A Follow-up to Quantization Noise Feature in COM Matlab Code

COM Commit Request Number 4p7_4

Hossein Shakiba
Huawei Technologies Canada
February 04, 2025

Introduction

- Quantization noise feature was added to Matlab code 480beta2 as per COM commit request 4p6_5 (see shakiba_3dj_COM_01_241029.pdf)
- The following follow-up changes are requested here:

1) Variable name change (for technical correctness)
2) Variable name conflict fix
3) Improvement and a potential issue fix in calculating quantization noise PDF after RxFFE
4) Changing quantization noise PSD calculation method during optimization iterations to be consistent with method used for COM calculation (improvement)
5) Parameter name change (also relates to Change 6 below)
6) Adding a quantization clip rate parameter, so it can be user defined
   - This change was a part of COM commit request 4p6_5 and should have been implemented in 480beta2 (see shakiba_3dj_COM_01_241029.pdf)

Change 1 and 6 (Partial) of 6 (Technical Crrectness)

- Variable name “ctle_pulse_sigma” misrepresents the variable content
  - Change variable name to a correct representative name “ctle_signal_sigma” (3 occurrences in 3 lines)
- Change 6 partially shown in blue
  - Change hard-coded clip rate to a parameter (see slide 11)

```
3274 -  adc_clip = -CDF_inv_ev(2*param.specBER, sig_after_ctle_pdf, sig_after_ctle_cdf);
3275 -  ctle_pulse_sigma = sqrt(sum((sig_after_ctle_pdf.x.^2).*sig_after_ctle_pdf.y));
3276 -  adc_lsb = 2*adc_clip/(2^param.ENOB-1);
3277 -  NS.sigma_Q = adc_lsb/sqrt(12);
3278 -  NS.sigma_before_clip = ctle_pulse_sigma;
3279 -  NS.peak_clip = adc_clip;
3280 -  NS.p2ptosigma_clip = 2*adc_clip/ctle_pulse_sigma;
```

Code before change

```
3274 -  adc_clip = -CDF_inv_ev(param.P_qc, sig_after_ctle_pdf, sig_after_ctle_cdf);
3275 -  ctle_signal_sigma = sqrt(sum((sig_after_ctle_pdf.x.^2).*sig_after_ctle_pdf.y));
3276 -  adc_lsb = 2*adc_clip/(2^param.N_qb-1);
3277 -  NS.sigma_Q = adc_lsb/sqrt(12);
3278 -  NS.sigma_before_clip = ctle_signal_sigma;
3279 -  NS.peak_clip = adc_clip;
3280 -  NS.p2ptosigma_clip = 2*adc_clip/ctle_signal_sigma;
```

Code after change

Change 2 of 6 (Bug Fix)

- Two different variables (quantization noise PDFs at the input and output of the RxFFE) used the same name “quantizaiton_noise_pdf”
  - Change PDF name at RxFFE input to “quantization_noise_in_pdf” (7 occurrences in 6 lines) and add a comment
  - Fix spelling error in “quantizaiton”

Insert here

```
3281 -  quantizaiton_noise_pdf = combined_interference_and_noise_pdf; % This is to copy the fields of the structure
3282 -  [~,adc_ind_right] = min(abs(quantizaiton_noise_pdf.x-adc_lsb/2));
3283 -  [~,adc_ind_left] = min(abs(quantizaiton_noise_pdf.x+adc_lsb/2));
3284     % adc_ind_right= find (    min(abs(quantizaiton_noise_pdf.x-adc_lsb/2)) ==  abs(quantizaiton_noise_pdf.x-adc_lsb/2)   );
3285     % adc_ind_left=  find (    min(abs(quantizaiton_noise_pdf.x+adc_lsb/2)) ==  abs(quantizaiton_noise_pdf.x+adc_lsb/2)   );
3286 -  quantizaiton_noise_pdf.y = zeros(size(quantizaiton_noise_pdf.x));
3287 -  quantizaiton_noise_pdf.y(adc_ind_left:adc_ind_right) = 1/(adc_ind_right-adc_ind_left+1);
3288     % Calculate quantization noise PDF after RxFFE
3289 -  h_rxffe = fom_result.RxFFE(find(fom_result.RxFFE ~= 0));
3290 -  for irxffe = 1:length(h_rxffe)
3291         if irxffe ~= param.ffe_pre_tap_len
3292 -           quantizaiton_noise_pdf_scale = scalePDF(quantizaiton_noise_pdf, abs(h_rxffe(irxffe)));
3293 -           quantizaiton_noise_pdf = conv_fct(quantizaiton_noise_pdf, quantizaiton_noise_pdf_scale);
3294 -       end
3295 -   end
3296 -   combined_interference_and_noise_pdf = conv_fct(combined_interference_and_noise_pdf, quantizaiton_noise_pdf);
3297 -   NS.quantizaiton_noise_pdf = quantizaiton_noise_pdf ;
```

Code before change

```
3281      % Construct the uniform quantization noise PDF at the injection point
3282 -    quantization_noise_in_pdf = combined_interference_and_noise_pdf; % This is to copy the fields of the structure
3283 -    [~,adc_ind_right] = min(abs(quantization_noise_in_pdf.x-adc_lsb/2));
3284 -    [~,adc_ind_left] = min(abs(quantization_noise_in_pdf.x+adc_lsb/2));
3285      % adc_ind_right= find (    min(abs(quantizaiton_noise_pdf.x-adc_lsb/2)) ==  abs(quantizaiton_noise_pdf.x-adc_lsb/2)   );
3286      % adc_ind_left=  find (    min(abs(quantizaiton_noise_pdf.x+adc_lsb/2)) ==  abs(quantizaiton_noise_pdf.x+adc_lsb/2)   );
3287 -    quantization_noise_in_pdf.y = zeros(size(quantization_noise_in_pdf.x));
3288 -    quantization_noise_in_pdf.y(adc_ind_left:adc_ind_right) = 1/(adc_ind_right-adc_ind_left+1);
3289      % Initialize quantization noise PDF after Rx FFE to a Dirac delta PDF
3290 -    quantization_noise_pdf = combined_interference_and_noise_pdf; % This is to copy the fields of the structure
3291 -    [~,ind_center] = min(abs(quantization_noise_pdf.x));
3292 -    quantization_noise_pdf.y = zeros(size(quantization_noise_pdf.x));
3293 -    quantization_noise_pdf.y(ind_center) = 1;
3294      % Calculate quantization noise PDF after RxFFE
3295 -    h_rxffe = fom_result.RxFFE(find(fom_result.RxFFE ~= 0));
3296 -    for irxffe = 1:length(h_rxffe)
3297 -        quantization_noise_in_pdf_scale = scalePDF(quantization_noise_in_pdf, abs(h_rxffe(irxffe)));
3298 -        quantization_noise_pdf = conv_fct(quantization_noise_pdf, quantization_noise_in_pdf_scale);
3299 -    end
3300 -    combined_interference_and_noise_pdf = conv_fct(combined_interference_and_noise_pdf, quantization_noise_pdf);
3301 -    NS.quantization_noise_pdf = quantization_noise_pdf ;
```

Relates to Change 3 (see next slide).

Code after change

Change 3 of 6 (Improvement and Bug Fix) – 1/2

- Quantization noise at the RxFFE output was initialized to a uniform PDF at the RxFFE input (same as the injection point), consequently the convolution loop had to skip the main tap (main tap = 1)
- An “if” statement was used to ignore the main tap (BTW, the “if” condition should have been “param.ffe_pre_tap_len+1”)
- For faster execution, RxFFE taps that were zero were excluded from convolution loop (important with floating taps)
- Potential issue: Location of the main tap changes if any of the pre-taps happens to be zero, causing an indexing issue (very unlikely)
- Fix: Initialize to Dirac delta and include the main tap in the loop to avoid the above altogether
  - 3.1: Add 5 lines for initialization (including 1 comment line)
  - 3.2: Remove the “if” statement (2 lines)
- 3.3: For consistency with the new variable name in Change 2, change variable name “quantization_noise_pdf_scale” to “quantization_noise_in_pdf_scale” (2 occurrences in 2 lines)

Change 3 of 6 (Improvement and Bug Fix) – 2/2

3.1: Insert here

3.2: Delete

```
3281 -  quantizaiton_noise_pdf = combined_interference_and_noise_pdf; % This is to copy the fields of the structure
3282 -  [~,adc_ind_right] = min(abs(quantizaiton_noise_pdf.x-adc_lsb/2));
3283 -  [~,adc_ind_left] = min(abs(quantizaiton_noise_pdf.x+adc_lsb/2));
3284    % adc_ind_right= find (    min(abs(quantizaiton_noise_pdf.x-adc_lsb/2)) ==  abs(quantizaiton_noise_pdf.x-adc_lsb/2)   );
3285    % adc_ind_left=  find (    min(abs(quantizaiton_noise_pdf.x+adc_lsb/2)) ==  abs(quantizaiton_noise_pdf.x+adc_lsb/2)   );
3286 -  quantizaiton_noise_pdf.y = zeros(size(quantizaiton_noise_pdf.x));
3287 -  quantizaiton_noise_pdf.y(adc_ind_left:adc_ind_right) = 1/(adc_ind_right-adc_ind_left+1);
3288    % Calculate quantization noise PDF after RxFFE
3289 -  h_rxffe = fom_result.RxFFE(find(fom_result.RxFFE ~= 0));
3290 -  for irxffe = 1:length(h_rxffe)
3291 -      if irxffe ~= param.ffe_pre_tap_len
3292 -          quantizaiton_noise_pdf_scale = scalePDF(quantizaiton_noise_pdf, abs(h_rxffe(irxffe)));
3293 -          quantizaiton_noise_pdf = conv_fct(quantizaiton_noise_pdf, quantizaiton_noise_pdf_scale);
3294 -      end
3295 -  end
3296 -  combined_interference_and_noise_pdf = conv_fct(combined_interference_and_noise_pdf, quantizaiton_noise_pdf);
3297 -  NS.quantizaiton_noise_pdf = quantizaiton_noise_pdf ;
```

Code before change

3.3

3.3

```
3281    % Construct the uniform quantization noise PDF at the injection point
3282 -  quantizaiton_noise_in_pdf = combined_interference_and_noise_pdf; % This is to copy the fields of the structure
3283 -  [~,adc_ind_right] = min(abs(quantizaiton_noise_in_pdf.x-adc_lsb/2));
3284 -  [~,adc_ind_left] = min(abs(quantizaiton_noise_in_pdf.x+adc_lsb/2));
3285    % adc_ind_right= find (    min(abs(quantizaiton_noise_pdf.x-adc_lsb/2)) ==  abs(quantizaiton_noise_pdf.x-adc_lsb/2)   );
3286    % adc_ind_left=  find (    min(abs(quantizaiton_noise_pdf.x+adc_lsb/2)) ==  abs(quantizaiton_noise_pdf.x+adc_lsb/2)   );
3287 -  quantizaiton_noise_in_pdf.y = zeros(size(quantizaiton_noise_in_pdf.x));
3288 -  quantizaiton_noise_in_pdf.y(adc_ind_left:adc_ind_right) = 1/(adc_ind_right-adc_ind_left+1);
3289    % Initialize quantization noise PDF after RxFFE to a Dirac delta PDF
3290 -  quantizaiton_noise_pdf = combined_interference_and_noise_pdf; % This is to copy the fields of the structure
3291 -  [~,ind_center] = min(abs(quantizaiton_noise_pdf.x));
3292 -  quantizaiton_noise_pdf.y = zeros(size(quantizaiton_noise_pdf.x));
3293 -  quantizaiton_noise_pdf.y(ind_center) = 1;
3294    % Calculate quantization noise PDF after RxFFE
3295 -  h_rxffe = fom_result.RxFFE(find(fom_result.RxFFE ~= 0));
3296 -  for irxffe = 1:length(h_rxffe)
3297 -      quantizaiton_noise_in_pdf_scale = scalePDF(quantizaiton_noise_in_pdf, abs(h_rxffe(irxffe)));
3298 -      quantizaiton_noise_pdf = conv_fct(quantizaiton_noise_pdf, quantizaiton_noise_in_pdf_scale);
3299 -  end
3300 -  combined_interference_and_noise_pdf = conv_fct(combined_interference_and_noise_pdf, quantizaiton_noise_pdf);
3301 -  NS.quantizaiton_noise_pdf = quantizaiton_noise_pdf ;
```

Code after change

Change 4 and 6 (Partial) of 6 (Improvement) – 1/3

- Clip level is calculated in two places, using two methods:
  1) To calculate quantization noise PDF for COM
  2) To calculate quantization noise PSD for optimization
- Current approach:
  - Method 1 is more accurate, but requires computing PDF of the signal using optimally-sampled pulse response
  - To avoid the above computation in every iteration of the optimization loop, a less accurate and simpler method was used to estimate the clip level to calculate quantization noise PSD in “get_PSDs” function
- Change to:
  - A closer look now indicates that pulse response “hext”, is already calculated for the purpose of Tx noise PSD calculation in the “get_PSDs” function
    - This pulse response can be reused and only needs one additional convolution to include Tx FFE
  - Use the more accurate method 1 in “get_PSDs” function as well
    - 4.1: Move “hext” calculation to outside the “if ~OP.COMPUTE_COM” statement to make it globally available
    - 4.2: Change the method of calculating “adc_clip” to method 1
    - 4.3: Add “adc_clip” and “ctle_signal_sigma” to the output structure (useful information for reporting)

Change 4 and 6 (Partial) of 6 (Improvement) – 2/3

```
4891
4892 -   %% Transmitter noise power spectral density
4893 -   if ~OP.COMPUTE_COM
4894 -       if ~OP.TDMODE
4895 -           htn=filter(ones(1,M),1,chdata(1).ctle_imp_response); % ctle_imp_response does not have TxFFE included
4896 -       else % only use when the input was a pulse response not s-parameters
4897 -           if isfield(chdata(1),'ctle_pulse_response')
4898 -               htn=chdata(1).ctle_pulse_response;
4899 -           else
4900 -               htn=filter(ones(1,param.samples_per_ui),1, chdata(1).ctle_imp_response);
4901 -           end
4902 -       end
4903 -       htn=htn(mod(cursor_i,M)+1:end-mod(cursor_i,M)); % align to sample point
4904 -       htn=reshape(htn,1,[]); % make row vectors
4905 -       htn=[ htn(1:floor(length(htn)/M)*M) ];
4906 -       htn= [htn zeros(1,num_ui*M-length(htn)) ];
4907 -       htn=htn(1:M:end);% resample
4908 -       if num_ui>length(htn)
4909 -           hext=[htn zeros(1,num_ui-length(htn))];
4910 -       else
4911 -           hext=htn(1:num_ui);
4912 -       end
4913 -       result.S_tn=sigma_X2*10^(-SNR_TX/10)*(abs(fft(hext))).^2/param.fb; % this corresponds to +/- pi
4914 -       result.S_tn_rms = sqrt(sum(result.S_tn)* delta_f);
4915 -   else
4916 -       result.S_tn=result.S_tn.*H_rxffe_2;
4917 -       result.S_tn_rms = sqrt(sum(result.S_tn)* delta_f);
    end
```

Code before change

4.1: Move here

```
4895
4896 -   %% Transmitter noise power spectral density
4897 -   if ~OP.TDMODE
4898 -       htn=filter(ones(1,M),1,chdata(1).ctle_imp_response); % ctle_imp_response does not have TxFFE included
4899 -   else % only use when the input was a pulse response not s-parameters
4900 -       if isfield(chdata(1),'ctle_pulse_response')
4901 -           htn=chdata(1).ctle_pulse_response;
4902 -       else
4903 -           htn=filter(ones(1,param.samples_per_ui),1, chdata(1).ctle_imp_response);
4904 -       end
4905 -   end
4906 -   htn=htn(mod(cursor_i,M)+1:end-mod(cursor_i,M)); % align to sample point
4907 -   htn=reshape(htn,1,[]); % make row vectors
4908 -   htn=[ htn(1:floor(length(htn)/M)*M) ];
4909 -   htn= [htn zeros(1,num_ui*M-length(htn)) ];
4910 -   htn=htn(1:M:end);% resample
4911 -   if num_ui>length(htn)
4912 -       hext=[htn zeros(1,num_ui-length(htn))];
4913 -   else
4914 -       hext=htn(1:num_ui);
4915 -   end
4916 -   if ~OP.COMPUTE_COM
4917 -       result.S_tn=sigma_X2*10^(-SNR_TX/10)*(abs(fft(hext))).^2/param.fb; % this corresponds to +/- pi
4918 -       result.S_tn_rms = sqrt(sum(result.S_tn)* delta_f);
4919 -   else
4920 -       result.S_tn=result.S_tn.*H_rxffe_2;
4921 -       result.S_tn_rms = sqrt(sum(result.S_tn)* delta_f);
    end
```

Code after change

Change 4 and 6 (Partial) of 6 (Improvement) – 3/3

- Change 6 partially shown in blue

```
4955 -      if(param.ENOB ~=0)
4956 -          if OP.INCLUDE_CTLE == 1
4957 -              eq_ir = TD_CTLE(chdata(1).uneq_imp_response, param.fb, param.CTLE_fz(1), param.CTLE_fp1(1), param.CTLE_fp2(1), G_DC, param.samples_per_ui);
4958 -              eq_ir = TD_CTLE(eq_ir, param.fb, param.f_HP(1), param.f_HP(1), 100e100 , G_DC2, param.samples_per_ui);
4959 -          else
4960 -              eq_ir = chdata(1).uneq_imp_response;
4961 -          end
4962 -          ctle_pulse = filter(ones(1, param.samples_per_ui), 1, eq_ir);
4963 -          ind_max = find(ctle_pulse == max(ctle_pulse));
4964 -          adc_clip = sum(abs([ctle_pulse(ind_max-param.samples_per_ui:-param.samples_per_ui:1); ctle_pulse(ind_max:param.samples_per_ui:end)]));
4965 -          adc_lsb = 2*adc_clip/(2^param.ENOB-1);
4966 -          sigma_Q = adc_lsb/sqrt(12);
4967 -          S_qn = sigma_Q^2/(length(result.S_rn)*delta_f)*ones(size(result.S_rn));
4968 -          result.S_qn = S_qn;
4969 -          result.qn_rms = sqrt(sum(result.S_qn)* delta_f);
4970 -      else
4971 -          result.S_qn=0;
4972 -          result.S_qn_rms = 0;
4973            %  result.S_n
4974 -      end
4975 -      result.S_n=result.S_rn+ result.S_tn+ result.S_xn+ result.S_jn+ result.S_qn;
4976 -      result.S_n_rms = sqrt(sum(result.S_n)* delta_f);
```

Code before change

4.2: Change to

4.3: Insert here

```
4959 -      if(param.N_qb ~=0)
4960 -          hext_txffe = filter(txffe, 1, hext);
4961 -          sig_after_ctle_pdf = get_pdf_from_sampled_signal(hext_txffe, param.levels, OP.BinSize);
4962 -          noise_after_ctle_pdf = sig_after_ctle_pdf;
4963 -          sigma_noise = sqrt(result.S_rn_rms^2+result.S_xn_rms^2+result.S_tn_rms^2+result.S_rj_rms^2);
4964 -          noise_after_ctle_pdf.y = 1/(sqrt(2*pi)*sigma_noise)*exp(-noise_after_ctle_pdf.x.^2/(2*sigma_noise^2))*OP.BinSize;
4965 -          sig_noise_after_ctle_pdf = conv_fct(sig_after_ctle_pdf, noise_after_ctle_pdf);
4966 -          sig_noise_after_ctle_cdf = cumsum(sig_noise_after_ctle_pdf.y);
4967 -          ctle_signal_sigma = sqrt(sum((sig_noise_after_ctle_pdf.x.^2).*sig_noise_after_ctle_pdf.y));
4968 -          adc_clip = -CDF_inv_ev(param.P_qc, sig_noise_after_ctle_pdf, sig_noise_after_ctle_cdf);
4969 -          adc_lsb = 2*adc_clip/(2^param.N_qb-1);
4970 -          sigma_Q = adc_lsb/sqrt(12);
4971 -          S_qn = sigma_Q^2/f_b*ones(size(hext));
4972 -          result.adc_clip = adc_clip;
4973 -          result.ctle_signal_sigma = ctle_signal_sigma;
4974 -          result.S_qn = S_qn;
4975 -          result.S_qn_rms = sqrt(sum(result.S_qn)* delta_f);
4976 -      else
4977 -          result.S_qn=0;
4978 -          result.S_qn_rms = 0;
4979            %  result.S_n
4980 -      end
4981 -      result.S_n=result.S_rn+ result.S_tn+ result.S_xn+ result.S_jn+ result.S_qn;
4982 -      result.S_n_rms = sqrt(sum(result.S_n)* delta_f);
```

Code after change

Change 5 of 6

- Change parameter name “ENOB” to “N_qb” (7 occurrences in 6 lines)
  - Based on the concerns around “ENOB” historically carrying a different meaning of number of bits

Code after change

```
1500 -  if param.N_qb ~=0
2617 -  if param.N_qb ~= 0
3276 -  adc_lsb = 2*adc_clip/(2^param.N_qb-1);
4959 -      if(param.N_qb ~=0)
4969 -          adc_lsb = 2*adc_clip/(2^param.N_qb-1);
8911 -  param.N_qb = xls_parameter(parameter, 'N_qb',true,0); % adc number of bits if 0 do not apply quantization
```

Code before change

```
1500 -  if param.ENOB ~=0
2617 -  if param.ENOB ~= 0
3276 -  adc_lsb = 2*adc_clip/(2^param.ENOB-1);
4955 -      if(param.ENOB ~=0)
4965 -          adc_lsb = 2*adc_clip/(2^param.ENOB-1);
8905 -  param.ENOB = xls_parameter(parameter, 'ENOB',true,0); % adc number of bits if 0 do not apply quantization
```

- Upon above, change parameter name “adc_clip_rate” to “P_qc” for consistency in naming convention (2 occurrences in 1 line)
  - “adc_clip_rate” should have been fully implemented in 480beta2 as per COM commit request 4p6_5 (see shakiba_3dj_COM_01_241029.pdf)

Code before change

```
8906 -  param.adc_clip_rate= xls_parameter(parameter, 'adc_clip_rate',true,2*param.specBER); % adc clipping probability
```

Code after change

```
8912 -  param.P_qc= xls_parameter(parameter, 'P_qc',true,2*param.specBER); % adc clipping probability
```

  - Also see Change 6 (next slide)

Change 6 of 6

- Implement parametrizing quantization clip rate (should have been implemented in COM 480beta2)
- See slides 3 and 9
- Implement along with Change 5 (slide 10)

Example Outputs

- For one example channel:

Number of Bits	Clip Rate	COM* [dB]	
		before Change	after Change
0	4E-4	-0.7485	-0.7485
	4E-5	-0.7485	-0.7485
6	4E-4	-1.6629	-1.1913
	4E-5	-1.6629	-1.2290

* MLSE not included

- Better COM results after fixing the bug in convolution (variable name conflict) that caused over-estimation of quantization noise
- Without the change clip rate does not work

Thank You ☺

Hossein Shakiba
Huawei Technologies Canada

Back-up Slide

- The following contributions relate to quantization noise:

1) shakiba_3dj_01a_2403.pdf
   - ❖ Quantization noise was first introduced as a part of the receiver implementation penalty
   - ❖ Includes theoretical background and data from test cases
2) shakiba_3dj_02_2405.pdf
   - ❖ Highlights the significance of 1uantization noise in COM evaluation
   - ❖ Includes theoretical background and data from test cases
3) shakiba_3dj_COM_02_241001.pdf
   - ❖ COM Commit Request 4p6_5 to add quantization noise capability to the COM Matlab function for exploration purposes
4) shakiba_3dj_COM_01_241029.pdf
   - ❖ Updated version of the above contribution