

ISO/IEC **FDIS 23092-2:20202023(E)**

ISO JTC 1/SC 29/WG 11

Secretariat: **XXXXJISC**

Date: 2023-11-13

**Information technology — Genomic information representation — Part 2:
Coding of genomic information**

iTeh Standards
(<https://standards.itih.ai>)
Document Preview

[ISO/IEC FDIS 23092-2](https://standards.itih.ai/catalog/standards/sist/8ca23faf-55be-4c1d-a61e-8027c513fe5d/iso-iec-fdis-23092-2)

<https://standards.itih.ai/catalog/standards/sist/8ca23faf-55be-4c1d-a61e-8027c513fe5d/iso-iec-fdis-23092-2>

© ISO-2020/IEC 2023, Published in Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office

Ch. de Blandonnet 8 • CP 401

CH-1214 Vernier, Geneva, Switzerland

Tel. + 41 22 749 01 11

Fax + 41 22 749 09 47

copyright@iso.org

www.iso.org

www.iso.org

iTeh Standards
(<https://standards.iteh.ai>)
Document Preview

ISO/IEC FDIS 23092-2

<https://standards.iteh.ai/catalog/standards/sist/8ca23faf-55be-4c1d-a61e-8027c513fe5d/iso-iec-fdis-23092-2>

Contents

| | |
|--|-----------|
| Foreword | 7 |
| Introduction | 9 |
| 1 — Scope | 1 |
| 2 — Normative references | 1 |
| 3 — Terms and definitions | 1 |
| 4 — Abbreviated terms | 6 |
| 5 — Conventions | 7 |
| 5.1 — General | 7 |
| 5.2 — Arithmetic operators | 7 |
| 5.3 — Logical operators | 7 |
| 5.4 — Relational operators | 7 |
| 5.5 — Bit-wise operators | 8 |
| 5.6 — Assignment operators | 8 |
| 5.7 — Range notation | 8 |
| 5.8 — Mathematical functions | 9 |
| 5.9 — Order of operation precedence | 9 |
| 5.10 — Variables, syntax elements and tables | 10 |
| 5.11 — Text description of logical operators | 11 |
| 5.12 — Processes | 12 |
| 6 — Syntax and semantics | 12 |
| 6.1 — Method of specifying syntax in tabular form | 12 |
| 6.2 — Bit ordering | 14 |
| 6.3 — Specification of syntax functions and data types | 14 |
| 6.4 — Semantics | 15 |
| 7 — Data structures | 15 |
| 7.1 — General | 15 |
| 7.2 — Data unit | 15 |
| 7.3 — Raw reference | 17 |
| 7.3.1 — General | 17 |
| 7.3.2 — Syntax and semantics | 17 |
| 7.4 — Parameter set | 17 |
| 7.4.1 — Syntax and semantics | 17 |
| 7.4.2 — Encoding parameters | 18 |
| 7.5 — Access unit | 25 |
| 7.5.1 — Syntax and semantics | 25 |
| 8 — Descriptors | 30 |
| 9 — Sequencing reads | 33 |
| 9.1 — General | 33 |
| 9.2 — Supported symbols | 34 |
| 9.3 — Paired-end reads | 35 |
| 9.4 — Reverse-complement reads | 36 |
| 9.5 — Data classes | 36 |
| 9.6 — Aligned data | 36 |
| 9.7 — Unaligned data | 37 |
| 10 — Decoding process | 38 |

| | | |
|----------------|---|------------|
| 10.1 | General | 38 |
| 10.2 | dataset_type = 0 or 1 | 39 |
| 10.2.1 | General | 39 |
| 10.2.2 | References padding | 39 |
| 10.2.3 | Type 1 AU (Class P) | 40 |
| 10.2.4 | Type 2 AU (Class N) | 41 |
| 10.2.5 | Type 3 AU (Class M) | 41 |
| 10.2.6 | Type 4 AU (Class I) | 42 |
| 10.2.7 | Type 5 AU (Class HM) | 44 |
| 10.2.8 | Type 6 AU (Class U) | 44 |
| 10.3 | dataset_type = 2 | 45 |
| 10.3.1 | General | 45 |
| 10.3.2 | Type 1 AU | 46 |
| 10.3.3 | Type 2 AU | 47 |
| 10.3.4 | Type 3 AU | 47 |
| 10.3.5 | Type 4 AU | 47 |
| 10.3.6 | Type 6 AU | 47 |
| 10.4 | Genomic descriptors | 48 |
| 10.4.1 | General | 48 |
| 10.4.2 | pos | 48 |
| 10.4.3 | rcomp | 49 |
| 10.4.4 | flags | 50 |
| 10.4.5 | mmpos | 51 |
| 10.4.6 | mmtyp | 53 |
| 10.4.7 | clips | 58 |
| 10.4.8 | ureads | 61 |
| 10.4.9 | rln | 61 |
| 10.4.10 | pair | 63 |
| 10.4.11 | mscore | 71 |
| 10.4.12 | mmap | 72 |
| 10.4.13 | msar | 75 |
| 10.4.14 | rtype | 76 |
| 10.4.15 | rgroup | 78 |
| 10.4.16 | qv | 78 |
| 10.4.17 | rname | 83 |
| 10.4.18 | rftp | 84 |
| 10.4.19 | rftt | 84 |
| 10.4.20 | tokentype descriptors | 85 |
| 10.5 | sequence | 94 |
| 10.5.1 | General | 94 |
| 10.5.2 | Aligned reads (Classes P, N, M, I, HM) | 94 |
| 10.5.3 | Unmapped reads (Class HM, U) | 95 |
| 10.6 | e-cigar | 96 |
| 10.6.1 | Syntax | 96 |
| 10.6.3 | Decoding process for other alignments | 106 |
| 10.6.4 | Reference transformation | 106 |
| 11 | Representation of reference sequences | 107 |
| 11.1 | External reference | 108 |
| 11.2 | Embedded reference | 108 |
| 11.3 | Computed reference | 108 |
| 11.3.1 | General | 108 |
| 11.3.2 | Supported Algorithms | 108 |

| | | |
|---------|--|-----|
| 11.3.3 | Reference transformation | 109 |
| 11.3.4 | PushIn | 109 |
| 11.3.5 | Local assembly | 111 |
| 11.3.6 | Global assembly | 112 |
| 12 | Block payload parsing process | 113 |
| 12.1 | General | 113 |
| 12.2 | Encoding Mode 0 | 113 |
| 12.3 | Inverse binarizations | 115 |
| 12.3.1 | General | 115 |
| 12.3.2 | Binary (BI) | 116 |
| 12.3.3 | Truncated unary (TU) | 116 |
| 12.3.4 | Exponential golomb (EG) | 116 |
| 12.3.5 | Truncated exponential golomb (TEG) | 117 |
| 12.3.6 | Signed truncated exponential golomb (STEG) | 117 |
| 12.3.7 | Split unit-wise truncated unary (SUTU) | 118 |
| 12.3.8 | Signed split unit-wise truncated unary (SSUTU) | 119 |
| 12.3.9 | Double truncated unary (DTU) | 119 |
| 12.3.10 | Signed double truncated unary (SDTU) | 120 |
| 12.4 | Decoder configuration | 120 |
| 12.4.1 | Sequences and quality values | 120 |
| 12.4.2 | Support values | 121 |
| 12.4.3 | CABAC binarizations | 122 |
| 12.4.4 | Transformation parameters | 125 |
| 12.4.5 | Msar descriptor and read identifiers | 128 |
| 12.4.6 | State variables | 130 |
| 12.6 | Arithmetic decoding engine | 134 |
| 12.6.1 | Initialization | 134 |
| 12.6.2 | Arithmetic decoding process | 134 |
| 12.7 | Decoding process for sequence descriptors | 142 |
| 12.7.1 | General | 142 |
| 12.7.2 | Block payload decoding process | 143 |
| 12.8 | BSC decoding process | 162 |
| 12.8.1 | decoding process | 162 |
| 13 | Output format | 164 |
| 13.1 | General | 164 |
| 13.2 | MPEG-G record | 164 |
| 13.2.1 | General | 164 |
| 13.2.2 | number_of_template_segments | 167 |
| 13.2.3 | number_of_record_segments | 167 |
| 13.2.4 | number_of_alignments | 168 |
| 13.2.5 | class_ID | 168 |
| 13.2.6 | read_group_len | 168 |
| 13.2.7 | reserved | 168 |
| 13.2.8 | read_1_first | 168 |
| 13.2.9 | seq_ID | 168 |
| 13.2.10 | as_depth | 168 |
| 13.2.11 | read_len | 168 |
| 13.2.12 | qv_depth | 168 |
| 13.2.13 | read_name_len | 169 |
| 13.2.14 | read_name | 169 |
| 13.2.15 | read_group | 169 |
| 13.2.16 | sequence | 169 |

| | | |
|--|---|-------------|
| 13.2.17 | quality_values | 169 |
| 13.2.18 | mapping_pos | 169 |
| 13.2.19 | ecigar_len | 169 |
| 13.2.20 | ecigar_string | 169 |
| 13.2.21 | reverse_comp | 169 |
| 13.2.22 | mapping_score | 169 |
| 13.2.23 | split_alignment | 169 |
| 13.2.24 | delta | 170 |
| 13.2.25 | split_pos | 170 |
| 13.2.26 | split_seq_ID | 170 |
| 13.2.27 | flags | 170 |
| 13.2.28 | more_alignments | 170 |
| 13.2.29 | next_pos | 170 |
| 13.2.30 | next_seq_ID | 170 |
| 13.3 | Initialization process | 170 |
| Annex A (informative) Tokenization of reads identifiers | | 174 |
| Annex B (informative) Mapping quality | | 176 |
| Annex C (informative) Inverse binarization examples | | 177 |
| C.1 | Binary (BI) binarization | 177 |
| C.2 | Truncated unary (TU) binarization | 177 |
| C.3 | Exponential golomb (EG) binarization | 177 |
| C.4 | Truncated exponential golomb (TEG) binarization | 178 |
| C.5 | Signed truncated exponential golomb (STEG) binarization | 178 |
| C.6 | Split unit-wise truncated unary (SUTU) binarization | 178 |
| C.7 | Signed split unit-wise truncated unary (SSUTU) binarization | 179 |
| C.8 | Double truncated unary (DTU) binarization | 179 |
| C.9 | Signed double truncated unary (SDTU) binarization | 180 |
| Annex D Block Sorting, Lossless Data Compression | | 181 |
| D.1 | Block Information | 181 |
| D.2 | BWT strings reconstruction | 183 |
| D.3 | Reconstructs the original string from Sort Transform | 184 |
| D.4 | Reconstructs the original string from Sort Transform | 185 |
| Foreword | | xi |
| Introduction | | xiii |
| 1 | Scope | 1 |
| 2 | Normative references | 1 |
| 3 | Terms and definitions | 1 |
| 4 | Abbreviated terms | 6 |
| 5 | Conventions | 6 |
| 5.1 | General | 6 |
| 5.2 | Arithmetic operators | 6 |

| | | |
|---------------|---|-----------|
| 5.3 | Logical operators | 7 |
| 5.4 | Relational operators | 7 |
| 5.5 | Bit-wise operators | 7 |
| 5.6 | Assignment operators | 8 |
| 5.7 | Range notation | 8 |
| 5.8 | Mathematical functions | 8 |
| 5.9 | Order of operation precedence | 9 |
| 5.10 | Variables, syntax elements and tables | 9 |
| 5.11 | Text description of logical operators | 11 |
| 5.12 | Processes | 12 |
| 6 | Syntax and semantics | 12 |
| 6.1 | Method of specifying syntax in tabular form | 12 |
| 6.2 | Bit ordering | 13 |
| 6.3 | Specification of syntax functions and data types | 13 |
| 6.4 | Semantics | 15 |
| 7 | Data structures | 15 |
| 7.1 | General | 15 |
| 7.2 | Data unit | 15 |
| 7.3 | Raw reference | 16 |
| 7.3.1 | General | 16 |
| 7.3.2 | Syntax and semantics | 16 |
| 7.4 | Parameter set | 17 |
| 7.4.1 | Syntax and semantics | 17 |
| 7.4.2 | Encoding parameters | 18 |
| 7.5 | Access unit | 25 |
| 7.5.1 | Syntax and semantics | 25 |
| 7.5.2 | Access unit types | 29 |
| 8 | Descriptors | 30 |
| 9 | Sequencing reads | 33 |
| 9.1 | General | 33 |
| 9.2 | Supported symbols | 33 |
| 9.3 | Paired-end reads | 35 |
| 9.4 | Reverse-complement reads | 35 |
| 9.5 | Data classes | 35 |
| 9.6 | Aligned data | 36 |
| 9.7 | Unaligned data | 37 |
| 10 | Decoding process | 38 |
| 10.1 | General | 38 |
| 10.2 | dataset type = 0 or 1 | 38 |
| 10.2.1 | General | 38 |
| 10.2.2 | References padding | 39 |
| 10.2.3 | Type 1 AU (Class P) | 39 |
| 10.2.4 | Type 2 AU (Class N) | 40 |
| 10.2.5 | Type 3 AU (Class M) | 41 |
| 10.2.6 | Type 4 AU (Class I) | 41 |
| 10.2.7 | Type 5 AU (Class HM) | 43 |
| 10.2.8 | Type 6 AU (Class U) | 44 |
| 10.3 | dataset type = 2 | 45 |
| 10.3.1 | General | 45 |
| 10.3.2 | Type 1 AU | 45 |

| | | |
|----------------|---|------------|
| 10.3.3 | Type 2 AU | 47 |
| 10.3.4 | Type 3 AU | 47 |
| 10.3.5 | Type 4 AU | 47 |
| 10.3.6 | Type 6 AU | 47 |
| 10.4 | Genomic descriptors | 48 |
| 10.4.1 | General | 48 |
| 10.4.2 | pos | 48 |
| 10.4.3 | rcomp | 49 |
| 10.4.4 | flags | 50 |
| 10.4.5 | mmpos | 51 |
| 10.4.6 | mmtyp | 54 |
| 10.4.7 | clips | 58 |
| 10.4.8 | ureads | 61 |
| 10.4.9 | rln | 61 |
| 10.4.10 | pair | 63 |
| 10.4.11 | msscore | 71 |
| 10.4.12 | mmap | 72 |
| 10.4.13 | msar | 75 |
| 10.4.14 | rtp | 77 |
| 10.4.15 | rgroup | 78 |
| 10.4.16 | qv | 79 |
| 10.4.17 | rname | 84 |
| 10.4.18 | rftp | 84 |
| 10.4.19 | rftt | 85 |
| 10.4.20 | tokentyp | 86 |
| 10.5 | sequence | 94 |
| 10.5.1 | General | 94 |
| 10.5.2 | Aligned reads (Classes P, N, M, I, HM) | 95 |
| 10.5.3 | Unmapped reads (Class HM, U) | 96 |
| 10.6 | e-cigar | 96 |
| 10.6.1 | Syntax | 96 |
| 10.6.2 | Decoding process for the first alignment | 98 |
| 10.6.3 | Decoding process for other alignments | 106 |
| 10.6.4 | Reference transformation | 107 |
| 11 | Representation of reference sequences | 108 |
| 11.1 | External reference | 109 |
| 11.2 | Embedded reference | 109 |
| 11.3 | Computed reference | 109 |
| 11.3.1 | General | 109 |
| 11.3.2 | Supported Algorithms | 109 |
| 11.3.3 | Reference transformation | 110 |
| 11.3.4 | PushIn | 110 |
| 11.3.5 | Local assembly | 112 |
| 11.3.6 | Global assembly | 113 |
| 12 | Block payload parsing process | 114 |
| 12.1 | General | 114 |
| 12.2 | Encoding Mode 0 | 114 |
| 12.3 | Inverse binarizations | 115 |
| 12.3.1 | General | 115 |
| 12.3.2 | Binary (BI) | 115 |
| 12.3.3 | Truncated unary (TU) | 116 |
| 12.3.4 | Exponential golomb (EG) | 116 |

| | |
|--|-----|
| 12.3.5 Truncated exponential golomb (TEG) | 117 |
| 12.3.6 Signed truncated exponential golomb (STEG) | 117 |
| 12.3.7 Split unit-wise truncated unary (SUTU) | 117 |
| 12.3.8 Signed split unit-wise truncated unary (SSUTU) | 118 |
| 12.3.9 Double truncated unary (DTU) | 118 |
| 12.3.10 Signed double truncated unary (SDTU) | 119 |
| 12.4 Decoder configuration | 119 |
| 12.4.1 Sequences and quality values | 119 |
| 12.4.2 Support values | 120 |
| 12.4.3 CABAC binarizations | 121 |
| 12.4.4 Transformation parameters | 124 |
| 12.4.5 Msar descriptor and read identifiers | 126 |
| 12.4.6 State variables | 127 |
| 12.5 Initialization process for context variables | 131 |
| 12.6 Arithmetic decoding engine | 131 |
| 12.6.1 Initialization | 131 |
| 12.6.2 Arithmetic decoding process | 132 |
| 12.7 Decoding process for sequence descriptors | 139 |
| 12.7.1 General | 139 |
| 12.7.2 Block payload decoding process | 140 |
| 12.8 BSC decoding process | 156 |
| 12.8.1 decoding process | 156 |
| 13 Output format | 158 |
| 13.1 General | 158 |
| 13.2 MPEG-G record | 158 |
| 13.2.1 General | 158 |
| 13.2.2 number of template segments | 161 |
| 13.2.3 number of record segments | 161 |
| 13.2.4 number of alignments | 161 |
| 13.2.5 class ID | 161 |
| 13.2.6 read group len | 161 |
| 13.2.7 reserved | 161 |
| 13.2.8 read 1 first | 161 |
| 13.2.9 seq ID | 162 |
| 13.2.10 as depth | 162 |
| 13.2.11 read len | 162 |
| 13.2.12 qv depth | 162 |
| 13.2.13 read name len | 162 |
| 13.2.14 read name | 162 |
| 13.2.15 read group | 162 |
| 13.2.16 sequence | 162 |
| 13.2.17 quality values | 162 |
| 13.2.18 mapping pos | 162 |
| 13.2.19 ecigar len | 162 |
| 13.2.20 ecigar string | 163 |
| 13.2.21 reverse comp | 163 |
| 13.2.22 mapping score | 163 |
| 13.2.23 split alignment | 163 |
| 13.2.24 delta | 163 |
| 13.2.25 split pos | 163 |
| 13.2.26 split seq ID | 163 |
| 13.2.27 flags | 163 |

| | |
|---|-----|
| <u>13.2.28</u> <u>more alignments</u> | 164 |
| <u>13.2.29</u> <u>next pos</u> | 164 |
| <u>13.2.30</u> <u>next seq ID</u> | 164 |
| <u>13.3</u> <u>Initialization process</u> | 164 |
| <u>Annex A (informative)</u> <u>Tokenization of reads identifiers</u> | 167 |
| <u>Annex B (informative)</u> <u>Mapping quality</u> | 169 |
| <u>Annex C (informative)</u> <u>Inverse binarization examples</u> | 170 |
| <u>Annex D</u> <u>Block Sorting, Lossless Data Compression</u> | 175 |

iTeh Standards
(<https://standards.iteh.ai>)
Document Preview

[ISO/IEC FDIS 23092-2](https://standards.iteh.ai/catalog/standards/sist/8ca23faf-55be-4c1d-a61e-8027c513fe5d/iso-iec-fdis-23092-2)

<https://standards.iteh.ai/catalog/standards/sist/8ca23faf-55be-4c1d-a61e-8027c513fe5d/iso-iec-fdis-23092-2>

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

~~Attention is drawn~~ **ISO and IEC draw attention** to the possibility that ~~some of the elements~~ **implementation** of this document may ~~be involve~~ **the subject** use of (a) patent(s). ~~ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and <https://patents.iec.ch>. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see <https://patents.iec.ch>).~~

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This **third** edition cancels and replaces the second edition (ISO/IEC 23092-2:2020), which has been technically revised.

The main changes are as follows:

- ~~Inclusion~~ **inclusion** of new low-complexity entropy coders in subclause ~~7.4.2.2~~ (Table ~~9~~): LZMA, ZSTD, BSC;
- ~~Inclusion~~ **inclusion** of new indexed entropy coder in subclause ~~7.4.2.2~~ (Table ~~9~~): PROCURUSTES;
- ~~Inclusion~~ **inclusion** of the specification of BSC decoding process in subclause ~~12.8~~ and Annex ~~D~~;
- ~~Inclusion~~ **inclusion** of a new flag (`extended_alignment_info`) in subclause ~~13.2.1~~ to represent split alignment information in the compressed bitstream.

A list of all parts in the ISO/IEC 23092 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

iTeh Standards
(<https://standards.itih.ai>)
Document Preview

ISO/IEC FDIS 23092-2

<https://standards.itih.ai/catalog/standards/sist/8ca23faf-55be-4c1d-a61e-8027c513fe5d/iso-iec-fdis-23092-2>

Introduction

The advent of high-throughput sequencing (HTS) technologies has the potential to boost the adoption of genomic information in everyday practice, ranging from biological research to personalized genomic medicine in clinics. As a consequence, the volume of generated data has increased dramatically during the last few years, and an even more pronounced growth is expected in the near future.

At the moment genomic information is mostly exchanged through a variety of data formats, such as FASTA/FASTQ for unaligned sequencing reads and SAM/BAM/CRAM for aligned reads. With respect to such formats, the ISO/IEC 23092 series provides a new solution for the representation and compression of genome sequencing information by:

- Specifying an abstract representation of the sequencing data rather than a specific format with its direct implementation.
- Being designed at a time point when technologies and use cases are more mature. This permits addressing one limitation of the textual SAM format, for which the incremental ad-hoc addition of features followed along the years, resulting in an overall redundant and suboptimal format which was unnecessarily complicated.
- Separating free-field user-defined information with no clear semantics from the genomic data representation. This allows a fully interoperable and automatic exchange of information between different data producers.
- Allowing multiplexing of relevant metadata information with the data since data and metadata are partitioned at different conceptual levels.
- Following a strict and supervised development process which has proven successful in the last 30 years in the domain of digital media for the transport format, the file format, the compressed representation and the application program interfaces.

The ISO/IEC 23092 series provides the enabling technology that will allow the community to create an ecosystem of novel, interoperable, solutions in the field of genomic information processing. In particular it offers:

- Consistent, general and properly designed format definitions and data structures to store sequencing and alignment information. A robust framework which can be used as a foundation to implement different compression algorithms.
- Speed and flexibility in the selective access to coded data, by means of newly designed data clustering and optimized storage methodologies.
- Low latency in data transmission and consequent fast availability at remote locations, based on transmission protocols inspired by real-time application domains.
- Built-in privacy and protection of sensitive information, thanks to a flexible framework which allows customizable secured access at all layers of the data hierarchy.
- Reliability of the technology and interoperability among tools and systems, owing to the provision of a procedure to assess conformance to this document on an exhaustive dataset.
- Support to the implementation of a complete ecosystem of compliant devices and applications, through the availability of a normative reference implementation covering the totality of the ISO/IEC 23092 series.

The fundamental structure of the ISO/IEC 23092 series data representation is the *genomic record*. The genomic record is a data structure consisting of either a single sequencing read, or a paired sequencing read, and its associated sequencing and alignment information; it may contain detailed mapping and alignment data, a single or paired read identifier (read name) and quality values.

Without breaking traditional approaches, the genomic record introduced in the ISO/IEC 23092 series provides a more compact, simpler and manageable data structure grouping all the information related to a single DNA template, from simple sequencing data to sophisticated alignment information.

The genomic record, although it is an appropriate logic data structure for interaction and manipulation of coded information, is not a suitable atomic data structure for compression. To achieve high compression ratios, it is necessary to group genomic records into clusters and to transform the information of the same type into sets of descriptors structured into homogeneous blocks. Furthermore, when dealing with selective data access, the genomic record unit is too small to allow effective and fast information retrieval.

For these reasons, this document introduces the concept of access unit, which is the fundamental structure for coding and access to information in the compressed domain.

The access unit is the smallest data structure that can be decoded by a decoder compliant with [ISO/IEC 23092-2:20202023](#). An access unit is composed of one block for each descriptor used to represent the information of its genomic records; therefore, a block payload is the coded representation of all the data of the same type (i.e. a descriptor) in a cluster.

In addition to clusters of genomic records compressed into access units, reads are further classified in six data classes: five classes are defined according to the result of their alignment against one or more reference sequences; the sixth class contains either reads that could not be mapped or raw sequencing data. The classification of sequencing reads into classes enables the development of powerful selective data access. In fact access units inherit a specific data characterization (e.g. perfect matches in class P, substitutions in class M, indels in class I, half-mapped reads in class HM) from the genomic records composing them, and thus constitute a data structure capable of providing powerful filtering capability for the efficient support of many different use cases.

Access units are the fundamental, finest grain data structure in terms of content protection and in terms of metadata association. In other words, each access unit can be individually and independently protected. Figure 1 shows how access units, blocks and genomic records relate to each other in the ISO/IEC 23092 series data structure.

