
  4. exposes the AALI-C-Mgmt interface towards the IMS. The interfaces between the HW Accelerator and the HAM are
  5. vendor specific and not in the scope of AAL specifications. The HW Accelerator Manager is managed by IMS as per
  6. [Figure 4.2.2-1: High Level AAL Architecture Diagram](#_bookmark28) of this document. The identifier of the HW Accelerator Manager
  7. used within the IMS is unique within an O-Cloud instance.
  8. The HW Accelerator Manager is under the control of the O-Cloud Infrastructure. It is integrated into the Cloud
  9. Infrastructure using the O-Cloud Infrastructure's installation procedures. The HW Accelerator Manager shall be
  10. certified by the O-Cloud vendor.
  11. The following list provides a description of capabilities offered by HAM. The list is not exhaustive:
  12.  Discovery and Life Cycle Management:
  13. o The HW Accelerator Manager provides a mechanism to expose inventory information and capabilities
  14. of the physical and logical partitioning of the hardware and software.
  15. o The HW Accelerator Manager provides the ability to discover the capabilities of the HW
  16. Accelerator(s).
      1.  Software/ Firmware upgrade services
      2. o The HW Accelerator Manager allows the update and/or upgrade of the software for a HW
      3. Accelerator(s) on the O-Cloud node. An example of this may include the programming or re-
      4. programming of a downloadable firmware or driver upgrades. Updates/Upgrades can be done locally
      5. or remotely.
      6.  Configuration
      7. o The HW Accelerator Manager allows the configuration of the HW Accelerator as prescribed by the
      8. IMS through the AALI-C-Mgmt interface. The configuration of the HW Accelerator Manager may
      9. include HW Accelerator resource assignment to AAL-LPUs.
      10.  Fault and Performance Monitoring:
      11. o The HW Accelerator Manager allows exposure of faults, logs and performance measurements toward
      12. the IMS.
      13. Deployment Scenarios
      14. The HAM’s software is deployed as part of the O-Cloud Platform software. The installation and update of
      15. the HAM’s software, deployed as a part of O-Cloud platform software, is described section 3.1.2 and
      16. section 3.1.6 of Cloudification and Orchestration Use Cases and Requirements for O-RAN Virtualized
      17. RAN [13].
      18. The HAM may be located on O-Cloud Nodes designated as part of the clusters used to deploy the AAL
      19. Applications (NFs) or otherwise as shown below. The figure shows HAM from the perspective of AAL-C-
      20. Mgmt interface termination point. The actual software entities (i.e., vendor specific libraries used by the
      21. HAM) for managing the HWA can reside in other locations as well.

O2 IMS O2 DMS

er

er

er

**IMS**

**DMS**

**NODE** OS

**N**s**O**e**D**rv**E**er

**NODE**

**NODE** OS

**DMS NODE NODE**

OS

**DMS**

**NODENODE**OS

OS

OS

**HAM**

**HAM**

**NODE**

server

HAM

O-Cloud Node Cluster

(Not hosting NFs)

O-Cloud Node Cluster

(hosting NFs)

O-Cloud Node Cluster

(hosting NFs)

HAM Container/VM on node cluster not used for NF deployment

Centralized HAM remote control of the deployed cluster HWA resources

O-Cloud

HAM container/VM within cluster deploying NF

HAM driver within local server OS

O-Cloud Node Cluster

(hosting NFs)

server

server

HWA

serv

HWA

serv

OS

se**N**r**O**v**D**er**E**

OS

server

OS

**NO**se**D**r**E**ver

OS

server

HWA

serv

OS

se**N**r**O**v**D**er**E**

OS

22

* + 1. NOTE 1: The scenarios depicted are considered an exemplification of the most common deployment scenarios.
    2. Other locations and modelling approaches are not precluded.
    3. **Figure 4.2.3-1 Example HAM software deployment scenarios**

### 4.2.4 AAL Interfaces

* + 1. The AAL interface API has two distinct parts, the first part corresponds to a set of common APIs (AALI-C) to address
    2. all the profile independent aspects of the underlying AAL Implementation(s) within an O-Cloud platform. The second
    3. part corresponds to a set of AAL profile specific APIs (AALI-P) which is specific to each defined AAL profile.
    4. There are two categories of AALI-C interface:
       1.  AALI-C-Mgmt: OAM management from the HAM toward the O-Cloud IMS for acceleration resources
       2. exposed by this interface.
       3.  AALI-C-App: Common operations/actions/events towards the RAN AAL Application for resources exposed
       4. by this interface.
       5. A candidate set of functionalities for AAL services supported by the AAL common API(s) potentially includes (but is
       6. not limited to) the following:
       7.  Inventory Management, Fault, Performance, Configuration Management
       8.  Software/ Firmware upgrade services
       9.  Operations (Query status, Reset/Restart)
       10.  Life Cycle Management of resources exposed by the interface. The AALI-C-Mgmt provides the Life Cycle
       11. Management of the HW Accelerator and the AAL-LPU resources.
       12.  Configuration of the state of these AAL-LPU(s) (for example, start, stop, or reset of an AAL-LPU).
       13.  Configuration of various counters and resources associated with AAL-LPU(s) (for example, performance
       14. measurements/indicators, performance monitoring metrics, events, faults etc.).
       15.  Discovery of AAL-profile(s) supported by these AAL-LPU(s) and associated configurations etc.
       16.  Abstraction of transport mechanism between the AAL Application and AAL Implementation 17

1. The information model and the exact list of operations and actions applicable across AAL-C-Mgmt and AAL-C-App
2. are defined in [[15].](#_bookmark14)
3. The second part of AAL interface corresponds to a set of AAL profile specific APIs (AALI-P) which is specific to each
4. defined AAL profile. The AAL profile can be common across the different AAL Implementation accelerating the same
5. set of AFs. It enables the AAL Application to efficiently offload an AAL profile workload to the AAL Implementation
6. in a consistent way without requiring the HW implementations to expose every single detail of the underlying HW
7. implementation to the AAL Applications. Figure 4.2.4-1 shows examples of the AAL APIs presented to an AAL
8. Application in three different scenarios.
9. The AALI-C-App & AALI-P APIs are internal to the AAL Application that utilize them. 27

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **AAL Application** | | | | |
|  |  |  |  |  |
|  | AAL  Common APIs |  | AAL Profile A APIs |  |
|  | AAL Interface | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **AAL Application** | | | | |
|  |  |  |  |  |
|  | AAL  Common APIs |  | AAL Profile B APIs |  |
|  | AAL Interface | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **AAL Application** | | | | | |
|  |  |  |  |  |  |
| AAL  Common APIs |  | AAL Profile B APIs |  | AAL Profile C APIs |  |
| AAL Interface | | | | | |

28

29 **Figure 4.2.4-1: AAL Application Common and profile APIs**

### 1 4.2.5 AAL Deployment in O-Cloud environments

2

**Virtual Machine**

3



O-RAN Network Function in a container

**API/Library**

O-Cloud Node

Host OS

Compute Svc, with Acceleration

O-Cloud Instance

Acc. Driver

E.g., AAL FEC Profile

**IMS**

**DMS**

Acc. Driver

**AAL Application**



O-RAN Network Function in a Virtual Machine

**API/Library**

O-Cloud Node

Host OS

Compute Svc, with Acceleration

O-Cloud Instance

**IMS**

**DMS**

Acc. Driver

Guest OS

Acc. Driver

E.g., AAL FEC Profile

**AAL Application**

1. **Figure 4.2.5-1: Accelerator APIs/Libraries in Container (left) and Virtual Machine Implementations**
2. **(right)**
3. The AAL specifications define the AAL interfaces (i.e AAL-C and AAL-P) and the AAL profiles that may be
4. supported. The AAL-C-App interface is used by AAL Application to access the AAL Implementation encompassing
5. HW Accelerator and associated SW libraries, drivers etc. In Figure 4.2.5-1 two deployment scenarios are shown, one
6. with Containers and the other with Virtual Machines.
7. Figure 4.2.5-1 also shows the O-Cloud Infrastructure Management Services and Accelerator management entity (i.e. the
8. HAM). The orchestration of the HAM as a managed entity is outside the scope of the AAL and shall be specified in the
9. O-RAN WG6 O2 specification [[11].](#_bookmark12)

#### 4.2.5.1 AAL Implementation Software Deployment Options

1. The following figures describe AAL Implementation software deployment options. In [Figure 4.2.5.1-1:](#_bookmark36) there is an
2. example where the entire AAL Implementation software is deployed as part of the O-Cloud Platform Software [3], and
3. there is no software component dependency between NF Deployments and O-Cloud Platform Software. While in [Figure](#_bookmark37)
4. [4.2.5.1-2:](#_bookmark37) and [Figure 4.2.5.1-3:](#_bookmark38) there are other examples where part of the AAL Implementation software is deployed as
5. part of the NF Deployment and part of O-Cloud Platform Software. In these and similar cases there is a software
6. component dependency between the NF Deployment and the O-Cloud Platform Software, as such updating AAL
7. Implementation software in the O-Cloud Platform Software may require an update to the NF Deployment, this is
8. outside the scope of the AAL.
9. The installation and update of the AAL Implementation software including but not limited to device drivers and
10. libraries, deployed as a part of O-Cloud Platform Software, is described in Cloudification and Orchestration Use Cases
11. and Requirements for O-RAN Virtualized RAN [[12],](#_bookmark13) clause 3.1.2 and 3.1.6.
12. In the diagrams below, ‘Container image(s)’ and ‘VM image(s)’ refer specifically to containers or VMs which are being
13. used to run AAL Applications and do not preclude the existence of other containers or images. These diagrams depict
14. several different possible ways in which the AAL Implementation software may be deployed. The AAL Implementation
15. software may exist wholly or partially inside or outside of the specific containers or VMs which are being used to
16. enclose the AAL Applications. 30

1

Container Image

O-Cloud Platform Software

**AAL Implementation Software**

**(shared library, Kernel space drivers etc..)**

**AAL Application**

2 **Figure 4.2.5.1-1: Example AAL Implementation Software deployment contained within O-Cloud**

3 **Platform Software only**

4

Container Image

O-Cloud Platform Software

**AAL Implementation Software**

**(shared library, Kernel space drivers etc..)**

**AAL Implementation Software (static library, user space drivers etc..)**

**AAL Application**

1. **Figure 4.2.5.1-2: Example AAL Implementation Software Deployment split between O-Cloud Platform**
2. **Software and NF Deployment**

1

VM Image

**Guest OS**

O-Cloud Platform Software

**AAL Implementation Software**

**(shared library, Kernel space drivers etc..)**

**AAL Implementation Software (shared library, kernel space drivers etc..)**

**AAL Implementation Software (static library, user space drivers etc..)**

**AAL Application**

1. **Figure 4.2.5.1-3: Example AAL Implementation Software Deployment split between O-Cloud Platform**
2. **Software and NF Deployment in a virtual machine environment**

## 4.3 Relationship with Standards

1. The O-RAN AAL interface shall leverage existing standards wherever possible.

### 4.3.1 Relationship with ETSI

1. In [4,5,10], ETSI has specified a generic acceleration and abstraction model as well as acceleration resource
2. management that have served as the basis of this specification.
3. ETSI GS NFV-IFA 001 [10] provides a classification of acceleration types and describes several related uses cases, for
4. example about compute acceleration when considering Virtual Base Stations (VBS). ETSI GS NFV-IFA 002 [4]
5. specifies an acceleration architectural model for NFV. The acceleration model considers that a single HW device
6. exposes multiple abstract virtual accelerators and multiple instances of these virtual accelerators per VNF. ETSI GS
7. NFV-IFA 002 [4] defines the interfaces between the NFVI and the VNF instances to support this concept. ETSI GS
8. NFV-IFA 019 [5] defines acceleration resource management interfaces between the VIM and the NFVI (e.g.,
9. Acceleration Image Management interface). From an NFV orchestration and management perspective, acceleration
10. capabilities are defined as part of the relevant Virtual Compute Resource information elements, i.e., a virtual compute
11. resource (e.g., a VM) can have associated acceleration capabilities from the set of capabilities offered by the
12. acceleration resources. For the relevant descriptors, ETSI GS NFV-IFA 011 [14] details the VirtualComputeDesc
13. information element in VNFD which describes CPU, Memory and acceleration requirements of the virtualisation
14. container (e.g., a VM or a set of OS containers, such as a Pod) realizing a VNFC (i.e., a component of a VNF). For
15. containerized deployments required acceleration capabilities can be described by the extendedResourcesRequests
16. attribute in the OsContainerDesc information element. In addition, in the case of a containerized deployment,
17. requirements about constraints on the placement of the set of one or more OS containers (a Pod) to cluster nodes with
18. certain capabilities, such as acceleration, can also be defined. In both cases (VM-based and OS container-based), the
19. VDU information element also supports an attribute to request for additional capabilities, being acceleration related
20. capabilities as one possible example.
21. In ETSI NFV the concept of the abstract virtual accelerator has been devised based on the idea of an extensible para-
22. virtualised device. However, the concept has only been defined considering its use in hypervisor-based virtualization
23. systems and is not ad-hoc applicable to the case of containerized deployments without further analysis. Also, there is no
24. relevant ETSI NFV stage 3 work which can be referenced for the interfaces described in ETSI GS NFV-IFA 002 [4]
25. and ETSI GS NFV-IFA 019 [5].This specification builds upon the concepts described by the aforementioned ETSI NFV
26. specifications but also goes beyond to consider the advances introduced by cloud native solutions, for example by
27. introducing the concept of LPUs. .
28. ETSI GR NFV-IFA 046 [15] profiles the NFV-MANO acceleration abstraction framework and compares it to the O-
29. RAN acceleration abstraction solution developed in the context of O-RAN, from which several challenges are further
30. described. A set of recommendations about future work in the NFV framework are derived in the same referenced
31. group report, some considering an alignment between the two designs to make ETSI NFV specifications more
32. expressive to support additional functionality.

# 1 5. AAL Interface definition General Principles and

2 Requirements

## 5.1 General Principles

1. The set of generic and profile-specific features of the AAL interface between the AAL application and the AAL
2. implementation described in the following subsections are defined from an AAL Application point of view and are
3. related to the AAL-C-App general principles and requirements.

### 5.1.1 Extensibility

1. O-RAN has defined the functions that can be accelerated by the cloud platform based on 3GPP specifications and O-RAN
2. deployment scenarios. However, the AAL should not limit innovation of future implementations and should evolve as
3. the specification requires. To that end, the AAL Application interface shall be extensible to accommodate future revisions
4. of the specification.

### 5.1.2 Interrupt and Poll Mode

1. The AAL Application interface shall support multiple design choices for AAL Application vendors and shall not preclude
2. an AAL Application/HW Accelerator vendor from adopting/supporting an interrupt-driven design or poll-mode design
3. or any combination of both. As such, the AAL Application interface shall support both interrupt mode, poll mode and
4. any combination of interrupt and poll modes for the data-path AAL Application interface.

### 5.1.3 HW Independence

1. AAL Application interface shall be independent of the underlying AAL Implementation.

### 5.1.4 Discovery and Configuration

1. The AAL Application interface shall enable AAL Application to discover and configure AAL-LPU(s). The AAL
2. Application interface shall allow an AAL Application to discover what physical resources have been assigned to it from
3. the upper layers and then to configure said resources for offload operations.

### 5.1.5 Multiple AAL-LPU Support

1. There may be scenarios where multiple AAL-LPUs (either implementing the same or different AAL profile(s) are
2. assigned to a single AAL Application, which uses one or more of these AAL-LPU(s) as needed. The AAL Application
3. interface shall support an AAL Application using one or more AAL-LPU(s) at the same time, as shown in Figure 5.1.5-
4. 1.

1

AAL Interface

AAL Interface

**AAL-LPU**

**AAL-LPU**

**AAL Profile**

**AAL Profile**

**AAL Application**

* 1. **Figure 5.1.5-1: Logical Representation of AAL Application interface support for multiple AAL-LPUs**

### 5.1.6 AAL offload capabilities

* 1. The AAL Application interface in supporting different AAL profiles and AAL Implementations shall support different
  2. offload architectures including look-aside, inline, and any combination of both. An AAL Implementation shall support
  3. one or more of these offload architectures depending on the supported AAL profile(s).

### 5.1.7 Look-aside Acceleration Model

* 1. The AAL Application interface shall support look-aside acceleration model where the AAL Application invokes a HW
  2. Accelerator for data processing and receives the result after processing is complete. A look-aside architecture,
  3. illustrated in Figure 5.1.7-1, allows the AAL Application to offload AF(s) specified by AAL profiles(s) to a HW
  4. Accelerator and continue to perform other work in parallel—this could be to continue to execute other software tasks in
  5. parallel or to sleep and wait for the HW Accelerator to complete. This model requires the AAL Application interface to
  6. support two operations, one for initiating the offload and another for retrieving the output data once complete.

CPU

AAL Interface

HW Accelerator

AAL Profile Instance(s)

**AAL Application**

 Input to Accelerator

* 1.  Output to Accelerator
  2. **Figure 5.1.7-1: AAL Application interface look-aside acceleration model - Data flow**

### 5.1.8 Inline Acceleration Model

* + 1. The AAL Application interface shall support inline acceleration model where the AAL Application, after invoking a HW
    2. Accelerator for offloading AF(s) specified by AAL profile(s), does not necessarily retrieve the post processed data. Unlike
    3. the look-aside acceleration model where data source/sink is always the AAL Application (i.e., the HW Accelerator always
    4. receives the data to be processed from the AAL Application and returns the post processed data to the same), a HW
    5. Accelerator operating in inline acceleration mode receives/returns data from/to a different source/destination endpoint
    6. than the AAL Application, depending on the direction of data flow (e.g., in downlink (DL) direction versus uplink (UL)
    7. direction). Figure 5.1.8-1 shows one possible implementation of an inline acceleration model.

CPU

**AAL Profile AAL Profile Instance(s) Instance(s)**

**Rx**

**Tx**

AAL Interface

**AAL Application**

Input to Accelerator

HW

Accelerator

* + 1.  Output to Accelerator
    2. **Figure 5.1.8-1: AAL Application interface inline acceleration model - Data flow**
    3. In Figure 5.1.8-1, “Tx” refers to the transmission of the data from the HW Accelerator through an egress port (e.g., an
    4. Ethernet interface) to a destination node (e.g., O-RU), while “Rx” refers to the reception of data through an ingress port
    5. (e.g., Ethernet interface) to the HW Accelerator from a source node (e.g O-RU).
    6. While the look-aside architecture (in DL) shall support dataflow from the CPU to the HW Accelerator and back to the
    7. CPU before being sent to the egress port (front-haul interface), the inline architecture (in DL) shall support data flow from
    8. the CPU to the HW Accelerator and directly from the HW Accelerator to the egress port (front-haul interface), instead of
    9. being sent back to the CPU. The typical user plane data flows for accelerating the O-DU High-PHY functions for the
    10. look-aside and inline architectures are as follows.
    11. Look-aside architecture user plane dataflow
    12. CPU ↔ HW Accelerator ↔ CPU ↔ front-haul: for a set of consecutive PHY functions offload (e.g., FEC)
    13. CPU ↔ HW Accelerator ↔ CPU ↔ HW Accelerator ↔…↔ CPU ↔ front-haul: for a set of non-consecutive PHY
    14. functions offload
    15. Inline architecture user plane dataflow
    16. CPU ↔ HW Accelerator ↔ front-haul: for a set of consecutive PHY functions offload (up to the end of the PHY pipeline)
    17. [Figure 5.1.8-2:](#_bookmark54) illustrates one possible implementation of the look-aside and inline architectures. While a set of PHY-
    18. layer functions are offloaded to the HW Accelerator for look-aside acceleration, the entire end-to-end High-PHY pipeline
    19. is offloaded to the accelerator for inline acceleration.

\*e.g. FAPI

**L2**

L2+ on CPU

**O-DU** L2+ PHY FH

HW Accelerator

AAL Interface

Look-Aside

model

FH (7.2)

**Func.**

**n**

**Func.**

**4**

**L2**

**Func.**

**1**

L1/L2

Interface\*

High Phy on CPU

L2+ on CPU

**Func.**

**2 on**

**accelerator**

**Func.**

**N-1 on**

**accelerator**

**Func.**

**3**

**O-RU**

High Phy on CPU

L1/L2

Interface\*

**Accelerator Interface on CPU**

Inline

model

**Func.**

**2**

FH (7.2)

HW Accelerator

AAL Interface

**Func.**

**n**

**Func.**

**n-1**

**Func.**

**3**

**Func.**

**2**

**O-RU**

Front Haul

Interface

I/O data flow

Input to Accelerator

Output to Accelerator

Front Haul

Interface

PHY downlink

PHY uplink

1

2 **Figure 5.1.8-2: User plane dataflow paths in look-aside and inline acceleration architectures.**

3

### 5.1.9 AAL Application interface Concurrency and Parallelism

1. To enable greater flexibility and design choice by AAL Application vendors, the AAL Application interface shall support
2. multi-threading environment allowing an AAL Application to offload acceleration requests in parallel from several
3. threads.

### 5.1.10 Separation of Control and User Plane AAL Application interface APIs

1. For efficiency and flexibility of AAL Implementation, AAL Application interface shall support separation of control and
2. user plane APIs with appropriate identifiers, as required by different AAL profiles.

### 5.1.11 Support of Versatile Acceleration Payload

1. Range of payload sizes can vary widely, depending on the specific layer of the RAN protocol stack from which the
2. workload for AAL profile(s) is offloaded to a HW Accelerator. AAL Application interface API shall be flexible to support
3. various ranges of payload sizes as required by different AAL profiles.

### 5.1.12 Support of Different Transport Mechanisms

1. The transport between an AAL Application and an AAL Implementation can be of different types (e.g., based on shared
2. memory, PCIe interconnect, over ethernet). The AAL Application interface shall support abstraction of these various
3. transport mechanisms between the AAL Application and the AAL Implementation.

### 5.1.13 AAL API namespace

1. For convenience of AAL Implementation, the AAL shall follow a unique name space for all AAL API functions.

### 5.1.14 Chaining of AAL Profiles

1. AAL profiles specify one or multiple AFs in specific order offloaded to the HWA. To take advantage of multiple AFs
2. offloaded to the same or different HWAs, the AAL Implementation can permit the chaining of AAL Profile(s)
3. executing the AF or set thereof. In such a case, the AAL Implementation redirects the output of preceding AAL Profile
4. as input to the subsequent AAL Profile belonging to the same or different LPU(s) on the same or different HWA(s).
5. Such a chaining of AAL Profiles allows data to be transferred from one AAL Profile to another AAL Profile without
6. the intervention of the AAL Application thereby reducing transfer latencies. Chained AAL Profile Instances can reside
7. on the same or different LPUs and HWAs. However, in this version of specification, only considers the chaining of
8. AAL Profiles on single AAL-LPU, and the HAM announces chained AAL Profiles which can be satisfied on the single
9. AAL- LPU and HWA. Other cases where the chained AAL Profile straddles across multiple AAL-LPUs and HWAs are
10. for further study.
11. The use of chaining two AAL Profiles is optional for the AAL Implementation and depends on the AAL Profiles
12. requested by the AAL Application. During CNF deployment as per K8s profile, or VNF deployment as per ETSI
13. profile, AAL Implementation can chain the AAL Profiles to create a chained AAL Profile with more AFs implemented
14. as part of the chained AAL Profile. The chained AAL Profile should conform to the AAL Profile requested by the
15. AAL Application, i.e., the chained AAL Profile has same properties towards the AAL Application as well as the IMS
16. and the DMS. It is the responsibility of the AAL Implementation to announce that the chained profile as an AAL Profile
17. as defined in AAL specifications to which the chain conforms. The first AAL Profile in the chain takes data from the
18. AAL Application and the last AAL Profile returns the transformed data to the AAL Application.
19. A chained AAL Profile Instance is valid only if the two or more AAL Profiles form semantically correct sequence of
20. consecutive blocks in the AF processing chain and no intervention is required by the AAL Application for control or
21. data exchanged between the two AAL Profiles. Additionally, the output format of the preceding AAL Profile should
22. match with the input format of the succeeding AAL Profile.
23. AAL Profile chaining may be applicable to both inline and look-aside acceleration. The data flows in these cases are
24. depicted in [Figure 5.1.14-3:](#_bookmark63) and [Figure 5.1.14-4:.](#_bookmark64)

28

**AAL Application**

**AAL Profile Instance n**

**AAL Profile Instance 2**

**AAL Profile Instance 1**

29 **Figure 5.1.14-1: Data flow through unchained AAL Profiles**

1

**AAL Profile 1**

AAL Chained Profile Instance

**AAL Application**

**AAL Profile n**

**AAL Profile 2**

1. NOTE 1: In Figure 5.1.14-1 and Figure 5.1.14-2 above, the chained AAL Profile Instance is a data stream transformer
2. that works upon the input data stream to produce output stream. Input data and parameters are passed on
3. from preceding AAL Profile to next. These are provided by the AAL Application in case of the first AAL
4. Profile. The difference is that in [Figure 5.1.14-1:,](#_bookmark61) the AAL Application would call a series of AAL Profile
5. Instances, while in **Error! Not a valid bookmark self-reference.**, the AAL Application calls only one AAL
6. Profile Instance, which happens to be the chained AAL Profile instance.
7. NOTE 2: The mechanisms for transfer of data and parameters to each of the AAL Profile Instances in unchained
8. mode and to the chained AAL Profile Instances follow the AAL-C-App interface’s transport as defined in
9. AAL Common API Specifications [14]. The AAL Profiles in the chained mode shall continue to follow the
10. AAL-C-App Interface specified in [14]
11. NOTE 3: The AAL-Implementation can configure the chaining of AAL-Profiles during its initialization in order to avoid
12. disruption in the processing chain while the AAL Application is in service.
13. **Figure 5.1.14-2: Data flow through chained AAL Profiles**
14. The chaining of AAL Profiles is possible across the HWAs and LPUs, however, it is beyond the current scope of
15. document and for future studies.
16. The chained look-aside acceleration architecture follows the same dataflow models as look-aside acceleration functions
17. with subtle difference. The first and the last blocks in the chain interfacing the CPU to HW Accelerator support
18. respectively dataflow from the CPU to the accelerator and then from accelerator back to the CPU. All the intermediate
19. blocks support dataflow to and from the preceding and succeeding accelerated functions when those are configured as
20. links in the chain. When not configured as chained accelerated functions, these should support dataflow as in look-aside
21. acceleration.
22. Look-aside chained architecture user plane dataflow.
23. CPU ↔ HW Accelerator (AF1) ↔ HW Accelerator (AF2) ↔ … ↔ HW Accelerator (AFn) ↔ CPU ↔ front-haul: for
24. offload of a set of consecutive PHY functions (e.g., FEC)
25. CPU ↔ HW Accelerator (AF1) ↔… ↔HW Accelerator (AFn) ↔ CPU ↔ HW Accelerator ↔…↔ CPU ↔ front-
26. haul: for offload of a set of non-consecutive PHY functions
27. [Figure 5.1.14-3:](#_bookmark63) illustrates the data flow through the chained HW Accelerated functions within an accelerator when
28. configured for consecutive PHY function offload while Figure 5.1.14-4 represents it for non-consecutive PHY function
29. offload.

Look-aside chaining Model

High Phy on CPU

L1/L2

Interface\*

**Func.**

**1**

**Func.**

**n**

FH (7.2)

AAL Interface

**Func.**

**2 on accelerator**

**Func.**

**N-1 on accelerator**

HW Accelerator

**O-RU**

L2+ on CPU

**L2**

Front Haul

Interface

**O-DU** L2+ PHY FH

 I/O data flow

1

PHY downlink PHY uplink

* 1. **Figure 5.1.14-3: Dataflow through chained lookaside HW Accelerator for consecutive Hi-PHY**
  2. **functions.**

Look-aside chaining Model

L1/L2

Interface\*

**Func. 1**

High Phy on CPU

**Func. 4**

AAL Interface

**Func. n**

FH (7.2)

**Func. Func.**

**2 3**

**Func. Func.**

**5 N-1**

HW Accelerator

**O-DU** L2+ PHY FH

**O-RU**

L2+ on CPU

**L2**

Front Haul

Interface

 I/O data flow

4

PHY downlink PHY uplink

1. **Figure 5.1.14-4: Dataflow through chained lookaside HW Accelerator for consecutive and non-**
2. **consecutive PHY.**

## 5.2 High-PHY Profile Specific Principles

1. The set of features of AAL described in the following subsections are relevant for inline High-PHY AAL profiles
2. (profile names with suffix ‘\_HIGH-PHY’) defined in Chapter 5

### 5.2.1 Separation of Cell and Slot Level Parameter Configurations

1. In general, “cell-specific” (typically static or semi-static in nature) parameters change less frequently than “slot-specific”
2. (typically dynamic in nature, specific to PHY channels/signals) parameters associated with inline, High-PHY AAL
3. profiles. Hence, for optimizing signalling overhead, the AAL Application interface shall support configuration of “cell-
4. specific” and “slot-specific” parameters to the AAL Implementation using separate AAL Application interface API
5. functions. It is noteworthy that the cell/slot specific configurations can include both control and user planes.

### 5.2.2 SFN/slot-based Synchronization

1. The AAL Application interface shall support system frame number (SFN) based or slot-based synchronization between
2. the AAL Application and the AAL Implementation supporting inline, high-PHY AAL profiles.

### 5.2.3 Compatibility with O-RAN FH interface

1. The functional split between the O-DU and the O-RU remains as defined by the WG4 OFH interface [17] and the AAL
2. Application interface API shall be compatible with O-RAN FH interface (7.2-x split) to enable communication between
3. the O-DU AAL Application and O-RU via AAL Implementation as required by inline, High-PHY AAL profile(s).

### 5.2.4 Inline Profile for High-PHY Stack

1. Inline data flow for the AAL Profile that specifies a complete stack of High-PHY functions implies that the set of
2. Accelerated Functions is constituted by the entire U-plane processing of the High-PHY channels or signals (with 7-2x
3. PHY functional split) and the IQ data (in DL) or decoded bits (in UL) (post processing) and these are transferred
4. directly from the HW Accelerator to the Fronthaul interface (in DL) or to the Layer 2 (in UL).
5. Inline acceleration for AAL Profiles that specify a partial stack of the High-PHY for UL and for DL Accelerated
6. Functions is also possible; in such case only parts of the U-Plane, i.e., not all the functions of the High-PHY stack of a
7. given channel, may be offloaded to a HW Accelerator. For inline acceleration of a partial High-PHY stack, the IQ data
8. or decoded bits are also transferred directly from the HW Accelerator to the Fronthaul (in DL) and to the non-
9. accelerated part of the High-PHY stack (in UL). The set of Accelerated Functions that are offloaded in Inline mode for
10. acceleration of a partial High-PHY stack would be AAL Profile specific. The support of AAL Profiles that specify the
11. support of partial High-PHY stack is for future study.

Partial Inline Model

L1/L2

Interface\*

FH (7.2)

**Func.**

**3**

HW Accelerator

**Func. N-1**

**Func. 4**

AAL Interface

**Func.**

**n**

**Func. 2**

**Func.**

**1**

High Phy on CPU

**O-DU** L2+ PHY FH

**O-RU**

L2+ on CPU

**L2**

Front Haul

Interface

 I/O data flow

17

PHY downlink PHY uplink

18 **Figure 5.2.4-1: Partial Inline Model for AAL Hi-PHY Profile**

19

# 1 6. AAL-LPU Principles

## 6.1 Overview

1. This section discusses about AAL-LPU(s) presented to AAL Applications using the AAL Application interface. An
2. AAL-LPU should not be confused with a physical HW Accelerator. Within a process address space each AAL-LPU
3. shall abstract the AAL Application from underlying HW Accelerator.
4. The following list provides a description of capabilities and characteristics of AAL-LPUs:
5.  An AAL-LPU maps to a single HW Accelerator. An AAL-LPU can be identified uniquely within a HW
6. Accelerator.
7.  A HW Accelerator may support 1 to N AAL-LPU’s.
8.  Each AAL-LPU shares the resources of the associated HW Accelerator with other AAL-LPU(s) mapped to the
9. same HW Accelerator. AAL-LPU can also represent a hard partition of the HW Accelerator where resources
10. are dedicated to the partition.
11.  Mapping of HW Accelerator resources to AAL-LPU shall be configurable from O2 interface
12.  An AAL-LPU may support more than one AAL profile. For each supported AAL profile, an AAL-LPU may
13. execute 0 to N AAL-Profile-Instances.
14.  An AAL-LPU can be assigned to a single POD or VM. Multiple LPU's can be assigned to a POD or VM.
15.  An AAL-LPU can provide service to 0 or more AAL Applications within a POD or VM.
16.  AAL-LPU is a virtual Managed Element as defined in [13].
17. Depending on HW design and implementation choice, a HW Accelerator may accelerate multiple profiles or offer
18. support for sharing HW Accelerator resources between multiple threads, processes, VMs, PODs. For this reason, a
19. second abstract construct known as AAL Profile Queue can optionally be used to
20.  distinguish between multiple supported AAL profiles per AAL-LPU
21.  prioritize access to AAL-LPU resources
22.  group operation requests
23.  allow parallel access through AAL Application interface for multiple threads
24. As an abstract construct, an AAL Profile Queue does not reflect a HW design specification or requirement.
25. From the AAL Application point of view, an AAL Profile queue is exposed by an AAL Profile Instance
26.  an AAL Profiles Instances exposes one or more AAL Profile Queues
27.  The AAL Profile Queue optionally also supports priority, allowing the AAL Application to schedule jobs of
28. different priorities.
29. NOTE: An AAL Profile Queue can be used by an AAL Application to share AAL-LPU resources between
30. threads/cores belonging to the same process address space
31. NOTE: An AAL Application may use multiple AAL Profile Queues to access different AAL Profile Instances
32. supported by an AAL-LPU

## 6.2 LPU Deployment and Operation

### 6.2.1 Example AAL-LPU Mapping

1. The following Section contains example deployments mapping AAL-LPUs to AAL Applications. The labels ‘profile-
2. instanceID’ and ‘queueID’ in the following diagrams denote AAL Profile Instance object handle and identifier of an
3. AAL Profile Queue respectively.
4.  **Scenario 1: Basic implementation:** A HW Accelerator supports a single AAL-LPU which exposes a single
5. AAL-Profile-Instance for one AAL Application to use 8



AAL Application

API(profile-instanceID, queueID)

HW Accelerator (Physical)

AAL-LPU0

AAL Profile Instance

#### A HW Accelerator supports a single AAL LPU which exposes a single AAL Profile Queue for one application to

9 use

1. **Figure 6.2.1-1: Scenario 1: A single AAL LPU exposes a single AAL Profile Queue used by a single**
2. **AAL Application.**

12

13

14

1. **Scenario 2: Basic Multi Application Support:** A HW Accelerator supports multiple AAL LPUs for multiple AAL
2. Applications

17



AAL Application

API(profile-instanceID, queueID)

HW Accelerator (Physical)

AAL-LPU0

AAL Profile Instance

AAL-LPU0

AAL Profile Instance

AAL Application

API(profile-instanceID, queueID)

##### Multi application support

Example showing multiple applications supported

* 1. by a single HW accelerator
  2. **Figure 6.2.1-2: A single HW Accelerator supporting two LPU's each assigned to individual AAL**
  3. **Applications**

4

5

1.  **Scenario 3: Multiple Accelerator Support:** Mapping example showing multiple HW Accelerators assigned
2. to a single AAL Application 8



AAL Application

API(profile-instanceID, queueID)

API(profile-instanceID, queueID)

HW Accelerator (Physical)

HW Accelerator (Physical)

AAL-LPU1

AAL-LPU0

AAL Profile Instance

AAL Profile Instance

##### Multi accelerator support

Example showing multiple HW accelerators

9 assigned to a single application

10

1. **Figure 6.2.1-3: Two HW Accelerators each supporting a single AAL-LPU assigned to a single AAL**
2. **Application.**
3.  **Scenario 4: Multi Queue Support:** AAL LPU mapping showing multiple AAL Profile Queue support



AAL Application

API(profile-instanceID, queueID)

HW Accelerator (Physical)

AAL-LPU0

AAL Profile Instance

**Multi queue support**

#### A single AAL LPU exposes two AAL Profile Queues used by a single AAL

* 1. Application
  2. **Figure 6.2.1-4: A single AAL LPU exposes two AAL Profile Queues used by a single AAL Application.**

3

4  **Scenario 5: Multi Profile Support:** Mapping example showing multi-function support 5



AAL Application

API(profile-instanceID, queueID)

API(profile-instanceID, queueID)

HW Accelerator (Physical)

AAL-LPU0

AAL Profile

Instance

AAL Profile

Instance

##### Multi Profile Support

A single AAL LPU supporting two AAL Profile

Instances exposes two AAL Profile Queues used

1. by a single AAL Application
2. **Figure 6.2.1-5: A single AAL LPU supporting two AAL Profile Instances exposes two AAL Profile**
3. **Queues used by a single AAL Application.**
4.  **Scenario 6: Chained Profile Support:** Mapping example showing a multi-profile chained AAL Profile
5. Instance on an AAL-LPU



AAL Application

API(profile-instanceID, queueID)

HW Accelerator (Physical)

AAL-LPU0

AAL Chained Profile Instance

AAL Profile 2

AAL Profile 1

**Chained Profile Support**

#### Example showing multiple AAL Profiles (1 & 2) chained together to implement a single AAL Profile Instance on an AAL-LPU which is

* 1. equivalent to Profile 1 + Profile 2
  2. **Figure 6.2.1-6: AAL-LPU Mapping example showing chained profile support**

3

### 6.2.2 Statistics

* 1. The AAL Application interface shall provide an AAL Application with general statistics upon request. Statistics may
  2. include but not limited to operation counts and error counts.

### 6.2.3 Memory Management

* 1. O-RAN network functions (O-DU, O-CU, etc.) will be responsible for input, output and operation structure memory
  2. allocation and freeing, using AAL defined memory management functions. All other AAL Application memory is not
  3. required to use the AAL memory management functions.
  4. Device Drivers are free to manage their own internal memory, DMA implementation as needed, the AAL specification
  5. does not add any memory requirements to device driver.
  6. Each AAL Implementation shall define its own memory requirements and implement its own memory backing if
  7. needed.
  8. Each AAL Implementation may define its own operation memory structure and allocation if needed.

### 6.2.4 Run Time Configurations

* 1. Operations are requested to the AAL-LPU to execute specific HW Accelerated Function(s). Each operation shall be
  2. represented by an operation struct that shall define all necessary metadata, configurations and information required for
  3. the operation to be processed on an AAL-LPU. The operation structs shall define the operation type to be performed,
  4. including an operation status and reference to the AAL profile specific operation data which can vary in size and
  5. content depending on the AAL profile. Each AAL profile shall define its own operation structure for its specific
  6. functions.

### 6.2.5 AAL Profile(s) offload, processing status query and processed data

1. retrieval

#### 6.2.5.1 Overview

1. An AAL Application aggregates one or more AAL profile(s) and offload to the AAL Implementation using a single
2. AAL Application interface API invocation. As one example, for High-PHY AAL profiles defined in clause[7,](#_bookmark92) multiple
3. AAL profiles (where an AAL profile refers to a PHY channel/signal for one or more than one cell(s) and one or more
4. than one UE(s)) scheduled within a slot can be aggregated and offloaded to an AAL-LPU by the AAL Application
5. using a single AAL Application interface API invocation.
6. The processing status of offloaded/enqueued AAL profile(s) can be queried by the AAL Application in an
7. ‘asynchronous’ manner, i.e., not necessarily in the same order in which the AAL profile(s) are offloaded. In case the
8. AAL Application retrieves the post-processed data from the AAL Implementation, a ‘processing status query’ request
9. can be bundled with a ‘processed data retrieval/dequeue’ request. In general, status query and dequeue request
10. corresponding to multiple enqueue requests can be bundled together by the AAL Application and invoked through a
11. single AAL Application interface API function.

#### 6.2.5.2 AAL Profile Functionality Bypass

1. An AAL Profile specifies a series of one or more Accelerated Functions (AFs) which can be offloaded to Hardware
2. Accelerator by the AAL Application. These AFs are linked internally and interface to each other in a manner
3. transparent to the AAL Application. However, under certain situations, it is desirable to bypass certain AFs
4. implemented by the AAL Profile. For example, consider an AAL Profile that has 3 AFs as shown in Figure-6.2.5.1-1
5. (a) below. With bypass mode, it is possible to get functionalities in (b), (c), (d), (e), (f) and (g).

From AAL Application

From AAL Application

From AAL Application

From AAL Application

From AAL Application

From AAL Application

From AAL Application

AF1 AF2 AF3

AF1

AF2

AF1

AF3

AF2

AF3

AF3

AF1

AF2

To AAL Application or NIC To AAL Application or NIC To AAL Application or NIC To AAL Application or NIC To AAL Application or NIC To AAL Application or NIC

To AAL Application or NIC

21 (a) (b) (c) (d) (e)

1. **Figure 6.2.5.1- 1 Various AAL Bypass scenarios**

(f) (g)

1. Any AF in the set of processing functions may be bypassed as long as the processing chain of the AAL Profile remains
2. unaffected. Bypass modes apply to both inline and look-aside accelerators.
3. The capability of the AAL Profile to support bypassing of certain AFs (“bypass mode” of operation) and the list of AFs
4. which can be bypassed is advertised by the LPU. The AAL Profile definition shall continue to include all the mandatory
5. accelerated functions that need to be supported by the vendors to comply with the O-RAN profile specification. 28

### 6.2.6 AAL-LPU Exposure

#### 6.2.6.1 Overview

1. After using the HW Accelerator Manager to create the AAL-LPU(s) and the profiles supported, we can then expose it to
2. the AAL Application (e.g. in KubernetesⓇ it will be part of the POD environment variables). The goal of this section is
3. to abstract the way the AAL-LPU and its supported profiles are exposed to the AAL Application in order to achieve
4. AAL Application portability. Hence, the AAL-LPU and its profile(s) shall be exposed the AAL Application in an
5. abstracted descriptor.
6. As of today many implementations refer to the AAL-LPU by its PCI address and describe a single profile for the entire
7. HW Accelerator, meaning exposure of one single profile for all LPUs supported. This section provides step-by-step
8. example that shows how the AAL-LPU could be exposed to abstract the PCIe address and the profile(s) supported by
9. the AAL-LPU. Although the below example is a KubernetesⓇ example, the outcome of it is independent of
10. orchestration technology supported.

#### 6.2.6.2 Example

1. 6.2.6.2.1 AAL-LPU Configuration
2. 1. A given HW Accelerator can support 16 AAL-LPUs and FEC LDPC, PHY profiles but the default setup of the
3. AAL-LPUs is FEC:

7

LPU-1 FEC

LPU-2 FEC

LPU-3 FEC

LPU-16 FEC

8

HW Accelerator (Physical)

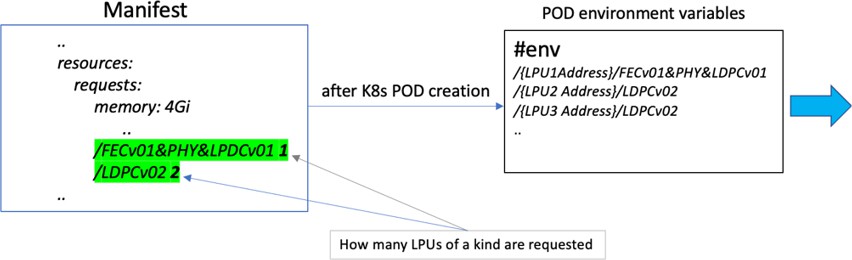
1. **Figure 6.2.6.2.1-1: Example AAL-LPU and profile supported**
2. 2. Now assume HW Accelerator Manager configured the HW Accelerator as follow: 11
3.  **LPU-1** supports FEC version 01, PHY and LDPC version 01 profiles
4.  **LPU-2** supports LDPC version 02 profile
5.  **LPU-3** supports LDPC version 02 profile
6.  **LPU-4** supports LDPC version 02 profile

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| LPU-1 |  |  |  |  |  |  |
| FECv01 & |  | LPU-2 |  | LPU-3 |  | LPU-4 |
| PHY & |  | LDPCv02 |  | LDPCv02 |  | LDPCv02 |
| LDPCv01 |  |  |  |  |  |  |

16

HW Accelerator (Physical)

1. **Figure 6.2.6.2.1-2: Example HW Accelerator configuration**
2. 3. Assigning it to a vDU POD and exposing the two LPU/Profiles needed by vDU (in green): 19



LPU-1 FECv01 & PHY & LDPCv01

LPU-2 LDPCv02

LPU-3 LDPCv02

HW Accelerator (Physical)

20

1. **Figure 6.2.6.2.1-3: Example assignment of AAL-LPUs and supported profiles to POD**
2. We can see that the AAL-LPU resources requested in the manifest is translated in the POD environment to three AAL-
3. LPU addresses plus the strings that describe the profiles supported by the LPU. Please note that the AAL Application
4. will be able to query the AAL-LPUs and their supported profiles via the AAL-C-App API.
5. 6.2.6.2.2 AAL-LPU Control by the AAL Application
6. The figure below shows how the AAL Application for a vDU can create several PHY profiles on LPU-1, for example,
7. PHY Profile-1 handles cell-1 and PHY Profile-2 handles cell-2, as well as running different LDPC profiles on different
8. LPUs and all in the same HW Accelerator. As mentioned before the AAL Application can query what profiles
9. supported by reading the POD environment variables or by querying it using the AAL-C-App API.

The numbers represent an application perspective (e.g. handlers)

FECv01 Profile Instance 1 FECv02 Profile Instance 1 PHY Profile Instance 1

PHY Profile Instance 2

LDPCv02 Profile Instance 1

LDPCv02 Profile Instance 1

LPU-1

LPU-2

LPU-3

6

HW Accelerator (Physical)

1. **Figure 6.2.6.2.2-1: Example AAL Application configured AAL-Profile-Instances**
2. 6.2.6.2.3 AAL-LPU resource tracking
3. It is a key to be able to track the AAL-LPU as a resource for providing the AAL Application with acceleration resources
4. it needs otherwise the attempt to create the NF will fail due to insufficient resources available.
5. There are few notes related to the tracking of the AAL-LPU resources in relation to the example’s above in :
6. • If another POD needs to use the LDPC profile, Kubernetes will allow it as 2 out of 3 were used (LPU-4
7. supports it).
8. • If another POD needs to use FEC and PHY profiles (FEC&PHY), Kubernetes will reject it as only one LPU is
9. configured this way.
10. • We need to consider if a logic is needed to track the AAL-LPU availability from the SMO. For example, a
11. total of ten AAL-LPU/FEC profiles available in a cluster, three is being used by a deployment, seven left for a
12. new deployment. The seven available could be revealed to the user or automation at the SMO level when
13. considering additional deployments. 20
14. The conclusion is that the SMO shall be able to track the AAL-LPU as a resource and its availability.

### 6.2.7 Accelerator configuration options between IMS & FOCOM

#### 6.2.7.1 Declarative approach

1. Declarative approach defines a desired system state and software is responsible for bringing the system to that state. In
2. context of accelerator management, it is the desired state of the accelerator configuration to support certain profile(s)
3. characteristics. This approach is flexible, reliable and repeatable, reducing or eliminating the need for imperative
4. deployment techniques such as scripting and manual command entry. The O-RAN WG6 O2 General Aspects and
5. Principles [11] clause 3.4.3.8 discusses the Cluster Template concept that this supports.
6. The following are examples of a desired state for an accelerator to support an AAL-Profile version 1.9.0 with two
7. LPUs.

3

*Example profile centric*

… Profile\_Name: **AAL-PROFILE**

**{**

Profile\_Version: **1.9.0**

LPUs\_Request: **2**

**}**

…

*Example LPU Centric*

…

LPUs\_Request: **2**

**{**

Profile\_Name: **AAL-PROFILE**

Profile\_Version: **1.9.0**

}

4

5

6

1. Or

8

9

10

11

1. NOTE 1: The SMO represented Cluster Template instance characteristics such as Capabilities and Capacities are not
2. described in the present document.

# 1 7. AAL Profiles

* 1. An AAL profile specifies a set of Accelerated Functions that a Hardware Accelerator processes on behalf of an AAL
  2. Application within an O-RAN Cloudified Network Function (e.g. O-DU, O-CU etc.). Accordingly, AAL profiles can be
  3. categorized as O-DU AAL profiles, O-CU AAL profiles and so on. The following sections describe these different
  4. AAL profile categories in further details.

## 7.1 O-DU AAL Profiles

* 1. An O-DU AAL profile can specify a set of Accelerated Functions within the O-DU protocol stack. These functions may
  2. belong to a single layer (e.g., PHY) or span across multiple layers (e.g., PHY and MAC) within O-DU. The current O-
  3. DU AAL profiles being studied by O-RAN WG6 are focusing on Accelerated Functions from PHY layer of O-DU.

### 7.1.1 O-DU Protocol Stack Reference

* 1. [Figure 7.1.1-1:](#_bookmark95) illustrates the building blocks for processing various O-DU PHY layer Downlink (DL) channels and
  2. signals (with 7.2-x functional split between O-DU and O-RU) defined by 3GPP in [[6]](#_bookmark8) & [[7]](#_bookmark9) as part of 5G NR
  3. specification.

L2+

**PDSCH TB(s)**

|  |
| --- |
| PBCH payload generation |
| Scrambling |
| TB CRC  attachment |
| Polar encoding |
| Rate matching |
| Data scrambling |
| Modulation  (QPSK) |

TB CRC attachment

CB segmentation + CB CRC attachment

LDPC encoding

**PDCCH (DCI)**

|  |
| --- |
| CRC attachment |
| Polar encoding |
| Rate matching |
| Scrambling |
| Modulation  (QPSK) |

**PBCH TB**

Rate matching

**PDSCH DM-RS**

Precoding\*

Sequence generation

Scrambling

CB concatenation

**PDCCH DM-RS**

RE mapping

Precoding\*

Modulation

Sequence Generation

Modulation

Sequence generation

RE mapping

Precoding\*

Precoding\*

Layer mapping

Modulation

Modulation

**PSS/SSS PBCH DM-RS**

**CSI-RS/PT-RS/TRS**

Modulation

Sequence Generation

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

14

O-RAN FH (7-2x lower layer split)

RE mapping

RE mapping

1. **Figure 7.1.1-1: O-DU PHY processing blocks for 5G NR Downlink**
2. The O-DU PHY layer in downlink consists of the following physical channels and reference signals:
3. • Physical Downlink Shared Channel (PDSCH) and associated Demodulation Reference Signal (PDSCH DM-RS).
4. • Physical Downlink Control Channel (PDCCH) and associated Demodulation Reference Signal (PDCCH DM-RS).
5. • Synchronization Signal Block (SSB) consisting of
6. o Physical Broadcast Channel (PBCH) and associated DMRS (PBCH DM-RS).
7. o Primary Synchronization Signal (PSS).
8. o Secondary Synchronization Signal (SSS).
   1. • Channel State Information-Reference Signal (CSI-RS) and Tracking Reference Signal (TRS).
   2. • Phase Tracking Reference Signal (PT-RS) for DL.
   3. The downlink physical channels (PDSCH, PDCCH, PBCH) carry information originating from higher layers (i.e. layer
   4. 2 and above).
   5. The downlink physical layer processing of data channel (PDSCH) carrying transport blocks consists of the following
   6. steps:
   7. TB CRC attachment: Error detection is provided on each transport block (TB) through a Cyclic Redundancy Check
   8. (CRC). Refer to Subclause 7.2.1 in [[7]](#_bookmark9) for details.
   9. CB segmentation and CRC attachment: The transport block is segmented when it exceeds the code block (CB) size
   10. specified by 3GPP [[7].](#_bookmark9) Code block segmentation and code block CRC attachment are performed according to
   11. Subclauses 7.2.3 and 5.2.2 of [[7].](#_bookmark9)
   12. LDPC encoding: Refer to Subclauses 7.2.4 and 5.3.2 in [[7]](#_bookmark9) for details.
   13. Rate matching: Refer to Subclauses 7.2.5 and 5.4.2 in [[7]](#_bookmark9) for details.
   14. CB concatenation: Refer to Subclauses 7.2.6 and 5.5 in [[7]](#_bookmark9) for details.
   15. Scrambling: Refer to Subclause 7.3.1.1 in [[6]](#_bookmark8) for details.
   16. Modulation: Refer to Subclause 7.3.1.2 in [[6]](#_bookmark8) for details.
   17. Layer mapping: Refer to Subclause 7.3.1.3 in [[6]](#_bookmark8) for details.
   18. RE mapping: Refer to Subclause 7.3.1.5 and 7.3.1.6 in [[6]](#_bookmark8) for details on Resource Element (RE) mapping.
   19. The downlink physical layer processing of control channel (PDCCH) carrying Downlink Control Information (DCI)
   20. consists of the following steps:
   21. CRC attachment: Error detection is provided on DCI transmissions through a Cyclic Redundancy Check (CRC). Refer
   22. to Subclause 7.3.2 in [[7]](#_bookmark9) for details.
   23. Polar encoding: Refer to Subclauses 7.3.3 and 5.3.1 in [[7]](#_bookmark9) for details.
   24. Rate matching: Refer to Subclauses 7.3.4 and 5.4.1 in [[7]](#_bookmark9) for details.
   25. Scrambling: Refer to Subclause 7.3.2.3 in [[6]](#_bookmark8) for details.
   26. Modulation: Refer to Subclause 7.3.2.4 in [[6]](#_bookmark8) for details.
   27. RE mapping: Refer to Subclause 7.3.2.5 in [[6]](#_bookmark8) for details.
   28. The downlink physical layer processing of broadcast channel (PBCH) carrying maximum one transport block consists
   29. of the following steps:
   30. PBCH payload generation: Refer to Subclause 7.1.1 in [[7]](#_bookmark9) for details.
   31. Scrambling: Refer to Subclause 7.1.2 in [[7]](#_bookmark9) for details.
   32. TB CRC attachment: Refer to Subclause 7.1.3 in [[7]](#_bookmark9) for details.
   33. Polar encoding: Refer to Subclauses 7.1.4 and 5.3.1 in [[7]](#_bookmark9) for details.
   34. Rate matching: Refer to Subclauses 7.1.5 and 5.4.1 in [[7]](#_bookmark9) for details.
   35. Data scrambling: Refer to Subclause 7.3.3.1 in [[6]](#_bookmark8) for details.
   36. Modulation: Refer to Subclause 7.3.3.2 in [[6]](#_bookmark8) for details.
   37. RE mapping: Refer to Subclause 7.3.3.3 in [[6]](#_bookmark8) for details.
9. The downlink physical signals (DM-RS, PSS, SSS, CSI-RS/TRS, PT-RS) correspond to a set of resource elements used
10. by the physical layer but does not carry information originated from higher layers (i.e. layer 2 and above).
11. Reference Signals (DM-RS, CSI-RS/TRS, PT-RS) and Synchronization Signals (PSS/SSS) are generated using the
12. following steps:
13. Sequence Generation and Modulation: Refer to Subclauses 7.4.1.1.1 (PDSCH DM-RS), 7.4.1.3.1 (PDCCH DM-RS),
14. 7.4.1.4.1 (PBCH DM-RS), 7.4.1.5.2 (CSI-RS/TRS), 7.4.1.2.1 (PT-RS), 7.4.2.2.1 (PSS) and 7.4.2.3.1 (SSS) in [[6]](#_bookmark8) for
15. details.
16. RE mapping: Refer to Subclauses 7.4.1.1.2 (PDSCH DM-RS), 7.4.1.3.2 (PDCCH DM-RS), 7.4.1.4.2 (PBCH DM-RS),
17. 7.4.1.5.3 (CSI-RS/TRS), 7.4.1.2.2 (PT-RS), 7.4.2.2.2 (PSS) and 7.4.2.3.2 (SSS) in [[6]](#_bookmark8) for details.
18. An O-DU AAL profile for 5G NR downlink shall specify a set of accelerated functions corresponding to one or more
19. than one physical downlink channel(s) and/or physical downlink signal(s).
20. In addition to the processing blocks mentioned above, each of these downlink physical channels/signals may include
21. some additional functional blocks (e.g. precoding, IQ compression) which are implementation specific and may also
22. depend on system configurations/capabilities (for example, whether a O-DU is connected to a CAT-A/CAT-B O-RU).
23. Each of these physical channels/signals can be implemented with/without these optional functional blocks. The AAL
24. Application interface shall expose to the AAL Application whether these functional blocks are supported or not within
25. the AAL Implementation.
26. [Figure 7.1.1-2:](#_bookmark96) illustrates the building blocks for processing various O-DU PHY layer Uplink (UL) channels and signals
27. (with 7.2-x functional split between O-DU and O-RU) defined by 3GPP [[6]](#_bookmark8) as part of 5G NR specification. 20

L2 +

**PUSCH**

**UL data**

|  |
| --- |
| TB CRC check |
| CB CRC + CB  Desegmentation |
| LDPC decoding |
| Rate dematching |
| CB  Deconcatenation |
| Descrambling |
| Demodulation |
| IDFT for DFT-s- OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 0 UCI**

RE demapping

Sequence detection

|  |
| --- |
| Demodulation |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 1 UCI**

**PUCCH format 2/3/4 UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PRACH**

|  |  |  |
| --- | --- | --- |
| Preamble detection + delay estimation | **SRS** | |
| Peak search |
| Noise estimation |
| IFFT |
| Root sequence correlation |  | Channel estimation |
| RE demapping |  | RE demapping |

**PT-RS**

RE demapping

Sequence detection

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

21

O-RAN FH (7-2x lower layer split)

1. **Figure 7.1.1-2: O-DU PHY processing blocks for 5G NR Uplink**
2. The O-DU PHY layer in uplink consists of the following physical channels and reference signals:
3. • Physical Uplink Shared Channel (PUSCH).
4. • Physical Uplink Control Channels (PUCCH) with formats 0/1/2/3/4.
5. • Physical Random-Access Channel (PRACH).
6. • Sounding Reference Signal (SRS).
7. • Phase Tracking Reference Signal (PT-RS) for UL.
8. The uplink physical channels (PUSCH, PUCCH, PRACH) carry information originating from higher layers (i.e. layer 2
9. and above).
10. The uplink physical layer processing of shared channel (PUSCH) carrying uplink data with or without Uplink Control
11. Information (UCI) consists of the following steps at the receiver (O-DU):
12. RE de-mapping: Refer to Subclauses 6.3.1.6, 6.3.1.7 and 6.4.1.1.3 of [[6]](#_bookmark8) for details on RE mapping at the transmitter.
13. Channel estimation and equalization: up to O-DU implementation.
14. Transform precoding (IDFT): optional, only required for DFT-s-OFDM waveform. Refer to Subclause 6.3.1.4 of [[6]](#_bookmark8) for
15. details on transform precoding (if applicable) applied at the transmitter.
16. Demodulation: Refer to Subclause 6.3.1.2 in [[6]](#_bookmark8) for details on modulation applied at the transmitter.
17. Descrambling: Refer to Subclause 6.3.1.1 in [[6]](#_bookmark8) for details on scrambling applied at the transmitter.
18. CB de-concatenation: Refer to Subclause 6.2.6 in [[7]](#_bookmark9) for details on CB concatenation applied at the transmitter.
19. Rate de-matching: Refer to Subclause 6.2.5 in [[7]](#_bookmark9) for details on rate matching applied at the transmitter.
20. LDPC decoding: Refer to Subclause 6.2.4 in [[7]](#_bookmark9) for details on LDPC encoding applied at the transmitter.
21. CB de-segmentation and CB CRC check: Refer to Subclause 6.2.3 in [[7]](#_bookmark9) for details on CB segmentation and CB CRC
22. attachment applied at the transmitter.
23. TB CRC check: Refer to Subclause 6.2.1 in [[7]](#_bookmark9) for details on TB level CRC attachments applied at the transmitter.
24. The uplink physical layer processing for control channel (PUCCH) carrying UCI depends on PUCCH formats.
25. PUCCH format 0 processing consists of the following steps at the receiver (O-DU):
26. RE de-mapping: Refer to subclause 6.3.2.3.2 of [[6]](#_bookmark8) for details on RE mapping applied at the transmitter.
27. Sequence detection: The transmitted sequence (refer to Subclause 6.3.2.3 in [[6]](#_bookmark8) for details) is detected at O-DU using a
28. non-coherent detector, since PUCCH format 0 does not carry any DM-RS. The detailed design is up to O-DU
29. implementation.
30. PUCCH format 1 processing consists of the following steps at the receiver (O-DU):
31. RE de-mapping: Refer to Subclauses 6.3.2.4.2 and 6.4.1.3.1.2 of [[6]](#_bookmark8) for details on RE mapping applied at the
32. transmitter.
33. Channel estimation and equalization: up to O-DU implementation.
34. Demodulation: Refer to Subclause 6.3.2.4.1 in [[6]](#_bookmark8) for details on modulation applied at the transmitter.
35. PUCCH formats 2/3/4 processing consists of the following steps at the receiver (O-DU):
36. RE de-mapping: Refer to Subclauses 6.3.2.5.3 and 6.4.1.3.2.2 (format 2); 6.3.2.6.5 and 6.4.1.3.3.2 (formats 3/4) of [[6]](#_bookmark8)
37. for details on RE mapping applied at the transmitter.
38. Channel estimation and equalization: up to O-DU implementation.
39. Transform precoding (IDFT): optional, only required for DFT-s-OFDM waveform. Refer to Subclause 6.3.2.6.4 of [[6]](#_bookmark8)
40. for details on transform precoding (applicable for formats 3/4) applied at the transmitter.
41. Demodulation: Refer to Subclause 6.3.2.5.2 (format 2) and 6.3.2.6.2 (formats 3/4) in [[6]](#_bookmark8) for details on modulation
42. applied at the transmitter.
43. Descrambling: Refer to Subclause 6.3.2.5.1 (format 2) and 6.3.2.6.1 (formats 3/4) in [[6]](#_bookmark8) for details on scrambling
44. applied at the transmitter.
45. Rate de-matching: Refer to Subclause 6.3.1.4 in [[7]](#_bookmark9) for details on rate matching applied at the transmitter.
46. Polar/Block decoding: Refer to Subclause 6.3.1.3 in [[7]](#_bookmark9) for details on Polar/Block encoding applied at the transmitter.
47. CRC check: Refer to Subclause 6.3.1.2 in [[7]](#_bookmark9) for details on CRC attachment applied at the transmitter.
48. The uplink physical layer processing for random access channel (PRACH) carrying preamble consists of the following
49. steps at the receiver (O-DU):
50. RE de-mapping: Refer to Subclause 6.3.3.2 in [[6]](#_bookmark8) for details on RE mapping applied at the transmitter.
51. Root sequence correlation: Perform correlation operation between root sequence and received signals. Refer to
52. Subclause 6.3.3.1 in [[6]](#_bookmark8) for details on root sequence generation.
53. IFFT: perform the inverse Fast Fourier Transform (iFFT) operation on the received signal(s).
54. Noise estimation: perform the noise estimation operation.
55. Peak search: detect the peak for different root sequences.
56. Preamble detection and Timing Advance (TA) or delay estimation: determine the preamble sequence(s) received and
57. the corresponding timing advance estimate(s).
58. The uplink physical signals (SRS, PT-RS) do not carry any information from the higher layers (i.e. layer 2 and above).
59. The Sounding Reference Signal (SRS) in uplink is received at O-DU using the following steps:
60. RE de-mapping: Refer to Subclauses 6.4.1.4.3 and 6.4.1.4.4 in [[6]](#_bookmark8) for details on RE mapping applied at the transmitter.
61. Sequence detection and Channel estimation: Up to O-DU implementation. Refer to 6.4.1.4.2 in [[6]](#_bookmark8) for details on SRS
62. sequence generation at the transmitter. Channel condition in uplink is estimated at the O-DU based on the processing of
63. received SRS.
64. The Phase-Tracking Reference Signal (PT-RS) in uplink is received at the O-DU using the following steps:
65. RE de-mapping: Refer to Subclause 6.4.1.2.2 in [[6]](#_bookmark8) for details on RE mapping applied at the transmitter.
66. Sequence detection: Up to O-DU implementation. Refer to Subclause 6.4.1.2.1 in [[6]](#_bookmark8) for details on sequence generation
67. at the transmitter.
68. An O-DU AAL profile for 5G NR uplink shall specify a set of accelerated functions corresponding to one or more than
69. one physical uplink channel(s) and/or physical uplink signal(s).
70. In addition to the processing blocks mentioned above, each of these uplink physical channels/signals may include an
71. additional functional block, viz. IQ decompression, which is implementation specific and may depend on system
72. configuration/capability. Each of these physical channels/signals can be implemented with/without this optional
73. functional block. The AAL Application interface shall expose to the AAL Application whether these functional blocks
74. are supported or not within the AAL Implementation.

### 7.1.2 O-DU Protocol Stack Reference for mMTC

1. Figure 7.1.2-1 illustrates the building blocks for processing various O-DU PHY layer Downlink (DL) channels and
2. signals (with 7.2-x functional split between O-DU and O-RU) defined by 3GPP in [[8]](#_bookmark10) & [[9]](#_bookmark11) as part of 4G/5G NR
3. specification.

L2+

**NPDSCH TB(s)**

###### NPBCH TB

TB CRC attachment

**N PDCCH (DCI)**

Tail Biting Convolutional Coding

Rate matching

CRC attachment

Tail Biting

Convolutional Coding

###### NPSS/NSSS/ NRS

CRC

attachment

PBCH payload generation

Layer mapping

Modulation

Modulation

Layer Mapping

Modulation

(QPSK)

Modulation

Scrambling

Sequence Generation

Scrambling

Rate matching

Scrambling

CB concatenation

Rate Matching

Tail Biting

Convolutional Coding

###### NPRS/NWUS

Sequence Generation

Modulation

Precoding\*

Precoding\*

Precoding\*

Precoding\*

RE mapping

RE mapping

RE mapping

RE mapping

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

1

O-RAN FH (7-2x lower layer split)

2 **Figure 7.1.2-1: O-DU PHY processing blocks for mMTC Downlink.**

3

1. The O-DU PHY layer in downlink consists of the following physical channels and reference signals:
2. • Narrow-band Physical Downlink Shared Channel (NPDSCH).
3. • Narrow-band Physical Downlink Control Channel (NPDCCH).
4. • Narrow-band Physical Broadcast Channel (NPBCH).
5. • Narrow-band Primary Synchronization Signal (NPSS).
6. • Narrow-band Secondary Synchronization Signal (NSSS).
7. • Narrow-band Reference Signal (NRS) and Narrow-band Position Reference Signal (NPRS).
8. • Narrow-band Wake-Up Signal (NWUS)
9. The Narrow-band downlink physical channels (NPDSCH, NPDCCH, NPBCH) carry information originating from
10. higher layers (i.e. layer 2 and above).
11. The Narrow-band downlink physical layer processing of data channel (NPDSCH) carrying transport blocks consists of
12. the following steps:
13. TB CRC attachment: Error detection is provided on each transport block (TB) through a Cyclic Redundancy Check
14. (CRC). Refer to Subclause 6.4 in [[9]](#_bookmark11) for details.
15. CRC attachment: CRC attachment is performed according to Subclauses 6.4 of [[9]](#_bookmark11)
16. Tail-biting Convolutional coding: Refer to Subclauses 6.2 and 5.1.3.1 in [[9]](#_bookmark11) for details.
17. Rate matching: Refer to Subclauses 6.4 in [[9]](#_bookmark11) for details.
18. Scrambling: Refer to Subclause 10.2.5.2 in [[8]](#_bookmark10) for details.
19. Modulation: Refer to Subclause 10.2.5.3 in [[8]](#_bookmark10) for details.
20. Layer mapping: Refer to Subclause 10.2.5.3in [[8]](#_bookmark10) for details.
21. RE mapping: Refer to Subclause 10.2.5.5in [[8]](#_bookmark10) for details on Resource Element (RE) mapping.
22. The downlink physical layer processing of control channel (PDCCH) carrying Downlink Control Information (DCI)
23. consists of the following steps:
24. CRC attachment: Error detection is provided on DCI transmissions through a Cyclic Redundancy Check (CRC). Refer
25. to Subclause 6.4 in [[9]](#_bookmark11) for details.
26. Tail-biting Convolutional coding: Refer to Subclauses 6.2 and 5.1.3.1 in [[9]](#_bookmark11) for details.
27. Scrambling: Refer to Subclause 10.2.5.2 in [[8]](#_bookmark10) for details.
28. Modulation: Refer to Subclause 10.2.5.3 in [[8]](#_bookmark10) for details.
29. Layer mapping: Refer to Subclause 10.2.5.3 in [[8]](#_bookmark10) for details.
30. RE mapping: Refer to Subclause 10.2.5.5 in [[8]](#_bookmark10) for details on Resource Element (RE) mapping.
31. The downlink physical layer processing of broadcast channel (NPBCH) carrying maximum one transport block consists
32. of the following steps:
33. NPBCH payload generation: Refer to Subclause 6.4.1 in [[9]](#_bookmark11) for details.
34. TB CRC attachment: Error detection is provided on each transport block (TB) through a Cyclic Redundancy Check
35. (CRC). Refer to Subclause 6.4 in [[9]](#_bookmark11) for details.
36. Scrambling: Refer to Subclause 10.2.5.2 in [[8]](#_bookmark10) for details.
37. Modulation: Refer to Subclause 10.2.5.3 in [[8]](#_bookmark10) for details.
38. Layer mapping: Refer to Subclause 10.2.5.3 in [[8]](#_bookmark10) for details.
39. RE mapping: Refer to Subclause 10.2.5.5 in [[8]](#_bookmark10) for details on Resource Element (RE) mapping.
40. The downlink physical signals (NRS, NPSS, NSSS, NPRS, NWUS) correspond to a set of resource elements used by
41. the physical layer but does not carry information originated from higher layers (i.e., layer 2 and above).
42. Reference Signals and Synchronization signals (NPSS/NSSS) are generated using the following steps:
43. Sequence Generation and Modulation and RE mapping: Refer to Subclauses, 10.2.6B (NWUS), 10.2.7.1 (NPSS) and
44. 10.2.7.2 (NSSS), 10.2.6 (NRS), 10.2.6A (NPRS) in [[8]](#_bookmark10) for details.
45. An O-DU AAL profile for 4G NR downlink shall specify a set of accelerated functions corresponding to one or more
46. than one physical downlink channel(s) and/or physical downlink signal(s).
47. In addition to the processing blocks mentioned above, each of these downlink physical channels/signals may include
48. some additional functional blocks (e.g., precoding, IQ compression) which are implementation specific and may also
49. depend on system configurations/capabilities (for example, whether a O-DU is connected to a CAT-A/CAT-B O-RU).
50. Each of these physical channels/signals can be implemented with/without these optional functional blocks. The AAL
51. Application interface shall expose to the AAL Application whether these functional blocks are supported or not within
52. the AAL Implementation.
53. Figure 7.1.2-2 illustrates the building blocks for processing various O-DU PHY layer Uplink (UL) channels and signals
54. (with 7.2-x functional split between O-DU and O-RU) defined by 3GPP in [[8]](#_bookmark10) & [[9]](#_bookmark11) as part of 4G/5G NR specification.

L2 +

**NPUSCH**

**UL data**

CRC check

**PUSCH Format 2 UCI**

Layer Demapping

Demodulation

Preamble detection + delay estimation

Channel decoding

Rate dematching

Turbo decoding

Descrambling

AC-NACK detection

**NPRACH**

Rate dematching

Descrambling

Peak search

**DMRS with PUSCH Format 1 and**

**Format 2**

IFFT

Channel equalization

Channel equalization

Noise estimation

Demodulation

IDFT for DFT-s- OFDM

Channel estimation

Channel Estimation

Channel estimation

Root sequence correlation

RE demapping

RE demapping

RE demapping

RE demapping

IQ

decompression\*

IQ

decompression\*

IQ decompression\*

IQ

decompression\*

1

O-RAN FH (7-2x lower layer split)

* 1. **Figure 7.1.2-2: O-DU PHY processing blocks for mMTC Uplink**
  2. The O-DU PHY layer in uplink consists of the following physical channels and reference signals:
  3. • Narrow-band Physical Uplink Shared Channel (NPUSCH).
  4. • Narrow-band Physical Random-Access Channel (NPRACH).
  5. The uplink physical channels (NPUSCH, NPRACH) carry information originating from higher layers (i.e., layer 2 and
  6. above).
  7. The uplink physical layer processing of shared channel (NPUSCH) carrying uplink data with or without Uplink Control
  8. Information (UCI) consists of the following steps at the receiver (O-DU):
  9. RE (de)mapping: Refer to Subclauses 5.3.4 of [[8]](#_bookmark10) for details on RE mapping at the transmitter/receiver.
  10. Channel estimation and equalization: up to O-DU implementation.
  11. Transform precoding (IDFT): optional, only required for DFT-s-OFDM waveform. Refer to Subclause 5.3.3A of [[8]](#_bookmark10) for
  12. details on transform precoding (if applicable) applied at the transmitter.
  13. Demodulation: Refer to Subclause 5.3.2 in [[8]](#_bookmark10) for details on modulation applied at the transmitter.
  14. Descrambling: Refer to Subclause 5.3.1 in [[8]](#_bookmark10) for details on scrambling applied at the transmitter.
  15. Rate de-matching: Refer to Subclause 5.1.4.1 in [[9]](#_bookmark11) for details on rate matching applied at the transmitter.

1. Turbo decoding: Refer to Subclause 5.1.3.2 in [[9]](#_bookmark11) for details on Turbo decoding applied at the transmitter.
2. CRC check: Refer to Subclauses 5.1.1 in [[9]](#_bookmark11) for details on TB and CB level CRC attachments applied at the transmitter.
3. The uplink physical layer processing for shared channel (NPUSCH) carrying UCI depends on NPUSCH formats.
4. An O-DU AAL profile for 4G/5G NR uplink shall specify a set of accelerated functions corresponding to one or more
5. than one physical uplink channel(s) and/or physical uplink signal(s).
6. In addition to the processing blocks mentioned above, each of these uplink physical channels/signals may include an
7. additional functional block, viz. IQ decompression, which is implementation specific and may depend on system
8. configuration/capability. Each of these physical channels/signals can be implemented with/without this optional
9. functional block. The AAL Application interface shall expose to the AAL Application whether these functional blocks
10. are supported or not within the AAL Implementation.

### 7.1.3 O-DU AAL Profile Definitions

1. O-DU AAL profiles are defined below with future specification(s) to define the AAL Application interface for each
2. profile.

#### 7.1.3.1 Profile Definitions General Guidelines

1. 7.1.3.1.1 Naming
2. As discussed above O-DU AAL profiles are specific to one or more physical channel(s) or signal(s) as such should
3. follow the naming guidelines
4.  O-DU AAL profiles shall be prefixed with “AAL\_”
5.  O-DU AAL profiles when specific to a single channel or signal shall include the channel or signal in the name
6. e.g. “AAL\_PUSCH”
7.  O-DU AAL profiles when common across multiple channels or signals shall not include the channel or signal
8. name, instead just reference the Accelerated Function(s), e.g. AAL\_RE-MAPPING
9.  O-DU AAL profiles that include a subset of the functional blocks within a channel or signal shall include a
10. functional description after the channel name e.g. AAL\_PUSCH\_CHANNEL\_ESTIMATION 25

26 7.1.3.1.2 Data Flow

1. O-DU AAL profiles shall specify the data flow supported by the AAL Profile.
2. Look aside data flow implies the remaining functions not included in the Profile that comprise the channel or signal are
3. implemented on the AAL Application or other entity associated with the AAL Application and not implemented in the
4. HW Accelerator.
5. Inline data flow implies that the signal (with a 7-2x PHY functional split) and the IQ data from the AAL Application (in
6. DL) or the decoded bits (in UL) (post processing) are transferred directly from the HW Accelerator to the Fronthaul
7. interface (in DL) or to the AAL Application (in UL).
8. The profile shall specify if the data flow includes only user plane, or only control plane, or both.

|  |  |  |
| --- | --- | --- |
| 1 | 7.1.3.2 | O-DU AAL Profiles |
| 2 | 7.1.3.2.1 | AAL\_MU-MIMO\_PRECODER\_WEIGHTS\_CALC |
|  |  | **User Grouping and Scheduling** |
|  |  |  |
|  |  | **Precoder/BF Weights** |
|  |  | **Calculation** |
|  |  |  |
|  |  | **Link Adaptation, MCS** |
|  |  | **selection** |
|  |  |  |
|  |  | **TB Formation and associated** L2 |
|  |  | **control information** |
|  |  |  |
|  |  |  |
| 3 |  | **High Phy** L1 |

1. **Figure 7.1.3.2.1-1: AAL\_MU-MIMO\_PRECODER\_WEIGHTS\_CALC**
2. The AAL\_MU-MIMO\_PRECODER\_WEIGHTS\_CALC is used by AAL Application to offload beamforming
3. (precoding) weight calculation to the hardware accelerator (HWA) in look-aside acceleration mode. The AAL
4. Application shall provide HWA with all the information required to calculate precoding weights.
5. This profile is implemented as a look aside accelerator.
6. The below figure shows an example use of the AAL\_MU-MIMO\_PRECODER\_WEIGHTS\_CALC in an O-DU with
7. AAL\_DOWNLINK\_HIGH-PHY and AAL\_UPLINK\_HIGH-PHY

GPP/CPU

**AAL Application**

Look-aside AAL Profile for precoder calculation

Inline High Phy AAL Profile for slot level data path procedure

**High PHY Profile**

**Precoder/BF Weights Calculation Function**

HW Accelerator

11

12 **Figure 7.1.3.2.1-2: Example AAL\_MU-MIMO\_PRECODER\_WEIGHTS\_CALC use.**

1 7.1.3.2.2 AAL\_FFT

L2 +

**PUSCH**

**UL data**

|  |
| --- |
| TB CRC check |
| CB CRC + CB  desegmentation |
| LDPC decoding |
| Rate dematching |
| CB  deconcatentation |
| Descrambling |
| Demodulation |
| **IDFT for DFT-s-**  **OFDM** |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 0 UCI**

RE demapping

Sequence detection

|  |
| --- |
| Demodulation |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 1 UCI**

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| **IDFT for DFT-s-**  **OFDM** |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PRACH**

|  |  |  |
| --- | --- | --- |
| Preamble detection + delay estimation | **SRS** | |
| Peak search |
| Noise estimation |
| **IFFT** |
| Root sequence correlation |  | **Channel estimation** |
| RE demapping |  | RE demapping |

**PT-RS**

RE demapping

Sequence detection

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

2

O-RAN FH (7-2x lower layer split)

3 **Figure 7.1.3.2.2-1: AAL\_FFT**

1. The AAL\_FFT is used by application to offload FFT/iFFT processing to the hardware accelerator (HWA) in look- aside
2. acceleration mode. The application shall provide HW Accelerator with all the information required to perform the FFT
3. operations. The AAL\_FFT Profile can be used for 3GPP specification 38.211 clause 5.3, and clause 6.4.1.4. The below
4. list and [Figure 7.1.3.2.2-1: AAL\_FFT](#_bookmark103) highlights the set of accelerated functions that define the AAL\_FFT Profile.
5.  Zero Padding
6.  iDFT
7.  Windowing
8.  DFT
9.  De padding
10.  Timing Error Correction
11.  Frequency De-Windowing
12.  Frequency Re-Sampling
13. The below figure shows an example use of the AAL\_FFT Profile for accelerating the SRS processing in an
14. O-DU. The highlighted blocks show the set the accelerated functions while the clear blocks are implemented
15. in software.

**SRS**

O-RAN FH (7-2x

lower layer split)

SNR and Channel Estimate.

**Freq re-sample\***

IQ

decompression\*

RE demapping

**Ncs DFT**

**Channel estimation**

**Ncs Windowing**

**iDFT**

L2 +

**Zero Padding**

Zadoff-Chu, RB/ Comb Extraction

|  |  |
| --- | --- |
|  | **Depadding** |
|  |
| **Time Error Correction\***  **De-windowing\*** | |

1

**FFT Power\***

* 1. Note The figure represents the two data outputs from the profile, the FFT Power estimate and the
  2. frequency samples.
  3. Note The figure is a representative example implementation of SRS Channel estimation only
  4. **Figure 7.1.3.2.2-2: AAL\_FFT example for SRS Processing**
  5. This profile is implemented as a look aside accelerator.

#### 7.1.3.3 O-DU AAL Profiles for Downlink

* 1. 7.1.3.3.1 AAL\_PDSCH\_FEC
  2. Figure 7.1.3.3.1-1 highlights the set of accelerated functions that define the AAL\_PDSCH\_FEC Profile. These include:
  3.  CRC Generation
  4.  LDPC Encoding
  5.  PDSCH Rate Matching 13

1. The AAL\_PDSCH\_FEC Profile is implemented as a look aside accelerator. The AAL\_PDSCH\_FEC Profile will
2. support both Transport Block and Code Block operations.

L2+

**PDSCH TB(s)**

**TB CRC**

**attachment**

**CB segmentation + CB**

**CRC attachment**

**LDPC encoding**

**PDCCH (DCI)**

**PBCH TB**

|  |
| --- |
| PBCH payload generation |
| Scrambling |
| TB CRC  attachment |
| Polar encoding |
| Rate matching |
| Data scrambling |
| Modulation  (QPSK) |

**Rate matching**

**PDSCH DM-RS**

Precoding\*

Sequence generation

Scrambling

CB concatenation

**PDCCH DM-RS**

RE mapping

Precoding\*

Modulation

Sequence Generation

Modulation

Sequence generation

RE mapping

Precoding\*

Precoding\*

Layer mapping

Modulation

Modulation

|  |
| --- |
| CRC attachment |
| Polar encoding |
| Rate matching |
| Scrambling |
| Modulation  (QPSK) |

**PSS/SSS PBCH DM-RS**

**CSI-RS/PT-RS/TRS**

Modulation

Sequence Generation

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

1

O-RAN FH (7-2x lower layer split)

RE mapping

RE mapping

* 1. **Figure 7.1.3.3.1-1: AAL\_PDSCH\_FEC Profile**
  2. 7.1.3.3.2 AAL\_PDSCH\_HIGH-PHY
  3. Figure 7.1.3.3.2-1 highlights the set of accelerated functions that defines the AAL\_PDSCH\_HIGH-PHY Profile, which
  4. includes the processing of PDSCH TB(s) and associated DM-RS.
  5. The set of accelerated functions associated with the processing of PDSCH TB(s) is as follows:
  6. • TB CRC attachment
  7. • CB segmentation and CRC attachment
  8. • LDPC encoding
  9. • Rate matching
  10. • CB concatenation
  11. • Scrambling
  12. • Modulation
  13. • Layer mapping
  14. • Precoding [1](#_bookmark108)
  15. • RE mapping
  16. • IQ compression[1](#_bookmark107)

1 Configurable functional block, depends on implementation and/or system configuration

1

1. The set of accelerated functions associated with the processing of PDSCH DM-RS is as follows:
2. • PDSCH DM-RS sequence generation
3. • Modulation
4. • Precoding[1](#_bookmark107)
5. • RE mapping
6. • IQ compression[1](#_bookmark107) 8

9 The AAL\_PDSCH\_HIGH-PHY Profile is executed in inline acceleration mode. 10

L2+

**PDSCH TB(s)**

|  |
| --- |
| **TB CRC**  **attachment** |
|  |
| **CB segmentation +**  **CRC attachment** |
|  |
| **LDPC encoding** |
|  |
| **Rate matching** |
|  |
| **CB concatenation** |
|  |
| **Scrambling** |
|  |
| **Modulation** |
|  |
| **Layer mapping** |

**PDSCH DM-RS**

**Sequence**

**generation**

**PDCCH (DCI)**

|  |
| --- |
| CRC attachment |
| Polar encoding |
| Rate matching |
| Scrambling |
| Modulation  (QPSK) |

**PDCCH DM-RS**

**Precoding\***

**Modulation**

**PBCH TB**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PBCH payload generation | **PSS/SSS**  **PBCH DM-RS CSI-RS/PT-RS/TRS** | | | |
| Scrambling |
| TB CRC  attachment |
| Polar encoding |
| Rate matching |
| Data scrambling |  | Sequence Generation | Sequence Generation  Modulation | |
| Modulation  (QPSK) |  | Modulation |
|  | | |
| Precoding\* | | |  | Precoding\* |
| RE mapping | | |  | RE mapping |
|  | | | IQ compression\* | |
| IQ compression\* | | |

|  |  |
| --- | --- |
|  | Sequence generation |
| Modulation |
|  | |
| Precoding\* | |
| RE mapping | |
|  | |
| IQ compression\* | |

11

|  |  |  |
| --- | --- | --- |
| **RE mapping** |  | |
| **IQ compression\*** |  |  |
|  | |
| O-RAN FH (7-2x lower layer split) | | |

1. **Figure 7.1.3.3.2-1: AAL\_PDSCH\_HIGH-PHY Profile**
2. 7.1.3.3.3 AAL\_PDCCH\_HIGH-PHY
3. Figure 7.1.3.3.3-1 highlights the set of accelerated functions that defines the AAL\_PDCCH\_HIGH-PHY Profile, which
4. includes the processing of PDCCH DCI and associated DM-RS.
5. The set of accelerated functions associated with the processing of PDCCH TB(s) is as follows:
6. • CRC attachment
7. • Polar encoding
8. • Rate matching
9. • Scrambling
10. • Modulation (QPSK)
11. • Precoding[1](#_bookmark107)
12. • RE mapping
13. • IQ compression[1](#_bookmark107) 4
14. The set of accelerated functions associated with the processing of PDCCH DM-RS is as follows:
15. • PDCCH DM-RS sequence generation
16. • Modulation
17. • Precoding[1](#_bookmark107)
18. • RE mapping
19. • IQ compression[1](#_bookmark107)
20. The AAL\_PDCCH\_HIGH-PHY Profile is executed in inline acceleration mode.

L2+

**PDSCH TB(s)**

|  |
| --- |
| TB CRC attachment |
|  |
| CB segmentation +  CRC attachment |
|  |
| LDPC encoding |
|  |
| Rate matching |
|  |
| CB concatenation |
|  |
| Scrambling |
|  |
| Modulation |
|  |
| Layer mapping |

12

|  |  |  |  |
| --- | --- | --- | --- |
| Precoding\* | **Precoding\***  RE mapping | Precoding\*  RE mapping | Precoding\*  RE mapping |
|  |
| RE mapping |
|  | | | |
| IQ compression\* | **IQ compression\*** | IQ compression\* | IQ compression\* |
|  | | |
| O-RAN FH (7-2x lower layer split) | | | |

13

**PBCH TB**

|  |
| --- |
| **CRC**  **attachment** |
| **Polar encoding** |
| **Rate matching** |
| **Scrambling** |
| **Modulation**  **(QPSK)** |

|  |
| --- |
| **Sequence** |
| **generation** |
| **Modulation** |

**Figure 7.1.3.3.3-1: AAL\_PDCCH\_HIGH-PHY Profile**

**PDCCH (DCI)**

**PDSCH DM-RS**

**PDCCH DM-RS**

**PSS/SSS**

**PBCH DM-RS**

**CSI-RS/PT-RS/TRS**

Modulation

Sequence Generation

Modulation

Modulation

(QPSK)

Sequence Generation

Data scrambling

Rate matching

Polar encoding

TB CRC

attachment

Scrambling

PBCH payload generation

Modulation

Sequence generation

1. 7.1.3.3.4 AAL\_PBCH\_HIGH-PHY
2. Figure 7.1.3.3.4-1 highlights the set of accelerated functions that defines the AAL\_PBCH\_HIGH-PHY Profile, which
3. includes the processing of PBCH TB and associated DM-RS, PSS and SSS, or in other words, the processing of SSB.
4. The set of accelerated functions associated with the processing of PBCH TB is as follows:
5. • PBCH payload generation
6. • Scrambling
7. • TB CRC attachment
8. • Polar encoding
9. • Rate matching
10. • Data scrambling
11. • Modulation (QPSK)
12. • Precoding[1](#_bookmark107)
13. • RE mapping
14. • IQ compression[1](#_bookmark107) 7
15. The set of accelerated functions associated with the processing of PBCH DM-RS/PSS/SSS is as follows:
16. • PDCCH DM-RS/PSS/SSS sequence generation
17. • Modulation
18. • Precoding[1](#_bookmark107)
19. • RE mapping
20. • IQ compression[1](#_bookmark107) 14

15 The AAL\_PBCH\_HIGH-PHY Profile is executed in inline acceleration mode.

L2+

**PDSCH TB(s)**

**PBCH TB**

TB CRC attachment

CB segmentation +

CB CRC attachment LDPC encoding Rate matching

CB concatenation Scrambling Modulation

Layer mapping

**PDCCH (DCI)**

PBCH payload generation

Scrambling TB CRC

attachment

Polar encoding

Rate matching

Data scrambling

Modulation

(QPSK)

**PDSCH DM-RS**

Sequence generation

**PDCCH DM-RS**

CRC attachment Polar encoding Rate matching Scrambling

Modulation

(QPSK)

**PSS/SSS PBCH DM-RS**

**CSI-RS/PT-RS/TRS**

Modulation

Sequence generation

Modulation

Sequence Generation

Modulation

Sequence Generation

Modulation

16

Precoding\*

Precoding\*

Precoding\*

Precoding\*

RE mapping

RE mapping

RE mapping

RE mapping

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

O-RAN FH (7-2x lower layer split)

17 **Figure 7.1.3.3.4-1: AAL\_PBCH\_HIGH-PHY Profile**

18

1. 7.1.3.3.5 AAL\_CSI-RS\_HIGH-PHY
2. Figure 7.1.3.3.5-1 highlights the set of accelerated functions that defines the AAL\_CSI-RS\_HIGH-PHY Profile, which
3. includes the following:
4. • CSI-RS sequence generation
5. • Modulation
6. • Precoding[1](#_bookmark107)
7. • RE mapping
8. • IQ compression[1](#_bookmark107) 6

7 The AAL\_CSI-RS\_HIGH-PHY Profile is executed in inline acceleration mode.

L2+

**PDSCH TB(s)**

TB CRC attachment

CB segmentation +

CB CRC attachment

LDPC encoding

**PDCCH (DCI)**

**PBCH TB**

|  |
| --- |
| PBCH payload generation |
| Scrambling |
| TB CRC  attachment |
| Polar encoding |
| Rate matching |
| Data scrambling |
| Modulation  (QPSK) |

Rate matching

**PDSCH DM-RS**

Precoding\*

Modulation

Sequence generation

Scrambling

CB concatenation

**PDCCH DM-RS**

RE mapping

Precoding\*

Modulation

Sequence Generation

Modulation

Sequence generation

RE mapping

Precoding\*

Precoding\*

Layer mapping

Modulation

Modulation

|  |
| --- |
| CRC attachment |
| Polar encoding |
| Rate matching |
| Scrambling |
| Modulation  (QPSK) |

**PSS/SSS PBCH DM-RS**

**CSI-RS/PT-RS/TRS**

Sequence Generation

RE mapping

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

8

O-RAN FH (7-2x lower layer split)

RE Mapping

9 **Figure 7.1.3.3.5-1: AAL\_CSI-RS\_HIGH-PHY Profile**

10

1. 7.1.3.3.6 AAL\_PT-RS-DL\_HIGH-PHY
2. Figure 7.1.3.3.6-1 highlights the set of accelerated functions that defines the AAL\_PT-RS-DL\_HIGH-PHY Profile,
3. which includes the following:
4. • PT-RS sequence generation
5. • Modulation
6. • Precoding[1](#_bookmark107)
7. • RE mapping
8. • IQ compression[1](#_bookmark107) 19

20 The AAL\_PT-RS-DL\_HIGH-PHY Profile is executed in inline acceleration mode.

**PDSCH TB(s)**

TB CRC attachment

CB segmentation +

CB CRC attachment

LDPC encoding

**PDCCH (DCI)**

**PBCH TB**

|  |
| --- |
| PBCH payload generation |
| Scrambling |
| TB CRC  attachment |
| Polar encoding |
| Rate matching |
| Data scrambling |
| Modulation  (QPSK) |

Rate matching

**PDSCH DM-RS**

Precoding\*

Modulation

Sequence generation

Scrambling

CB concatenation

**PDCCH DM-RS**

RE mapping

Precoding\*

Modulation

Sequence Generation

Modulation

Sequence generation

RE mapping

Precoding\*

Precoding\*

Layer mapping

Modulation

Modulation

|  |
| --- |
| CRC attachment |
| Polar encoding |
| Rate matching |
| Scrambling |
| Modulation  (QPSK) |

**PSS/SSS PBCH DM-RS**

**PT-RS-DL**

Sequence Generation

RE mapping

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

1

O-RAN FH (7-2x lower layer split)

RE Mapping

1. **Figure 7.1.3.3.6-1: AAL\_PT-RS-DL\_HIGH-PHY Profile**
2. 7.1.3.3.7 AAL\_DOWNLINK\_HIGH-PHY
3. Figure 7.1.3.3.7-1 highlights the set of accelerated functions that defines the AAL\_DOWNLINK\_HIGH-PHY Profile.
4. This profile includes the aggregation of all the individual downlink channel profiles as follows:
5. **PDSCH**
6.  **Data**: see list of accelerated functions associated with the processing of PDSCH TB(s), per section [7.1.3.3.2](#_bookmark106)
7.  **DM-RS**: see list of accelerated functions associated with the processing of PDSCH DM-RS, per section

9 [7.1.3.3.2.](#_bookmark106)

10  **PT-RS**: see list of accelerated functions listed in section [7.1.3.3.6.](#_bookmark116) 11

1. **PDCCH**:
2.  **Data**: see list of accelerated functions associated with the processing of PDCCH TB(s), per section [7.1.3.3.3](#_bookmark110)
3.  **DM-RS**: see list of accelerated functions associated with the processing of PDCCH DM-RS, per section

15 [7.1.3.3.3](#_bookmark110)

16

17 **SSB**:

18  PSS + SSS: see list of accelerated functions associated with the processing of PSS/SSS, per section [7.1.3.3.4](#_bookmark112)

19  PBCH DM-RS: see list of accelerated functions associated with the processing of PBCH DM-RS, per section

20 [7.1.3.3.4](#_bookmark112)

21  PBCH: see list of accelerated functions associated with the processing of PBCH TB(s), per section [7.1.3.3.4](#_bookmark112) 22

23 **CSI-RS:**

24  see list of accelerated functions listed in section [7.1.3.3.5](#_bookmark114) 25

26 The AAL\_DOWNLINK\_HIGH-PHY Profile is executed in inline acceleration mode.

**PDSCH TB(s)**

**TB CRC**

**attachment**

**CB segmentation +**

**CRC attachment**

**PDCCH (DCI)**

**PBCH TB**

|  |
| --- |
| **PBCH payload**  **generation** |
| **Scrambling** |
| **TB CRC**  **attachment** |
| **Polar encoding** |
| **Rate matching** |
| **Data scrambling** |
| **Modulation**  **(QPSK)** |

**Rate matching**

**PDSCH DM-RS**

**Sequence**

**generation**

Modulation

Sequence Generation

**Scrambling**

**CB concatenation**

**LDPC encoding**

|  |
| --- |
| **Sequence** |
| **Generation** |
| **Modulation** |

**PDCCH DM-RS**

**Layer mapping**

**Modulation**

**Modulation**

|  |
| --- |
| **CRC attachment** |
| **Polar encoding** |
| **Rate matching** |
| **Scrambling** |
| **Modulation**  **(QPSK)** |

|  |
| --- |
| **Sequence** |
| **generation** |
| **Modulation** |

**PSS/SSS PBCH DM-RS**

**CSI-RS/PT-RS/TRS**

**Precoding\***

**Precoding\***

**Precoding\***

Precoding\*

**RE mapping**

**RE mapping**

**RE mapping**

RE Mapping

**IQ compression\***

**IQ compression\***

**IQ compression\***

IQ compression\*

1

O-RAN FH (7-2x lower layer split)

1. **Figure 7.1.3.3.7-1AAL\_DOWNLINK\_ HIGH-PHY Profile**

#### 7.1.3.4 O-DU AAL Profiles for Uplink

1. 7.1.3.4.1 AAL\_PUSCH\_FEC
2. Figure 7.1.3.4.1-1 highlights the set of accelerated functions that define the AAL\_PUSCH\_FEC Profile. These include:
3.  PUSCH Rate De-matching
4.  LDPC Decoder
5.  CRC Check 9
6. The AAL\_PUSCH\_FEC Profile is implemented as a look aside accelerator. The AAL\_PUSCH\_FEC Profile will
7. support both Transport Block and Code Block operations.

L2 +

**PUSCH**

**UL data**

|  |
| --- |
| **TB CRC check** |
| **CB CRC + CB** |
| **desegmentation** |
| **LDPC decoding** |
| **Rate dematching** |
| **CB**  **deconcatentation** |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-  OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 0 UCI**

RE demapping

Sequence detection

|  |
| --- |
| Demodulation |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 1 UCI**

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PRACH**

|  |  |  |
| --- | --- | --- |
| Preamble detection + delay estimation | **SRS** | |
| Peak search |
| Noise estimation |
| IFFT |
| Root sequence correlation |  | Channel estimation |
| RE demapping |  | RE demapping |

**PT-RS**

RE demapping

Sequence detection

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

1

O-RAN FH (7-2x lower layer split)

* 1. **Figure 7.1.3.4.1-1: AAL\_PUSCH\_FEC Profile**
  2. 7.1.3.4.2 AAL\_PUSCH\_HIGH-PHY
  3. Figure 7.1.3.4.2-1 highlights the set of accelerated functions that defines the AAL\_PUSCH\_HIGH-PHY Profile, which
  4. includes the processing of PUSCH data (with or without UCI).
  5. The set of accelerated functions associated with the processing of PUSCH data is as follows:
  6. • IQ decompression[1](#_bookmark107)
  7. • RE de-mapping
  8. • Channel estimation
  9. • Channel equalization
  10. • Transform precoding (optional- only required for DFT-s-OFDM waveform)
  11. • Demodulation
  12. • Descrambling
  13. • Rate de-matching
  14. • LDPC decoding
  15. • CRC check 17

18 The AAL\_PUSCH\_HIGH-PHY Profile is executed in inline acceleration mode. 19

1

L2 +

**PUSCH**

**UL data**

|  |
| --- |
| **TB CRC check** |
| **CB CRC + CB** |
| **desegmentation** |
| **LDPC decoding** |
| **Rate dematching** |
| **CB**  **deconcatentation** |
| **Descrambling** |
| **Demodulation** |
| **IDFT for DFT-s-**  **OFDM** |
| **Channel equalization** |
| **Channel estimation** |

**PUCCH format 1 UCI**

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PRACH**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Preamble detection + delay estimation | **SRS PT-RS** | | | |
| Peak search |
| Noise estimation |
| IFFT |
| Root sequence correlation |  | Channel estimation |  | Sequence detection |
| RE demapping |  | RE demapping |  | RE demapping |
| IQ  decompression\* |  | IQ  decompression\* |  | IQ  decompression\* |

**IQ**

**decompression\***

RE mapping

|  |  |  |
| --- | --- | --- |
| **PUCCH format 0 UCI** | | Demodulation |
| Channel equalization |
| Sequence detection |  | Channel estimation |
| RE demapping |  | RE demapping |
| IQ  decompression\* |  | IQ  decompression\* |

2

O-RAN FH (7-2x lower layer split)

3 **Figure 7.1.3.4.2-1 : AAL\_PUSCH\_HIGH-PHY Profile**

4

1. 7.1.3.4.3 AAL\_PUCCH\_HIGH-PHY
2. Figure 7.1.3.4.3.1-1, Figure 7.1.3.4.3.2-1 and Figure 7.1.3.4.3.2-1 highlight the set of accelerated functions that defines
3. the AAL\_PUCCH\_HIGH-PHY Profile, which includes the processing of UCI.
4. The set of accelerated functions associated with the processing of PUCCH UCI depends on the PUCCH format being
5. configured by the AAL Application.
6. 7.1.3.4.3.1 PUCCH format 0
7. The set of accelerated functions associated with the processing of PUCCH UCI using PUCCH format 0 is as follows:
8. • IQ decompression[1](#_bookmark107)
9. • RE de-mapping
10. • Sequence detection

L2 +

**PUSCH**

**UL data**

|  |
| --- |
| TB CRC check |
| CB CRC + CB  Desegmentation |
| LDPC decoding |
| Rate dematching |
| CB  Deconcatenation |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-  OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 0 UCI**

**Sequence detection**

|  |
| --- |
| Demodulation |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 1 UCI**

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PRACH**

|  |  |  |
| --- | --- | --- |
| Preamble detection + delay estimation | **SRS** | |
| Peak search |
| Noise estimation |
| IFFT |
| Root sequence correlation |  | Channel estimation |
| RE demapping |  | RE demapping |

**PT-RS**

RE mapping

RE demapping

Sequence detection

IQ

decompression\*

**IQ**

**decompression\***

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

1

O-RAN FH (7-2x lower layer split)

2 **Figure 7.1.3.4.3.1-1: AAL\_PUCCH\_HIGH-PHY Profile (PUCCH format 0)**

3

4 7.1.3.4.3.2 PUCCH format 1

1. The set of accelerated functions associated with the processing of PUCCH UCI using PUCCH format 1 is as follows:
2. • IQ decompression[1](#_bookmark107)
3. • RE de-mapping
4. • Channel estimation
5. • Channel equalization
6. • Demodulation

11

L2 +

**PUSCH**

**UL data**

|  |
| --- |
| TB CRC check |
| CB CRC + CB  Desegmentation |
| LDPC decoding |
| Rate dematching |
| CB  Deconcatenation |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-  OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 0 UCI**

RE demapping

Sequence detection

|  |
| --- |
| **Demodulation** |
| **Channel equalization** |
| **Channel estimation** |

**PUCCH format 1 UCI**

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PRACH**

|  |  |  |
| --- | --- | --- |
| Preamble detection + delay estimation | **SRS** | |
| Peak search |
| Noise estimation |
| IFFT |
| Root sequence correlation |  | Channel estimation |
| RE demapping |  | RE demapping |

**PT-RS**

RE mapping

RE demapping

Sequence detection

IQ

decompression\*

IQ

decompression\*

**IQ**

**decompression\***

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

1

O-RAN FH (7-2x lower layer split)

* 1. **Figure 7.1.3.4.3.2-1: AAL\_PUCCH\_HIGH-PHY Profile (PUCCH format 1)**

3

4 7.1.3.4.3.3 PUCCH format 2/3/4

1. The set of accelerated functions associated with the processing of PUCCH UCI using PUCCH format 2/3/4 is as
2. follows:
3. • IQ decompression[1](#_bookmark107)
4. • RE de-mapping
5. • Channel estimation
6. • Channel equalization
7. • Transform precoding (optional- only required for DFT-s-OFDM waveform)
8. • Demodulation
9. • Descrambling
10. • Rate de-matching
11. • Polar/Block decoding
12. • CRC check 17

1

L2 +

**PUSCH**

**UL data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TB CRC check | **PUCCH format 1 UCI** | | | |
| CB CRC + CB  Desegmentation |
| LDPC decoding |
| Rate dematching |
| CB  Deconcatenation |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-  OFDM |
| **PUCCH format 0 UCI** | | | Demodulation |
| Channel equalization | Channel equalization |
| Channel estimation |  | Sequence detection |  | Channel estimation |
| RE demapping |  | RE demapping |  | RE demapping |
| IQ  decompression\* |  | IQ  decompression\* |  | IQ  decompression\* |

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| **CRC check** |
| **Polar/Block decoding** |
| **Rate dematching** |
| **Descrambling** |
| **Demodulation** |
| **IDFT for DFT-s-**  **OFDM** |
| **Channel equalization** |
| **Channel estimation** |
|  |
| **RE mapping** |
|  |
| **IQ decompression\*** |

**PRACH**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Preamble  detection + delay estimation | **SRS PT-RS** | | | |
| Peak search |
| Noise estimation |
| IFFT |
| Root sequence correlation |  | Channel estimation |  | Sequence detection |
| RE demapping |  | RE demapping |  | RE demapping |
| IQ  decompression\* |  | IQ  decompression\* |  | IQ  decompression\* |

2

O-RAN FH (7-2x lower layer split)

* 1. **Figure 7.1.3.4.3.3-1: AAL\_PUCCH\_HIGH-PHY Profile (PUCCH format 2/3/4)**
  2. The AAL\_ PUCCH\_ HIGH -PHY profile is executed in inline acceleration mode.
  3. 7.1.3.4.4 AAL\_PRACH\_HIGH-PHY
  4. Figure 7.1.3.4.4-1 highlights the set of accelerated functions that defines the AAL\_PRACH\_HIGH-PHY Profile.
  5. The set of accelerated functions associated with the processing of PRACH preamble is as follows:
  6. • IQ decompression[1](#_bookmark107)
  7. • RE de-mapping
  8. • Root sequence generation and correlation
  9. • IFFT
  10. • Noise estimation
  11. • Peak search for power delay profile
  12. • Preamble detection and delay/timing advance estimation 15

L2 +

**PUSCH**

**UL data**

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PUCCH format 0 UCI**

RE demapping

Sequence detection

|  |
| --- |
| TB CRC check |
| CB CRC + CB  Desegmentation |
| LDPC decoding |
| Rate dematching |
| CB  Deconcatenation |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-  OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |

|  |
| --- |
| Demodulation |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 1 UCI**

**PRACH**

**PT-RS**

RE demapping

Sequence detection

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

**IQ**

**decompression\***

IQ

decompression\*

IQ

decompression\*

1

O-RAN FH (7-2x lower layer split)

|  |  |  |
| --- | --- | --- |
| **Preamble detection**  **+ delay estimation** | **SRS** | |
| **Peak search** |
| **Noise estimation** |
| **IFFT** |
| **Root sequence correlation** |  | Channel estimation |
| **RE mapping** |  | RE demapping |

* + 1. **Figure 7.1.3.4.4-1: AAL\_PRACH\_HIGH-PHY Profile**
    2. The AAL\_PRACH\_HIGH-PHY Profile is executed in inline acceleration mode.
    3. 7.1.3.4.5 AAL\_SRS\_HIGH-PHY
    4. Figure 7.1.3.4.5-1 highlights the set of accelerated functions that defines the AAL\_SRS\_HIGH-PHY Profile.
    5. The set of accelerated functions associated with the processing of SRS is as follows:
    6. • IQ decompression[1](#_bookmark107)
    7. • RE de-mapping
    8. • Channel estimation 10

L2 +

**PUSCH**

**UL data**

|  |
| --- |
| TB CRC check |
| CB CRC + CB  Desegmentation |
| LDPC decoding |
| Rate dematching |
| CB  Deconcatenation |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-  OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 0 UCI**

RE demapping

Sequence detection

|  |
| --- |
| Demodulation |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 1 UCI**

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PRACH**

|  |  |  |
| --- | --- | --- |
| Preamble detection + delay estimation | **SRS** | |
| Peak search |
| Noise estimation |
| IFFT |
| Root sequence correlation |  | **Channel estimation** |
| RE demapping |  | **RE mapping** |

**PT-RS**

RE demapping

Sequence detection

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

**IQ**

**decompression\***

IQ

decompression\*

1

O-RAN FH (7-2x lower layer split)

1. **Figure 7.1.3.4.5-1: AAL\_SRS\_HIGH-PHY Profile**
2. The AAL\_SRS\_ HIGH-PHY Profile is executed in inline acceleration mode.
3. 7.1.3.4.6 AAL\_PT-RS-UL\_HIGH-PHY
4. Figure 7.1.3.4.6-1 highlights the set of accelerated functions that defines the AAL\_PT-RS-UL\_HIGH-PHY Profile.
5. The set of accelerated functions associated with the processing of PT-RS-UL sequence is as follows:
6. • IQ decompression[1](#_bookmark107)
7. • RE de-mapping
8. • Sequence detection

L2 +

**PUSCH**

**UL data**

|  |
| --- |
| TB CRC check |
| CB CRC + CB  Desegmentation |
| LDPC decoding |
| Rate dematching |
| CB  Deconcatenation |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-  OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 0 UCI**

RE demapping

Sequence detection

|  |
| --- |
| Demodulation |
| Channel equalization |
| Channel estimation |
| RE demapping |

**PUCCH format 1 UCI**

**PUCCH format 2/3/4**

**UCI**

|  |
| --- |
| CRC check |
| Polar/Block decoding |
| Rate dematching |
| Descrambling |
| Demodulation |
| IDFT for DFT-s-OFDM |
| Channel equalization |
| Channel estimation |
| RE demapping |
| IQ decompression\* |

**PRACH**

|  |  |  |
| --- | --- | --- |
| Preamble detection + delay estimation | **SRS** | |
| Peak search |
| Noise estimation |
| IFFT |
| Root sequence correlation |  | Channel estimation |
| RE demapping |  | RE demapping |

**PT-RS**

**Sequence detection**

**RE mapping**

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

IQ

decompression\*

**IQ**

**decompression\***

1

O-RAN FH (7-2x lower layer split)

1. **Figure 7.1.3.4.6-1: AAL\_PT-RS-UL\_HIGH-PHY profile**
2. The AAL\_PT-RS-UL\_HIGH-PHY profile is executed in inline acceleration mode.
3. 7.1.3.4.7 AAL\_UPLINK\_HIGH-PHY
4. Figure 7.1.3.4.7-1 highlights the set of accelerated functions that defines the AAL\_UPLINK\_HIGH-PHY Profile, this
5. profile includes the aggregation of all the individual uplink channel profiles as follows:
6. **PUSCH**:
7.  **Data:** see list of accelerated functions associated with the processing of PUSCH data, per section [7.1.3.4.2](#_bookmark120)
8.  **DM-RS**: see list of accelerated functions listed in section [7.1.3.4.6,](#_bookmark132) implemented to process DM-RS IQ
9. samples.
10.  **PT-RS:** see list of accelerated functions listed in section [7.1.3.4.6](#_bookmark132) 12
11. **PUCCH:**
12.  **Format 0:** see list of accelerated functions listed in section [7.1.3.4.3.1](#_bookmark122)
13.  **Format 1:**
14. o **UCI:** see list of accelerated functions listed in section [7.1.3.4.3.2](#_bookmark124)
15. o **DM-RS:** see list of accelerated functions listed in section [7.1.3.4.6,](#_bookmark132) implemented to process DM-RS
16. IQ samples.
17.  **Formats 2/3/4:**
18. o **UCI:** see list of accelerated functions listed in section [7.1.3.4.3.3](#_bookmark126)
19. o **DM-RS:** see list of accelerated functions listed in section [7.1.3.4.6,](#_bookmark132) implemented to process DM-RS
20. IQ samples.

23

1. **PRACH:**
2.  see list of accelerated functions listed in section [7.1.3.4.4](#_bookmark128) 26
   1. **SRS:**
   2.  see list of accelerated functions listed in section [7.1.3.4.5](#_bookmark130) 3

4 The AAL\_UPLINK\_HIGH-PHY Profile is executed in inline acceleration mode. 5

L2 +

**PUSCH**

**UL data**

**PUCCH format 2/3/4 UCI**

**PRACH**

|  |
| --- |
| **Preamble detection**  **+ delay estimation** |
| **Peak search** |
| **Noise estimation** |
| **IFFT** |
| **Root sequence correlation** |
| **RE mapping** |

**PUCCH format 0 UCI**

**Sequence detection**

|  |
| --- |
| **TB CRC check** |
| **CB CRC + CB** |
| **desegmentation** |
| **LDPC decoding** |
| **Rate dematching** |
| **CB**  **deconcatentation** |
| **Descrambling** |
| **Demodulation** |
| **IDFT for DFT-s-**  **OFDM** |
| **Channel equalization** |
| **Channel estimation** |

**PUCCH format 1**

**UCI**

|  |
| --- |
| **Demodulation** |
| **Channel equalization** |
| **Channel estimation** |

**SRS**

**Channel estimation**

**RE mapping**

**RE mapping**

**RE mapping**

RE mapping

|  |
| --- |
| **CRC check** |
| **Polar/Block decoding** |
| **Rate dematching** |
| **Descrambling** |
| **Demodulation** |
| **IDFT for DFT-s- OFDM** |
| **Channel equalization** |
| **Channel estimation** |

**PT-RS**

**Sequence detection**

**IQ**

**decompression\***

RE mapping

RE mapping

6

O-RAN FH (7-2x lower layer split)

**IQ**

**decompression\***

**IQ**

**decompression\***

**IQ**

**decompression\***

**IQ**

**decompression\***

**IQ**

**decompression\***

**IQ decompression\***

7 **Figure 7.1.3.4.7-1: AAL\_UPLINK\_ HIGH-PHY Profile**

8

#### 7.1.3.5 O-DU AAL Profiles for mMTC

1. 7.1.3.5.1 AAL\_NPDSCH\_FEC
2. [Figure 7.1.3.5.1-1:](#_bookmark135) highlights the set of accelerated functions that define the AAL\_NPDSCH\_FEC Profile. These
3. include:
4. • CRC Generation
5. • Tail-Biting Convolutional Coding
6. • Rate Matching
7. The AAL\_NPDSCH\_FEC Profile is implemented as a look aside accelerator.

1

L2+

**NPDSCH TB(s)**

###### NPBCH TB

**TB CRC**

**attachment**

**N PDCCH (DCI)**

CRC

attachment

PBCH payload generation

**Tail Biting Convolutional Coding**

**Rate matching**

CRC attachment

Tail Biting

Convolutional Coding

###### NPSS/NSSS/ NRS

Layer mapping

Modulation

Modulation

Layer Mapping

Modulation

(QPSK)

Modulation

Scrambling

Sequence Generation

Scrambling

Rate matching

Scrambling

CB concatenation

Rate Matching

Tail Biting

Convolutional Coding

###### NPRS/NWUS

Sequence Generation

Modulation

Precoding\*

Precoding\*

Precoding\*

Precoding\*

RE mapping

RE mapping

RE mapping

RE mapping

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

2

O-RAN FH (7-2x lower layer split)

* 1. **Figure 7.1.3.5.1-1: AAL\_NPDSCH\_FEC Profile**
  2. 7.1.3.5.2 AAL\_NPDCCH\_FEC
  3. Figure 7.1.3.5.2-1 highlights the set of accelerated functions that define the AAL\_NPDCCH\_FEC Profile. These
  4. include
  5. • CRC Generation
  6. • Tail-Biting Convolutional Coding
  7. • Rate Matching
  8. The AAL\_NPDCCH\_FEC Profile is implemented as a look aside accelerator.

L2+

**NPDSCH TB(s)**

###### NPBCH TB

TB CRC attachment

**N PDCCH (DCI)**

CRC

attachment

PBCH payload generation

Tail Biting Convolutional Coding

Rate matching

**CRC attachment**

Tail Biting

Convolutional Coding

###### NPSS/NSSS/ NRS

Layer mapping

Modulation

Modulation

Layer Mapping

Modulation

(QPSK)

Modulation

Scrambling

Sequence Generation

Scrambling

**Rate matching**

Scrambling

CB concatenation

Rate Matching

**Tail Biting**

**Convolutional Coding**

###### NPRS/NWUS

Sequence Generation

Modulation

Precoding\*

Precoding\*

Precoding\*

Precoding\*

RE mapping

RE mapping

RE mapping

RE mapping

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

1

O-RAN FH (7-2x lower layer split)

* + 1. **Figure 7.1.3.5.2-1: AAL\_NPDCCH\_FEC Profile**
    2. 7.1.3.5.3 AAL\_NPBCH\_FEC
    3. Figure 7.1.3.5.3-1 highlights the set of accelerated functions that define the AAL\_NPBCH\_FEC Profile. These include
    4. • CRC Generation
    5. • Tail-Biting Convolutional Coding
    6. • Rate Matching
    7. The AAL\_NPBCH\_FEC Profile is implemented as a look aside accelerator.

L2+

**NPDSCH TB(s)**

###### NPBCH TB

TB CRC attachment

**N PDCCH (DCI)**

**CRC**

**attachment**

PBCH payload generation

Tail Biting Convolutional Coding

Rate matching

CRC attachment

**Tail Biting**

**Convolutional Coding**

###### NPSS/NSSS/ NRS

Layer mapping

Modulation

Modulation

Layer Mapping

Modulation

(QPSK)

Modulation

Scrambling

Sequence Generation

Scrambling

Rate matching

Scrambling

CB concatenation

**Rate Matching**

Tail Biting

Convolutional Coding

###### NPRS/NWUS

Sequence Generation

Modulation

Precoding\*

Precoding\*

Precoding\*

Precoding\*

RE mapping

RE mapping

RE mapping

RE mapping

IQ compression\*

IQ compression\*

IQ compression\*

IQ compression\*

1

O-RAN FH (7-2x lower layer split)

1. **Figure 7.1.3.5.3-1: AAL\_NPBCH\_FEC Profile**
2. 7.1.3.5.4 AAL\_NPUSCH\_FEC
3. Figure 7.1.3.5.4-1 highlights the set of accelerated functions that define the AAL\_NPUSCH\_FEC Profile. These include
4. • CRC Generation
5. • Turbo Decoding
6. • Rate Matching
7. The AAL\_NPUSCH\_FEC Profile is implemented as a look aside accelerator.

L2 +

**NPUSCH**

**UL data**

**CRC check**

**PUSCH Format 2 UCI**

Layer Demapping

Demodulation

Preamble detection + delay estimation

Channel decoding

**Rate dematching**

**Turbo decoding**

Descrambling

AC-NACK detection

**NPRACH**

Rate dematching

Descrambling

Peak search

**DMRS with PUSCH Format 1 and**

**Format 2**

IFFT

Channel equalization

Channel equalization

Noise estimation

Demodulation

IDFT for DFT-s- OFDM

Channel estimation

Channel Estimation

Channel estimation

Root sequence correlation

RE demapping

RE demapping

RE demapping

RE demapping

IQ

decompression\*

IQ

decompression\*

IQ decompression\*

IQ

decompression\*

1

O-RAN FH (7-2x lower layer split)

2 **Figure 7.1.3.5.4-1: AAL\_NPUSCH\_FEC Profile**

|  |  |  |
| --- | --- | --- |
| 3 | 7.2 | O-CU AAL Profiles |
| 4 | The O-CU | AAL profiles shall be part of further study for O-RAN WG6. |
| 5 |  | |
| 6 |
| 7 |
| 8 |
| 9 |

# 1 Annex (informative):

2 Change History

|  |  |  |
| --- | --- | --- |
| **Date** | **Revision** | **Description** |
| 2024.11.25 | 10.00.03 | Update AAL FFT Profile functionality |
| 2024.11.25 | 10.00.02 | Intorduce AAL AF Bypass functionality in 6.2.5. |
| 2024.10.09 | 10.00.01 | Introducing a definition for what it means to have a declarative approach in new clause 6.2.7 |
| 2024.07.19 | 09.00.05 | Minor editorial fix to figure 4.2.2-1 archicture diagram, change O1dms to O1. |
| 2024.07.04 | 09.00.04 | Editorial changes to update figures throughout the document.  Updated the number of the LPU’s on architecture diagram. Terminate IMS interface to HAM in example HAM software deployment scenarios. |
| 2024.07.04 | 09.00.03 | Clause 4.2.3 add requirement stating that the scope of the HAM identifier is within an O-Cloud instance. |
| 2024.07.04 | 09.00.02 | Adding picture for inline example in General Aspetcs section as it can be confusing for reader as inline is missing from general discription. |
| 2024.05.09 | 09.00.01 | Editorial changes for removal of “cloudifed RAN” which is not definied on higher levels |
| 2024.03.22 | 08.00.06 | Updated with minor editorial fixes for publication. |
| 2024.03.07 | 08.00.05 | Modified high level architecture diagram with deletion of the interface between DMS and NF so that readers are not confused about API which is not existent yet and is out of scope of this document. |
| 2024.03.07 | 08.00.04 | Updates from CR [ERI-2024.1.23-WG6-CR-0046-AAL GAP CADS reference](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI-2024.1.23-WG6-CR-0046-AAL%20GAP%20CADS%20reference%20correction.docx?api=v2) [correction.docx](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI-2024.1.23-WG6-CR-0046-AAL%20GAP%20CADS%20reference%20correction.docx?api=v2)   1. “VOID” the reference to the CADS TR in the normative reference section. 2. Change the reference to CADS for O-Cloud Platform Software to a reference of the WG1 OAD TS. 3. Change the reference to CADS for the synchronization topologies to the WG4 CUS TS. |
| 2024.03.07 | 08.00.03 | 1. Correct definitions of Accelerated Function and AAL Profile to remove the constraint that it be a “Cloudified Network Function” (that term is no longer defined anyway). 2. Add clarification text to figure 4.2-1 to point out that support of O1 by the AAL Application is not a requirement. 3. Move the text at the bottom of the terms and conditions section to section 4.2. The text is unchanged. 4. Update Figure 4.2-5 to change “NF Application” to “AAL Application”. |
| 2024.01.18 | 08.00.02 | To improve readability and improve clarity, existing text was moved to different clauses and additional clarification points were considered from [DCM-2023.10.24-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/DCM-2023.10.24-WG6-GAP-updates_section_2.2_v03.docx?api=v2) [WG6-GAP-updates\_section\_2.2\_v03](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/DCM-2023.10.24-WG6-GAP-updates_section_2.2_v03.docx?api=v2) |
| 2024.01.18 | 08.00.01 | Updated diagrams to align format. |

|  |  |  |
| --- | --- | --- |
| 2023.11.28 | 08.00 | Published from version 07.00.09 |
| 2023.10.14 | 07.00.09 | Editorial Updates, fix broken reference. |
| 2023.10.11 | 07.00.08 | Update format to latest O-RAN TS template |
| 2023.10.11 | 07.00.07 | Add text that describes the DMS to AAL Application interface. [https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/QCM-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/QCM-2023.10.24-WG6-CR-0048-Add_AAL_DMS_Description.docx?api=v2) [2023.10.24-WG6-CR-0048-Add\_AAL\_DMS\_Description.docx?api=v2](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/QCM-2023.10.24-WG6-CR-0048-Add_AAL_DMS_Description.docx?api=v2)  Editorial updates. |
| 2023.10.11 | 07.00.06 | Avoid mixing definitions with characteristics and requirements, clause 1.3.1 and  4.1. Updated definition section to provide additional clarity. |
| 2023.09.11 | 07.00.05 | Editorial updates in clauses 1.2, 1.3.2, 2.1, 3.1,4.2 [https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/DCM-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/DCM-2023.10.24-WG6-GAP_editorial_updates_v03.docx?api=v2) [2023.10.24-WG6-GAP\_editorial\_updates\_v03.docx?api=v2](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/DCM-2023.10.24-WG6-GAP_editorial_updates_v03.docx?api=v2) |
| 2023.09.11 | 07.00.04 | Removed reference to alarms which are no longer in scope of this document. |
| 2023.09.11 | 07.00.03 | Add example diagrams of how AAL Software drivers are deployments, added clarification on how AAL SW drivers are installed and managed.  [https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/INT-10-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/INT-10-06-23-WG6-CR-18-AAL-GAnP-driver-deployments-clarification.docx?api=v2) [06-23-WG6-CR-18-AAL-GAnP-driver-deployments-clarification.docx?api=v2](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/INT-10-06-23-WG6-CR-18-AAL-GAnP-driver-deployments-clarification.docx?api=v2) |
| 2023.09.11 | 07.00.02 | Captured the architectural aspects for use of AAL APIs without impacting the O- RAN architecture. [https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI.AO-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI.AO-2023.08.03-WG6-CR-006-AAL%20GAP%20Agreements%20to%20support%20open%20multi-vendor%20Interfaces%20for%20AAL%20APIs.docx?api=v2) [2023.08.03-WG6-CR-006-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI.AO-2023.08.03-WG6-CR-006-AAL%20GAP%20Agreements%20to%20support%20open%20multi-vendor%20Interfaces%20for%20AAL%20APIs.docx?api=v2)  [AAL%20GAP%20Agreements%20to%20support%20open%20multi-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI.AO-2023.08.03-WG6-CR-006-AAL%20GAP%20Agreements%20to%20support%20open%20multi-vendor%20Interfaces%20for%20AAL%20APIs.docx?api=v2) [vendor%20Interfaces%20for%20AAL%20APIs.docx?api=v2](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI.AO-2023.08.03-WG6-CR-006-AAL%20GAP%20Agreements%20to%20support%20open%20multi-vendor%20Interfaces%20for%20AAL%20APIs.docx?api=v2) |
| 2023.09.11 | 07.00.01 | Add description of a software repository use by HAM in section 1.3.1. [https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI-2023.08.08-WG6-CR-005-AAL%20GAP%20SW%20Repo%20Clarifications.docx?api=v2) [2023.08.08-WG6-CR-005-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI-2023.08.08-WG6-CR-005-AAL%20GAP%20SW%20Repo%20Clarifications.docx?api=v2)  [AAL%20GAP%20SW%20Repo%20Clarifications.docx?api=v2](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI-2023.08.08-WG6-CR-005-AAL%20GAP%20SW%20Repo%20Clarifications.docx?api=v2) |
| 2023.14.07 | 06.00.07 | Implemented CR from [https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/INT-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/INT-VMW-06-30-23-WG6-CR-AAL-GAnP-Editorial-rearrange-principles-and-arch_v1.docx?api=v2) [VMW-06-30-23-WG6-CR-AAL-GAnP-Editorial-rearrange-principles-and-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/INT-VMW-06-30-23-WG6-CR-AAL-GAnP-Editorial-rearrange-principles-and-arch_v1.docx?api=v2) [arch\_v1.docx?api=v2](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/INT-VMW-06-30-23-WG6-CR-AAL-GAnP-Editorial-rearrange-principles-and-arch_v1.docx?api=v2) Move 2.5 to chapter 3 and move 3.1 & 3.2 to chapter 2 to better align with chapter contents. |
| 2023.14.07 | 06.00.06 | Implemented CR from [SLA-2023.05.16-ORAN.WG6-CR 03-AAL-GAnP-R003-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/SLA-2023.05.16-ORAN.WG6-CR%20003-AAL-GAnP-R003-v06.00.docx?api=v2)  [v06.00.doc.](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/SLA-2023.05.16-ORAN.WG6-CR%20003-AAL-GAnP-R003-v06.00.docx?api=v2) State that HAM is part of and is integrated along with O-Cloud Platform software |
| 2023.30.06 | 06.00.05 | Implemented CR from [VMW-2023.05.31-WG6-CR-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/VMW-2023.05.31-WG6-CR-AAL%20Architecture%20in%20GAnP.docx?api=v2) [AAL%20Architecture%20in%20GAnP.docx](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/VMW-2023.05.31-WG6-CR-AAL%20Architecture%20in%20GAnP.docx?api=v2) |
| 2023.30.06 | 06.00.04 | Implemented CR from [SLA-2023.05.17-ORAN.WG6.CR%20004-AAL-GAnP-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/SLA-2023.05.17-ORAN.WG6.CR%20004-AAL-GAnP-R003-v06.00.docx?api=v2)  [R003-v06.00.docx](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/SLA-2023.05.17-ORAN.WG6.CR%20004-AAL-GAnP-R003-v06.00.docx?api=v2) |
| 2023.30.06 | 06.00.03 | Implemented CR from [DCM-2023.05.17-WG6-AAL-ETSI](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/DCM-2023.05.17-WG6-AAL-ETSI%20NFV%20description%20update%20in%20Gap.docx?api=v2) |
| 2023.30.06 | 06.00.02 | Implemented CR from [QCM-2023.04-24-WG6-CR-0034-AAL-GAnP-Clarify-](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/QCM-2023.04-24-WG6-CR-0034-AAL-GAnP-Clarify-AAL-Mgmt-Functions.docx?api=v2) [AAL-Mgmt-Functions.docx](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/QCM-2023.04-24-WG6-CR-0034-AAL-GAnP-Clarify-AAL-Mgmt-Functions.docx?api=v2) |

|  |  |  |
| --- | --- | --- |
| 2023.27.04 | 06.00.01 | Implemented CR from [ERI-202304.02-WG6\_CR\_AAL-GAnP Editorial.docx](https://oranalliance.atlassian.net/wiki/download/attachments/2210398415/ERI-202304.02-WG6_CR_AAL-GAnP%20Editorial.docx?api=v2%20%20) and various editorial comments from [July Train WIKI](https://oranalliance.atlassian.net/wiki/spaces/COWG/pages/2813329465/July%2B2023%2BAAL%2BCR%2BComment%2BWiki) |
| 2023.10.03 | 05.00.04 | Definition of new FFT Profile |
| 2023.10.03 | 05.00.03 | Describe the chaining of AAL Profile Instnaces in Sec 4.1.6 AAL-LPU Principles |
| 2023.10.03 | 05.00.02 | Modification of Figure 2.7 and 2.8 in GAnP, changes to sections 2.5.2 and 5.1.3.1.2.. |
| 2022.09.12 | 05.00.01 | Implemented CR from QCM-24 Editorial modifications to correct AAL terms |
| 2022.11.07 | 04.00.04 | Implemented CR from INT-0014 (example AAL-LPU mapping diagrams) |
| 2022.11.07 | 04.00.03 | Implemented CR from QCM-0013 (ER Diagram) |
| 2022.11.07 | 04.00.02 | Implemented CR from QCM-0020 (AAL App Definition) |
| 2022.10.25 | 04.00.01 | Implemented CRs NOK-0001 (AALProfile\_v02) |
| 2022.09.02 | 04.00 | Final version 04.00 |

1