



INDRAPRASTHA INSTITUTE *of*  
INFORMATION TECHNOLOGY  
DELHI

Department  
of  
Electronics & Communication Engineering

Embedded Logic Design  
Lab 10 Submission

SHIVAM SHUKLA  
2022478

# Source Code

## 16 Point FFT:

```
Lab10_HW.sdk - Debug - Lab10_fft/src/helloworld.c - Xilinx SDK
File Edit Source Refactor Navigate Search Project Run Xilinx Window Help
system.hdf system.mss dma_init.h helloworld.c asm_vectors.S _exit.c

#include <complex.h>
#include <stdio.h>
#include "xaxidma.h"
#include "platform.h"
#include "xil_printf.h"
#include "xparameters.h"
#include <stdlib.h>
#include <time.h>
#include "dma_init.h"

#define N 16

const int rev16[N] = {0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11, 7, 15};
const float complex W1[N] = {1 - 0 * I, 0 - 1 * I, 0, 0, 1 - 0 * I, 0 - 1 * I, 0, 0, 1 - 0 * I, 0 - 1 * I, 0, 0, 1 - 0 * I, 0 - 1 * I, 0, 0};
const float complex W2[N] = {1 - 0 * I, (1 - 1 * I) / sqrt(2), 0 - 1 * I, -0.707107 * I, 0, 0, 0, 1 - 0 * I, (1 - 1 * I) / sqrt(2), 0 - 1 * I, -0.707107 * I, 0, 0, 0, 0};
const float complex W3[N / 2] = {1 - 0 * I, 0.923880 - 0.382683 * I, 0.707107 - 0.707107 * I, 0.382683 - 0.923880 * I, 0 - 1 * I, -0.382683 - 0.923880 * I, -0.707107 - 0.707107 * I, -0.923880 - 0.382683 * I};
// for(int i=0; i < N/2; i++){
//     W3[i] = cos((i*M_PI)/8) - sin((i*M_PI)/8)*I;
//     //printf("%f %f\n",creal(W3[i]),cimag(W3[i]));
// }

void bitreverse(float complex dataIn[N], float complex dataOut[N]) {
    bit_reversal:
    for (int i = 0; i < N; i++) {
        dataOut[i] = dataIn[rev16[i]];
    }
}

void FFT_stages(float complex FFT_input[N], float complex FFT_output[N]) {
    float complex temp1[N], temp2[N], temp3[N];
    stage1:
    for (int i = 0; i < N; i = i + 2) {
        temp1[i] = FFT_input[i] + FFT_input[i + 1];
        temp1[i + 1] = FFT_input[i] - FFT_input[i + 1];
    }
    stage2:
    for (int i = 0; i < 2; i = i + 1) {
        temp2[i] = temp1[i] + W1[i] * temp1[i + 2];
        temp2[i + 2] = temp1[i] - W1[i] * temp1[i + 2];
    }
    for (int i = 4; i < 6; i = i + 1) {
        temp2[i] = temp1[i] + W1[i] * temp1[i + 2];
        temp2[i + 2] = temp1[i] - W1[i] * temp1[i + 2];
    }
    for (int i = 8; i < 10; i = i + 1) {
        temp2[i] = temp1[i] + W1[i] * temp1[i + 2];
        temp2[i + 2] = temp1[i] - W1[i] * temp1[i + 2];
    }
    for (int i = 12; i < 14; i = i + 1) {
        temp2[i] = temp1[i] + W1[i] * temp1[i + 2];
        temp2[i + 2] = temp1[i] - W1[i] * temp1[i + 2];
    }
    temp3[i + 2] = temp2[i + 1] - W1[i + 1] * temp2[i + 2];
}

Writable Smart Insert 91:11
```

```
Lab10_HW.sdk - Debug - Lab10_fft/src/helloworld.c - Xilinx SDK
File Edit Source Refactor Navigate Search Project Run Xilinx Window Help
system.hdf system.mss dma_init.h helloworld.c asm_vectors.S _exit.c

void bitreverse(float complex dataIn[N], float complex dataOut[N]) {
    bit_reversal:
    for (int i = 0; i < N; i++) {
        dataOut[i] = dataIn[rev16[i]];
    }
}

void FFT_stages(float complex FFT_input[N], float complex FFT_output[N]) {
    float complex temp1[N], temp2[N], temp3[N];
    stage1:
    for (int i = 0; i < N; i = i + 2) {
        temp1[i] = FFT_input[i] + FFT_input[i + 1];
        temp1[i + 1] = FFT_input[i] - FFT_input[i + 1];
    }
    stage2:
    for (int i = 0; i < 2; i = i + 1) {
        temp2[i] = temp1[i] + W1[i] * temp1[i + 2];
        temp2[i + 2] = temp1[i] - W1[i] * temp1[i + 2];
    }
    for (int i = 4; i < 6; i = i + 1) {
        temp2[i] = temp1[i] + W1[i] * temp1[i + 2];
        temp2[i + 2] = temp1[i] - W1[i] * temp1[i + 2];
    }
    for (int i = 8; i < 10; i = i + 1) {
        temp2[i] = temp1[i] + W1[i] * temp1[i + 2];
        temp2[i + 2] = temp1[i] - W1[i] * temp1[i + 2];
    }
    for (int i = 12; i < 14; i = i + 1) {
        temp2[i] = temp1[i] + W1[i] * temp1[i + 2];
        temp2[i + 2] = temp1[i] - W1[i] * temp1[i + 2];
    }
    stage3:
    for (int i = 0; i < N / 4; i = i + 1) {
        temp3[i] = temp2[i] + W2[i] * temp2[i + 4];
        temp3[i + 4] = temp2[i] - W2[i] * temp2[i + 4];
    }
    for (int i = 8; i < 12; i = i + 1) {
        temp3[i] = temp2[i] + W2[i] * temp2[i + 4];
        temp3[i + 4] = temp2[i] - W2[i] * temp2[i + 4];
    }
    stage4:
    for (int i = 0; i < N / 2; i = i + 1) {
        FFT_output[i] = temp3[i] + W3[i] * temp3[i + 8];
        FFT_output[i + 8] = temp3[i] - W3[i] * temp3[i + 8];
    }
    // printf("\n Printinf temp1\n\n");
    // for (int i = 0; i < N; i++) {
    //     printf("%f %f\n", creal(temp1[i]), cimag(temp1[i]));
    // }
}

Writable Smart Insert 91:11
```

```
Lab10_HWsdk - Debug - Lab10_Rt/src/helloworld.c - Xilinx SDK
File Edit Source Refactor Navigate Search Project Run Xilinx Window Help
system.hdf system.mss dma_init.h helloworld.c asm_vectors.S @_exit.c
+int main() {
    init_platform();
    // Initializing Timer instances for PS and PL
    XTime_PL_start_time, PL_end_time;
    XTime_PS_start_time, PS_end_time;
    // Initializing software and hardware output buffers
    const float complex FFT_input[N] = {11 + 23*I, 32 + 10*I, 91 + 94*I, 15 + 69*I, 47 + 96*I, 44 + 12*I, 96 + 17*I, 49 + 58*I,
    11 + 23*I, 32 + 10*I, 91 + 94*I, 15 + 69*I, 47 + 96*I, 44 + 12*I, 96 + 17*I, 49 + 58*I};
    float complex FFT_output_sw[N], FFT_output_hw[N];
    float complex FFT_rev_sw[N];

    // Software 8-point FFT
    XTime_SetTime(0); // Setting Timer to value 0
    XTime_GetTime(&PS_start_time); // Get Start Time
    bitreverse(FFT_input, FFT_rev_sw);
    FFT_stages(FFT_rev_sw, FFT_output_sw);
    XTime_GetTime(&PS_end_time); // Get End Time

    // Hardware 16-point FFT
    int status;
    XAxiDma AxiDma;
    status = DMA_Init(&AxiDma, XPAR_AXI_DMA_0_DEVICE_ID);
    if(status)
        return 1;

    XTime_SetTime(0); // Setting Timer to value 0
    XTime_GetTime(&PL_start_time); // Get Start Time
    // Simple DMA Transfers
    status = XAxiDma_SimpleTransfer(&AxiDma, (UINTPTR)FFT_output_hw, (sizeof(float complex)*N), XAXIDMA_DEVICE_TO_DMA);
    status = XAxiDma_SimpleTransfer(&AxiDma, (UINTPTR)FFT_input, (sizeof(float complex)*N), XAXIDMA_DMA_TO_DEVICE);

    // POLLING-Check whether the DMA-to-Device and Device-to-DMA transfers are complete
    while(XAxiDma_Busy(&AxiDma, XAXIDMA_DMA_TO_DEVICE));

    while(XAxiDma_Busy(&AxiDma, XAXIDMA_DEVICE_TO_DMA));

    XTime_GetTime(&PL_end_time); // Get End Time

    // Verifying Hardware result with Software
    for(int i=0; i<N; i++){
        printf("\nPS Output: %f+%fi, PL Output: %f+%fi", crealf(FFT_output_sw[i]), cimagf(FFT_output_sw[i]), crealf(FFT_output_hw[i]), cimagf(FFT_output_hw[i]));
        float diff1= abs(crealf(FFT_output_sw[i])-crealf(FFT_output_hw[i]));
        float diff2= abs(cimagf(FFT_output_sw[i])-cimagf(FFT_output_hw[i]));
        if(diff1>=0.0001 && diff2>=0.0001){
            printf("\nData Mismatch Found at index %d ", i);
            break;
        }
        else{
            // ...
        }
    }
}
```

```
Lab10_HWsdk - Debug - Lab10_Rt/src/helloworld.c - Xilinx SDK
File Edit Source Refactor Navigate Search Project Run Xilinx Window Help
system.hdf system.mss dma_init.h helloworld.c asm_vectors.S @_exit.c
XTime_GetTime(&PS_start_time); // Get Start Time
bitreverse(FFT_input, FFT_rev_sw);
FFT_stages(FFT_rev_sw, FFT_output_sw);
XTime_GetTime(&PS_end_time); // Get End Time

// Hardware 16-point FFT
int status;
XAxiDma AxiDma;
status = DMA_Init(&AxiDma, XPAR_AXI_DMA_0_DEVICE_ID);
if(status)
    return 1;

XTime_SetTime(0); // Setting Timer to value 0
XTime_GetTime(&PL_start_time); // Get Start Time
// Simple DMA Transfers
status = XAxiDma_SimpleTransfer(&AxiDma, (UINTPTR)FFT_output_hw, (sizeof(float complex)*N), XAXIDMA_DEVICE_TO_DMA);
status = XAxiDma_SimpleTransfer(&AxiDma, (UINTPTR)FFT_input, (sizeof(float complex)*N), XAXIDMA_DMA_TO_DEVICE);

// POLLING-Check whether the DMA-to-Device and Device-to-DMA transfers are complete
while(XAxiDma_Busy(&AxiDma, XAXIDMA_DMA_TO_DEVICE));

while(XAxiDma_Busy(&AxiDma, XAXIDMA_DEVICE_TO_DMA));

XTime_GetTime(&PL_end_time); // Get End Time

// Verifying Hardware result with Software
for(int i=0; i<N; i++){
    printf("\nPS Output: %f+%fi, PL Output: %f+%fi", crealf(FFT_output_sw[i]), cimagf(FFT_output_sw[i]), crealf(FFT_output_hw[i]), cimagf(FFT_output_hw[i]));
    float diff1= abs(crealf(FFT_output_sw[i])-crealf(FFT_output_hw[i]));
    float diff2= abs(cimagf(FFT_output_sw[i])-cimagf(FFT_output_hw[i]));
    if(diff1>=0.0001 && diff2>=0.0001){
        printf("\nData Mismatch Found at index %d ", i);
        break;
    }
    else{
        printf("DMA Transfer Successful!");
    }
}

printf("\n- Execution Time Comparison-");
float time=0;
time= (float)1.0*(PS_end_time-PS_start_time)/(COUNTS_PER_SECOND/1000000);
printf("\nExecution time for PS in micro-seconds: %f", time);
time=0;
time= (float)1.0 * (PL_end_time-PL_start_time)/(COUNTS_PER_SECOND/1000000);
printf("\nExecution time for PL in micro-seconds: %f", time);
return 0;
}
```

# BLOCK DESIGN

Lab10\_HW - [D:/Drive\_D/ELD2023/Lab10\_HW/Lab10\_HW.xpr] - Vivado 2019.1

File Edit Flow Tools Reports Window Layout View Help Quick Access

write\_bitstream Complete

Default Layout

Flow Navigator

- PROJECT MANAGER
  - Settings
  - Add Sources
  - Language Templates
  - IP Catalog
- IP INTEGRATOR
  - Create Block Design
  - Open Block Design
  - Generate Block Design
- SIMULATION
  - Run Simulation
- RTL ANALYSIS
  - Open Elaborated Design
- SYNTHESIS
  - Run Synthesis
  - Open Synthesized Design
- IMPLEMENTATION
  - Run Implementation
  - Open Implemented Design
- PROGRAM AND DEBUG
  - Generate Bitstream
  - Open Hardware Manager

BLOCK DESIGN - design\_fft

Sources Design Signals Board

design\_fft

- External Interfaces
- Interface Connections
- Nets
- axi\_dma\_0 (AXI Direct Memory Access7.1)
- axi\_smc (AXI SmartConnect1.0)
- processing\_system7\_0 (ZYNQ7 Processing System5.5)
- ps7\_0\_axi\_periph
- rst\_ps7\_0\_100M (Processor System Reset5.0)

Properties

Select an object to see properties

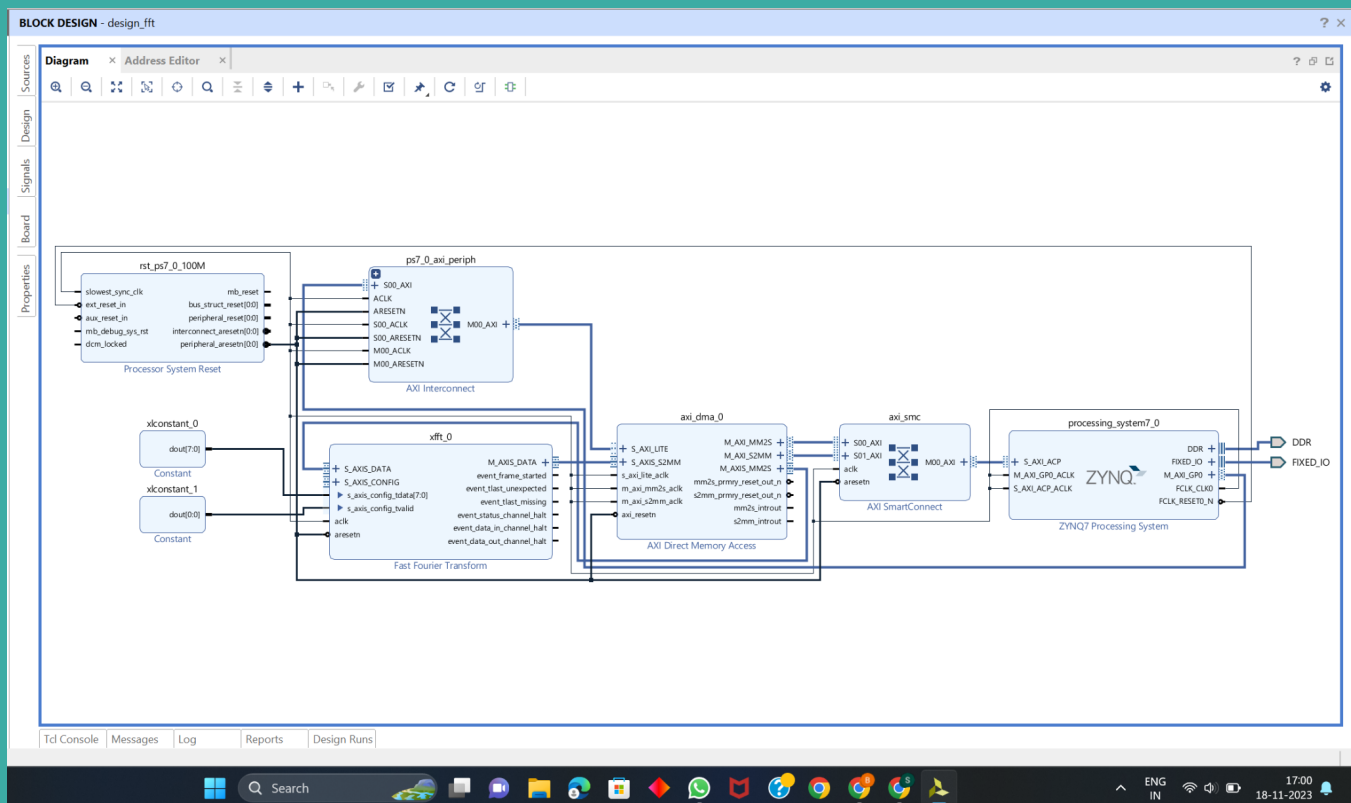
Diagram Address Editor

Tcl Console

```

Adding component instance block -- xilinx.com:ip:xlconstant:1.1 - xlconstant_1
Excluding </processing_system7_0/S_AXI_ACP/ACP_IOP> from </axi_dma_0/Data_MM2S>
Excluding </processing_system7_0/S_AXI_ACP/ACP_M_AXI_GPO> from </axi_dma_0/Data_MM2S>
Excluding </processing_system7_0/S_AXI_ACP/ACP_IOP> from </axi_dma_0/Data_S2MM>
Excluding </processing_system7_0/S_AXI_ACP/ACP_M_AXI_GPO> from </axi_dma_0/Data_S2MM>
Successfully read diagram <design_fft> from BD file <D:/Drive_D/ELD2023/Lab10_HW/Lab10_HW.srcs/sources_1/bd/design_fft/design_fft.bd>
open_bd_design: Time (s): cpu = 00:00:03 ; elapsed = 00:00:07 . Memory (MB): peak = 1524.727 ; gain = 52.793
  
```

Type a Tcl command here



# jtag terminal output

## 16 Point FFT:

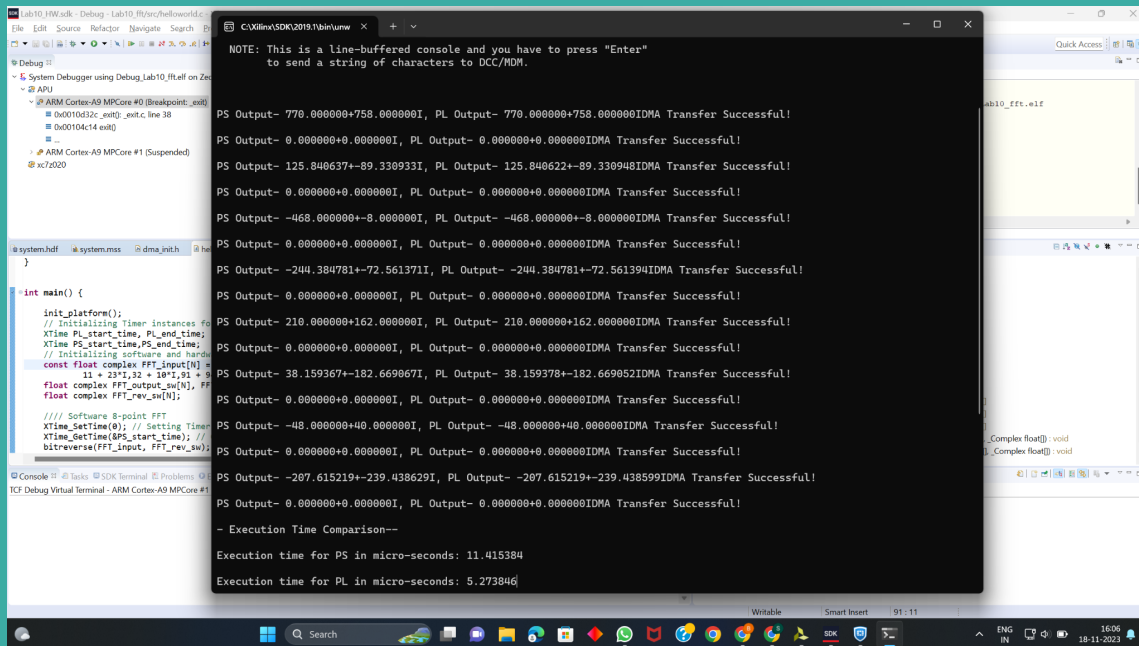
```
C:\Xilinx\SDK\2019.1\bin\unw x + v
JTAG-based Hyperterminal.
Connected to JTAG-based Hyperterminal over TCP port : 53406
(using socket : sock680)
Help :
Terminal requirements :
(i) Processor's STDOUT is redirected to the ARM DCC/MDM UART
(ii) Processor's STDIN is redirected to the ARM DCC/MDM UART.
Then, text input from this console will be sent to DCC/MDM's UART port.
NOTE: This is a line-buffered console and you have to press "Enter"
to send a string of characters to DCC/MDM.

PS Output- 770.000000+758.000000I, PL Output- 770.000000+758.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 125.840637+-89.330933I, PL Output- 125.840622+-89.330948IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -468.000000+-8.000000I, PL Output- -468.000000+-8.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -244.384781+-72.561371I, PL Output- -244.384781+-72.561394IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 210.000000+162.000000I, PL Output- 210.000000+162.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 38.159367+-182.669067I, PL Output- 38.159378+-182.669052IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -48.000000+40.000000I, PL Output- -48.000000+40.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -207.615219+-239.438629I, PL Output- -207.615219+-239.438599IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
```

```
C:\Xilinx\SDK\2019.1\bin\unw x + v
NOTE: This is a line-buffered console and you have to press "Enter"
to send a string of characters to DCC/MDM.

PS Output- 770.000000+758.000000I, PL Output- 770.000000+758.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 125.840637+-89.330933I, PL Output- 125.840622+-89.330948IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -468.000000+-8.000000I, PL Output- -468.000000+-8.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -244.384781+-72.561371I, PL Output- -244.384781+-72.561394IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 210.000000+162.000000I, PL Output- 210.000000+162.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 38.159367+-182.669067I, PL Output- 38.159378+-182.669052IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -48.000000+40.000000I, PL Output- -48.000000+40.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -207.615219+-239.438629I, PL Output- -207.615219+-239.438599IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!

- Execution Time Comparison--
Execution time for PS in micro-seconds: 11.415384
Execution time for PL in micro-seconds: 5.273846
```



## (ZOOMED OUTPUT)

```
PS Output- 770.000000+758.000000I, PL Output- 770.000000+758.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 125.840637+-89.330933I, PL Output- 125.840622+-89.330948IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -468.000000+-8.000000I, PL Output- -468.000000+-8.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -244.384781+-72.561371I, PL Output- -244.384781+-72.561394IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 210.000000+162.000000I, PL Output- 210.000000+162.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- 38.159367+-182.669067I, PL Output- 38.159378+-182.669052IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -48.000000+40.000000I, PL Output- -48.000000+40.000000IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!
PS Output- -207.615219+-239.438629I, PL Output- -207.615219+-239.438599IDMA Transfer Successful!
PS Output- 0.000000+0.000000I, PL Output- 0.000000+0.000000IDMA Transfer Successful!

- Execution Time Comparison--

Execution time for PS in micro-seconds: 11.415384
Execution time for PL in micro-seconds: 5.273846
```

## Expected output from Matlab

```
>> x = [11 + 23i, 32 + 10i, 91 + 94i, 15 + 69i, 47 + 96i, 44 + 12i, 96 + 17i, 49 + 58i, 11 + 23i, 32 + 10i, 91 + 94i, 15 + 69i, 47 + 96i, 44 + 12i, 96 + 17i, 49 + 58i];
>> fft(x)

ans =

    1.0e+02 *

Columns 1 through 9

    7.7000 + 7.5800i    0.0000 + 0.0000i    1.2584 - 0.8933i    0.0000 + 0.0000i   -4.6800 - 0.0800i    0.0000 + 0.0000i   -2.4438 - 0.7256i    0.0000 + 0.0000i    2.1000 + 1.6200i

Columns 10 through 16

    0.0000 + 0.0000i    0.3816 - 1.8267i    0.0000 + 0.0000i   -0.4800 + 0.4000i    0.0000 + 0.0000i   -2.0762 - 2.3944i    0.0000 + 0.0000i
```

**Thank You**