IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| EDMG Encoding Examples | | | | |
| Date: 2018-10-08 | | | | |
| Author(s): | | | | |
| Name | Affiliation | Address | Phone | email |
| Artyom Lomayev | Intel | Turgeneva 30, Nizhny Novgorod 603024, Russia | +7 (831) 2969444 | artyom.lomayev@intel.com |
| Miki Genossar | Intel |  |  | miki.genossar@intel.com |
| Alexander Maltsev | Intel |  |  | alexander.maltsev@intel.com |
| Claudio da Silva | Intel |  |  | claudio.da.silva@intel.com |
| Carlos Cordeiro | Intel |  |  | carlos.cordeiro@intel.com |

Abstract

This document provides the encoding examples for EDMG PHY to be included into the Annex I in [1].

The <**EDMGEncodingExamples.zip**> file below contains the ZIP file with configurations and corresponding encoding examples in respect to D2.0.

The output files were generated using MATLAB R2017b.



Files are encrypted with the password: 802p11d18s1346r0

**EDMG encoding examples**

This document describes the encoding examples for EDMG PHY defined in Clause 29 in [1]. The encoding examples are provided for FORMAT parameter set to EDMG and EDMG\_MODULATION parameter set to EDMG\_C\_MODE (EDMG encoding example #1) and EDMG\_SC\_MODE. For EDMG\_SC\_MODE two subcases are considered including the single PPDU (EDMG encoding example #2) and two PPDUs aggregated into the same EDMG A-PPDU (EDMG encoding example #3). The input and output data files are contained in the <**EDMGEncodingExamples.zip**> file embedded into this document at the page 1.

**Input files**

The input files include common <**PSDU.tx**t> file which contains the PSDU payload bits of length 64000. This file is common and used in all transmission mode configurations. In the MATLAB environment it can be read by standard dlmread(‘PSDU.txt’) function to extract the payload content. The comma delimiter is used to separate one bit from another.

In case of the single PPDU, the PSDU bit content starts at the first bit and ends at the bit corresponding to the length of the given PSDU.

In case of the EDMG A-PPDU configuration, the bit content for the first PSDU is read out first and then the bit content for the second PSDU is read out second from the same file and etc. The first aggregated PSDU bit content starts at the first bit and ends at the bit corresponding to the length of the first PSDU. The second aggregated PSDU bit content starts at the last bit of the first PSDU plus one and ends at the last bit of the first PSDU plus the length of the second PSDU and etc. The total length of all PSDUs is selected in the way not to exceed the maximum length of 64000 bits.

To configure the PPDU, <**IOT\_D\_2\_0\_EDMG\_C\_MODE\_v\_0\_0.xlsx**>, <**IOT\_D\_2\_0\_EDMG\_SC\_MODE\_v\_0\_0.xlsx**>, or <**IOT\_D\_2\_0\_EDMG\_SC\_MODE\_A\_PPDU\_v\_0\_0.xlsx**> XLSX file is used. The file name reflects the draft version of the standard which was used, for example, D2.0 in the considered example, the EDMG\_MODULATION mode, and the version of the configuration file. The additional part “A\_PPDU” indicates that this file is defined specifically for the EDMG A-PPDU configuration. The prefix “IOT” stands for “Input Output Test”.

Each configuration file contains three sheets, namely, “General”, “CFG #1”, and “CFG #2”. The sheet “General” contains general description of the file and some notes on the used conventions. In each file it is assumed that the sheets “CFG #1” and “CFG #2” define the configuration parameters for the channel bandwidth set to CBW216 and CBW432 accordingly. The first two columns (“A” and “B”) in each sheet define the configuration fields divided into three groups as “General”, “L fields”, and “EDMG fields” related. The field names are aligned with the field names used in the TX/RXVECTOR, L-Header, and EDMG-Header-A. The field values are represented in the decimal notation, the number of bits per field is selected in accordance with its definition in the EDMG PHY.

The columns starting from the column “C” define the parameters for given PPDU. Note, that not all fields may be relevant for given configuration. In that case the irrelevant fields are denoted as “NA” and highlighted by red colour. More conventions on the used colour indications can be found in the “General” sheet of each file.

In order to download the content of the configuration file, one can use the standard MATLAB function [~, ~, Dscr] = xlsread(‘FileName’, SheetNum), where ‘FileName’ is one of the files specified above and SheetNum is the XLSX file sheet number. The output descriptor Dscr(k, n) is a two dimensional MATLAB cell array representing the XLSX file fields. The row index k selects the particular field and the column index n selects the particular index of the PPDU. Afterwards the cell array can be converted to the ordinary array using standard MATLAB function cell2mat( Dscr(k, n) ) or to the “char” array using function char( Dscr(k, n) ).

**Output files**

The output files include the encoding examples contained in the <**IOT\_D\_2\_0\_EDMG\_C\_MODE\_v\_0\_0.mat**>, <**IOT\_D\_2\_0\_EDMG\_SC\_MODE\_v\_0\_0.mat**>, or <**IOT\_D\_2\_0\_EDMG\_SC\_MODE\_A\_PPDU\_v\_0\_0.mat**> MAT file. Note that each output file keeps the same name as the input file (except for the file extension) to which it is coupled.

Each MAT output file contains the cells array named as “CFG”. The “CFG” cells array has M by N dimensions. The first dimension corresponds to the sheet number in the input configuration file and the second dimension corresponds to the PPDU column configuration. The content of the MAT-file can be downloaded using the standard MATLAB function load(‘FileName’), where the ‘FileName’ is one of the files specified above.

To extract the output PPDU, one needs to read out the particular cell entry in the “CFG” matrix as CFG(i, j), where i = 1, 2, …, M and j = 1, 2, …, N. To read out the Output structure contained in the given cell, one can use the standard MATLAB function Output = cell2mat( CFG(i, j) ). The format of the Output structure is dependent on the transmission mode and is described in the sections below.

**EDMG encoding example #1 – EDMG\_C\_MODE**

The first encoding example corresponds to the EDMG\_MODULATION set to EDMG\_C\_MODE.

The encoded and modulated PPDU fields are defined at the original chip rate with no shaping filter applied. Therefore, the Preamble and Data fields are combined into the single array and TRN field is provided separately.

The Output structure for the EDMG\_C\_MODE has the following fields:

* Output.PreambleData – combines the Preamble and Data fields defined at the chip rate Fc = 1.76 GHz
* Output.Trn – TRN field defined at the Fc\_EDMG = 1.76 x NCB GHz, returned as empty array if TRN filed is not present, NCB defines the channel bonding factor

Each row in the array (Output.PreambleData or Output.Trn) corresponds to the given transmit chain number iTX, where iTX = 1, 2, …, NTX. The number of columns corresponds to the length of fields in chips. The chip time duration may be different (dependent on the NCB channel bonding factor) as specified above.

**EDMG encoding example #2 – EDMG\_SC\_MODE, single PPDU**

The second encoding example corresponds to the EDMG\_MODULATION set to EDMG\_SC\_MODE with single PPDU.

The encoded and modulated PPDU fields are defined at the original chip rate with no shaping filter applied. Therefore, the pre-EDMG and EDMG modulated fields in general case are provided in the different arrays.

The Output structure for the EDMG\_SC\_MODE has the following fields:

* Output.PreEdmg – pre-EDMG modulated fields defined at the chip rate Fc = 1.76 GHz
* Output.EDMG – EDMG modulated fields defined at the Fc\_EDMG = 1.76 x NCB GHz, NCB defines the channel bonding factor
* Output.Tx – concatenation of pre-EDMG and EDMG modulated fields in case of NSS = 1 and Fc\_EDMG = Fc = 1.76 GHz

In case of single spatial stream NSS = 1 and channel bandwidth set to CBW216, the Output structure is returned as follows:

* Output.PreEdmg – returned as empty array
* Output.EDMG – returned as empty array
* Output.Tx – concatenation of pre-EDMG and EDMG modulated fields

In that case both pre-EDMG and EDMG modulated fields have common symbol blocking structure and defined at the same chip rate.

In all other cases (excluding the NSS = 1 and BW = CBW216), the pre-EDMG and EDMG modulated fields have different symbol blocking structures separated by the EDMG-STF and EDMG-CEF fields and the Output structure is returned as follows:

* Output.PreEdmg – pre-EDMG modulated fields defined at the chip rate Fc = 1.76 GHz
* Output.EDMG – EDMG modulated fields defined at the Fc\_EDMG = 1.76 x NCB GHz, NCB defines the channel bonding factor
* Output.Tx – returned as empty array

Each row in the array corresponds to the given transmit chain number iTX, where iTX = 1, 2, …, NTX. The number of columns corresponds to the length of fields in chips. The chip time duration may be different (dependent on the NCB channel bonding factor) as specified above.

**EDMG encoding example #3 – EDMG\_SC\_MODE, EDMG A-PPDU**

The third encoding example corresponds to the EDMG\_MODULATION set to EDMG\_SC\_MODE with EDMG A-PPDU.

The Output structure has the same format as the structure defined in the EDMG encoding example #2.

In case when several PPDUs are aggregated into the single EDMG A-PPDU, the “additional\_edmg\_ppdu” field in the XLSX file is set to 1, except the last one, which is set to 0. The “General” and “L fields” values are kept unchanged over all PPDUs included into the same EDMG A-PPDU. The “EDMG fields” values are set in accordance with the rules defined in the EDMG PHY.

**Mapper and π/2 rotation implementation in MATLAB**

The mapper equations and π/2 rotation implementation approach are described below.

The π/2 rotation is implemented as a multiplication by the R( mod(n-1,4)+1 ) element of the array R = [1, 1j, -1, -1j], where index n goes from 1 and up to the size of the output array.

The output symbols of the π/2-BPSK modulation are defined for the input array of bits c as follows:

L1 = floor(length(c)/4); L2 = length(c) - L1\*4;

RR = [repmat(R,1,L1),R(1:L2)];

Output = (2.\*c-1).\*RR;

The output symbols of the π/2-QPSK modulation are defined for the input array of bits c as follows:

c = reshape(c,2,[]);

Output = ((c(1,:)+c(2,:)-1) + 1j.\*(c(2,:)-c(1,:))).\*RR;

The output symbols of the π/2-16-QAM modulation are defined for the input array of bits c as follows:

c = reshape(c,4,[]);

Real = (2.\*c(1,:)-1).\*(3-2.\*c(2,:));

Imag = (2.\*c(3,:)-1).\*(3-2.\*c(4,:));

Output = (1./sqrt(10)).\*( Real + 1j.\*Imag ).\*RR;

The output symbols of the π/2-64-QAM modulation are defined for the input array of bits c as follows:

c = reshape(c,6,[]);

Real = ((8.\*c(1,:)-4)-(2.\*c(1,:)-1).\*(4.\*c(2,:)-2)+(2.\*c(1,:)-1).\*(2.\*c(2,:)-1).\*(2.\*c(3,:)-1));

Imag = ((8.\*c(4,:)-4)-(2.\*c(4,:)-1).\*(4.\*c(5,:)-2)+(2.\*c(4,:)-1).\*(2.\*c(5,:)-1).\*(2.\*c(6,:)-1));

Output = (1./sqrt(42)).\*( Real + 1j.\*Imag ); % Rotation is applied at the output of interleaver

**References:**

1. Draft P802.11ay\_D2.0