|  |  |
| --- | --- |
| **ITU – Telecommunications Standardization Sector**STUDY GROUP 21 Question C/16 (ex Q6/16)**Video Coding Experts Group (VCEG)**75th Meeting: 2-8 November 2024, Kemer, Türkiye | Document VCEG-BW03-v1 |

|  |  |
| --- | --- |
| Question: | QC/16 SG21 (VCEG) |
| Source: | **Christof Fersch, Simon Gauntlett, Kristofer Kjörling, Janusz Klejsa, Brian Lee, Heidi-Maria Lehtonen, Harald Mundt, Jonas Samuelsson, Michael Schug, Panji Setiawan, Gary Sullivan, Lars Villemoes, Mark Vinton (Dolby Laboratories)** | Email: | christof.fersch@dolby.comSimon.Gauntlett@dolby.com  |
| Title: | **Description of Dolby’s response to the Call for Proposals on the compression of biomedical waveform data** |
| Purpose: | Proposal |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Abstract

This document contains the Dolby Laboratories response to the CfP [1] on the compression of biomedical waveform data, providing the algorithm description, tests results and other required information according to the CfP, as well as supplementary material.

# Introduction

The Dolby proposal consists of a core codec compressing the raw waveforms, and a flexible and extensible bitstream format representing systemic features.

The core coder is conceptually simple with a single architecture supporting both lossy and lossless operation points. It supports full random access, with configurable latency. To enable lossless operation in the DCT domain, an integer invertible DCT is introduced using a specific implementation that ensures that the number of lifting steps per sample has a constant upper bound independent of the transform size.

The proposed bitstream format provides a set of systemic features beyond pure compression performance that are considered important for the success of the format. These include authentication of signals, metadata that allow temporal alignment of different biomedical and related signals, tagging of certain information, such as distinctive features available in the coded signal, and the ability to signal distortion measures for evaluating the impact of lossy coding from the decoded stream.

In addition to the codec and system description, this document presents the high-level syntax and error measures; percentage root mean square distortion (PRD), channel-normalized percentage root mean square distortion (CPRD), and peak signal-to-noise ratio (PSNR) values are reported along with the number of bits per sample (BPS) for the three categories listed in the CfP. Performance on filtered Electromyography (EMG) signals and an additional signal category, Photoplethysmogram (PPG) are presented in supplementary material. All error measures are calculated on the raw payload.

In summary the proposal provides:

* A flexible and extensible bitstream format enabling important systemic features
* A codec for compressing the biomedical signals providing a raw compressed representation
* A conceptually simple system with a single architecture supporting both lossy and lossless operation points
* Support for full random access, with configurable latency
* An integer invertible DCT using a specific implementation that ensures that the number of lifting steps per sample has a constant upper bound independent of the transform size.

The system, with its unified architecture and conceptual simplicity, is designed to provide a clean and very approachable starting point for further collaborative standards work.

# Codec description

This Section presents a technical description of the proposed core technology. First, the overall architecture of the codec is presented, followed by a more detailed description of the transform, prediction, and entropy coding.

## Codec architecture

The high-level block diagrams of the encoder and decoder of the proposed system are presented in Figure 1 and Figure 2, respectively. The codec is based on an integer DCT in combination with linear prediction and quantization in the transform domain, and a straightforward entropy codec based on combination of multi-dimensional Huffman and Golomb-Rice coding. It is a single harmonized and conceptually simple structure that supports both lossless and lossy coding, where for the lossless operation both DCT-domain and time-domain coding can be applied. The algorithm supports full random access, i.e. there are no dependencies on previous frames, and the latency of the system is configurable by selection of the frame-size. The codec first attempts to do lossless coding with the given bitrate budget, and if this is not possible, it will fall back to lossy coding.



Figure 1. Encoder of the proposed system. (Quantization is only applied for lossy coding, and in the lossless case, skipping this is equivalent to using a quantization step size of 1.)



Figure 2. Decoder of the proposed system. (Inverse quantization is only applied for lossy coding, and in the lossless case, skipping this is equivalent to using an inverse quantization step size of 1.)

## Transform

The DCT in Figure 1 is an integerized DCT of type II and the IDCT in Figure 2 is its inverse. Nonoverlapping frames of the input signal channels are processed at stride $L$, which can be chosen as any power of two from 32 to 2048 samples. The normalization of the DCT is chosen to achieve an orthogonal transform of its input frame of samples $x\left(1\right), x\left(2\right),… ,x(L-1) $to frequency bins $X\left(0\right),X\left(1\right),… ,X(L-1) $ according to

$$\begin{array}{c}X(j)=\left\{\begin{matrix}\sqrt{\frac{2}{L}}\sum\_{l=0}^{L-1}x(l)\cos(\left(\frac{πj\left(2l+1\right)}{2L}\right))&j=1,2…, L-1; \\\frac{1}{\sqrt{L}}\sum\_{l=0}^{L-1}x(l)&j=0.\end{matrix}\right\} .\#(1) \#\end{array}$$

An integer invertible DCT of size $L$ is obtained from an integer invertible DFT of size $L/2$ by adapting the rotation and twiddling steps described in [3] to orthogonality and subsequently factorizing those to integer invertible lifting steps. The integer invertible DFT is derived by a generalization of the multidimensional lifting method of [4]. More precisely, a dual input integer invertible DFT is implemented by the flow diagram depicted in Figure 3. For a single input, a reduction to the dual input case with half the block size is obtained by splitting the input into even and odd samples and factorizing the Cooley-Tukey FFT butterfly into integer invertible lifting steps. With this implementation, the number of lifting steps per sample has a constant upper bound independent of the transform size $L$. Since each lifting step is followed by rounding, this leads to a good approximation of the target DCT of Equation (1), even for large $L$.



Figure 3. Calculation flow diagram for dual integer invertible DFTs with lifting steps.

## Prediction and quantization

Prediction is applied in the DCT domain in two ways: prediction from reconstructed samples in the current channel and frame (intra-channel prediction) and/or from reconstructed samples of coded channels of the same frame (inter-channel prediction). There is no (temporal) inter-frame dependency for prediction. Two flags per frame and channel are transmitted in the bitstream to signal which type of prediction is active. Prediction and adaptation of the prediction coefficients run from high to low frequencies. Figure 4 and Figure 5 provide block diagrams describing the encoder-side prediction in the DCT domain and the decoder-side processing, respectively.



Figure 4. Block diagram presenting encoder-side intra-channel and inter-channel prediction in DCT domain. The dashed line corresponds to sending the updated prediction coefficients to the predictor. The channel selector selects *V* most recent coded channels.



Figure 5. Block diagram presenting decoder-side intra-channel and inter-channel prediction in DCT domain. The dashed line corresponds to sending the updated prediction coefficients to the predictor. The channel selector selects *V* most recent coded channels.

Given a reconstructed spectral sample $\tilde{X}\left(i,j\right) $for channel *i* and frequency bin *j*, the prediction process starts at *j = L −* 1 where *L* is the transform length and proceeds until *j = 0* (the lowest frequency bin). For channel *i* the prediction $P\_{i}\left(j\right) $of sample $X\left(i,j\right)$ is computed as

$$\begin{array}{c}P\_{i}\left(j\right)= \sum\_{u=0}^{U-1}a\_{i}\left(j,u\right)\tilde{X}\left(i,j+1+u\right)+\sum\_{v=0}^{V-1}b\_{i}\left(j,v\right)\tilde{X}\left(i-1-v,j\right), \#\left(2\right)\end{array}$$

where *U* is the prediction order, *V* is the number of prediction channels and $\tilde{X}\left(i,j\right)$ is initialized as

$$\begin{array}{c}\tilde{X}\left(i,j\right)=0 for i<0 or j>L-1 ,\#\left(3\right)\end{array}$$

and the prediction coefficients are initialized as

$$\begin{array}{c}\begin{matrix}a\_{i}\left(L-1,u\right)=0,&u=0,1,…,U-1;\\b\_{i}\left(L-1,v\right)=0,&v=0,1,…,V-1.\end{matrix} \#\left(4\right)\end{array}$$

Prediction order *U* is dependent on the transform length *L* as described in Table 1. All channels are coded in a successive manner such that for the first coded channel there is no prediction from other channels, and for the *i*th channel, $V=min\left\{i,32\right\}$.

Table 1. Prediction order *U* corresponding to the supported transform lengths *L*.

|  |  |
| --- | --- |
| *L* | *U* |
| 2048 | 40 |
| 1024 | 20 |
| 512 | 10 |
| 256 | 5 |
| 128 | 4 |
| 64 | 4 |
| 32 | 4 |

In the encoder, the prediction residual is first computed as the difference between the signal and the prediction, as shown in Eq. (5).

$$\begin{array}{c}D\_{i}\left(j\right)= X\left(i,j\right)-P\_{i}\left(j\right). \#\left(5\right)\end{array}$$

The prediction residual is then quantized with a dead-zone scalar quantizer using a quantization scale factor as shown in Eq. (6).

$$\begin{array}{c}D\_{i}^{Q}\left(j\right)= sign\left(D\_{i}\left(j\right)\right)∙\left⌊2^{\frac{-GlobalGain}{32}}∙abs\left(D\_{i}\left(j\right)\right)+α\right⌋. \#\left(6\right)\end{array}$$

Where, $GlobalGain$ is a quantization control parameter that ranges between 0 and 1023 and is transmitted to the decoder. The choice of the value of α is an encoder-only matter that does not affect interoperability, and ordinarily, *α* would be in the range of 0.0 to 0.5; as tested, *α* was set to 0.4054.

Samples are reconstructed from the residual using uniform reconstruction quantization (URQ), and the reconstruction is added to the prediction as

$$\begin{array}{c}\tilde{D}\_{i}(j)= 2^{\frac{GlobalGain}{32}}∙D\_{i}^{Q}\left(j\right). \#\left(7\right)\end{array}$$

$$\begin{array}{c}\tilde{X}\left(i,j\right)= \tilde{D}\_{i}(j)+P\_{i}\left(j\right). \#\left(8\right)\end{array}$$

After each prediction and reconstruction step, prediction coefficients $a\_{i}$ and $b\_{i}$ are updated as

$$\begin{array}{c} \begin{matrix}a\_{i}\left(j-1, u\right)= a\_{i}\left(j,u\right)+\tilde{D}\_{i}(j) g\_{i}\left(j\right) \tilde{X}\left(i,j+1+u\right),&u=0,1,..,U-1;\\b\_{i}\left(j-1, v\right)= b\_{i}\left(j,v\right)+\tilde{D}\_{i}(j) g\_{i}\left(j\right) \tilde{X}\left(i-1-v,n\right),&v=0,1,…,V-1,\end{matrix} \#\left(9\right)\end{array}$$

where the update step gain $g\_{i}(j)$ is computed as

$$\begin{array}{c}g\_{i}\left(j\right)=0.25\frac{1}{X\_{i,norm}\left(j\right)max\left\{X\_{i,norm}\left(j\right),\left|\tilde{D}\_{i}(j)\right|\right\}}, \#\left(10\right)\end{array}$$

and the input vector norm $X\_{i,norm}(j)$ is computed as

$$\begin{array}{c}X\_{i,norm}\left(j\right)= \sqrt{X\_{i,energy}\left(j\right) }. \#\left(11\right)\end{array}$$

Here, the predictor input total energy is computed as

$$\begin{array}{c}X\_{i,energy}\left(j\right)=\sum\_{u=0}^{U-1}\left(\tilde{X}(i,j+1+u)\right)^{2}+\sum\_{v=0}^{V-1}\left(\tilde{X}\left(i-1-v,j\right)\right)^{2}. \#\left(12\right)\end{array}$$

In a fixed-point implementation, the term $1/X\_{i,norm}(j)$ is efficiently approximated by a right-bit-shift of $ceil\left\{log\_{2}X\_{i,energy}\left(j\right)\right\}+ 1 $bits.

## Entropy coding

The entropy coding of the residual signal is performed using a simple combination of multi-dimension Huffman coding and Golomb-Rice coding. The signal is first divided into sub-regions, where the length of each sub-region is dependent on the frame length as shown in Table 2. The sub-regions are of uniform length except the last sub-region that is twice the previous sub-region lengths. In each sub-region the signal is encoded with one of 22 possible codebooks, that are either Huffman or Golomb-Rice codes. The 22 possible codebooks that can be selected for each sub-region are specified in Table 3. The codebooks are selected in each sub-region such that the codebook introduces no distortion (the maximum absolute value in a sub-region must be less than the largest absolute value (LAV) specified in Table 3), and the overall bitrate is minimized including the cost to transmit the codebook selection in each sub-region. The codebook index for the codebook selected in the 0th sub-region is transmitted as a 5bit number, while the codebook index for the subsequent sub-regions is transmitted as a Huffman encoded difference relative to the previous sub-region.

A Viterbi algorithm is employed to select the best codebook for each sub-region such that the overall bitrate is minimized.

Table 2. Sub-region lengths for each available frame length

|  |  |  |
| --- | --- | --- |
| Frame length  | Sub-region lengths | Sub-region count |
| 32 | [4,4,…,4,8] | 7 |
| 64 | [4,4,…,4,8] | 15 |
| 128 | [4,4,…,4,8] | 31 |
| 256 | [4,4,…,4,8] | 63 |
| 512 | [8,8,…,8,16] | 63 |
| 1024 | [16,16,…,16,32] | 63 |
| 2048 | [32,32,….,32,64] | 63 |

Table 3. Codebook type, largest absolute value (LAV), signed/unsigned codebook, and codebook size for each codebook index

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Codebook index** | **Codebook type** | **LAV** | **Dimension** | **Signed** | **Codebook size** |
| 0 | N/A | 0 | N/A | N/A | 0 |
| 1 | Huffman | 1 | 4 | Yes | 81 |
| 2 | Huffman | 1 | 4 | Yes | 81 |
| 3 | Huffman | 2 | 4 | No | 81 |
| 4 | Huffman | 2 | 4 | No | 81 |
| 5 | Huffman | 4 | 2 | Yes | 81 |
| 6 | Huffman | 4 | 2 | Yes | 81 |
| 7 | Huffman | 7 | 2 | No | 64 |
| 8 | Huffman | 7 | 2 | No | 64 |
| 9 | Huffman | 12 | 2 | No | 169 |
| 10 | Huffman | 12 | 2 | No | 169 |
| 11 | GR (2) | Inf | 1 | No | N/A |
| 12 | GR (3) | Inf | 1 | No | N/A |
| 13 | GR (4) | Inf | 1 | No | N/A |
| 14 | GR (5) | Inf | 1 | No | N/A |
| 15 | GR (6) | Inf | 1 | No | N/A |
| 16 | GR (7) | Inf | 1 | No | N/A |
| 17 | GR (8) | Inf | 1 | No | N/A |
| 18 | GR (9) | Inf | 1 | No | N/A |
| 19 | GR (10) | Inf | 1 | No | N/A |
| 20 | GR (11) | Inf | 1 | No | N/A |
| 21 | GR (12) | Inf | 1 | No | N/A |
| 22 | GR (13) | Inf | 1 | No | N/A |

## Lossless coding

Lossless encoding and decoding follow the same principles as the lossy encoding and decoding, although in this case the transform must be integer invertible and the DCT-domain LMS predictor must also be integer invertible. Furthermore, the encoder considers both DCT-domain and time-domain prediction for the frame that is being coded and chooses the operation mode that results in a lower bitrate for that frame. The integer invertible transform is described in Section 2.2. While the modifications to the LMS predictor equations are detailed in Section 2.5.1.

### Integer invertible DCT-domain LMS predictor

Both the lossy and lossless encoding and decoding can share the same DCT-domain LMS predictor (as described in Figure 4), however, for lossless coding the quantization of the prediction must be explicit to ensure integer invertible behavior. Furthermore, the update step for the LMS predictor is explicitly fixed point.

The modification to prediction calculation is provided in the following equation.

$$\begin{array}{c}P\_{i}\left(j\right)= \left(\sum\_{u=0}^{U-1}a\_{i}\left(j,u\right)X\left(i,j+1+u\right)+\sum\_{v=0}^{V-1}b\_{i}\left(j,v\right)X\left(i-1-v,j\right)+W\_{offset}\right)\gg W ,\#\left(13\right)\end{array}$$

where$ W\_{offset}$ is 1 << 14 and $W$ is 15.

As for the lossy case, the signal value for the *i*th channel and *j*th sample can be reconstructed according to Eq. (5) from the residual $D\_{i}\left(j\right)$ transmitted in the bitstream to the decoder and the prediction $P\_{i}(j)$. The signal energy is also computed according to Eq. (12).

The integer approximation of $log\_{2}$ of the square root of the signal energy is then computed as shown in the following equation,

$$\begin{array}{c}L\_{i,norm}(j)=ceil(log\_{2}(X\_{i,energy}\left(j\right)))\gg 1, \#\left(14\right)\end{array}$$

where$ "ceil"$ means rounding up to nearest integer.

The update gain is then computed using the following equation.

$$\begin{array}{c}g\_{i}\left(j\right)=\left(\left(D\_{i}\left(j\right)\ll W\right)\gg L\_{i,norm}\left(j\right)+2\right)\gg 2 . \#\left(15\right)\end{array}$$

The value of $g\_{i}\left(j\right)$ is then limited clipped to the range shown in the following equation.

$$\begin{array}{c}-\left(1\ll \left(W-2\right)\right)\leq g\_{i}\left(j\right)\leq \left(1\ll \left(W-2\right)\right). \#\left(16\right)\end{array}$$

Finally, the predictors are updated as shown in the following equations.

$$\begin{array}{c}\begin{matrix}a\_{i}\left(j-1, u\right)= a\_{i}\left(j,u\right)+\left(g\_{i}\left(j\right) X\left(i,j+1+u\right)+L\_{offset}\right)\gg L\_{i,norm}\left(j\right);\\b\_{i}\left(j-1, v\right)= b\_{i}\left(j,v\right)+ \left(g\_{i}\left(j\right) X\left(i-1-v,n\right)+L\_{offset}\right)\gg L\_{i,norm}\left(j\right), \end{matrix}\#\left(17\right)\end{array}$$

where $L\_{offset}=\left(1\ll \left(L\_{i,norm}\left(j\right)-1\right)\right).$

### Lossless time-domain linear predictive coding and forward-adaptive inter-channel prediction

The lossless encoder and decoder can switch on a frame-by-frame basis from the DCT domain to the time domain. As such it supports time-domain linear predictive coding (LPC), as well as inter-channel prediction in the time domain. Similarly to the DCT-domain operation, there is no temporal inter-frame dependency for prediction.

The visualizations in Figure 6 and Figure 7 illustrate the operation of the predictors in the time domain, for the encoder and decoder.



Figure 6. LPC encoder

 

Figure 7. LPC decoder

If time-domain LPC is enabled, the encoder calculates prediction coefficients and transmits the coefficients to the decoder. The time-domain LPC mode allows the update of the prediction coefficients at 32 sample intervals within a frame. The LPC coefficients are transmitted as reflection coefficients [5], which ensures the prediction filter is stable even with quantization. The lossless encoder and decoder support LPC filter orders ranging from 0th order (no LPC) to 31st order. The LPC filter order is transmitted in the bitstream to the decoder.

The following equation shows the reconstruction of the signal $x(i,j)$ from the residual signal $d\_{i}(j)$ transmitted to the decoder.

$$\begin{array}{c}x\left(i,j\right)=d\_{i}\left(j\right)-\left(\sum\_{k=1}^{K}a\left(k\right)x\left(i,j-k\right)+W\_{offset}\right)\gg W ,\#\left(18\right)\end{array}$$

where $x\left(i,j\right)$ is the signal for the *i*th channel and *j*th sample in a frame,

 $d\_{i}\left(j\right)$ is the residual signal transmitted to the decoder for the *i*th channel and *j*th sample in a frame,

 $a(k)$ are the direct-form LPC coefficients,

$K$ is the LPC order, and

 $W\_{offset}$ and $W$ are (1<<22) and 23 respectively for LPC coding.

the LPC coefficients are transmitted as reflection coefficients, the direct-form LPC coefficients $a\_{k}$ must be calculated at the decoder. The conversion of the reflection coefficients is recursive and shown in the following equation.

$$\begin{array}{c}a\_{m}\left(k\right)=a\_{m-1}\left(k\right)+\left(r\left(k\right)\*a\_{m-1}\left(m-k\right)+ W\_{offset}\right)\gg W, k=1,2,..m, \#\left(19\right)\end{array}$$

where the recursion index $m=1,2,…K$,

 $a\_{m}\left(k\right)$ is the direct-form LPC coefficients for the *m*th recursion, and

 $r\left(k\right)$ is the reflection coefficients transmitted to the decoder.

The LPC filter buffer is reset at frame boundaries, however, if the LPC coefficients are updated within a frame then the filter buffer is not reset, and the filter coefficients are swapped at the update boundary.

If inter-channel prediction in the time-domain is enabled, the encoder transmits a reference channel and a prediction gain, for a given target channel. Computation of the original signal $x\left(i,j\right)$ from a residual signal $d\_{i}\left(j\right)$ is given by the following equation.

$$\begin{array}{c}x\left(i,j\right)=d\_{i}\left(j\right)+\left(P\left(i\right)\*x\left(Ref\_{i},j\right)+8\right)\gg 4, \#\left(20\right)\end{array}$$

where

* $x\left(i,j\right)$ is the signal for the *i*th channel and *j*th sample in a frame,
* $d\_{i}\left(j\right)$ is the residual signal transmitted to the decoder for the *i*th channel and *j*th sample in a frame,
* $P\left(i\right)$ is the inter-channel prediction coefficient for the *i*th channel transmitted to the decoder,

$Ref$ is the reference channel used to predict the *i*th channel transmitted to the decoder.

# Systemic features

This Section includes a description of the proposed bitstream structure and features which are enabled by proposed syntactical elements.

## Flexible and future-proof bitstream structure

The field of capturing, coding, processing, and analyzing medical bitstreams is very wide and complex. With this standardization being in its early phase, we anticipate receiving feature requests that we cannot even imagine yet. Therefore, we believe it is paramount to base this effort on a flexible and extensible bitstream structure that allows future additions without breaking backward compatibility to existing functionality. We therefore propose a packetized bitstream format based on ISO/IEC 23008-3 (MPEG-H 3D audio), which has already proven to be very flexible and extensible, while enabling simple and straightforward handling of such bitstreams. Figure 8 shows the high-level structure of the proposed bitstream.



Figure 8. Basic bitstream structure using packets.

Every packet in such a stream (msStreamPacket) consists of the elements illustrated in Table 4, where the mnemonic “ev” indicates use of the escapedValue() encoding scheme (see details in the accompanying syntax description document).

Table 4. Syntax of msStreamPacket()

| **Syntax** | **No. of bits** | **Mnemonic** |
| --- | --- | --- |
| msStreamPacket () { |  |  |
|  msStreamPacketType = escapedValue(3,8,8); | **3,11,19** | **ev** |
|  msStreamPacketLabel = escapedValue(2,8,32); | **2,10,42** | **ev** |
|  msStreamPacketLength= escapedValue(11,24,24); | **11,35,59** | **ev** |
|  msStreamPacketPayload(msStreamPacketType); |  |  |
| } |  |  |
| NOTE With the given bit allocation, msStreamPacketPayload() is always byte-aligned. |

## Authentication of signals

We believe that – as with multimedia content – the trustworthy usability of biomedical signals is increasingly important in a world where sophisticated forgeries and manipulations are relatively easily achievable using AI-based methods. Therefore, it is going to be essential that trustworthy verification can be enabled for biomedical signals. We therefore propose to add the possibility to carry authentication information in a standardized and extensible way as part of the standardization effort for H.BWC. Care needs to be taken in a way that the solutions are tailored to match specific characteristics of the related biomedical signals. For instance, it is important that the inclusion of authentication information in a biomedical bitstream can be done in a way that there is no significant increase in bitrate (average and peak). In addition, we consider it to be important that the related syntax is flexible enough that bitstreams can be changed, e.g. for annotations, without breaking authentication mechanisms, and can also carry information which enables relating different biomedical signals to each other, e.g. from the same user.

We believe we are addressing those requirements with the proposed syntax.

Related packet types:

* MS\_AUTH\_START
* MS\_AUTH\_SIG
* MS\_TIMESTAMP
* MS\_UUID\_S
* MS\_UUID\_U

Please find more details in the accompanying syntax description document.

## Distortion measure metadata

As opposed to multimedia signals, biomedical signals are seldomly captured to be played out to human observers. While in some cases, these signals may be directly inspected (e.g., by a clinician), the majority of use cases comprises automated analysis, or using the captured signals for training of automated analysis tools. For instance, these signals are often used to derive some performance parameters or estimate other signals of interest. For example, a photoplethysmogram (PPG) may be captured, encoded, decoded, and reconstructed to perform estimation of heart rate. According to another example, the PPG signal may be used to derive a heart rate variability (HRV) signal, which can be then used to estimate instantaneous HRV.

The captured signals may be encoded, and the bitstreams may be stored in medical records. It may happen, that the analysis task to be performed on the decoder side remains unknown on the encoder side. This typically would not be problematic if lossless coding was used since the source codec would not introduce any coding distortion. However, if lossy coding is used, the distortion which was introduced during coding may become problematic, especially without knowledge about the distortion on the decoder/analysis side. For example, some medical signals encoded in a lossy manner may become undesired for including them in analysis performed on the decoder side due to their high coding distortion. According to another example, some medical signals coded in a lossy manner may need to be excluded from a signal dataset used for training of AI-based analysis methods.

The coding distortion would typically not only depend on the bitrate but also on a particular realization of the signal that was encoded. This dependence is generally unknown on the decoder side. If lossy coding was used, it may happen that the coding distortion was prohibitive to accomplish some decoder-side task (e.g., a training of a neural network based on the reconstructed signal), but it typically cannot be known on the decoder side just by inspecting the decoded bitstreams alone. Therefore, since the performance of coding depends not only on the operating bitrate, but also on a realization of a signal, it could be beneficial to include in the bitstream some side information characterizing the performance of encoder on this realization of the signal.

Related packet types:

* MS\_CFG
* MS\_FRAME

Please find more details in the accompanying syntax description document.

## Metadata for annotating features of the coded signal

In order to enable easy parsing of biomedical bitstreams without decoding the whole bitstream, we suggest introducing metadata which allows for “tagging” of certain information, such as distinctive features available in the coded signal. One use-case could be that a deep neural network is used to annotate a signal automatically, preparing for review of an expert/doctor afterwards, who can focus only on the areas of the signal which have been tagged by the deep neural network. Another use-case could be that one expert annotates the coded signal (e.g. using EEGNet) for an additional analysis by another expert for further review, while enabling easy search for certain features.

Related packet types:

* MS\_CFG
* MS\_FRAME

If, however, a dedicated packet is preferred to identify a feature, a new packet type is certainly possible to be introduced, e.g.,

* MS\_FEATURE

Please find more details in the accompanying syntax description document.

## Metadata for temporal alignment of signals

To enable temporal alignment of different utility signals captured by different sensors, we suggest introducing metadata which allow temporal alignment of different biomedical and related signals.

Amongst many others, one use-case could be that a deep neural network may be trained not only on one signal captured by one sensor, but on two or more signals captured by two or more different sensors (e.g. EEG and ECG). In such scenarios, it would be essential to have the two or more signals temporally aligned, in order to enable proper correlation of those signals. In addition, in a clinical context, it may be very beneficial to enable inspection of certain health issues in a way that one can easily check if the effects that are visible in one type of a signal (e.g. EEG) have a correlated effect in another type of signal (e.g. PPG).

In order support such use-cases, the related syntax of a biomedical bitstream has to support ways to properly annotate the bitstream using accurate timing and to enable identification of both the user (in an anonymous way) and the used sensor.

Related packet types:

* MS\_TIMESTAMP
* MS\_UUID\_S
* MS\_UUID\_U

Please find more details in the accompanying syntax description document.

# Error measurement results

## Introduction and definitions

This Section describes the objective test results of our proposal. Sections 4.3 - 4.5 present the average PRD, CPRD, and PSNR results as a function of BPS. PRD and CPRD values for each bitstream along with the number of BPS are provided in the supplementary material associated with this document. Furthermore, in the supplementary material, three graphs per input sequence to illustrate the behavior of the PRD, CPRD, and PSNR metrics over the BPS values corresponding to the eight working points WP1 to WP8 are provided.

As per the CfP document [1], the PRD, CPRD, and PSNR error metrics are defined as follows. Let $x\_{i,j}$ be the *j*-th sample (with $0\leq j<M$) of channel *i* (with $0\leq i<N$) and $\tilde{x}\_{i,j}$ the corresponding reconstructed sample after decoding a bitstream. The *PRD* metric is defined as follows:

$$PRD=100\%∙\sqrt{\frac{\sum\_{i=0}^{N-1}\sum\_{j=0}^{M-1}\left(x\_{i,j}-\tilde{x}\_{i,j}\right)^{2}}{\sum\_{i=0}^{N-1}\sum\_{j=0}^{M-1}\left(x\_{i,j}-m\_{i}\right)^{2}}}, (21)$$

where

$$m\_{i}=\frac{1}{M}∙\sum\_{j=0}^{M-1}x\_{i,j} (22)$$

is the mean value of the input data for channel *i*. The CPRD metric is defined as follows:

$CPRD=\frac{100\%}{N}∙\sum\_{i=0}^{N-1}\sqrt{\frac{\sum\_{j=0}^{M-1}\left(x\_{i,j}-\tilde{x}\_{i,j}\right)^{2}}{\sum\_{j=0}^{M-1}\left(x\_{i,j}-m\_{i}\right)^{2}}}, (23)$

where all indices *i* for which

$$\sum\_{j=0}^{M-1}\left(x\_{i,j}-m\_{i}\right)^{2}=0 (24)$$

are to be excluded. The PSNR metric is defined as follows:

$$PSNR\_{avg}=\frac{1}{N}\sum\_{i=0}^{N-1}PSNR\_{i}, (25)$$

where

$$PSNR\_{i}=-10\*log\_{10}\left(\frac{MSE\_{i}}{\left(2^{B}-1\right)^{2}}\right), (26)$$

and

$MSE\_{i}=\frac{1}{M}\sum\_{j=0}^{M-1}\left(x\_{i,j}-\tilde{x}\_{i,j}\right)^{2}, B=16. (27)$

Some of the items in the EEG test set contain channels with input data all zeros, leading to PSNR = Inf for those channels. These channels are excluded from the PSNR measurement.

For some items in the datasets, it can happen that the codec can do lossless coding at some of the working points corresponding to lower bitrates, which will lead to averaged PSNR = Inf. In this case, to be able to present the averaged PSRN over all items for working points WP1-WP8, PSNR for the losslessly coded item has been set to $-10log\_{10}(eps)$ = 156.54 dB, where eps = 2.2204e−16.

Finally, bits per sample (BPS) is defined as follows:

$BPS=\frac{number of bits in the bitstream}{N∙M}. (28) $

## Overview of data – discussion and observations

In the following Sections the results as requested by the CfP are given in graphs and tables. In addition, supplementary information is provided in supplementary material for an additional signal type, Photoplethysmogram (PPG), as well as for filtered EMG data as provided in combination of the raw EMG data. The supplementary information is provided to show the applicability of the proposed system for signal categories and variations outside of the strict limits of the CfP.

The EEG, ECG, and EMG datasets are agreed upon with VCEG and they are downloaded from the ftp server indicated in the CfP. The PPG dataset is publicly available at Physionet, <https://archive.physionet.org/physiobank/database/bidmc/>. License information is provided in the corresponding folder containing the bitstreams provided as part of the submission.

The following high-level observations can be made from the data provided in the subsequent Section, in response to the CfP:

* For EEG signals, lossless performance is obtained at 5.0106 BPS, for the 0.75 bits per sample operation point the codec provides a PRD of 10.563, a CPRD of 12.4039, and a PSNR of 76.1291 dB.
* For ECG signals, lossless performance is obtained at 3.7758 BPS, for the 0.75 bits per sample operation point the codec provides a PRD of 4.1074, a CPRD of 4.5656, and a PSNR of 87.9996 dB.
* For EMG signals, lossless performance is obtained at 6.5819 BPS, for the 0.75 bits per sample operation point the codec provides a PRD of 1.8359, a CPRD of 9.1897, and a PSNR of 66.953 dB.

When observing the data including the data in the supplementary information, we make observations pertaining to the metrics stipulated by the CfP.

The PRD metric seems to be a metric that is very sensitive to aspects that may not be relevant from the perspective of drawing conclusions from the waveforms, and a lower PRD may not necessarily imply a more accurate diagnosis from observing the waveforms, than a higher PRD [[6](https://pmc.ncbi.nlm.nih.gov/articles/PMC6077674/)].

Furthermore, the PRD seems very sensitive to noise and drift. Specifically, as can be seen in Eq. (21), the PRD metric is a function of the signal variance, which heavily affects the result for the EMG signals for filtered vs. the raw signals. An error metric that is a function of the signal variance, may be applicable to one signal category, but not necessarily applicable to another. In the specific example of the present raw and filtered EMG signals, we observe a higher PRD for the filtered signals (due to the lower variance of the input), but a lower variance of the error compared to the raw data.

We believe it to be advisable to exercise caution when drawing conclusions on the systems overall performance and applicability to the tasks based on solely PRD data. We believe that different use-cases and signal categories may benefit from different error metrics, that a system needs to cater for distortion constrained operations, with information provided at decode under which constraints the signal was compressed. We believe that the performance data on the table should be observed in that larger context.

## Electroencephalography (EEG) signals

Average results for the EEG dataset are provided in Figure 9 and Table 5. In Figure 9, PSNR is only shown WP1-WP8. For WP0, PSNR is always Inf, and hence left out from the averaged results. The per-item results, including three graphs providing the PRD, CPRD, and PSNR metrics as a function of the BPS values for the working points WP1-WP8, and tables that list PRD, CPRD, and PSNR values for each bitstream, are provided in Appendices (see separate files).



Figure 9. Average PRD, CPRD, and PSNR values over bits per sample (BPS) averaged per working point for the EEG dataset. Vertical grid lines correspond to the working points WP1-WP8 as defined in [1].

Table 5. Average BPS, PRD, CPRD, and PSNR values for the EEG dataset.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Working point | BPS | PRD | CPRD | PSNR |
| WP0 | 5.0106 | 0 | 0 | Inf |
| WP1 | 3.9027 | 0.5863 | 0.7283 | 107.1313 |
| WP2 | 3.4639 | 0.8743 | 1.0767 | 101.2505 |
| WP3 | 2.9836 | 1.3541 | 1.6439 | 94.0129 |
| WP4 | 2.4874 | 1.97 | 2.3804 | 90.7854 |
| WP5 | 1.9888 | 2.9201 | 3.5118 | 87.4024 |
| WP6 | 1.4906 | 4.5325 | 5.4164 | 83.6023 |
| WP7 | 0.9929 | 7.6726 | 9.0817 | 78.9791 |
| WP8 | 0.7441 | 10.563 | 12.4039 | 76.1291 |

## Electrocardiography (ECG) signals

Average results for the ECG dataset are provided in Figure 10 and Table 6. In Figure 10, PSNR is only shown WP1-WP8. For WP0, PSNR is always Inf, and hence left out from the averaged results. The per-item results, including three graphs providing the PRD, CPRD, and PSNR metrics as a function of the BPS values for the working points WP1-WP8, and tables that list PRD, CPRD, and PSNR values for each bitstream, are provided in Appendices (see separate files).



Figure 10. Average PRD, CPRD, and PSNR values over bits per sample (BPS) averaged per working point for the ECG dataset. Vertical grid lines correspond to the working points WP1-WP8 as defined in [1].

Table 6. Average BPS, PRD, CPRD, and PSNR values for the ECG dataset.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Working point | BPS | PRD | CPRD | PSNR |
| WP0 | 3.7758 | 0 | 0 | Inf |
| WP1 | 3.7362 | 0.0905 | 0.1006 | 145.9983 |
| WP2 | 3.4511 | 0.5066 | 0.5638 | 110.1851 |
| WP3 | 2.9866 | 0.8244 | 0.9206 | 101.9863 |
| WP4 | 2.4832 | 1.133 | 1.2646 | 99.2332 |
| WP5 | 1.9863 | 1.5379 | 1.717 | 96.5803 |
| WP6 | 1.4864 | 2.0694 | 2.3085 | 93.9999 |
| WP7 | 0.9921 | 2.9434 | 3.2776 | 90.9054 |
| WP8 | 0.745 | 4.1074 | 4.5656 | 87.9996 |

## Electromyography (EMG) signals (raw)

Average results for the ECG dataset are provided in Figure 11 and Table 7. In Figure 11, PSNR is only shown WP1-WP8. For WP0, PSNR is always Inf, and hence left out from the averaged results. The per-item results, including three graphs providing the PRD, CPRD, and PSNR metrics as a function of the BPS values for the working points WP1-WP8, and tables that list PRD, CPRD, and PSNR values for each bitstream, are provided in Appendices (see separate files).

Results for the filtered EMG signals that were provided as a part of the EMG test data can be found in the supplementary material.



Figure 11. Average PRD, CPRD, and PSNR values over bits per sample (BPS) averaged per working point for the raw EMG dataset. Vertical grid lines correspond to the working points WP1-WP8 as defined in [1].

Table 7. Average BPS, PRD, CPRD, and PSNR values for the raw EMG dataset.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Working point | BPS | PRD | CPRD | PSNR |
| WP0 | 6.5819 | 0 | 0 | Inf |
| WP1 | 3.9833 | 0.1124 | 0.6906 | 90.8414 |
| WP2 | 3.4862 | 0.1619 | 0.9596 | 87.7165 |
| WP3 | 2.9851 | 0.2356 | 1.3499 | 84.4974 |
| WP4 | 2.4867 | 0.3441 | 1.9203 | 81.2385 |
| WP5 | 1.9883 | 0.5123 | 2.7943 | 77.8381 |
| WP6 | 1.4894 | 0.7927 | 4.2396 | 74.0999 |
| WP7 | 0.9928 | 1.334 | 6.8988 | 69.6582 |
| WP8 | 0.7442 | 1.8359 | 9.1897 | 66.953 |

# Draft specification text

A draft specification text of the proposed technology can be readily made available, based on the present technology description and the provided syntax description.

# Complexity characteristics

To characterize the complexity of the proposed codec, the average encoding and decoding times over the datasets and compare that to the real time were evaluated. The measurements were performed using a single thread of Intel(R) Core(TM) i9-7920X CPU @ 2.90GHz. The datasets contained signals with different number of channels. The results are shown in Figure 12, Figure 13, and Figure 14 below for the EEG, ECG, and EMG datasets, respectively. Complexity characteristics for the two additional signal classes, filtered EMG and PPG signals are reported in the supplementary material.

The real-time factor for the encoder is the mean ratio between an item length in seconds and an encode time in seconds. The real-time factor for the decoder is the mean ratio between an item length in seconds and a decode time. To facilitate comparisons, separate measurements for subset of signals with the same number of channels (blue bars) were performed. Furthermore, also average results normalized with respect to the number of channels (red bars), which allow to compare performance across the datasets are provided. The performance results are presented separately for lossless and lossy operating points of the codec.

It is noted that the provided results come from a research implementation that has not been optimized. However, the results still allow to compare encoder and decoder complexity and study the effect of number of channels and the sampling frequency on the complexity.



Figure 12. Real-time factor for encoding (left) and decoding (right) for the EEG dataset.



Figure 13. Real-time factor for encoding (left) and decoding (right) for the ECG dataset.



Figure 14. Real-time factor for encoding (left) and decoding (right) for the EMG (raw) dataset.

It can be seen that both encoder and decoder can operate orders of magnitude faster than real-time even for high number of channels. The analysis indicates that in the proposed solution the complexity of the decoder is always lower than the complexity of the encoder. This is due to the configuration of the encoder that operates in a bitrate constrained setting and currently needs to optimize its distortion versus bitrate trade-off using a binary search.
The complexity of lossless encoder is lower than the lossy encoder. This is because the lossy encoder currently also includes a lossless mode. It first attempts to perform lossless encoding, and only if the bitrate constraint is not fulfilled it resorts to lossy encoding. This means that in the current setup of the lossy encoder, the lossless encoding step is always performed once, hence the complexity of the lossy encoder is greater than the lossless encoder. It is noted that this is a consequence of the operation with a constrained bitrate, where an encoder attempts to minimize the coding distortion subject to the bitrate constraint. It is further noted that if such a bitrate allocation procedure was performed on per segment basis (as opposed to per signal basis), the complexity of the encoder would drop.

As expected, the computational complexity of the encoder and the decoder increases as a function of number of channels and as the sampling frequency increases.

# Submission package

Table 8 provides a high-level overview of the folders contained in the submission package and a short description of what is included in those folders.

Table 8. Contents of the submission package.

|  |  |  |
| --- | --- | --- |
| Folder | Subfolder | Description of contents |
| Bitstreams\_Executables\_SourceCode | .\Bitstreams | The subfolder ‘PacketizedBitstreams’ contains the packetized bitstreams, including examples of the systemic features.The subfolder ‘**RawBitstreams**’ contains the raw bitstreams that are used for measurements and plots. **These bitstreams shall be included in the evaluation.** |
|  | .\Executables | Windows-executables for encoding and decoding. Executables for running the core coder and creating the packetized bitstreams are provided separately. See the associated README-file for details. |
|  | .\SourceCode | C-Source code of the encoder and decoder. |
| Documentation | .\Documents | Document containing the response to the CfP (this document), Syntax description, and Supplementary material. |
|  | .\Appendices | Per-item figures and tables providing PRD and CPRD results for each bitstream. |

# Conclusions

In this document, Dolby Laboratories’ response to the coding of biomedical waveform data Call for Proposal was presented. Algorithm description, test results, and other information required by the CfP were reported. The proposal consists of **a** core codec that compresses the raw waveforms **and** is conceptually simple, with a unified architecture that supports both lossy and lossless coding. **In addition,** a **flexible and extensible** bitstream containing systemic features that are considered important for the success of the format **are proposed**. The codec has a configurable latency, and it supports full random access. The results presented in the document show that the proposed conceptually simple and lean codec can provide high performance with a limited set of coding tools. The flexible architecture provides a good starting point for experimentation with additional tools.

# Patent rights declarations(s)

Dolby Laboratories may have current or pending patent rights relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation (per box 2 of the ITU-T/ITU-R/ISO/IEC patent statement and licensing declaration form).

# References

1. Call for Proposals on the coding of biomedical waveform data. ([link](https://www.itu.int/en/ITU-T/studygroups/2022-2024/16/Documents/docs/CfP-H.BWC-TD-PLEN-0286-R1-Clean.pdf))
2. J. Pfaff and J. Halford, “Call for Evidence on the coding of biomedical waveform data,” Q.6/SG16, doc*.VCEG-BT07*, Hannover, Nov. 2023.
3. J. Makhoul, “A fast cosine transform in one and two dimensions,” in *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 1, pp. 27-34, February 1980.
4. R. Geiger, Y. Yokotani and G. Schuller, “Improved integer transforms for lossless audio coding,” *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers*, 2003, Pacific Grove, CA, USA, 2003, pp. 2119-2123 Vol.2.
5. J. G. Proakis and B. G. Manolakis, Digital Signal Processing Principles, Algorithms, and Applications (3rd Ed.). 1996, Prentice-Hall Inc. Upper Saddle River, N.J. pp 863-864.
6. A. Němcová, R. Smíšek, L. Maršánová, L. Smital, M. Vítek. “A Comparative Analysis of Methods for Evaluation of ECG Signal Quality after Compression,” *Biomed Res Int.* 2018 Jul 18;2018:1868519. doi: 10.1155/2018/1868519. PMID: 30112363; PMCID: PMC6077674.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_