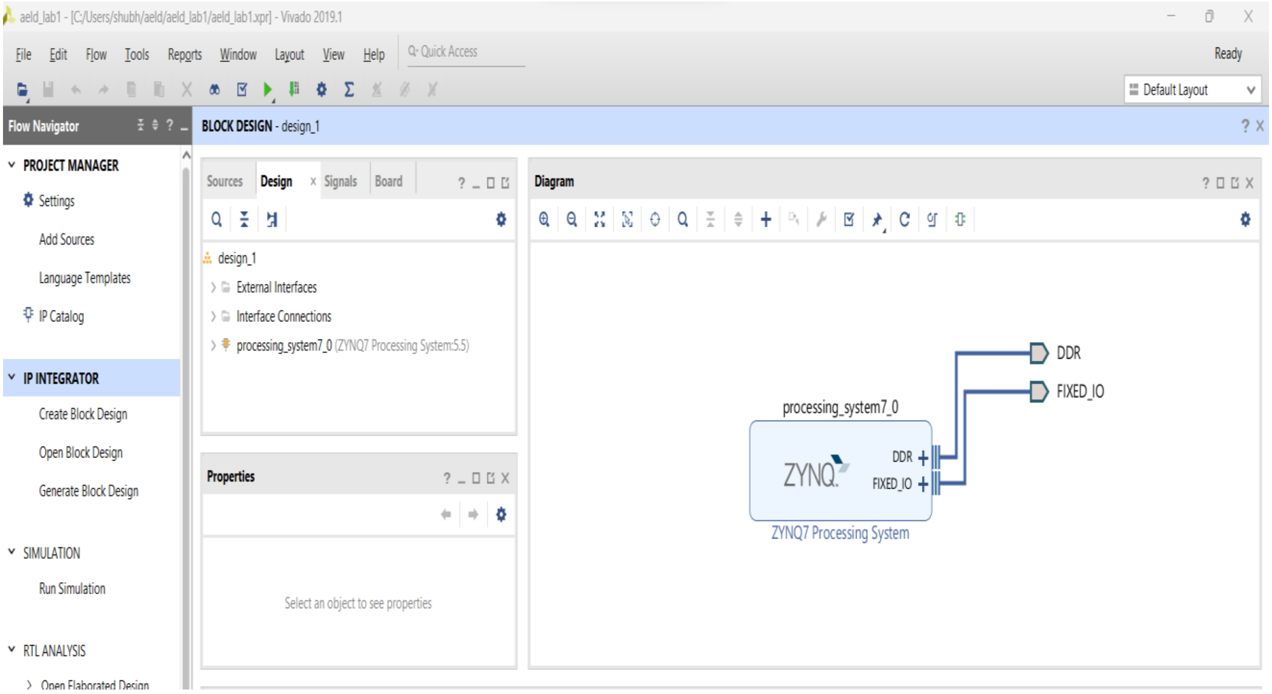
LAB ASSIGNMENT -2

AELD

NAME – ARYAN GUPTA

ROLL NO – MT22154

BLOCK DESIGN

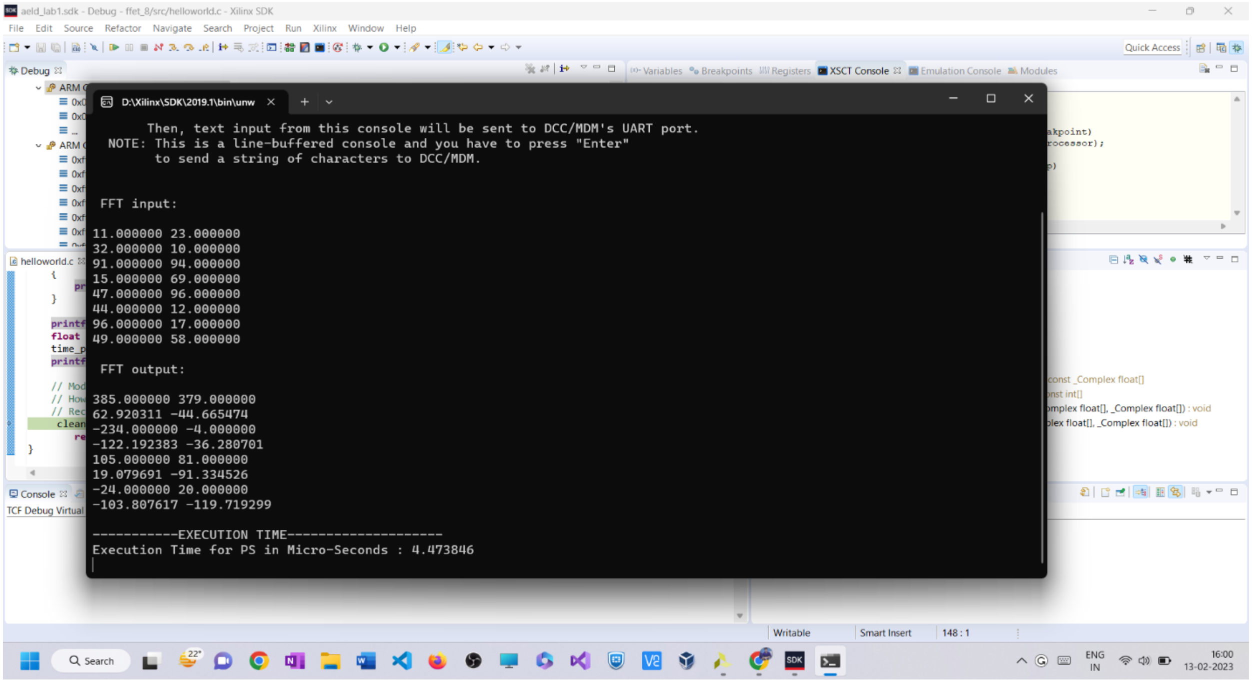


8 FFT DESIGN

```
#include <stdio.h>
#include <stdlib.h>
#include <complex.h>
#include <xtime_l.h> // Timer for execution time calculations
#include "platform.h"
#include "xil_printf.h"

#define FFT_Size 8


const float complex twiddle_factors[FFT_Size/2] = {1-0*I, 0.7071067811865476-0.7071067811865475*I, 0.0-1*I,  -0.7071067811865475-0.7071067811865476*I};

// This function reorders the input to get the output in the normal order
// Refer the handout for the desired input order
```

```
const int input_reorder[FFT_Size] = {0, 4, 2, 6, 1, 5, 3, 7};
void InputReorder(float complex dataIn[FFT_Size], float complex
dataOut[FFT_Size])
{
    for (int i = 0 ; i < FFT_Size ; i++)
    {
        dataOut[i] = dataIn[input_reorder[i]];
    }
}

// For FFT of size FFT_Size, the number of butterfly stages are 2^stages =
FFT_Size.
// For 8-point FFT, there are three butterfly stages.
void FFTStages(float complex FFT_input[FFT_Size], float complex
FFT_output[FFT_Size])
{
    float complex stage1_out[FFT_Size], stage2_out[FFT_Size];
    // Stage 1
    for (int i = 0 ; i < FFT_Size ; i=i+2)
    {
        stage1_out[i] = FFT_input[i] + FFT_input[i+1];
        stage1_out[i+1] = FFT_input[i] - FFT_input[i+1];
    }
    // Stage 2
    for (int i = 0 ; i < FFT_Size ; i=i+4)
    {
        for (int j = 0; j < 2; ++j)
        {
            stage2_out[i+j] = stage1_out[i+j] +
twiddle_factors[2*j]*stage1_out[i+j+2];
            stage2_out[i+2+j] = stage1_out[i+j] -
twiddle_factors[2*j]*stage1_out[i+j+2];
        }

    }
    // Stage 3
    for (int i = 0 ; i < FFT_Size/2 ; i++)
    {
        FFT_output[i] = stage2_out[i] + twiddle_factors[i]*stage2_out[i+4];
        FFT_output[i+4] = stage2_out[i] - twiddle_factors[i]*stage2_out[i+4];
    }

}

//////////////////////////////////////////////////////////////////////////////
/////////////////////////////////
int main()
{
```

```
    init_platform();
    // For FFT_Size point FFT, define the input.
    // You may modify the code to take the input from user via UART
    float complex FFT_input[FFT_Size] =
{11+23*I,32+10*I,91+94*I,15+69*I,47+96*I,44+12*I,96+17*I,49+58*I};
    // FFT output will be stored in this variable
    float complex FFT_output[FFT_Size];
    // Variable for intermediate outputs
    float complex FFT_rev[FFT_Size];

    // to store the time at which certain processes starts and ends
    XTime time_PS_start , time_PS_end;

    // Print the FFT input on the UART
    printf("\n FFT input: \r\n");
      for (int i = 0 ; i < FFT_Size ; i++)
      {
          printf("%f %f\n", crealf(FFT_input[i]), cimagf(FFT_input[i]));
      }
     XTime_SetTime(0);
     XTime_GetTime(&time_PS_start); // Capture the timer value at the start
    // As discussed in the handout, FFT involves two tasks:
    // 1) Reorder of the inputs to get output in the normal order
    // 2) Multiplications using multi-stage butterfly approach
    InputReorder(FFT_input, FFT_rev); // Task 1
    FFTStages(FFT_rev, FFT_output); // Task 2
    XTime_GetTime(&time_PS_end); // Capture the timer value at the start

    // Print the FFT output on the UART
    printf("\n FFT output: \r\n");
    for (int i = 0 ; i < FFT_Size ; i++)
    {
        printf("%f %f\n", crealf(FFT_output[i]), cimagf(FFT_output[i]));
    }

    printf("\n-----------EXECUTION TIME--------------------\n");
    float time_processor = 0;
    time_processor = (float)1.0 * (time_PS_end - time_PS_start) /
(COUNTS_PER_SECOND/1000000);
    printf("Execution Time for PS in Micro-Seconds : %f\n" , time_processor);

    // Modify this code for large size FFT
    // How you can generalize the code for any FFT size (limited to power of
two)
    // Receive the FFT size and FFT input from User
     cleanup_platform();
        return 0;
}
```



Time for PS –> 4.473846 MicroSeconds

1024 FFT DESIGN

```
#ifndef SRC_FFTSW_H_
#define SRC_FFTSW_H_

#define FFTSIZE 1024        /* FFTSIZE OF FFT */
#define FFTSTAGES 10            /* Number of Stages = Log2N */

typedef float DTYPE;
typedef unsigned int INTTYPE;

// For N-point FFT, it consists of N/2 weights (real part)
// For 4-point FFT, it consists of W_4^0,W_4^1
// For 8-point FFT, it consists of W_8^0,W_8^1,W_8^2,W_8^3
const DTYPE W_real[]={1.0, 0.9999811752826011, 0.9999247018391445,
0.9998305817958234, 0.9996988186962042, 0.9995294175010931,
0.9993223845883495, 0.9990777277526454, 0.9987954562051724,
0.9984755805732948, 0.9981181129001492, 0.9977230666441916,
0.9972904566786902, 0.9968202992911657, 0.996312612182778, 0.9957674144676598,
0.9951847266721969, 0.9945645707342554, 0.9939069700023561,
0.9932119492347945, 0.99247953459871, 0.9917097536690995, 0.99090263542778,
0.9900582102622971, 0.989176509964781, 0.9882575677307495, 0.9873014181578584,
0.9863080972445987, 0.9852776423889412, 0.984210092386929, 0.9831054874312163,
0.9819638691095552, 0.9807852804032304, 0.9795697656854405,
0.9783173707196277, 0.9770281426577544, 0.9757021300385286,
0.9743393827855759, 0.9729399522055602, 0.9715038909862518, 0.970031253194544,
0.9685220942744174, 0.9669764710448521, 0.9653944416976894,
```

```
0.9637760657954398, 0.9621214042690416, 0.9604305194155658,
0.9587034748958716, 0.9569403357322088, 0.9551411683057708,
0.9533060403541939, 0.9514350209690083, 0.9495281805930367,
0.9475855910177411, 0.9456073253805213, 0.9435934581619604,
0.9415440651830208, 0.9394592236021899, 0.937339011912575, 0.9351835099389476,
0.932992798834739, 0.9307669610789837, 0.9285060804732156, 0.9262102421383114,
0.9238795325112867, 0.921514039342042, 0.9191138516900578, 0.9166790599210427,
0.9142097557035307, 0.9117060320054299, 0.9091679830905224,
0.9065957045149153, 0.9039892931234433, 0.901348847046022, 0.8986744656939538,
0.8959662497561852, 0.8932243011955153, 0.8904487232447579,
0.8876396204028539, 0.8847970984309378, 0.881921264348355, 0.8790122264286335,
0.8760700941954066, 0.8730949784182901, 0.8700869911087115,
0.8670462455156926, 0.8639728561215868, 0.8608669386377673,
0.8577286100002721, 0.8545579883654005, 0.8513551931052652,
0.8481203448032972, 0.8448535652497071, 0.8415549774368984,
0.8382247055548381, 0.83486287498638, 0.8314696123025452, 0.8280450452577558,
0.8245893027850253, 0.8211025149911046, 0.8175848131515837,
0.8140363297059484, 0.8104571982525948, 0.8068475535437993,
0.8032075314806449, 0.799537269107905, 0.7958369046088836, 0.7921065773002124,
0.7883464276266063, 0.7845565971555752, 0.7807372285720945,
0.7768884656732324, 0.773010453362737, 0.7691033376455797, 0.765167265622459,
0.7612023854842618, 0.7572088465064846, 0.7531867990436125,
0.7491363945234594, 0.7450577854414661, 0.7409511253549591,
0.7368165688773699, 0.7326542716724128, 0.7284643904482252, 0.724247082951467,
0.7200025079613817, 0.7157308252838186, 0.7114321957452164,
0.7071067811865476, 0.7027547444572253, 0.6983762494089729, 0.693971460889654,
0.6895405447370669, 0.6850836677727004, 0.6806009977954531, 0.676092703575316,
0.6715589548470183, 0.6669999223036375, 0.6624157775901718,
0.6578066932970786, 0.6531728429537768, 0.6485144010221126,
0.6438315428897915, 0.6391244448637757, 0.6343932841636455,
0.6296382389149271, 0.6248594881423865, 0.6200572117632892,
0.6152315905806268, 0.6103828062763095, 0.6055110414043255, 0.600616479383869,
0.5956993044924335, 0.5907597018588743, 0.5857978574564389,
0.5808139580957645, 0.5758081914178453, 0.5707807458869674,
0.5657318107836132, 0.560661576197336, 0.5555702330196023, 0.5504579729366048,
0.5453249884220465, 0.540171472729893, 0.5349976198870973, 0.5298036246862948,
0.5245896826784688, 0.5193559901655895, 0.5141027441932217, 0.508830142543107,
0.5035383837257176, 0.49822766697278187, 0.4928981922297841,
0.48755016014843605, 0.48218377207912283, 0.47679923006332225,
0.4713967368259978, 0.4659764957679661, 0.46053871095824, 0.45508358712634384,
0.4496113296546066, 0.44412214457042926, 0.4386162385385277,
0.433093818853152, 0.4275550934302822, 0.4220002707997998, 0.4164295600976373,
0.4108431710579039, 0.40524131400498986, 0.3996241998456468,
0.3939920400610481, 0.3883450466988263, 0.38268343236508984,
0.3770074102164183, 0.3713171939518376, 0.36561299780477396,
0.3598950365349883, 0.3541635254204905, 0.3484186802494345,
0.3426607173119944, 0.33688985339222005, 0.33110630575987643,
0.325310292162263, 0.31950203081601575, 0.3136817403988916, 0.30784964004153535,
```

```
0.3020059493192282, 0.29615088824362396, 0.29028467725446233,
0.2844075372112718, 0.27851968938505306, 0.272621355449949,
0.2667127574748984, 0.26079411791527557, 0.2548656596045146,
0.24892760574572026, 0.24298017990326398, 0.23702360599436734,
0.23105810828067128, 0.22508391135979278, 0.21910124015686977,
0.21311031991609136, 0.20711137619221856, 0.20110463484209196,
0.19509032201612833, 0.18906866414980628, 0.18303988795514106,
0.17700422041214886, 0.17096188876030136, 0.1649131204899701,
0.1588581433338614, 0.1527971852584434, 0.14673047445536175,
0.14065823933284924, 0.13458070850712622, 0.12849811079379322,
0.12241067519921628, 0.11631863091190488, 0.11022220729388318,
0.10412163387205473, 0.09801714032956077, 0.0919089564971327,
0.08579731234443988, 0.07968243797143013, 0.07356456359966745,
0.0674439195636641, 0.06132073630220865, 0.05519524434969003,
0.049067674327418126, 0.0429382569349409, 0.03680722294135899,
0.03067480317663658, 0.024541228522912264, 0.01840672990580482,
0.012271538285719944, 0.006135884649154515, 6.123233995736766e-17, -
0.006135884649154393, -0.012271538285719823, -0.018406729905804695, -
0.024541228522912142, -0.03067480317663646, -0.036807222941358866, -
0.042938256934940834, -0.04906767432741801, -0.05519524434968991, -
0.06132073630220853, -0.06744391956366398, -0.07356456359966733, -
0.07968243797143001, -0.08579731234443976, -0.09190895649713257, -
0.09801714032956065, -0.1041216338720546, -0.11022220729388306, -
0.11631863091190475, -0.12241067519921615, -0.1284981107937931, -
0.1345807085071261, -0.14065823933284913, -0.14673047445536164, -
0.152797185258443, -0.15885814333386128, -0.16491312048996995, -
0.17096188876030124, -0.17700422041214875, -0.18303988795514092, -
0.18906866414980616, -0.1950903220161282, -0.20110463484209182, -
0.20711137619221845, -0.21311031991609125, -0.21910124015686966, -
0.22508391135979267, -0.23105810828067114, -0.23702360599436723, -
0.24298017990326387, -0.24892760574572012, -0.2548656596045145, -
0.26079411791527546, -0.2667127574748983, -0.27262135544994887, -
0.27851968938505295, -0.2844075372112717, -0.29028467725446216, -
0.29615088824362384, -0.3020059493192281, -0.30784964004153487, -
0.3136817403988914, -0.31950203081601564, -0.32531029216226287, -
0.3311063057598763, -0.33688985339221994, -0.34266071731199427, -
0.3484186802494344, -0.3541635254204904, -0.35989503653498817, -
0.36561299780477385, -0.37131719395183755, -0.3770074102164182, -
0.3826834323650897, -0.3883450466988262, -0.393992040061048, -
0.3996241998456467, -0.40524131400498975, -0.4108431710579038, -
0.416429560097637, -0.4220002707997997, -0.42755509343028186, -
0.4330938188531519, -0.4386162385385274, -0.44412214457042914, -
0.4496113296546067, -0.4550835871263437, -0.46053871095824006, -
0.465976495767966, -0.4713967368259977, -0.4767992300633219, -
0.4821837720791227, -0.4875501601484357, -0.492898192229784, -
0.4982276669727816, -0.5035383837257175, -0.5088301425431071, -
0.5141027441932217, -0.5193559901655896, -0.5245896826784687, -
0.5298036246862947, -0.534997619887097, -0.5401714727298929, -
```

```
0.5453249884220462, -0.5504579729366047, -0.555570233019602, -
0.5606615761973359, -0.5657318107836132, -0.5707807458869671, -
0.5758081914178453, -0.5808139580957644, -0.5857978574564389, -
0.590759701858874, -0.5956993044924334, -0.6006164793838688, -
0.6055110414043254, -0.6103828062763096, -0.6152315905806267, -
0.6200572117632892, -0.6248594881423862, -0.6296382389149271, -
0.6343932841636454, -0.6391244448637757, -0.6438315428897913, -
0.6485144010221124, -0.6531728429537765, -0.6578066932970786, -
0.6624157775901719, -0.6669999223036374, -0.6715589548470184, -
0.6760927035753158, -0.680600997795453, -0.6850836677727002, -
0.6895405447370669, -0.6939714608896538, -0.6983762494089728, -
0.7027547444572251, -0.7071067811865475, -0.7114321957452165, -
0.7157308252838186, -0.7200025079613817, -0.7242470829514668, -
0.7284643904482252, -0.7326542716724127, -0.7368165688773699, -
0.7409511253549589, -0.745057785441466, -0.7491363945234591, -
0.7531867990436124, -0.7572088465064846, -0.7612023854842617, -
0.765167265622459, -0.7691033376455795, -0.773010453362737, -
0.7768884656732323, -0.7807372285720945, -0.784556597155575, -
0.7883464276266062, -0.7921065773002122, -0.7958369046088835, -
0.7995372691079051, -0.8032075314806448, -0.8068475535437993, -
0.8104571982525947, -0.8140363297059484, -0.8175848131515836, -
0.8211025149911046, -0.8245893027850251, -0.8280450452577557, -
0.8314696123025453, -0.83486287498638, -0.8382247055548381, -
0.8415549774368983, -0.8448535652497071, -0.8481203448032971, -
0.8513551931052652, -0.8545579883654004, -0.857728610000272, -
0.8608669386377671, -0.8639728561215867, -0.8670462455156928, -
0.8700869911087113, -0.8730949784182901, -0.8760700941954065, -
0.8790122264286335, -0.8819212643483549, -0.8847970984309378, -
0.8876396204028538, -0.8904487232447579, -0.8932243011955152, -
0.8959662497561851, -0.8986744656939539, -0.9013488470460219, -
0.9039892931234433, -0.9065957045149153, -0.9091679830905224, -
0.9117060320054298, -0.9142097557035307, -0.9166790599210426, -
0.9191138516900578, -0.9215140393420418, -0.9238795325112867, -
0.9262102421383114, -0.9285060804732155, -0.9307669610789837, -
0.9329927988347388, -0.9351835099389476, -0.9373390119125748, -
0.9394592236021899, -0.9415440651830207, -0.9435934581619604, -
0.9456073253805212, -0.9475855910177411, -0.9495281805930367, -
0.9514350209690083, -0.9533060403541939, -0.9551411683057707, -
0.9569403357322088, -0.9587034748958715, -0.9604305194155658, -
0.9621214042690415, -0.9637760657954398, -0.9653944416976893, -
0.9669764710448521, -0.9685220942744174, -0.970031253194544, -
0.9715038909862518, -0.9729399522055601, -0.9743393827855759, -
0.9757021300385285, -0.9770281426577544, -0.9783173707196275, -
0.9795697656854405, -0.9807852804032304, -0.9819638691095552, -
0.9831054874312163, -0.984210092386929, -0.9852776423889412, -
0.9863080972445986, -0.9873014181578584, -0.9882575677307495, -
0.989176509964781, -0.990058210262297, -0.99090263542778, -0.9917097536690995,
-0.99247953459871, -0.9932119492347945, -0.9939069700023561, -
```

```
0.9945645707342554, -0.9951847266721968, -0.9957674144676598, -
0.996312612182778, -0.9968202992911657, -0.9972904566786902, -
0.9977230666441916, -0.9981181129001492, -0.9984755805732948, -
0.9987954562051724, -0.9990777277526454, -0.9993223845883495, -
0.9995294175010931, -0.9996988186962042, -0.9998305817958234, -
0.9999247018391445, -0.9999811752826011};

// For N-point FFT, it consists of N/2 weights (imag part)
// For 4-point FFT, it consists of W_4^0,W_4^1
// For 8-point FFT, it consists of W_8^0,W_8^1,W_8^2,W_8^3
const DTYPE W_imag[]={-0.0, -0.006135884649154475, -0.012271538285719925, -
0.01840672990580482, -0.024541228522912288, -0.030674803176636626, -
0.03680722294135883, -0.04293825693494082, -0.049067674327418015, -
0.05519524434968994, -0.06132073630220858, -0.06744391956366405, -
0.07356456359966743, -0.07968243797143013, -0.0857973123444399, -
0.09190895649713272, -0.0980171403295606, -0.10412163387205459, -
0.11022220729388306, -0.11631863091190475, -0.1224106751992162, -
0.12849811079379317, -0.13458070850712617, -0.1406582393328492, -
0.14673047445536175, -0.15279718525844344, -0.15885814333386145, -
0.1649131204899699, -0.17096188876030122, -0.17700422041214875, -
0.18303988795514095, -0.1890686641498062, -0.19509032201612825, -
0.2011046348420919, -0.20711137619221856, -0.21311031991609136, -
0.2191012401568698, -0.22508391135979283, -0.2310581082806711, -
0.2370236059943672, -0.24298017990326387, -0.24892760574572015, -
0.25486565960451457, -0.2607941179152755, -0.26671275747489837, -
0.272621355449949, -0.27851968938505306, -0.2844075372112719, -
0.29028467725446233, -0.2961508882436238, -0.3020059493192281, -
0.30784964004153487, -0.3136817403988915, -0.3195020308160157, -
0.3253102921622629, -0.33110630575987643, -0.33688985339222005, -
0.3426607173119944, -0.34841868024943456, -0.35416352542049034, -
0.3598950365349881, -0.36561299780477385, -0.37131719395183754, -
0.37700741021641826, -0.3826834323650898, -0.38834504669882625, -
0.3939920400610481, -0.3996241998456468, -0.40524131400498986, -
0.4108431710579039, -0.41642956009763715, -0.4220002707997997, -
0.4275550934302821, -0.43309381885315196, -0.43861623853852766, -
0.4441221445704292, -0.44961132965460654, -0.45508358712634384, -
0.46053871095824, -0.4659764957679662, -0.47139673682599764, -
0.4767992300633221, -0.4821837720791227, -0.48755016014843, -
0.49289819222978404, -0.49822766697278187, -0.5035383837257176, -
0.508830142543107, -0.5141027441932217, -0.5193559901655896, -
0.524589682678469, -0.5298036246862946, -0.5349976198870972, -
0.5401714727298929, -0.5453249884220465, -0.5504579729366048, -
0.5555702330196022, -0.560661576197336, -0.5657318107836131, -
0.5707807458869673, -0.5758081914178453, -0.5808139580957645, -
0.5857978574564389, -0.5907597018588742, -0.5956993044924334, -
0.600616479383869, -0.6055110414043255, -0.6103828062763095, -
0.6152315905806268, -0.6200572117632891, -0.6248594881423863, -
0.629638238914927, -0.6343932841636455, -0.6391244448637757, -
```

```
0.6438315428897914, -0.6485144010221124, -0.6531728429537768, -
0.6578066932970786, -0.6624157775901718, -0.6669999223036375, -
0.6715589548470183, -0.6760927035753159, -0.680600997795453, -
0.6850836677727004, -0.6895405447370668, -0.693971460889654, -
0.6983762494089729, -0.7027547444572253, -0.7071067811865475, -
0.7114321957452164, -0.7157308252838186, -0.7200025079613817, -
0.7242470829514669, -0.7284643904482252, -0.7326542716724128, -
0.7368165688773698, -0.7409511253549591, -0.745057785441466, -
0.7491363945234593, -0.7531867990436124, -0.7572088465064845, -
0.7612023854842618, -0.765167265622459, -0.7691033376455796, -
0.773010453362737, -0.7768884656732324, -0.7807372285720944, -
0.7845565971555752, -0.7883464276266062, -0.7921065773002124, -
0.7958369046088835, -0.799537269107905, -0.8032075314806448, -
0.8068475535437992, -0.8104571982525948, -0.8140363297059483, -
0.8175848131515837, -0.8211025149911046, -0.8245893027850253, -
0.8280450452577558, -0.8314696123025452, -0.83486287498638, -
0.838224705554838, -0.8415549774368983, -0.844853565249707, -
0.8481203448032971, -0.8513551931052652, -0.8545579883654005, -
0.8577286100002721, -0.8608669386377673, -0.8639728561215867, -
0.8670462455156926, -0.8700869911087113, -0.8730949784182901, -
0.8760700941954066, -0.8790122264286334, -0.8819212643483549, -
0.8847970984309378, -0.8876396204028539, -0.8904487232447579, -
0.8932243011955153, -0.8959662497561851, -0.8986744656939538, -
0.901348847046022, -0.9039892931234433, -0.9065957045149153, -
0.9091679830905223, -0.9117060320054299, -0.9142097557035307, -
0.9166790599210427, -0.9191138516900578, -0.9215140393420419, -
0.9238795325112867, -0.9262102421383113, -0.9285060804732155, -
0.9307669610789837, -0.9329927988347388, -0.9351835099389475, -
0.937339011912575, -0.9394592236021899, -0.9415440651830208, -
0.9435934581619604, -0.9456073253805213, -0.9475855910177411, -
0.9495281805930367, -0.9514350209690083, -0.9533060403541938, -
0.9551411683057707, -0.9569403357322089, -0.9587034748958716, -
0.9604305194155658, -0.9621214042690416, -0.9637760657954398, -
0.9653944416976894, -0.9669764710448521, -0.9685220942744173, -
0.970031253194544, -0.9715038909862518, -0.9729399522055601, -
0.9743393827855759, -0.9757021300385286, -0.9770281426577544, -
0.9783173707196277, -0.9795697656854405, -0.9807852804032304, -
0.9819638691095552, -0.9831054874312163, -0.984210092386929, -
0.9852776423889412, -0.9863080972445987, -0.9873014181578584, -
0.9882575677307495, -0.989176509964781, -0.9900582102622971, -
0.99090263542778, -0.9917097536690995, -0.99247953459871, -0.9932119492347945,
-0.9939069700023561, -0.9945645707342554, -0.9951847266721968, -
0.9957674144676598, -0.996312612182778, -0.9968202992911657, -
0.9972904566786902, -0.9977230666441916, -0.9981181129001492, -
0.9984755805732948, -0.9987954562051724, -0.9990777277526454, -
0.9993223845883495, -0.9995294175010931, -0.9996988186962042, -
0.9998305817958234, -0.9999247018391445, -0.9999811752826011, -1.0, -
0.9999811752826011, -0.9999247018391445, -0.9998305817958234, -
```

0.9996988186962042, -0.9995294175010931, -0.9993223845883495, -
0.9990777277526454, -0.9987954562051724, -0.9984755805732948, -
0.9981181129001492, -0.9977230666441916, -0.9972904566786902, -
0.9968202992911658, -0.996312612182778, -0.9957674144676598, -
0.9951847266721969, -0.9945645707342554, -0.9939069700023561, -
0.9932119492347945, -0.99247953459871, -0.9917097536690995, -0.99090263542778,
-0.9900582102622971, -0.989176509964781, -0.9882575677307495, -
0.9873014181578584, -0.9863080972445987, -0.9852776423889412, -
0.984210092386929, -0.9831054874312163, -0.9819638691095552, -
0.9807852804032304, -0.9795697656854405, -0.9783173707196277, -
0.9770281426577544, -0.9757021300385286, -0.9743393827855759, -
0.9729399522055602, -0.9715038909862518, -0.970031253194544, -
0.9685220942744174, -0.9669764710448521, -0.9653944416976894, -
0.9637760657954398, -0.9621214042690416, -0.9604305194155659, -
0.9587034748958716, -0.9569403357322089, -0.9551411683057707, -
0.9533060403541939, -0.9514350209690083, -0.9495281805930367, -
0.9475855910177412, -0.9456073253805214, -0.9435934581619604, -
0.9415440651830208, -0.9394592236021899, -0.937339011912575, -
0.9351835099389476, -0.9329927988347388, -0.9307669610789837, -
0.9285060804732156, -0.9262102421383114, -0.9238795325112867, -
0.921514039342042, -0.9191138516900578, -0.9166790599210427, -
0.9142097557035307, -0.9117060320054299, -0.9091679830905225, -
0.9065957045149153, -0.9039892931234434, -0.901348847046022, -
0.8986744656939539, -0.8959662497561852, -0.8932243011955152, -
0.890448723244758, -0.8876396204028539, -0.8847970984309379, -
0.881921264348355, -0.8790122264286335, -0.8760700941954066, -
0.8730949784182902, -0.8700869911087115, -0.8670462455156928, -
0.8639728561215868, -0.8608669386377672, -0.8577286100002721, -
0.8545579883654005, -0.8513551931052652, -0.8481203448032972, -
0.8448535652497072, -0.8415549774368984, -0.8382247055548382, -
0.8348628749863801, -0.8314696123025455, -0.8280450452577558, -
0.8245893027850252, -0.8211025149911048, -0.8175848131515837, -
0.8140363297059485, -0.8104571982525948, -0.8068475535437994, -
0.8032075314806449, -0.7995372691079052, -0.7958369046088836, -
0.7921065773002123, -0.7883464276266063, -0.7845565971555751, -
0.7807372285720946, -0.7768884656732324, -0.7730104533627371, -
0.7691033376455796, -0.7651672656224591, -0.7612023854842619, -
0.7572088465064847, -0.7531867990436125, -0.7491363945234593, -
0.7450577854414661, -0.740951125354959, -0.73681656887737, -
0.7326542716724128, -0.7284643904482253, -0.7242470829514669, -
0.7200025079613818, -0.7157308252838187, -0.7114321957452167, -
0.7071067811865476, -0.7027547444572252, -0.6983762494089729, -
0.693971460889654, -0.689540544737067, -0.6850836677727004, -
0.6806009977954532, -0.6760927035753159, -0.6715589548470186, -
0.6669999223036376, -0.662415777590172, -0.6578066932970787, -
0.6531728429537766, -0.6485144010221126, -0.6438315428897914, -
0.6391244448637758, -0.6343932841636455, -0.6296382389149272, -
0.6248594881423863, -0.6200572117632894, -0.6152315905806269, -

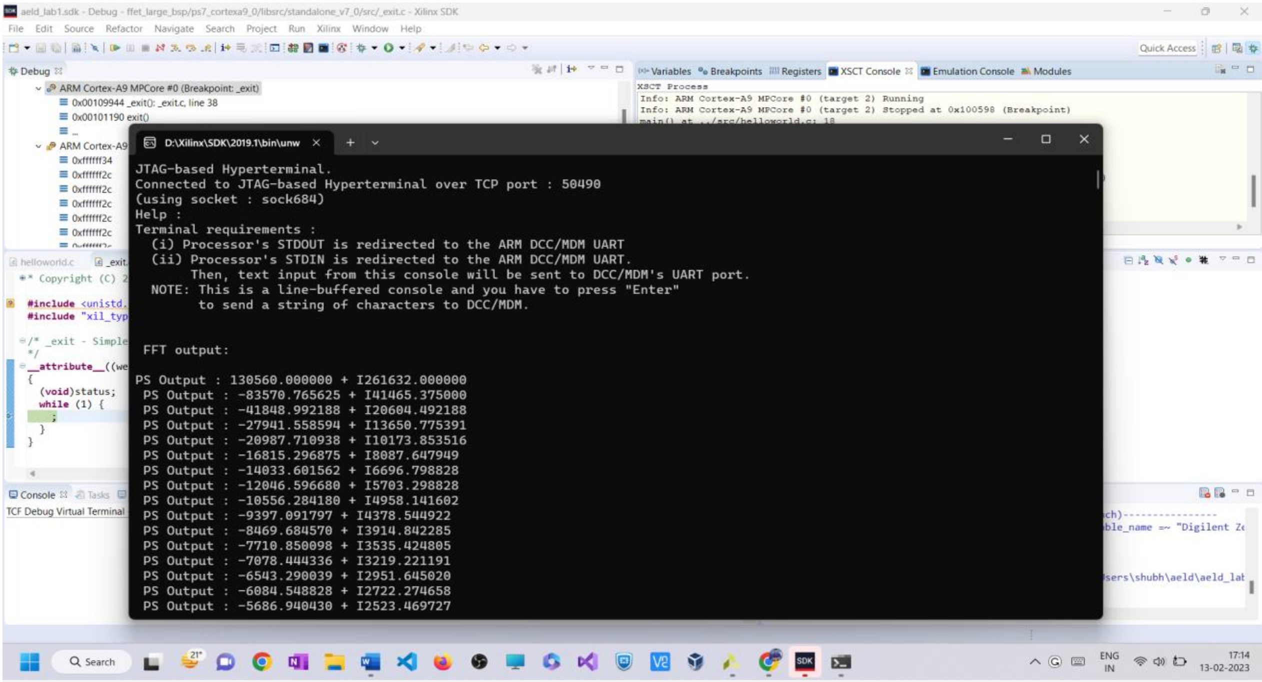
```
0.6103828062763097, -0.6055110414043257, -0.6006164793838689, -
0.5956993044924335, -0.5907597018588742, -0.585797857456439, -
0.5808139580957645, -0.5758081914178454, -0.5707807458869673, -
0.5657318107836135, -0.5606615761973361, -0.5555702330196022, -
0.5504579729366049, -0.5453249884220464, -0.540171472729893, -
0.5349976198870972, -0.5298036246862948, -0.524589682678469, -
0.5193559901655898, -0.5141027441932218, -0.5088301425431073, -
0.5035383837257177, -0.49822766697278176, -0.49289819222978415, -
0.4875501601484359, -0.4821837720791229, -0.4767992300633221, -
0.47139673682599786, -0.4659764957679662, -0.4605387109582402, -
0.4550835871263439, -0.4496113296546069, -0.4441221445704293, -
0.43861623853852755, -0.43309381885315207, -0.42755509343028203, -
0.4220002707997985, -0.41642956009763715, -0.41084317105790413, -
0.4052413140049899, -0.39962419984564707, -0.39399204006104815, -
0.3883450466988266, -0.3826834323650899, -0.37700741021641815, -
0.3713171939518377, -0.3656129978047738, -0.35989503653498833, -
0.3541635254204904, -0.3484186802494348, -0.34266071731199443, -
0.33688985339222033, -0.3311063057598765, -0.32531029216226326, -
0.3195020308160158, -0.3136817403988914, -0.30784964004153503, -
0.30200594931922803, -0.296150888243624, -0.2902846772544624, -
0.2844075372112721, -0.27851968938505317, -0.27262135544994925, -
0.2667127574748985, -0.26079411791527585, -0.2548656596045147, -
0.2489276057457201, -0.24298017990326407, -0.23702360599436717, -
0.23105810828067133, -0.22508391135979283, -0.21910124015687005, -
0.21311031991609142, -0.20711137619221884, -0.201104634842092, -
0.1950903220161286, -0.18906866414980636, -0.1830398879551409, -
0.17700422041214894, -0.17096188876030122, -0.16491312048997014, -
0.15885814333386147, -0.15279718525844369, -0.1467304744553618, -
0.14065823933284954, -0.13458070850712628, -0.12849811079379309, -
0.12241067519921635, -0.11631863091190471, -0.11022220729388324, -
0.10412163387205457, -0.09801714032956083, -0.09190895649713275, -
0.08579731234444016, -0.0796824379714302, -0.07356456359966773, -
0.06744391956366418, -0.06132073630220849, -0.05519524434969094, -
0.049067674327417966, -0.04293825693494102, -0.03680722294135883, -
0.030674803176636865, -0.024541228522912326, -0.0184067299058051, -
0.012271538285720007, -0.006135884649154799};


#endif /* SRC_FFTSW_H_ */
```

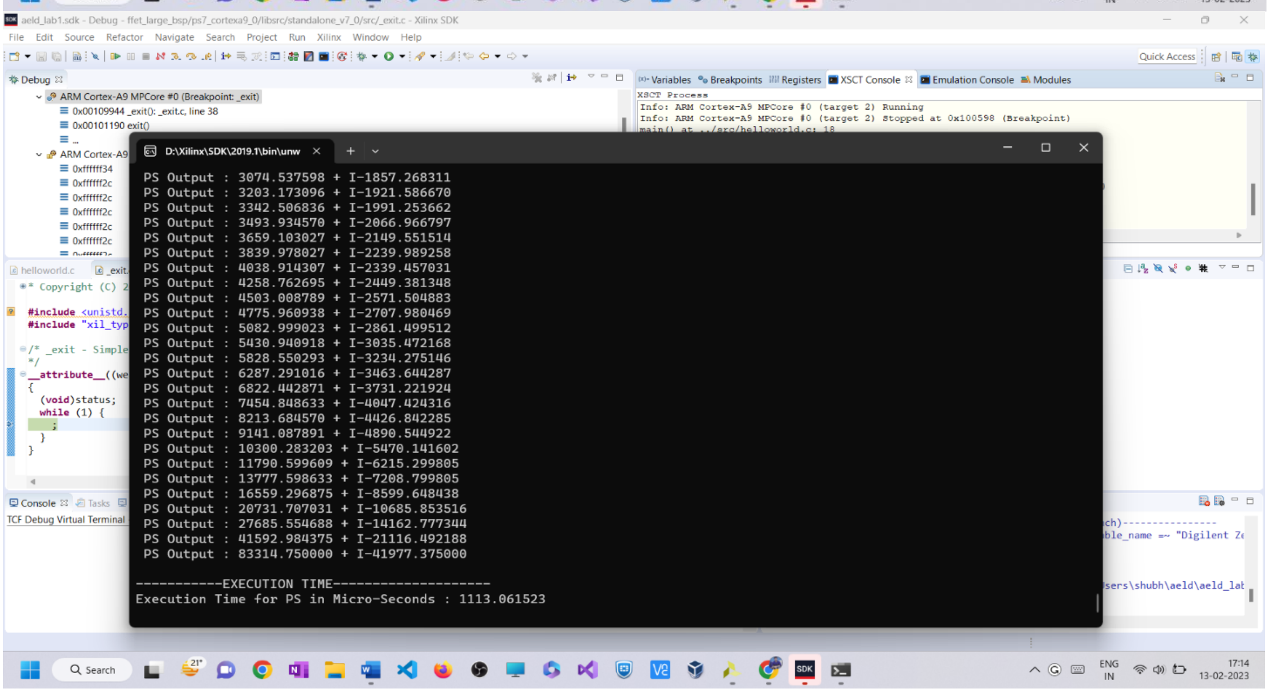
```
#include <stdio.h>
#include <stdbool.h>
#include "platform.h"
#include <stdlib.h>
#include <complex.h>
#include <xtime_l.h>
#include "fftsw.h"
```

```
// This function is designed to perform FFT operation on PS
// Default word length is single precision floating point
// You may explore double precision floating point and effect on execution
time
void FFT_sw(float FFTIn_I[FFTSIZE], float FFTIn_R[FFTSIZE], float
FFTOut_I[FFTSIZE], float FFTOut_R[FFTSIZE]);



int main()
{
  init_platform();
    // Instead of constant input, we will generate the input for FFT
    float DataIN_R[FFTSIZE]; // Real part of the input
    float DataIN_I[FFTSIZE]; // Imaginary part of the input
    // Complex representation of the input
    float complex FFTIn_C[FFTSIZE];
    // input to the FFT in PS and PL
    for (int k = 0; k<FFTSIZE;k++)
    {
      FFTIn_C[k] = (k/4) + (k/2)*I;
      DataIN_R[k] = creal(FFTIn_C[k]);
      DataIN_I[k] = cimag(FFTIn_C[k]);
    }
    //FFT output
    float FFTOut_R[FFTSIZE];
    float FFTOut_I[FFTSIZE];

    // to store the time at which certain processes starts and ends
    XTime time_PS_start , time_PS_end;
    XTime_SetTime(0); // Reset the global timer
    //--------------------- PS PART ---------------------------------//
    XTime_GetTime(&time_PS_start); // Capture the timer value at the start
    FFT_sw(DataIN_I, DataIN_R, FFTOut_I,FFTOut_R);
    XTime_GetTime(&time_PS_end); // Capture the timer value at the start


    // Print the FFT output on the UART
    printf("\n FFT output: \r\n");
    for (int j = 0 ; j < FFTSIZE ; j++)
    {
        printf("PS Output : %f + I%f \n " , FFTOut_R[j], FFTOut_I[j]);
        usleep(0.1); // Always add some buffer time between display
    }


    printf("\n-----------EXECUTION TIME--------------------\n");
    float time_processor = 0;
```

```c
    time_processor = (float)1.0 * (time_PS_end - time_PS_start) /
(COUNTS_PER_SECOND/1000000);
    printf("Execution Time for PS in Micro-Seconds : %f\n" , time_processor);
    cleanup_platform();
    return 0;

}

#define reverse(n) ((n & 0x1) << 9) | ((n & 0x2) << 7) | ((n & 0x4) << 5) |
((n & 0x8) << 3) | ((n & 0x10) << 1) | ((n & 0x20) >> 1) | ((n & 0x40) >> 3) |
((n & 0x80) >> 5) | ((n & 0x100) >> 7) | ((n & 0x200) >> 9)

void FFT_sw(float FFTIn_I[FFTSIZE], float FFTIn_R[FFTSIZE], float
FFTOut_I[FFTSIZE], float FFTOut_R[FFTSIZE])
{

  DTYPE temp_R;   /*temporary storage complex variable*/
  DTYPE temp_I;   /*temporary storage complex variable*/

  int i,j;      /* loop indexes */
  int i_lower;    /* Index of lower point in butterfly */

  int stage;
  int subFFTSize; //Size of FFT in each stage of FFT
  int BFWidth;      /*Butterfly Width*/

  /*====================BEGIN BIT REBERSAL=========================*/
  for (i = 0; i < FFTSIZE; ++i) {
    FFTOut_R[reverse(i)] = FFTIn_R[i];
    FFTOut_I[reverse(i)] = FFTIn_I[i];
  }
  /*+++++++++++++++++++++END OF BIT REVERSAL+++++++++++++++++++++++++*/

  /*======================BEGIN: FFT=======================*/
  // Do FFTSTAGES of butterflies
  DTYPE BFWeight_R, BFWeight_I;
 // For N-point FFT, there are log2(N) stages
  stages:for(stage=1; stage<= FFTSTAGES; stage++)
  {
    subFFTSize = 1 << stage;    // DFT = 2^stage = points in sub DFT
    BFWidth = subFFTSize >> 1;       // Butterfly WIDTHS in sub-FFT (FFTSIZE
of sub-FFT/2) no of weights

    // Perform butterflies for j-th stage
    // This loop runs for the iteration equal to BF width
    // In 4-point FFT, BF width is 1 in stage 1 and 2 in stage 2
    // In 8-point FFT, BF width is 1 in stage 1, 2 in stage 2 and 4 in stage 3
    butterfly:for(j=0; j<BFWidth; j++)
```

```
    {
      //Note that weights of all butterfly units are same in a given stage
      // We can reduce the number of memory read by using this for loop
      BFWeight_R = W_real[j * (FFTSIZE>>stage)];
      BFWeight_I = W_imag[j * (FFTSIZE>>stage)];

    // This loop is for all butterflies in a stage that use same W**k
    // In 4-point FFT, we have two BFs in stage 1
    // In 8-point FFT, we have four BFs in stage 1 and two BFs in stage 2
      // Each butter fly weight affects two outputs and hence we have two
outputs for each iteration
    subDFTSize:for(i =j ; i < FFTSIZE; i += subFFTSize) // This loop runs for
FFTSIZE/SubFFTSize iterations
      {
      i_lower = i + BFWidth;      //index of lower point in butterfly
      temp_R = FFTOut_R[i_lower] * BFWeight_R - FFTOut_I[i_lower] *
BFWeight_I;
      temp_I = FFTOut_I[i_lower] * BFWeight_R + FFTOut_R[i_lower] *
BFWeight_I;

      FFTOut_R[i_lower] = FFTOut_R[i] - temp_R;
      FFTOut_I[i_lower] = FFTOut_I[i] - temp_I;
      FFTOut_R[i] = FFTOut_R[i] + temp_R;
      FFTOut_I[i] = FFTOut_I[i] + temp_I;
      }
  }
  }
}
```





Time for PS -> 1113.061523 MicroSeconds

512 FFT DESIGN

```
#ifndef SRC_FFTSW_H_
#define SRC_FFTSW_H_

#define FFTSIZE 512          /* FFTSIZE OF FFT */
```

```c
#define FFTSTAGES 9              /* Number of Stages = Log2N */



typedef float DTYPE;
typedef unsigned int INTTYPE;
const DTYPE
W_imag[]={0,0.0122715382857199,0.0245412285229123,0.0368072229413588,0.0490676
743274180,0.0613207363022086,0.0735645635996674,0.0857973123444399,0.098017140
3295606,0.110222207293883,0.122410675199216,0.134580708507126,0.14673047445536
2,0.158858143333861,0.170961888760301,0.183039887955141,0.195090322016128,0.20
7111376192219,0.219101240156870,0.231058108280671,0.242980179903264,0.25486565
9604515,0.266712757474898,0.278519689385053,0.290284677254462,0.30200594931922
8,0.313681740398892,0.325310292162263,0.336889853392220,0.348418680249435,0.35
9895036534988,0.371317193951838,0.382683432365090,0.393992040061048,0.40524131
4004990,0.416429560097637,0.427555093430282,0.438616238538528,0.44961132965460
7,0.460538710958240,0.471396736825998,0.482183772079123,0.492898192229784,0.50
3538383725718,0.514102744193222,0.524589682678469,0.534997619887097,0.54532498
8422047,0.555570233019602,0.565731810783613,0.575808191417845,0.58579785745643
9,0.595699304492433,0.605511041404326,0.615231590580627,0.624859488142386,0.63
4393284163646,0.643831542889791,0.653172842953777,0.662415777590172,0.67155895
4847018,0.680600997795453,0.689540544737067,0.698376249408973,0.70710678118654
8,0.715730825283819,0.724247082951467,0.732654271672413,0.740951125354959,0.74
9136394523459,0.757208846506485,0.765167265622459,0.773010453362737,0.78073722
8572094,0.788346427626606,0.795836904608884,0.803207531480645,0.81045719825259
5,0.817584813151584,0.824589302785025,0.831469612302545,0.838224705554838,0.84
4853565249707,0.851355193105265,0.857728610000272,0.863972856121587,0.87008699
1108711,0.876070094195407,0.881921264348355,0.887639620402854,0.89322430119551
5,0.898674465693954,0.903989293123443,0.909167983090522,0.914209755703531,0.91
9113851690058,0.923879532511287,0.928506080473216,0.932992798834739,0.93733901
1912575,0.941544065183021,0.945607325380521,0.949528180593037,0.95330604035419
4,0.956940335732209,0.960430519415566,0.963776065795440,0.966976471044852,0.97
0031253194544,0.972939952205560,0.975702130038529,0.978317370719628,0.98078528
0403230,0.983105487431216,0.985277642388941,0.987301418157858,0.98917650996478
1,0.990902635427780,0.992479534598710,0.993906970002356,0.995184726672197,0.99
6312612182778,0.997290456678690,0.998118112900149,0.998795456205172,0.99932238
4588350,0.999698818696204,0.999924701839145,1,0.999924701839145,0.999698818696
204,0.999322384588350,0.998795456205172,0.998118112900149,0.997290456678690,0.
996312612182778,0.995184726672197,0.993906970002356,0.992479534598710,0.990902
635427780,0.989176509964781,0.987301418157858,0.985277642388941,0.983105487431
216,0.980785280403230,0.978317370719628,0.975702130038529,0.972939952205560,0.
970031253194544,0.966976471044852,0.963776065795440,0.960430519415566,0.956940
335732209,0.953306040354194,0.949528180593037,0.945607325380521,0.941544065183
021,0.937339011912575,0.932992798834739,0.928506080473216,0.923879532511287,0.
919113851690058,0.914209755703531,0.909167983090523,0.903989293123443,0.898674
465693954,0.893224301195515,0.887639620402854,0.881921264348355,0.876070094195
407,0.870086991108712,0.863972856121587,0.857728610000272,0.851355193105265,0.
844853565249707,0.838224705554838,0.831469612302545,0.824589302785025,0.817584
813151584,0.810457198252595,0.803207531480645,0.795836904608884,0.788346427626
```

```
606,0.780737228572095,0.773010453362737,0.765167265622459,0.757208846506485,0.
749136394523459,0.740951125354959,0.732654271672413,0.724247082951467,0.715730
825283819,0.707106781186548,0.698376249408973,0.689540544737067,0.680600997795
453,0.671558954847019,0.662415777590172,0.653172842953777,0.643831542889791,0.
634393284163646,0.624859488142386,0.615231590580627,0.605511041404326,0.595699
304492434,0.585797857456439,0.575808191417845,0.565731810783614,0.555570233019
602,0.545324988422046,0.534997619887097,0.524589682678469,0.514102744193222,0.
503538383725718,0.492898192229784,0.482183772079123,0.471396736825998,0.460538
710958240,0.449611329654607,0.438616238538528,0.427555093430282,0.416429560097
637,0.405241314004990,0.393992040061048,0.382683432365090,0.371317193951838,0.
359895036534988,0.348418680249435,0.336889853392220,0.325310292162263,0.313681
740398891,0.302005949319228,0.290284677254462,0.278519689385053,0.266712757474
899,0.254865659604515,0.242980179903264,0.231058108280671,0.219101240156870,0.
207111376192219,0.195090322016129,0.183039887955141,0.170961888760301,0.158858
143333861,0.146730474455362,0.134580708507126,0.122410675199216,0.110222207293
883,0.0980171403295608,0.0857973123444402,0.0735645635996677,0.061320736302208
5,0.0490676743274180,0.0368072229413588,0.0245412285229123,0.0122715382857200,
1.22464679914735e-16,-0.0122715382857198,-0.0245412285229121,-
0.0368072229413586,-0.0490676743274177,-0.0613207363022082,-
0.0735645635996675,-0.0857973123444399,-0.0980171403295606,-
0.110222207293883,-0.122410675199216,-0.134580708507126,-0.146730474455362,-
0.158858143333861,-0.170961888760301,-0.183039887955141,-0.195090322016128,-
0.207111376192219,-0.219101240156870,-0.231058108280671,-0.242980179903264,-
0.254865659604514,-0.266712757474898,-0.278519689385053,-0.290284677254462,-
0.302005949319228,-0.313681740398891,-0.325310292162263,-0.336889853392220,-
0.348418680249435,-0.359895036534988,-0.371317193951837,-0.382683432365090,-
0.393992040061048,-0.405241314004990,-0.416429560097637,-0.427555093430282,-
0.438616238538527,-0.449611329654607,-0.460538710958240,-0.471396736825998,-
0.482183772079123,-0.492898192229784,-0.503538383725718,-0.514102744193222,-
0.524589682678469,-0.534997619887097,-0.545324988422046,-0.555570233019602,-
0.565731810783613,-0.575808191417845,-0.585797857456439,-0.595699304492433,-
0.605511041404325,-0.615231590580627,-0.624859488142386,-0.634393284163645,-
0.643831542889791,-0.653172842953777,-0.662415777590172,-0.671558954847018,-
0.680600997795453,-0.689540544737067,-0.698376249408973,-0.707106781186548,-
0.715730825283819,-0.724247082951467,-0.732654271672413,-0.740951125354959,-
0.749136394523459,-0.757208846506484,-0.765167265622459,-0.773010453362737,-
0.780737228572094,-0.788346427626606,-0.795836904608884,-0.803207531480645,-
0.810457198252595,-0.817584813151584,-0.824589302785025,-0.831469612302545,-
0.838224705554838,-0.844853565249707,-0.851355193105265,-0.857728610000272,-
0.863972856121587,-0.870086991108711,-0.876070094195407,-0.881921264348355,-
0.887639620402854,-0.893224301195515,-0.898674465693954,-0.903989293123443,-
0.909167983090522,-0.914209755703531,-0.919113851690058,-0.923879532511287,-
0.928506080473216,-0.932992798834739,-0.937339011912575,-0.941544065183021,-
0.945607325380521,-0.949528180593037,-0.953306040354194,-0.956940335732209,-
0.960430519415566,-0.963776065795440,-0.966976471044852,-0.970031253194544,-
0.972939952205560,-0.975702130038529,-0.978317370719628,-0.980785280403230,-
0.983105487431216,-0.985277642388941,-0.987301418157858,-0.989176509964781,-
0.990902635427780,-0.992479534598710,-0.993906970002356,-0.995184726672197,-
```

```
0.996312612182778,-0.997290456678690,-0.998118112900149,-0.998795456205172,-
0.999322384588349,-0.999698818696204,-0.999924701839145,-1,-
0.999924701839145,-0.999698818696204,-0.999322384588350,-0.998795456205172,-
0.998118112900149,-0.997290456678690,-0.996312612182778,-0.995184726672197,-
0.993906970002356,-0.992479534598710,-0.990902635427780,-0.989176509964781,-
0.987301418157858,-0.985277642388941,-0.983105487431216,-0.980785280403230,-
0.978317370719628,-0.975702130038529,-0.972939952205560,-0.970031253194544,-
0.966976471044852,-0.963776065795440,-0.960430519415566,-0.956940335732209,-
0.953306040354194,-0.949528180593037,-0.945607325380521,-0.941544065183021,-
0.937339011912575,-0.932992798834739,-0.928506080473216,-0.923879532511287,-
0.919113851690058,-0.914209755703531,-0.909167983090523,-0.903989293123443,-
0.898674465693954,-0.893224301195515,-0.887639620402854,-0.881921264348355,-
0.876070094195407,-0.870086991108712,-0.863972856121587,-0.857728610000272,-
0.851355193105265,-0.844853565249707,-0.838224705554838,-0.831469612302546,-
0.824589302785025,-0.817584813151584,-0.810457198252595,-0.803207531480645,-
0.795836904608884,-0.788346427626606,-0.780737228572095,-0.773010453362737,-
0.765167265622459,-0.757208846506485,-0.749136394523460,-0.740951125354959,-
0.732654271672413,-0.724247082951467,-0.715730825283819,-0.707106781186548,-
0.698376249408973,-0.689540544737067,-0.680600997795453,-0.671558954847019,-
0.662415777590172,-0.653172842953777,-0.643831542889792,-0.634393284163646,-
0.624859488142387,-0.615231590580627,-0.605511041404326,-0.595699304492433,-
0.585797857456439,-0.575808191417845,-0.565731810783614,-0.555570233019602,-
0.545324988422047,-0.534997619887097,-0.524589682678469,-0.514102744193222,-
0.503538383725718,-0.492898192229784,-0.482183772079123,-0.471396736825998,-
0.460538710958240,-0.449611329654607,-0.438616238538528,-0.427555093430283,-
0.416429560097637,-0.405241314004990,-0.393992040061048,-0.382683432365090,-
0.371317193951838,-0.359895036534988,-0.348418680249435,-0.336889853392220,-
0.325310292162263,-0.313681740398892,-0.302005949319229,-0.290284677254463,-
0.278519689385054,-0.266712757474899,-0.254865659604514,-0.242980179903264,-
0.231058108280671,-0.219101240156870,-0.207111376192219,-0.195090322016129,-
0.183039887955141,-0.170961888760302,-0.158858143333862,-0.146730474455362,-
0.134580708507126,-0.122410675199216,-0.110222207293883,-0.0980171403295605,-
0.0857973123444403,-0.0735645635996674,-0.0613207363022091,-
0.0490676743274181,-0.0368072229413594,-0.0245412285229124,-
0.0122715382857206};
const DTYPE
W_real[]={1,0.999924701839145,0.999698818696204,0.999322384588350,0.9987954562
05172,0.998118112900149,0.997290456678690,0.996312612182778,0.995184726672197,
0.993906970002356,0.992479534598710,0.990902635427780,0.989176509964781,0.9873
01418157858,0.985277642388941,0.983105487431216,0.980785280403230,0.9783173707
19628,0.975702130038529,0.972939952205560,0.970031253194544,0.966976471044852,
0.963776065795440,0.960430519415566,0.956940335732209,0.953306040354194,0.9495
28180593037,0.945607325380521,0.941544065183021,0.937339011912575,0.9329927988
34739,0.928506080473216,0.923879532511287,0.919113851690058,0.914209755703531,
0.909167983090522,0.903989293123443,0.898674465693954,0.893224301195515,0.8876
39620402854,0.881921264348355,0.876070094195407,0.870086991108712,0.8639728561
21587,0.857728610000272,0.851355193105265,0.844853565249707,0.838224705554838,
0.831469612302545,0.824589302785025,0.817584813151584,0.810457198252595,0.8032
```

```
07531480645,0.795836904608884,0.788346427626606,0.780737228572095,0.7730104533
62737,0.765167265622459,0.757208846506485,0.749136394523459,0.740951125354959,
0.732654271672413,0.724247082951467,0.715730825283819,0.707106781186548,0.6983
76249408973,0.689540544737067,0.680600997795453,0.671558954847018,0.6624157775
90172,0.653172842953777,0.643831542889792,0.634393284163646,0.624859488142387,
0.615231590580627,0.605511041404326,0.595699304492434,0.585797857456439,0.5758
08191417845,0.565731810783613,0.555570233019602,0.545324988422047,0.5349976198
87097,0.524589682678469,0.514102744193222,0.503538383725718,0.492898192229784,
0.482183772079123,0.471396736825998,0.460538710958240,0.449611329654607,0.4386
16238538528,0.427555093430282,0.416429560097637,0.405241314004990,0.3939920400
61048,0.382683432365090,0.371317193951838,0.359895036534988,0.348418680249435,
0.336889853392220,0.325310292162263,0.313681740398892,0.302005949319228,0.2902
84677254462,0.278519689385053,0.266712757474898,0.254865659604515,0.2429801799
03264,0.231058108280671,0.219101240156870,0.207111376192219,0.195090322016128,
0.183039887955141,0.170961888760301,0.158858143333861,0.146730474455362,0.1345
80708507126,0.122410675199216,0.110222207293883,0.0980171403295608,0.085797312
3444399,0.0735645635996675,0.0613207363022087,0.0490676743274181,0.03680722294
13590,0.0245412285229123,0.0122715382857199,6.12323399573677e-17,-
0.0122715382857198,-0.0245412285229121,-0.0368072229413589,-
0.0490676743274180,-0.0613207363022085,-0.0735645635996673,-
0.0857973123444398,-0.0980171403295607,-0.110222207293883,-0.122410675199216,-
0.134580708507126,-0.146730474455362,-0.158858143333861,-0.170961888760301,-
0.183039887955141,-0.195090322016128,-0.207111376192218,-0.219101240156870,-
0.231058108280671,-0.242980179903264,-0.254865659604515,-0.266712757474898,-
0.278519689385053,-0.290284677254462,-0.302005949319228,-0.313681740398891,-
0.325310292162263,-0.336889853392220,-0.348418680249434,-0.359895036534988,-
0.371317193951838,-0.382683432365090,-0.393992040061048,-0.405241314004990,-
0.416429560097637,-0.427555093430282,-0.438616238538527,-0.449611329654607,-
0.460538710958240,-0.471396736825998,-0.482183772079123,-0.492898192229784,-
0.503538383725718,-0.514102744193222,-0.524589682678469,-0.534997619887097,-
0.545324988422046,-0.555570233019602,-0.565731810783613,-0.575808191417845,-
0.585797857456439,-0.595699304492433,-0.605511041404325,-0.615231590580627,-
0.624859488142386,-0.634393284163645,-0.643831542889791,-0.653172842953777,-
0.662415777590172,-0.671558954847018,-0.680600997795453,-0.689540544737067,-
0.698376249408973,-0.707106781186548,-0.715730825283819,-0.724247082951467,-
0.732654271672413,-0.740951125354959,-0.749136394523459,-0.757208846506485,-
0.765167265622459,-0.773010453362737,-0.780737228572095,-0.788346427626606,-
0.795836904608884,-0.803207531480645,-0.810457198252595,-0.817584813151584,-
0.824589302785025,-0.831469612302545,-0.838224705554838,-0.844853565249707,-
0.851355193105265,-0.857728610000272,-0.863972856121587,-0.870086991108711,-
0.876070094195407,-0.881921264348355,-0.887639620402854,-0.893224301195515,-
0.898674465693954,-0.903989293123443,-0.909167983090522,-0.914209755703531,-
0.919113851690058,-0.923879532511287,-0.928506080473216,-0.932992798834739,-
0.937339011912575,-0.941544065183021,-0.945607325380521,-0.949528180593037,-
0.953306040354194,-0.956940335732209,-0.960430519415566,-0.963776065795440,-
0.966976471044852,-0.970031253194544,-0.972939952205560,-0.975702130038529,-
0.978317370719628,-0.980785280403230,-0.983105487431216,-0.985277642388941,-
0.987301418157858,-0.989176509964781,-0.990902635427780,-0.992479534598710,-
```

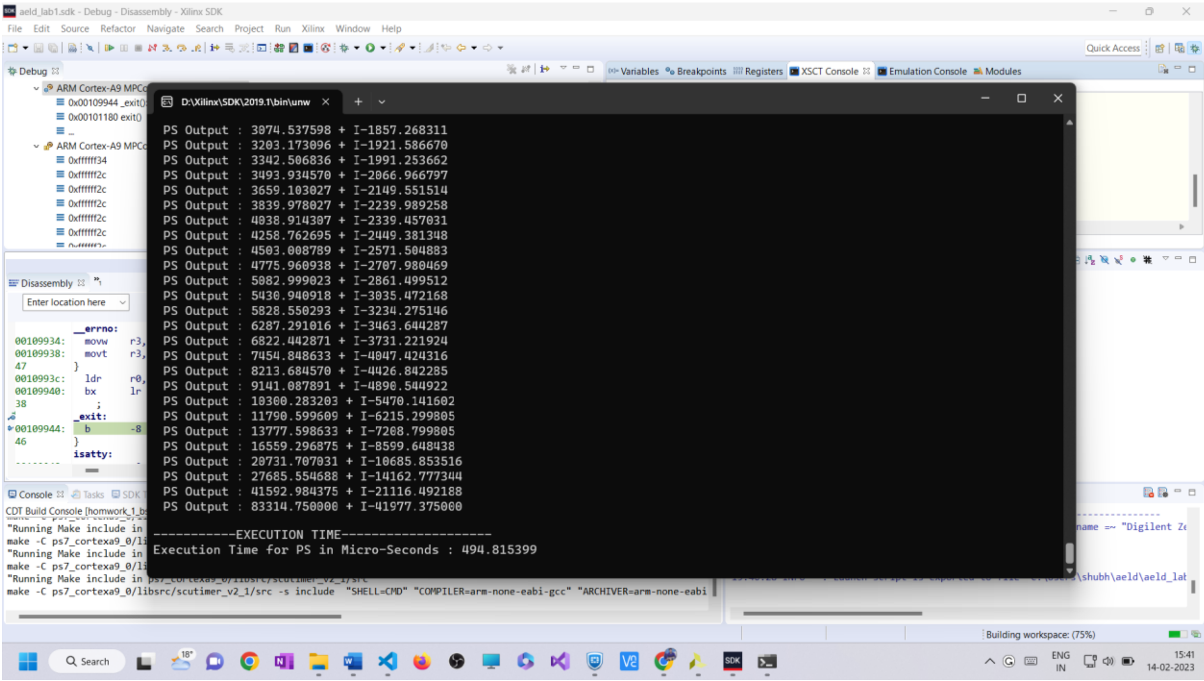
```
0.993906970002356,-0.995184726672197,-0.996312612182778,-0.997290456678690,-
0.998118112900149,-0.998795456205172,-0.999322384588350,-0.999698818696204,-
0.999924701839145,-1,-0.999924701839145,-0.999698818696204,-
0.999322384588350,-0.998795456205172,-0.998118112900149,-0.997290456678690,-
0.996312612182778,-0.995184726672197,-0.993906970002356,-0.992479534598710,-
0.990902635427780,-0.989176509964781,-0.987301418157858,-0.985277642388941,-
0.983105487431216,-0.980785280403230,-0.978317370719628,-0.975702130038529,-
0.972939952205560,-0.970031253194544,-0.966976471044852,-0.963776065795440,-
0.960430519415566,-0.956940335732209,-0.953306040354194,-0.949528180593037,-
0.945607325380521,-0.941544065183021,-0.937339011912575,-0.932992798834739,-
0.928506080473216,-0.923879532511287,-0.919113851690058,-0.914209755703531,-
0.909167983090523,-0.903989293123443,-0.898674465693954,-0.893224301195515,-
0.887639620402854,-0.881921264348355,-0.876070094195407,-0.870086991108712,-
0.863972856121587,-0.857728610000272,-0.851355193105265,-0.844853565249707,-
0.838224705554838,-0.831469612302546,-0.824589302785025,-0.817584813151584,-
0.810457198252595,-0.803207531480645,-0.795836904608884,-0.788346427626606,-
0.780737228572095,-0.773010453362737,-0.765167265622459,-0.757208846506485,-
0.749136394523459,-0.740951125354959,-0.732654271672413,-0.724247082951467,-
0.715730825283819,-0.707106781186548,-0.698376249408973,-0.689540544737067,-
0.680600997795453,-0.671558954847019,-0.662415777590172,-0.653172842953777,-
0.643831542889792,-0.634393284163646,-0.624859488142387,-0.615231590580627,-
0.605511041404326,-0.595699304492433,-0.585797857456439,-0.575808191417845,-
0.565731810783614,-0.555570233019602,-0.545324988422047,-0.534997619887097,-
0.524589682678469,-0.514102744193222,-0.503538383725718,-0.492898192229784,-
0.482183772079123,-0.471396736825998,-0.460538710958240,-0.449611329654607,-
0.438616238538528,-0.427555093430283,-0.416429560097637,-0.405241314004990,-
0.393992040061048,-0.382683432365090,-0.371317193951838,-0.359895036534988,-
0.348418680249435,-0.336889853392220,-0.325310292162263,-0.313681740398891,-
0.302005949319229,-0.290284677254462,-0.278519689385054,-0.266712757474899,-
0.254865659604514,-0.242980179903264,-0.231058108280671,-0.219101240156870,-
0.207111376192218,-0.195090322016129,-0.183039887955141,-0.170961888760302,-
0.158858143333862,-0.146730474455362,-0.134580708507126,-0.122410675199216,-
0.110222207293883,-0.0980171403295605,-0.0857973123444402,-
0.0735645635996674,-0.0613207363022090,-0.0490676743274180,-
0.0368072229413593,-0.0245412285229124,-0.0122715382857205,-1.83697019872103e-
16,0.0122715382857201,0.0245412285229120,0.0368072229413590,0.0490676743274177
,0.0613207363022086,0.0735645635996670,0.0857973123444399,0.0980171403295601,0
.110222207293883,0.122410675199216,0.134580708507126,0.146730474455362,0.15885
8143333861,0.170961888760301,0.183039887955141,0.195090322016128,0.20711137619
2218,0.219101240156870,0.231058108280671,0.242980179903264,0.254865659604514,0
.266712757474898,0.278519689385053,0.290284677254462,0.302005949319228,0.31368
1740398891,0.325310292162263,0.336889853392220,0.348418680249435,0.35989503653
4988,0.371317193951837,0.382683432365090,0.393992040061048,0.405241314004990,0
.416429560097637,0.427555093430282,0.438616238538527,0.449611329654607,0.46053
8710958240,0.471396736825998,0.482183772079122,0.492898192229784,0.50353838372
5718,0.514102744193222,0.524589682678469,0.534997619887097,0.545324988422047,0
.555570233019602,0.565731810783613,0.575808191417845,0.585797857456439,0.59569
9304492433,0.605511041404325,0.615231590580627,0.624859488142386,0.63439328416
```

```
3646,0.643831542889791,0.653172842953777,0.662415777590172,0.671558954847018,0
.680600997795453,0.689540544737067,0.698376249408972,0.707106781186547,0.71573
0825283819,0.724247082951467,0.732654271672413,0.740951125354959,0.74913639452
3459,0.757208846506484,0.765167265622459,0.773010453362737,0.780737228572094,0
.788346427626606,0.795836904608883,0.803207531480645,0.810457198252595,0.81758
4813151584,0.824589302785025,0.831469612302545,0.838224705554838,0.84485356524
9707,0.851355193105265,0.857728610000272,0.863972856121586,0.870086991108711,0
.876070094195407,0.881921264348355,0.887639620402854,0.893224301195515,0.89867
4465693954,0.903989293123443,0.909167983090522,0.914209755703531,0.91911385169
0058,0.923879532511287,0.928506080473216,0.932992798834739,0.937339011912575,0
.941544065183021,0.945607325380521,0.949528180593037,0.953306040354194,0.95694
0335732209,0.960430519415566,0.963776065795440,0.966976471044852,0.97003125319
4544,0.972939952205560,0.975702130038529,0.978317370719628,0.980785280403230,0
.983105487431216,0.985277642388941,0.987301418157858,0.989176509964781,0.99090
2635427780,0.992479534598710,0.993906970002356,0.995184726672197,0.99631261218
2778,0.997290456678690,0.998118112900149,0.998795456205172,0.999322384588349,0
.999698818696204,0.999924701839145};

#endif /* SRC_FFTSW_H_ */

#include <stdio.h>
#include <stdbool.h>
#include "platform.h"
#include <stdlib.h>
#include <complex.h>
#include <xtime_l.h>
#include "fftsw.h"

// This function is designed to perform FFT operation on PS
// Default word length is single precision floating point
// You may explore double precision floating point and effect on execution
time
//void FFT_sw(float FFTIn_I[FFTSIZE], float FFTIn_R[FFTSIZE], float
FFTOut_I[FFTSIZE], float FFTOut_R[FFTSIZE]);
void FFT_sw(float FFTIn_I[FFTSIZE], float FFTIn_R[FFTSIZE], float
FFTOut_I[FFTSIZE], float FFTOut_R[FFTSIZE]);



int main()
{
  init_platform();
    // Instead of constant input, we will generate the input for FFT
    float DataIN_R[FFTSIZE]; // Real part of the input
    float DataIN_I[FFTSIZE]; // Imaginary part of the input
    // Complex representation of the input
    float complex FFTIn_C[FFTSIZE];
    // input to the FFT in PS and PL
    for (int k = 0; k<FFTSIZE;k++)
```

```
    {
      FFTIn_C[k] = (k/4) + (k/2)*I;
      DataIN_R[k] = creal(FFTIn_C[k]);
      DataIN_I[k] = cimag(FFTIn_C[k]);
    }
    //FFT output
    float FFTOut_R[FFTSIZE];
    float FFTOut_I[FFTSIZE];

    // to store the time at which certain processes starts and ends
    XTime time_PS_start , time_PS_end;
    XTime_SetTime(0); // Reset the global timer
    //-------------------- PS PART --------------------------------//
    XTime_GetTime(&time_PS_start); // Capture the timer value at the start
    FFT_sw(DataIN_I, DataIN_R, FFTOut_I,FFTOut_R);
    XTime_GetTime(&time_PS_end); // Capture the timer value at the start


    // Print the FFT output on the UART
    printf("\n FFT output: \r\n");
    for (int j = 0 ; j < FFTSIZE ; j++)
    {
        printf("PS Output : %f + I%f \n " , FFTOut_R[j], FFTOut_I[j]);
        usleep(0.1); // Always add some buffer time between display
    }


    printf("\n-----------EXECUTION TIME--------------------\n");
    float time_processor = 0;
    time_processor = (float)1.0 * (time_PS_end - time_PS_start) /
(COUNTS_PER_SECOND/1000000);
    printf("Execution Time for PS in Micro-Seconds : %f\n" , time_processor);
    cleanup_platform();
    return 0;

}

#define reverse(n) ((n & 0x1) << 8) | ((n & 0x2) << 6) | ((n & 0x4) << 4) |
((n & 0x8) << 2) | ((n & 0x10) << 0) | ((n & 0x20) >> 0) | ((n & 0x40) >> 2) |
((n & 0x80) >> 4) | ((n & 0x100) >> 6) | ((n & 0x200) >> 8)

void FFT_sw(float FFTIn_I[FFTSIZE], float FFTIn_R[FFTSIZE], float
FFTOut_I[FFTSIZE], float FFTOut_R[FFTSIZE])
{

  DTYPE temp_R;    /*temporary storage complex variable*/
  DTYPE temp_I;    /*temporary storage complex variable*/

  int i,j;      /* loop indexes */
```

```
  int i_lower;     /* Index of lower point in butterfly */

  int stage;
  int subFFTSize; //Size of FFT in each stage of FFT
  int BFWidth;      /*Butterfly Width*/

  /*=====================BEGIN BIT REBERSAL===========================*/
  for (i = 0; i < FFTSIZE; ++i) {
    FFTOut_R[reverse(i)] = FFTIn_R[i];
    FFTOut_I[reverse(i)] = FFTIn_I[i];
  }
  /*++++++++++++++++++++++END OF BIT REVERSAL++++++++++++++++++++++++++*/

  /*=======================BEGIN: FFT=========================*/
  // Do FFTSTAGES of butterflies
  DTYPE BFWeight_R, BFWeight_I;
 // For N-point FFT, there are log2(N) stages
  stages:for(stage=1; stage<= FFTSTAGES; stage++)
  {
    subFFTSize = 1 << stage;    // DFT = 2^stage = points in sub DFT
    BFWidth = subFFTSize >> 1;       // Butterfly WIDTHS in sub-FFT (FFTSIZE
of sub-FFT/2) no of weights

    // Perform butterflies for j-th stage
    // This loop runs for the iteration equal to BF width
    // In 4-point FFT, BF width is 1 in stage 1 and 2 in stage 2
    // In 8-point FFT, BF width is 1 in stage 1, 2 in stage 2 and 4 in stage 3
    butterfly:for(j=0; j<BFWidth; j++)
    {
      //Note that weights of all butterfly units are same in a given stage
      // We can reduce the number of memory read by using this for loop
      BFWeight_R = W_real[j * (FFTSIZE>>stage)];
      BFWeight_I = W_imag[j * (FFTSIZE>>stage)];

    // This loop is for all butterflies in a stage that use same W**k
    // In 4-point FFT, we have two BFs in stage 1
    // In 8-point FFT, we have four BFs in stage 1 and two BFs in stage 2
      // Each butter fly weight affects two outputs and hence we have two
outputs for each iteration
    subDFTSize:for(i =j ; i < FFTSIZE; i += subFFTSize) // This loop runs for
FFTSIZE/SubFFTSize iterations
      {
      i_lower = i + BFWidth;      //index of lower point in butterfly
      temp_R = FFTOut_R[i_lower] * BFWeight_R - FFTOut_I[i_lower] *
BFWeight_I;
      temp_I = FFTOut_I[i_lower] * BFWeight_R + FFTOut_R[i_lower] *
BFWeight_I;
```

```
        FFTOut_R[i_lower] = FFTOut_R[i] - temp_R;
        FFTOut_I[i_lower] = FFTOut_I[i] - temp_I;
        FFTOut_R[i] = FFTOut_R[i] + temp_R;
        FFTOut_I[i] = FFTOut_I[i] + temp_I;
        }
    }
    }
}
```

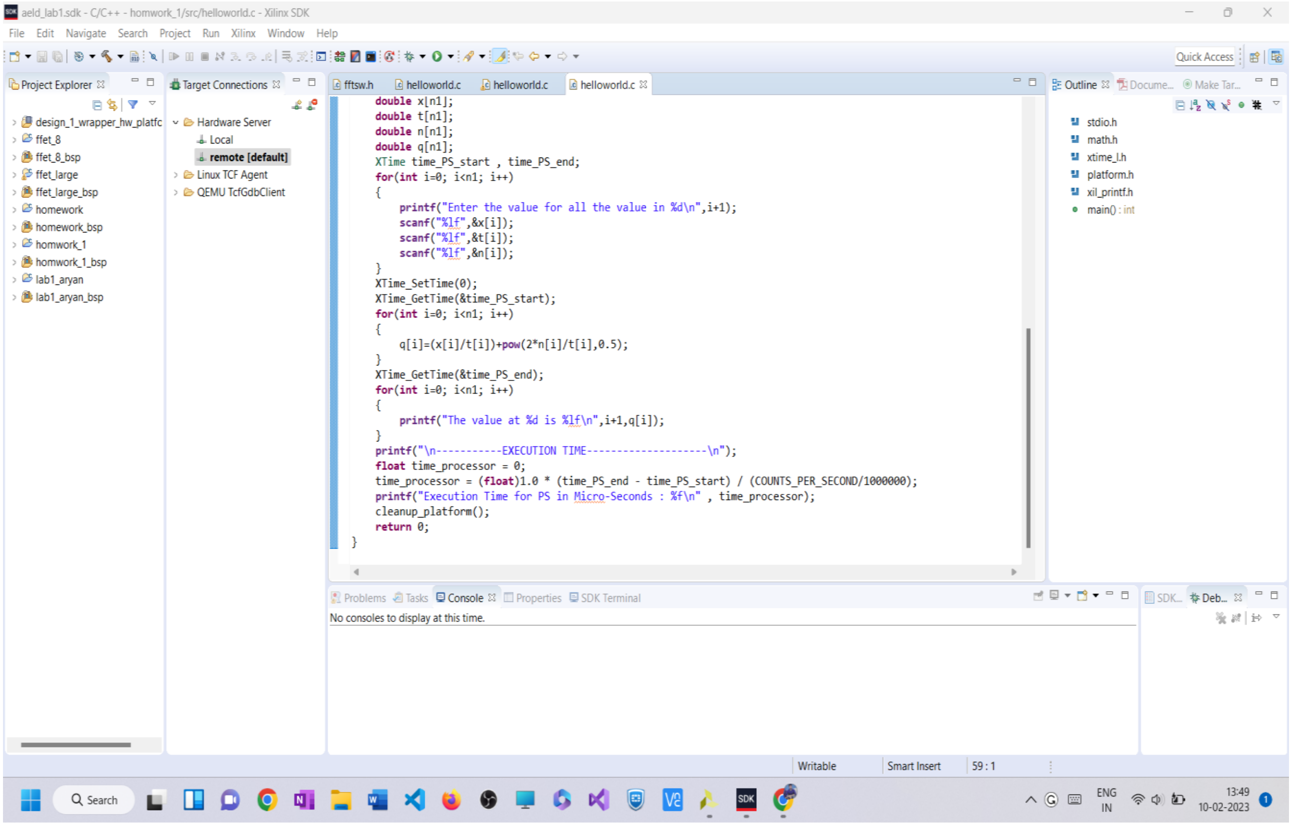


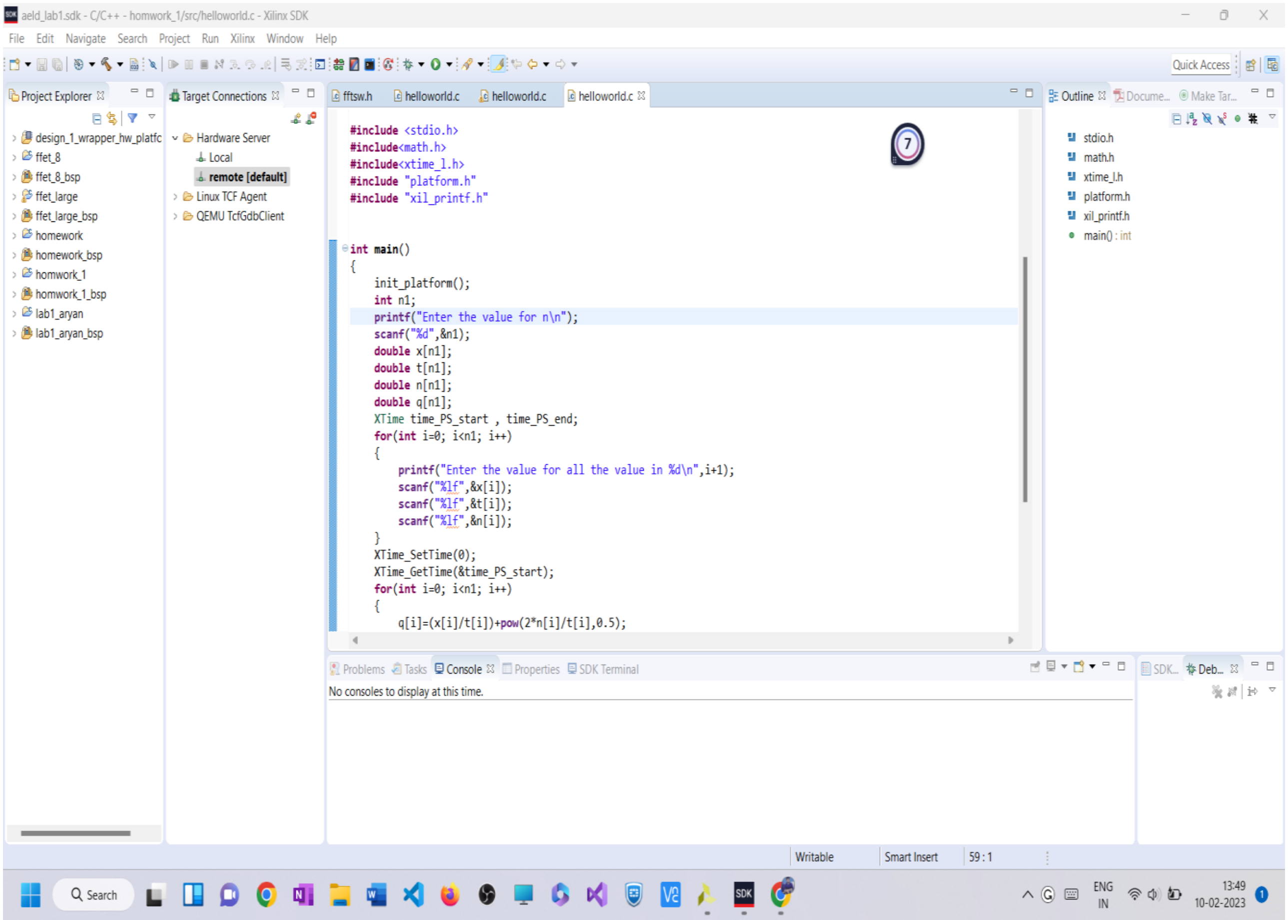
Time for PS -> 494.815399 MicroSeconds

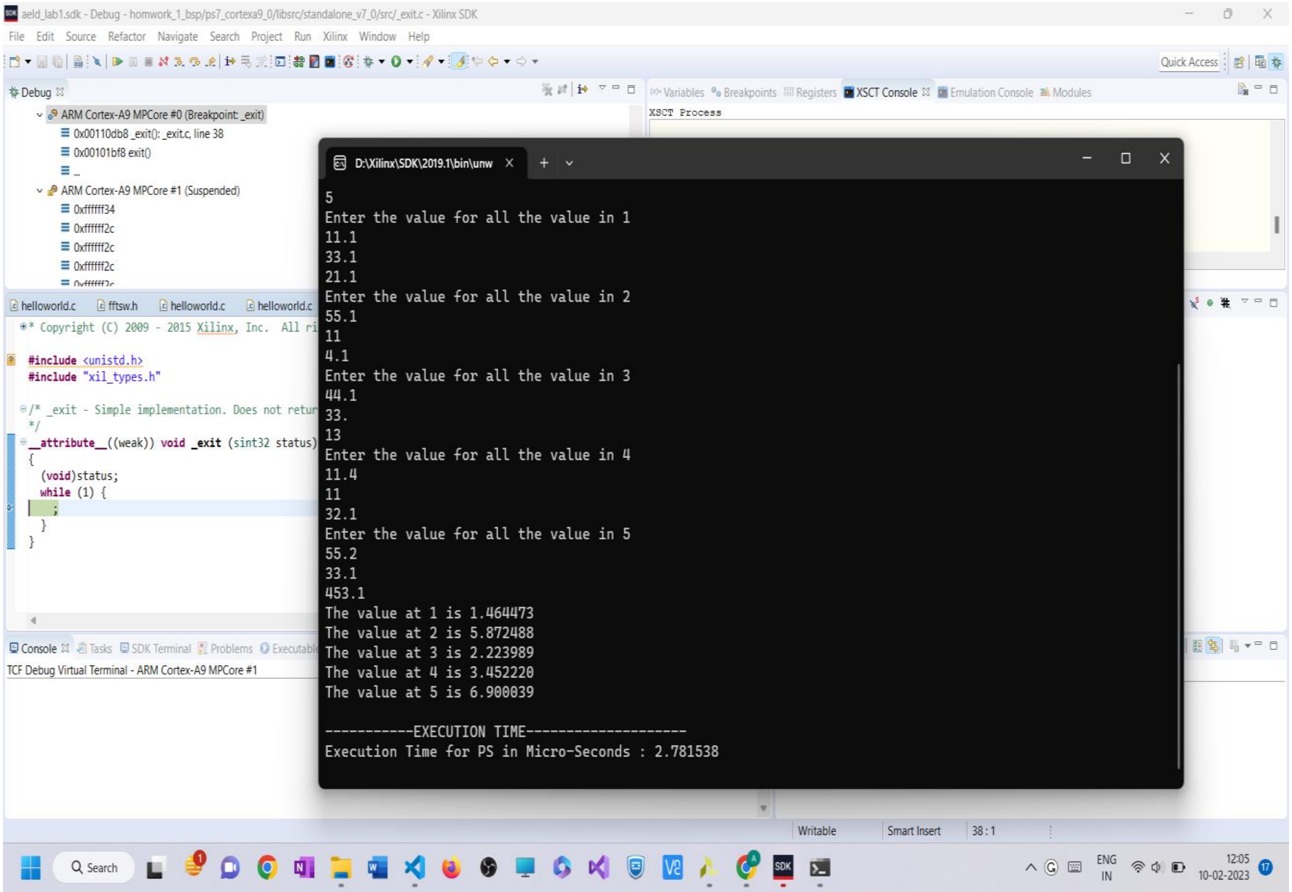
HOMEWORK PROBLEM

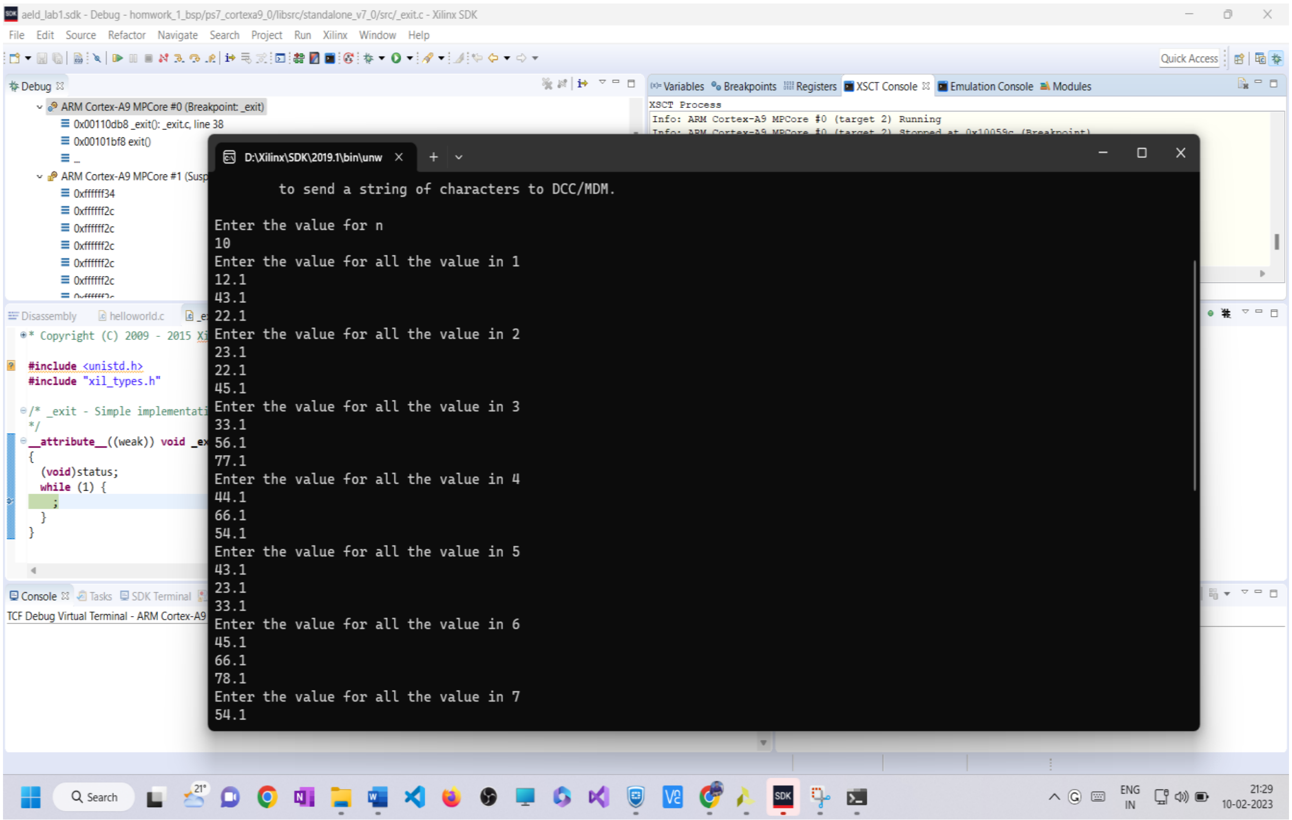
```
#include <stdio.h>
#include<math.h>
#include<xtime_l.h>
#include "platform.h"
#include "xil_printf.h"


int main()
{
    init_platform();
    int n1;
```

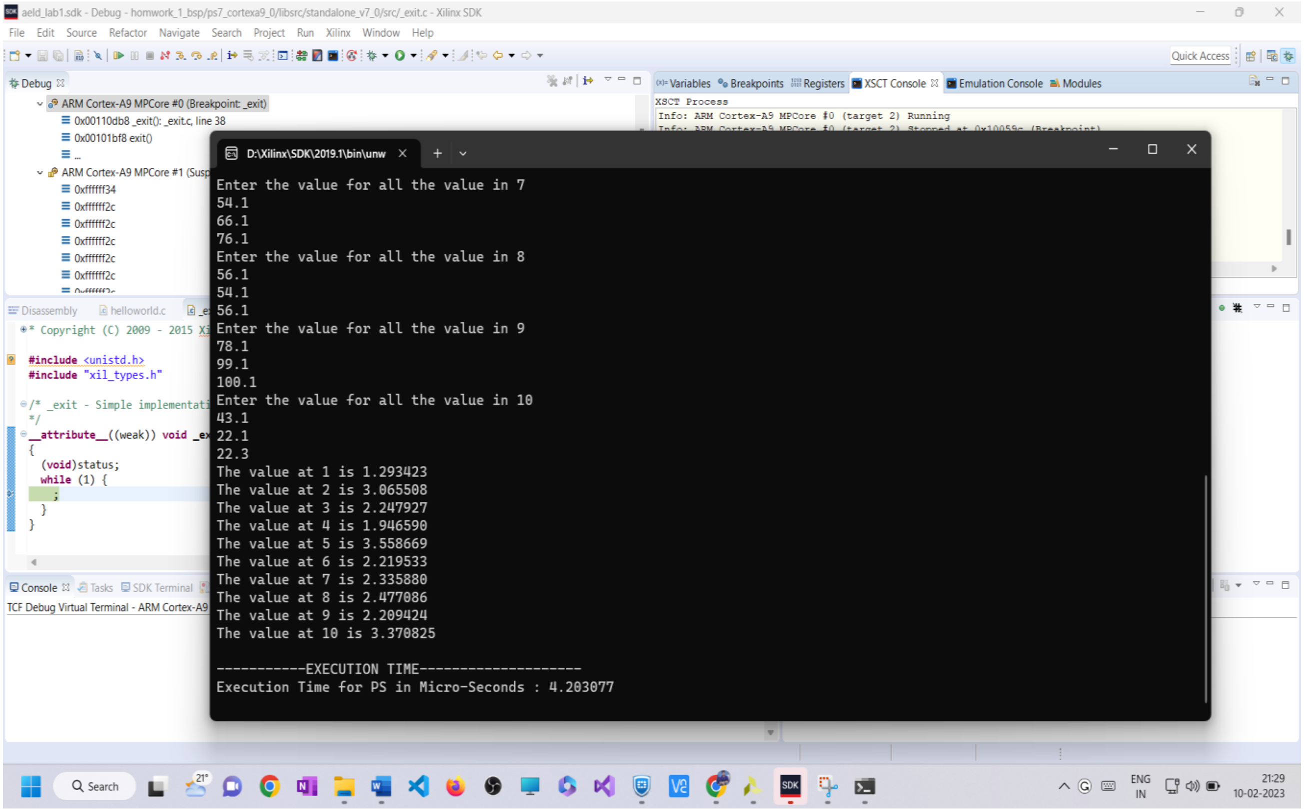
```c
    printf("Enter the value for n\n");
    scanf("%d",&n1);
    double x[n1];
    double t[n1];
    double n[n1];
    double q[n1];
    XTime time_PS_start , time_PS_end;
    for(int i=0; i<n1; i++)
    {
        printf("Enter the value for all the value in %d\n",i+1);
        scanf("%lf",&x[i]);
        scanf("%lf",&t[i]);
        scanf("%lf",&n[i]);
    }
    XTime_SetTime(0);
    XTime_GetTime(&time_PS_start);
    for(int i=0; i<n1; i++)
    {
        q[i]=(x[i]/t[i])+pow(2*n[i]/t[i],0.5);
    }
    XTime_GetTime(&time_PS_end);
    for(int i=0; i<n1; i++)
    {
        printf("The value at %d is %lf\n",i+1,q[i]);
    }
    printf("\n-----------EXECUTION TIME--------------------\n");
    float time_processor = 0;
    time_processor = (float)1.0 * (time_PS_end - time_PS_start) /
(COUNTS_PER_SECOND/1000000);
    printf("Execution Time for PS in Micro-Seconds : %f\n" , time_processor);
    cleanup_platform();
    return 0;
}
```


aeld_lab1.sdk - C/C++ - homwork_1/src/helloworld.c - Xilinx SDK
double x[n1];
double t[n1];
double n[n1];
double q[n1];
XTime time_PS_start , time_PS_end;
for(int i=0; i<n1; i++)
{
printf("Enter the value for all the value in %d\n",i+1);
scanf("%lf",&x[i]);
scanf("%lf",&t[i]);
scanf("%lf",&n[i]);
}
XTime_SetTime(0);
XTime_GetTime(&time_PS_start);
for(int i=0; i<n1; i++)
{
q[i]=(x[i]/t[i])+pow(2*n[i]/t[i],0.5);
}
XTime_GetTime(&time_PS_end);
for(int i=0; i<n1; i++)
{
printf("The value at %d is %lf\n",i+1,q[i]);
}
printf("\n-----------EXECUTION TIME-------------------\n");
float time_processor = 0;
time_processor = (float)1.0 * (time_PS_end - time_PS_start) / (COUNTS_PER_SECOND/1000000);
printf("Execution Time for PS in Micro-Seconds : %f\n" , time_processor);
cleanup_platform();
return 0;
}
No consoles to display at this time.

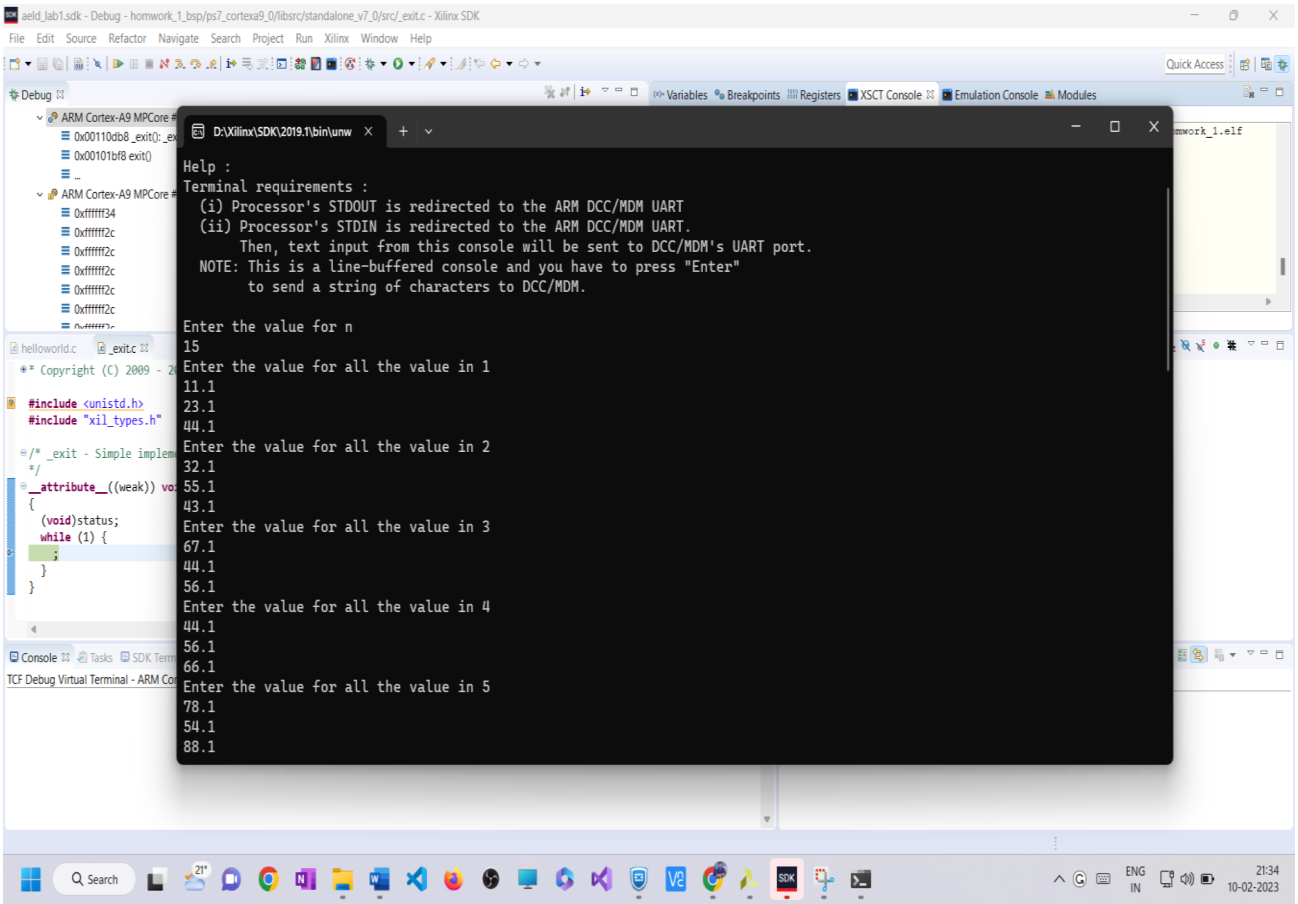

aeld_lab1.sdk - C/C++ - homwork_1/src/helloworld.c - Xilinx SDK
#include <stdio.h>
#include<math.h>
#include<xtime_l.h>
#include "platform.h"
#include "xil_printf.h"
int main()
{
init_platform();
int n1;
printf("Enter the value for n\n");
scanf("%d",&n1);
double x[n1];
double t[n1];
double n[n1];
double q[n1];
XTime time_PS_start , time_PS_end;
for(int i=0; i<n1; i++)
{
printf("Enter the value for all the value in %d\n",i+1);
scanf("%lf",&x[i]);
scanf("%lf",&t[i]);
scanf("%lf",&n[i]);
}
XTime_SetTime(0);
XTime_GetTime(&time_PS_start);
for(int i=0; i<n1; i++)
{
q[i]=(x[i]/t[i])+pow(2*n[i]/t[i],0.5);
No consoles to display at this time.


aeld_lab1.sdk - Debug - homwork_1_bsp/ps7_cortexa9_0/libsrc/standalone_v7_0/src/_exit.c - Xilinx SDK
D:\Xilinx\SDK\2019.1\bin\unw
5
Enter the value for all the value in 1
11.1
33.1
21.1
Enter the value for all the value in 2
55.1
11
4.1
Enter the value for all the value in 3
44.1
33.
13
Enter the value for all the value in 4
11.4
11
32.1
Enter the value for all the value in 5
55.2
33.1
453.1
The value at 1 is 1.464473
The value at 2 is 5.872488
The value at 3 is 2.223989
The value at 4 is 3.452220
The value at 5 is 6.900039
-----------EXECUTION TIME-------------------
Execution Time for PS in Micro-Seconds : 2.781538

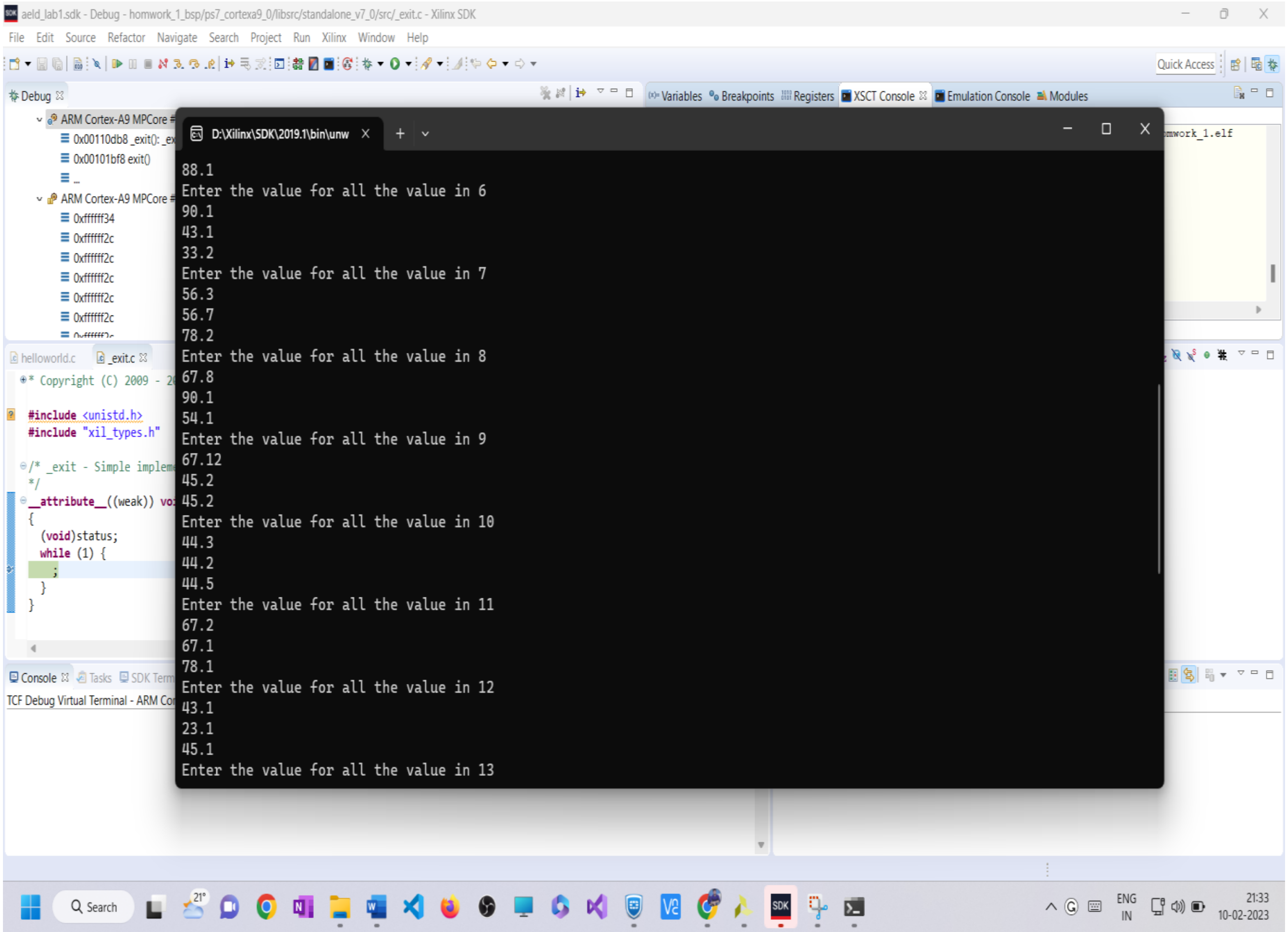

aeld_lab1.sdk - Debug - homwork_1_bsp/ps7_cortexa9_0/libsrc/standalone_v7_0/src/_exit.c - Xilinx SDK
D:\Xilinx\SDK\2019.1\bin\unw
to send a string of characters to DCC/MDM.
Enter the value for n
10
Enter the value for all the value in 1
12.1
43.1
22.1
Enter the value for all the value in 2
23.1
22.1
45.1
Enter the value for all the value in 3
33.1
56.1
77.1
Enter the value for all the value in 4
44.1
66.1
54.1
Enter the value for all the value in 5
43.1
23.1
33.1
Enter the value for all the value in 6
45.1
66.1
78.1
Enter the value for all the value in 7
54.1

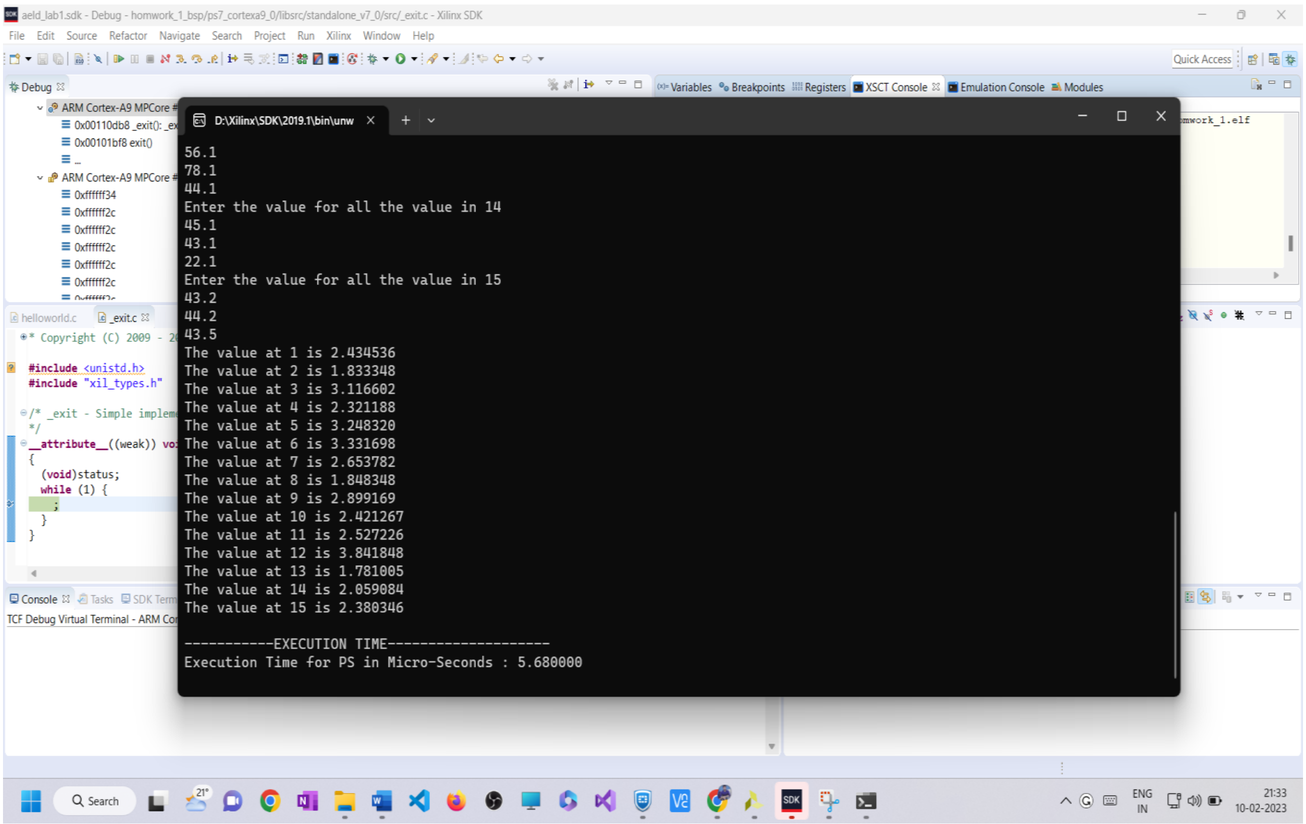


D:\Xilinx\SDK\2019.1\bin\unw
Enter the value for all the value in 7
54.1
66.1
76.1
Enter the value for all the value in 8
56.1
54.1
56.1
Enter the value for all the value in 9
78.1
99.1
100.1
Enter the value for all the value in 10
43.1
22.1
22.3
The value at 1 is 1.293423
The value at 2 is 3.065508
The value at 3 is 2.247927
The value at 4 is 1.946590
The value at 5 is 3.558669
The value at 6 is 2.219533
The value at 7 is 2.335880
The value at 8 is 2.477086
The value at 9 is 2.209424
The value at 10 is 3.370825
-----------EXECUTION TIME--------------------
Execution Time for PS in Micro-Seconds : 4.203077



D:\Xilinx\SDK\2019.1\bin\unw
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.
Enter the value for n
15
Enter the value for all the value in 1
11.1
23.1
44.1
Enter the value for all the value in 2
32.1
55.1
43.1
Enter the value for all the value in 3
67.1
44.1
56.1
Enter the value for all the value in 4
44.1
56.1
66.1
Enter the value for all the value in 5
78.1
54.1
88.1


aeld_lab1.sdk - Debug - homwork_1_bsp/ps7_cortexa9_0/libsrc/standalone_v7_0/src/_exit.c - Xilinx SDK
D:\Xilinx\SDK\2019.1\bin\unw
88.1
Enter the value for all the value in 6
90.1
43.1
33.2
Enter the value for all the value in 7
56.3
56.7
78.2
Enter the value for all the value in 8
67.8
90.1
54.1
Enter the value for all the value in 9
67.12
45.2
45.2
Enter the value for all the value in 10
44.3
44.2
44.5
Enter the value for all the value in 11
67.2
67.1
78.1
Enter the value for all the value in 12
43.1
23.1
45.1
Enter the value for all the value in 13
10-02-2023


aeld_lab1.sdk - Debug - homwork_1_bsp/ps7_cortexa9_0/libsrc/standalone_v7_0/src/_exit.c - Xilinx SDK
D:\Xilinx\SDK\2019.1\bin\unw
56.1
78.1
44.1
Enter the value for all the value in 14
45.1
43.1
22.1
Enter the value for all the value in 15
43.2
44.2
43.5
The value at 1 is 2.434536
The value at 2 is 1.833348
The value at 3 is 3.116602
The value at 4 is 2.321188
The value at 5 is 3.248320
The value at 6 is 3.331698
The value at 7 is 2.653782
The value at 8 is 1.848348
The value at 9 is 2.899169
The value at 10 is 2.421267
The value at 11 is 2.527226
The value at 12 is 3.841848
The value at 13 is 1.781005
The value at 14 is 2.059084
The value at 15 is 2.380346
-----------EXECUTION TIME--------------------
Execution Time for PS in Micro-Seconds : 5.680000
10-02-2023

Time of Execution: -

For N=5 -> 2.781583 Micro-Seconds

For N=10 -> 4.203077 Micro-Seconds

For N=15 -> 5.680000 Micro-Seconds