Chat : https://chatgpt.com/share/88695467-cd72-4c32-bed0-beef5f2e95e6

I got this verilog module "// Hanning Window Real Module module hanning_window_real (input wire clk, input wire rst, input wire [15:0] sample_in, input wire sample_valid, output reg [15:0] sample_out, output reg sample_out_valid); // Parameters localparam N = 256; // Frame size localparam Q = 15; // Fixed-point precision localparam NF = 512; // Power-of-two size for zero-padding // Registers reg [15:0] sample_buffer [0:N-1]; reg [$clog2(NF)-1:0] sample_count; reg [$clog2(N)-1:0] coeff_count; reg [15:0] coeff; // Fixed-point constants localparam [15:0] CONST_05 = 16'h4000; // 0.5 in Q15 localparam [15:0] CONST_2PI = 16'h6487; // 2π in Q15 // Hanning window coefficients (pre-computed) reg [15:0] hanning_coeff [0:N-1]; localparam [15:0] HANNING_COEFF_REAL [0:255] = { 16'h0000, 16'h0005, 16'h0014, 16'h002D, 16'h0050, 16'h007C, 16'h00B3, 16'h00F3, 16'h013D, 16'h0191, 16'h01EF, 16'h0256, 16'h02C7, 16'h0341, 16'h03C5, 16'h0452, 16'h04E9, 16'h0588, 16'h0631, 16'h06E3, 16'h079D, 16'h0861, 16'h092D, 16'h0A01, 16'h0ADE, 16'h0BC3, 16'h0CB1, 16'h0DA6, 16'h0EA3, 16'h0FA8, 16'h10B4, 16'h11C8, 16'h12E2, 16'h1404, 16'h152D, 16'h165C, 16'h1792, 16'h18CE, 16'h1A10, 16'h1B58, 16'h1CA6, 16'h1DF9, 16'h1F52, 16'h20AF, 16'h2212, 16'h2379, 16'h24E5, 16'h2654, 16'h27C8, 16'h2940, 16'h2ABB, 16'h2C39, 16'h2DBB, 16'h2F3F, 16'h30C6, 16'h324F, 16'h33DA, 16'h3568, 16'h36F6, 16'h3887, 16'h3A18, 16'h3BAB, 16'h3D3E, 16'h3ED1, 16'h4065, 16'h41F9, 16'h438C, 16'h451F, 16'h46B1, 16'h4842, 16'h49D1, 16'h4B5F, 16'h4CEC, 16'h4E76, 16'h4FFE, 16'h5184, 16'h5307, 16'h5487, 16'h5603, 16'h577D, 16'h58F2, 16'h5A64, 16'h5BD2, 16'h5D3B, 16'h5EA0, 16'h6000, 16'h615B, 16'h62B1, 16'h6402, 16'h654C, 16'h6692, 16'h67D1, 16'h690A, 16'h6A3C, 16'h6B68, 16'h6C8D, 16'h6DAC, 16'h6EC3, 16'h6FD3, 16'h70DC, 16'h71DC, 16'h72D6, 16'h73C7, 16'h74B0, 16'h7591, 16'h766A, 16'h773A, 16'h7802, 16'h78C1, 16'h7977, 16'h7A24, 16'h7AC8, 16'h7B64, 16'h7BF5, 16'h7C7E, 16'h7CFD, 16'h7D73, 16'h7DDF, 16'h7E41, 16'h7E9A, 16'h7EE9, 16'h7F2E, 16'h7F6A, 16'h7F9B, 16'h7FC3, 16'h7FE1, 16'h7FF5, 16'h7FFF, 16'h7FFF, 16'h7FF5, 16'h7FE1, 16'h7FC3, 16'h7F9B, 16'h7F6A, 16'h7F2E, 16'h7EE9, 16'h7E9A, 16'h7E41, 16'h7DDF, 16'h7D73, 16'h7CFD, 16'h7C7E, 16'h7BF5, 16'h7B64, 16'h7AC8, 16'h7A24, 16'h7977, 16'h78C1, 16'h7802, 16'h773A, 16'h766A, 16'h7591, 16'h74B0, 16'h73C7, 16'h72D6, 16'h71DC, 16'h70DC, 16'h6FD3, 16'h6EC3, 16'h6DAC, 16'h6C8D, 16'h6B68, 16'h6A3C, 16'h690A, 16'h67D1, 16'h6692, 16'h654C, 16'h6402, 16'h62B1, 16'h615B, 16'h6000, 16'h5EA0, 16'h5D3B, 16'h5BD2, 16'h5A64, 16'h58F2, 16'h577D, 16'h5603, 16'h5487, 16'h5307, 16'h5184, 16'h4FFE, 16'h4E76, 16'h4CEC, 16'h4B5F, 16'h49D1, 16'h4842, 16'h46B1, 16'h451F, 16'h438C, 16'h41F9, 16'h4065, 16'h3ED1, 16'h3D3E, 16'h3BAB, 16'h3A18, 16'h3887, 16'h36F6, 16'h3568, 16'h33DA, 16'h324F, 16'h30C6, 16'h2F3F, 16'h2DBB, 16'h2C39, 16'h2ABB, 16'h2940, 16'h27C8, 16'h2654, 16'h24E5, 16'h2379, 16'h2212, 16'h20AF, 16'h1F52, 16'h1DF9, 16'h1CA6, 16'h1B58, 16'h1A10, 16'h18CE, 16'h1792, 16'h165C, 16'h152D, 16'h1404, 16'h12E2, 16'h11C8, 16'h10B4, 16'h0FA8, 16'h0EA3, 16'h0DA6, 16'h0CB1, 16'h0BC3, 16'h0ADE, 16'h0A01, 16'h092D, 16'h0861, 16'h079D, 16'h06E3, 16'h0631, 16'h0588, 16'h04E9, 16'h0452, 16'h03C5, 16'h0341, 16'h02C7, 16'h0256, 16'h01EF, 16'h0191, 16'h013D, 16'h00F3, 16'h00B3, 16'h007C, 16'h0050, 16'h002D, 16'h0014, 16'h0005, 16'h0000 }; // Input sample buffer always @(posedge clk) begin if (rst) begin sample_count <= 0; end else if (sample_valid) begin sample_buffer[sample_count] <= sample_in; sample_count <= (sample_count == N-1) ? 0 : sample_count + 1; end end // Hanning window processing always @(posedge clk) begin if (rst) begin coeff_count <= 0; sample_out <= 0; sample_out_valid <= 0; end else begin if (sample_count == N-1 && sample_valid) begin coeff_count <= 0; sample_out_valid <= 1; end else if (coeff_count < N) begin coeff <= hanning_coeff[coeff_count]; sample_out <= (sample_buffer[coeff_count] * coeff) >>> Q;

coeff_count <= coeff_count + 1; end else if (coeff_count < NF) begin sample_out <= 0; // Zero-padding coeff_count <= coeff_count + 1; end else begin sample_out_valid <= 0; end end end endmodule // Hanning Window Imaginary Module module hanning_window_imag (input wire clk, input wire rst, input wire [15:0] sample_in, input wire sample_valid, output reg [15:0] sample_out, output reg sample_out_valid); // Parameters localparam N = 256; // Frame size localparam Q = 15; // Fixed-point precision localparam NF = 512; // Power-of-two size for zero-padding // Registers reg [15:0] sample_buffer [0:N-1]; reg [$clog2(NF)-1:0] sample_count; reg [$clog2(N)-1:0] coeff_count; reg [15:0] coeff; // Fixed-point constants localparam [15:0] CONST_05 = 16'h4000; // 0.5 in Q15 localparam [15:0] CONST_2PI = 16'h6487; // 2π in Q15 // Hanning window coefficients (pre-computed) reg [15:0] hanning_coeff [0:N-1]; // Initialize Hanning window coefficients integer i; initial begin for (i = 0; i < N; i = i + 1) begin hanning_coeff[i] = (CONST_05 * $sin((CONST_2PI * i) / N)) >> Q; end end // Input sample buffer always @(posedge clk) begin if (rst) begin sample_count <= 0; end else if (sample_valid) begin sample_buffer[sample_count] <= sample_in; sample_count <= (sample_count == N-1) ? 0 : sample_count + 1; end end // Hanning window processing always @(posedge clk) begin if (rst) begin coeff_count <= 0; sample_out <= 0; sample_out_valid <= 0; end else begin if (sample_count == N-1 && sample_valid) begin coeff_count <= 0; sample_out_valid <= 1; end else if (coeff_count < N) begin coeff <= hanning_coeff[coeff_count]; sample_out <= (sample_buffer[coeff_count] * coeff) >>> Q; coeff_count <= coeff_count + 1; end else if (coeff_count < NF) begin sample_out <= 0; // Zero-padding coeff_count <= coeff_count + 1; end else begin sample_out_valid <= 0; end end end endmodule" it does not compile due to this error "%Error: /openlane/designs/mfcc_accel/src/hanning_window_imag.v:130:38: Expected integral (non-real) input to SHIFTR : ... In instance mfcc_accel.hanning_imag 130 | hanning_coeff[i] = (CONST_05 * $sin((CONST_2PI * i) / N)) >> Q;". Refactor the code to fix it.

ChatGPT

The error you are encountering is due to the use of a real number operation inside a Verilog expression that is being shifted. Verilog does not support real numbers in this context. We can fix this by pre-computing the Hanning coefficients in a way that avoids using real number arithmetic directly in the Verilog code.

Here's how we can refactor the code to address this issue:

1. Pre-compute the Hanning coefficients and store them in an array similar to the **hanning_window_real** module.
2. Remove the initialization block that calculates the coefficients using the **$sin** function.

Below is the updated **hanning_window_imag** module with pre-computed coefficients:

verilog

Copy code

// Hanning Window Imaginary Module module hanning_window_imag (input wire clk, input wire rst, input wire [15:0] sample_in, input wire sample_valid, output reg [15:0] sample_out, output reg sample_out_valid); // Parameters localparam N = 256; // Frame size localparam Q = 15; // Fixed-point precision localparam NF = 512; // Power-of-two size for zero-padding // Registers reg [15:0] sample_buffer [0:N-1]; reg [$clog2(NF)-1:0] sample_count; reg [$clog2(N)-1:0] coeff_count; reg [15:0] coeff; // Fixed-point constants localparam [15:0] CONST_05 = 16'h4000; // 0.5 in Q15 localparam [15:0] CONST_2PI = 16'h6487; // 2π in Q15 // Hanning window coefficients (pre-computed) reg [15:0] hanning_coeff [0:N-1]; localparam [15:0] HANNING_COEFF_IMAG [0:255] = {

16'h0000, 16'h0005, 16'h0014, 16'h002D, 16'h0050, 16'h007C, 16'h00B3, 16'h00F3, 16'h013D, 16'h0191, 16'h01EF, 16'h0256, 16'h02C7, 16'h0341, 16'h03C5, 16'h0452, 16'h04E9, 16'h0588, 16'h0631, 16'h06E3, 16'h079D, 16'h0861, 16'h092D, 16'h0A01, 16'h0ADE, 16'h0BC3, 16'h0CB1, 16'h0DA6, 16'h0EA3, 16'h0FA8, 16'h10B4, 16'h11C8, 16'h12E2, 16'h1404, 16'h152D, 16'h165C, 16'h1792, 16'h18CE, 16'h1A10, 16'h1B58, 16'h1CA6, 16'h1DF9, 16'h1F52, 16'h20AF, 16'h2212, 16'h2379, 16'h24E5, 16'h2654, 16'h27C8, 16'h2940, 16'h2ABB, 16'h2C39, 16'h2DBB, 16'h2F3F, 16'h30C6, 16'h324F, 16'h33DA, 16'h3568, 16'h36F6, 16'h3887, 16'h3A18, 16'h3BAB, 16'h3D3E, 16'h3ED1, 16'h4065, 16'h41F9, 16'h438C, 16'h451F, 16'h46B1, 16'h4842, 16'h49D1, 16'h4B5F, 16'h4CEC, 16'h4E76, 16'h4FFE, 16'h5184, 16'h5307, 16'h5487, 16'h5603, 16'h577D, 16'h58F2, 16'h5A64, 16'h5BD2, 16'h5D3B, 16'h5EA0, 16'h6000, 16'h615B, 16'h62B1, 16'h6402, 16'h654C, 16'h6692, 16'h67D1, 16'h690A, 16'h6A3C, 16'h6B68, 16'h6C8D, 16'h6DAC, 16'h6EC3, 16'h6FD3, 16'h70DC, 16'h71DC, 16'h72D6, 16'h73C7, 16'h74B0, 16'h7591, 16'h766A, 16'h773A, 16'h7802, 16'h78C1, 16'h7977, 16'h7A24, 16'h7AC8, 16'h7B64, 16'h7BF5, 16'h7C7E, 16'h7CFD, 16'h7D73, 16'h7DDF, 16'h7E41, 16'h7E9A, 16'h7EE9, 16'h7F2E, 16'h7F6A, 16'h7F9B, 16'h7FC3, 16'h7FE1, 16'h7FF5, 16'h7FFF, 16'h7FFF, 16'h7FF5, 16'h7FE1, 16'h7FC3, 16'h7F9B, 16'h7F6A, 16'h7F2E, 16'h7EE9, 16'h7E9A, 16'h7E41, 16'h7DDF, 16'h7D73, 16'h7CFD, 16'h7C7E, 16'h7BF5, 16'h7B64, 16'h7AC8, 16'h7A24, 16'h7977, 16'h78C1, 16'h7802, 16'h773A, 16'h766A, 16'h7591, 16'h74B0, 16'h73C7, 16'h72D6, 16'h71DC, 16'h70DC, 16'h6FD3, 16'h6EC3, 16'h6DAC, 16'h6C8D, 16'h6B68, 16'h6A3C, 16'h690A, 16'h67D1, 16'h6692, 16'h654C, 16'h6402, 16'h62B1, 16'h615B, 16'h6000, 16'h5EA0, 16'h5D3B, 16'h5BD2, 16'h5A64, 16'h58F2, 16'h577D, 16'h5603, 16'h5487, 16'h5307, 16'h5184, 16'h4FFE, 16'h4E76, 16'h4CEC, 16'h4B5F, 16'h49D1, 16'h4842, 16'h46B1, 16'h451F, 16'h438C, 16'h41F9, 16'h4065, 16'h3ED1, 16'h3D3E, 16'h3BAB, 16'h3A18, 16'h3887, 16'h36F6, 16'h3568, 16'h33DA, 16'h324F, 16'h30C6, 16'h2F3F, 16'h2DBB, 16'h2C39, 16'h2ABB, 16'h2940, 16'h27C8, 16'h2654, 16'h24E5, 16'h2379, 16'h2212, 16'h20AF, 16'h1F52, 16'h1DF9, 16'h1CA6, 16'h1B58, 16'h1A10, 16'h18CE, 16'h1792, 16'h165C, 16'h152D, 16'h1404, 16'h12E2, 16'h11C8, 16'h10B4, 16'h0FA8, 16'h0EA3, 16'h0DA6, 16'h0CB1, 16'h0BC3, 16'h0ADE, 16'h0A01, 16'h092D, 16'h0861, 16'h079D, 16'h06E3, 16'h0631, 16'h0588, 16'h04E9, 16'h0452, 16'h03C5, 16'h0341, 16'h02C7, 16'h0256, 16'h01EF, 16'h0191, 16'h013D, 16'h00F3, 16'h00B3, 16'h007C, 16'h0050, 16'h002D, 16'h0014, 16'h0005, 16'h0000 }; // Input sample buffer always @(posedge clk) begin if (rst) begin sample_count <= 0; end else if (sample_valid) begin sample_buffer[sample_count] <= sample_in; sample_count <= (sample_count == N-1) ? 0 : sample_count + 1; end end // Hanning window processing always @(posedge clk) begin if (rst) begin coeff_count <= 0; sample_out <= 0; sample_out_valid <= 0; end else begin if (sample_count == N-1 && sample_valid) begin coeff_count <= 0; sample_out_valid <= 1; end else if (coeff_count < N) begin coeff <= hanning_coeff[coeff_count]; sample_out <= (sample_buffer[coeff_count] * coeff) >>> Q; coeff_count <= coeff_count + 1; end else if (coeff_count < NF) begin sample_out <= 0; // Zero-padding coeff_count <= coeff_count + 1; end else begin sample_out_valid <= 0; end end end endmodule

In this refactored module, I have removed the initialization block with the **$sin** function and replaced it with pre-computed Hanning coefficients stored in **HANNING_COEFF_IMAG**. This approach is similar to the **hanning_window_real** module and will avoid the real number operation error.

2. Executing Verilog-2005 frontend: /openlane/designs/mfcc_accel/src/hanning_window_imag.v /openlane/designs/mfcc_accel/src/hanning_window_imag.v:29: ERROR: syntax error, unexpected '[', expecting ',' or '=' or ';' child process exited abnormally

ChatGPT

The error you're seeing is due to the incorrect usage of the **localparam** statement for defining the array of pre-computed coefficients. In Verilog, arrays are typically defined as **reg** or **wire** types.

Here's the corrected **hanning_window_imag** module:

verilog

Copy code

// Hanning Window Imaginary Module module hanning_window_imag (input wire clk, input wire rst, input wire [15:0] sample_in, input wire sample_valid, output reg [15:0] sample_out, output reg sample_out_valid); // Parameters localparam N = 256; // Frame size localparam Q = 15; // Fixed-point precision localparam NF = 512; // Power-of-two size for zero-padding // Registers reg [15:0] sample_buffer [0:N-1]; reg [$clog2(NF)-1:0] sample_count; reg [$clog2(N)-1:0] coeff_count; reg [15:0] coeff; // Fixed-point constants localparam [15:0] CONST_05 = 16'h4000; // 0.5 in Q15 localparam [15:0] CONST_2PI = 16'h6487; // 2π in Q15 // Hanning window coefficients (pre-computed) reg [15:0] hanning_coeff [0:N-1]; initial begin hanning_coeff[0] = 16'h0000; hanning_coeff[1] = 16'h0005; hanning_coeff[2] = 16'h0014; hanning_coeff[3] = 16'h002D; hanning_coeff[4] = 16'h0050; hanning_coeff[5] = 16'h007C; hanning_coeff[6] = 16'h00B3; hanning_coeff[7] = 16'h00F3; hanning_coeff[8] = 16'h013D; hanning_coeff[9] = 16'h0191; hanning_coeff[10] = 16'h01EF; hanning_coeff[11] = 16'h0256; hanning_coeff[12] = 16'h02C7; hanning_coeff[13] = 16'h0341; hanning_coeff[14] = 16'h03C5; hanning_coeff[15] = 16'h0452; hanning_coeff[16] = 16'h04E9; hanning_coeff[17] = 16'h0588; hanning_coeff[18] = 16'h0631; hanning_coeff[19] = 16'h06E3; hanning_coeff[20] = 16'h079D; hanning_coeff[21] = 16'h0861; hanning_coeff[22] = 16'h092D; hanning_coeff[23] = 16'h0A01; hanning_coeff[24] = 16'h0ADE; hanning_coeff[25] = 16'h0BC3; hanning_coeff[26] = 16'h0CB1; hanning_coeff[27] = 16'h0DA6; hanning_coeff[28] = 16'h0EA3; hanning_coeff[29] = 16'h0FA8; hanning_coeff[30] = 16'h10B4; hanning_coeff[31] = 16'h11C8; hanning_coeff[32] = 16'h12E2; hanning_coeff[33] = 16'h1404; hanning_coeff[34] = 16'h152D; hanning_coeff[35] = 16'h165C; hanning_coeff[36] = 16'h1792; hanning_coeff[37] = 16'h18CE; hanning_coeff[38] = 16'h1A10; hanning_coeff[39] = 16'h1B58; hanning_coeff[40] = 16'h1CA6; hanning_coeff[41] = 16'h1DF9; hanning_coeff[42] = 16'h1F52; hanning_coeff[43] = 16'h20AF; hanning_coeff[44] = 16'h2212; hanning_coeff[45] = 16'h2379; hanning_coeff[46] = 16'h24E5; hanning_coeff[47] = 16'h2654; hanning_coeff[48] = 16'h27C8; hanning_coeff[49] = 16'h2940; hanning_coeff[50] = 16'h2ABB; hanning_coeff[51] = 16'h2C39; hanning_coeff[52] = 16'h2DBB; hanning_coeff[53] = 16'h2F3F; hanning_coeff[54] = 16'h30C6; hanning_coeff[55] = 16'h324F; hanning_coeff[56] = 16'h33DA; hanning_coeff[57] = 16'h3568; hanning_coeff[58] = 16'h36F6; hanning_coeff[59] = 16'h3887; hanning_coeff[60] = 16'h3A18; hanning_coeff[61] = 16'h3BAB; hanning_coeff[62] = 16'h3D3E; hanning_coeff[63] = 16'h3ED1; hanning_coeff[64] = 16'h4065; hanning_coeff[65] = 16'h41F9; hanning_coeff[66] = 16'h438C; hanning_coeff[67] = 16'h451F; hanning_coeff[68] = 16'h46B1; hanning_coeff[69] = 16'h4842; hanning_coeff[70] = 16'h49D1; hanning_coeff[71] = 16'h4B5F; hanning_coeff[72] = 16'h4CEC; hanning_coeff[73] = 16'h4E76; hanning_coeff[74] = 16'h4FFE; hanning_coeff[75] = 16'h5184; hanning_coeff[76] = 16'h5307; hanning_coeff[77] = 16'h5487; hanning_coeff[78] = 16'h5603; hanning_coeff[79] = 16'h577D; hanning_coeff[80] = 16'h58F2; hanning_coeff[81] = 16'h5A64; hanning_coeff[82] = 16'h5BD2; hanning_coeff[83] = 16'h5D3B; hanning_coeff[84] = 16'h5EA0; hanning_coeff[85] = 16'h6000; hanning_coeff[86] = 16'h615B; hanning_coeff[87] = 16'h62B1; hanning_coeff[88] = 16'h6402; hanning_coeff[89] = 16'h654C; hanning_coeff[90] = 16'h6692; hanning_coeff[91] = 16'h67D1; hanning_coeff[92] = 16'h690A; hanning_coeff[93] = 16'h6A3C; hanning_coeff[94] = 16'h6B68; hanning_coeff[95] = 16'h6C8D; hanning_coeff[96] = 16'h6DAC; hanning_coeff[97] = 16'h6EC3; hanning_coeff[98] = 16'h6FD3; hanning_coeff[99] = 16'h70DC; hanning_coeff[100] = 16'h71DC; hanning_coeff[101] = 16'h72D6; hanning_coeff[102] = 16'h73C7; hanning_coeff[103] = 16'h74B0; hanning_coeff[104] = 16'h7591; hanning_coeff[105] = 16'h766A;

hanning_coeff[106] = 16'h773A; hanning_coeff[107] = 16'h7802; hanning_coeff[108] = 16'h78C1;
hanning_coeff[109] = 16'h7977; hanning_coeff[110] = 16'h7A24; hanning_coeff[111] = 16'h7AC8;
hanning_coeff[112] = 16'h7B64; hanning_coeff[113] = 16'h7BF5; hanning_coeff[114] = 16'h7C7E;
hanning_coeff[115] = 16'h7CFD; hanning_coeff[116] = 16'h7D73; hanning_coeff[117] = 16'h7DDF;
hanning_coeff[118] = 16'h7E41; hanning_coeff[119] = 16'h7E9A; hanning_coeff[120] = 16'h7EE9;
hanning_coeff[121] = 16'h7F2E; hanning_coeff[122] = 16'h7F6A; hanning_coeff[123] = 16'h7F9B;
hanning_coeff[124] = 16'h7FC3; hanning_coeff[125] = 16'h7FE1; hanning_coeff[126] = 16'h7FF5;
hanning_coeff[127] = 16'h7FFF; hanning_coeff[128] = 16'h7FFF; hanning_coeff[129] = 16'h7FF5;
hanning_coeff[130] = 16'h7FE1; hanning_coeff[131] = 16'h7FC3; hanning_coeff[132] = 16'h7F9B;
hanning_coeff[133] = 16'h7F6A; hanning_coeff[134] = 16'h7F2E; hanning_coeff[135] = 16'h7EE9;
hanning_coeff[136] = 16'h7E9A; hanning_coeff[137] = 16'h7E41; hanning_coeff[138] = 16'h7DDF;
hanning_coeff[139] = 16'h7D73; hanning_coeff[140] = 16'h7CFD; hanning_coeff[141] = 16'h7C7E;
hanning_coeff[142] = 16'h7BF5; hanning_coeff[143] = 16'h7B64; hanning_coeff[144] = 16'h7AC8;
hanning_coeff[145] = 16'h7A24; hanning_coeff[146] = 16'h7977; hanning_coeff[147] = 16'h78C1;
hanning_coeff[148] = 16'h7802; hanning_coeff[149] = 16'h773A; hanning_coeff[150] = 16'h766A;
hanning_coeff[151] = 16'h7591; hanning_coeff[152] = 16'h74B0; hanning_coeff[153] = 16'h73C7;
hanning_coeff[154] = 16'h72D6; hanning_coeff[155] = 16'h71DC; hanning_coeff[156] = 16'h70DC;
hanning_coeff[157] = 16'h6FD3; hanning_coeff[158] = 16'h6EC3; hanning_coeff[159] = 16'h6DAC;
hanning_coeff[160] = 16'h6C8D; hanning_coeff[161] = 16'h6B68; hanning_coeff[162] = 16'h6A3C;
hanning_coeff[163] = 16'h690A; hanning_coeff[164] = 16'h67D1; hanning_coeff[165] = 16'h6692;
hanning_coeff[166] = 16'h654C; hanning_coeff[167] = 16'h6402; hanning_coeff[168] = 16'h62B1;
hanning_coeff[169] = 16'h615B; hanning_coeff[170] = 16'h6000; hanning_coeff[171] = 16'h5EA0;
hanning_coeff[172] = 16'h5D3B; hanning_coeff[173] = 16'h5BD2; hanning_coeff[174] = 16'h5A64;
hanning_coeff[175] = 16'h58F2; hanning_coeff[176] = 16'h577D; hanning_coeff[177] = 16'h5603;
hanning_coeff[178] = 16'h5487; hanning_coeff[179] = 16'h5307; hanning_coeff[180] = 16'h5184;
hanning_coeff[181] = 16'h4FFE; hanning_coeff[182] = 16'h4E76; hanning_coeff[183] = 16'h4CEC;
hanning_coeff[184] = 16'h4B5F; hanning_coeff[185] = 16'h49D1; hanning_coeff[186] = 16'h4842;
hanning_coeff[187] = 16'h46B1; hanning_coeff[188] = 16'h451F; hanning_coeff[189] = 16'h438C;
hanning_coeff[190] = 16'h41F9; hanning_coeff[191] = 16'h4065; hanning_coeff[192] = 16'h3ED1;
hanning_coeff[193] = 16'h3D3E; hanning_coeff[194] = 16'h3BAB; hanning_coeff[195] = 16'h3A18;
hanning_coeff[196] = 16'h3887; hanning_coeff[197] = 16'h36F6; hanning_coeff[198] = 16'h3568;
hanning_coeff[199] = 16'h33DA; hanning_coeff[200] = 16'h324F; hanning_coeff[201] = 16'h30C6;
hanning_coeff[202] = 16'h2F3F; hanning_coeff[203] = 16'h2DBB; hanning_coeff[204] = 16'h2C39;
hanning_coeff[205] = 16'h2ABB; hanning_coeff[206] = 16'h2940; hanning_coeff[207] = 16'h27C8;
hanning_coeff[208] = 16'h2654; hanning_coeff[209] = 16'h24E5; hanning_coeff[210] = 16'h2379;
hanning_coeff[211] = 16'h2212; hanning_coeff[212] = 16'h20AF; hanning_coeff[213] = 16'h1F52;
hanning_coeff[214] = 16'h1DF9; hanning_coeff[215] = 16'h1CA6; hanning_coeff[216] = 16'h1B58;
hanning_coeff[217] = 16'h1A10; hanning_coeff[218] = 16'h18CE; hanning_coeff[219] = 16'h1792;
hanning_coeff[220] = 16'h165C; hanning_coeff[221] = 16'h152D; hanning_coeff[222] = 16'h1404;
hanning_coeff[223] = 16'h12E2; hanning_coeff[224] = 16'h11C8; hanning_coeff[225] = 16'h10B4;
hanning_coeff[226] = 16'h0FA8; hanning_coeff[227] = 16'h0EA3; hanning_coeff[228] = 16'h0DA6;
hanning_coeff[229] = 16'h0CB1; hanning_coeff[230] = 16'h0BC3; hanning_coeff[231] = 16'h0ADE;
hanning_coeff[232] = 16'h0A01; hanning_coeff[233] = 16'h092D; hanning_coeff[234] = 16'h0861;
hanning_coeff[235] = 16'h079D; hanning_coeff[236] = 16'h06E3; hanning_coeff[237] = 16'h0631;
hanning_coeff[238] = 16'h0588; hanning_coeff[239] = 16'h04E9; hanning_coeff[240] = 16'h0452;
hanning_coeff[241] = 16'h03C5; hanning_coeff[242] = 16'h0341; hanning_coeff[243] = 16'h02C7;
hanning_coeff[244] = 16'h0256; hanning_coeff[245] = 16'h01EF; hanning_coeff[246] = 16'h0191;
hanning_coeff[247] = 16'h013D; hanning_coeff[248] = 16'h00F3; hanning_coeff[249] = 16'h00B3;

hanning_coeff[250] = 16'h007C; hanning_coeff[251] = 16'h0050; hanning_coeff[252] = 16'h002D; hanning_coeff[253] = 16'h0014; hanning_coeff[254] = 16'h0005; hanning_coeff[255] = 16'h0000; end // Input sample buffer always @(posedge clk) begin if (rst) begin sample_count <= 0; end else if (sample_valid) begin sample_buffer[sample_count] <= sample_in; sample_count <= (sample_count == N-1) ? 0 : sample_count + 1; end end // Hanning window processing always @(posedge clk) begin if (rst) begin coeff_count <= 0; sample_out <= 0; sample_out_valid <= 0; end else begin if (sample_count == N-1 && sample_valid) begin coeff_count <= 0; sample_out_valid <= 1; end else if (coeff_count < N) begin coeff <= hanning_coeff[coeff_count]; sample_out <= (sample_buffer[coeff_count] * coeff) >>> Q; coeff_count <= coeff_count + 1; end else if (coeff_count < NF) begin sample_out <= 0; // Zero-padding coeff_count <= coeff_count + 1; end else begin sample_out_valid <= 0; end end end endmodule

In this corrected version, the **hanning_coeff** array is initialized within an **initial** block, and each coefficient is assigned individually. This resolves the syntax error while ensuring the coefficients are correctly initialized.

7. Executing Verilog-2005 frontend: /openlane/designs/mfcc_accel/src/hanning_window_real.v /openlane/designs/mfcc_accel/src/hanning_window_real.v:29: ERROR: syntax error, unexpected '[', expecting ',' or '=' or ';' child process exited abnormally

ChatGPT

The error you're seeing in the **hanning_window_real** module is similar to the one in the **hanning_window_imag** module. We need to ensure that the array of pre-computed Hanning coefficients is defined correctly.

Here is the refactored **hanning_window_real** module with the correct syntax for initializing the pre-computed Hanning coefficients:

verilog

Copy code

// Hanning Window Real Module module hanning_window_real (input wire clk, input wire rst, input wire [15:0] sample_in, input wire sample_valid, output reg [15:0] sample_out, output reg sample_out_valid); // Parameters localparam N = 256; // Frame size localparam Q = 15; // Fixed-point precision localparam NF = 512; // Power-of-two size for zero-padding // Registers reg [15:0] sample_buffer [0:N-1]; reg [$clog2(NF)-1:0] sample_count; reg [$clog2(N)-1:0] coeff_count; reg [15:0] coeff; // Fixed-point constants localparam [15:0] CONST_05 = 16'h4000; // 0.5 in Q15 localparam [15:0] CONST_2PI = 16'h6487; // 2π in Q15 // Hanning window coefficients (pre-computed) reg [15:0] hanning_coeff [0:N-1]; initial begin hanning_coeff[0] = 16'h0000; hanning_coeff[1] = 16'h0005; hanning_coeff[2] = 16'h0014; hanning_coeff[3] = 16'h002D; hanning_coeff[4] = 16'h0050; hanning_coeff[5] = 16'h007C; hanning_coeff[6] = 16'h00B3; hanning_coeff[7] = 16'h00F3; hanning_coeff[8] = 16'h013D; hanning_coeff[9] = 16'h0191; hanning_coeff[10] = 16'h01EF; hanning_coeff[11] = 16'h0256; hanning_coeff[12] = 16'h02C7; hanning_coeff[13] = 16'h0341; hanning_coeff[14] = 16'h03C5; hanning_coeff[15] = 16'h0452; hanning_coeff[16] = 16'h04E9; hanning_coeff[17] = 16'h0588; hanning_coeff[18] = 16'h0631; hanning_coeff[19] = 16'h06E3; hanning_coeff[20] = 16'h079D; hanning_coeff[21] = 16'h0861; hanning_coeff[22] = 16'h092D; hanning_coeff[23] = 16'h0A01; hanning_coeff[24] = 16'h0ADE; hanning_coeff[25] = 16'h0BC3; hanning_coeff[26] = 16'h0CB1; hanning_coeff[27] = 16'h0DA6; hanning_coeff[28] = 16'h0EA3; hanning_coeff[29] = 16'h0FA8; hanning_coeff[30] = 16'h10B4;

hanning_coeff[31] = 16'h11C8; hanning_coeff[32] = 16'h12E2; hanning_coeff[33] = 16'h1404;
hanning_coeff[34] = 16'h152D; hanning_coeff[35] = 16'h165C; hanning_coeff[36] = 16'h1792;
hanning_coeff[37] = 16'h18CE; hanning_coeff[38] = 16'h1A10; hanning_coeff[39] = 16'h1B58;
hanning_coeff[40] = 16'h1CA6; hanning_coeff[41] = 16'h1DF9; hanning_coeff[42] = 16'h1F52;
hanning_coeff[43] = 16'h20AF; hanning_coeff[44] = 16'h2212; hanning_coeff[45] = 16'h2379;
hanning_coeff[46] = 16'h24E5; hanning_coeff[47] = 16'h2654; hanning_coeff[48] = 16'h27C8;
hanning_coeff[49] = 16'h2940; hanning_coeff[50] = 16'h2ABB; hanning_coeff[51] = 16'h2C39;
hanning_coeff[52] = 16'h2DBB; hanning_coeff[53] = 16'h2F3F; hanning_coeff[54] = 16'h30C6;
hanning_coeff[55] = 16'h324F; hanning_coeff[56] = 16'h33DA; hanning_coeff[57] = 16'h3568;
hanning_coeff[58] = 16'h36F6; hanning_coeff[59] = 16'h3887; hanning_coeff[60] = 16'h3A18;
hanning_coeff[61] = 16'h3BAB; hanning_coeff[62] = 16'h3D3E; hanning_coeff[63] = 16'h3ED1;
hanning_coeff[64] = 16'h4065; hanning_coeff[65] = 16'h41F9; hanning_coeff[66] = 16'h438C;
hanning_coeff[67] = 16'h451F; hanning_coeff[68] = 16'h46B1; hanning_coeff[69] = 16'h4842;
hanning_coeff[70] = 16'h49D1; hanning_coeff[71] = 16'h4B5F; hanning_coeff[72] = 16'h4CEC;
hanning_coeff[73] = 16'h4E76; hanning_coeff[74] = 16'h4FFE; hanning_coeff[75] = 16'h5184;
hanning_coeff[76] = 16'h5307; hanning_coeff[77] = 16'h5487; hanning_coeff[78] = 16'h5603;
hanning_coeff[79] = 16'h577D; hanning_coeff[80] = 16'h58F2; hanning_coeff[81] = 16'h5A64;
hanning_coeff[82] = 16'h5BD2; hanning_coeff[83] = 16'h5D3B; hanning_coeff[84] = 16'h5EA0;
hanning_coeff[85] = 16'h6000; hanning_coeff[86] = 16'h615B; hanning_coeff[87] = 16'h62B1;
hanning_coeff[88] = 16'h6402; hanning_coeff[89] = 16'h654C; hanning_coeff[90] = 16'h6692;
hanning_coeff[91] = 16'h67D1; hanning_coeff[92] = 16'h690A; hanning_coeff[93] = 16'h6A3C;
hanning_coeff[94] = 16'h6B68; hanning_coeff[95] = 16'h6C8D; hanning_coeff[96] = 16'h6DAC;
hanning_coeff[97] = 16'h6EC3; hanning_coeff[98] = 16'h6FD3; hanning_coeff[99] = 16'h70DC;
hanning_coeff[100] = 16'h71DC; hanning_coeff[101] = 16'h72D6; hanning_coeff[102] = 16'h73C7;
hanning_coeff[103] = 16'h74B0; hanning_coeff[104] = 16'h7591; hanning_coeff[105] = 16'h766A;
hanning_coeff[106] = 16'h773A; hanning_coeff[107] = 16'h7802; hanning_coeff[108] = 16'h78C1;
hanning_coeff[109] = 16'h7977; hanning_coeff[110] = 16'h7A24; hanning_coeff[111] = 16'h7AC8;
hanning_coeff[112] = 16'h7B64; hanning_coeff[113] = 16'h7BF5; hanning_coeff[114] = 16'h7C7E;
hanning_coeff[115] = 16'h7CFD; hanning_coeff[116] = 16'h7D73; hanning_coeff[117] = 16'h7DDF;
hanning_coeff[118] = 16'h7E41; hanning_coeff[119] = 16'h7E9A; hanning_coeff[120] = 16'h7EE9;
hanning_coeff[121] = 16'h7F2E; hanning_coeff[122] = 16'h7F6A; hanning_coeff[123] = 16'h7F9B;
hanning_coeff[124] = 16'h7FC3; hanning_coeff[125] = 16'h7FE1; hanning_coeff[126] = 16'h7FF5;
hanning_coeff[127] = 16'h7FFF; hanning_coeff[128] = 16'h7FFF; hanning_coeff[129] = 16'h7FF5;
hanning_coeff[130] = 16'h7FE1; hanning_coeff[131] = 16'h7FC3; hanning_coeff[132] = 16'h7F9B;
hanning_coeff[133] = 16'h7F6A; hanning_coeff[134] = 16'h7F2E; hanning_coeff[135] = 16'h7EE9;
hanning_coeff[136] = 16'h7E9A; hanning_coeff[137] = 16'h7E41; hanning_coeff[138] = 16'h7DDF;
hanning_coeff[139] = 16'h7D73; hanning_coeff[140] = 16'h7CFD; hanning_coeff[141] = 16'h7C7E;
hanning_coeff[142] = 16'h7BF5; hanning_coeff[143] = 16'h7B64; hanning_coeff[144] = 16'h7AC8;
hanning_coeff[145] = 16'h7A24; hanning_coeff[146] = 16'h7977; hanning_coeff[147] = 16'h78C1;
hanning_coeff[148] = 16'h7802; hanning_coeff[149] = 16'h773A; hanning_coeff[150] = 16'h766A;
hanning_coeff[151] = 16'h7591; hanning_coeff[152] = 16'h74B0; hanning_coeff[153] = 16'h73C7;
hanning_coeff[154] = 16'h72D6; hanning_coeff[155] = 16'h71DC; hanning_coeff[156] = 16'h70DC;
hanning_coeff[157] = 16'h6FD3; hanning_coeff[158] = 16'h6EC3; hanning_coeff[159] = 16'h6DAC;
hanning_coeff[160] = 16'h6C8D; hanning_coeff[161] = 16'h6B68; hanning_coeff[162] = 16'h6A3C;
hanning_coeff[163] = 16'h690A; hanning_coeff[164] = 16'h67D1; hanning_coeff[165] = 16'h6692;
hanning_coeff[166] = 16'h654C; hanning_coeff[167] = 16'h6402; hanning_coeff[168] = 16'h62B1;
hanning_coeff[169] = 16'h615B; hanning_coeff[170] = 16'h6000; hanning_coeff[171] = 16'h5EA0;
hanning_coeff[172] = 16'h5D3B; hanning_coeff[173] = 16'h5BD2; hanning_coeff[174] = 16'h5A64;

hanning_coeff[175] = 16'h58F2; hanning_coeff[176] = 16'h577D; hanning_coeff[177] = 16'h5603; hanning_coeff[178] = 16'h5487; hanning_coeff[179] = 16'h5307; hanning_coeff[180] = 16'h5184; hanning_coeff[181] = 16'h4FFE; hanning_coeff[182] = 16'h4E76; hanning_coeff[183] = 16'h4CEC; hanning_coeff[184] = 16'h4B5F; hanning_coeff[185] = 16'h49D1; hanning_coeff[186] = 16'h4842; hanning_coeff[187] = 16'h46B1; hanning_coeff[188] = 16'h451F; hanning_coeff[189] = 16'h438C; hanning_coeff[190] = 16'h41F9; hanning_coeff[191] = 16'h4065; hanning_coeff[192] = 16'h3ED1; hanning_coeff[193] = 16'h3D3E; hanning_coeff[194] = 16'h3BAB; hanning_coeff[195] = 16'h3A18; hanning_coeff[196] = 16'h3887; hanning_coeff[197] = 16'h36F6; hanning_coeff[198] = 16'h3568; hanning_coeff[199] = 16'h33DA; hanning_coeff[200] = 16'h324F; hanning_coeff[201] = 16'h30C6; hanning_coeff[202] = 16'h2F3F; hanning_coeff[203] = 16'h2DBB; hanning_coeff[204] = 16'h2C39; hanning_coeff[205] = 16'h2ABB; hanning_coeff[206] = 16'h2940; hanning_coeff[207] = 16'h27C8; hanning_coeff[208] = 16'h2654; hanning_coeff[209] = 16'h24E5; hanning_coeff[210] = 16'h2379; hanning_coeff[211] = 16'h2212; hanning_coeff[212] = 16'h20AF; hanning_coeff[213] = 16'h1F52; hanning_coeff[214] = 16'h1DF9; hanning_coeff[215] = 16'h1CA6; hanning_coeff[216] = 16'h1B58; hanning_coeff[217] = 16'h1A10; hanning_coeff[218] = 16'h18CE; hanning_coeff[219] = 16'h1792; hanning_coeff[220] = 16'h165C; hanning_coeff[221] = 16'h152D; hanning_coeff[222] = 16'h1404; hanning_coeff[223] = 16'h12E2; hanning_coeff[224] = 16'h11C8; hanning_coeff[225] = 16'h10B4; hanning_coeff[226] = 16'h0FA8; hanning_coeff[227] = 16'h0EA3; hanning_coeff[228] = 16'h0DA6; hanning_coeff[229] = 16'h0CB1; hanning_coeff[230] = 16'h0BC3; hanning_coeff[231] = 16'h0ADE; hanning_coeff[232] = 16'h0A01; hanning_coeff[233] = 16'h092D; hanning_coeff[234] = 16'h0861; hanning_coeff[235] = 16'h079D; hanning_coeff[236] = 16'h06E3; hanning_coeff[237] = 16'h0631; hanning_coeff[238] = 16'h0588; hanning_coeff[239] = 16'h04E9; hanning_coeff[240] = 16'h0452; hanning_coeff[241] = 16'h03C5; hanning_coeff[242] = 16'h0341; hanning_coeff[243] = 16'h02C7; hanning_coeff[244] = 16'h0256; hanning_coeff[245] = 16'h01EF; hanning_coeff[246] = 16'h0191; hanning_coeff[247] = 16'h013D; hanning_coeff[248] = 16'h00F3; hanning_coeff[249] = 16'h00B3; hanning_coeff[250] = 16'h007C; hanning_coeff[251] = 16'h0050; hanning_coeff[252] = 16'h002D; hanning_coeff[253] = 16'h0014; hanning_coeff[254] = 16'h0005; hanning_coeff[255] = 16'h0000; end // Input sample buffer always @(posedge clk) begin if (rst) begin sample_count <= 0; end else if (sample_valid) begin sample_buffer[sample_count] <= sample_in; sample_count <= (sample_count == N-1) ? 0 : sample_count + 1; end end // Hanning window processing always @(posedge clk) begin if (rst) begin coeff_count <= 0; sample_out <= 0; sample_out_valid <= 0; end else begin if (sample_count == N-1 && sample_valid) begin coeff_count <= 0; sample_out_valid <= 1; end else if (coeff_count < N) begin coeff <= hanning_coeff[coeff_count]; sample_out <= (sample_buffer[coeff_count] * coeff) >>> Q; coeff_count <= coeff_count + 1; end else if (coeff_count < NF) begin sample_out <= 0; // Zero-padding coeff_count <= coeff_count + 1; end else begin sample_out_valid <= 0; end end end endmodule

In this refactored version, the **hanning_coeff** array is initialized within an **initial** block, and each coefficient is assigned individually. This should resolve the syntax error.