Follow-up to COM Commit Request Number 4p8_5

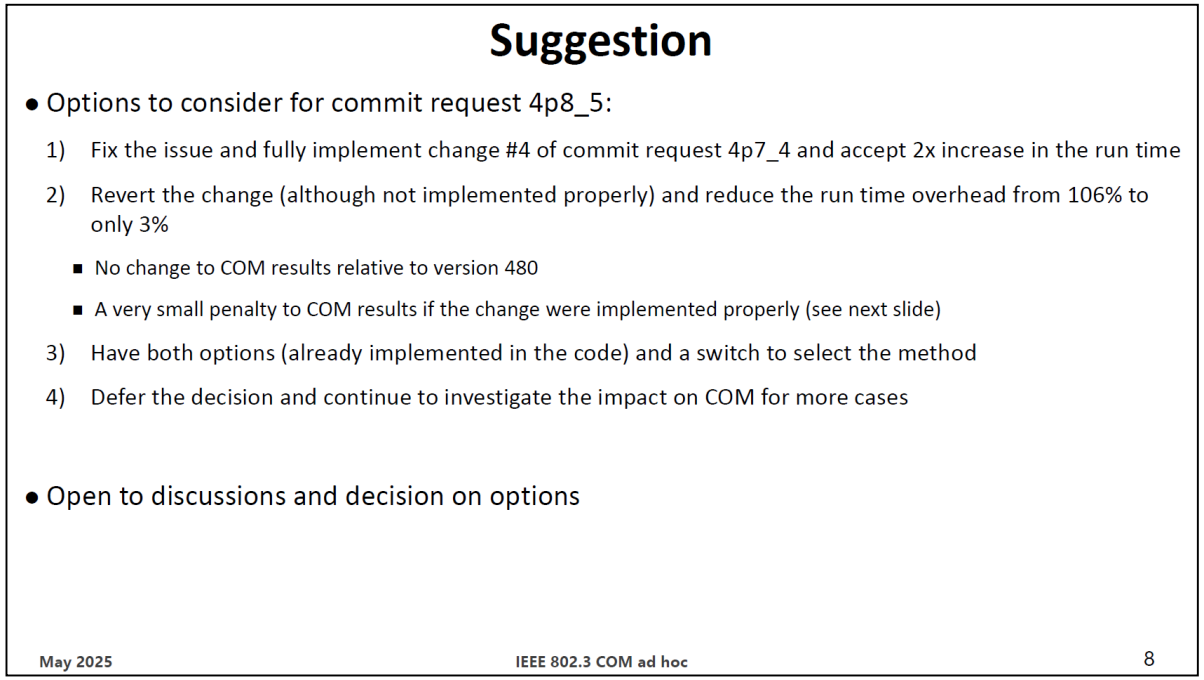
COM Commit Request Number 4p9_1

Hossein Shakiba
Huawei Technologies Canada
June 17 2025

Background

- Commit request 4p8_5 was presented in the COM ad hoc meeting on May 05, 2025 during the May interim in New Orleans (shakiba_3dj_COM_03_2505.pdf)
- The request was to address an issue with implementation of an earlier commit request (change #4 of commit request 4p7_4) as well as to decide on the opportunity to reduce the runtime when quantization noise feature is enabled
- Four options were presented →
- Consensus was to proceed with Option 3
- A follow-up was requested to provide more content on option 3 and a code submission request through the open source repository
- Since now version 4p90 is available, this follow-up presentation and the code change request are relative to version 4p90

Slide 8 of "shakiba_3dj_COM_03_2505.pdf"



Suggestion

- Options to consider for commit request 4p8_5:
  1) Fix the issue and fully implement change #4 of commit request 4p7_4 and accept 2x increase in the run time
  2) Revert the change (although not implemented properly) and reduce the run time overhead from 106% to only 3%
     - No change to COM results relative to version 480
     - A very small penalty to COM results if the change were implemented properly (see next slide)
  3) Have both options (already implemented in the code) and a switch to select the method
  4) Defer the decision and continue to investigate the impact on COM for more cases
- Open to discussions and decision on options

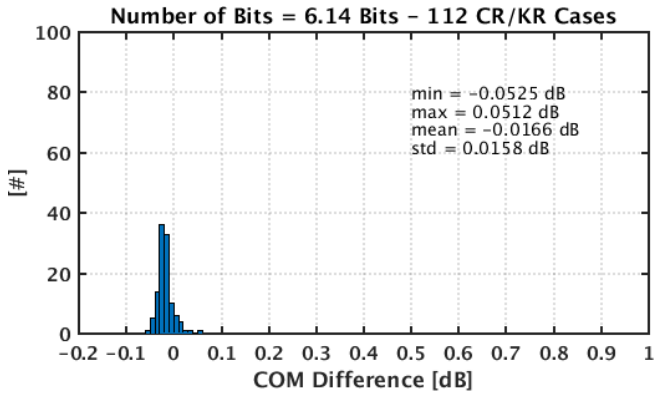
May 2025 IEEE 802.3 COM ad hoc 8

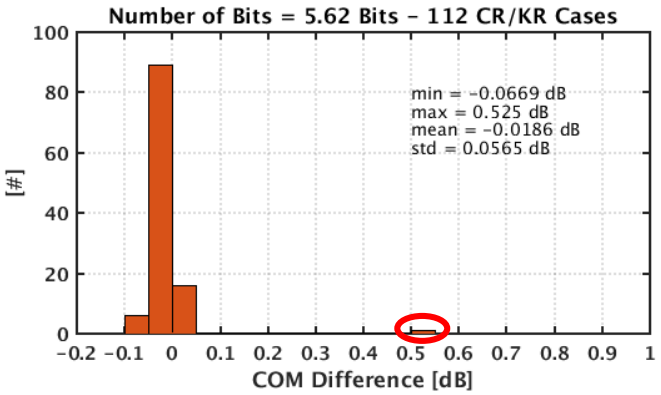
Introduction

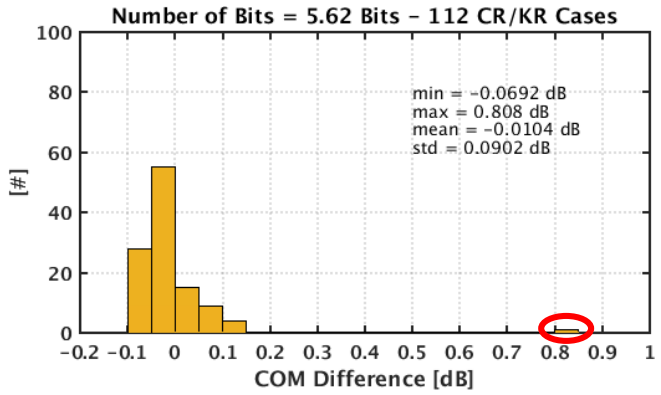
- Two methods have been considered for calculation of quantizer clip level during the optimization loop
  1) Fast and less accurate
  2) Slow and more accurate

Average Runtime Overhead Fast Method	Average Runtime Overhead Slow Method
3% Overhead	106% Overhead

- For 3x112 of test cases COM difference between two methods is almost negligible except for two cases







- Option 3 implements both methods and enables the user to select one through a switch defined as a parameter in the COM configuration

Description of the Change to Implement Option 3

- Both methods are already available in the code (in function “get_PSDs”)
- What the change does:
  1) Addition of a switch to select between two methods in the “get_PSDs” function
     a) Bypass calculation of pulse response during optimization iterations if the slow method is not selected
     b) Only calculate signal PDF during optimization iterations if the slow method is selected
  2) Addition of a new parameter in the “read_ParamConfigFile” function to select the method
- Further runtime reduction is expected if the slow method is not selected due to the additional saving of 1)a) above
- Link to the branch containing new version of the code with the above changes:

  https://opensource.ieee.org/shakiba/com_code/-/tree/Quantization_Noise?ref_type=heads
- Link to the merge request:

  https://opensource.ieee.org/802-com/com_code/-/merge_requests/7

Change 1)a) “get_PSDs”

src/get_PSDs.m +49 -46

```
@@ -126,25 +126,27 @@ else % find noise for item that set have tx ffe for each loop
126 126      %% S_tn from eq 178A-17
127 127      %% if not in the opimization use value found in optimize_fom times |Hrxffe|^2
128 128      %% Transmitter noise power spectral density
129     -    if ~OP.TDMODE
130     -        htn=filter(ones(1,M),1,chdata(1).ctle_imp_response); % ctle_imp_response does not have TxFFE included
131     -    else % only use when the input was a pulse response not s-parameters
132     -        if isfield(chdata(1),'ctle_pulse_response')
133     -            htn=chdata(1).ctle_pulse_response;
    129 +    if ~OP.COMPUTE_COM || strcmpi(param.clip_method, 'slow') % "if" to "end" section changed by Hossein Shakiba to implement commit request 4p9_1
    130 +        if ~OP.TDMODE
    131 +            htn=filter(ones(1,M),1,chdata(1).ctle_imp_response); % ctle_imp_response does not have TxFFE included
    132 +        else % only use when the input was a pulse response not s-parameters
    133 +            if isfield(chdata(1),'ctle_pulse_response')
    134 +                htn=chdata(1).ctle_pulse_response;
    135 +            else
    136 +                htn=filter(ones(1,param.samples_per_ui),1, chdata(1).ctle_imp_response);
    137 +            end
    138 +        end
    139 +        htn=htn(mod(cursor_i,M)+1:end-mod(cursor_i,M)); % align to sample point
    140 +        htn=reshape(htn,1,[]); % make row vectors
    141 +        htn=[ htn(1:floor(length(htn)/M)*M) ];
    142 +        htn= [htn zeros(1,num_ui*M-length(htn)) ];
    143 +        htn=htn(1:M:end);% resample
    144 +        if num_ui>length(htn)
    145 +            hext=[htn zeros(1,num_ui-length(htn))];
134 146          else
135     -            htn=filter(ones(1,param.samples_per_ui),1, chdata(1).ctle_imp_response);
    147 +            hext=htn(1:num_ui);
136 148          end
137 149      end
138     -    htn=htn(mod(cursor_i,M)+1:end-mod(cursor_i,M)); % align to sample point
139     -    htn=reshape(htn,1,[]); % make row vectors
140     -    htn=[ htn(1:floor(length(htn)/M)*M) ];
141     -    htn= [htn zeros(1,num_ui*M-length(htn)) ];
142     -    htn=htn(1:M:end);% resample
143     -    if num_ui>length(htn)
144     -        hext=[htn zeros(1,num_ui-length(htn))];
145     -    else
146     -        hext=htn(1:num_ui);
147     -    end
```

Change 1)b) “get_PSDs”

src/get_PSDs.m +49 -46

```
-   %  result.S_qn
-   if(param.N_qb ~=0)
-       hext_txffe=filter(txffe,1,hext);
-       sig_aftert_ctle_pdf = get_pdf_from_sampled_signal(hext_txffe,param.levels,GP.BinSize);
-       noise_after_ctle_pdf = sig_aftert_ctle_pdf;
-       sigma_noise = sqrt(result.S_rn_rms^2+result.S_xn_rms^2++result.S_tn_rms^2++result.S_rj_rms^2);
-       noise_after_ctle_pdf.y = 1/(sqrt(2*pi)*sigma_noise)*exp(-noise_after_ctle_pdf.x.^2/(2*sigma_noise^2))*GP.BinSize;
-       sig_noise_after_ctle_pdf= conv_fct(sig_aftert_ctle_pdf,noise_after_ctle_pdf);
-       sig_noise_after_ctle_cdf = cumsum(sig_noise_after_ctle_pdf.y);
-       ctle_signal_sigma = sqrt(sum((sig_noise_after_ctle_pdf.x.^2).*sig_noise_after_ctle_pdf.y));
-       adc_clip=-CDF_inv_ev(param.P_qc, sig_noise_after_ctle_pdf,sig_noise_after_ctle_cdf);
-       adc_lsb=2*adc_clip/(2^param.N_qb-1);
-       sigma_Q=adc_lsb/sqrt(12);
-       S_qn=sigma_Q^2/f_b*ones(size(hext));
-       result.adc_clip=adc_clip;
-       result.ctle_signal_sigma=ctle_signal_sigma;
-       result.S_qn=S_qn;
-       result.s_qn_rms=sqrt(sum(result.S_qn)*delta_f);
-       if GP.INCLUDE_CTLE == 1
-           eq_ir = TD_CTLE(chdata(1).uneq_imp_response, param.fb, param.CTLE_fz(1), param.CTLE_fp1(1), param.CTLE_fp2(1), 0_DC, param.samples_per_ui);
-           eq_ir = TD_CTLE(eq_ir, param.fb, param.f_HP(1), param.f_HP(1), 100e100 , 0_DC2, param.samples_per_ui);
+
+   %%  Quantization Noise (S_qn)
+   if(param.N_qb ~=0) % "if" to "else" section changed by Hossein Shakiba to implement commit request 4p9_1
+       if strcmpi(param.clip_method, 'slow')
+           hext_txffe=filter(txffe,1,hext);
+           sig_aftert_ctle_pdf = get_pdf_from_sampled_signal(hext_txffe,param.levels,GP.BinSize);
+           noise_after_ctle_pdf = sig_aftert_ctle_pdf;
+           sigma_noise = sqrt(result.S_rn_rms^2+result.S_xn_rms^2++result.S_tn_rms^2++result.S_rj_rms^2);
+           noise_after_ctle_pdf.y = 1/(sqrt(2*pi)*sigma_noise)*exp(-noise_after_ctle_pdf.x.^2/(2*sigma_noise^2))*GP.BinSize;
+           sig_noise_after_ctle_pdf= conv_fct(sig_aftert_ctle_pdf,noise_after_ctle_pdf);
+           sig_noise_after_ctle_cdf = cumsum(sig_noise_after_ctle_pdf.y);
+           ctle_signal_sigma = sqrt(sum((sig_noise_after_ctle_pdf.x.^2).*sig_noise_after_ctle_pdf.y));
+           adc_clip=-CDF_inv_ev(param.P_qc, sig_noise_after_ctle_pdf,sig_noise_after_ctle_cdf);
+           adc_lsb=2*adc_clip/(2^param.N_qb-1);
+           sigma_Q=adc_lsb/sqrt(12);
+           S_qn=sigma_Q^2/f_b*ones(size(hext));
+           result.ctle_signal_sigma=ctle_signal_sigma;
        else
-           eq_ir = chdata(1).uneq_imp_response;
+           if GP.INCLUDE_CTLE == 1
+               eq_ir = TD_CTLE(chdata(1).uneq_imp_response, param.fb, param.CTLE_fz(1), param.CTLE_fp1(1), param.CTLE_fp2(1), 0_DC, param.samples_per_ui);
+               eq_ir = TD_CTLE(eq_ir, param.fb, param.f_HP(1), param.f_HP(1), 100e100 , 0_DC2, param.samples_per_ui);
+           else
+               eq_ir = chdata(1).uneq_imp_response;
+           end
+           ctle_pulse = filter(ones(1, param.samples_per_ui), 1, eq_ir);
+           ind_max = find(ctle_pulse == max(ctle_pulse));
+           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)]));
+           adc_lsb = 2*adc_clip/(2^param.N_qb-1);
+           sigma_Q = adc_lsb/sqrt(12);
+           S_qn = sigma_Q^2/(length(result.S_rn)*delta_f)*ones(size(result.S_rn));
        end
-       ctle_pulse = filter(ones(1, param.samples_per_ui), 1, eq_ir);
-       ind_max = find(ctle_pulse == max(ctle_pulse));
-       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)]));
-       adc_lsb = 2*adc_clip/(2^param.N_qb-1);
-       sigma_Q = adc_lsb/sqrt(12);
-       S_qn = sigma_Q^2/(length(result.S_rn)*delta_f)*ones(size(result.S_rn));
+       result.adc_clip=adc_clip;
        result.S_qn = S_qn;
-       result.qn_rms = sqrt(sum(result.S_qn)* delta_f);
+       result.S_qn_rms = sqrt(sum(result.S_qn)* delta_f);
    else
        result.S_qn=0;
        result.S_qn_rms = 0;
-       %  result.S_n
    end
```

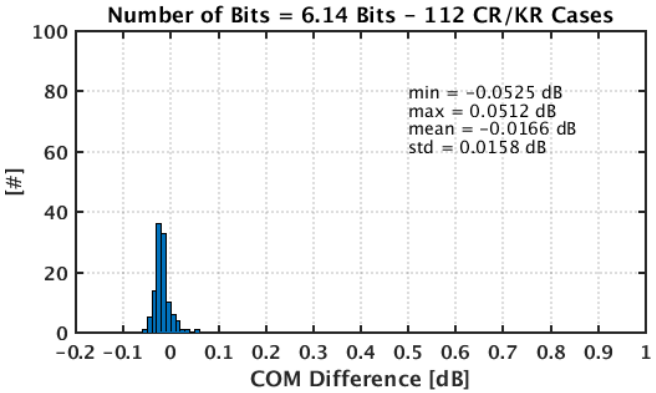
Change 2) “read_ParamConfigFile”

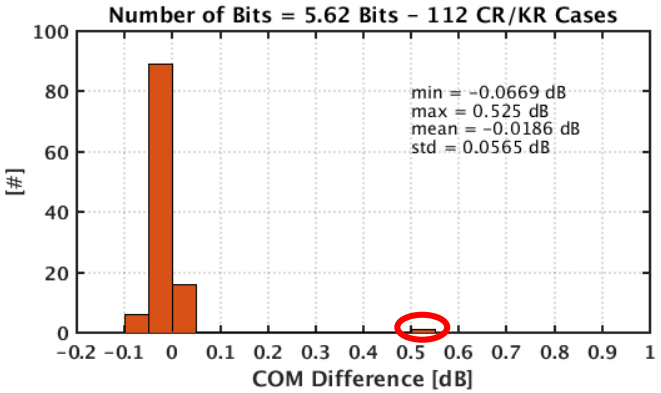
src/read_ParamConfigFile.m +1 -0

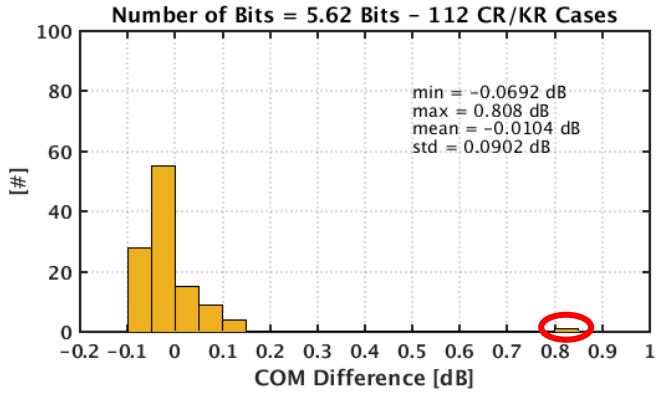
```
@@ -241,6 +241,7 @@ param.specBER = xls_parameter(parameter, 'DER_0'); % Target detector error ratio
241 241  param.DER_CDR = xls_parameter(parameter, 'DER_CDR',true,1e-2); % min DER required for a CDR
242 242  param.N_qb = xls_parameter(parameter, 'N_qb',true,0); % adc number of bits if 0 do not apply quantization
243 243  param.P_qc= xls_parameter(parameter, 'P_qc',true,2*param.specBER); % adc clipping probability
    244 + param.clip_method = xls_parameter(parameter, 'Clip Method', false, 'Fast'); % "Clip Method" parameter for Quantization Noise ('Fast' or 'Slow')
244 245  param.pass_threshold = xls_parameter(parameter, 'COM Pass threshold',false,0); % the pass fail threshold for COM in dB
245 246  param.add_rx_noise = xls_parameter(parameter, 'add_rx_noise', true, param.pass_threshold); % additional receiver noise target in dB
246 247  param.ERL_pass_threshold = xls_parameter(parameter, 'ERL Pass threshold',false,0); % the pass fail threshold for ERL in dB
```

Test Results and Final Suggestion

- After adding the switch, the same 3x112 test cases were run again and exact same COM difference between two methods was confirmed







- Runtime overheads with two fast and slow methods relative to when quantization noise is disabled demonstrated an almost 2x slower runtime for the slow method
- Fast method overhead reduced from 3% to 1% due to additional saving explained in slide 4

Average Runtime Overhead Fast Method	Average Runtime Overhead Slow Method
1% Overhead	99% Overhead

- It is suggested to proceed with the change and default the “Clip Method” switch to fast

Thank You ☺

Hossein Shakiba
Huawei Technologies Canada