

Project 4, Project: Fast Fourier Transform (FFT) and OFDM Receiver

Ricardo Lizárraga

rilizarraga@ucsd.edu PID: **A69028483****Submit to:**

Ryan Charles Kastner

(TA) Zhenghua Ma

(TA) Abarajithan Ganeswaran

fft1024_Baseline

TEST BENCH : PASS

```

1 INFO: [SIM 2] ***** CSIM start *****
2 INFO: [SIM 4] CSIM will launch GCC as the compiler.
3   Compiling ../../../../fft_test.cpp in debug mode
4   Compiling ../../../../fft.cpp in debug mode
5   Generating csim.exe
6 INPUTS
7 Printing FFT Output
8   0      0.000000  0.000000
9   1     512.000000  0.000000
10  2     256.000000  0.000000
11  3     768.000000  0.000000
12  4     128.000000  0.000000
13  5     640.000000  0.000000
14  6     384.000000  0.000000
15  7     896.000000  0.000000
16  8      64.000000  0.000000
17  9     576.000000  0.000000
18 10     320.000000  0.000000

```

```

1024 1016      127.000000  0.000000
1025 1017      639.000000  0.000000
1026 1018      383.000000  0.000000
1027 1019      895.000000  0.000000
1028 1020      255.000000  0.000000
1029 1021      767.000000  0.000000
1030 1022      511.000000  0.000000
1031 1023      1023.000000  0.000000
1032 Comparing against output data
1033 -----
1034   RMSE(R)           RMSE(I)
1035 0.047008574008942  0.013548693619668
1036 -----
1037 *****
1038 PASS: The output matches the golden output!
1039 *****
1040 INFO: [SIM 1] CSim done with 0 errors.
1041 INFO: [SIM 3] ***** CSIM finish *****

```

Target	Estimated	Uncertainty
10.00 ns	7.256 ns	2.70 ns

```

bit_reverse(in_R, in_I, Stage0_R, Stage0_I);
fft_stage_first(Stage0_R, Stage0_I, Stage1_R, Stage1_I);
fft_stages<2>(Stage1_R, Stage1_I, 2, Stage2_R, Stage2_I);
fft_stages<3>(Stage2_R, Stage2_I, 3, Stage3_R, Stage3_I);
fft_stages<4>(Stage3_R, Stage3_I, 4, Stage4_R, Stage4_I);
fft_stages<5>(Stage4_R, Stage4_I, 5, Stage5_R, Stage5_I);
fft_stages<6>(Stage5_R, Stage5_I, 6, Stage6_R, Stage6_I);
fft_stages<7>(Stage6_R, Stage6_I, 7, Stage7_R, Stage7_I);
fft_stages<8>(Stage7_R, Stage7_I, 8, Stage8_R, Stage8_I);
fft_stages<9>(Stage8_R, Stage8_I, 9, Stage9_R, Stage9_I);
fft_stage_last(Stage9_R, Stage9_I, buf_o_R, buf_o_I);

```

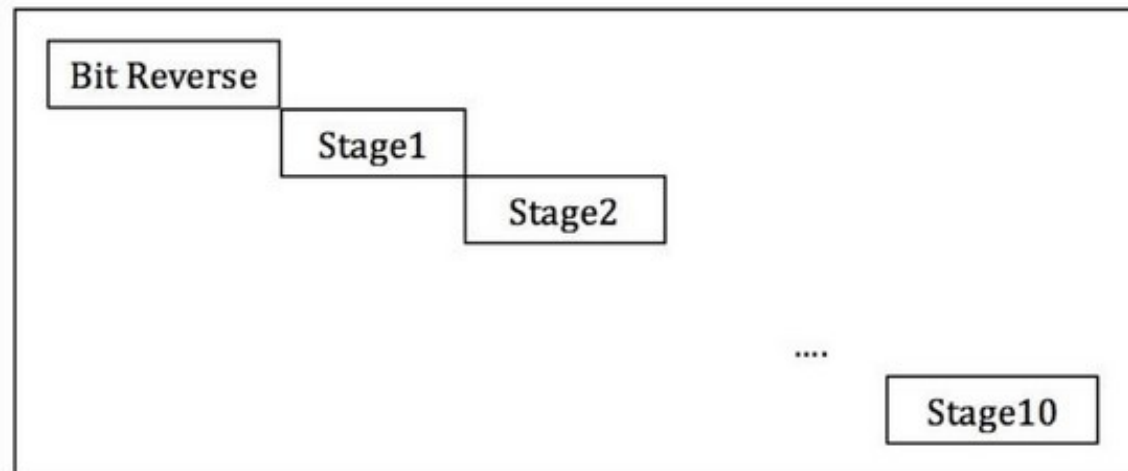


Figure 1: A staged implementation of a 1024 FFT. Bit reversal is followed by 10 stages of butterfly computations. This architecture is capable of pipeline both within the stages and across the stages.

An example of the bit reversed data for an 8 point FFT is as follows:

Input Decimal Address	Input Binary Address	Reversed Binary Address	Reversed Decimal Address
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
fft				-	273932	2.739E6	-	136450	-	dataflow	98	234	25653	38194	0
▶ bit_reverse				-	1026	1.026E4	-	1026	-	no	0	0	25	66	0
▶ fft_stage_first				-	529	5.290E3	-	529	-	no	0	18	2303	3206	0
▶ fft_stages_1				-	1067	1.067E4	-	1067	-	no	2	24	2592	3852	0
▶ fft_stages_2				-	2133	2.133E4	-	2133	-	no	2	24	2594	3854	0
▶ fft_stages_3				-	4265	4.265E4	-	4265	-	no	2	24	2596	3856	0
▶ fft_stages_4				-	8529	8.529E4	-	8529	-	no	2	24	2598	3857	0
▶ fft_stages_5				-	17057	1.710E5	-	17057	-	no	2	24	2600	3858	0
▶ fft_stages_6				-	34113	3.410E5	-	34113	-	no	2	24	2602	3859	0
▶ fft_stages_7				-	68225	6.820E5	-	68225	-	no	2	24	2604	3860	0
▶ fft_stages				-	136449	1.364E6	-	136449	-	no	2	24	2606	3859	0
▶ fft_stage_last				-	529	5.290E3	-	529	-	no	2	24	2513	3765	0

fft1024_Best Optimizations

- Data type

```
// stage 1
void fft_stage_first(DTYPE X_R[SIZE], DTYPE X_I[SIZE], DTYPE OUT_R[SIZE], DTYPE OUT_I[SIZE])
{
    // Insert your code here
    unsigned short i_lower;
    ap_int<12> c_twiddle, s_twiddle; //DTYPE
    ap_int<12> in_R, in_I, in_R_lower, in_I_lower, temp_R, temp_I; //DTYPE
```

- Using the W_real and W_imag tables:

```
const DTYPE W_real[]={1.000000, 0.999981,0.999925,0.999831,0.999699,0.999529,0.999322,0.999078,0.998795,0.998476,0.998125,0.997750,0.997350,0.996925,0.996475,0.996000,0.995500,0.994975,0.994425,0.993850,0.993250,0.992625,0.991975,0.991300,0.990600,0.989875,0.989125,0.988350,0.987550,0.986725,0.985875,0.984999,0.984099,0.983175,0.982225,0.981250,0.980250,0.979225,0.978175,0.977100,0.976000,0.974875,0.973725,0.972550,0.971350,0.970125,0.968875,0.967600,0.966300,0.964975,0.963625,0.962250,0.960850,0.959425,0.957975,0.956500,0.955000,0.953475,0.951925,0.950350,0.948750,0.947125,0.945475,0.943800,0.942100,0.940375,0.938625,0.936850,0.935050,0.933225,0.931375,0.929500,0.927600,0.925675,0.923725,0.921750,0.919750,0.917725,0.915675,0.913600,0.911500,0.909375,0.907225,0.905050,0.902850,0.900625,0.898375,0.896100,0.893800,0.891475,0.889125,0.886750,0.884350,0.881925,0.879475,0.877000,0.874500,0.872000,0.869475,0.866925,0.864350,0.861750,0.859125,0.856475,0.853800,0.851100,0.848375,0.845625,0.842850,0.840050,0.837225,0.834375,0.831500,0.828600,0.825675,0.822725,0.819750,0.816750,0.813725,0.810675,0.807600,0.804500,0.801375,0.798225,0.795050,0.791850,0.788625,0.785375,0.782100,0.778800,0.775475,0.772125,0.768750,0.765350,0.761925,0.758475,0.755000,0.751500,0.748000,0.744475,0.740925,0.737350,0.733750,0.730125,0.726475,0.722800,0.719100,0.715375,0.711625,0.707850,0.704050,0.700225,0.696375,0.692500,0.688600,0.684675,0.680725,0.676750,0.672750,0.668725,0.664675,0.660600,0.656500,0.652375,0.648225,0.644050,0.639850,0.635625,0.631375,0.627100,0.622800,0.618475,0.614125,0.609750,0.605350,0.600925,0.596475,0.592000,0.587500,0.583000,0.578475,0.573925,0.569350,0.564750,0.560125,0.555500,0.550875,0.546225,0.541550,0.536850,0.532125,0.527375,0.522600,0.517800,0.513000,0.508175,0.503325,0.498450,0.493550,0.488625,0.483675,0.478700,0.473700,0.468675,0.463625,0.458550,0.453450,0.448325,0.443175,0.438000,0.432800,0.427575,0.422325,0.417050,0.411750,0.406425,0.401075,0.395700,0.390300,0.384875,0.379425,0.373950,0.368450,0.362925,0.357375,0.351800,0.346200,0.340575,0.334925,0.329250,0.323550,0.317825,0.312075,0.306300,0.300500,0.294675,0.288825,0.282950,0.277050,0.271125,0.265175,0.259200,0.253200,0.247175,0.241125,0.235050,0.228950,0.222825,0.216675,0.210500,0.204300,0.198075,0.191825,0.185550,0.179250,0.172925,0.166575,0.160200,0.153800,0.147375,0.140925,0.134450,0.127950,0.121425,0.114875,0.108300,0.101700,0.095075,0.088425,0.081750,0.075050,0.068325,0.061575,0.054800,0.048000,0.041175,0.034325,0.027450,0.020550,0.013625,0.006675,0.0.000000,0.006675,0.013625,0.020550,0.027450,0.034325,0.041175,0.048000,0.054800,0.061575,0.068325,0.075050,0.081750,0.088425,0.095075,0.101700,0.108300,0.114875,0.121425,0.127950,0.134450,0.140925,0.147375,0.153800,0.160200,0.166575,0.172925,0.179250,0.185550,0.191825,0.198075,0.204300,0.210500,0.216675,0.222825,0.228950,0.235050,0.241175,0.247175,0.253200,0.259200,0.265175,0.271125,0.277050,0.282950,0.288825,0.294675,0.300500,0.306300,0.312075,0.317825,0.323550,0.329250,0.334925,0.340575,0.346200,0.351800,0.357375,0.362925,0.368450,0.373950,0.379425,0.384875,0.390300,0.395700,0.401075,0.406425,0.411750,0.417050,0.422325,0.427575,0.432800,0.438000,0.443175,0.448325,0.453450,0.458550,0.463625,0.468675,0.473700,0.478700,0.483725,0.488675,0.493550,0.498450,0.503325,0.508175,0.513000,0.517800,0.522600,0.527375,0.532125,0.536850,0.541550,0.546225,0.550875,0.555500,0.560125,0.564750,0.569350,0.573925,0.578475,0.583000,0.587500,0.592000,0.596475,0.600925,0.605375,0.609750,0.614125,0.618475,0.622800,0.627100,0.631375,0.635625,0.639850,0.644050,0.648225,0.652375,0.656500,0.660600,0.664675,0.668725,0.672750,0.676750,0.680725,0.684675,0.688625,0.692500,0.696375,0.700225,0.704050,0.707850,0.711625,0.715375,0.719100,0.722800,0.726475,0.730125,0.733750,0.737375,0.740925,0.744475,0.748000,0.751500,0.755000,0.758475,0.761925,0.765375,0.768750,0.772125,0.775475,0.778800,0.782100,0.785375,0.788625,0.791875,0.795075,0.798225,0.801375,0.804500,0.807600,0.810675,0.813725,0.816750,0.819750,0.822800,0.825800,0.828750,0.831675,0.834575,0.837450,0.840300,0.843125,0.845925,0.848700,0.851450,0.854175,0.856875,0.859550,0.862200,0.864825,0.867425,0.870000,0.872550,0.875075,0.877575,0.880050,0.882500,0.884925,0.887325,0.889700,0.892050,0.894375,0.896675,0.898950,0.901200,0.903425,0.905625,0.907800,0.909950,0.912075,0.914175,0.916250,0.918300,0.920325,0.922325,0.924300,0.926250,0.928175,0.930075,0.931950,0.933800,0.935625,0.937425,0.939200,0.940950,0.942675,0.944375,0.946050,0.947700,0.949325,0.950925,0.952500,0.954050,0.955575,0.957075,0.958550,0.959999,0.961425,0.962825,0.964200,0.965550,0.966875,0.968175,0.969450,0.970700,0.971925,0.973125,0.974300,0.975450,0.976575,0.977675,0.978750,0.979800,0.980825,0.981825,0.982800,0.983750,0.984675,0.985575,0.986450,0.987300,0.988125,0.988925,0.989700,0.990450,0.991175,0.991875,0.992550,0.993200,0.993825,0.994425,0.995000,0.995550,0.996075,0.996575,0.997050,0.997500,0.997925,0.998325,0.998700,0.999050,0.999375,0.999675,0.999950,1.000000,0.999981,0.999925,0.999831,0.999699,0.999529,0.999322,0.999078,0.998795,0.998476,0.998125,0.997750,0.997350,0.996925,0.996475,0.996000,0.995500,0.994975,0.994425,0.993850,0.993250,0.992625,0.991975,0.991300,0.990600,0.989875,0.989125,0.988350,0.987550,0.986725,0.985875,0.984999,0.984099,0.983175,0.982225,0.981250,0.980250,0.979225,0.978175,0.977100,0.976000,0.974875,0.973725,0.972550,0.971350,0.970125,0.968875,0.967600,0.966300,0.964975,0.963625,0.962250,0.960850,0.959425,0.957975,0.956500,0.955000,0.953475,0.951925,0.950350,0.948750,0.947125,0.945475,0.943800,0.942100,0.940375,0.938625,0.936850,0.935050,0.933225,0.931375,0.929500,0.927600,0.925675,0.923725,0.921750,0.919750,0.917725,0.915675,0.913600,0.911500,0.909375,0.907225,0.905050,0.902850,0.900625,0.898375,0.896100,0.893800,0.891475,0.889125,0.886750,0.884350,0.881925,0.879475,0.877000,0.874500,0.872000,0.869475,0.866925,0.864350,0.861750,0.859125,0.856475,0.853800,0.851100,0.848375,0.845625,0.842850,0.840050,0.837225,0.834375,0.831500,0.828600,0.825675,0.822725,0.819750,0.816750,0.813725,0.810675,0.807600,0.804500,0.801375,0.798225,0.795050,0.791850,0.788625,0.785375,0.782100,0.778800,0.775475,0.772125,0.768750,0.765375,0.761925,0.758475,0.755000,0.751500,0.748000,0.744475,0.740925,0.737375,0.733750,0.730125,0.726475,0.722800,0.719100,0.715375,0.711625,0.707850,0.704050,0.700225,0.696375,0.692500,0.688625,0.684675,0.680725,0.676750,0.672750,0.668725,0.664675,0.660600,0.656500,0.652375,0.648225,0.644050,0.639850,0.635625,0.631375,0.627100,0.622800,0.618475,0.614125,0.609750,0.605375,0.600925,0.596475,0.592000,0.587500,0.583000,0.578475,0.573925,0.569350,0.564750,0.560125,0.555500,0.550875,0.546225,0.541550,0.536850,0.532125,0.527375,0.522600,0.517800,0.513000,0.508175,0.503325,0.498450,0.493550,0.488625,0.483725,0.478800,0.473850,0.468875,0.463875,0.458850,0.453800,0.448725,0.443625,0.438500,0.433350,0.428175,0.422975,0.417750,0.412500,0.407225,0.401925,0.396600,0.391250,0.385875,0.380475,0.375050,0.369600,0.364125,0.358625,0.353100,0.347550,0.341975,0.336375,0.330750,0.325100,0.319425,0.313725,0.308000,0.302250,0.296475,0.290675,0.284850,0.279000,0.273125,0.267225,0.261300,0.255350,0.249375,0.243375,0.237350,0.231300,0.225225,0.219125,0.213000,0.206850,0.200675,0.194475,0.188250,0.182000,0.175725,0.169425,0.163100,0.156750,0.150375,0.143975,0.137550,0.131100,0.124625,0.118125,0.111600,0.105050,0.098475,0.091875,0.085250,0.078600,0.071925,0.065225,0.058500,0.051750,0.045000,0.038225,0.031425,0.024600,0.017750,0.010875,0.003975,0.0.000000,0.003975,0.010875,0.017750,0.024600,0.031425,0.038225,0.045000,0.051750,0.058500,0.065225,0.071925,0.078600,0.085250,0.091875,0.098475,0.105050,0.111600,0.118125,0.124625,0.131100,0.137550,0.143975,0.150375,0.156750,0.163100,0.169425,0.175725,0.182000,0.188250,0.194475,0.200675,0.206850,0.213000,0.219125,0.225225,0.231300,0.237350,0.243375,0.249375,0.255350,0.261300,0.267225,0.273125,0.279000,0.284850,0.290675,0.296475,0.302250,0.308000,0.313725,0.319425,0.325100,0.330750,0.336375,0.341975,0.347550,0.353100,0.358625,0.364125,0.369600,0.375050,0.380475,0.385875,0.391250,0.396600,0.401925,0.407225,0.412500,0.417750,0.422975,0.428175,0.433350,0.438500,0.443625,0.448725,0.453800,0.458850,0.463875,0.468875,0.473850,0.478800,0.483725,0.488625,0.493550,0.498450,0.503325,0.508175,0.513000,0.517800,0.522600,0.527375,0.532125,0.536850,0.541550,0.546225,0.550875,0.555500,0.560125,0.564750,0.569350,0.573925,0.578475,0.583000,0.587500,0.592000,0.596475,0.600925,0.605375,0.609750,0.614125,0.618475,0.622800,0.627100,0.631375,0.635625,0.639850,0.644050,0.648225,0.652375,0.656500,0.660600,0.664675,0.668725,0.672750,0.676750,0.680725,0.684675,0.688625,0.692500,0.696375,0.699950,0.703925,0.707875,0.711800,0.715675,0.719500,0.723275,0.727000,0.730675,0.734300,0.737875,0.741400,0.744875,0.748300,0.751675,0.755000,0.758275,0.761500,0.764675,0.767800,0.770875,0.773900,0.776875,0.779800,0.782675,0.785500,0.788275,0.791000,0.793675,0.796300,0.798875,0.801400,0.803875,0.806300,0.808675,0.811000,0.813275,0.815500,0.817675,0.819800,0.821875,0.823900,0.825875,0.827800,0.829675,0.831500,0.833275,0.835000,0.836675,0.838300,0.839875,0.841400,0.842875,0.844300,0.845675,0.847000,0.848300,0.849575,0.850800,0.851975,0.853100,0.854175,0.855200,0.856175,0.857100,0.858000,0.858875,0.859700,0.860475,0.861200,0.861875,0.862500,0.863075,0.863600,0.864075,0.864500,0.864875,0.865200,0.865475,0.865700,0.865875,0.866000,0.866075,0.866100,0.866125,0.866150,0.866175,0.866200,0.866225,0.866250,0.866275,0.866300,0.866325,0.866350,0.866375,0.866400,0.866425,0.866450,0.866475,0.866500,0.866525,0.866550,0.866575,0.866600,0.866625,0.866650,0.866675,0.866700,0.866725,0.866750,0.866775,0.866800,0.866825,0.866850,0.866875,0.866900,0.866925,0.866950,0.866975,0.867000,0.867025,0.867050,0.867075,0.867100,0.867125,0.867150,0.867175,0.867200,0.867225,0.867250,0.867275,0.867300,0.867325,0.867350,0.867375,0.867400,0.867425,0.867450,0.867475,0.867500,0.867525,0.867550,0.867575,0.867600,0.867625,0.867650,0.867675,0.867700,0.867725,0.867750,0.867775,0.867800,0.867825,0.867850,0.867875,0.867900,0.867925,0.867950,0.867975,0.868000,0.868025,0.868050,0.868075,0.868100,0.868125,0.868150,0.868175,0.868200,0.868225,0.868250,0.868275,0.868300,0.868325,0.868350,0.868375,0.868400,0.868425,0.868450,0.868475,0.868500,0.868525,0.868550,0.868575,0.868600,0.868625,0.868650,0.868675,0.868700,0.868725,0.868750,0.868775,0.868800,0.868825,0.868850,0.868875,0.868900,0.868925,0.868950,0.868975,0.869000,0.869025,0.869050,0.869075,0.869100,0.869125,0.869150,0.869175,0.869200,0.869225,0.869250,0.869275,0.869300,0.869325,0.869350,0.869375,0.869400,0.869425,0.869450,0.869475,0.869500,0.869525,0.869550,0.869575,0.869600,0.869625,0.869650,0.869675,0.869700,0.869725,0.869750,0.869775,0.869800,0.869825,0.869850,0.869875,0.869900,0.869925,0.869950,0.869975,0.870000,0.870025,0.870050,0.870075,0.870100,0.870125,0.870150,0.870175,0.870200,0.870225,0.870250,0.870275,0.870300,0.870325,0.870350,0.870375,0.870400,0.870425,0.870450,0.870475,0.870500,0.870525,0.870550,0.870575,0.870600,0.870625,0.870650,0.870675,0.870700,0.870725,0.870750,0.870775,0.870800,0.870825,0.870850,0.870875,0.870900,0.870925,0.870950,0.870975,0.871000,0.871025,0.871050,0.871075,0.871100,0.871125,0.871150,0.871175,0.871200,0.871225,0.871250,0.871275,0.871300,0.871325,0.871350,0.871375,0.871400,0.871425,0.871450,0.871475,0.871500,0.871525,0.871550,0.871575,0.871600,0.871625,0.871650,0.871675,0.871700,0.871725,0.871750,0.871775,0.871800,0.871825,0.871850,0.871875,0.871900,0.871925,0.871950,0.871975,0.872000,0.872025,0.872050,0.872075,0.872100,0.872125,0.872150,0.872175,0.872200,0.872225,0.872250,0.872275,0.872300,0.872325,0.872350,0.872375,0.872400,0.872425,0.872450,0.872475,0.872500,0.872525,0.872550,0.872575,0.872600,0.872625,0.872650,0.872675,0.872700,0.872725,0.872750,0.872775,0.872800,0.872825,0.872850,0.872875,0.872900,0.872925,0.872950,0.872975,0.873000,0.873025,0.873050,0.873075,0.873100,0.873125,0.873150,0.873175,0.873200,0.873225,0.873250,0.873275,0.873300
```


- **Memory partitioning:** #pragma HLS ARRAY_PARTITION is used to partition an array into multiple smaller arrays with more number of ports for read and write operations. This results in improved throughput of the design.

```

DTYPE in_R[SIZE], in_I[SIZE];
DTYPE Stage0_R[SIZE], Stage0_I[SIZE];
DTYPE Stage1_R[SIZE], Stage1_I[SIZE];
DTYPE Stage2_R[SIZE], Stage2_I[SIZE];
DTYPE Stage3_R[SIZE], Stage3_I[SIZE];
DTYPE Stage4_R[SIZE], Stage4_I[SIZE];
DTYPE Stage5_R[SIZE], Stage5_I[SIZE];
DTYPE Stage6_R[SIZE], Stage6_I[SIZE];
DTYPE Stage7_R[SIZE], Stage7_I[SIZE];
DTYPE Stage8_R[SIZE], Stage8_I[SIZE];
DTYPE Stage9_R[SIZE], Stage9_I[SIZE];
DTYPE buf_o_R[SIZE], buf_o_I[SIZE];
const int Factor = fac_ary;

#pragma HLS ARRAY_PARTITION variable = in_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = in_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage0_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage1_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage1_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage2_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage3_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage3_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage4_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage5_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage5_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage6_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage6_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage7_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage7_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage8_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage8_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage9_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = Stage9_I type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = buf_o_R type = cyclic factor = Factor
#pragma HLS ARRAY_PARTITION variable = buf_o_I type = cyclic factor = Factor

```

```

1 void fft_stages(DTYPE X_R[SIZE], DTYPE X_I[SIZE], int stage, DTYPE OUT_R[SIZE], DTYPE OUT_I[SIZE])
2 {
3     #pragma HLS DEPENDENCE dependent = false type = inter variable = OUT_R
4     #pragma HLS DEPENDENCE dependent = false type = inter variable = OUT_I
5     // Insert your code here

```

- **Pipelining**
#pragma HLS pipeline

```

1 void fft_stages(DTYPE X_R[SIZE], DTYPE X_I[SIZE], int stage, DTYPE OUT_R[SIZE], DTYPE OUT_I[SIZE])
2 {
3     #pragma HLS DEPENDENCE dependent = false type = inter variable = OUT_R
4     #pragma HLS DEPENDENCE dependent = false type = inter variable = OUT_I
5     // Insert your code here
6
7     if (stage == 2)
8     {
9         unsigned short j, j_lower, k, idx;
10        DTYPE c_twiddle, s_twiddle;
11        DTYPE in_R, in_I, in_R_lower, in_I_lower, temp_R, temp_I;
12
13        Stage_loop_2:
14        for (unsigned short i = 0; i < SIZE2; i++)
15        {
16            #pragma HLS PIPELINE
17
18            k = i & 1;
19            idx = k << 8;
20            j = i + (i & ~1);
21            j_lower = j + 2;
22
23            c_twiddle = W_real[idx];
24            s_twiddle = W_imag[idx];
25
26            in_R = X_R[j];
27            in_I = X_I[j];
28            in_R_lower = X_R[j_lower];
29            in_I_lower = X_I[j_lower];
30
31            temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;
32            temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;
33
34            OUT_R[j_lower] = in_R - temp_R;
35            OUT_I[j_lower] = in_I - temp_I;
36            OUT_R[j] = in_R + temp_R;
37            OUT_I[j] = in_I + temp_I;
38        }
39    }

```

```
#pragma HLS PIPELINE
```

```
    k = i & 3;  
    idx = k << 7;  
    j = i + (i & ~3);  
    j_lower = j + 4;
```

```
    c_twiddle = W_real[idx];  
    s_twiddle = W_imag[idx];
```

```
    in_R = X_R[j];  
    in_I = X_I[j];  
    in_R_lower = X_R[j_lower];  
    in_I_lower = X_I[j_lower];
```

```
    temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;  
    temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;
```

```
    OUT_R[j_lower] = in_R - temp_R;  
    OUT_I[j_lower] = in_I - temp_I;  
    OUT_R[j] = in_R + temp_R;  
    OUT_I[j] = in_I + temp_I;
```

```
#pragma HLS PIPELINE
```

```
    k = i & 7;  
    idx = k << 6;  
    j = i + (i & ~7);  
    j_lower = j + 8;
```

```
    c_twiddle = W_real[idx];  
    s_twiddle = W_imag[idx];
```

```
    in_R = X_R[j];  
    in_I = X_I[j];  
    in_R_lower = X_R[j_lower];  
    in_I_lower = X_I[j_lower];
```

```
    temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;  
    temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;
```

```
    OUT_R[j_lower] = in_R - temp_R;  
    OUT_I[j_lower] = in_I - temp_I;  
    OUT_R[j] = in_R + temp_R;  
    OUT_I[j] = in_I + temp_I;
```

```
#pragma HLS PIPELINE
```

```
    k = i & 15;  
    idx = k << 5;  
    j = i + (i & ~15);  
    j_lower = j + 16;
```

```
    c_twiddle = W_real[idx];  
    s_twiddle = W_imag[idx];
```

```
    in_R = X_R[j];  
    in_I = X_I[j];  
    in_R_lower = X_R[j_lower];  
    in_I_lower = X_I[j_lower];
```

```
    temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;  
    temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;
```

```
    OUT_R[j_lower] = in_R - temp_R;  
    OUT_I[j_lower] = in_I - temp_I;  
    OUT_R[j] = in_R + temp_R;  
    OUT_I[j] = in_I + temp_I;
```

```
}
```

```
8 #pragma HLS PIPELINE
```

```
9  
0    k = i & 31;  
1    idx = k << 4;  
2    j = i + (i & ~31);  
3    j_lower = j + 32;
```

```
4  
5    c_twiddle = W_real[idx];  
6    s_twiddle = W_imag[idx];
```

```
7  
8    in_R = X_R[j];  
9    in_I = X_I[j];  
0    in_R_lower = X_R[j_lower];  
1    in_I_lower = X_I[j_lower];
```

```
2  
3    temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;  
4    temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;
```

```
5  
6    OUT_R[j_lower] = in_R - temp_R;  
7    OUT_I[j_lower] = in_I - temp_I;  
8    OUT_R[j] = in_R + temp_R;  
9    OUT_I[j] = in_I + temp_I;
```

```
0 }
```



```

#pragma HLS PIPELINE

    k = i & 63;
    idx = k << 3;
    j = i + (i & ~63);
    j_lower = j + 64;

    c_twiddle = W_real[idx];
    s_twiddle = W_imag[idx];

    in_R = X_R[j];
    in_I = X_I[j];
    in_R_lower = X_R[j_lower];
    in_I_lower = X_I[j_lower];

    temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;
    temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;

    OUT_R[j_lower] = in_R - temp_R;
    OUT_I[j_lower] = in_I - temp_I;
    OUT_R[j] = in_R + temp_R;
    OUT_I[j] = in_I + temp_I;

```

```

#pragma HLS PIPELINE

    k = i & 127;
    idx = k << 2;
    j = i + (i & ~127);
    j_lower = j + 128;

    c_twiddle = W_real[idx];
    s_twiddle = W_imag[idx];

    in_R = X_R[j];
    in_I = X_I[j];
    in_R_lower = X_R[j_lower];
    in_I_lower = X_I[j_lower];

    temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;
    temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;

    OUT_R[j_lower] = in_R - temp_R;
    OUT_I[j_lower] = in_I - temp_I;
    OUT_R[j] = in_R + temp_R;
    OUT_I[j] = in_I + temp_I;
}

```

```

#pragma HLS PIPELINE

    k = i & 255;
    idx = k << 1;
    j = i + (i & ~255);
    j_lower = j + 256;

    c_twiddle = W_real[idx];
    s_twiddle = W_imag[idx];

    in_R = X_R[j];
    in_I = X_I[j];
    in_R_lower = X_R[j_lower];
    in_I_lower = X_I[j_lower];

    temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;
    temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;

    OUT_R[j_lower] = in_R - temp_R;
    OUT_I[j_lower] = in_I - temp_I;
    OUT_R[j] = in_R + temp_R;
    OUT_I[j] = in_I + temp_I;
}

```

```

#pragma HLS pipeline
    j = i + ((i >> 9) << 9);
    j_lower = j + (1 << 9);

    c_twiddle = W_real[i];
    s_twiddle = W_imag[i];

    in_R = X_R[j];
    in_I = X_I[j];
    in_R_lower = X_R[j_lower];
    in_I_lower = X_I[j_lower];

    temp_R = in_R_lower * c_twiddle - in_I_lower * s_twiddle;
    temp_I = in_I_lower * c_twiddle + in_R_lower * s_twiddle;

    OUT_R[j_lower] = in_R - temp_R;
    OUT_I[j_lower] = in_I - temp_I;
    OUT_R[j] = in_R + temp_R;
    OUT_I[j] = in_I + temp_I;
}

```

- **Pragma HLS inline:** Removes a function as a separate entity in the hierarchy. After inlining, the function is dissolved into the calling function and no longer appears as a separate level of hierarchy in the RTL. In some cases, inlining a function allows operations within the function to be shared and optimized more effectively with the calling function. However, an inlined function cannot be shared or reused, so if the parent function calls the inlined function multiple times, this can increase the area required for implementing the RTL.

#pragma HLS inline region

It was removed, because it was causing II Violation

TEST BENCH : PASS

```

1 INFO: [SIM 2] ***** CSIM start *
2 INFO: [SIM 4] CSIM will launch GCC as the
3   Compiling ../../../../fft.cpp in debug
4   Generating csim.exe
5 INPUTS
6 Printing FFT Output
7   0   0.000000  0.000000
8   1  512.000000  0.000000
9   2  256.000000  0.000000
10  3  768.000000  0.000000
11  4  128.000000  0.000000
12  5  640.000000  0.000000
13  6  384.000000  0.000000
14  7  896.000000  0.000000
15  8   64.000000  0.000000
16  9  576.000000  0.000000
17 10  320.000000  0.000000

```

```

1027 1020    255.000000  0.000000
1028 1021    767.000000  0.000000
1029 1022    511.000000  0.000000
1030 1023   1023.000000  0.000000
1031 Comparing against output data
1032 -----
1033      RMSE(R)          RMSE(I)
1034 0.047008574008942  0.013548693619668
1035 -----
1036 *****
1037 PASS: The output matches the golden output!
1038 *****
1039 INFO: [SIM 1] CSim done with 0 errors.
1040 INFO: [SIM 3] ***** CSIM finish *****
1041

```

Target	Estimated	Uncertainty
10.00 ns	9.448 ns	2.70 ns

fft1024_Baseline

Target	Estimated	Uncertainty
10.00 ns	7.256 ns	2.70 ns

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
▼ m				-	273932	2.739E6	-	136450	-	dataflow	98	234	25653	38194	0
▶ bit_reverse			-	-	1026	1.026E4	-	1026	-	no	0	0	25	66	0
▶ ft_stage_first			-	-	529	5.290E3	-	529	-	no	0	18	2303	3206	0
▶ ft_stages_1			-	-	1067	1.067E4	-	1067	-	no	2	24	2592	3852	0
▶ ft_stages_2			-	-	2133	2.133E4	-	2133	-	no	2	24	2594	3854	0
▶ ft_stages_3			-	-	4265	4.265E4	-	4265	-	no	2	24	2594	3854	0
▶ ft_stages_4			-	-	8529	8.529E4	-	8529	-	no	2	24	2598	3857	0
▶ ft_stages_5			-	-	17057	1.710E5	-	17057	-	no	2	24	2600	3858	0
▶ ft_stages_6			-	-	34113	3.410E5	-	34113	-	no	2	24	2602	3859	0
▶ ft_stages_7			-	-	68225	6.820E5	-	68225	-	no	2	24	2604	3860	0
▶ ft_stages			-	-	136449	1.364E6	-	136449	-	no	2	24	2606	3859	0
▶ ft_stage_last			-	-	529	5.290E3	-	529	-	no	2	24	2513	3765	0

fft1024_Best

Target	Estimated	Uncertainty
10.00 ns	7.300 ns	2.70 ns

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
▼ m				-	7901	7.901E4	-	7902	-	no	91	24	15353	18951	0
▶ ft_Pipeline_1			-	-	1027	1.027E4	-	1027	-	no	0	0	69	75	0
▶ ft_Pipeline_2			-	-	1027	1.027E4	-	1027	-	no	0	0	69	75	0
▶ ft_Pipeline_Reverse_Operation			-	-	516	5.160E3	-	516	-	no	1	0	368	192	0
▶ ft_stage_first			-	-	524	5.240E3	-	524	-	no	0	0	963	1448	0
▶ ft_Pipeline_Stage_loop_2			-	-	530	5.300E3	-	530	-	no	0	0	1167	386	0
▶ ft_Pipeline_Stage_loop_3			-	-	530	5.300E3	-	530	-	no	0	0	970	272	0
▶ ft_Pipeline_Stage_loop_4			-	-	530	5.300E3	-	530	-	no	0	0	1167	386	0
▶ ft_Pipeline_Stage_loop_5			-	-	530	5.300E3	-	530	-	no	0	0	970	272	0
▶ ft_Pipeline_Stage_loop_6			-	-	530	5.300E3	-	530	-	no	0	0	905	258	0
▶ ft_Pipeline_Stage_loop_7			-	-	530	5.300E3	-	530	-	no	0	0	905	258	0
▶ ft_Pipeline_Stage_loop_8			-	-	530	5.300E3	-	530	-	no	0	0	905	258	0
▶ ft_Pipeline_Stage_loop_9			-	-	530	5.300E3	-	530	-	no	0	0	905	258	0
▶ ft_Pipeline_last_stage_loop			-	-	529	5.290E3	-	529	-	no	0	0	836	226	0
▶ ft_Pipeline_13			-	-	1027	1.027E4	-	1027	-	no	0	0	50	82	0
▶ ft_Pipeline_14			-	-	1027	1.027E4	-	1027	-	no	0	0	50	82	0

Cosimulation Report for 'fft'

General Information

Date: Sun Dec 1 02:42:38 PM PST 2024
Version: 2022.2 (Build 3670227 on Oct 13 2022)
Project: hls
Status: **Pass**

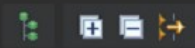
Solution: solution1 (Vivado IP F
Product family: zynq
Target device: xc7z020-clg400-1

Cosim Options

Tool: Vivado XSIM

RTL: Verilog

Performance Estimates



Modules & Loops	Avg II	Max II	Min II	Avg Latency	Max Latency	Min Latency
▼ ● fft				8027	8027	8027
▶ ● fft_Pipeline_1				1028	1028	1028
▶ ● fft_Pipeline_2				1028	1028	1028
▶ ● fft_Pipeline_Reverse_Operation				514	514	514
▶ ● fft_stage_first				522	522	522
▶ ● fft_Pipeline_Stage_loop_2				528	528	528
▶ ● fft_Pipeline_Stage_loop_3				528	528	528
▶ ● fft_Pipeline_Stage_loop_4				528	528	528
▶ ● fft_Pipeline_Stage_loop_5				528	528	528
▶ ● fft_Pipeline_Stage_loop_6				528	528	528
▶ ● fft_Pipeline_Stage_loop_7				528	528	528
▶ ● fft_Pipeline_Stage_loop_8				528	528	528
▶ ● fft_Pipeline_Stage_loop_9				528	528	528
▶ ● fft_Pipeline_last_stage_loop				527	527	527
▶ ● fft_Pipeline_13				1138	1138	1138
▶ ● fft_Pipeline_14				1138	1138	1138

INTERFACE DESIGN

```
void fft(DTYPE X_R[SIZE], DTYPE X_I[SIZE], DTYPE OUT_R[SIZE], DTYPE OUT_I[SIZE])
{

#pragma HLS INTERFACE s_axilite port=return bundle=fft

#pragma HLS INTERFACE m_axi depth=1024 port=OUT_R offset=slave bundle=output1
#pragma HLS INTERFACE m_axi depth=1024 port=OUT_I offset=slave bundle=output2

#pragma HLS INTERFACE m_axi depth=1024 port=X_R offset=slave bundle=input1
#pragma HLS INTERFACE m_axi depth=1024 port=X_I offset=slave bundle=input2

#pragma HLS INTERFACE s_axilite port=X_R bundle=fft
#pragma HLS INTERFACE s_axilite port=X_I bundle=fft
#pragma HLS INTERFACE s_axilite port=OUT_R bundle=fft
#pragma HLS INTERFACE s_axilite port=OUT_I bundle=fft
...
}
```

Synthesis

Synthesis Summary Report of 'fft'

▼ General Information

Date: Thu Dec 12 22:54:55 2024
Version: 2022.2 (Build 3670227 on Oct 13 2022)
Project: hls












```
Solution:      solution1 (Vivado IP Flow Target)
Product family: zynq
Target device: xc7z020-clg400-1
```

▼ Timing Estimate


Target	Estimated	Uncertainty
10.00 ns	7.300 ns	2.70 ns







▼ Performance & Resource Estimates ⓘ

☒ Modules
 ☒ Loops

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	
▼  fft				-	7901	7.901E4		-	7902	-	no
▶  fft_Pipeline_1				-	1027	1.027E4		-	1027	-	no
▶  fft_Pipeline_2				-	1027	1.027E4		-	1027	-	no
▶  fft_Pipeline_Reverse_Operation				-	516	5.160E3		-	516	-	no
▶  fft_stage_first				-	524	5.240E3		-	524	-	no
▶  fft_Pipeline_Stage_loop_2				-	530	5.300E3		-	530	-	no
▶  fft_Pipeline_Stage_loop_3				-	530	5.300E3		-	530	-	no
▶  fft_Pipeline_Stage_loop_4				-	530	5.300E3		-	530	-	no
▶  fft_Pipeline_Stage_loop_5				-	530	5.300E3		-	530	-	no
▶  fft_Pipeline_Stage_loop_6				-	530	5.300E3		-	530	-	no
▶  fft_Pipeline_Stage_loop_7				-	530	5.300E3		-	530	-	no

[Console](#)
[Errors](#)
[Warnings](#)
[Guidance](#)
[Properties](#)
[Man Pages](#)
[Git Repositories](#)
[Modules/Loops](#)





 76 Guidance-Infos
  0 Guidance-Warnings
  0 Guidance-Errors
 

Name	Web Help	Details
▼ 📁 All Categories		
▼ 📁 SCHEDULE		
🔗 [HLS 200-1470]		Pipelining result : Target II = NA, Final II = 1, Depth = 3, loop 'Loop 1'
🔗 [HLS 200-1470]		Pipelining result : Target II = NA, Final II = 1, Depth = 3, loop 'Loop 1'
🔗 [HLS 200-1470]		Pipelining result : Target II = NA, Final II = 1, Depth = 4, loop 'Reverse_Operation'
🔗 [HLS 200-1470]		Pipelining result : Target II = NA, Final II = 1, Depth = 12, loop 'First_stage_loop'
🔗 [HLS 200-1470]		Pipelining result : Target II = NA, Final II = 1, Depth = 18, loop 'Stage_loop 2'

Cosimulation Report for 'fft'

General Information

Date: Thu Dec 12 10:58:10 PM PST 2024
Version: 2022.2 (Build 3670227 on Oct 13 2022)
Project: hls
Status: **Pass**

Solution: solution1 (Vivado IP Flow Target)
Product family: zynq
Target device: xc7z020-clg400-1

Cosim Options

Tool: Vivado XSIM

RTL: Verilog

Performance Estimates



Modules & Loops	Avg II	Max II	Min II	Avg Latency	Max Latency	Min Latency
▼ fft				8027	8027	8027
▶ fft_Pipeline_1				1028	1028	1028
▶ fft_Pipeline_2				1028	1028	1028
▶ fft_Pipeline_Reverse_Operation				514	514	514
▶ fft_stage_first				522	522	522
▶ fft_Pipeline_Stage_loop_2				528	528	528
▶ fft_Pipeline_Stage_loop_3				528	528	528
▶ fft_Pipeline_Stage_loop_4				528	528	528
▶ fft_Pipeline_Stage_loop_5				528	528	528
▶ fft_Pipeline_Stage_loop_6				528	528	528
▶ fft_Pipeline_Stage_loop_7				528	528	528
▶ fft_Pipeline_Stage_loop_8				528	528	528
▶ fft_Pipeline_Stage_loop_9				528	528	528
▶ fft_Pipeline_last_stage_loop				527	527	527
▶ fft Pipeline 13				1138	1138	1138

Console Errors Warnings Guidance x Properties Man Pages Git Repositories Modules/Loops

77 Guidance-Infos 0 Guidance-Warnings 0 Guidance-Errors

Name

Web Help

Details

▼ All Categories

▼ SCHEDULE

[HLS 200-1470]

Pipelining result : Target II = NA, Final II = 1, Depth = 3, loop 'Loop 1'

[HLS 200-1470]

Pipelining result : Target II = NA, Final II = 1, Depth = 3, loop 'Loop 1'

[HLS 200-1470]

Pipelining result : Target II = NA, Final II = 1, Depth = 4, loop 'Reverse_Operation'

[HLS 200-1470]

Pipelining result : Target II = NA, Final II = 1, Depth = 12, loop 'First_stage_loop'

[HLS 200-1470]

Pipelining result : Target II = NA, Final II = 1, Depth = 18, loop 'Stage_loop_2'

▼ HW Interfaces

▼ M_AXI

Interface	Data Width (SW->HW)	Address Width	Latency	Offset	Register	Max Widen Bitwidth	Max Read Burst Length	Max Write Burst Length	Num
m_axi_input1	32 -> 32	32	0	slave	0	0	16	16	
m_axi_input2	32 -> 32	32	0	slave	0	0	16	16	
m_axi_output1	32 -> 32	32	0	slave	0	0	16	16	
m_axi_output2	32 -> 32	32	0	slave	0	0	16	16	

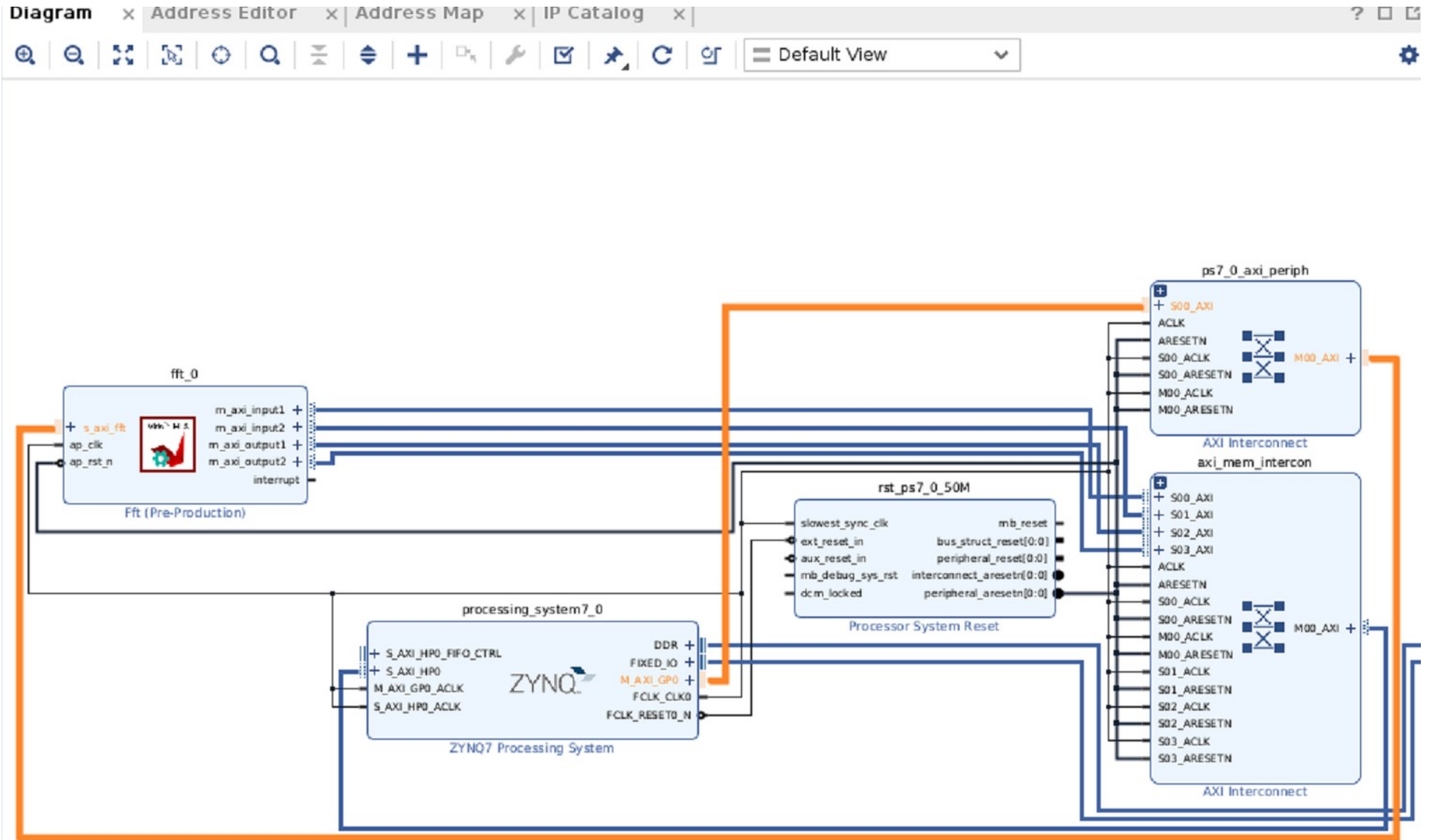
▼ S_AXILITE Interfaces

Interface	Data Width	Address Width	Offset	Register
s_axi_fft	32	6	16	0

▼ S_AXILITE Registers

Interface	Register	Offset	Width	Access	Description	Bit Fields
s_axi_fft	CTRL	0x00	32	RW	Control signals	0=AP_START 1=AP_DONE 2=AP_IDLE 3=AP_READY 7=AUTO_RESTART 9=INTERRUPT
s_axi_fft	GIER	0x04	32	RW	Global Interrupt Enable Register	0=Enable
s_axi_fft	IP_IER	0x08	32	RW	IP Interrupt Enable Register	0=CHAN0_INT_EN 1=CHAN1_INT_EN
s_axi_fft	IP_ISR	0x0c	32	RW	IP Interrupt Status Register	0=CHAN0_INT_ST 1=CHAN1_INT_ST
s_axi_fft	X_R	0x10	32	W	Data signal of X_R	
s_axi_fft	X_I	0x18	32	W	Data signal of X_I	
s_axi_fft	OUT_R	0x20	32	W	Data signal of OUT_R	
s_axi_fft	OUT_I	0x28	32	W	Data signal of OUT_I	

VIVADO



FFT TESTBENCH

This notebook takes two inputs (real and imaginary) and gives the real and imaginary parts of the FFT outputs using AXI4. It is then compared with software version of FFT

```
In [5]: 1 from pynq import Overlay
        2 import numpy as np
        3 from pynq import Xlnk
        4 from pynq.lib import dma
        5 from scipy.linalg import dft
        6 import matplotlib.pyplot as plt
        7 import time
```

```
-----
ImportError                                Traceback (most recent call last)
Input In [5], in <cell line: 3>()
      1 from pynq import Overlay
      2 import numpy as np
----> 3 from pynq import Xlnk
      4 from pynq.lib import dma
      5 from scipy.linalg import dft

ImportError: cannot import name 'Xlnk' from 'pynq' (/usr/local/share/pynq-venv/lib/python3.10/site-packages/pynq/__init__.py)
```

I had problems with **Xln**, so I had to implement **Allocate** instead

DEMO

fft1024_Best DEMO

FFT TESTBENCH

This notebook takes two inputs (real and imaginary) and gives the real and imaginary parts of the FFT outputs using AXI4. It is then compared with software version of FFT

```
In [1]: from pyng import Overlay
import numpy as np
from pyng import allocate
from pyng.lib import dma
from scipy.linalg import dft
import matplotlib.pyplot as plt
import time
```

```
In [2]: ol=Overlay('fft.bit')
```

```
In [3]: NUM_SAMPLES = 1024

real_error=np.zeros(NUM_SAMPLES)
imag_error=np.zeros(NUM_SAMPLES)
ind=np.arange(NUM_SAMPLES)
real_rse=np.zeros(NUM_SAMPLES)
imag_rse=np.zeros(NUM_SAMPLES)
```

```
In [5]: in_r = allocate(shape=(NUM_SAMPLES,), dtype=np.float32)
in_i = allocate(shape=(NUM_SAMPLES,), dtype=np.float32)
out_r = allocate(shape=(NUM_SAMPLES,), dtype=np.float32)
out_i = allocate(shape=(NUM_SAMPLES,), dtype=np.float32)
a = [1 for i in range(NUM_SAMPLES)]
a=np.cast(a)
# Change input real and imaginary value here
real=a.real
imag=a.imag
np.copyto(in_r, real)
np.copyto(in_i, imag)
```

```
In [6]: fft_ip = ol.fft_0
fft_ip.write(x10,in_r.device_address)
fft_ip.write(x10,in_i.device_address)
fft_ip.write(x20,out_r.device_address)
fft_ip.write(x20,out_i.device_address)
vrlime.time()
fft_ip.write(x00,1)
print(time.time()-v)
```

0.000072747802734375

Verifying Functionality

```
In [7]: c=time.time()
golden=np.fft.fft(a)
print(time.time()-c)
for i in range(NUM_SAMPLES):
    real_error[i]="{:0:.6f}".format(abs(out_r[i]-golden_op_real[i]))
    imag_error[i]="{:0:.6f}".format(abs(out_i[i]-golden_op_imag[i]))
```

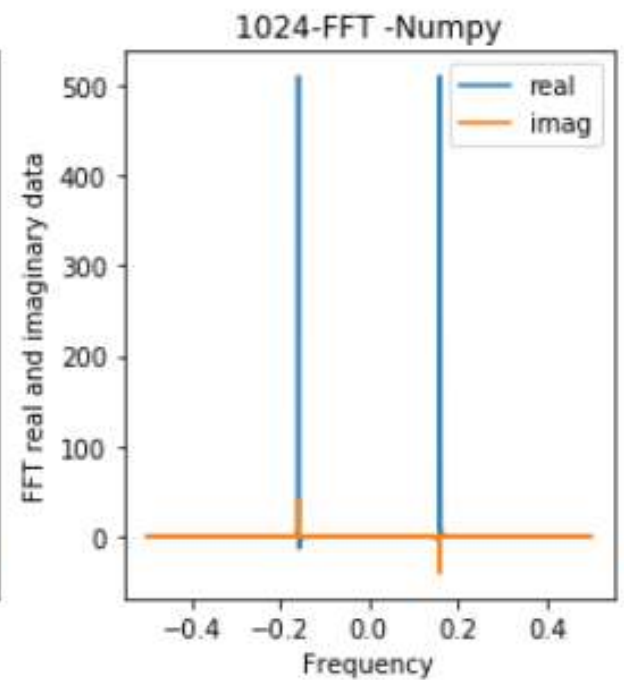
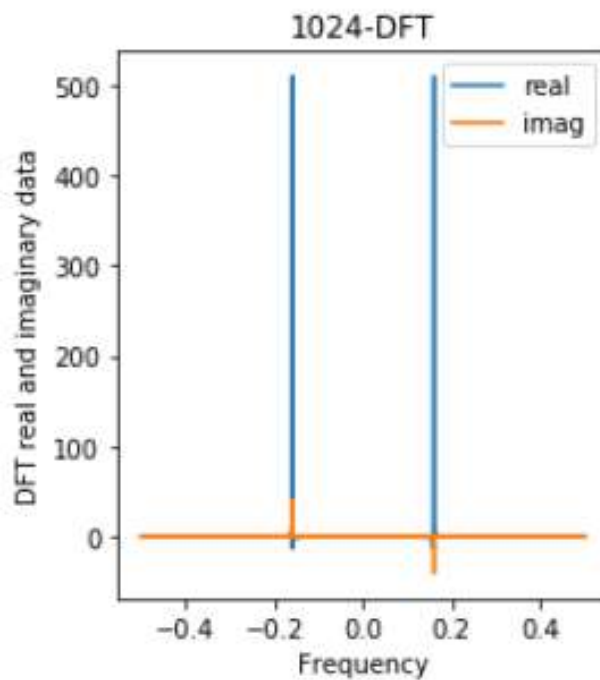
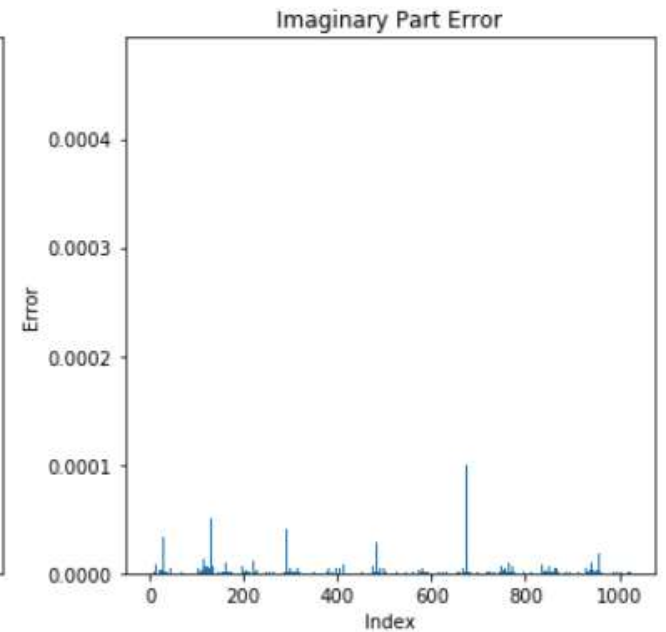
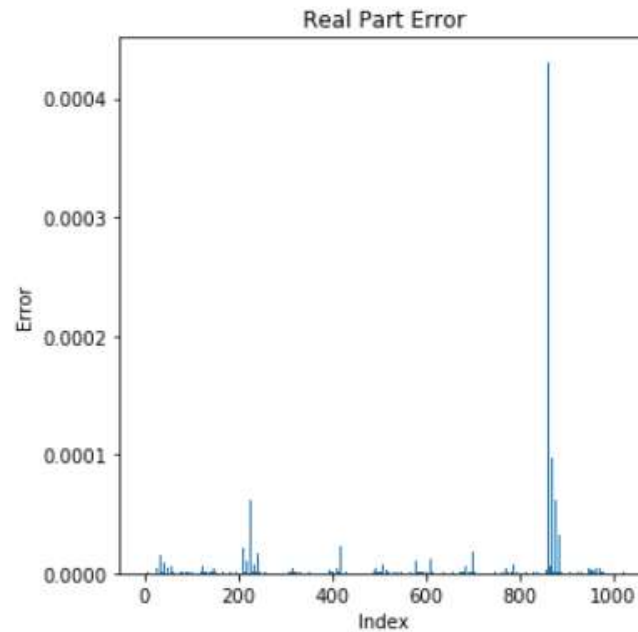
0.001837710407148438

```
In [8]: sum_sq_real=0
sum_sq_imag=0
for i in range(NUM_SAMPLES):
    sum_sq_real=sum_sq_real+(real_error[i]**2)
    sum_sq_imag=sum_sq_imag+(imag_error[i]**2)
    real_rse = np.sqrt(sum_sq_real / (i+1))
    imag_rse = np.sqrt(sum_sq_imag / (i+1))
print("Real Part RMSE: ", real_rse, "Imaginary Part RMSE:", imag_rse)
if real_rse<0.001 and imag_rse<0.001:
    print("PASS")
else:
    print("FAIL")
```

Real Part RMSE: 2.045955819550222e-05 Imaginary Part RMSE: 1.800892248477111e-05
PASS

```
In [11]: plt.figure(figsize=(18, 6))
plt.subplot(1,2,1)
plt.bar(ind,real_error)
plt.title("Real Part Error")
plt.xlabel("Index")
plt.ylabel("Error")
plt.xticks(ind)
plt.tight_layout()

plt.subplot(1,2,2)
plt.bar(ind,imag_error)
plt.title("Imaginary Part Error")
plt.xlabel("Index")
plt.ylabel("Error")
plt.xticks(ind)
plt.tight_layout()
```



OFDM Receiver

```
fft.cpp  ofdm_test.cpp  fft.h  ofdm_receive... x  »3
1 INFO: [SIM 2] ***** CSIM start *****
2 INFO: [SIM 4] CSIM will launch GCC as the compiler.
3   Compiling ../../../../fft.cpp in debug mode
4   Generating csim.exe
5 Reading INPUTS
6 Comparing with golden output
7 Comparing against output data
8 *****
9 PASS: The output matches the golden output!
10 *****
11 INFO: [SIM 1] CSim done with 0 errors.
12 INFO: [SIM 3] ***** CSIM finish *****
13 |
```

General Information

Date: Thu Dec 12 12:09:03 2024
Version: 2022.2 (Build 3670227 on Oct 13 2022)
Project: hls_ofdm_rx

Solution: solution1 (Vivado IP Flow Target)
Product family: zynq
Target device: xc7z020-cig400-1

Timing Estimate

Target	Estimated	Uncertainty
10.00 ns	7.256 ns	2.70 ns

Performance & Resource Estimates

Modules Loops

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
ofdm_receiver				-	8921	8.921E4	-	8922	-	no	101	24	12371	13509	0
fft				-	7887	7.887E4	-	7887	-	no	91	24	12111	12987	0
fft_Pipeline_1				-	1026	1.026E4	-	1026	-	no	0	0	24	62	0
fft_Pipeline_2				-	1026	1.026E4	-	1026	-	no	0	0	24	62	0
fft_Pipeline_Reverse_Operation				-	516	5.160E3	-	516	-	no	1	0	369	192	0
fft_stage_first				-	524	5.240E3	-	524	-	no	0	0	963	1448	0
fft_Pipeline_Stage_loop_2				-	530	5.300E3	-	530	-	no	0	0	1167	386	0
fft_Pipeline_Stage_loop_3				-	530	5.300E3	-	530	-	no	0	0	979	272	0
fft_Pipeline_Stage_loop_4				-	530	5.300E3	-	530	-	no	0	0	1167	386	0
fft_Pipeline_Stage_loop_5				-	530	5.300E3	-	530	-	no	0	0	979	272	0
fft_Pipeline_Stage_loop_6				-	530	5.300E3	-	530	-	no	0	0	905	258	0
fft_Pipeline_Stage_loop_7				-	530	5.300E3	-	530	-	no	0	0	905	258	0
fft_Pipeline_Stage_loop_8				-	530	5.300E3	-	530	-	no	0	0	905	258	0
fft_Pipeline_Stage_loop_9				-	530	5.300E3	-	530	-	no	0	0	905	258	0
fft_Pipeline_Stage_loop_10				-	529	5.290E3	-	529	-	no	0	0	836	226	0
fft_Pipeline_13				-	1027	1.027E4	-	1027	-	no	0	0	70	71	0
fft_Pipeline_14				-	1027	1.027E4	-	1027	-	no	0	0	70	71	0
ofdm_receiver_Pipeline_VITIS_LOOP_9_1				-	1029	1.029E4	-	1029	-	no	0	0	203	242	0
VITIS_LOOP_9_1				-	1027	1.027E4	5	1	1024	yes	-	-	-	-	-

Performance Pragma

Modules Loops

Modules & Loops	Target TI(cycles)	TI(cycles)	TI met
ofdm_receiver	-	-	-
fft	-	-	-
fft_Pipeline_1	-	-	-
fft_Pipeline_2	-	-	-
fft_Pipeline_Reverse_Operation	-	-	-
fft_stage_first	-	-	-
fft_Pipeline_Stage_loop_2	-	-	-
fft_Pipeline_Stage_loop_3	-	-	-
fft_Pipeline_Stage_loop_4	-	-	-
fft_Pipeline_Stage_loop_5	-	-	-
fft_Pipeline_Stage_loop_6	-	-	-
fft_Pipeline_Stage_loop_7	-	-	-
fft_Pipeline_Stage_loop_8	-	-	-
fft_Pipeline_Stage_loop_9	-	-	-
fft_Pipeline_Stage_loop_10	-	-	-
fft_Pipeline_13	-	-	-
fft_Pipeline_14	-	-	-
ofdm_receiver_Pipeline_VITIS_LOOP_9_1	-	-	-
VITIS_LOOP_9_1	-	-	-

HW Interfaces

AP_FIFO

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM
▼ ofdm_receiver				-	8921	8.921E4	-	8922	-	no	101	24	12371	13509	0
▼ fft				-	7887	7.887E4	-	7887	-	no	91	24	12111	12987	0
▶ fft_Pipeline_1				-	1026	1.026E4	-	1026	-	no	0	0	24	62	0
▶ fft_Pipeline_2				-	1026	1.026E4	-	1026	-	no	0	0	24	62	0
▶ fft_Pipeline_Reverse_Operation				-	516	5.160E3	-	516	-	no	1	0	369	192	0
▶ fft_stage_first				-	524	5.240E3	-	524	-	no	0	0	963	1448	0
▶ fft_Pipeline_Stage_loop_2				-	530	5.300E3	-	530	-	no	0	0	1167	386	0
▶ fft_Pipeline_Stage_loop_3				-	530	5.300E3	-	530	-	no	0	0	979	272	0
▶ fft_Pipeline_Stage_loop_4				-	530	5.300E3	-	530	-	no	0	0	1167	386	0
▶ fft_Pipeline_Stage_loop_5				-	530	5.300E3	-	530	-	no	0	0	979	272	0
▶ fft_Pipeline_Stage_loop_6				-	530	5.300E3	-	530	-	no	0	0	905	258	0
▶ fft_Pipeline_Stage_loop_7				-	530	5.300E3	-	530	-	no	0	0	905	258	0
▶ fft_Pipeline_Stage_loop_8				-	530	5.300E3	-	530	-	no	0	0	905	258	0
▶ fft_Pipeline_Stage_loop_9				-	530	5.300E3	-	530	-	no	0	0	905	258	0
▶ fft_Pipeline_last_stage_loop				-	529	5.290E3	-	529	-	no	0	0	836	226	0
▶ fft_Pipeline_13				-	1027	1.027E4	-	1027	-	no	0	0	70	71	0
▶ fft_Pipeline_14				-	1027	1.027E4	-	1027	-	no	0	0	70	71	0
▼ ofdm_receiver_Pipeline_VITIS_LOOP_9_1				-	1029	1.029E4	-	1029	-	no	0	0	203	242	0
VITIS_LOOP_9_1				-	1027	1.027E4	5	1	1024	yes	-	-	-	-	-

DEMO for OFDM receiver was not required