VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING

Winter Semester 2024-2025

BECE301P – Digital Signal Processing LAB

L47 +L48

FACULTY: SUDHAKAR M S

Task -3

Realization of OFDM waveforms For the given OFDM system generate the waveforms corresponding to each block.

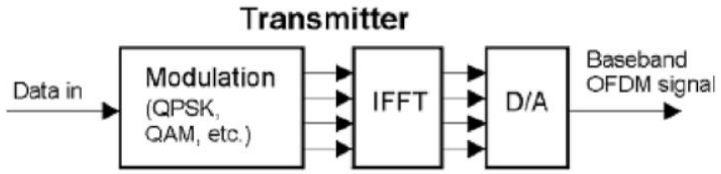
DONE BY

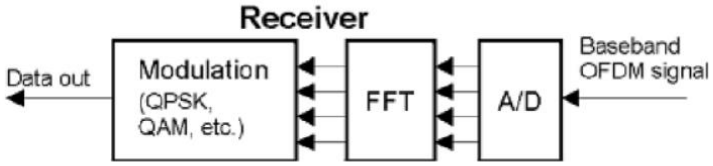
KAUSHIK KUMAR PS
23BEC0142

Task 3

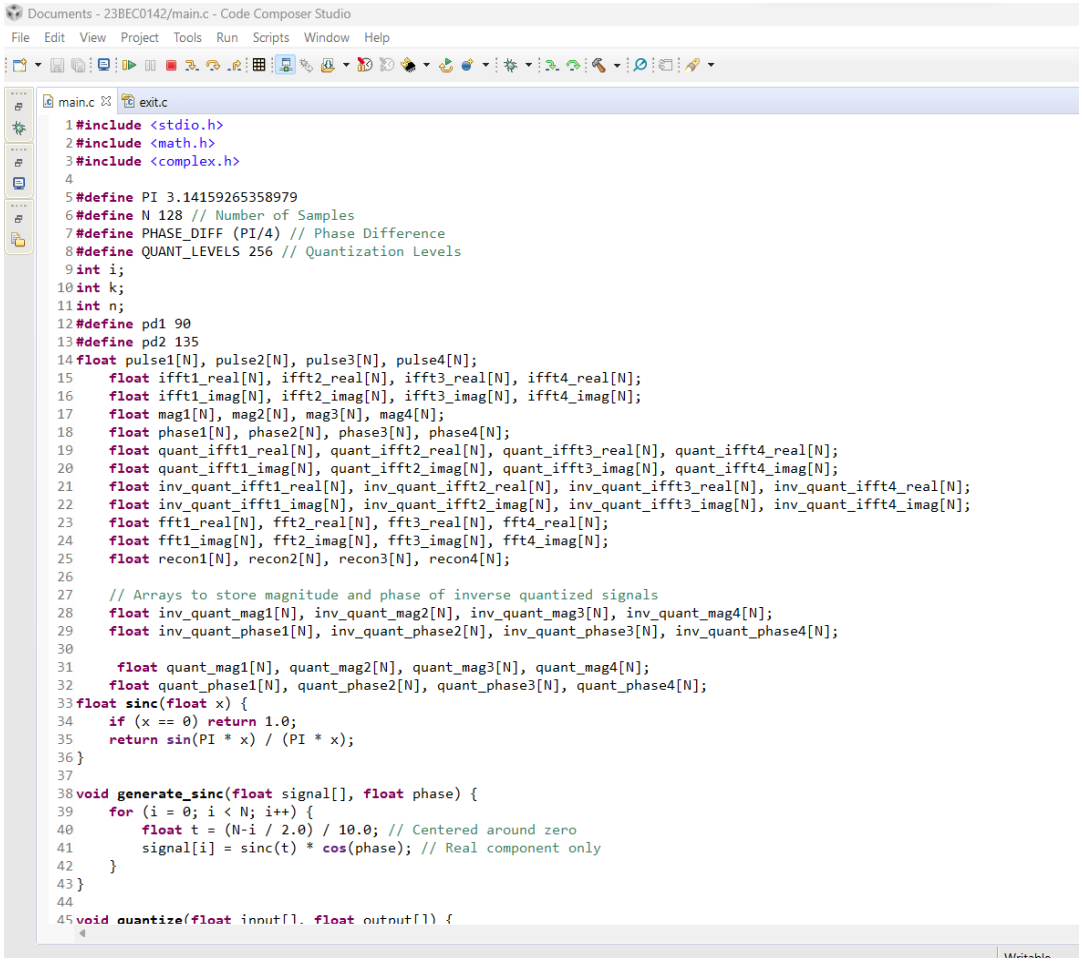
Realization of OFDM waveforms

For the given OFDM system generate the waveforms corresponding to each block.





CODE:

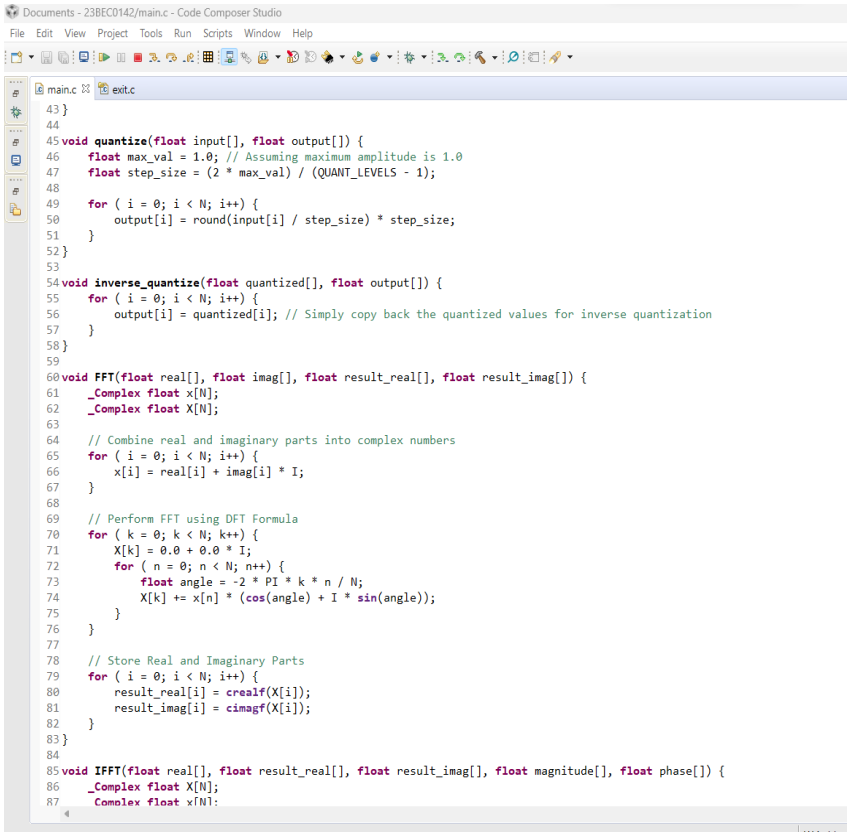


Documents - 23BEC0142/main.c - Code Composer Studio

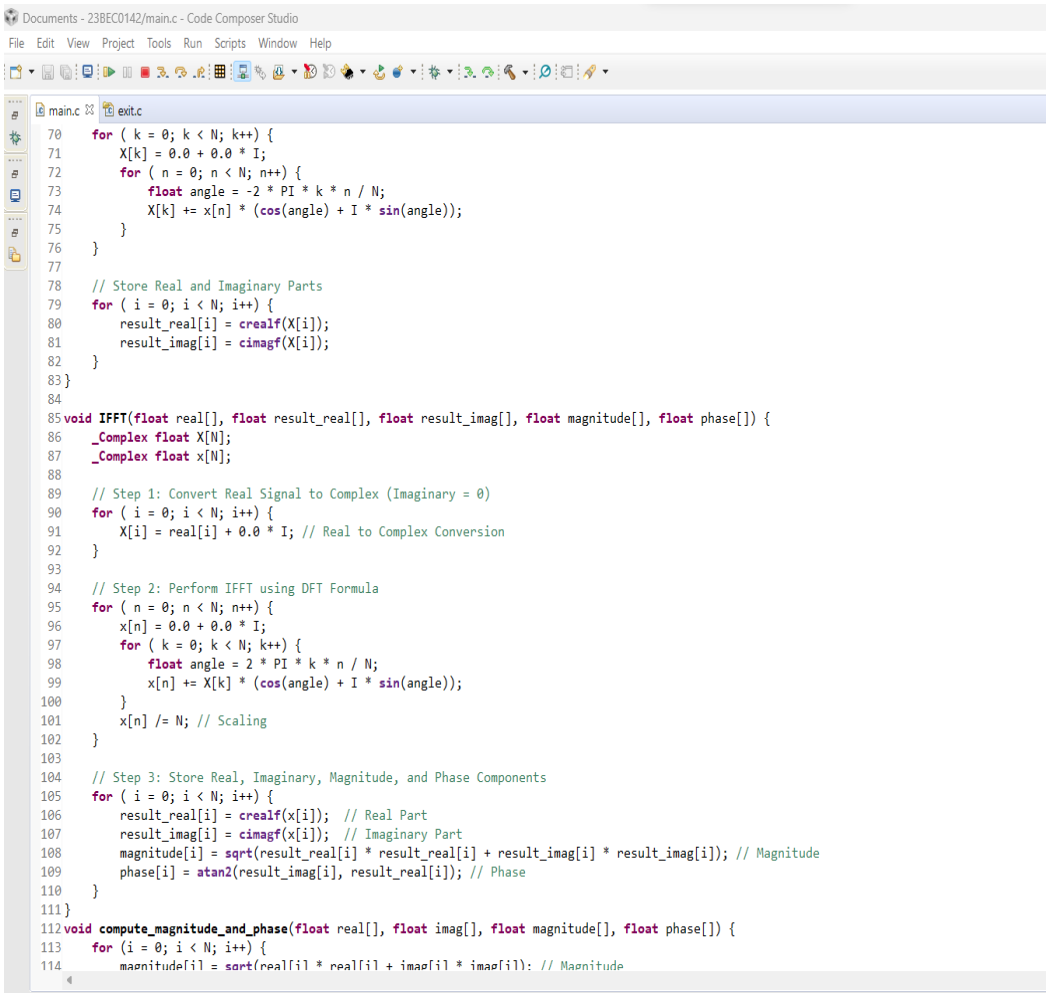
File Edit View Project Tools Run Scripts Window Help

main.c exit.c

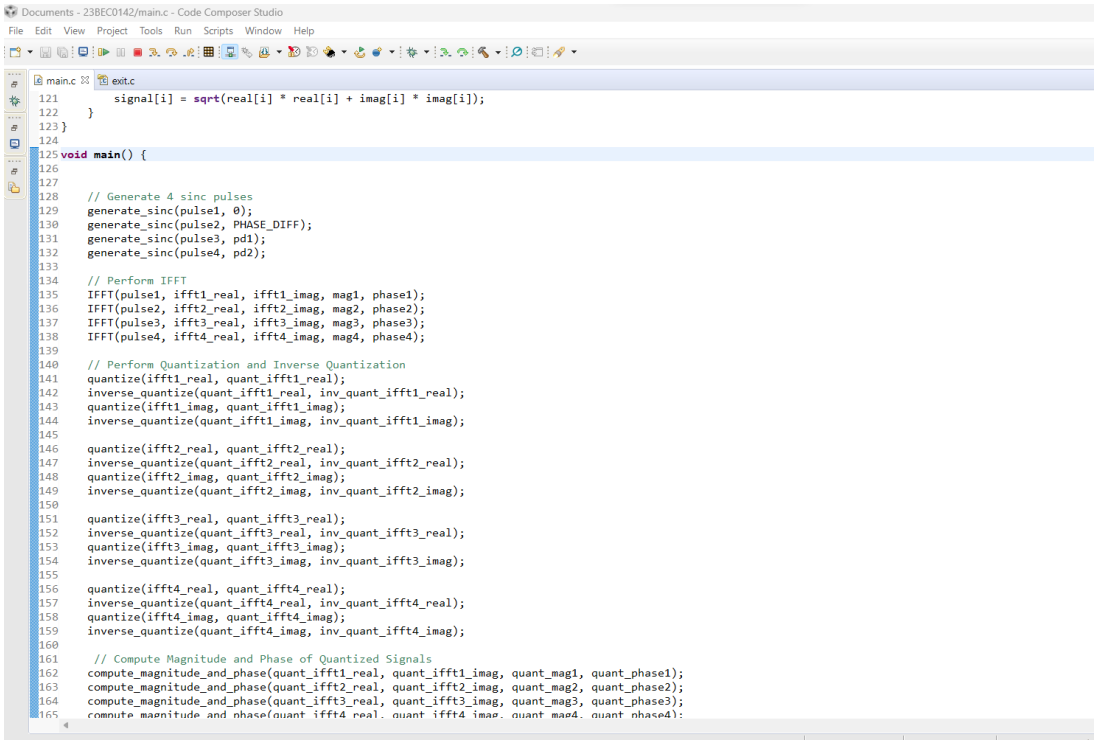
```c
#include <stdio.h>
#include <math.h>
#include <complex.h>

#define PI 3.14159265358979
#define N 128 // Number of Samples
#define PHASE_DIFF (PI/4) // Phase Difference
#define QUANT_LEVELS 256 // Quantization Levels
int i;
int k;
int n;
#define pd1 90
#define pd2 135
float pulse1[N], pulse2[N], pulse3[N], pulse4[N];
    float ifft1_real[N], ifft2_real[N], ifft3_real[N], ifft4_real[N];
    float ifft1_imag[N], ifft2_imag[N], ifft3_imag[N], ifft4_imag[N];
    float mag1[N], mag2[N], mag3[N], mag4[N];
    float phase1[N], phase2[N], phase3[N], phase4[N];
    float quant_ifft1_real[N], quant_ifft2_real[N], quant_ifft3_real[N], quant_ifft4_real[N];
    float quant_ifft1_imag[N], quant_ifft2_imag[N], quant_ifft3_imag[N], quant_ifft4_imag[N];
    float inv_quant_ifft1_real[N], inv_quant_ifft2_real[N], inv_quant_ifft3_real[N], inv_quant_ifft4_real[N];
    float inv_quant_ifft1_imag[N], inv_quant_ifft2_imag[N], inv_quant_ifft3_imag[N], inv_quant_ifft4_imag[N];
    float fft1_real[N], fft2_real[N], fft3_real[N], fft4_real[N];
    float fft1_imag[N], fft2_imag[N], fft3_imag[N], fft4_imag[N];
    float recon1[N], recon2[N], recon3[N], recon4[N];

    // Arrays to store magnitude and phase of inverse quantized signals
    float inv_quant_mag1[N], inv_quant_mag2[N], inv_quant_mag3[N], inv_quant_mag4[N];
    float inv_quant_phase1[N], inv_quant_phase2[N], inv_quant_phase3[N], inv_quant_phase4[N];

     float quant_mag1[N], quant_mag2[N], quant_mag3[N], quant_mag4[N];
    float quant_phase1[N], quant_phase2[N], quant_phase3[N], quant_phase4[N];
float sinc(float x) {
    if (x == 0) return 1.0;
    return sin(PI * x) / (PI * x);
}

void generate_sinc(float signal[], float phase) {
    for (i = 0; i < N; i++) {
        float t = (N-i / 2.0) / 10.0; // Centered around zero
        signal[i] = sinc(t) * cos(phase); // Real component only
    }
}

void quantize(float input[], float output[]) {
```

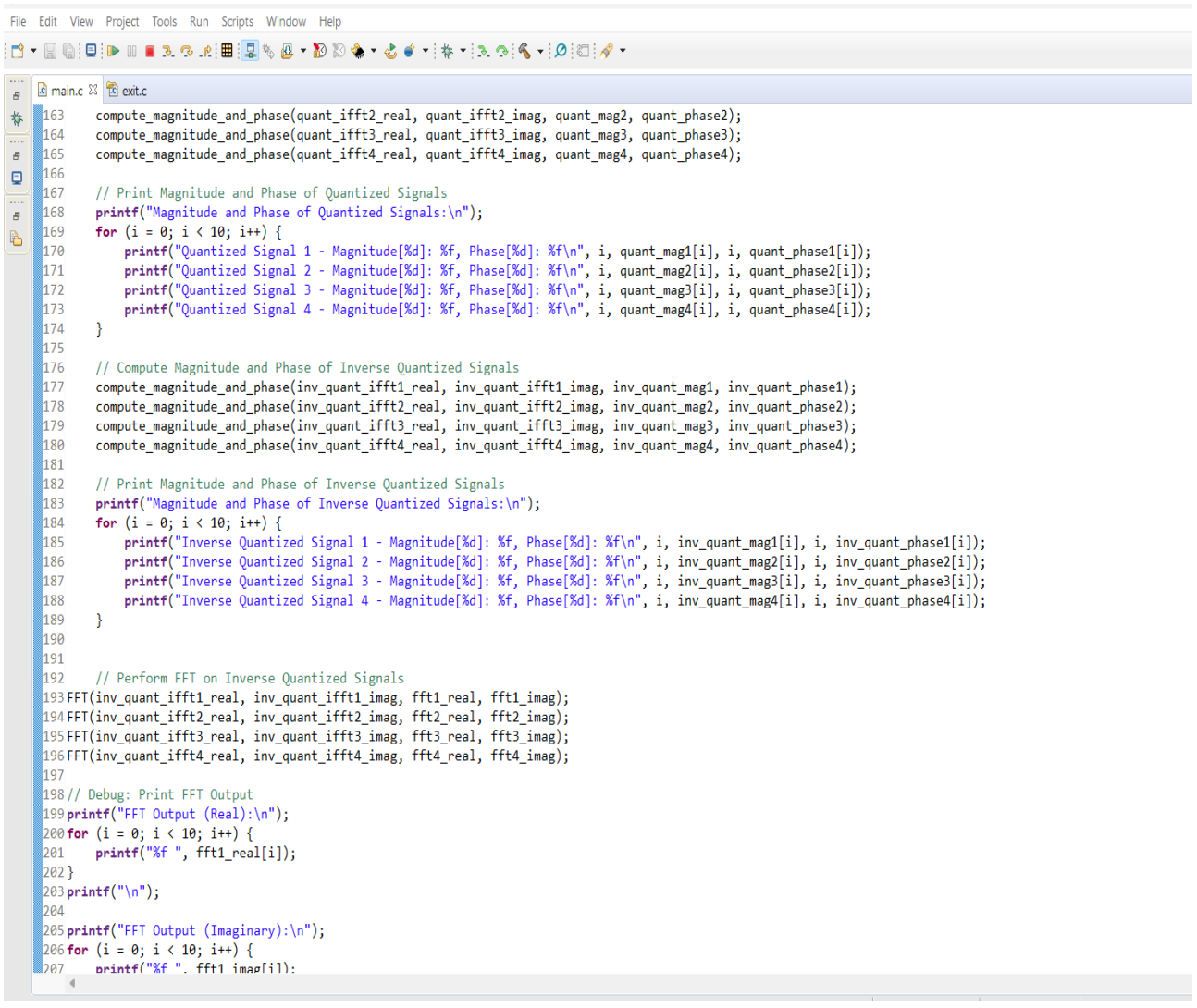



```
}

void quantize(float input[], float output[]) {
    float max_val = 1.0; // Assuming maximum amplitude is 1.0
    float step_size = (2 * max_val) / (QUANT_LEVELS - 1);

    for ( i = 0; i < N; i++) {
        output[i] = round(input[i] / step_size) * step_size;
    }
}

void inverse_quantize(float quantized[], float output[]) {
    for ( i = 0; i < N; i++) {
        output[i] = quantized[i]; // Simply copy back the quantized values for inverse quantization
    }
}

void FFT(float real[], float imag[], float result_real[], float result_imag[]) {
    _Complex float x[N];
    _Complex float X[N];

    // Combine real and imaginary parts into complex numbers
    for ( i = 0; i < N; i++) {
        x[i] = real[i] + imag[i] * I;
    }

    // Perform FFT using DFT Formula
    for ( k = 0; k < N; k++) {
        X[k] = 0.0 + 0.0 * I;
        for ( n = 0; n < N; n++) {
            float angle = -2 * PI * k * n / N;
            X[k] += x[n] * (cos(angle) + I * sin(angle));
        }
    }

    // Store Real and Imaginary Parts
    for ( i = 0; i < N; i++) {
        result_real[i] = crealf(X[i]);
        result_imag[i] = cimagf(X[i]);
    }
}

void IFFT(float real[], float result_real[], float result_imag[], float magnitude[], float phase[]) {
    _Complex float X[N];
    _Complex float x[N];
```




```
    for ( k = 0; k < N; k++) {
        X[k] = 0.0 + 0.0 * I;
        for ( n = 0; n < N; n++) {
            float angle = -2 * PI * k * n / N;
            X[k] += x[n] * (cos(angle) + I * sin(angle));
        }
    }

    // Store Real and Imaginary Parts
    for ( i = 0; i < N; i++) {
        result_real[i] = crealf(X[i]);
        result_imag[i] = cimagf(X[i]);
    }
}

void IFFT(float real[], float result_real[], float result_imag[], float magnitude[], float phase[]) {
    _Complex float X[N];
    _Complex float x[N];

    // Step 1: Convert Real Signal to Complex (Imaginary = 0)
    for ( i = 0; i < N; i++) {
        X[i] = real[i] + 0.0 * I; // Real to Complex Conversion
    }

    // Step 2: Perform IFFT using DFT Formula
    for ( n = 0; n < N; n++) {
        x[n] = 0.0 + 0.0 * I;
        for ( k = 0; k < N; k++) {
            float angle = 2 * PI * k * n / N;
            x[n] += X[k] * (cos(angle) + I * sin(angle));
        }
        x[n] /= N; // Scaling
    }

    // Step 3: Store Real, Imaginary, Magnitude, and Phase Components
    for ( i = 0; i < N; i++) {
        result_real[i] = crealf(x[i]);  // Real Part
        result_imag[i] = cimagf(x[i]);  // Imaginary Part
        magnitude[i] = sqrt(result_real[i] * result_real[i] + result_imag[i] * result_imag[i]); // Magnitude
        phase[i] = atan2(result_imag[i], result_real[i]); // Phase
    }
}
void compute_magnitude_and_phase(float real[], float imag[], float magnitude[], float phase[]) {
    for (i = 0; i < N; i++) {
        magnitude[i] = sqrt(real[i] * real[i] + imag[i] * imag[i]); // Magnitude
```


Documents - 23BEC0142/main.c - Code Composer Studio
File Edit View Project Tools Run Scripts Window Help
main.c
exit.c
signal[i] = sqrt(real[i] * real[i] + imag[i] * imag[i]);
}
}
void main() {
// Generate 4 sinc pulses
generate_sinc(pulse1, 0);
generate_sinc(pulse2, PHASE_DIFF);
generate_sinc(pulse3, pd1);
generate_sinc(pulse4, pd2);
// Perform IFFT
IFFT(pulse1, ifft1_real, ifft1_imag, mag1, phase1);
IFFT(pulse2, ifft2_real, ifft2_imag, mag2, phase2);
IFFT(pulse3, ifft3_real, ifft3_imag, mag3, phase3);
IFFT(pulse4, ifft4_real, ifft4_imag, mag4, phase4);
// Perform Quantization and Inverse Quantization
quantize(ifft1_real, quant_ifft1_real);
inverse_quantize(quant_ifft1_real, inv_quant_ifft1_real);
quantize(ifft1_imag, quant_ifft1_imag);
inverse_quantize(quant_ifft1_imag, inv_quant_ifft1_imag);
quantize(ifft2_real, quant_ifft2_real);
inverse_quantize(quant_ifft2_real, inv_quant_ifft2_real);
quantize(ifft2_imag, quant_ifft2_imag);
inverse_quantize(quant_ifft2_imag, inv_quant_ifft2_imag);
quantize(ifft3_real, quant_ifft3_real);
inverse_quantize(quant_ifft3_real, inv_quant_ifft3_real);
quantize(ifft3_imag, quant_ifft3_imag);
inverse_quantize(quant_ifft3_imag, inv_quant_ifft3_imag);
quantize(ifft4_real, quant_ifft4_real);
inverse_quantize(quant_ifft4_real, inv_quant_ifft4_real);
quantize(ifft4_imag, quant_ifft4_imag);
inverse_quantize(quant_ifft4_imag, inv_quant_ifft4_imag);
// Compute Magnitude and Phase of Quantized Signals
compute_magnitude_and_phase(quant_ifft1_real, quant_ifft1_imag, quant_mag1, quant_phase1);
compute_magnitude_and_phase(quant_ifft2_real, quant_ifft2_imag, quant_mag2, quant_phase2);
compute_magnitude_and_phase(quant_ifft3_real, quant_ifft3_imag, quant_mag3, quant_phase3);


File Edit View Project Tools Run Scripts Window Help
main.c
exit.c
compute_magnitude_and_phase(quant_ifft2_real, quant_ifft2_imag, quant_mag2, quant_phase2);
compute_magnitude_and_phase(quant_ifft3_real, quant_ifft3_imag, quant_mag3, quant_phase3);
compute_magnitude_and_phase(quant_ifft4_real, quant_ifft4_imag, quant_mag4, quant_phase4);
// Print Magnitude and Phase of Quantized Signals
printf("Magnitude and Phase of Quantized Signals:\n");
for (i = 0; i < 10; i++) {
printf("Quantized Signal 1 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, quant_mag1[i], i, quant_phase1[i]);
printf("Quantized Signal 2 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, quant_mag2[i], i, quant_phase2[i]);
printf("Quantized Signal 3 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, quant_mag3[i], i, quant_phase3[i]);
printf("Quantized Signal 4 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, quant_mag4[i], i, quant_phase4[i]);
}
// Compute Magnitude and Phase of Inverse Quantized Signals
compute_magnitude_and_phase(inv_quant_ifft1_real, inv_quant_ifft1_imag, inv_quant_mag1, inv_quant_phase1);
compute_magnitude_and_phase(inv_quant_ifft2_real, inv_quant_ifft2_imag, inv_quant_mag2, inv_quant_phase2);
compute_magnitude_and_phase(inv_quant_ifft3_real, inv_quant_ifft3_imag, inv_quant_mag3, inv_quant_phase3);
compute_magnitude_and_phase(inv_quant_ifft4_real, inv_quant_ifft4_imag, inv_quant_mag4, inv_quant_phase4);
// Print Magnitude and Phase of Inverse Quantized Signals
printf("Magnitude and Phase of Inverse Quantized Signals:\n");
for (i = 0; i < 10; i++) {
printf("Inverse Quantized Signal 1 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, inv_quant_mag1[i], i, inv_quant_phase1[i]);
printf("Inverse Quantized Signal 2 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, inv_quant_mag2[i], i, inv_quant_phase2[i]);
printf("Inverse Quantized Signal 3 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, inv_quant_mag3[i], i, inv_quant_phase3[i]);
printf("Inverse Quantized Signal 4 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, inv_quant_mag4[i], i, inv_quant_phase4[i]);
}
// Perform FFT on Inverse Quantized Signals
FFT(inv_quant_ifft1_real, inv_quant_ifft1_imag, fft1_real, fft1_imag);
FFT(inv_quant_ifft2_real, inv_quant_ifft2_imag, fft2_real, fft2_imag);
FFT(inv_quant_ifft3_real, inv_quant_ifft3_imag, fft3_real, fft3_imag);
FFT(inv_quant_ifft4_real, inv_quant_ifft4_imag, fft4_real, fft4_imag);
// Debug: Print FFT Output
printf("FFT Output (Real):\n");
for (i = 0; i < 10; i++) {
printf("%f ", fft1_real[i]);
}
printf("\n");
printf("FFT Output (Imaginary):\n");
for (i = 0; i < 10; i++) {

```
        printf("Inverse Quantized Signal 4 - Magnitude[%d]: %f, Phase[%d]: %f\n", i, inv_quant_mag4[i], i, inv_quant_phase4[i]);
    }


    // Perform FFT on Inverse Quantized Signals
FFT(inv_quant_ifft1_real, inv_quant_ifft1_imag, fft1_real, fft1_imag);
FFT(inv_quant_ifft2_real, inv_quant_ifft2_imag, fft2_real, fft2_imag);
FFT(inv_quant_ifft3_real, inv_quant_ifft3_imag, fft3_real, fft3_imag);
FFT(inv_quant_ifft4_real, inv_quant_ifft4_imag, fft4_real, fft4_imag);

// Debug: Print FFT Output
printf("FFT Output (Real):\n");
for (i = 0; i < 10; i++) {
    printf("%f ", fft1_real[i]);
}
printf("\n");

printf("FFT Output (Imaginary):\n");
for (i = 0; i < 10; i++) {
    printf("%f ", fft1_imag[i]);
}
printf("\n");

// Reconstruct Signals
reconstruct_signal(fft1_real, fft1_imag, recon1);
reconstruct_signal(fft2_real, fft2_imag, recon2);
reconstruct_signal(fft3_real, fft3_imag, recon3);
reconstruct_signal(fft4_real, fft4_imag, recon4);

// Debug: Print Reconstructed Signals
printf("Reconstructed Signals:\n");
for (i = 0; i < 10; i++) {
    printf("Reconstructed Signal1[%d]: %f\n", i, recon1[i]);
    printf("Reconstructed Signal2[%d]: %f\n", i, recon2[i]);
    printf("Reconstructed Signal3[%d]: %f\n", i, recon3[i]);
    printf("Reconstructed Signal4[%d]: %f\n", i, recon4[i]);
}
}
```

OUTPUT:

Console

23BEC0142:CIO

```
[C674X_0] Magnitude and Phase of Quantized Signals:
Quantized Signal 1 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Quantized Signal 2 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Quantized Signal 3 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Quantized Signal 4 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Quantized Signal 1 - Magnitude[1]: 0.000000, Phase[1]: 0.000000
Quantized Signal 2 - Magnitude[1]: 0.000000, Phase[1]: 0.000000
Quantized Signal 3 - Magnitude[1]: 0.000000, Phase[1]: 0.000000
Quantized Signal 4 - Magnitude[1]: 0.000000, Phase[1]: 0.000000
Quantized Signal 1 - Magnitude[2]: 0.000000, Phase[2]: 0.000000
Quantized Signal 2 - Magnitude[2]: 0.000000, Phase[2]: 0.000000
Quantized Signal 3 - Magnitude[2]: 0.000000, Phase[2]: 0.000000
Quantized Signal 4 - Magnitude[2]: 0.000000, Phase[2]: 0.000000
Quantized Signal 1 - Magnitude[3]: 0.015686, Phase[3]: 0.000000
Quantized Signal 2 - Magnitude[3]: 0.007843, Phase[3]: 0.000000
Quantized Signal 3 - Magnitude[3]: 0.007843, Phase[3]: 3.141593
Quantized Signal 4 - Magnitude[3]: 0.015686, Phase[3]: 3.141593
Quantized Signal 1 - Magnitude[4]: 0.000000, Phase[4]: 0.000000
Quantized Signal 2 - Magnitude[4]: 0.000000, Phase[4]: 0.000000
Quantized Signal 3 - Magnitude[4]: 0.000000, Phase[4]: 0.000000
Quantized Signal 4 - Magnitude[4]: 0.000000, Phase[4]: 0.000000
Quantized Signal 1 - Magnitude[5]: 0.000000, Phase[5]: 0.000000
Quantized Signal 2 - Magnitude[5]: 0.000000, Phase[5]: 0.000000
Quantized Signal 3 - Magnitude[5]: 0.000000, Phase[5]: 0.000000
Quantized Signal 4 - Magnitude[5]: 0.000000, Phase[5]: 0.000000
Quantized Signal 1 - Magnitude[6]: 0.000000, Phase[6]: 0.000000
Quantized Signal 2 - Magnitude[6]: 0.000000, Phase[6]: 0.000000
Quantized Signal 3 - Magnitude[6]: 0.000000, Phase[6]: 0.000000
Quantized Signal 4 - Magnitude[6]: 0.000000, Phase[6]: 0.000000
Quantized Signal 1 - Magnitude[7]: 0.000000, Phase[7]: 0.000000
Quantized Signal 2 - Magnitude[7]: 0.000000, Phase[7]: 0.000000
Quantized Signal 3 - Magnitude[7]: 0.000000, Phase[7]: 0.000000
Quantized Signal 4 - Magnitude[7]: 0.000000, Phase[7]: 0.000000
Quantized Signal 1 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Quantized Signal 2 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Quantized Signal 3 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Quantized Signal 4 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Quantized Signal 1 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Quantized Signal 2 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Quantized Signal 3 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Quantized Signal 4 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Magnitude and Phase of Inverse Quantized Signals:
Inverse Quantized Signal 1 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Inverse Quantized Signal 2 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
```

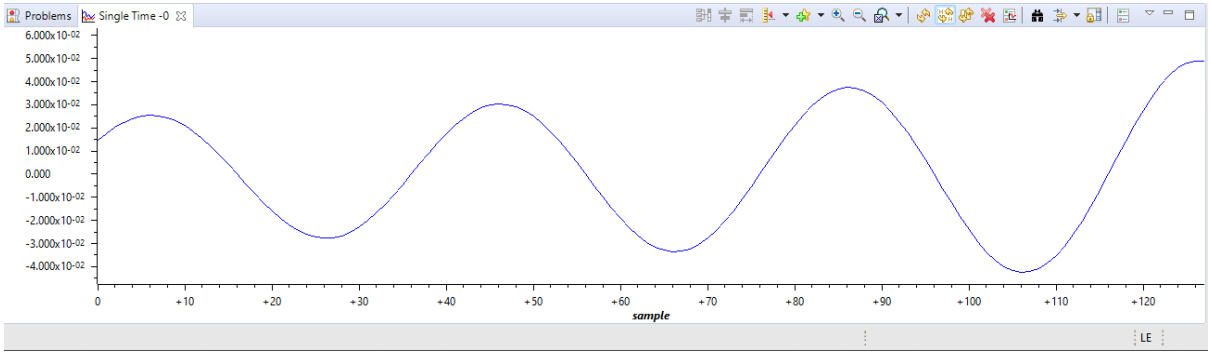
```
23BEC0142:CIO
Quantized Signal 4 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Quantized Signal 1 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Quantized Signal 2 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Quantized Signal 3 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Quantized Signal 4 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Magnitude and Phase of Inverse Quantized Signals:
Inverse Quantized Signal 1 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Inverse Quantized Signal 2 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Inverse Quantized Signal 3 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Inverse Quantized Signal 4 - Magnitude[0]: 0.000000, Phase[0]: 0.000000
Inverse Quantized Signal 1 - Magnitude[1]: 0.000000, Phase[1]: 0.000000
Inverse Quantized Signal 2 - Magnitude[1]: 0.000000, Phase[1]: 0.000000
Inverse Quantized Signal 3 - Magnitude[1]: 0.000000, Phase[1]: 0.000000
Inverse Quantized Signal 4 - Magnitude[1]: 0.000000, Phase[1]: 0.000000
Inverse Quantized Signal 1 - Magnitude[2]: 0.000000, Phase[2]: 0.000000
Inverse Quantized Signal 2 - Magnitude[2]: 0.000000, Phase[2]: 0.000000
Inverse Quantized Signal 3 - Magnitude[2]: 0.000000, Phase[2]: 0.000000
Inverse Quantized Signal 4 - Magnitude[2]: 0.000000, Phase[2]: 0.000000
Inverse Quantized Signal 1 - Magnitude[3]: 0.015686, Phase[3]: 0.000000
Inverse Quantized Signal 2 - Magnitude[3]: 0.007843, Phase[3]: 0.000000
Inverse Quantized Signal 3 - Magnitude[3]: 0.007843, Phase[3]: 3.141593
Inverse Quantized Signal 4 - Magnitude[3]: 0.015686, Phase[3]: 3.141593
Inverse Quantized Signal 1 - Magnitude[4]: 0.000000, Phase[4]: 0.000000
Inverse Quantized Signal 2 - Magnitude[4]: 0.000000, Phase[4]: 0.000000
Inverse Quantized Signal 3 - Magnitude[4]: 0.000000, Phase[4]: 0.000000
Inverse Quantized Signal 4 - Magnitude[4]: 0.000000, Phase[4]: 0.000000
Inverse Quantized Signal 1 - Magnitude[5]: 0.000000, Phase[5]: 0.000000
Inverse Quantized Signal 2 - Magnitude[5]: 0.000000, Phase[5]: 0.000000
Inverse Quantized Signal 3 - Magnitude[5]: 0.000000, Phase[5]: 0.000000
Inverse Quantized Signal 4 - Magnitude[5]: 0.000000, Phase[5]: 0.000000
Inverse Quantized Signal 1 - Magnitude[6]: 0.000000, Phase[6]: 0.000000
Inverse Quantized Signal 2 - Magnitude[6]: 0.000000, Phase[6]: 0.000000
Inverse Quantized Signal 3 - Magnitude[6]: 0.000000, Phase[6]: 0.000000
Inverse Quantized Signal 4 - Magnitude[6]: 0.000000, Phase[6]: 0.000000
Inverse Quantized Signal 1 - Magnitude[7]: 0.000000, Phase[7]: 0.000000
Inverse Quantized Signal 2 - Magnitude[7]: 0.000000, Phase[7]: 0.000000
Inverse Quantized Signal 3 - Magnitude[7]: 0.000000, Phase[7]: 0.000000
Inverse Quantized Signal 4 - Magnitude[7]: 0.000000, Phase[7]: 0.000000
Inverse Quantized Signal 1 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Inverse Quantized Signal 2 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Inverse Quantized Signal 3 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Inverse Quantized Signal 4 - Magnitude[8]: 0.000000, Phase[8]: 0.000000
Inverse Quantized Signal 1 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Inverse Quantized Signal 2 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
```

```
23BEC0142:CIO
Inverse Quantized Signal 1 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Inverse Quantized Signal 2 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Inverse Quantized Signal 3 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
Inverse Quantized Signal 4 - Magnitude[9]: 0.000000, Phase[9]: 0.000000
FFT Output (Real):
0.031373 0.031033 0.030022 0.028360 0.026085 0.023246 0.019903 0.016129 0.012006 0.007623
FFT Output (Imaginary):
0.000000 -0.000000 -0.000000 -0.000000 -0.000000 0.000000 -0.000000 0.000000 -0.000000 -0.000000
Reconstructed Signals:
Reconstructed Signal1[0]: 0.031373
Reconstructed Signal2[0]: 0.015686
Reconstructed Signal3[0]: 0.015686
Reconstructed Signal4[0]: 0.031373
Reconstructed Signal1[1]: 0.031033
Reconstructed Signal2[1]: 0.015516
Reconstructed Signal3[1]: 0.015516
Reconstructed Signal4[1]: 0.031033
Reconstructed Signal1[2]: 0.030022
Reconstructed Signal2[2]: 0.015011
Reconstructed Signal3[2]: 0.015011
Reconstructed Signal4[2]: 0.030022
Reconstructed Signal1[3]: 0.028360
Reconstructed Signal2[3]: 0.014180
Reconstructed Signal3[3]: 0.014180
Reconstructed Signal4[3]: 0.028360
Reconstructed Signal1[4]: 0.026085
Reconstructed Signal2[4]: 0.013043
Reconstructed Signal3[4]: 0.013043
Reconstructed Signal4[4]: 0.026085
Reconstructed Signal1[5]: 0.023246
Reconstructed Signal2[5]: 0.011623
Reconstructed Signal3[5]: 0.011623
Reconstructed Signal4[5]: 0.023246
Reconstructed Signal1[6]: 0.019903
Reconstructed Signal2[6]: 0.009951
Reconstructed Signal3[6]: 0.009951
Reconstructed Signal4[6]: 0.019903
Reconstructed Signal1[7]: 0.016129
Reconstructed Signal2[7]: 0.008064
Reconstructed Signal3[7]: 0.008064
Reconstructed Signal4[7]: 0.016129
Reconstructed Signal1[8]: 0.012006
Reconstructed Signal2[8]: 0.006003
Reconstructed Signal3[8]: 0.006003
```

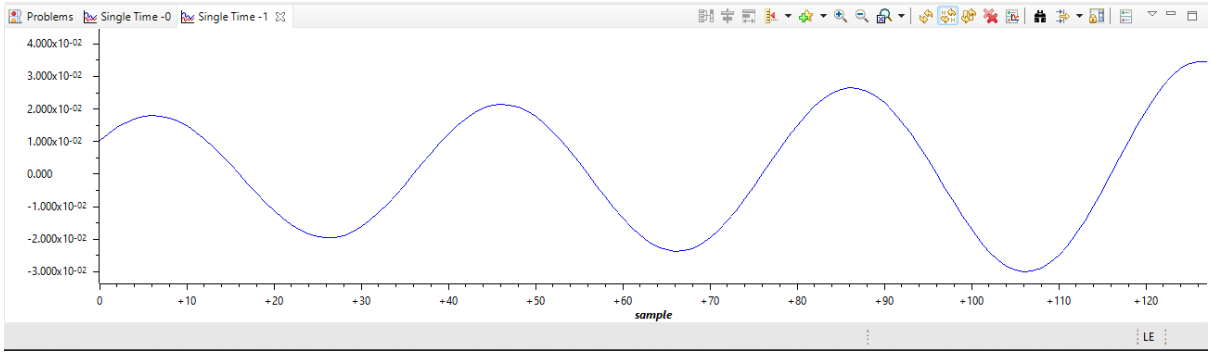
```
FFT output (imaginary):
0.000000 -0.000000 -0.000000 -0.000000 -0.000000 0.000000 -0.0006
Reconstructed Signals:
Reconstructed Signal1[0]: 0.031373
Reconstructed Signal2[0]: 0.015686
Reconstructed Signal3[0]: 0.015686
Reconstructed Signal4[0]: 0.031373
Reconstructed Signal1[1]: 0.031033
Reconstructed Signal2[1]: 0.015516
Reconstructed Signal3[1]: 0.015516
Reconstructed Signal4[1]: 0.031033
Reconstructed Signal1[2]: 0.030022
Reconstructed Signal2[2]: 0.015011
Reconstructed Signal3[2]: 0.015011
Reconstructed Signal4[2]: 0.030022
Reconstructed Signal1[3]: 0.028360
Reconstructed Signal2[3]: 0.014180
Reconstructed Signal3[3]: 0.014180
Reconstructed Signal4[3]: 0.028360
Reconstructed Signal1[4]: 0.026085
Reconstructed Signal2[4]: 0.013043
Reconstructed Signal3[4]: 0.013043
Reconstructed Signal4[4]: 0.026085
Reconstructed Signal1[5]: 0.023246
Reconstructed Signal2[5]: 0.011623
Reconstructed Signal3[5]: 0.011623
Reconstructed Signal4[5]: 0.023246
Reconstructed Signal1[6]: 0.019903
Reconstructed Signal2[6]: 0.009951
Reconstructed Signal3[6]: 0.009951
Reconstructed Signal4[6]: 0.019903
Reconstructed Signal1[7]: 0.016129
Reconstructed Signal2[7]: 0.008064
Reconstructed Signal3[7]: 0.008064
Reconstructed Signal4[7]: 0.016129
Reconstructed Signal1[8]: 0.012006
Reconstructed Signal2[8]: 0.006003
Reconstructed Signal3[8]: 0.006003
Reconstructed Signal4[8]: 0.012006
Reconstructed Signal1[9]: 0.007623
Reconstructed Signal2[9]: 0.003811
Reconstructed Signal3[9]: 0.003811
Reconstructed Signal4[9]: 0.007623
```

WAVEFORMS:

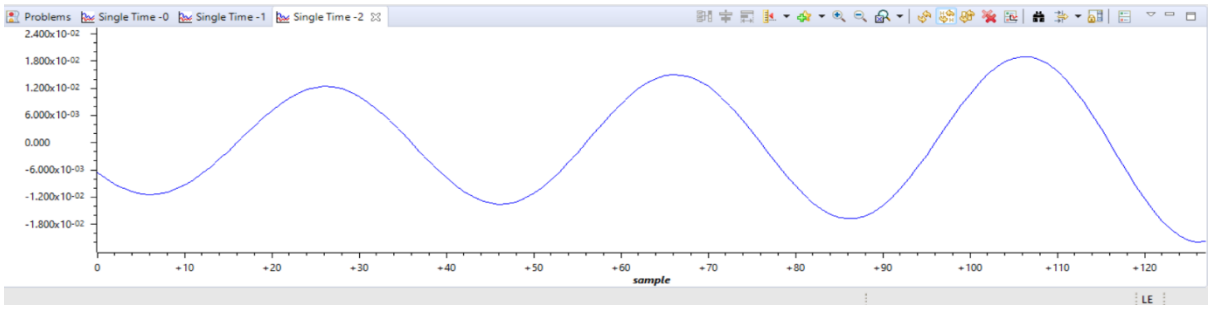
Sinc function 1



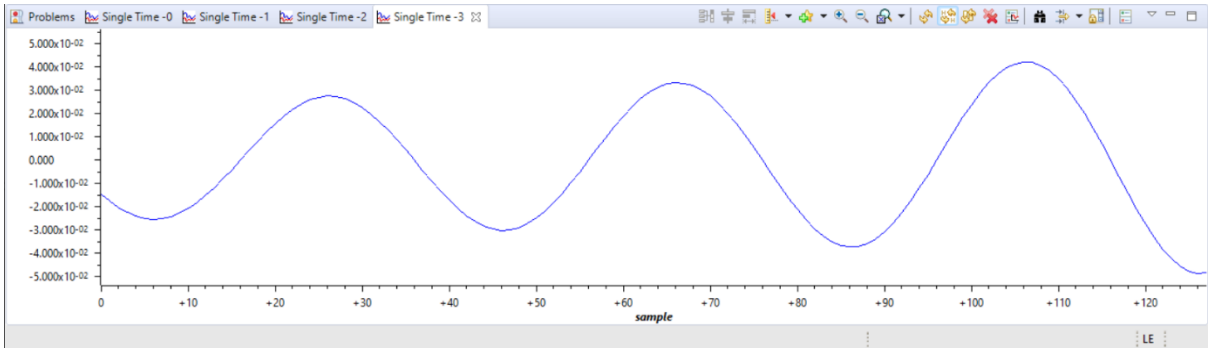
Sinc function 2



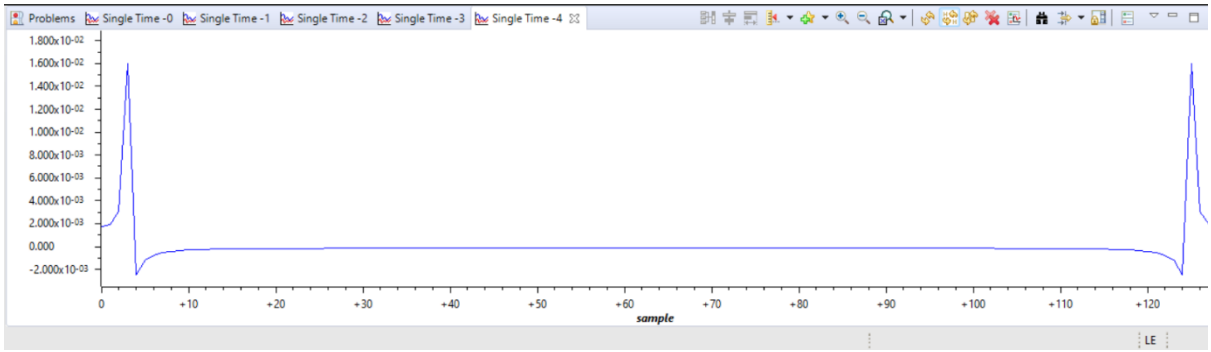
Sinc function 3



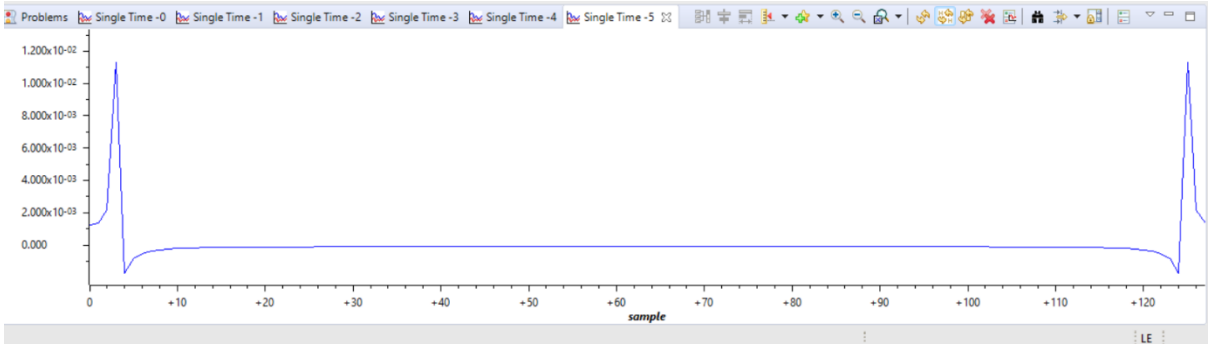
Sinc function 4



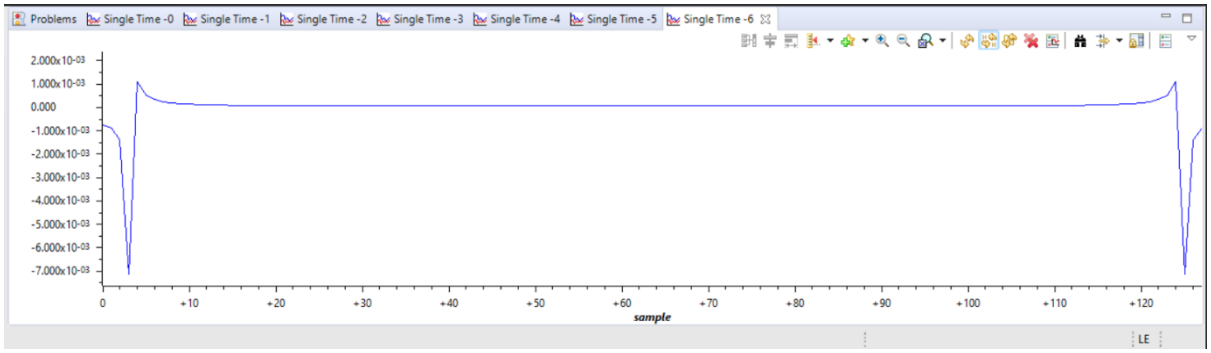
Inverse FFT real value 1



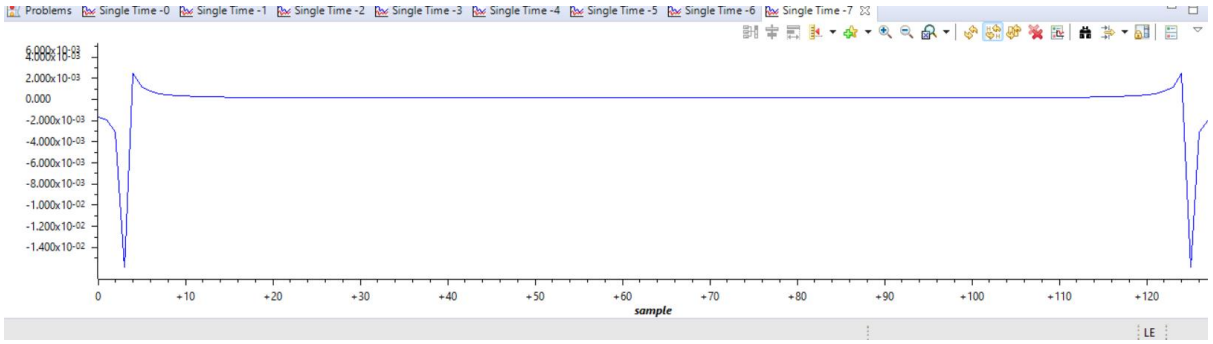
Inverse FFT real value 2



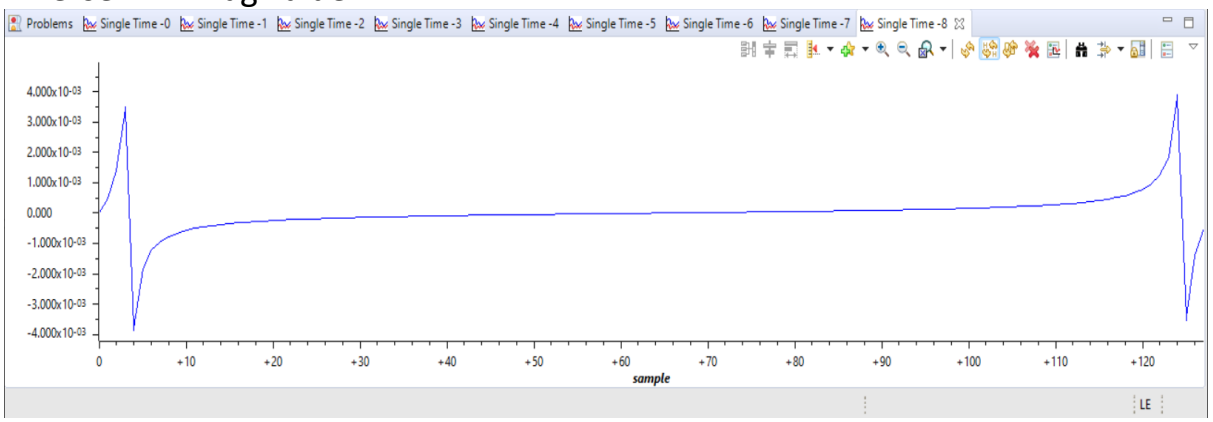
Inverse FFT real value 3



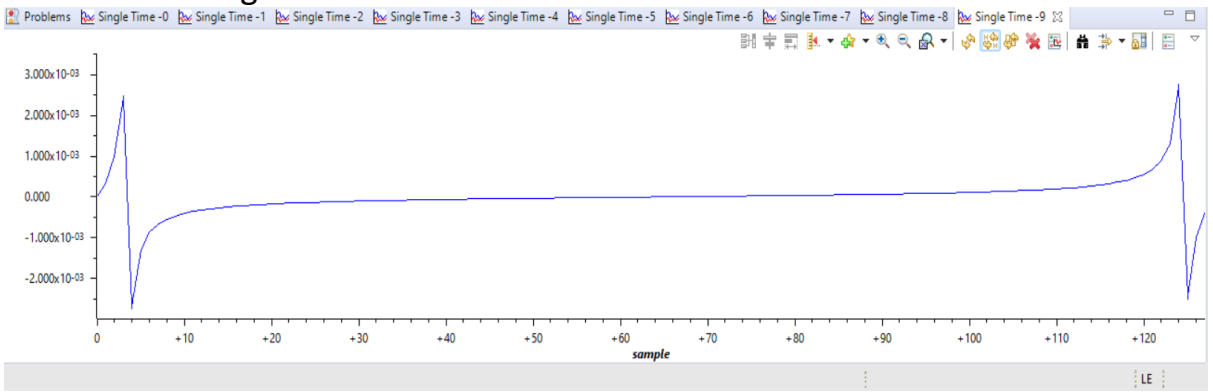
Inverse FFT real value 4



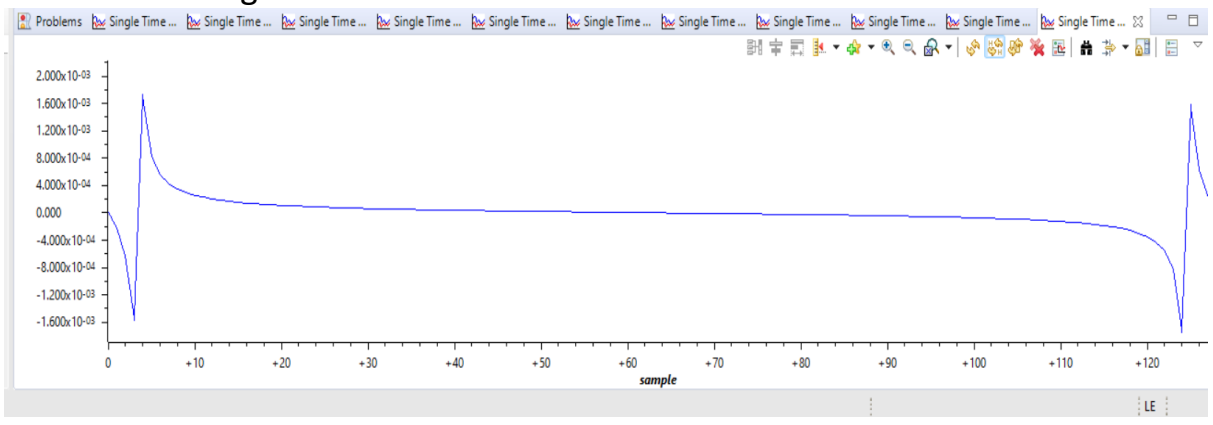
Inverse FFT imag value 1



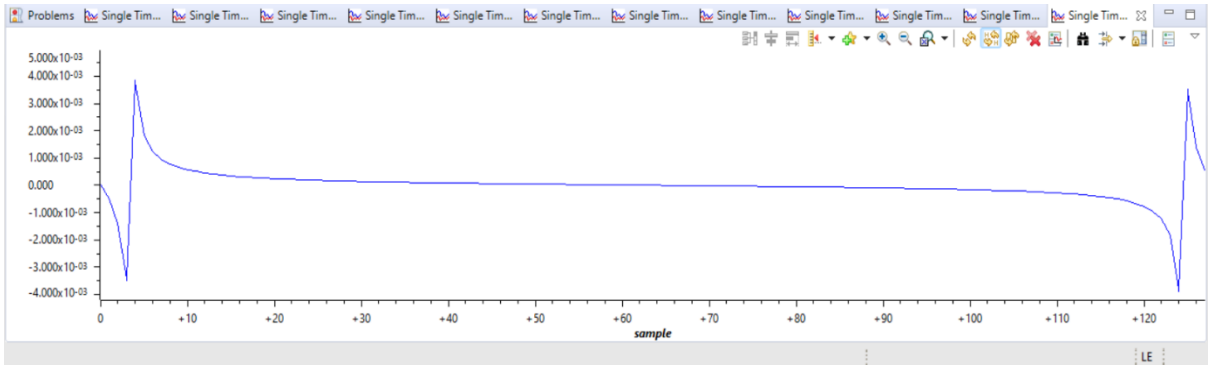
Inverse FFT imag value 2



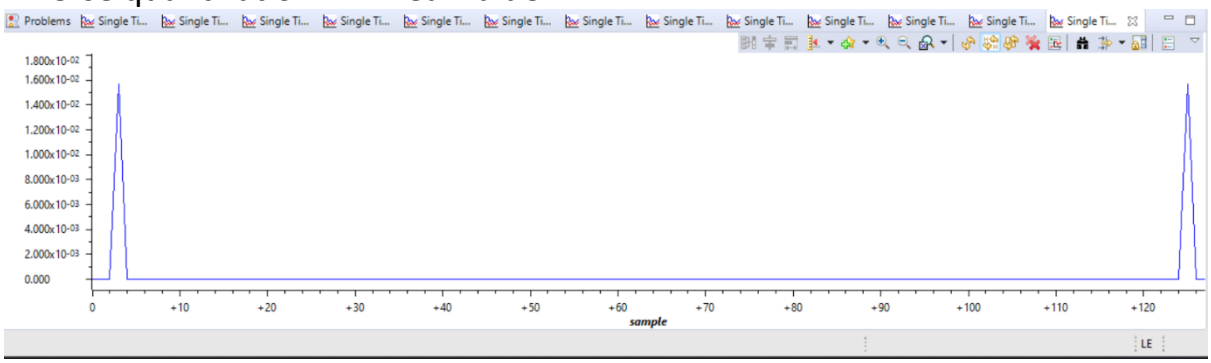
Inverse FFT imag value 3



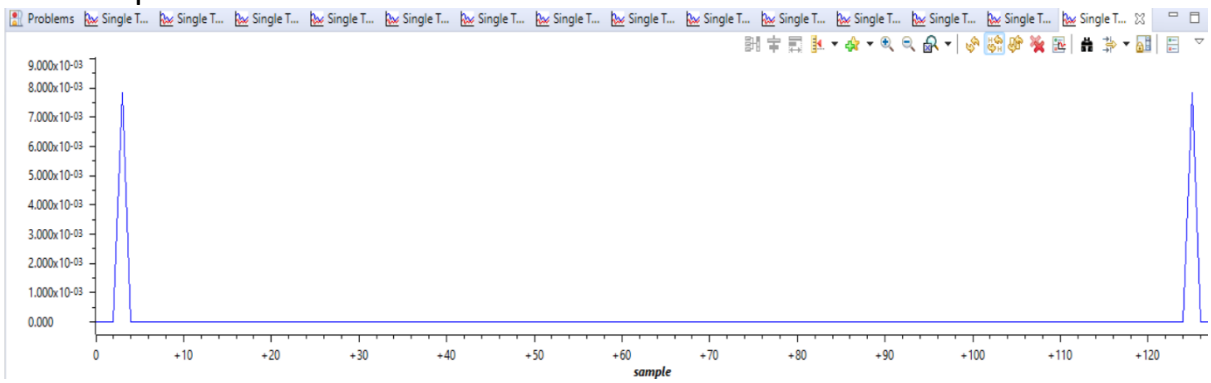
Inverse FFT imag value 4



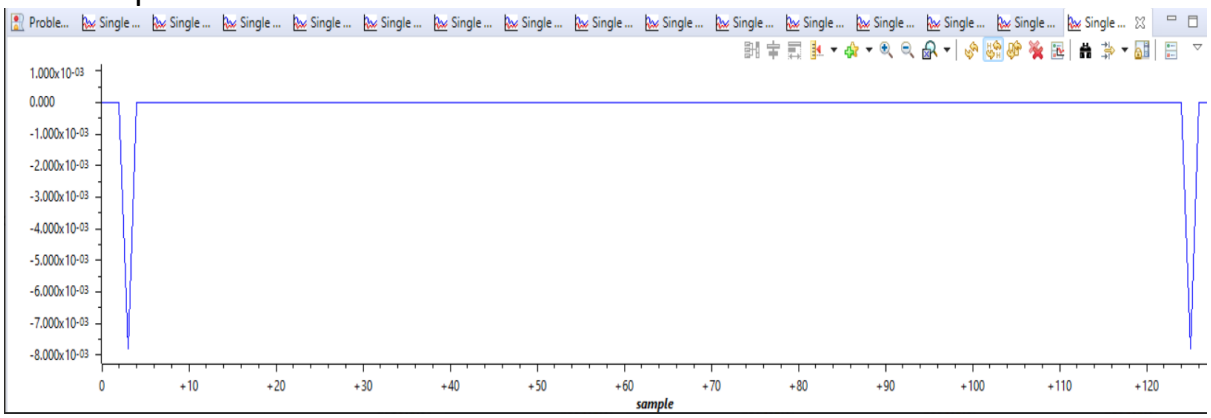
Inverse quantization IFFT real value 1



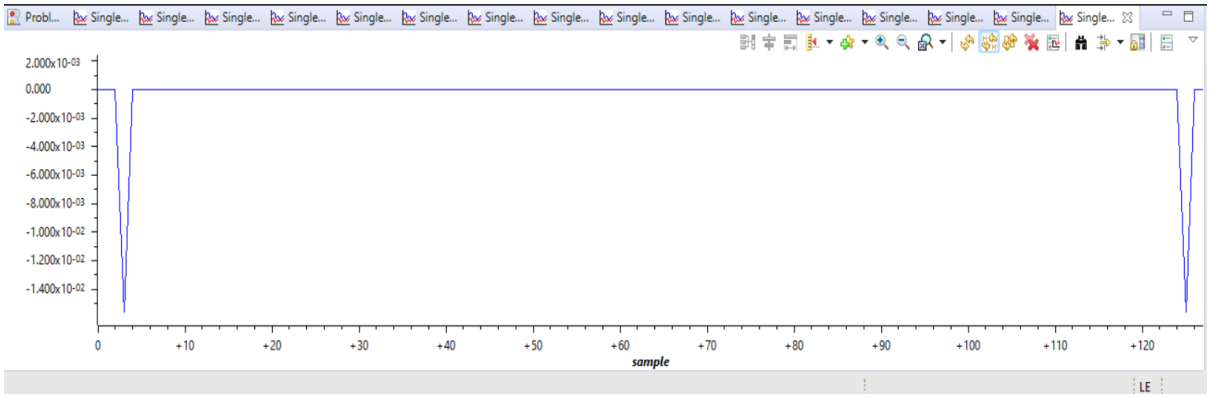
Inverse quantization IFFT real value 2



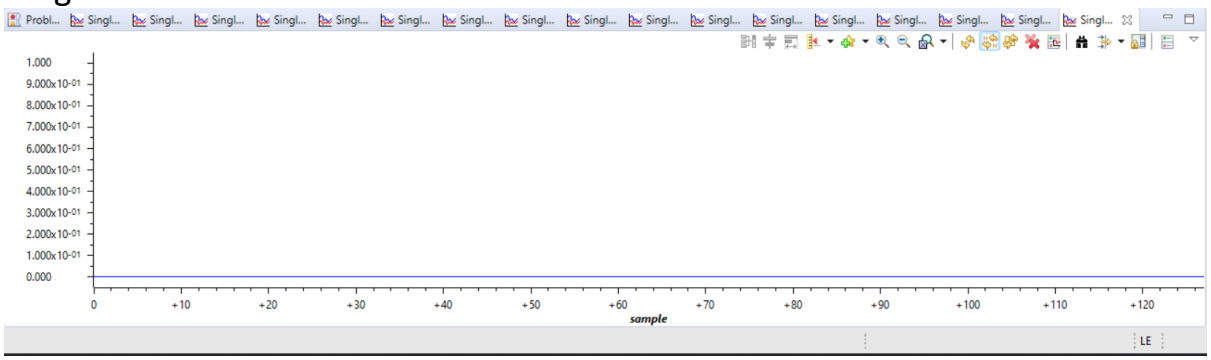
Inverse quantization IFFT real value 3



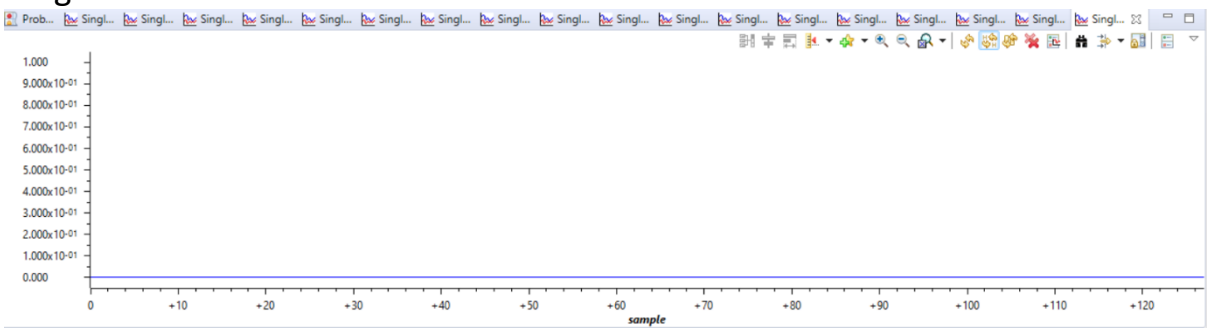
Inverse quantization IFFT real value 4



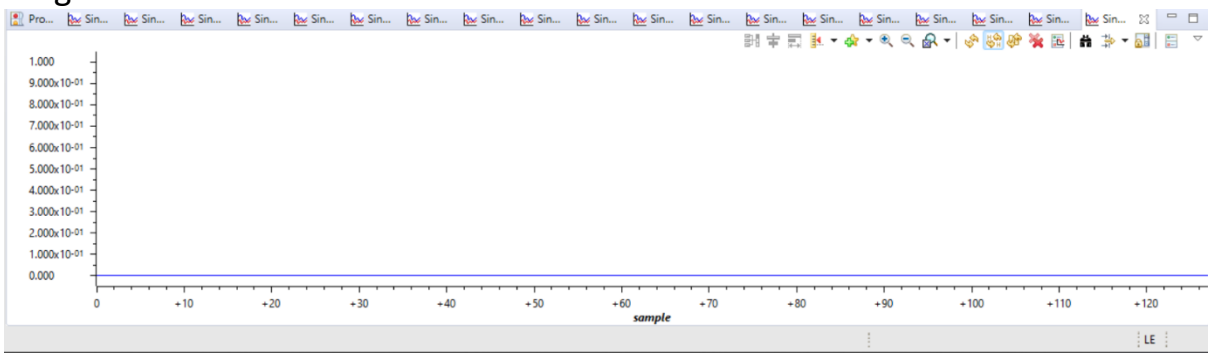
Magnitude of Sinc1 FFT



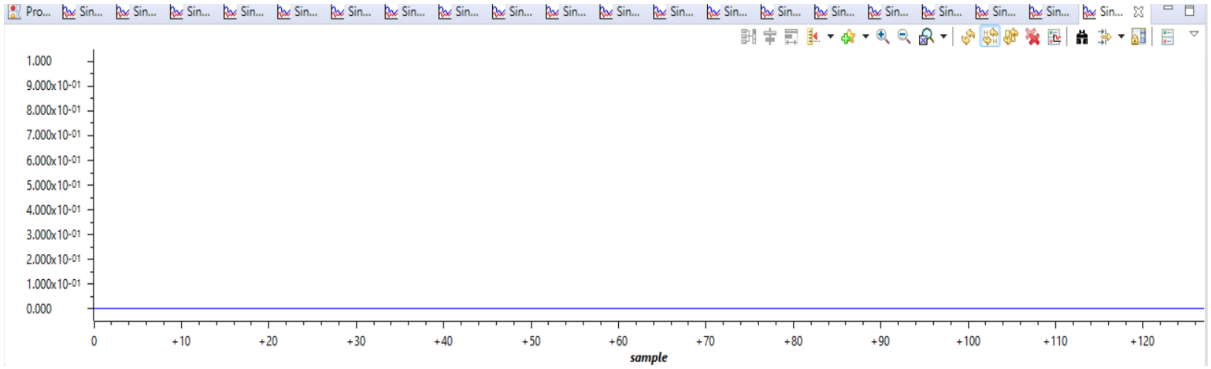
Magnitude of Sinc2 FFT



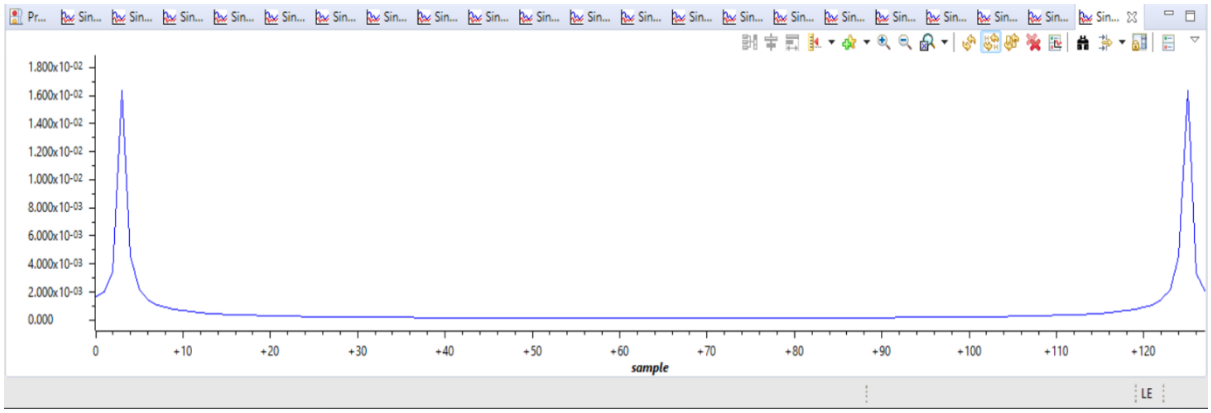
Magnitude of Sinc3 FFT



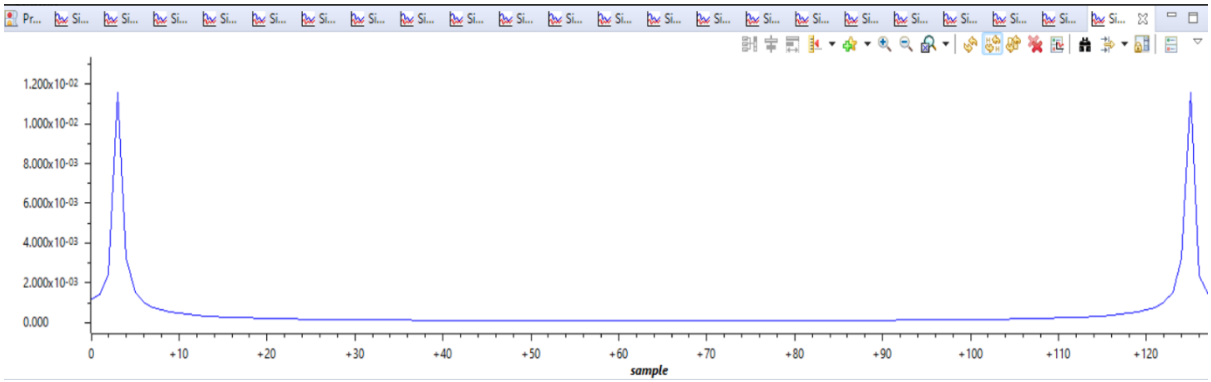
Magnitude of Sinc4 FFT



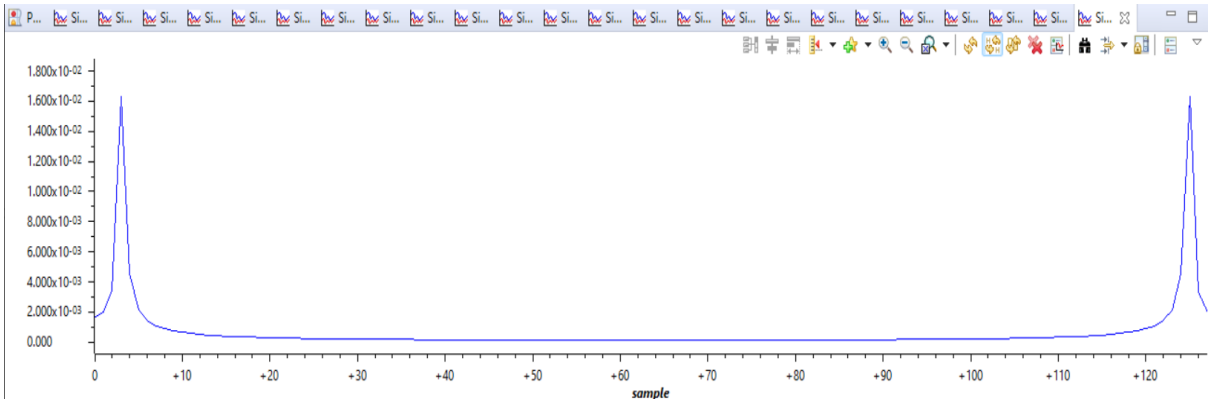
Phase of Sinc1 FFT



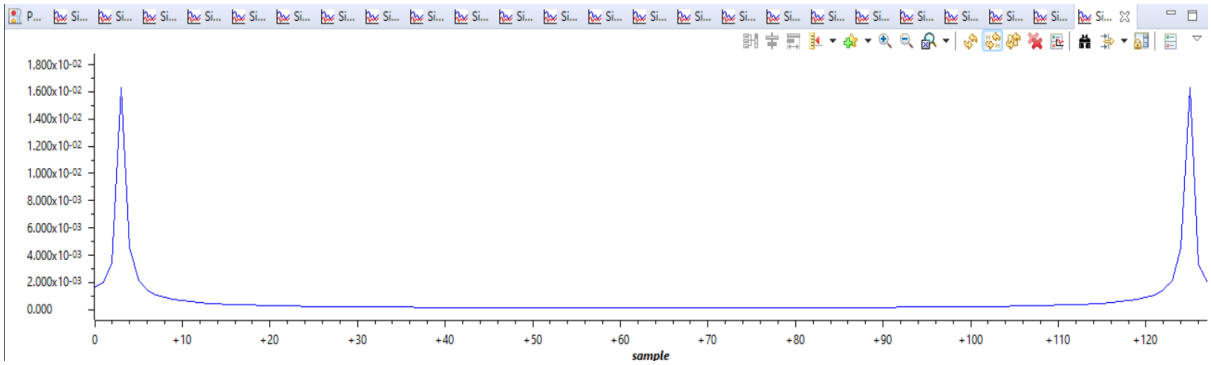
Phase of Sinc2 FFT



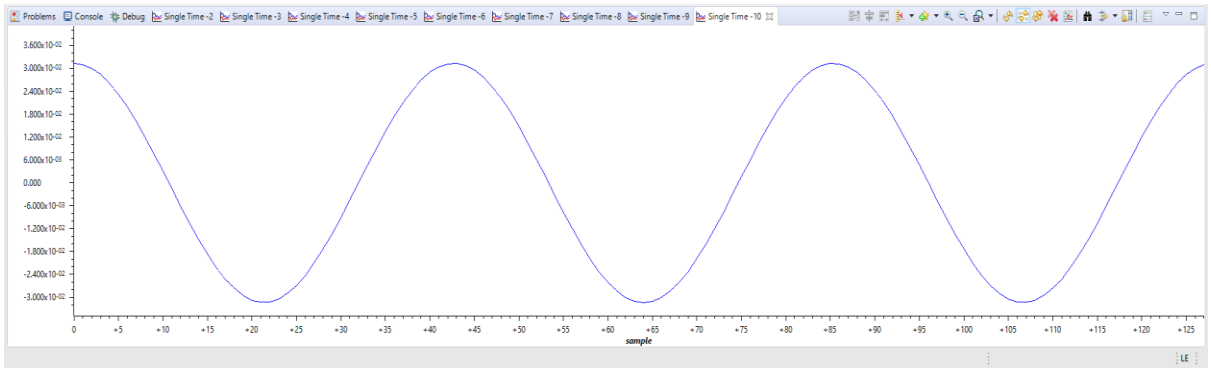
Phase of Sinc3 FFT



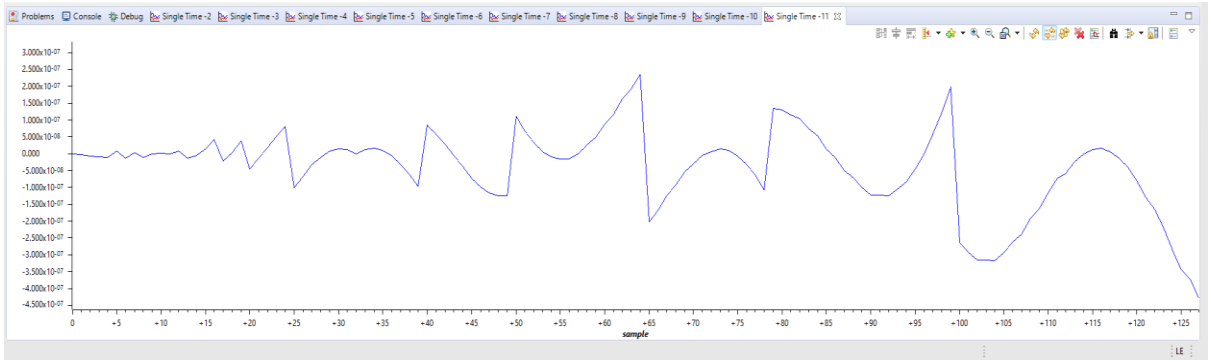
Phase of Sinc4 FFT



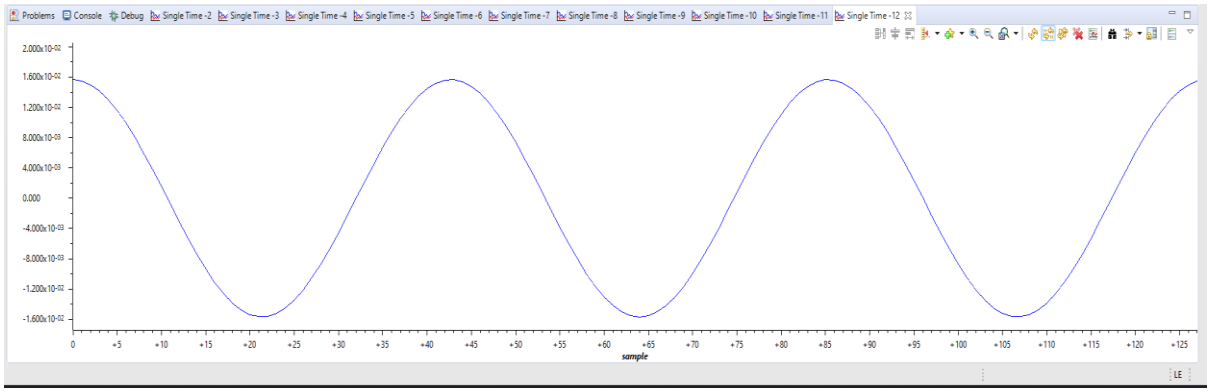
Sinc function1 FFT real



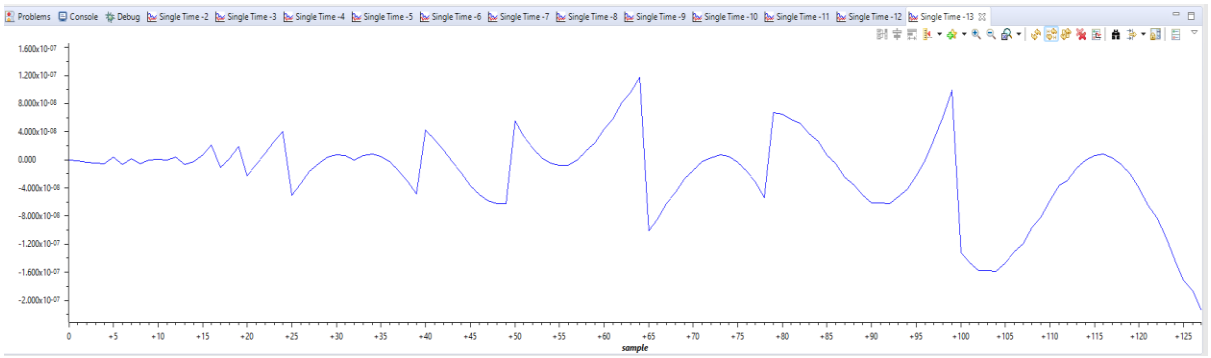
Sinc function1 FFT imag



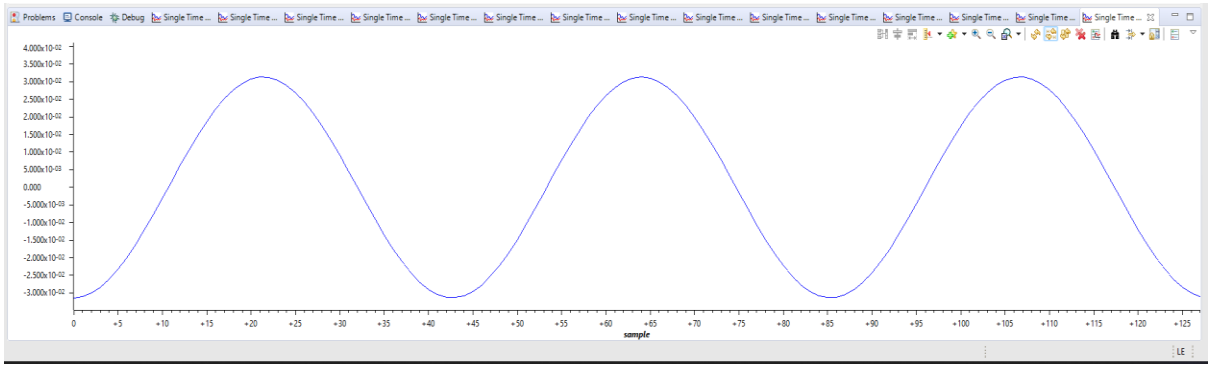
Sinc function2 FFT real



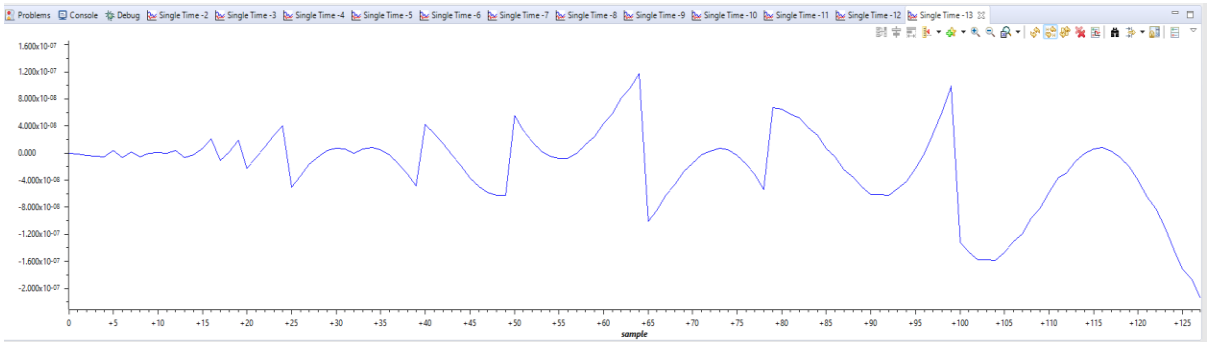
Sinc function2 FFT imag



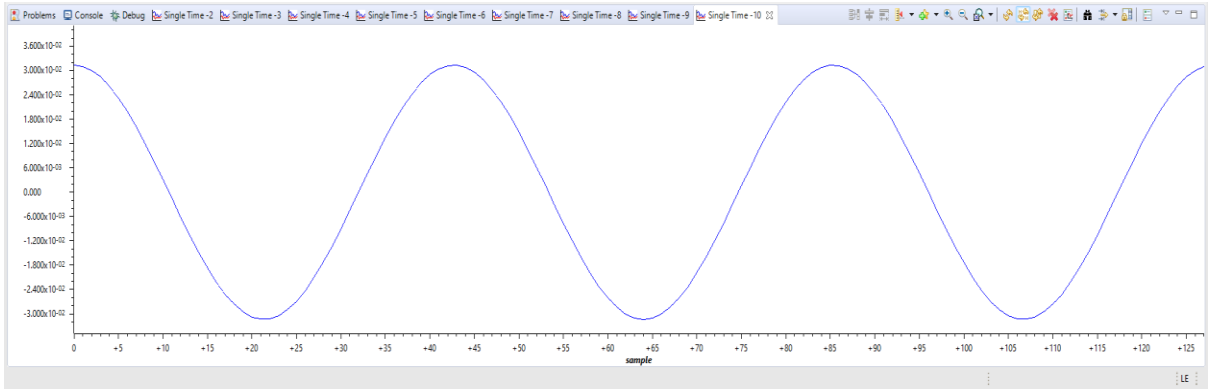
Sinc function3 FFT real



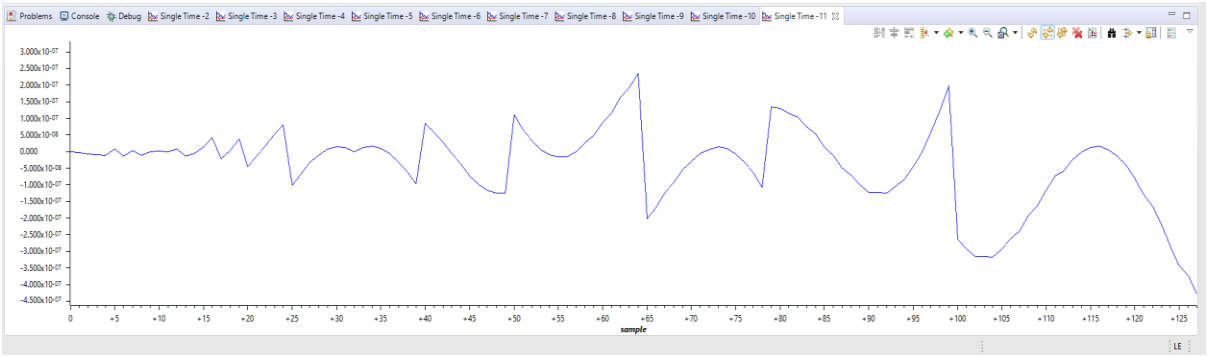
Sinc function3 FFT imag



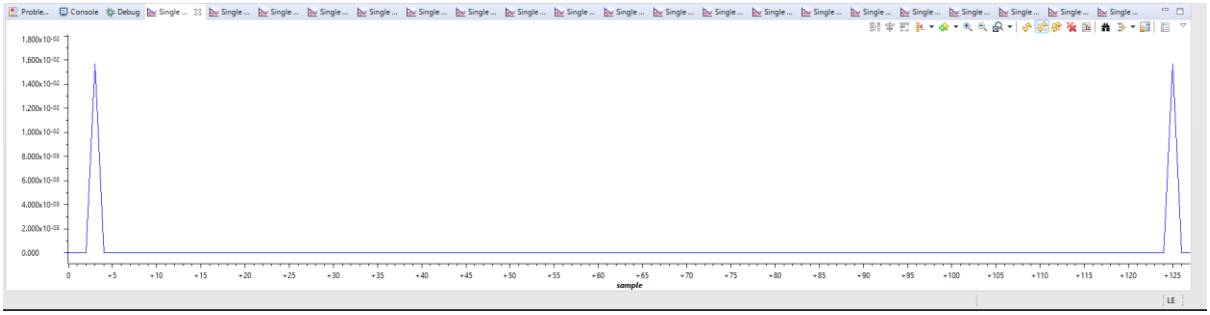
Sinc function4 FFT real



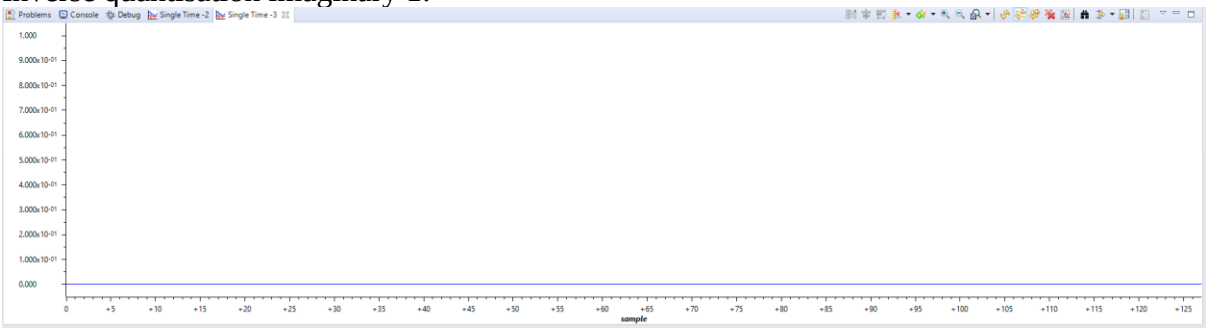
Sinc function4 FFT imag



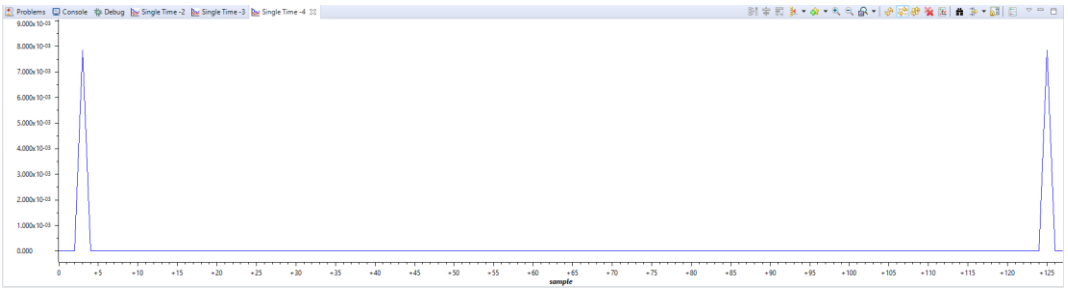
Inverse quantisation real 1:



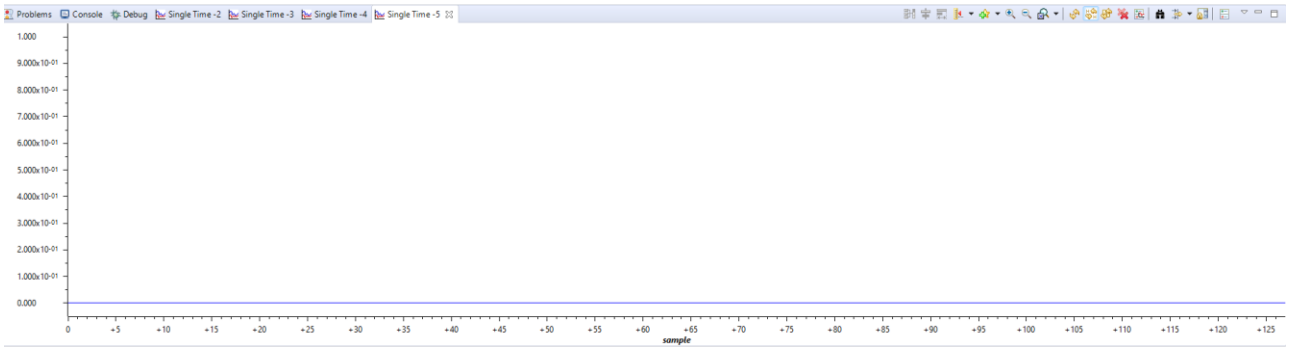
Inverse quantisation imaginary 1:



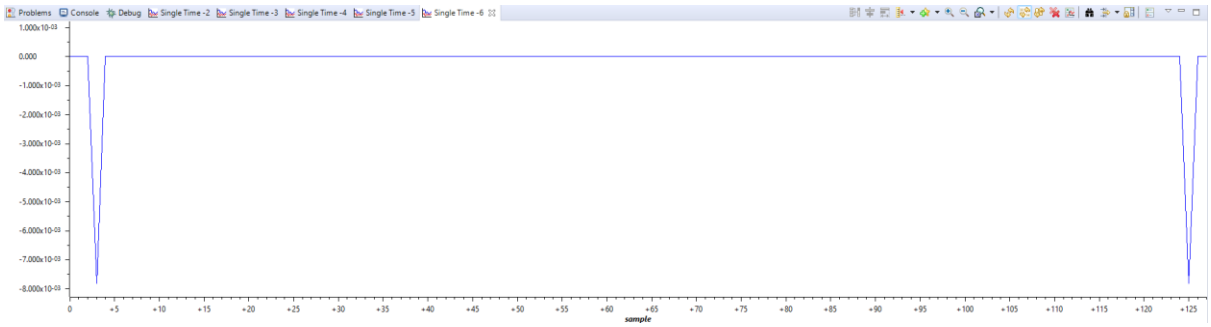
Inverse quantisation real 2:



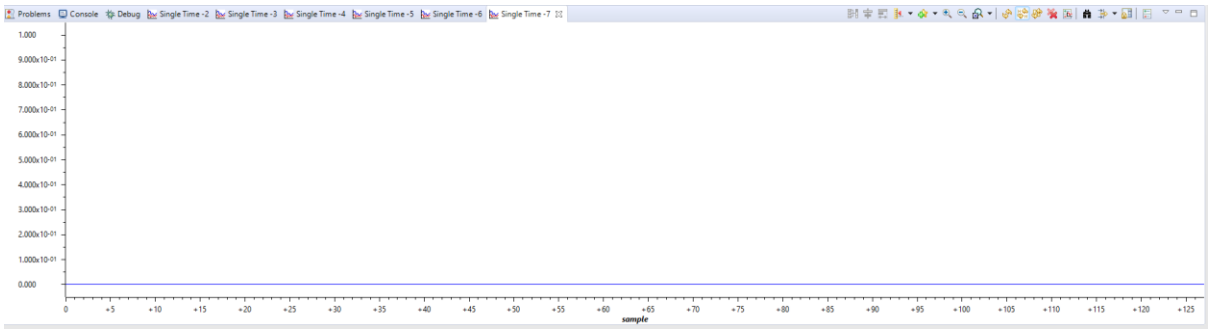
Inverse quantisation imaginary 2:



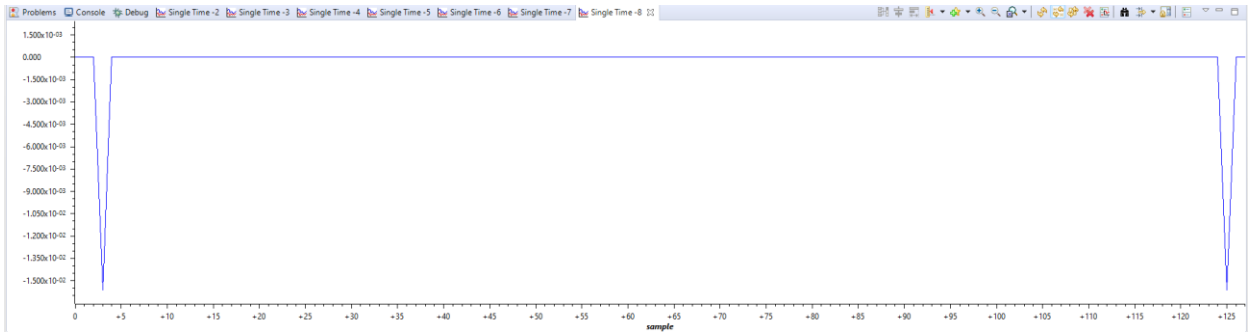
Inverse quantisation real 3:



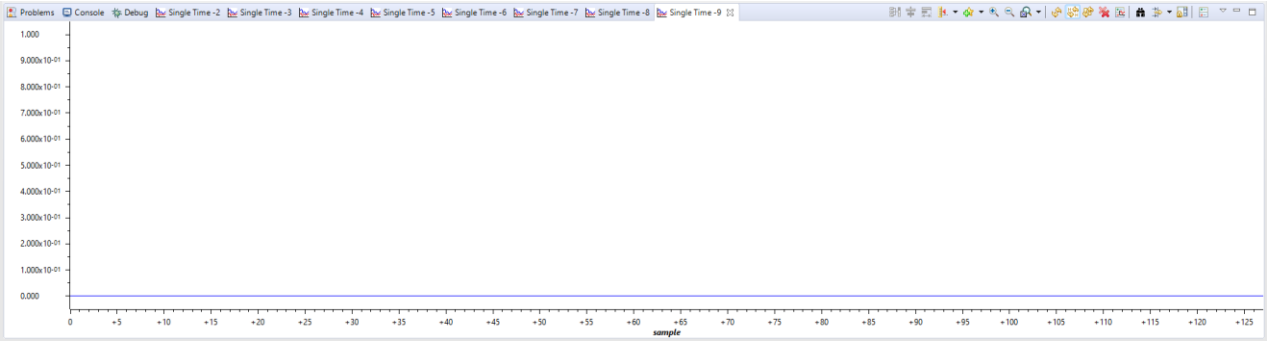
Inverse quantisation imaginary 3:



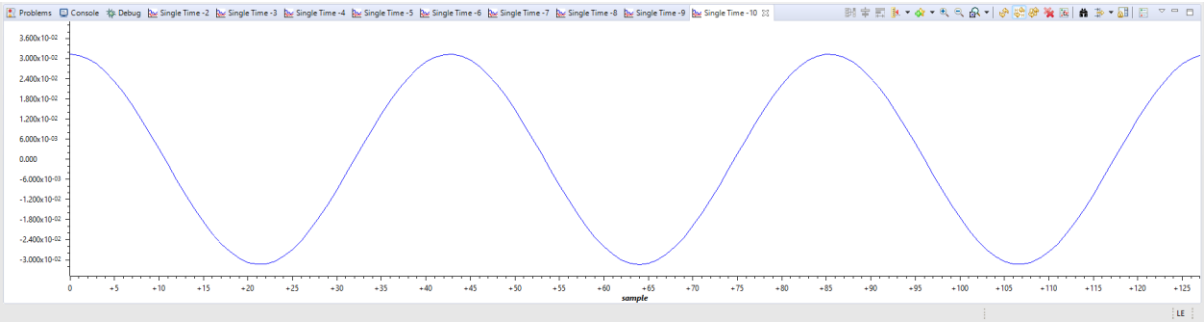
Inverse quantization real 4:



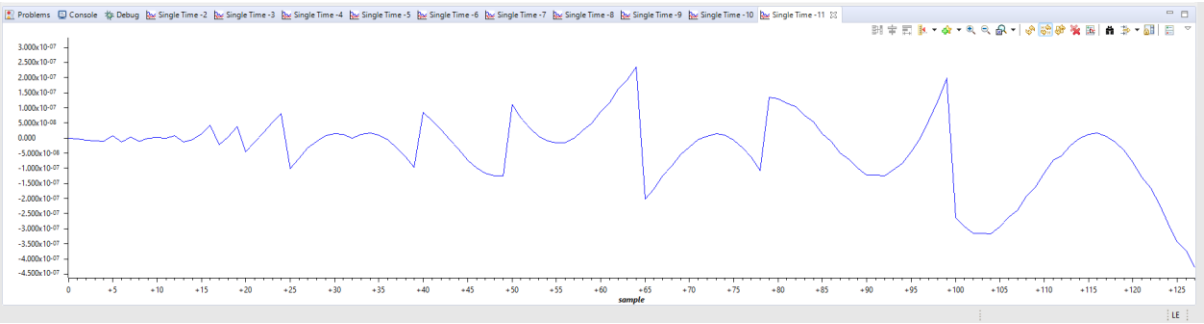
Inverse quantization imaginary 4:



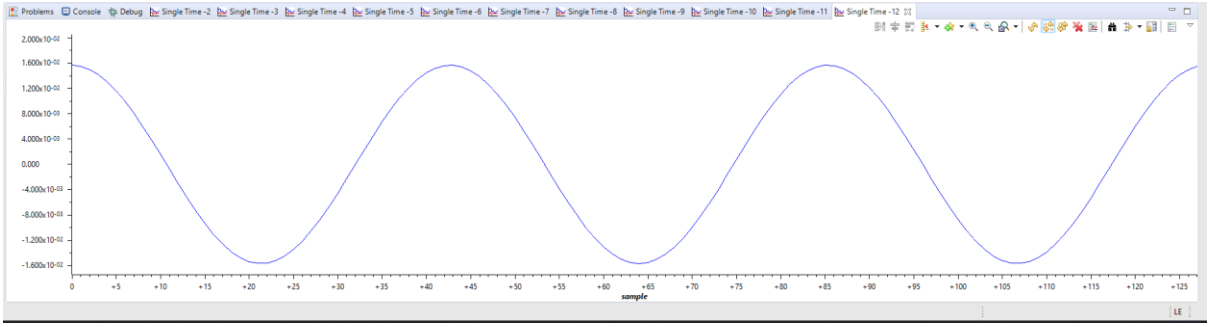
Fast fourier transform real 1:



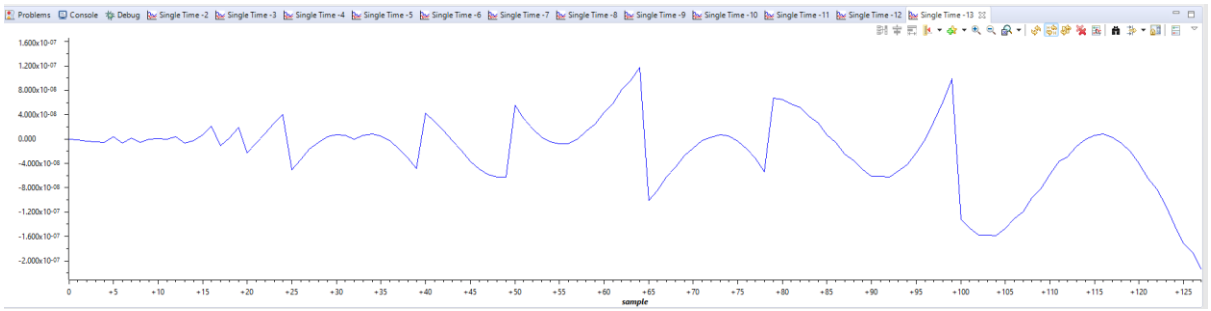
Fast fourier transform imaginary 1



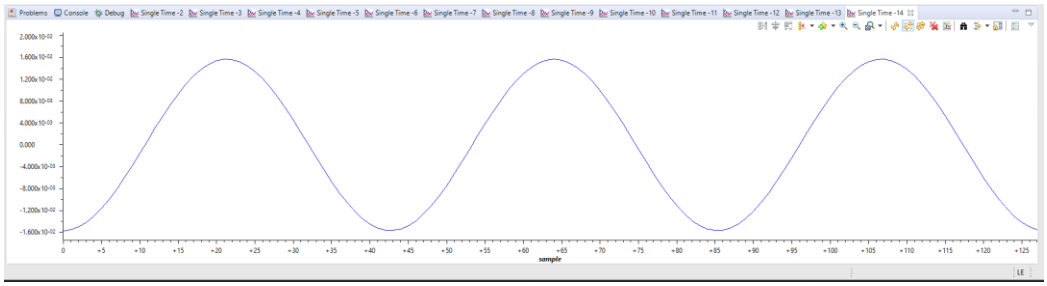
Fast fourier transform real 2



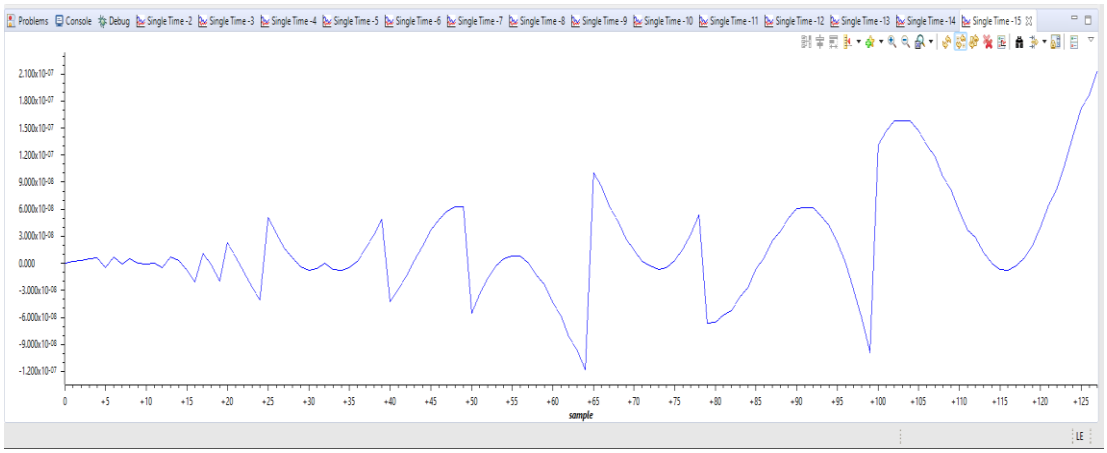
Fast fourier transform imaginary 2



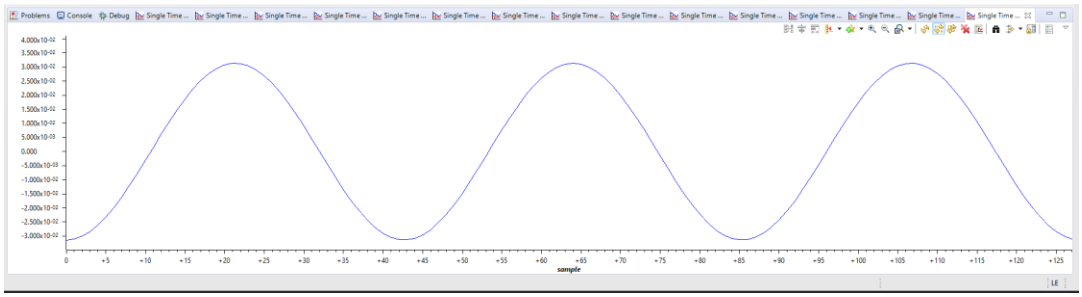
Fast fourier transform real 3



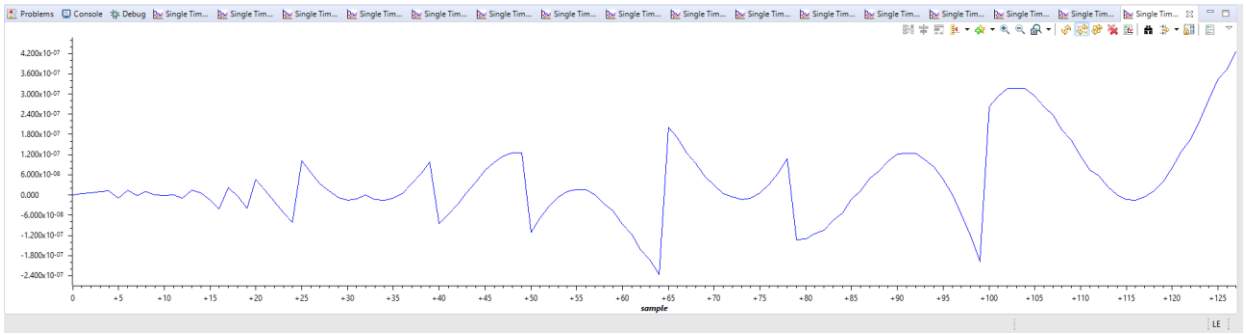
Fast fourier transform imaginary 3



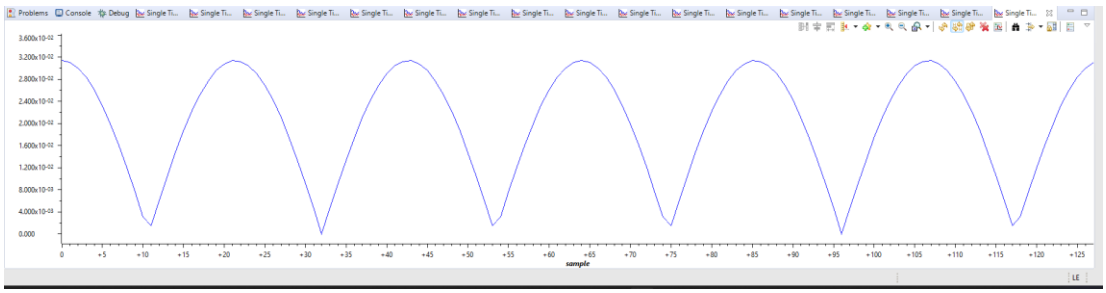
Fast fourier transform real 4



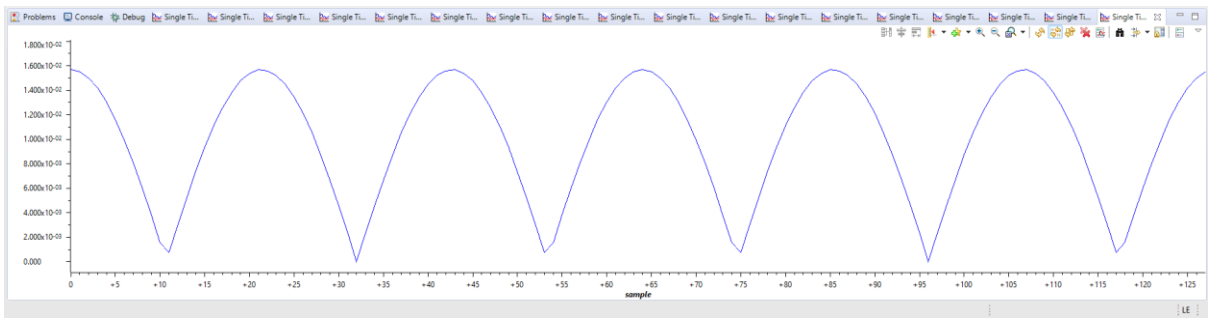
Fast fourier transform imaginary 4



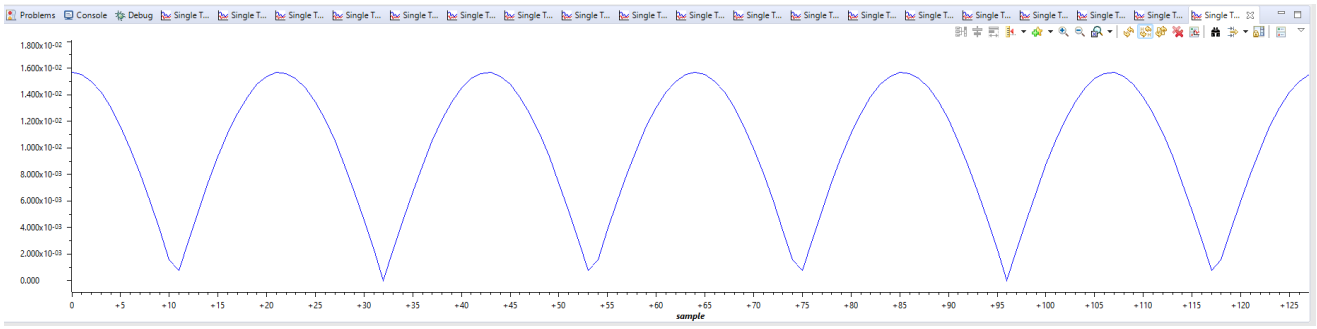
Reconstructed signal 1



Reconstructed signal 2



Reconstructed signal 3


Problems
Console
Debug
1.800x10-02
1.600x10-02
1.400x10-02
1.200x10-02
1.000x10-02
8.000x10-03
6.000x10-03
4.000x10-03
2.000x10-03
0.000
0
+5
+10
+15
+20
+25
+30
+35
+40
+45
+50
+55
+60
+65
+70
+75
+80
+85
+90
+95
+100
+105
+110
+115
+120
+125
sample