中国科学院大学

University of Chinese Academy of Sciences

基于64点8bit输入20bit输出的FFT

撰写人：杨登天-202028015926089

目录

基于 64 点 8bit input-20bit output 的 FFT

杨登天-202028015926089-微电子研究所

一、任务设计要求

任务要求：采用硬件 RTL 代码实现 FFT 算法，可对连续输入串行数据以 64 点进行分组完成 FFT 运算。具体实现方法和输入输出接口不做硬性要求，可以自行定义。

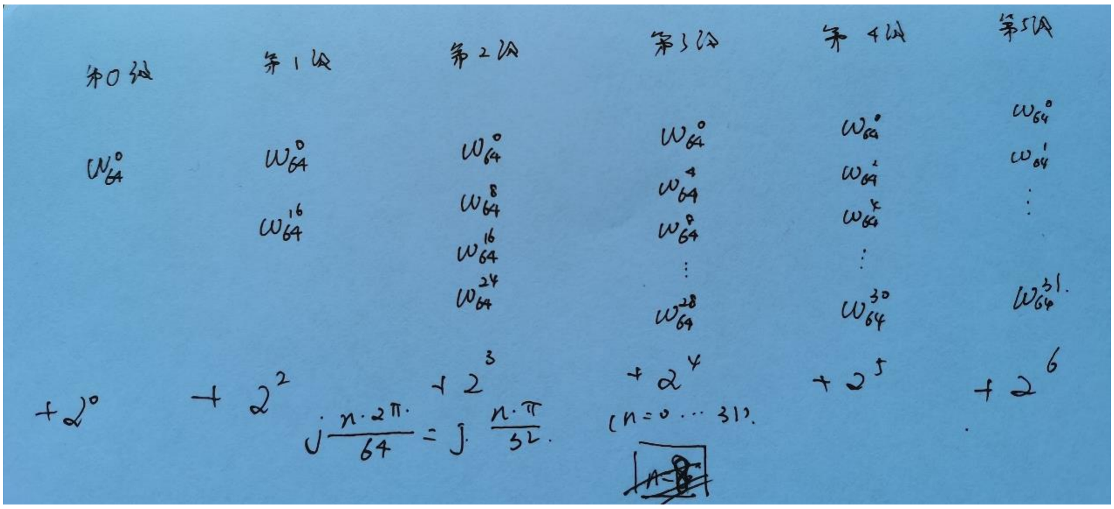
要求 1、FFT 输入 I/Q 数据为 8bits 有符号数，输出 I/Q 数据 20bits；

要求 2、数据存储模块用 Verilog 二维数组建模。

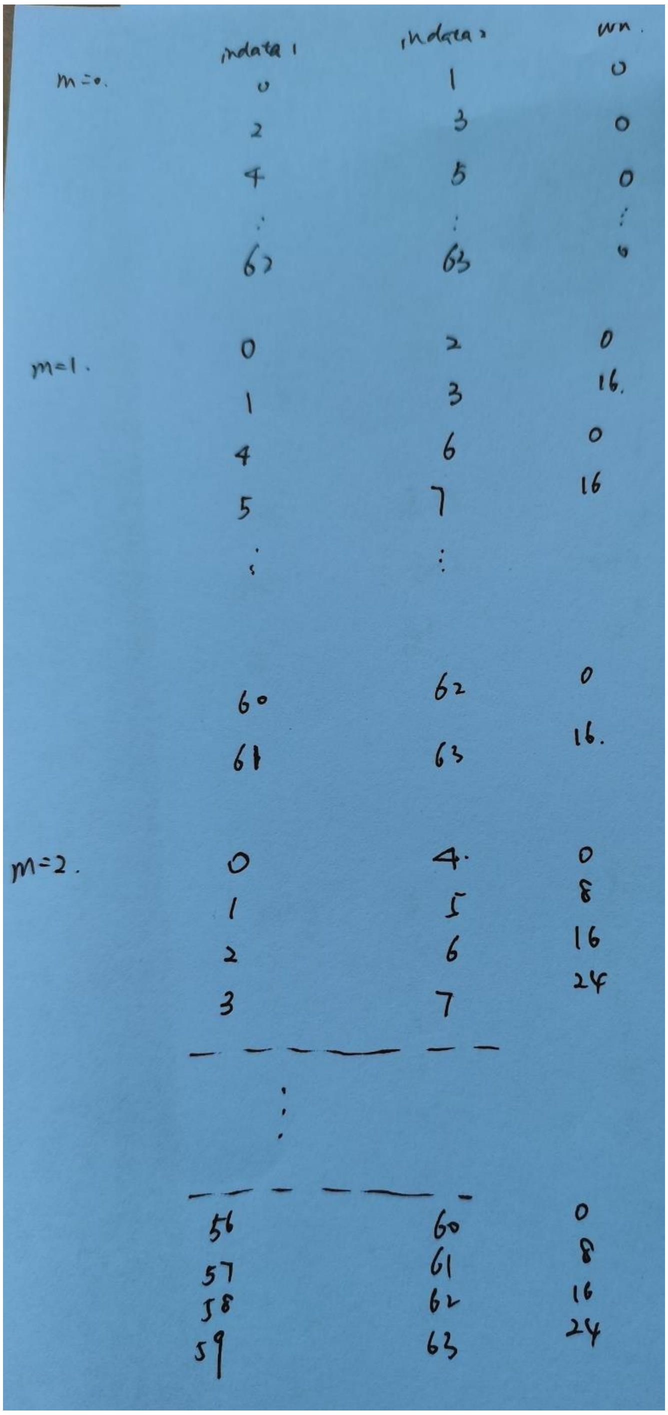
要求 3、64 点 FFT

二、解答步骤一——RTL 分析

2.1 步骤一、各级因子


第0级
第1级
第2级
第3级
第4级
第5级
W_{64}^{0}
W_{64}^{0} W_{64}^{16}
W_{64}^{0} W_{64}^{8} W_{64}^{16} W_{64}^{24}
W_{64}^{0} W_{64}^{4} W_{64}^{8} ⋮ W_{64}^{28}
W_{64}^{0} W_{64}^{2} W_{64}^{4} ⋮ W_{64}^{30}
W_{64}^{0} W_{64}^{1} ⋮ W_{64}^{31}.
$+2^{0}$
$+2^{2}$
$+2^{3}$
$+2^{4}$
$+2^{5}$
$+2^{6}$
$j\frac{n\cdot 2\pi}{64} = j\cdot\frac{n\cdot\pi}{32}$.
$(n=0\cdots 31)$.

2.2 步骤二、各级计算方式



	indata1	indata2	wn.
m=0.	0	1	0
	2	3	0
	4	5	0
	⋮	⋮	⋮
	62	63	0
m=1.	0	2	0
	1	3	16.
	4	6	0
	5	7	16
	⋮	⋮	
	60	62	0
	61	63	16.
m=2.	0	4.	0
	1	5	8
	2	6	16
	3	7	24
	⋮		
	56	60	0
	57	61	8
	58	62	16
	59	63	24

后续的 m=3、m=4、m=5 同理可得。

2.3 步骤三、输入二进制的逆置

详细见附录七的 matlab 生成倒置输入样本序号的 code

三、解答步骤二——设计与验证

3.1 步骤一、编辑方式和编辑工具

按照上述伪代码的方式用 notepad++编辑 Verilog HDL。

由于 Verilog 部分程序内容具有重复性，因而采用 MATLAB 生成。(见附录)

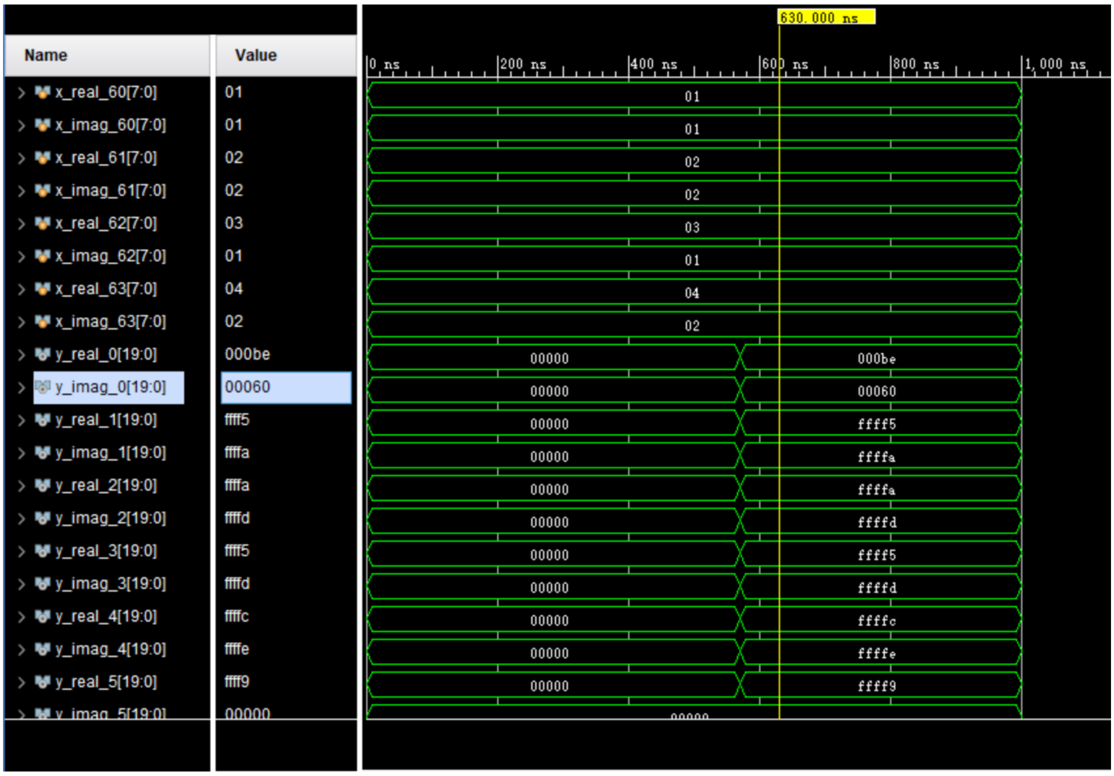
3.2 步骤二、验证方式和验证工具

验证方式采用 vivado 内置仿真工具分析。（design 和 test 程序见附录）

由于数据庞大，对比验证采用 MATLAB 软件验证 vivado 输出结果是否为真。（见附录）

四、解答步骤三——结果分析

4.1 步骤一、通过 vivado 内置仿真工具得到仿真图



Name	Value
x_real_60[7:0]	01
x_imag_60[7:0]	01
x_real_61[7:0]	02
x_imag_61[7:0]	02
x_real_62[7:0]	03
x_imag_62[7:0]	01
x_real_63[7:0]	04
x_imag_63[7:0]	02
y_real_0[19:0]	000be
y_imag_0[19:0]	00060
y_real_1[19:0]	ffff5
y_imag_1[19:0]	ffffa
y_real_2[19:0]	ffffa
y_imag_2[19:0]	ffffd
y_real_3[19:0]	ffff5
y_imag_3[19:0]	ffffd
y_real_4[19:0]	ffffc
y_imag_4[19:0]	ffffe
y_real_5[19:0]	ffff9
y_imag_5[19:0]	00000

图 1 仿真图结果

能够顺利得到结果，表明程序运行无问题。

4.2 步骤二、分析仿真图内的结果是否正确

首先可以根据图 1 的结果，y0 的实部为(000be)h，即为(190)d，其虚部为(00060)h，即为(96)d。y1 的实部为(ffff5)h，即为(-11)d，其虚部为(ffffa)h，即为(-6)d。y2 的实部为(ffffa)h，即为(-6)d，其虚部为(ffffd)h，即为(-2)d。

进一步观察附录二的表格，最终结果证明在一定误差范围内，结果正确。但误差总体呈现中间小、两边大的特点。中间部分如下，截取自附录二

点	Matlab 实部	Verilog 实部	Matlab 虚部	Verilog 虚部
12	-2	-2	-6.12000549666336	-8
13	-2.00000000000000	-2	-13.3166516116474	-14
14	-2.00000000000000	-2	50.2144971057246	51
15	-2.00000000000000	-2	7.63214375955181	8
16	-2.00000000000000	-5	3.61933302609203	4
17	-2	-2	2	2
18	-2.00000000000000	-3	1.04526874969524	0
19	-2.00000000000000	-5	0.348027313604026	-1
20	-2	-2	-0.24800814871235	-1
21	-2.00000000000000	-2	-0.83137023740879	-1

22	-2.00000000000000	-4	-1.48210040656926	-2
23	-2.00000000000000	-3	-2.31895254504705	-3
24	-2	-6	-3.60734217166246	-2
25	-2	-2	-6.24264068711929	-7
26	-2	-3	-17.0325338741947	-18
27	-2	-2	25.2312526272185	25
28	-2	-2	6.86178845113054	4
29	-2.00000000000000	-2	3.64205822297390	4
30	-2.00000000000000	-2	2.19169992877870	3
31	-2.00000000000000	-2	1.28703213819840	1
32	-2	-4	0.601653975593530	-2
33	-2	-2	-32	-32
34	-2	-2	-0.60165397559353	-1
35	-2	-3	-1.28703213819840	-2
36	-2.00000000000000	0	-2.19169992877869	-4
37	-2.00000000000000	-3	-3.64205822297389	-5
38	-2.00000000000000	-4	-6.86178845113054	-7
39	-2.00000000000000	-2	-25.2312526272185	-26
40	-2	0	17.0325338741947	16
41	-2	-2	6.24264068711929	6
42	-2.00000000000000	-4	3.60734217166246	2
43	-2.00000000000000	-2	2.31895254504705	0
44	-2.00000000000000	-3	1.48210040656926	-1
45	-2.00000000000000	-2	0.831370237408794	-1
46	-2.00000000000000	-3	0.248008148712346	-2
47	-2	-3	-0.34802731360402	-1
48	-2.00000000000000	-2	-1.04526874969524	-3
49	-2	-2	-2	-2
50	-2.00000000000000	-4	-3.61933302609203	-5
51	-2.00000000000000	-2	-7.63214375955181	-10
52	-2.00000000000000	-2	-50.2144971057246	-52
53	-2.00000000000000	-2	13.3166516116474	12

五、做题过程

整个程序花了不少时间，主要用 matlab 生成 Verilog 脚本，结构过于五花八门。其次因为参考了附录一的网址代码，其中关于 generate…for…块的编写，附录一的模块调用中的因子是错误的，因为我本来也抱着侥幸心理写这次作业，但是后来发现如果直接照搬附录一网址的程序（相应的位数做依次修改），得到的结果是有问题的。如下为附录一 generate 块程序

```
--------------------------------------------------------------------------------
genvar                    m, k;
    generate
    //3 stage
    for(m=0; m<=2; m=m+1) begin: stage
        for (k=0; k<=3; k=k+1) begin: unit

            butterfly                   u_butter(
                .clk             (clk                        ) ,
                .rstn            (rstn                       ) ,
                .en               (en_connect[m*4 + k] ) ,
                        //是否再组内？组编号+组内编号：下组编号+新组内编号
                .xp_real       (xm_real[ m ] [k[m:0] < (1<<m) ?
                            (k[3:m] << (m+1)) + k[m:0] :
                            (k[3:m] << (m+1)) + (k[m:0]-(1<<m))] ),
                .xp_imag       (xm_imag[ m ] [k[m:0] < (1<<m) ?
                            (k[3:m] << (m+1)) + k[m:0] :
                            (k[3:m] << (m+1)) + (k[m:0]-(1<<m))] ),
                .xq_real       (xm_real[ m ] [(k[m:0] < (1<<m) ?
                            (k[3:m] << (m+1)) + k[m:0] :
                            (k[3:m]   <<   (m+1))   +   (k[m:0]-(1<<m)))   +   (1<<m)   ]),
//增加蝶形单元两个输入端口间距离
                .xq_imag       (xm_imag[ m ] [(k[m:0] < (1<<m) ?
                            (k[3:m] << (m+1)) + k[m:0] :
                            (k[3:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ]),

                .factor_real(factor_real[k[m:0]<(1<<m)?
                             k[m:0] : k[m:0]-(1<<m) ]),
                .factor_imag(factor_imag[k[m:0]<(1<<m)?
                             k[m:0] : k[m:0]-(1<<m) ]),

                //output data
                .valid         (en_connect[ (m+1)*4 + k ]    ),
                .yp_real       (xm_real[ m+1 ][k[m:0] < (1<<m) ?
                            (k[3:m] << (m+1)) + k[m:0] :
                            (k[3:m] << (m+1)) + (k[m:0]-(1<<m))] ),
                .yp_imag       (xm_imag[ m+1 ][(k[m:0]) < (1<<m) ?
                            (k[3:m] << (m+1)) + k[m:0] :
                            (k[3:m] << (m+1)) + (k[m:0]-(1<<m))] ),
                .yq_real       (xm_real[ m+1 ][(k[m:0] < (1<<m) ?
                            (k[3:m] << (m+1)) + k[m:0] :
                            (k[3:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ]),
                .yq_imag       (xm_imag[ m+1 ][((k[m:0]) < (1<<m) ?
                            (k[3:m] << (m+1)) + k[m:0] :
                            (k[3:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ])
                );
            end
        end
    endgenerate
```

红色标记为错误内容，后来我想借用这个框架编写正确的因子块，尝试用

```
 wire [5:0] value [5:0] [31:0];
 genvar                      m, k;
    generate
    //3 stage
    for(m=0; m<=5; m=m+1) begin: stage
        for (k=0; k<=31; k=k+1) begin: unit

           assign value [m][k] = k[m:0] < (1<<m) ? (k[6:m] << (m+1)) + k[m:0] :
                                                  (k[6:m] << (m+1)) + (k[m:0]-(1<<m));


            butterfly         my(
               .clk             (clk                        ) ,
               .rstn            (rstn                       ) ,
                        //是否再组内？组编号+组内编号：下组编号+新组内编号
               .indata1_real       (x_real_all[ m ] [k[m:0] < (1<<m) ?
                            (k[6:m] << (m+1)) + k[m:0] :                    //原来是[3:m]
                            (k[6:m] << (m+1)) + (k[m:0]-(1<<m))] ),
               .indata1_imag       (x_imag_all[ m ] [k[m:0] < (1<<m) ?
                            (k[6:m] << (m+1)) + k[m:0] :
                            (k[6:m] << (m+1)) + (k[m:0]-(1<<m))] ),
               .indata2_real       (x_real_all[ m ] [(k[m:0] < (1<<m) ?
                            (k[6:m] << (m+1)) + k[m:0] :
                            (k[6:m]   <<   (m+1))   +   (k[m:0]-(1<<m)))   +   (1<<m)   ]),
//增加蝶形单元两个输入端口间距离
               .indata2_imag       (x_imag_all[ m ] [(k[m:0] < (1<<m) ?
                            (k[6:m] << (m+1)) + k[m:0] :
                            (k[6:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ]),

               .wn_real(fac_real[value[m][k][m-1:0] << (5-m)]),

               .wn_imag(fac_imag[value[m][k][m-1:0] << (5-m)]),

               .regular_signal        (signal_box[m*32 + k] ) ,
               //output data
               .oudata1_real       (x_real_all[ m+1 ][k[m:0] < (1<<m) ?
                            (k[6:m] << (m+1)) + k[m:0] :
                            (k[6:m] << (m+1)) + (k[m:0]-(1<<m))] ),
               .oudata1_imag       (x_imag_all[ m+1 ][(k[m:0]) < (1<<m) ?
                            (k[6:m] << (m+1)) + k[m:0] :
                            (k[6:m] << (m+1)) + (k[m:0]-(1<<m))] ),
               .oudata2_real       (x_real_all[ m+1 ][(k[m:0] < (1<<m) ?
                            (k[6:m] << (m+1)) + k[m:0] :
                            (k[6:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ]),
               .oudata2_imag       (x_imag_all[ m+1 ][((k[m:0]) < (1<<m) ?
                            (k[6:m] << (m+1)) + k[m:0] :
                            (k[6:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ]),
               .over_signal        (signal_box[ (m+1)*32 + k ]    )
               );
            end
        end
    endgenerate
------------------------------------------------------------------------------
```

找正确的规律耗费时间，理论上想法没有问题，我手算验证过三次都毫无问题，但是在程序运行时总是报错，后来通过网络查询相关知识也没能解决问题。于是，后来还是打算用最笨的方法去编辑，就是将 6 级计算全部写完，共 6×32=192 个模块调用。个人还是很遗憾generate…for…语块失败，打算之后再找找相关知识。

六、心得

这次作业难度不算太大，但就是繁琐，实现了以下要求

1、 8bit 输入、20bit 输出，因子精度设置为 13 位。
2、 蝶形模块实现 3 级流水线。
3、 顶层模块调用蝶形模块实现 6 阶计算。
4、 比对 matlab 直接调用 fft()和 Verilog 生成的结果，结果误差中间小，两边大。

没有做到的地方

1、 程序还是太繁琐，尝试用 generate…for…简化，但是问题比较大，这个已经在第五部分—做题过程中详细说明。
2、 因为误差还是比较明显，所以不太满意。

七、附录

一点说明：本次作业通过 matlab 生成脚本较多，以下仅贴出不完全的脚本生成代码，另外还有 Verilog 程序中的大量 assign 块、input、output 等等不贴出。

附录一、参考网址

https://www.runoob.com/w3cnote/verilog-fft.html

附录二、matlab64 点结果和 verilog64 点结果比对

点	Matlab 实部	Verilog 实部	Matlab 虚部	Verilog 虚部
1	190	190	96	96
2	-2.00000000000000	-11	-0.1971743530683	-6
3	-2.00000000000000	-6	-0.3994141643309	-3
4	-2.00000000000000	-11	-0.6124012633738	-3
5	-2.00000000000000	-4	-0.8432231512646	-2

6	-2.00000000000000	-7	-1.1015975885987	0
7	-2.00000000000000	-5	-1.4020505960762	-2
8	-2.00000000000000	-5	-1.76824635509111	-1
9	-2	-2	-2.24264068711929	-2
10	-2.00000000000000	-7	-2.91128754445441	-5
11	-2.00000000000000	-5	-3.98169301986463	-3
12	-2	-2	-6.12000549666336	-8
13	-2.00000000000000	-2	-13.3166516116474	-14
14	-2.00000000000000	-2	50.2144971057246	51
15	-2.00000000000000	-2	7.63214375955181	8
16	-2.00000000000000	-5	3.61933302609203	4
17	-2	-2	2	2
18	-2.00000000000000	-3	1.04526874969524	0
19	-2.00000000000000	-5	0.348027313604026	-1
20	-2	-2	-0.24800814871235	-1
21	-2.00000000000000	-2	-0.83137023740879	-1
22	-2.00000000000000	-4	-1.48210040656926	-2
23	-2.00000000000000	-3	-2.31895254504705	-3
24	-2	-6	-3.60734217166246	-2
25	-2	-2	-6.24264068711929	-7
26	-2	-3	-17.0325338741947	-18
27	-2	-2	25.2312526272185	25
28	-2	-2	6.86178845113054	4
29	-2.00000000000000	-2	3.64205822297390	4
30	-2.00000000000000	-2	2.19169992877870	3
31	-2.00000000000000	-2	1.28703213819840	1
32	-2	-4	0.601653975593530	-2
33	-2	-2	-32	-32
34	-2	-2	-0.60165397559353	-1
35	-2	-3	-1.28703213819840	-2
36	-2.00000000000000	0	-2.19169992877869	-4
37	-2.00000000000000	-3	-3.64205822297389	-5
38	-2.00000000000000	-4	-6.86178845113054	-7
39	-2.00000000000000	-2	-25.2312526272185	-26
40	-2	0	17.0325338741947	16
41	-2	-2	6.24264068711929	6
42	-2.00000000000000	-4	3.60734217166246	2
43	-2.00000000000000	-2	2.31895254504705	0
44	-2.00000000000000	-3	1.48210040656926	-1
45	-2.00000000000000	-2	0.831370237408794	-1
46	-2.00000000000000	-3	0.248008148712346	-2
47	-2	-3	-0.34802731360402	-1
48	-2.00000000000000	-2	-1.04526874969524	-3

49	-2	-2	-2	-2
50	-2.00000000000000	-4	-3.61933302609203	-5
51	-2.00000000000000	-2	-7.63214375955181	-10
52	-2.00000000000000	-2	-50.2144971057246	-52
53	-2.00000000000000	-2	13.3166516116474	12
54	-2.00000000000000	-1	6.12000549666336	1
55	-2.00000000000000	-1	3.98169301986463	1
56	-2	-1	2.91128754445441	-1
57	-2	-2	2.24264068711929	2
58	-2	-2	1.76824635509111	-3
59	-2	-3	1.40205059607625	-2
60	-2.00000000000000	0	1.10159758859873	-3
61	-2.00000000000000	-2	0.843223151264676	-1
62	-2.00000000000000	-4	0.612401263373801	-8
63	-2	-4	0.399414164330956	-4
64	-2.00000000000000	-5	0.197174353068384	-7

附录三、design code 的顶层模块

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2020/12/18 16:21:42
// Design Name:
// Module Name: fft64
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////
// 顶层例化
// 分析级数 64 点 FFT 计算
// input --.>--->o inner1 --.>---->o ...
// log2(64) = 6,总共需要 6 级 W，将其标记为 m=0、 1、 2、3、4、5
// 每一级都有若干的组，组数目记为 n，        n=32、16、8、4、2、1
module fft64(
input clk,
input rstn,
input input_signal,
input signed [7:0] x_real_0,    input signed [7:0] x_imag_0,
input signed [7:0] x_real_1,    input signed [7:0] x_imag_1,
input signed [7:0] x_real_2,    input signed [7:0] x_imag_2,
input signed [7:0] x_real_3,    input signed [7:0] x_imag_3,
input signed [7:0] x_real_4,    input signed [7:0] x_imag_4,
input signed [7:0] x_real_5,    input signed [7:0] x_imag_5,
input signed [7:0] x_real_6,    input signed [7:0] x_imag_6,
input signed [7:0] x_real_7,    input signed [7:0] x_imag_7,
input signed [7:0] x_real_8,    input signed [7:0] x_imag_8,
input signed [7:0] x_real_9,    input signed [7:0] x_imag_9,
input signed [7:0] x_real_10,    input signed [7:0] x_imag_10,
input signed [7:0] x_real_11,    input signed [7:0] x_imag_11,
input signed [7:0] x_real_12,    input signed [7:0] x_imag_12,
input signed [7:0] x_real_13,    input signed [7:0] x_imag_13,
input signed [7:0] x_real_14,    input signed [7:0] x_imag_14,
input signed [7:0] x_real_15,    input signed [7:0] x_imag_15,
input signed [7:0] x_real_16,    input signed [7:0] x_imag_16,
input signed [7:0] x_real_17,    input signed [7:0] x_imag_17,
input signed [7:0] x_real_18,    input signed [7:0] x_imag_18,
input signed [7:0] x_real_19,    input signed [7:0] x_imag_19,
input signed [7:0] x_real_20,    input signed [7:0] x_imag_20,
input signed [7:0] x_real_21,    input signed [7:0] x_imag_21,
input signed [7:0] x_real_22,    input signed [7:0] x_imag_22,
input signed [7:0] x_real_23,    input signed [7:0] x_imag_23,
input signed [7:0] x_real_24,    input signed [7:0] x_imag_24,
input signed [7:0] x_real_25,    input signed [7:0] x_imag_25,
input signed [7:0] x_real_26,    input signed [7:0] x_imag_26,
input signed [7:0] x_real_27,    input signed [7:0] x_imag_27,
input signed [7:0] x_real_28,    input signed [7:0] x_imag_28,
input signed [7:0] x_real_29,    input signed [7:0] x_imag_29,
input signed [7:0] x_real_30,    input signed [7:0] x_imag_30,
input signed [7:0] x_real_31,    input signed [7:0] x_imag_31,
input signed [7:0] x_real_32,    input signed [7:0] x_imag_32,
input signed [7:0] x_real_33,    input signed [7:0] x_imag_33,
input signed [7:0] x_real_34,    input signed [7:0] x_imag_34,
input signed [7:0] x_real_35,    input signed [7:0] x_imag_35,
input signed [7:0] x_real_36,    input signed [7:0] x_imag_36,
input signed [7:0] x_real_37,    input signed [7:0] x_imag_37,
input signed [7:0] x_real_38,    input signed [7:0] x_imag_38,
input signed [7:0] x_real_39,    input signed [7:0] x_imag_39,
input signed [7:0] x_real_40,    input signed [7:0] x_imag_40,
input signed [7:0] x_real_41,    input signed [7:0] x_imag_41,
input signed [7:0] x_real_42,    input signed [7:0] x_imag_42,
input signed [7:0] x_real_43,    input signed [7:0] x_imag_43,
input signed [7:0] x_real_44,    input signed [7:0] x_imag_44,
input signed [7:0] x_real_45,    input signed [7:0] x_imag_45,
input signed [7:0] x_real_46,    input signed [7:0] x_imag_46,
input signed [7:0] x_real_47,    input signed [7:0] x_imag_47,
input signed [7:0] x_real_48,    input signed [7:0] x_imag_48,
input signed [7:0] x_real_49,    input signed [7:0] x_imag_49,
input signed [7:0] x_real_50,    input signed [7:0] x_imag_50,
input signed [7:0] x_real_51,    input signed [7:0] x_imag_51,
input signed [7:0] x_real_52,    input signed [7:0] x_imag_52,
input signed [7:0] x_real_53,    input signed [7:0] x_imag_53,
input signed [7:0] x_real_54,    input signed [7:0] x_imag_54,
input signed [7:0] x_real_55,    input signed [7:0] x_imag_55,
input signed [7:0] x_real_56,    input signed [7:0] x_imag_56,
input signed [7:0] x_real_57,    input signed [7:0] x_imag_57,
input signed [7:0] x_real_58,    input signed [7:0] x_imag_58,
input signed [7:0] x_real_59,    input signed [7:0] x_imag_59,
input signed [7:0] x_real_60,    input signed [7:0] x_imag_60,
input signed [7:0] x_real_61,    input signed [7:0] x_imag_61,
input signed [7:0] x_real_62,    input signed [7:0] x_imag_62,
input signed [7:0] x_real_63,    input signed [7:0] x_imag_63,

output signed [19:0] y_real_0,    output signed [19:0] y_imag_0,
output signed [19:0] y_real_1,    output signed [19:0] y_imag_1,
output signed [19:0] y_real_2,    output signed [19:0] y_imag_2,
output signed [19:0] y_real_3,    output signed [19:0] y_imag_3,
output signed [19:0] y_real_4,    output signed [19:0] y_imag_4,
output signed [19:0] y_real_5,    output signed [19:0] y_imag_5,
output signed [19:0] y_real_6,    output signed [19:0] y_imag_6,
output signed [19:0] y_real_7,    output signed [19:0] y_imag_7,
output signed [19:0] y_real_8,    output signed [19:0] y_imag_8,
output signed [19:0] y_real_9,    output signed [19:0] y_imag_9,
output signed [19:0] y_real_10,    output signed [19:0] y_imag_10,
output signed [19:0] y_real_11,    output signed [19:0] y_imag_11,
output signed [19:0] y_real_12,    output signed [19:0] y_imag_12,
output signed [19:0] y_real_13,    output signed [19:0] y_imag_13,
output signed [19:0] y_real_14,    output signed [19:0] y_imag_14,
output signed [19:0] y_real_15,    output signed [19:0] y_imag_15,
output signed [19:0] y_real_16,    output signed [19:0] y_imag_16,
output signed [19:0] y_real_17,    output signed [19:0] y_imag_17,
output signed [19:0] y_real_18,    output signed [19:0] y_imag_18,
output signed [19:0] y_real_19,    output signed [19:0] y_imag_19,
output signed [19:0] y_real_20,    output signed [19:0] y_imag_20,
output signed [19:0] y_real_21,    output signed [19:0] y_imag_21,
output signed [19:0] y_real_22,    output signed [19:0] y_imag_22,
output signed [19:0] y_real_23,    output signed [19:0] y_imag_23,
output signed [19:0] y_real_24,    output signed [19:0] y_imag_24,
output signed [19:0] y_real_25,    output signed [19:0] y_imag_25,
output signed [19:0] y_real_26,    output signed [19:0] y_imag_26,
output signed [19:0] y_real_27,    output signed [19:0] y_imag_27,
output signed [19:0] y_real_28,    output signed [19:0] y_imag_28,
output signed [19:0] y_real_29,    output signed [19:0] y_imag_29,
output signed [19:0] y_real_30,    output signed [19:0] y_imag_30,
output signed [19:0] y_real_31,    output signed [19:0] y_imag_31,
output signed [19:0] y_real_32,    output signed [19:0] y_imag_32,
output signed [19:0] y_real_33,    output signed [19:0] y_imag_33,
output signed [19:0] y_real_34,    output signed [19:0] y_imag_34,
output signed [19:0] y_real_35,    output signed [19:0] y_imag_35,
output signed [19:0] y_real_36,    output signed [19:0] y_imag_36,
output signed [19:0] y_real_37,    output signed [19:0] y_imag_37,
output signed [19:0] y_real_38,    output signed [19:0] y_imag_38,
output signed [19:0] y_real_39,    output signed [19:0] y_imag_39,
output signed [19:0] y_real_40,    output signed [19:0] y_imag_40,
output signed [19:0] y_real_41,    output signed [19:0] y_imag_41,
output signed [19:0] y_real_42,    output signed [19:0] y_imag_42,
output signed [19:0] y_real_43,    output signed [19:0] y_imag_43,
output signed [19:0] y_real_44,    output signed [19:0] y_imag_44,
output signed [19:0] y_real_45,    output signed [19:0] y_imag_45,
output signed [19:0] y_real_46,    output signed [19:0] y_imag_46,
output signed [19:0] y_real_47,    output signed [19:0] y_imag_47,
output signed [19:0] y_real_48,    output signed [19:0] y_imag_48,
output signed [19:0] y_real_49,    output signed [19:0] y_imag_49,
output signed [19:0] y_real_50,    output signed [19:0] y_imag_50,
output signed [19:0] y_real_51,    output signed [19:0] y_imag_51,
output signed [19:0] y_real_52,    output signed [19:0] y_imag_52,
output signed [19:0] y_real_53,    output signed [19:0] y_imag_53,
output signed [19:0] y_real_54,    output signed [19:0] y_imag_54,
output signed [19:0] y_real_55,    output signed [19:0] y_imag_55,
output signed [19:0] y_real_56,    output signed [19:0] y_imag_56,
output signed [19:0] y_real_57,    output signed [19:0] y_imag_57,
output signed [19:0] y_real_58,    output signed [19:0] y_imag_58,
output signed [19:0] y_real_59,    output signed [19:0] y_imag_59,
output signed [19:0] y_real_60,    output signed [19:0] y_imag_60,
output signed [19:0] y_real_61,    output signed [19:0] y_imag_61,
output signed [19:0] y_real_62,    output signed [19:0] y_imag_62,
output signed [19:0] y_real_63,    output signed [19:0] y_imag_63,
 output output_signal
 );
 // 以下存放 6 级过程和最后输出结果，共 6:0 的 7 层
 wire signed [19:0] x_real_all [6:0][63:0];
 wire signed [19:0] x_imag_all [6:0][63:0];
 // 以下由 matlab 生成  符号位的扩展  //以下的写法是允许的，已在 butterfly_test 中尝
试
 assign x_real_all [ 0 ][ 0 ] = {{12{x_real_0 [ 7 ]}},x_real_0 };  assign x_imag_all [ 0 ][ 0 ] =
{{12{x_imag_0 [ 7 ]}},x_imag_0 };
 assign x_real_all [ 0 ][ 1 ] = {{12{x_real_32 [ 7 ]}},x_real_32 };  assign x_imag_all [ 0 ][ 1 ] =
{{12{x_imag_32 [ 7 ]}},x_imag_32 };
 assign x_real_all [ 0 ][ 2 ] = {{12{x_real_16 [ 7 ]}},x_real_16 };  assign x_imag_all [ 0 ][ 2 ] =
{{12{x_imag_16 [ 7 ]}},x_imag_16 };
 assign x_real_all [ 0 ][ 3 ] = {{12{x_real_48 [ 7 ]}},x_real_48 };  assign x_imag_all [ 0 ][ 3 ] =
{{12{x_imag_48 [ 7 ]}},x_imag_48 };
 assign x_real_all [ 0 ][ 4 ] = {{12{x_real_8 [ 7 ]}},x_real_8 };  assign x_imag_all [ 0 ][ 4 ] =
{{12{x_imag_8 [ 7 ]}},x_imag_8 };
 assign x_real_all [ 0 ][ 5 ] = {{12{x_real_40 [ 7 ]}},x_real_40 };  assign x_imag_all [ 0 ][ 5 ] =
{{12{x_imag_40 [ 7 ]}},x_imag_40 };
 assign x_real_all [ 0 ][ 6 ] = {{12{x_real_24 [ 7 ]}},x_real_24 };  assign x_imag_all [ 0 ][ 6 ] =
{{12{x_imag_24 [ 7 ]}},x_imag_24 };
 assign x_real_all [ 0 ][ 7 ] = {{12{x_real_56 [ 7 ]}},x_real_56 };  assign x_imag_all [ 0 ][ 7 ] =
{{12{x_imag_56 [ 7 ]}},x_imag_56 };
 assign x_real_all [ 0 ][ 8 ] = {{12{x_real_4 [ 7 ]}},x_real_4 };  assign x_imag_all [ 0 ][ 8 ] =
{{12{x_imag_4 [ 7 ]}},x_imag_4 };
 assign x_real_all [ 0 ][ 9 ] = {{12{x_real_36 [ 7 ]}},x_real_36 };  assign x_imag_all [ 0 ][ 9 ] =
{{12{x_imag_36 [ 7 ]}},x_imag_36 };
 assign x_real_all [ 0 ][ 10 ] = {{12{x_real_20 [ 7 ]}},x_real_20 };  assign x_imag_all [ 0 ][ 10 ] =
{{12{x_imag_20 [ 7 ]}},x_imag_20 };
 assign x_real_all [ 0 ][ 11 ] = {{12{x_real_52 [ 7 ]}},x_real_52 };  assign x_imag_all [ 0 ][ 11 ] =
{{12{x_imag_52 [ 7 ]}},x_imag_52 };
 assign x_real_all [ 0 ][ 12 ] = {{12{x_real_12 [ 7 ]}},x_real_12 };  assign x_imag_all [ 0 ][ 12 ] =
{{12{x_imag_12 [ 7 ]}},x_imag_12 };
 assign x_real_all [ 0 ][ 13 ] = {{12{x_real_44 [ 7 ]}},x_real_44 };  assign x_imag_all [ 0 ][ 13 ] =
{{12{x_imag_44 [ 7 ]}},x_imag_44 };
 assign x_real_all [ 0 ][ 14 ] = {{12{x_real_28 [ 7 ]}},x_real_28 };  assign x_imag_all [ 0 ][ 14 ] =
{{12{x_imag_28 [ 7 ]}},x_imag_28 };
 assign x_real_all [ 0 ][ 15 ] = {{12{x_real_60 [ 7 ]}},x_real_60 };  assign x_imag_all [ 0 ][ 15 ] =
{{12{x_imag_60 [ 7 ]}},x_imag_60 };
 assign x_real_all [ 0 ][ 16 ] = {{12{x_real_2 [ 7 ]}},x_real_2 };  assign x_imag_all [ 0 ][ 16 ] =
{{12{x_imag_2 [ 7 ]}},x_imag_2 };
 assign x_real_all [ 0 ][ 17 ] = {{12{x_real_34 [ 7 ]}},x_real_34 };  assign x_imag_all [ 0 ][ 17 ] =
{{12{x_imag_34 [ 7 ]}},x_imag_34 };
 assign x_real_all [ 0 ][ 18 ] = {{12{x_real_18 [ 7 ]}},x_real_18 };  assign x_imag_all [ 0 ][ 18 ] =
{{12{x_imag_18 [ 7 ]}},x_imag_18 };
 assign x_real_all [ 0 ][ 19 ] = {{12{x_real_50 [ 7 ]}},x_real_50 };     assign x_imag_all [ 0 ][ 19 ] =
{{12{x_imag_50 [ 7 ]}},x_imag_50 };
 assign x_real_all [ 0 ][ 20 ] = {{12{x_real_10 [ 7 ]}},x_real_10 };     assign x_imag_all [ 0 ][ 20 ] =
{{12{x_imag_10 [ 7 ]}},x_imag_10 };
 assign x_real_all [ 0 ][ 21 ] = {{12{x_real_42 [ 7 ]}},x_real_42 };     assign x_imag_all [ 0 ][ 21 ] =
{{12{x_imag_42 [ 7 ]}},x_imag_42 };
 assign x_real_all [ 0 ][ 22 ] = {{12{x_real_26 [ 7 ]}},x_real_26 };     assign x_imag_all [ 0 ][ 22 ] =
{{12{x_imag_26 [ 7 ]}},x_imag_26 };
 assign x_real_all [ 0 ][ 23 ] = {{12{x_real_58 [ 7 ]}},x_real_58 };     assign x_imag_all [ 0 ][ 23 ] =
{{12{x_imag_58 [ 7 ]}},x_imag_58 };
 assign x_real_all [ 0 ][ 24 ] = {{12{x_real_6 [ 7 ]}},x_real_6 };     assign x_imag_all [ 0 ][ 24 ] =
{{12{x_imag_6 [ 7 ]}},x_imag_6 };
 assign x_real_all [ 0 ][ 25 ] = {{12{x_real_38 [ 7 ]}},x_real_38 };     assign x_imag_all [ 0 ][ 25 ] =
{{12{x_imag_38 [ 7 ]}},x_imag_38 };
 assign x_real_all [ 0 ][ 26 ] = {{12{x_real_22 [ 7 ]}},x_real_22 };     assign x_imag_all [ 0 ][ 26 ] =
{{12{x_imag_22 [ 7 ]}},x_imag_22 };
 assign x_real_all [ 0 ][ 27 ] = {{12{x_real_54 [ 7 ]}},x_real_54 };     assign x_imag_all [ 0 ][ 27 ] =
{{12{x_imag_54 [ 7 ]}},x_imag_54 };
 assign x_real_all [ 0 ][ 28 ] = {{12{x_real_14 [ 7 ]}},x_real_14 };     assign x_imag_all [ 0 ][ 28 ] =
{{12{x_imag_14 [ 7 ]}},x_imag_14 };
 assign x_real_all [ 0 ][ 29 ] = {{12{x_real_46 [ 7 ]}},x_real_46 };     assign x_imag_all [ 0 ][ 29 ] =
{{12{x_imag_46 [ 7 ]}},x_imag_46 };
 assign x_real_all [ 0 ][ 30 ] = {{12{x_real_30 [ 7 ]}},x_real_30 };     assign x_imag_all [ 0 ][ 30 ] =
{{12{x_imag_30 [ 7 ]}},x_imag_30 };
 assign x_real_all [ 0 ][ 31 ] = {{12{x_real_62 [ 7 ]}},x_real_62 };     assign x_imag_all [ 0 ][ 31 ] =
{{12{x_imag_62 [ 7 ]}},x_imag_62 };
 assign x_real_all [ 0 ][ 32 ] = {{12{x_real_1 [ 7 ]}},x_real_1 };     assign x_imag_all [ 0 ][ 32 ] =
{{12{x_imag_1 [ 7 ]}},x_imag_1 };
 assign x_real_all [ 0 ][ 33 ] = {{12{x_real_33 [ 7 ]}},x_real_33 };     assign x_imag_all [ 0 ][ 33 ] =
{{12{x_imag_33 [ 7 ]}},x_imag_33 };
 assign x_real_all [ 0 ][ 34 ] = {{12{x_real_17 [ 7 ]}},x_real_17 };     assign x_imag_all [ 0 ][ 34 ] =
{{12{x_imag_17 [ 7 ]}},x_imag_17 };
 assign x_real_all [ 0 ][ 35 ] = {{12{x_real_49 [ 7 ]}},x_real_49 };     assign x_imag_all [ 0 ][ 35 ] =
{{12{x_imag_49 [ 7 ]}},x_imag_49 };
 assign x_real_all [ 0 ][ 36 ] = {{12{x_real_9 [ 7 ]}},x_real_9 };     assign x_imag_all [ 0 ][ 36 ] =
{{12{x_imag_9 [ 7 ]}},x_imag_9 };
 assign x_real_all [ 0 ][ 37 ] = {{12{x_real_41 [ 7 ]}},x_real_41 };     assign x_imag_all [ 0 ][ 37 ] =
{{12{x_imag_41 [ 7 ]}},x_imag_41 };
 assign x_real_all [ 0 ][ 38 ] = {{12{x_real_25 [ 7 ]}},x_real_25 };     assign x_imag_all [ 0 ][ 38 ] =
{{12{x_imag_25 [ 7 ]}},x_imag_25 };
 assign x_real_all [ 0 ][ 39 ] = {{12{x_real_57 [ 7 ]}},x_real_57 };     assign x_imag_all [ 0 ][ 39 ] =
{{12{x_imag_57 [ 7 ]}},x_imag_57 };
 assign x_real_all [ 0 ][ 40 ] = {{12{x_real_5 [ 7 ]}},x_real_5 };     assign x_imag_all [ 0 ][ 40 ] =
{{12{x_imag_5 [ 7 ]}},x_imag_5 };
 assign x_real_all [ 0 ][ 41 ] = {{12{x_real_37 [ 7 ]}},x_real_37 };     assign x_imag_all [ 0 ][ 41 ] =
{{12{x_imag_37 [ 7 ]}},x_imag_37 };
 assign x_real_all [ 0 ][ 42 ] = {{12{x_real_21 [ 7 ]}},x_real_21 };     assign x_imag_all [ 0 ][ 42 ] =
{{12{x_imag_21 [ 7 ]}},x_imag_21 };
 assign x_real_all [ 0 ][ 43 ] = {{12{x_real_53 [ 7 ]}},x_real_53 };     assign x_imag_all [ 0 ][ 43 ] =
{{12{x_imag_53 [ 7 ]}},x_imag_53 };
 assign x_real_all [ 0 ][ 44 ] = {{12{x_real_13 [ 7 ]}},x_real_13 };     assign x_imag_all [ 0 ][ 44 ] =
{{12{x_imag_13 [ 7 ]}},x_imag_13 };
 assign x_real_all [ 0 ][ 45 ] = {{12{x_real_45 [ 7 ]}},x_real_45 };     assign x_imag_all [ 0 ][ 45 ] =
{{12{x_imag_45 [ 7 ]}},x_imag_45 };
 assign x_real_all [ 0 ][ 46 ] = {{12{x_real_29 [ 7 ]}},x_real_29 };     assign x_imag_all [ 0 ][ 46 ] =
{{12{x_imag_29 [ 7 ]}},x_imag_29 };
 assign x_real_all [ 0 ][ 47 ] = {{12{x_real_61 [ 7 ]}},x_real_61 };     assign x_imag_all [ 0 ][ 47 ] =
{{12{x_imag_61 [ 7 ]}},x_imag_61 };
 assign x_real_all [ 0 ][ 48 ] = {{12{x_real_3 [ 7 ]}},x_real_3 };     assign x_imag_all [ 0 ][ 48 ] =
{{12{x_imag_3 [ 7 ]}},x_imag_3 };
 assign x_real_all [ 0 ][ 49 ] = {{12{x_real_35 [ 7 ]}},x_real_35 };     assign x_imag_all [ 0 ][ 49 ] =
{{12{x_imag_35 [ 7 ]}},x_imag_35 };
 assign x_real_all [ 0 ][ 50 ] = {{12{x_real_19 [ 7 ]}},x_real_19 };     assign x_imag_all [ 0 ][ 50 ] =
{{12{x_imag_19 [ 7 ]}},x_imag_19 };
 assign x_real_all [ 0 ][ 51 ] = {{12{x_real_51 [ 7 ]}},x_real_51 };     assign x_imag_all [ 0 ][ 51 ] =
{{12{x_imag_51 [ 7 ]}},x_imag_51 };
 assign x_real_all [ 0 ][ 52 ] = {{12{x_real_11 [ 7 ]}},x_real_11 };     assign x_imag_all [ 0 ][ 52 ] =
{{12{x_imag_11 [ 7 ]}},x_imag_11 };
 assign x_real_all [ 0 ][ 53 ] = {{12{x_real_43 [ 7 ]}},x_real_43 };     assign x_imag_all [ 0 ][ 53 ] =
{{12{x_imag_43 [ 7 ]}},x_imag_43 };
 assign x_real_all [ 0 ][ 54 ] = {{12{x_real_27 [ 7 ]}},x_real_27 };     assign x_imag_all [ 0 ][ 54 ] =
{{12{x_imag_27 [ 7 ]}},x_imag_27 };
 assign x_real_all [ 0 ][ 55 ] = {{12{x_real_59 [ 7 ]}},x_real_59 };     assign x_imag_all [ 0 ][ 55 ] =
{{12{x_imag_59 [ 7 ]}},x_imag_59 };
 assign x_real_all [ 0 ][ 56 ] = {{12{x_real_7 [ 7 ]}},x_real_7 };     assign x_imag_all [ 0 ][ 56 ] =
{{12{x_imag_7 [ 7 ]}},x_imag_7 };
 assign x_real_all [ 0 ][ 57 ] = {{12{x_real_39 [ 7 ]}},x_real_39 };     assign x_imag_all [ 0 ][ 57 ] =
{{12{x_imag_39 [ 7 ]}},x_imag_39 };
 assign x_real_all [ 0 ][ 58 ] = {{12{x_real_23 [ 7 ]}},x_real_23 };     assign x_imag_all [ 0 ][ 58 ] =
{{12{x_imag_23 [ 7 ]}},x_imag_23 };
 assign x_real_all [ 0 ][ 59 ] = {{12{x_real_55 [ 7 ]}},x_real_55 };     assign x_imag_all [ 0 ][ 59 ] =
{{12{x_imag_55 [ 7 ]}},x_imag_55 };
 assign x_real_all [ 0 ][ 60 ] = {{12{x_real_15 [ 7 ]}},x_real_15 };     assign x_imag_all [ 0 ][ 60 ] =
{{12{x_imag_15 [ 7 ]}},x_imag_15 };
 assign x_real_all [ 0 ][ 61 ] = {{12{x_real_47 [ 7 ]}},x_real_47 };     assign x_imag_all [ 0 ][ 61 ] =
{{12{x_imag_47 [ 7 ]}},x_imag_47 };
 assign x_real_all [ 0 ][ 62 ] = {{12{x_real_31 [ 7 ]}},x_real_31 };     assign x_imag_all [ 0 ][ 62 ] =
{{12{x_imag_31 [ 7 ]}},x_imag_31 };
 assign x_real_all [ 0 ][ 63 ] = {{12{x_real_63 [ 7 ]}},x_real_63 };   assign x_imag_all [ 0 ][ 63 ] =
{{12{x_imag_63 [ 7 ]}},x_imag_63 };
 // 以下存放因子
 wire signed [15:0] fac_real [31:0];
 wire signed [15:0] fac_imag [31:0];
 // 以下由 matlab 生成
 assign fac_real[ 0 ]=16'h2000 ; assign fac_imag[ 0 ]=16'h0000 ;
 assign fac_real[ 1 ]=16'h1FD8 ; assign fac_imag[ 1 ]=16'hFCDD ;
 assign fac_real[ 2 ]=16'h1F62 ; assign fac_imag[ 2 ]=16'hF9C1 ;
 assign fac_real[ 3 ]=16'h1E9F ; assign fac_imag[ 3 ]=16'hF6B5 ;
 assign fac_real[ 4 ]=16'h1D90 ; assign fac_imag[ 4 ]=16'hF3C1 ;
 assign fac_real[ 5 ]=16'h1C38 ; assign fac_imag[ 5 ]=16'hF0EA ;
 assign fac_real[ 6 ]=16'h1A9B ; assign fac_imag[ 6 ]=16'hEE38 ;
 assign fac_real[ 7 ]=16'h18BC ; assign fac_imag[ 7 ]=16'hEBB3 ;
 assign fac_real[ 8 ]=16'h16A0 ; assign fac_imag[ 8 ]=16'hE95F ;
 assign fac_real[ 9 ]=16'h144C ; assign fac_imag[ 9 ]=16'hE743 ;
 assign fac_real[ 10 ]=16'h11C7 ; assign fac_imag[ 10 ]=16'hE564 ;
 assign fac_real[ 11 ]=16'h0F15 ; assign fac_imag[ 11 ]=16'hE3C7 ;
 assign fac_real[ 12 ]=16'h0C3E ; assign fac_imag[ 12 ]=16'hE26F ;
 assign fac_real[ 13 ]=16'h094A ; assign fac_imag[ 13 ]=16'hE160 ;
 assign fac_real[ 14 ]=16'h063E ; assign fac_imag[ 14 ]=16'hE09D ;
 assign fac_real[ 15 ]=16'h0322 ; assign fac_imag[ 15 ]=16'hE027 ;
 assign fac_real[ 16 ]=16'h0000 ; assign fac_imag[ 16 ]=16'hE000 ;
 assign fac_real[ 17 ]=16'hFCDD ; assign fac_imag[ 17 ]=16'hE027 ;
 assign fac_real[ 18 ]=16'hF9C1 ; assign fac_imag[ 18 ]=16'hE09D ;
 assign fac_real[ 19 ]=16'hF6B5 ; assign fac_imag[ 19 ]=16'hE160 ;
 assign fac_real[ 20 ]=16'hF3C1 ; assign fac_imag[ 20 ]=16'hE26F ;
 assign fac_real[ 21 ]=16'hF0EA ; assign fac_imag[ 21 ]=16'hE3C7 ;
 assign fac_real[ 22 ]=16'hEE38 ; assign fac_imag[ 22 ]=16'hE564 ;
 assign fac_real[ 23 ]=16'hEBB3 ; assign fac_imag[ 23 ]=16'hE743 ;
 assign fac_real[ 24 ]=16'hE95F ; assign fac_imag[ 24 ]=16'hE95F ;
 assign fac_real[ 25 ]=16'hE743 ; assign fac_imag[ 25 ]=16'hEBB3 ;
 assign fac_real[ 26 ]=16'hE564 ; assign fac_imag[ 26 ]=16'hEE38 ;
 assign fac_real[ 27 ]=16'hE3C7 ; assign fac_imag[ 27 ]=16'hF0EA ;
 assign fac_real[ 28 ]=16'hE26F ; assign fac_imag[ 28 ]=16'hF3C1 ;
 assign fac_real[ 29 ]=16'hE160 ; assign fac_imag[ 29 ]=16'hF6B5 ;
 assign fac_real[ 30 ]=16'hE09D ; assign fac_imag[ 30 ]=16'hF9C1 ;
 assign fac_real[ 31 ]=16'hE027 ; assign fac_imag[ 31 ]=16'hFCDD ;

 // 流水部分
 // 引用模块为: butterfly
my(clk,rstn,indata1_real,indata1_imag,indata2_real,indata2_imag,wn_real,wn_imag,oudata1_r
eal,oudata1_imag,oudata2_real,oudata2_imag);
```

```
  // 第0级——邻近两两相互作用
// 补充说明各级次的r
// w=0时，按照顺序是w 2^1 0
// w=1时，按照顺序是w 2^2 0 1    (0 16)        /64
// w=2时，按照顺序是w 2^3 0 1 2 3    (0 8 16 24)
// w=3时，按照顺序是w 2^4 0 1 2 3 4 5 6 7 (0 4 8 12 16 20 24 28)
// w=4时，按照顺序是w 2^5 0 1...15 (0 2 4 6 ... 30)
// w=5时，按照顺序是w 2^6 0 1...31 (0 1 2 3 ... 31)
// 唯一的难点在于如何寻找规律
  wire     signal_box [223:0];    //预备前32个单元置input_signal 7*32=224
// assign signal_box [31:0] = {32{input_signal}};
// wire [223:0] signal_box;
//assign signal_box[31:0] = {32{input_signal}};

  assign signal_box[ 0 ] = input_signal;
  assign signal_box[ 1 ] = input_signal;
  assign signal_box[ 2 ] = input_signal;
  assign signal_box[ 3 ] = input_signal;
  assign signal_box[ 4 ] = input_signal;
  assign signal_box[ 5 ] = input_signal;
  assign signal_box[ 6 ] = input_signal;
  assign signal_box[ 7 ] = input_signal;
  assign signal_box[ 8 ] = input_signal;
  assign signal_box[ 9 ] = input_signal;
  assign signal_box[ 10 ] = input_signal;
  assign signal_box[ 11 ] = input_signal;
  assign signal_box[ 12 ] = input_signal;
  assign signal_box[ 13 ] = input_signal;
  assign signal_box[ 14 ] = input_signal;
  assign signal_box[ 15 ] = input_signal;
  assign signal_box[ 16 ] = input_signal;
  assign signal_box[ 17 ] = input_signal;
  assign signal_box[ 18 ] = input_signal;
  assign signal_box[ 19 ] = input_signal;
  assign signal_box[ 20 ] = input_signal;
  assign signal_box[ 21 ] = input_signal;
  assign signal_box[ 22 ] = input_signal;
  assign signal_box[ 23 ] = input_signal;
  assign signal_box[ 24 ] = input_signal;
  assign signal_box[ 25 ] = input_signal;
  assign signal_box[ 26 ] = input_signal;
  assign signal_box[ 27 ] = input_signal;
  assign signal_box[ 28 ] = input_signal;
  assign signal_box[ 29 ] = input_signal;
  assign signal_box[ 30 ] = input_signal;
  assign signal_box[ 31 ] = input_signal;
  //  关于级次和 regular_signal 信号的更新问题    2020/12/21
  // 0-31 32-63 64-95 96-127 128-159 160-191 192-223

// wire [5:0] value [5:0] [31:0];
/*
  genvar                              m, k;
        generate
        //3 stage
        for(m=0; m<=5; m=m+1) begin: stage              //  原来 m<=2
                for (k=0; k<=31; k=k+1) begin: unit        //  原来 k<=3


                  //wire [5:0] value;
              //    assign    value = k[m:0] < (1<<m) ?(k[6:m] << (m+1)) + k[m:0] :
              //                                                       (k[6:m] << (m+1)) + (k[m:0]-(1<<m));
              //       assign value [m][k] = k[m:0] < (1<<m) ? (k[6:m] << (m+1)) + k[m:0] :
              //                                                                              (k[6:m]  <<  (m+1))  +  (k[m:0]-(1<<m));



                      butterfly            my(
                            .clk                (clk                               ) ,
                            .rstn              (rstn                              ) ,
                                            //是否再组内？组编号+组内编号：下组编号+新组内编号
                            .indata1_real        (x_real_all[ m ] [k[m:0] < (1<<m) ?
                                                      (k[6:m] << (m+1)) + k[m:0] :                              //原来是[3:m]
                                                      (k[6:m] << (m+1)) + (k[m:0]-(1<<m))] ),
                            .indata1_imag        (x_imag_all[ m ] [k[m:0] < (1<<m) ?
                                                      (k[6:m] << (m+1)) + k[m:0] :
                                                      (k[6:m] << (m+1)) + (k[m:0]-(1<<m))] ),
                            .indata2_real        (x_real_all[ m ] [(k[m:0] < (1<<m) ?
                                                      (k[6:m] << (m+1)) + k[m:0] :
                                                      (k[6:m]    <<    (m+1))    +    (k[m:0]-(1<<m)))    +    (1<<m)    ]), //增加蝶形单元两个输入端口间距离
                            .indata2_imag        (x_imag_all[ m ] [(k[m:0] < (1<<m) ?
                                                      (k[6:m] << (m+1)) + k[m:0] :
                                                      (k[6:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ]),


                            //.wn_real(fac_real[k[m:0]<(1<<m)?
                            //                     k[m:0] : k[m:0]-(1<<m) ]),
                            //.wn_real(fac_real[(((k[m:0] < (1<<m) ?
                            //                                (k[6:m] << (m+1)) + k[m:0] :                           //原来是[3:m]
                            //                    (k[6:m] << (m+1)) + (k[m:0]-(1<<m)))[m-1:0]) << (5-m))])
                            .wn_real(fac_real[value[m][k][m-1:0] << (5-m)]),
                            .wn_imag(fac_imag[value[m][k][m-1:0] << (5-m)]),
                            //.wn_imag(fac_imag[k[m:0]<(1<<m)?
                            //              k[m:0] : k[m:0]-(1<<m) ]),
                            //.wn_imag(fac_imag[(((k[m:0] < (1<<m) ?
                            //                    (k[6:m] << (m+1)) + k[m:0] :                   //原来是[3:m]
                            //                    (k[6:m] << (m+1)) + (k[m:0]-(1<<m)))[m-1:0]) << (5-m))])

                            .regular_signal      (signal_box[m*32 + k] ) ,
                            //output data
                            .oudata1_real     (x_real_all[ m+1 ][k[m:0] < (1<<m) ?
                                        (k[6:m] << (m+1)) + k[m:0] :
                                        (k[6:m] << (m+1)) + (k[m:0]-(1<<m))] ),
                            .oudata1_imag     (x_imag_all[ m+1 ][(k[m:0]) < (1<<m) ?
                                        (k[6:m] << (m+1)) + k[m:0] :
                                        (k[6:m] << (m+1)) + (k[m:0]-(1<<m))] ),
                            .oudata2_real     (x_real_all[ m+1 ][(k[m:0] < (1<<m) ?
                                        (k[6:m] << (m+1)) + k[m:0] :
                                        (k[6:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ]),
                            .oudata2_imag     (x_imag_all[ m+1 ][((k[m:0]) < (1<<m) ?
                                        (k[6:m] << (m+1)) + k[m:0] :
                                        (k[6:m] << (m+1)) + (k[m:0]-(1<<m))) + (1<<m) ]),
                            .over_signal       (signal_box[ (m+1)*32 + k ]   )
                            );
                      end
                end
        endgenerate
*/
// 阶段 1
butterfly
my0_0(clk,rstn,x_real_all[0][0],x_imag_all[0][0],x_real_all[0][1],x_imag_all[0][1],fac_real[0],fac_imag[0],signal_box[0],x_real_all[1][0],x_imag_all[1][0],x_real_all[1][1],x_imag_all[1][1],signal_box[32+0]);
butterfly
my0_1(clk,rstn,x_real_all[0][2],x_imag_all[0][2],x_real_all[0][3],x_imag_all[0][3],fac_real[0],fac_imag[0],signal_box[1],x_real_all[1][2],x_imag_all[1][2],x_real_all[1][3],x_imag_all[1][3],signal_box[32+1]);
butterfly
my0_2(clk,rstn,x_real_all[0][4],x_imag_all[0][4],x_real_all[0][5],x_imag_all[0][5],fac_real[0],fac_imag[0],signal_box[2],x_real_all[1][4],x_imag_all[1][4],x_real_all[1][5],x_imag_all[1][5],signal_bo
x[32+2]);
butterfly
my0_3(clk,rstn,x_real_all[0][6],x_imag_all[0][6],x_real_all[0][7],x_imag_all[0][7],fac_real[0],fac_i
mag[0],signal_box[3],x_real_all[1][6],x_imag_all[1][6],x_real_all[1][7],x_imag_all[1][7],signal_bo
x[32+3]);
butterfly
my0_4(clk,rstn,x_real_all[0][8],x_imag_all[0][8],x_real_all[0][9],x_imag_all[0][9],fac_real[0],fac_i
mag[0],signal_box[4],x_real_all[1][8],x_imag_all[1][8],x_real_all[1][9],x_imag_all[1][9],signal_bo
x[32+4]);
butterfly
my0_5(clk,rstn,x_real_all[0][10],x_imag_all[0][10],x_real_all[0][11],x_imag_all[0][11],fac_real[0],f
ac_imag[0],signal_box[5],x_real_all[1][10],x_imag_all[1][10],x_real_all[1][11],x_imag_all[1][11],si
gnal_box[32+5]);
butterfly
my0_6(clk,rstn,x_real_all[0][12],x_imag_all[0][12],x_real_all[0][13],x_imag_all[0][13],fac_real[0],f
ac_imag[0],signal_box[6],x_real_all[1][12],x_imag_all[1][12],x_real_all[1][13],x_imag_all[1][13],si
gnal_box[32+6]);
butterfly
my0_7(clk,rstn,x_real_all[0][14],x_imag_all[0][14],x_real_all[0][15],x_imag_all[0][15],fac_real[0],f
ac_imag[0],signal_box[7],x_real_all[1][14],x_imag_all[1][14],x_real_all[1][15],x_imag_all[1][15],si
gnal_box[32+7]);
butterfly
my0_8(clk,rstn,x_real_all[0][16],x_imag_all[0][16],x_real_all[0][17],x_imag_all[0][17],fac_real[0],f
ac_imag[0],signal_box[8],x_real_all[1][16],x_imag_all[1][16],x_real_all[1][17],x_imag_all[1][17],si
gnal_box[32+8]);
butterfly
my0_9(clk,rstn,x_real_all[0][18],x_imag_all[0][18],x_real_all[0][19],x_imag_all[0][19],fac_real[0],f
ac_imag[0],signal_box[9],x_real_all[1][18],x_imag_all[1][18],x_real_all[1][19],x_imag_all[1][19],si
gnal_box[32+9]);
butterfly
my0_10(clk,rstn,x_real_all[0][20],x_imag_all[0][20],x_real_all[0][21],x_imag_all[0][21],fac_real[0],
fac_imag[0],signal_box[10],x_real_all[1][20],x_imag_all[1][20],x_real_all[1][21],x_imag_all[1][21]
,signal_box[32+10]);
butterfly
my0_11(clk,rstn,x_real_all[0][22],x_imag_all[0][22],x_real_all[0][23],x_imag_all[0][23],fac_real[0],
fac_imag[0],signal_box[11],x_real_all[1][22],x_imag_all[1][22],x_real_all[1][23],x_imag_all[1][23]
,signal_box[32+11]);
butterfly
my0_12(clk,rstn,x_real_all[0][24],x_imag_all[0][24],x_real_all[0][25],x_imag_all[0][25],fac_real[0],
fac_imag[0],signal_box[12],x_real_all[1][24],x_imag_all[1][24],x_real_all[1][25],x_imag_all[1][25]
,signal_box[32+12]);
butterfly
my0_13(clk,rstn,x_real_all[0][26],x_imag_all[0][26],x_real_all[0][27],x_imag_all[0][27],fac_real[0],
fac_imag[0],signal_box[13],x_real_all[1][26],x_imag_all[1][26],x_real_all[1][27],x_imag_all[1][27]
```

```
,signal_box[32+13]);
butterfly
my0_14(clk,rstn,x_real_all[0][28],x_imag_all[0][28],x_real_all[0][29],x_imag_all[0][29],fac_real[0],
fac_imag[0],signal_box[14],x_real_all[1][28],x_imag_all[1][28],x_real_all[1][29],x_imag_all[1][29]
,signal_box[32+14]);
butterfly
my0_15(clk,rstn,x_real_all[0][30],x_imag_all[0][30],x_real_all[0][31],x_imag_all[0][31],fac_real[0],
fac_imag[0],signal_box[15],x_real_all[1][30],x_imag_all[1][30],x_real_all[1][31],x_imag_all[1][31]
,signal_box[32+15]);
butterfly
my0_16(clk,rstn,x_real_all[0][32],x_imag_all[0][32],x_real_all[0][33],x_imag_all[0][33],fac_real[0],
fac_imag[0],signal_box[16],x_real_all[1][32],x_imag_all[1][32],x_real_all[1][33],x_imag_all[1][33]
,signal_box[32+16]);
butterfly
my0_17(clk,rstn,x_real_all[0][34],x_imag_all[0][34],x_real_all[0][35],x_imag_all[0][35],fac_real[0],
fac_imag[0],signal_box[17],x_real_all[1][34],x_imag_all[1][34],x_real_all[1][35],x_imag_all[1][35]
,signal_box[32+17]);
butterfly
my0_18(clk,rstn,x_real_all[0][36],x_imag_all[0][36],x_real_all[0][37],x_imag_all[0][37],fac_real[0],
fac_imag[0],signal_box[18],x_real_all[1][36],x_imag_all[1][36],x_real_all[1][37],x_imag_all[1][37]
,signal_box[32+18]);
butterfly
my0_19(clk,rstn,x_real_all[0][38],x_imag_all[0][38],x_real_all[0][39],x_imag_all[0][39],fac_real[0],
fac_imag[0],signal_box[19],x_real_all[1][38],x_imag_all[1][38],x_real_all[1][39],x_imag_all[1][39]
,signal_box[32+19]);
butterfly
my0_20(clk,rstn,x_real_all[0][40],x_imag_all[0][40],x_real_all[0][41],x_imag_all[0][41],fac_real[0],
fac_imag[0],signal_box[20],x_real_all[1][40],x_imag_all[1][40],x_real_all[1][41],x_imag_all[1][41]
,signal_box[32+20]);
butterfly
my0_21(clk,rstn,x_real_all[0][42],x_imag_all[0][42],x_real_all[0][43],x_imag_all[0][43],fac_real[0],
fac_imag[0],signal_box[21],x_real_all[1][42],x_imag_all[1][42],x_real_all[1][43],x_imag_all[1][43]
,signal_box[32+21]);
butterfly
my0_22(clk,rstn,x_real_all[0][44],x_imag_all[0][44],x_real_all[0][45],x_imag_all[0][45],fac_real[0],
fac_imag[0],signal_box[22],x_real_all[1][44],x_imag_all[1][44],x_real_all[1][45],x_imag_all[1][45]
,signal_box[32+22]);
butterfly
my0_23(clk,rstn,x_real_all[0][46],x_imag_all[0][46],x_real_all[0][47],x_imag_all[0][47],fac_real[0],
fac_imag[0],signal_box[23],x_real_all[1][46],x_imag_all[1][46],x_real_all[1][47],x_imag_all[1][47]
,signal_box[32+23]);
butterfly
my0_24(clk,rstn,x_real_all[0][48],x_imag_all[0][48],x_real_all[0][49],x_imag_all[0][49],fac_real[0],
fac_imag[0],signal_box[24],x_real_all[1][48],x_imag_all[1][48],x_real_all[1][49],x_imag_all[1][49]
```

```
,signal_box[32+24]);
butterfly
my0_25(clk,rstn,x_real_all[0][50],x_imag_all[0][50],x_real_all[0][51],x_imag_all[0][51],fac_real[0],
fac_imag[0],signal_box[25],x_real_all[1][50],x_imag_all[1][50],x_real_all[1][51],x_imag_all[1][51]
,signal_box[32+25]);
butterfly
my0_26(clk,rstn,x_real_all[0][52],x_imag_all[0][52],x_real_all[0][53],x_imag_all[0][53],fac_real[0],
fac_imag[0],signal_box[26],x_real_all[1][52],x_imag_all[1][52],x_real_all[1][53],x_imag_all[1][53]
,signal_box[32+26]);
butterfly
my0_27(clk,rstn,x_real_all[0][54],x_imag_all[0][54],x_real_all[0][55],x_imag_all[0][55],fac_real[0],
fac_imag[0],signal_box[27],x_real_all[1][54],x_imag_all[1][54],x_real_all[1][55],x_imag_all[1][55]
,signal_box[32+27]);
butterfly
my0_28(clk,rstn,x_real_all[0][56],x_imag_all[0][56],x_real_all[0][57],x_imag_all[0][57],fac_real[0],
fac_imag[0],signal_box[28],x_real_all[1][56],x_imag_all[1][56],x_real_all[1][57],x_imag_all[1][57]
,signal_box[32+28]);
butterfly
my0_29(clk,rstn,x_real_all[0][58],x_imag_all[0][58],x_real_all[0][59],x_imag_all[0][59],fac_real[0],
fac_imag[0],signal_box[29],x_real_all[1][58],x_imag_all[1][58],x_real_all[1][59],x_imag_all[1][59]
,signal_box[32+29]);
butterfly
my0_30(clk,rstn,x_real_all[0][60],x_imag_all[0][60],x_real_all[0][61],x_imag_all[0][61],fac_real[0],
fac_imag[0],signal_box[30],x_real_all[1][60],x_imag_all[1][60],x_real_all[1][61],x_imag_all[1][61]
,signal_box[32+30]);
butterfly
my0_31(clk,rstn,x_real_all[0][62],x_imag_all[0][62],x_real_all[0][63],x_imag_all[0][63],fac_real[0],
fac_imag[0],signal_box[31],x_real_all[1][62],x_imag_all[1][62],x_real_all[1][63],x_imag_all[1][63]
,signal_box[32+31]);

//阶段 2
butterfly
my1_0(clk,rstn,x_real_all[1][0],x_imag_all[1][0],x_real_all[1][2],x_imag_all[1][2],fac_real[0],fac_i
mag[0],signal_box[32+0],x_real_all[2][0],x_imag_all[2][0],x_real_all[2][2],x_imag_all[2][2],signal
_box[2*32+0]);
butterfly
my1_1(clk,rstn,x_real_all[1][1],x_imag_all[1][1],x_real_all[1][3],x_imag_all[1][3],fac_real[16],fac_i
mag[16],signal_box[32+1],x_real_all[2][1],x_imag_all[2][1],x_real_all[2][3],x_imag_all[2][3],sign
al_box[2*32+1]);
butterfly
my1_2(clk,rstn,x_real_all[1][4],x_imag_all[1][4],x_real_all[1][6],x_imag_all[1][6],fac_real[0],fac_i
mag[0],signal_box[32+2],x_real_all[2][4],x_imag_all[2][4],x_real_all[2][6],x_imag_all[2][6],signal
_box[2*32+2]);
butterfly
```

```
my1_3(clk,rstn,x_real_all[1][5],x_imag_all[1][5],x_real_all[1][7],x_imag_all[1][7],fac_real[16],fac_imag[16],signal_box[32+3],x_real_all[2][5],x_imag_all[2][5],x_real_all[2][7],x_imag_all[2][7],signal_box[2*32+3]);
butterfly
my1_4(clk,rstn,x_real_all[1][8],x_imag_all[1][8],x_real_all[1][10],x_imag_all[1][10],fac_real[0],fac_imag[0],signal_box[32+4],x_real_all[2][8],x_imag_all[2][8],x_real_all[2][10],x_imag_all[2][10],signal_box[2*32+4]);
butterfly
my1_5(clk,rstn,x_real_all[1][9],x_imag_all[1][9],x_real_all[1][11],x_imag_all[1][11],fac_real[16],fac_imag[16],signal_box[32+5],x_real_all[2][9],x_imag_all[2][9],x_real_all[2][11],x_imag_all[2][11],signal_box[2*32+5]);
butterfly
my1_6(clk,rstn,x_real_all[1][12],x_imag_all[1][12],x_real_all[1][14],x_imag_all[1][14],fac_real[0],fac_imag[0],signal_box[32+6],x_real_all[2][12],x_imag_all[2][12],x_real_all[2][14],x_imag_all[2][14],signal_box[2*32+6]);
butterfly
my1_7(clk,rstn,x_real_all[1][13],x_imag_all[1][13],x_real_all[1][15],x_imag_all[1][15],fac_real[16],fac_imag[16],signal_box[32+7],x_real_all[2][13],x_imag_all[2][13],x_real_all[2][15],x_imag_all[2][15],signal_box[2*32+7]);
butterfly
my1_8(clk,rstn,x_real_all[1][16],x_imag_all[1][16],x_real_all[1][18],x_imag_all[1][18],fac_real[0],fac_imag[0],signal_box[32+8],x_real_all[2][16],x_imag_all[2][16],x_real_all[2][18],x_imag_all[2][18],signal_box[2*32+8]);
butterfly
my1_9(clk,rstn,x_real_all[1][17],x_imag_all[1][17],x_real_all[1][19],x_imag_all[1][19],fac_real[16],fac_imag[16],signal_box[32+9],x_real_all[2][17],x_imag_all[2][17],x_real_all[2][19],x_imag_all[2][19],signal_box[2*32+9]);
butterfly
my1_10(clk,rstn,x_real_all[1][20],x_imag_all[1][20],x_real_all[1][22],x_imag_all[1][22],fac_real[0],fac_imag[0],signal_box[32+10],x_real_all[2][20],x_imag_all[2][20],x_real_all[2][22],x_imag_all[2][22],signal_box[2*32+10]);
butterfly
my1_11(clk,rstn,x_real_all[1][21],x_imag_all[1][21],x_real_all[1][23],x_imag_all[1][23],fac_real[16],fac_imag[16],signal_box[32+11],x_real_all[2][21],x_imag_all[2][21],x_real_all[2][23],x_imag_all[2][23],signal_box[2*32+11]);
butterfly
my1_12(clk,rstn,x_real_all[1][24],x_imag_all[1][24],x_real_all[1][26],x_imag_all[1][26],fac_real[0],fac_imag[0],signal_box[32+12],x_real_all[2][24],x_imag_all[2][24],x_real_all[2][26],x_imag_all[2][26],signal_box[2*32+12]);
butterfly
my1_13(clk,rstn,x_real_all[1][25],x_imag_all[1][25],x_real_all[1][27],x_imag_all[1][27],fac_real[16],fac_imag[16],signal_box[32+13],x_real_all[2][25],x_imag_all[2][25],x_real_all[2][27],x_imag_all[2][27],signal_box[2*32+13]);
butterfly
```

```
my1_14(clk,rstn,x_real_all[1][28],x_imag_all[1][28],x_real_all[1][30],x_imag_all[1][30],fac_real[0],
fac_imag[0],signal_box[32+14],x_real_all[2][28],x_imag_all[2][28],x_real_all[2][30],x_imag_all[2]
[30],signal_box[2*32+14]);
butterfly
my1_15(clk,rstn,x_real_all[1][29],x_imag_all[1][29],x_real_all[1][31],x_imag_all[1][31],fac_real[1
6],fac_imag[16],signal_box[32+15],x_real_all[2][29],x_imag_all[2][29],x_real_all[2][31],x_imag_a
ll[2][31],signal_box[2*32+15]);
butterfly
my1_16(clk,rstn,x_real_all[1][32],x_imag_all[1][32],x_real_all[1][34],x_imag_all[1][34],fac_real[0],
fac_imag[0],signal_box[32+16],x_real_all[2][32],x_imag_all[2][32],x_real_all[2][34],x_imag_all[2]
[34],signal_box[2*32+16]);
butterfly
my1_17(clk,rstn,x_real_all[1][33],x_imag_all[1][33],x_real_all[1][35],x_imag_all[1][35],fac_real[1
6],fac_imag[16],signal_box[32+17],x_real_all[2][33],x_imag_all[2][33],x_real_all[2][35],x_imag_a
ll[2][35],signal_box[2*32+17]);
butterfly
my1_18(clk,rstn,x_real_all[1][36],x_imag_all[1][36],x_real_all[1][38],x_imag_all[1][38],fac_real[0],
fac_imag[0],signal_box[32+18],x_real_all[2][36],x_imag_all[2][36],x_real_all[2][38],x_imag_all[2]
[38],signal_box[2*32+18]);
butterfly
my1_19(clk,rstn,x_real_all[1][37],x_imag_all[1][37],x_real_all[1][39],x_imag_all[1][39],fac_real[1
6],fac_imag[16],signal_box[32+19],x_real_all[2][37],x_imag_all[2][37],x_real_all[2][39],x_imag_a
ll[2][39],signal_box[2*32+19]);
butterfly
my1_20(clk,rstn,x_real_all[1][40],x_imag_all[1][40],x_real_all[1][42],x_imag_all[1][42],fac_real[0],
fac_imag[0],signal_box[32+20],x_real_all[2][40],x_imag_all[2][40],x_real_all[2][42],x_imag_all[2]
[42],signal_box[2*32+20]);
butterfly
my1_21(clk,rstn,x_real_all[1][41],x_imag_all[1][41],x_real_all[1][43],x_imag_all[1][43],fac_real[1
6],fac_imag[16],signal_box[32+21],x_real_all[2][41],x_imag_all[2][41],x_real_all[2][43],x_imag_a
ll[2][43],signal_box[2*32+21]);
butterfly
my1_22(clk,rstn,x_real_all[1][44],x_imag_all[1][44],x_real_all[1][46],x_imag_all[1][46],fac_real[0],
fac_imag[0],signal_box[32+22],x_real_all[2][44],x_imag_all[2][44],x_real_all[2][46],x_imag_all[2]
[46],signal_box[2*32+22]);
butterfly
my1_23(clk,rstn,x_real_all[1][45],x_imag_all[1][45],x_real_all[1][47],x_imag_all[1][47],fac_real[1
6],fac_imag[16],signal_box[32+23],x_real_all[2][45],x_imag_all[2][45],x_real_all[2][47],x_imag_a
ll[2][47],signal_box[2*32+23]);
butterfly
my1_24(clk,rstn,x_real_all[1][48],x_imag_all[1][48],x_real_all[1][50],x_imag_all[1][50],fac_real[0],
fac_imag[0],signal_box[32+24],x_real_all[2][48],x_imag_all[2][48],x_real_all[2][50],x_imag_all[2]
[50],signal_box[2*32+24]);
butterfly
```

```
my1_25(clk,rstn,x_real_all[1][49],x_imag_all[1][49],x_real_all[1][51],x_imag_all[1][51],fac_real[16],fac_imag[16],signal_box[32+25],x_real_all[2][49],x_imag_all[2][49],x_real_all[2][51],x_imag_all[2][51],signal_box[2*32+25]);
butterfly
my1_26(clk,rstn,x_real_all[1][52],x_imag_all[1][52],x_real_all[1][54],x_imag_all[1][54],fac_real[0],fac_imag[0],signal_box[32+26],x_real_all[2][52],x_imag_all[2][52],x_real_all[2][54],x_imag_all[2][54],signal_box[2*32+26]);
butterfly
my1_27(clk,rstn,x_real_all[1][53],x_imag_all[1][53],x_real_all[1][55],x_imag_all[1][55],fac_real[16],fac_imag[16],signal_box[32+27],x_real_all[2][53],x_imag_all[2][53],x_real_all[2][55],x_imag_all[2][55],signal_box[2*32+27]);
butterfly
my1_28(clk,rstn,x_real_all[1][56],x_imag_all[1][56],x_real_all[1][58],x_imag_all[1][58],fac_real[0],fac_imag[0],signal_box[32+28],x_real_all[2][56],x_imag_all[2][56],x_real_all[2][58],x_imag_all[2][58],signal_box[2*32+28]);
butterfly
my1_29(clk,rstn,x_real_all[1][57],x_imag_all[1][57],x_real_all[1][59],x_imag_all[1][59],fac_real[16],fac_imag[16],signal_box[32+29],x_real_all[2][57],x_imag_all[2][57],x_real_all[2][59],x_imag_all[2][59],signal_box[2*32+29]);
butterfly
my1_30(clk,rstn,x_real_all[1][60],x_imag_all[1][60],x_real_all[1][62],x_imag_all[1][62],fac_real[0],fac_imag[0],signal_box[32+30],x_real_all[2][60],x_imag_all[2][60],x_real_all[2][62],x_imag_all[2][62],signal_box[2*32+30]);
butterfly
my1_31(clk,rstn,x_real_all[1][61],x_imag_all[1][61],x_real_all[1][63],x_imag_all[1][63],fac_real[16],fac_imag[16],signal_box[32+31],x_real_all[2][61],x_imag_all[2][61],x_real_all[2][63],x_imag_all[2][63],signal_box[2*32+31]);

//阶段 3
butterfly
my2_0(clk,rstn,x_real_all[2][0],x_imag_all[2][0],x_real_all[2][4],x_imag_all[2][4],fac_real[0],fac_imag[0],signal_box[2*32+0],x_real_all[3][0],x_imag_all[3][0],x_real_all[3][4],x_imag_all[3][4],signal_box[3*32+0]);
butterfly
my2_1(clk,rstn,x_real_all[2][1],x_imag_all[2][1],x_real_all[2][5],x_imag_all[2][5],fac_real[8],fac_imag[8],signal_box[2*32+1],x_real_all[3][1],x_imag_all[3][1],x_real_all[3][5],x_imag_all[3][5],signal_box[3*32+1]);
butterfly
my2_2(clk,rstn,x_real_all[2][2],x_imag_all[2][2],x_real_all[2][6],x_imag_all[2][6],fac_real[16],fac_imag[16],signal_box[2*32+2],x_real_all[3][2],x_imag_all[3][2],x_real_all[3][6],x_imag_all[3][6],signal_box[3*32+2]);
butterfly
my2_3(clk,rstn,x_real_all[2][3],x_imag_all[2][3],x_real_all[2][7],x_imag_all[2][7],fac_real[24],fac_imag[24],signal_box[2*32+3],x_real_all[3][3],x_imag_all[3][3],x_real_all[3][7],x_imag_all[3][7],sig
nal_box[3*32+3]);
butterfly
my2_4(clk,rstn,x_real_all[2][8],x_imag_all[2][8],x_real_all[2][12],x_imag_all[2][12],fac_real[0],fac
_imag[0],signal_box[2*32+4],x_real_all[3][8],x_imag_all[3][8],x_real_all[3][12],x_imag_all[3][12],
signal_box[3*32+4]);
butterfly
my2_5(clk,rstn,x_real_all[2][9],x_imag_all[2][9],x_real_all[2][13],x_imag_all[2][13],fac_real[8],fac
_imag[8],signal_box[2*32+5],x_real_all[3][9],x_imag_all[3][9],x_real_all[3][13],x_imag_all[3][13],
signal_box[3*32+5]);
butterfly
my2_6(clk,rstn,x_real_all[2][10],x_imag_all[2][10],x_real_all[2][14],x_imag_all[2][14],fac_real[16],
fac_imag[16],signal_box[2*32+6],x_real_all[3][10],x_imag_all[3][10],x_real_all[3][14],x_imag_all
[3][14],signal_box[3*32+6]);
butterfly
my2_7(clk,rstn,x_real_all[2][11],x_imag_all[2][11],x_real_all[2][15],x_imag_all[2][15],fac_real[24],
fac_imag[24],signal_box[2*32+7],x_real_all[3][11],x_imag_all[3][11],x_real_all[3][15],x_imag_all
[3][15],signal_box[3*32+7]);
butterfly
my2_8(clk,rstn,x_real_all[2][16],x_imag_all[2][16],x_real_all[2][20],x_imag_all[2][20],fac_real[0],f
ac_imag[0],signal_box[2*32+8],x_real_all[3][16],x_imag_all[3][16],x_real_all[3][20],x_imag_all[3
][20],signal_box[3*32+8]);
butterfly
my2_9(clk,rstn,x_real_all[2][17],x_imag_all[2][17],x_real_all[2][21],x_imag_all[2][21],fac_real[8],f
ac_imag[8],signal_box[2*32+9],x_real_all[3][17],x_imag_all[3][17],x_real_all[3][21],x_imag_all[3
][21],signal_box[3*32+9]);
butterfly
my2_10(clk,rstn,x_real_all[2][18],x_imag_all[2][18],x_real_all[2][22],x_imag_all[2][22],fac_real[1
6],fac_imag[16],signal_box[2*32+10],x_real_all[3][18],x_imag_all[3][18],x_real_all[3][22],x_imag
_all[3][22],signal_box[3*32+10]);
butterfly
my2_11(clk,rstn,x_real_all[2][19],x_imag_all[2][19],x_real_all[2][23],x_imag_all[2][23],fac_real[2
4],fac_imag[24],signal_box[2*32+11],x_real_all[3][19],x_imag_all[3][19],x_real_all[3][23],x_imag
_all[3][23],signal_box[3*32+11]);
butterfly
my2_12(clk,rstn,x_real_all[2][24],x_imag_all[2][24],x_real_all[2][28],x_imag_all[2][28],fac_real[0],
fac_imag[0],signal_box[2*32+12],x_real_all[3][24],x_imag_all[3][24],x_real_all[3][28],x_imag_all
[3][28],signal_box[3*32+12]);
butterfly
my2_13(clk,rstn,x_real_all[2][25],x_imag_all[2][25],x_real_all[2][29],x_imag_all[2][29],fac_real[8],
fac_imag[8],signal_box[2*32+13],x_real_all[3][25],x_imag_all[3][25],x_real_all[3][29],x_imag_all
[3][29],signal_box[3*32+13]);
butterfly
my2_14(clk,rstn,x_real_all[2][26],x_imag_all[2][26],x_real_all[2][30],x_imag_all[2][30],fac_real[1
6],fac_imag[16],signal_box[2*32+14],x_real_all[3][26],x_imag_all[3][26],x_real_all[3][30],x_imag
```

```
_all[3][30],signal_box[3*32+14]);
butterfly
my2_15(clk,rstn,x_real_all[2][27],x_imag_all[2][27],x_real_all[2][31],x_imag_all[2][31],fac_real[2
4],fac_imag[24],signal_box[2*32+15],x_real_all[3][27],x_imag_all[3][27],x_real_all[3][31],x_imag
_all[3][31],signal_box[3*32+15]);
butterfly
my2_16(clk,rstn,x_real_all[2][32],x_imag_all[2][32],x_real_all[2][36],x_imag_all[2][36],fac_real[0],
fac_imag[0],signal_box[2*32+16],x_real_all[3][32],x_imag_all[3][32],x_real_all[3][36],x_imag_all
[3][36],signal_box[3*32+16]);
butterfly
my2_17(clk,rstn,x_real_all[2][33],x_imag_all[2][33],x_real_all[2][37],x_imag_all[2][37],fac_real[8],
fac_imag[8],signal_box[2*32+17],x_real_all[3][33],x_imag_all[3][33],x_real_all[3][37],x_imag_all
[3][37],signal_box[3*32+17]);
butterfly
my2_18(clk,rstn,x_real_all[2][34],x_imag_all[2][34],x_real_all[2][38],x_imag_all[2][38],fac_real[1
6],fac_imag[16],signal_box[2*32+18],x_real_all[3][34],x_imag_all[3][34],x_real_all[3][38],x_imag
_all[3][38],signal_box[3*32+18]);
butterfly
my2_19(clk,rstn,x_real_all[2][35],x_imag_all[2][35],x_real_all[2][39],x_imag_all[2][39],fac_real[2
4],fac_imag[24],signal_box[2*32+19],x_real_all[3][35],x_imag_all[3][35],x_real_all[3][39],x_imag
_all[3][39],signal_box[3*32+19]);
butterfly
my2_20(clk,rstn,x_real_all[2][40],x_imag_all[2][40],x_real_all[2][44],x_imag_all[2][44],fac_real[0],
fac_imag[0],signal_box[2*32+20],x_real_all[3][40],x_imag_all[3][40],x_real_all[3][44],x_imag_all
[3][44],signal_box[3*32+20]);
butterfly
my2_21(clk,rstn,x_real_all[2][41],x_imag_all[2][41],x_real_all[2][45],x_imag_all[2][45],fac_real[8],
fac_imag[8],signal_box[2*32+21],x_real_all[3][41],x_imag_all[3][41],x_real_all[3][45],x_imag_all
[3][45],signal_box[3*32+21]);
butterfly
my2_22(clk,rstn,x_real_all[2][42],x_imag_all[2][42],x_real_all[2][46],x_imag_all[2][46],fac_real[1
6],fac_imag[16],signal_box[2*32+22],x_real_all[3][42],x_imag_all[3][42],x_real_all[3][46],x_imag
_all[3][46],signal_box[3*32+22]);
butterfly
my2_23(clk,rstn,x_real_all[2][43],x_imag_all[2][43],x_real_all[2][47],x_imag_all[2][47],fac_real[2
4],fac_imag[24],signal_box[2*32+23],x_real_all[3][43],x_imag_all[3][43],x_real_all[3][47],x_imag
_all[3][47],signal_box[3*32+23]);
butterfly
my2_24(clk,rstn,x_real_all[2][48],x_imag_all[2][48],x_real_all[2][52],x_imag_all[2][52],fac_real[0],
fac_imag[0],signal_box[2*32+24],x_real_all[3][48],x_imag_all[3][48],x_real_all[3][52],x_imag_all
[3][52],signal_box[3*32+24]);
butterfly
my2_25(clk,rstn,x_real_all[2][49],x_imag_all[2][49],x_real_all[2][53],x_imag_all[2][53],fac_real[8],
fac_imag[8],signal_box[2*32+25],x_real_all[3][49],x_imag_all[3][49],x_real_all[3][53],x_imag_all
```

```
[3][53],signal_box[3*32+25]);
butterfly
my2_26(clk,rstn,x_real_all[2][50],x_imag_all[2][50],x_real_all[2][54],x_imag_all[2][54],fac_real[1
6],fac_imag[16],signal_box[2*32+26],x_real_all[3][50],x_imag_all[3][50],x_real_all[3][54],x_imag
_all[3][54],signal_box[3*32+26]);
butterfly
my2_27(clk,rstn,x_real_all[2][51],x_imag_all[2][51],x_real_all[2][55],x_imag_all[2][55],fac_real[2
4],fac_imag[24],signal_box[2*32+27],x_real_all[3][51],x_imag_all[3][51],x_real_all[3][55],x_imag
_all[3][55],signal_box[3*32+27]);
butterfly
my2_28(clk,rstn,x_real_all[2][56],x_imag_all[2][56],x_real_all[2][60],x_imag_all[2][60],fac_real[0],
fac_imag[0],signal_box[2*32+28],x_real_all[3][56],x_imag_all[3][56],x_real_all[3][60],x_imag_all
[3][60],signal_box[3*32+28]);
butterfly
my2_29(clk,rstn,x_real_all[2][57],x_imag_all[2][57],x_real_all[2][61],x_imag_all[2][61],fac_real[8],
fac_imag[8],signal_box[2*32+29],x_real_all[3][57],x_imag_all[3][57],x_real_all[3][61],x_imag_all
[3][61],signal_box[3*32+29]);
butterfly
my2_30(clk,rstn,x_real_all[2][58],x_imag_all[2][58],x_real_all[2][62],x_imag_all[2][62],fac_real[1
6],fac_imag[16],signal_box[2*32+30],x_real_all[3][58],x_imag_all[3][58],x_real_all[3][62],x_imag
_all[3][62],signal_box[3*32+30]);
butterfly
my2_31(clk,rstn,x_real_all[2][59],x_imag_all[2][59],x_real_all[2][63],x_imag_all[2][63],fac_real[2
4],fac_imag[24],signal_box[2*32+31],x_real_all[3][59],x_imag_all[3][59],x_real_all[3][63],x_imag
_all[3][63],signal_box[3*32+31]);

//阶段 4
butterfly
my3_0(clk,rstn,x_real_all[3][0],x_imag_all[3][0],x_real_all[3][8],x_imag_all[3][8],fac_real[0],fac_i
mag[0],signal_box[3*32+0],x_real_all[4][0],x_imag_all[4][0],x_real_all[4][8],x_imag_all[4][8],sign
al_box[4*32+0]);
butterfly
my3_1(clk,rstn,x_real_all[3][1],x_imag_all[3][1],x_real_all[3][9],x_imag_all[3][9],fac_real[4],fac_i
mag[4],signal_box[3*32+1],x_real_all[4][1],x_imag_all[4][1],x_real_all[4][9],x_imag_all[4][9],sign
al_box[4*32+1]);
butterfly
my3_2(clk,rstn,x_real_all[3][2],x_imag_all[3][2],x_real_all[3][10],x_imag_all[3][10],fac_real[8],fac
_imag[8],signal_box[3*32+2],x_real_all[4][2],x_imag_all[4][2],x_real_all[4][10],x_imag_all[4][10],
signal_box[4*32+2]);
butterfly
my3_3(clk,rstn,x_real_all[3][3],x_imag_all[3][3],x_real_all[3][11],x_imag_all[3][11],fac_real[12],fa
c_imag[12],signal_box[3*32+3],x_real_all[4][3],x_imag_all[4][3],x_real_all[4][11],x_imag_all[4][1
1],signal_box[4*32+3]);
butterfly
```

```
my3_4(clk,rstn,x_real_all[3][4],x_imag_all[3][4],x_real_all[3][12],x_imag_all[3][12],fac_real[16],fac_imag[16],signal_box[3*32+4],x_real_all[4][4],x_imag_all[4][4],x_real_all[4][12],x_imag_all[4][12],signal_box[4*32+4]);
butterfly
my3_5(clk,rstn,x_real_all[3][5],x_imag_all[3][5],x_real_all[3][13],x_imag_all[3][13],fac_real[20],fac_imag[20],signal_box[3*32+5],x_real_all[4][5],x_imag_all[4][5],x_real_all[4][13],x_imag_all[4][13],signal_box[4*32+5]);
butterfly
my3_6(clk,rstn,x_real_all[3][6],x_imag_all[3][6],x_real_all[3][14],x_imag_all[3][14],fac_real[24],fac_imag[24],signal_box[3*32+6],x_real_all[4][6],x_imag_all[4][6],x_real_all[4][14],x_imag_all[4][14],signal_box[4*32+6]);
butterfly
my3_7(clk,rstn,x_real_all[3][7],x_imag_all[3][7],x_real_all[3][15],x_imag_all[3][15],fac_real[28],fac_imag[28],signal_box[3*32+7],x_real_all[4][7],x_imag_all[4][7],x_real_all[4][15],x_imag_all[4][15],signal_box[4*32+7]);
butterfly
my3_8(clk,rstn,x_real_all[3][16],x_imag_all[3][16],x_real_all[3][24],x_imag_all[3][24],fac_real[0],fac_imag[0],signal_box[3*32+8],x_real_all[4][16],x_imag_all[4][16],x_real_all[4][24],x_imag_all[4][24],signal_box[4*32+8]);
butterfly
my3_9(clk,rstn,x_real_all[3][17],x_imag_all[3][17],x_real_all[3][25],x_imag_all[3][25],fac_real[4],fac_imag[4],signal_box[3*32+9],x_real_all[4][17],x_imag_all[4][17],x_real_all[4][25],x_imag_all[4][25],signal_box[4*32+9]);
butterfly
my3_10(clk,rstn,x_real_all[3][18],x_imag_all[3][18],x_real_all[3][26],x_imag_all[3][26],fac_real[8],fac_imag[8],signal_box[3*32+10],x_real_all[4][18],x_imag_all[4][18],x_real_all[4][26],x_imag_all[4][26],signal_box[4*32+10]);
butterfly
my3_11(clk,rstn,x_real_all[3][19],x_imag_all[3][19],x_real_all[3][27],x_imag_all[3][27],fac_real[12],fac_imag[12],signal_box[3*32+11],x_real_all[4][19],x_imag_all[4][19],x_real_all[4][27],x_imag_all[4][27],signal_box[4*32+11]);
butterfly
my3_12(clk,rstn,x_real_all[3][20],x_imag_all[3][20],x_real_all[3][28],x_imag_all[3][28],fac_real[16],fac_imag[16],signal_box[3*32+12],x_real_all[4][20],x_imag_all[4][20],x_real_all[4][28],x_imag_all[4][28],signal_box[4*32+12]);
butterfly
my3_13(clk,rstn,x_real_all[3][21],x_imag_all[3][21],x_real_all[3][29],x_imag_all[3][29],fac_real[20],fac_imag[20],signal_box[3*32+13],x_real_all[4][21],x_imag_all[4][21],x_real_all[4][29],x_imag_all[4][29],signal_box[4*32+13]);
butterfly
my3_14(clk,rstn,x_real_all[3][22],x_imag_all[3][22],x_real_all[3][30],x_imag_all[3][30],fac_real[24],fac_imag[24],signal_box[3*32+14],x_real_all[4][22],x_imag_all[4][22],x_real_all[4][30],x_imag_all[4][30],signal_box[4*32+14]);
butterfly
```

```
my3_15(clk,rstn,x_real_all[3][23],x_imag_all[3][23],x_real_all[3][31],x_imag_all[3][31],fac_real[28],fac_imag[28],signal_box[3*32+15],x_real_all[4][23],x_imag_all[4][23],x_real_all[4][31],x_imag_all[4][31],signal_box[4*32+15]);
butterfly
my3_16(clk,rstn,x_real_all[3][32],x_imag_all[3][32],x_real_all[3][40],x_imag_all[3][40],fac_real[0],fac_imag[0],signal_box[3*32+16],x_real_all[4][32],x_imag_all[4][32],x_real_all[4][40],x_imag_all[4][40],signal_box[4*32+16]);
butterfly
my3_17(clk,rstn,x_real_all[3][33],x_imag_all[3][33],x_real_all[3][41],x_imag_all[3][41],fac_real[4],fac_imag[4],signal_box[3*32+17],x_real_all[4][33],x_imag_all[4][33],x_real_all[4][41],x_imag_all[4][41],signal_box[4*32+17]);
butterfly
my3_18(clk,rstn,x_real_all[3][34],x_imag_all[3][34],x_real_all[3][42],x_imag_all[3][42],fac_real[8],fac_imag[8],signal_box[3*32+18],x_real_all[4][34],x_imag_all[4][34],x_real_all[4][42],x_imag_all[4][42],signal_box[4*32+18]);
butterfly
my3_19(clk,rstn,x_real_all[3][35],x_imag_all[3][35],x_real_all[3][43],x_imag_all[3][43],fac_real[12],fac_imag[12],signal_box[3*32+19],x_real_all[4][35],x_imag_all[4][35],x_real_all[4][43],x_imag_all[4][43],signal_box[4*32+19]);
butterfly
my3_20(clk,rstn,x_real_all[3][36],x_imag_all[3][36],x_real_all[3][44],x_imag_all[3][44],fac_real[16],fac_imag[16],signal_box[3*32+20],x_real_all[4][36],x_imag_all[4][36],x_real_all[4][44],x_imag_all[4][44],signal_box[4*32+20]);
butterfly
my3_21(clk,rstn,x_real_all[3][37],x_imag_all[3][37],x_real_all[3][45],x_imag_all[3][45],fac_real[20],fac_imag[20],signal_box[3*32+21],x_real_all[4][37],x_imag_all[4][37],x_real_all[4][45],x_imag_all[4][45],signal_box[4*32+21]);
butterfly
my3_22(clk,rstn,x_real_all[3][38],x_imag_all[3][38],x_real_all[3][46],x_imag_all[3][46],fac_real[24],fac_imag[24],signal_box[3*32+22],x_real_all[4][38],x_imag_all[4][38],x_real_all[4][46],x_imag_all[4][46],signal_box[4*32+22]);
butterfly
my3_23(clk,rstn,x_real_all[3][39],x_imag_all[3][39],x_real_all[3][47],x_imag_all[3][47],fac_real[28],fac_imag[28],signal_box[3*32+23],x_real_all[4][39],x_imag_all[4][39],x_real_all[4][47],x_imag_all[4][47],signal_box[4*32+23]);
butterfly
my3_24(clk,rstn,x_real_all[3][48],x_imag_all[3][48],x_real_all[3][56],x_imag_all[3][56],fac_real[0],fac_imag[0],signal_box[3*32+24],x_real_all[4][48],x_imag_all[4][48],x_real_all[4][56],x_imag_all[4][56],signal_box[4*32+24]);
butterfly
my3_25(clk,rstn,x_real_all[3][49],x_imag_all[3][49],x_real_all[3][57],x_imag_all[3][57],fac_real[4],fac_imag[4],signal_box[3*32+25],x_real_all[4][49],x_imag_all[4][49],x_real_all[4][57],x_imag_all[4][57],signal_box[4*32+25]);
butterfly
```

```
my3_26(clk,rstn,x_real_all[3][50],x_imag_all[3][50],x_real_all[3][58],x_imag_all[3][58],fac_real[8],
fac_imag[8],signal_box[3*32+26],x_real_all[4][50],x_imag_all[4][50],x_real_all[4][58],x_imag_all
[4][58],signal_box[4*32+26]);
butterfly
my3_27(clk,rstn,x_real_all[3][51],x_imag_all[3][51],x_real_all[3][59],x_imag_all[3][59],fac_real[1
2],fac_imag[12],signal_box[3*32+27],x_real_all[4][51],x_imag_all[4][51],x_real_all[4][59],x_imag
_all[4][59],signal_box[4*32+27]);
butterfly
my3_28(clk,rstn,x_real_all[3][52],x_imag_all[3][52],x_real_all[3][60],x_imag_all[3][60],fac_real[1
6],fac_imag[16],signal_box[3*32+28],x_real_all[4][52],x_imag_all[4][52],x_real_all[4][60],x_imag
_all[4][60],signal_box[4*32+28]);
butterfly
my3_29(clk,rstn,x_real_all[3][53],x_imag_all[3][53],x_real_all[3][61],x_imag_all[3][61],fac_real[2
0],fac_imag[20],signal_box[3*32+29],x_real_all[4][53],x_imag_all[4][53],x_real_all[4][61],x_imag
_all[4][61],signal_box[4*32+29]);
butterfly
my3_30(clk,rstn,x_real_all[3][54],x_imag_all[3][54],x_real_all[3][62],x_imag_all[3][62],fac_real[2
4],fac_imag[24],signal_box[3*32+30],x_real_all[4][54],x_imag_all[4][54],x_real_all[4][62],x_imag
_all[4][62],signal_box[4*32+30]);
butterfly
my3_31(clk,rstn,x_real_all[3][55],x_imag_all[3][55],x_real_all[3][63],x_imag_all[3][63],fac_real[2
8],fac_imag[28],signal_box[3*32+31],x_real_all[4][55],x_imag_all[4][55],x_real_all[4][63],x_imag
_all[4][63],signal_box[4*32+31]);

//阶段 5
butterfly
my4_0(clk,rstn,x_real_all[4][0],x_imag_all[4][0],x_real_all[4][16],x_imag_all[4][16],fac_real[0],fac
_imag[0],signal_box[4*32+0],x_real_all[5][0],x_imag_all[5][0],x_real_all[5][16],x_imag_all[5][16],
signal_box[5*32+0]);
butterfly
my4_1(clk,rstn,x_real_all[4][1],x_imag_all[4][1],x_real_all[4][17],x_imag_all[4][17],fac_real[2],fac
_imag[2],signal_box[4*32+1],x_real_all[5][1],x_imag_all[5][1],x_real_all[5][17],x_imag_all[5][17],
signal_box[5*32+1]);
butterfly
my4_2(clk,rstn,x_real_all[4][2],x_imag_all[4][2],x_real_all[4][18],x_imag_all[4][18],fac_real[4],fac
_imag[4],signal_box[4*32+2],x_real_all[5][2],x_imag_all[5][2],x_real_all[5][18],x_imag_all[5][18],
signal_box[5*32+2]);
butterfly
my4_3(clk,rstn,x_real_all[4][3],x_imag_all[4][3],x_real_all[4][19],x_imag_all[4][19],fac_real[6],fac
_imag[6],signal_box[4*32+3],x_real_all[5][3],x_imag_all[5][3],x_real_all[5][19],x_imag_all[5][19],
signal_box[5*32+3]);
butterfly
my4_4(clk,rstn,x_real_all[4][4],x_imag_all[4][4],x_real_all[4][20],x_imag_all[4][20],fac_real[8],fac
_imag[8],signal_box[4*32+4],x_real_all[5][4],x_imag_all[5][4],x_real_all[5][20],x_imag_all[5][20],
signal_box[5*32+4]);
butterfly
my4_5(clk,rstn,x_real_all[4][5],x_imag_all[4][5],x_real_all[4][21],x_imag_all[4][21],fac_real[10],fa
c_imag[10],signal_box[4*32+5],x_real_all[5][5],x_imag_all[5][5],x_real_all[5][21],x_imag_all[5][2
1],signal_box[5*32+5]);
butterfly
my4_6(clk,rstn,x_real_all[4][6],x_imag_all[4][6],x_real_all[4][22],x_imag_all[4][22],fac_real[12],fa
c_imag[12],signal_box[4*32+6],x_real_all[5][6],x_imag_all[5][6],x_real_all[5][22],x_imag_all[5][2
2],signal_box[5*32+6]);
butterfly
my4_7(clk,rstn,x_real_all[4][7],x_imag_all[4][7],x_real_all[4][23],x_imag_all[4][23],fac_real[14],fa
c_imag[14],signal_box[4*32+7],x_real_all[5][7],x_imag_all[5][7],x_real_all[5][23],x_imag_all[5][2
3],signal_box[5*32+7]);
butterfly
my4_8(clk,rstn,x_real_all[4][8],x_imag_all[4][8],x_real_all[4][24],x_imag_all[4][24],fac_real[16],fa
c_imag[16],signal_box[4*32+8],x_real_all[5][8],x_imag_all[5][8],x_real_all[5][24],x_imag_all[5][2
4],signal_box[5*32+8]);
butterfly
my4_9(clk,rstn,x_real_all[4][9],x_imag_all[4][9],x_real_all[4][25],x_imag_all[4][25],fac_real[18],fa
c_imag[18],signal_box[4*32+9],x_real_all[5][9],x_imag_all[5][9],x_real_all[5][25],x_imag_all[5][2
5],signal_box[5*32+9]);
butterfly
my4_10(clk,rstn,x_real_all[4][10],x_imag_all[4][10],x_real_all[4][26],x_imag_all[4][26],fac_real[2
0],fac_imag[20],signal_box[4*32+10],x_real_all[5][10],x_imag_all[5][10],x_real_all[5][26],x_imag
_all[5][26],signal_box[5*32+10]);
butterfly
my4_11(clk,rstn,x_real_all[4][11],x_imag_all[4][11],x_real_all[4][27],x_imag_all[4][27],fac_real[2
2],fac_imag[22],signal_box[4*32+11],x_real_all[5][11],x_imag_all[5][11],x_real_all[5][27],x_imag
_all[5][27],signal_box[5*32+11]);
butterfly
my4_12(clk,rstn,x_real_all[4][12],x_imag_all[4][12],x_real_all[4][28],x_imag_all[4][28],fac_real[2
4],fac_imag[24],signal_box[4*32+12],x_real_all[5][12],x_imag_all[5][12],x_real_all[5][28],x_imag
_all[5][28],signal_box[5*32+12]);
butterfly
my4_13(clk,rstn,x_real_all[4][13],x_imag_all[4][13],x_real_all[4][29],x_imag_all[4][29],fac_real[2
6],fac_imag[26],signal_box[4*32+13],x_real_all[5][13],x_imag_all[5][13],x_real_all[5][29],x_imag
_all[5][29],signal_box[5*32+13]);
butterfly
my4_14(clk,rstn,x_real_all[4][14],x_imag_all[4][14],x_real_all[4][30],x_imag_all[4][30],fac_real[2
8],fac_imag[28],signal_box[4*32+14],x_real_all[5][14],x_imag_all[5][14],x_real_all[5][30],x_imag
_all[5][30],signal_box[5*32+14]);
butterfly
my4_15(clk,rstn,x_real_all[4][15],x_imag_all[4][15],x_real_all[4][31],x_imag_all[4][31],fac_real[3
0],fac_imag[30],signal_box[4*32+15],x_real_all[5][15],x_imag_all[5][15],x_real_all[5][31],x_imag
```

```
_all[5][31],signal_box[5*32+15]);
butterfly
my4_32(clk,rstn,x_real_all[4][32],x_imag_all[4][32],x_real_all[4][48],x_imag_all[4][48],fac_real[0],
fac_imag[0],signal_box[4*32+16],x_real_all[5][32],x_imag_all[5][32],x_real_all[5][48],x_imag_all
[5][48],signal_box[5*32+16]);
butterfly
my4_33(clk,rstn,x_real_all[4][33],x_imag_all[4][33],x_real_all[4][49],x_imag_all[4][49],fac_real[2],
fac_imag[2],signal_box[4*32+17],x_real_all[5][33],x_imag_all[5][33],x_real_all[5][49],x_imag_all
[5][49],signal_box[5*32+17]);
butterfly
my4_34(clk,rstn,x_real_all[4][34],x_imag_all[4][34],x_real_all[4][50],x_imag_all[4][50],fac_real[4],
fac_imag[4],signal_box[4*32+18],x_real_all[5][34],x_imag_all[5][34],x_real_all[5][50],x_imag_all
[5][50],signal_box[5*32+18]);
butterfly
my4_35(clk,rstn,x_real_all[4][35],x_imag_all[4][35],x_real_all[4][51],x_imag_all[4][51],fac_real[6],
fac_imag[6],signal_box[4*32+19],x_real_all[5][35],x_imag_all[5][35],x_real_all[5][51],x_imag_all
[5][51],signal_box[5*32+19]);
butterfly
my4_36(clk,rstn,x_real_all[4][36],x_imag_all[4][36],x_real_all[4][52],x_imag_all[4][52],fac_real[8],
fac_imag[8],signal_box[4*32+20],x_real_all[5][36],x_imag_all[5][36],x_real_all[5][52],x_imag_all
[5][52],signal_box[5*32+20]);
butterfly
my4_37(clk,rstn,x_real_all[4][37],x_imag_all[4][37],x_real_all[4][53],x_imag_all[4][53],fac_real[1
0],fac_imag[10],signal_box[4*32+21],x_real_all[5][37],x_imag_all[5][37],x_real_all[5][53],x_imag
_all[5][53],signal_box[5*32+21]);
butterfly
my4_38(clk,rstn,x_real_all[4][38],x_imag_all[4][38],x_real_all[4][54],x_imag_all[4][54],fac_real[1
2],fac_imag[12],signal_box[4*32+22],x_real_all[5][38],x_imag_all[5][38],x_real_all[5][54],x_imag
_all[5][54],signal_box[5*32+22]);
butterfly
my4_39(clk,rstn,x_real_all[4][39],x_imag_all[4][39],x_real_all[4][55],x_imag_all[4][55],fac_real[1
4],fac_imag[14],signal_box[4*32+23],x_real_all[5][39],x_imag_all[5][39],x_real_all[5][55],x_imag
_all[5][55],signal_box[5*32+23]);
butterfly
my4_40(clk,rstn,x_real_all[4][40],x_imag_all[4][40],x_real_all[4][56],x_imag_all[4][56],fac_real[1
6],fac_imag[16],signal_box[4*32+24],x_real_all[5][40],x_imag_all[5][40],x_real_all[5][56],x_imag
_all[5][56],signal_box[5*32+24]);
butterfly
my4_41(clk,rstn,x_real_all[4][41],x_imag_all[4][41],x_real_all[4][57],x_imag_all[4][57],fac_real[1
8],fac_imag[18],signal_box[4*32+25],x_real_all[5][41],x_imag_all[5][41],x_real_all[5][57],x_imag
_all[5][57],signal_box[5*32+25]);
butterfly
my4_42(clk,rstn,x_real_all[4][42],x_imag_all[4][42],x_real_all[4][58],x_imag_all[4][58],fac_real[2
0],fac_imag[20],signal_box[4*32+26],x_real_all[5][42],x_imag_all[5][42],x_real_all[5][58],x_imag
```

```
_all[5][58],signal_box[5*32+26]);
butterfly
my4_43(clk,rstn,x_real_all[4][43],x_imag_all[4][43],x_real_all[4][59],x_imag_all[4][59],fac_real[22],fac_imag[22],signal_box[4*32+27],x_real_all[5][43],x_imag_all[5][43],x_real_all[5][59],x_imag_all[5][59],signal_box[5*32+27]);
butterfly
my4_44(clk,rstn,x_real_all[4][44],x_imag_all[4][44],x_real_all[4][60],x_imag_all[4][60],fac_real[24],fac_imag[24],signal_box[4*32+28],x_real_all[5][44],x_imag_all[5][44],x_real_all[5][60],x_imag_all[5][60],signal_box[5*32+28]);
butterfly
my4_45(clk,rstn,x_real_all[4][45],x_imag_all[4][45],x_real_all[4][61],x_imag_all[4][61],fac_real[26],fac_imag[26],signal_box[4*32+29],x_real_all[5][45],x_imag_all[5][45],x_real_all[5][61],x_imag_all[5][61],signal_box[5*32+29]);
butterfly
my4_46(clk,rstn,x_real_all[4][46],x_imag_all[4][46],x_real_all[4][62],x_imag_all[4][62],fac_real[28],fac_imag[28],signal_box[4*32+30],x_real_all[5][46],x_imag_all[5][46],x_real_all[5][62],x_imag_all[5][62],signal_box[5*32+30]);
butterfly
my4_47(clk,rstn,x_real_all[4][47],x_imag_all[4][47],x_real_all[4][63],x_imag_all[4][63],fac_real[30],fac_imag[30],signal_box[4*32+31],x_real_all[5][47],x_imag_all[5][47],x_real_all[5][63],x_imag_all[5][63],signal_box[5*32+31]);

//阶段 6
butterfly
my5_0(clk,rstn,x_real_all[5][0],x_imag_all[5][0],x_real_all[5][32],x_imag_all[5][32],fac_real[0],fac_imag[0],signal_box[5*32+0],x_real_all[6][0],x_imag_all[6][0],x_real_all[6][32],x_imag_all[6][32],signal_box[6*32+0]);
butterfly
my5_1(clk,rstn,x_real_all[5][1],x_imag_all[5][1],x_real_all[5][33],x_imag_all[5][33],fac_real[1],fac_imag[1],signal_box[5*32+1],x_real_all[6][1],x_imag_all[6][1],x_real_all[6][33],x_imag_all[6][33],signal_box[6*32+1]);
butterfly
my5_2(clk,rstn,x_real_all[5][2],x_imag_all[5][2],x_real_all[5][34],x_imag_all[5][34],fac_real[2],fac_imag[2],signal_box[5*32+2],x_real_all[6][2],x_imag_all[6][2],x_real_all[6][34],x_imag_all[6][34],signal_box[6*32+2]);
butterfly
my5_3(clk,rstn,x_real_all[5][3],x_imag_all[5][3],x_real_all[5][35],x_imag_all[5][35],fac_real[3],fac_imag[3],signal_box[5*32+3],x_real_all[6][3],x_imag_all[6][3],x_real_all[6][35],x_imag_all[6][35],signal_box[6*32+3]);
butterfly
my5_4(clk,rstn,x_real_all[5][4],x_imag_all[5][4],x_real_all[5][36],x_imag_all[5][36],fac_real[4],fac_imag[4],signal_box[5*32+4],x_real_all[6][4],x_imag_all[6][4],x_real_all[6][36],x_imag_all[6][36],signal_box[6*32+4]);
butterfly
```

```
my5_5(clk,rstn,x_real_all[5][5],x_imag_all[5][5],x_real_all[5][37],x_imag_all[5][37],fac_real[5],fac
_imag[5],signal_box[5*32+5],x_real_all[6][5],x_imag_all[6][5],x_real_all[6][37],x_imag_all[6][37],
signal_box[6*32+5]);
butterfly
my5_6(clk,rstn,x_real_all[5][6],x_imag_all[5][6],x_real_all[5][38],x_imag_all[5][38],fac_real[6],fac
_imag[6],signal_box[5*32+6],x_real_all[6][6],x_imag_all[6][6],x_real_all[6][38],x_imag_all[6][38],
signal_box[6*32+6]);
butterfly
my5_7(clk,rstn,x_real_all[5][7],x_imag_all[5][7],x_real_all[5][39],x_imag_all[5][39],fac_real[7],fac
_imag[7],signal_box[5*32+7],x_real_all[6][7],x_imag_all[6][7],x_real_all[6][39],x_imag_all[6][39],
signal_box[6*32+7]);
butterfly
my5_8(clk,rstn,x_real_all[5][8],x_imag_all[5][8],x_real_all[5][40],x_imag_all[5][40],fac_real[8],fac
_imag[8],signal_box[5*32+8],x_real_all[6][8],x_imag_all[6][8],x_real_all[6][40],x_imag_all[6][40],
signal_box[6*32+8]);
butterfly
my5_9(clk,rstn,x_real_all[5][9],x_imag_all[5][9],x_real_all[5][41],x_imag_all[5][41],fac_real[9],fac
_imag[9],signal_box[5*32+9],x_real_all[6][9],x_imag_all[6][9],x_real_all[6][41],x_imag_all[6][41],
signal_box[6*32+9]);
butterfly
my5_10(clk,rstn,x_real_all[5][10],x_imag_all[5][10],x_real_all[5][42],x_imag_all[5][42],fac_real[1
0],fac_imag[10],signal_box[5*32+10],x_real_all[6][10],x_imag_all[6][10],x_real_all[6][42],x_imag
_all[6][42],signal_box[6*32+10]);
butterfly
my5_11(clk,rstn,x_real_all[5][11],x_imag_all[5][11],x_real_all[5][43],x_imag_all[5][43],fac_real[1
1],fac_imag[11],signal_box[5*32+11],x_real_all[6][11],x_imag_all[6][11],x_real_all[6][43],x_imag
_all[6][43],signal_box[6*32+11]);
butterfly
my5_12(clk,rstn,x_real_all[5][12],x_imag_all[5][12],x_real_all[5][44],x_imag_all[5][44],fac_real[1
2],fac_imag[12],signal_box[5*32+12],x_real_all[6][12],x_imag_all[6][12],x_real_all[6][44],x_imag
_all[6][44],signal_box[6*32+12]);
butterfly
my5_13(clk,rstn,x_real_all[5][13],x_imag_all[5][13],x_real_all[5][45],x_imag_all[5][45],fac_real[1
3],fac_imag[13],signal_box[5*32+13],x_real_all[6][13],x_imag_all[6][13],x_real_all[6][45],x_imag
_all[6][45],signal_box[6*32+13]);
butterfly
my5_14(clk,rstn,x_real_all[5][14],x_imag_all[5][14],x_real_all[5][46],x_imag_all[5][46],fac_real[1
4],fac_imag[14],signal_box[5*32+14],x_real_all[6][14],x_imag_all[6][14],x_real_all[6][46],x_imag
_all[6][46],signal_box[6*32+14]);
butterfly
my5_15(clk,rstn,x_real_all[5][15],x_imag_all[5][15],x_real_all[5][47],x_imag_all[5][47],fac_real[1
5],fac_imag[15],signal_box[5*32+15],x_real_all[6][15],x_imag_all[6][15],x_real_all[6][47],x_imag
_all[6][47],signal_box[6*32+15]);
butterfly
```

```
my5_16(clk,rstn,x_real_all[5][16],x_imag_all[5][16],x_real_all[5][48],x_imag_all[5][48],fac_real[16],fac_imag[16],signal_box[5*32+16],x_real_all[6][16],x_imag_all[6][16],x_real_all[6][48],x_imag_all[6][48],signal_box[6*32+16]);
butterfly
my5_17(clk,rstn,x_real_all[5][17],x_imag_all[5][17],x_real_all[5][49],x_imag_all[5][49],fac_real[17],fac_imag[17],signal_box[5*32+17],x_real_all[6][17],x_imag_all[6][17],x_real_all[6][49],x_imag_all[6][49],signal_box[6*32+17]);
butterfly
my5_18(clk,rstn,x_real_all[5][18],x_imag_all[5][18],x_real_all[5][50],x_imag_all[5][50],fac_real[18],fac_imag[18],signal_box[5*32+18],x_real_all[6][18],x_imag_all[6][18],x_real_all[6][50],x_imag_all[6][50],signal_box[6*32+18]);
butterfly
my5_19(clk,rstn,x_real_all[5][19],x_imag_all[5][19],x_real_all[5][51],x_imag_all[5][51],fac_real[19],fac_imag[19],signal_box[5*32+19],x_real_all[6][19],x_imag_all[6][19],x_real_all[6][51],x_imag_all[6][51],signal_box[6*32+19]);
butterfly
my5_20(clk,rstn,x_real_all[5][20],x_imag_all[5][20],x_real_all[5][52],x_imag_all[5][52],fac_real[20],fac_imag[20],signal_box[5*32+20],x_real_all[6][20],x_imag_all[6][20],x_real_all[6][52],x_imag_all[6][52],signal_box[6*32+20]);
butterfly
my5_21(clk,rstn,x_real_all[5][21],x_imag_all[5][21],x_real_all[5][53],x_imag_all[5][53],fac_real[21],fac_imag[21],signal_box[5*32+21],x_real_all[6][21],x_imag_all[6][21],x_real_all[6][53],x_imag_all[6][53],signal_box[6*32+21]);
butterfly
my5_22(clk,rstn,x_real_all[5][22],x_imag_all[5][22],x_real_all[5][54],x_imag_all[5][54],fac_real[22],fac_imag[22],signal_box[5*32+22],x_real_all[6][22],x_imag_all[6][22],x_real_all[6][54],x_imag_all[6][54],signal_box[6*32+22]);
butterfly
my5_23(clk,rstn,x_real_all[5][23],x_imag_all[5][23],x_real_all[5][55],x_imag_all[5][55],fac_real[23],fac_imag[23],signal_box[5*32+23],x_real_all[6][23],x_imag_all[6][23],x_real_all[6][55],x_imag_all[6][55],signal_box[6*32+23]);
butterfly
my5_24(clk,rstn,x_real_all[5][24],x_imag_all[5][24],x_real_all[5][56],x_imag_all[5][56],fac_real[24],fac_imag[24],signal_box[5*32+24],x_real_all[6][24],x_imag_all[6][24],x_real_all[6][56],x_imag_all[6][56],signal_box[6*32+24]);
butterfly
my5_25(clk,rstn,x_real_all[5][25],x_imag_all[5][25],x_real_all[5][57],x_imag_all[5][57],fac_real[25],fac_imag[25],signal_box[5*32+25],x_real_all[6][25],x_imag_all[6][25],x_real_all[6][57],x_imag_all[6][57],signal_box[6*32+25]);
butterfly
my5_26(clk,rstn,x_real_all[5][26],x_imag_all[5][26],x_real_all[5][58],x_imag_all[5][58],fac_real[26],fac_imag[26],signal_box[5*32+26],x_real_all[6][26],x_imag_all[6][26],x_real_all[6][58],x_imag_all[6][58],signal_box[6*32+26]);
butterfly
```

```
my5_27(clk,rstn,x_real_all[5][27],x_imag_all[5][27],x_real_all[5][59],x_imag_all[5][59],fac_real[2
7],fac_imag[27],signal_box[5*32+27],x_real_all[6][27],x_imag_all[6][27],x_real_all[6][59],x_imag
_all[6][59],signal_box[6*32+27]);
butterfly
my5_28(clk,rstn,x_real_all[5][28],x_imag_all[5][28],x_real_all[5][60],x_imag_all[5][60],fac_real[2
8],fac_imag[28],signal_box[5*32+28],x_real_all[6][28],x_imag_all[6][28],x_real_all[6][60],x_imag
_all[6][60],signal_box[6*32+28]);
butterfly
my5_29(clk,rstn,x_real_all[5][29],x_imag_all[5][29],x_real_all[5][61],x_imag_all[5][61],fac_real[2
9],fac_imag[29],signal_box[5*32+29],x_real_all[6][29],x_imag_all[6][29],x_real_all[6][61],x_imag
_all[6][61],signal_box[6*32+29]);
butterfly
my5_30(clk,rstn,x_real_all[5][30],x_imag_all[5][30],x_real_all[5][62],x_imag_all[5][62],fac_real[3
0],fac_imag[30],signal_box[5*32+30],x_real_all[6][30],x_imag_all[6][30],x_real_all[6][62],x_imag
_all[6][62],signal_box[6*32+30]);
butterfly
my5_31(clk,rstn,x_real_all[5][31],x_imag_all[5][31],x_real_all[5][63],x_imag_all[5][63],fac_real[3
1],fac_imag[31],signal_box[5*32+31],x_real_all[6][31],x_imag_all[6][31],x_real_all[6][63],x_imag
_all[6][63],signal_box[6*32+31]);


  assign output_signal = signal_box[192];
  assign y_real_0 = x_real_all [6] [0];    assign y_imag_0 = x_imag_all [6] [0];
  assign y_real_1 = x_real_all [6] [1];    assign y_imag_1 = x_imag_all [6] [1];
  assign y_real_2 = x_real_all [6] [2];    assign y_imag_2 = x_imag_all [6] [2];
  assign y_real_3 = x_real_all [6] [3];    assign y_imag_3 = x_imag_all [6] [3];
  assign y_real_4 = x_real_all [6] [4];    assign y_imag_4 = x_imag_all [6] [4];
  assign y_real_5 = x_real_all [6] [5];    assign y_imag_5 = x_imag_all [6] [5];
  assign y_real_6 = x_real_all [6] [6];    assign y_imag_6 = x_imag_all [6] [6];
  assign y_real_7 = x_real_all [6] [7];    assign y_imag_7 = x_imag_all [6] [7];
  assign y_real_8 = x_real_all [6] [8];    assign y_imag_8 = x_imag_all [6] [8];
  assign y_real_9 = x_real_all [6] [9];    assign y_imag_9 = x_imag_all [6] [9];
  assign y_real_10 = x_real_all [6] [10];    assign y_imag_10 = x_imag_all [6] [10];
  assign y_real_11 = x_real_all [6] [11];    assign y_imag_11 = x_imag_all [6] [11];
  assign y_real_12 = x_real_all [6] [12];    assign y_imag_12 = x_imag_all [6] [12];
  assign y_real_13 = x_real_all [6] [13];    assign y_imag_13 = x_imag_all [6] [13];
  assign y_real_14 = x_real_all [6] [14];    assign y_imag_14 = x_imag_all [6] [14];
  assign y_real_15 = x_real_all [6] [15];    assign y_imag_15 = x_imag_all [6] [15];
  assign y_real_16 = x_real_all [6] [16];    assign y_imag_16 = x_imag_all [6] [16];
  assign y_real_17 = x_real_all [6] [17];    assign y_imag_17 = x_imag_all [6] [17];
  assign y_real_18 = x_real_all [6] [18];    assign y_imag_18 = x_imag_all [6] [18];
  assign y_real_19 = x_real_all [6] [19];    assign y_imag_19 = x_imag_all [6] [19];
  assign y_real_20 = x_real_all [6] [20];    assign y_imag_20 = x_imag_all [6] [20];
  assign y_real_21 = x_real_all [6] [21];    assign y_imag_21 = x_imag_all [6] [21];
  assign y_real_22 = x_real_all [6] [22];    assign y_imag_22 = x_imag_all [6] [22];
  assign y_real_23 = x_real_all [6] [23];    assign y_imag_23 = x_imag_all [6] [23];
  assign y_real_24 = x_real_all [6] [24];    assign y_imag_24 = x_imag_all [6] [24];
  assign y_real_25 = x_real_all [6] [25];    assign y_imag_25 = x_imag_all [6] [25];
  assign y_real_26 = x_real_all [6] [26];    assign y_imag_26 = x_imag_all [6] [26];
  assign y_real_27 = x_real_all [6] [27];    assign y_imag_27 = x_imag_all [6] [27];
  assign y_real_28 = x_real_all [6] [28];    assign y_imag_28 = x_imag_all [6] [28];
  assign y_real_29 = x_real_all [6] [29];    assign y_imag_29 = x_imag_all [6] [29];
  assign y_real_30 = x_real_all [6] [30];    assign y_imag_30 = x_imag_all [6] [30];
  assign y_real_31 = x_real_all [6] [31];    assign y_imag_31 = x_imag_all [6] [31];
  assign y_real_32 = x_real_all [6] [32];    assign y_imag_32 = x_imag_all [6] [32];
  assign y_real_33 = x_real_all [6] [33];    assign y_imag_33 = x_imag_all [6] [33];
  assign y_real_34 = x_real_all [6] [34];    assign y_imag_34 = x_imag_all [6] [34];
  assign y_real_35 = x_real_all [6] [35];    assign y_imag_35 = x_imag_all [6] [35];
  assign y_real_36 = x_real_all [6] [36];    assign y_imag_36 = x_imag_all [6] [36];
  assign y_real_37 = x_real_all [6] [37];    assign y_imag_37 = x_imag_all [6] [37];
  assign y_real_38 = x_real_all [6] [38];    assign y_imag_38 = x_imag_all [6] [38];
  assign y_real_39 = x_real_all [6] [39];    assign y_imag_39 = x_imag_all [6] [39];
  assign y_real_40 = x_real_all [6] [40];    assign y_imag_40 = x_imag_all [6] [40];
  assign y_real_41 = x_real_all [6] [41];    assign y_imag_41 = x_imag_all [6] [41];
  assign y_real_42 = x_real_all [6] [42];    assign y_imag_42 = x_imag_all [6] [42];
  assign y_real_43 = x_real_all [6] [43];    assign y_imag_43 = x_imag_all [6] [43];
  assign y_real_44 = x_real_all [6] [44];    assign y_imag_44 = x_imag_all [6] [44];
  assign y_real_45 = x_real_all [6] [45];    assign y_imag_45 = x_imag_all [6] [45];
  assign y_real_46 = x_real_all [6] [46];    assign y_imag_46 = x_imag_all [6] [46];
  assign y_real_47 = x_real_all [6] [47];    assign y_imag_47 = x_imag_all [6] [47];
  assign y_real_48 = x_real_all [6] [48];    assign y_imag_48 = x_imag_all [6] [48];
  assign y_real_49 = x_real_all [6] [49];    assign y_imag_49 = x_imag_all [6] [49];
  assign y_real_50 = x_real_all [6] [50];    assign y_imag_50 = x_imag_all [6] [50];
  assign y_real_51 = x_real_all [6] [51];    assign y_imag_51 = x_imag_all [6] [51];
  assign y_real_52 = x_real_all [6] [52];    assign y_imag_52 = x_imag_all [6] [52];
  assign y_real_53 = x_real_all [6] [53];    assign y_imag_53 = x_imag_all [6] [53];
  assign y_real_54 = x_real_all [6] [54];    assign y_imag_54 = x_imag_all [6] [54];
  assign y_real_55 = x_real_all [6] [55];    assign y_imag_55 = x_imag_all [6] [55];
  assign y_real_56 = x_real_all [6] [56];    assign y_imag_56 = x_imag_all [6] [56];
  assign y_real_57 = x_real_all [6] [57];    assign y_imag_57 = x_imag_all [6] [57];
  assign y_real_58 = x_real_all [6] [58];    assign y_imag_58 = x_imag_all [6] [58];
  assign y_real_59 = x_real_all [6] [59];    assign y_imag_59 = x_imag_all [6] [59];
  assign y_real_60 = x_real_all [6] [60];    assign y_imag_60 = x_imag_all [6] [60];
  assign y_real_61 = x_real_all [6] [61];    assign y_imag_61 = x_imag_all [6] [61];
  assign y_real_62 = x_real_all [6] [62];    assign y_imag_62 = x_imag_all [6] [62];
  assign y_real_63 = x_real_all [6] [63];    assign y_imag_63 = x_imag_all [6] [63];

endmodule
```

附录四、design code 的蝶形结构模块

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2020/12/17 11:43:12
// Design Name:
// Module Name: butterfly
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////

//  如果是如下的蝴蝶
// indata1    -----·-------------·----> indata1'
//                    \        /
//                     \      /
//                      \    /
//                       \  /
//                        \/
//                        /\
//                       /  \
//                      /    \
//                     /      \
//                    /        \
// indata2    -----·-------------·----> indata2'
//             Wn          -1
// indata1'_实部  = indata1_实部  + indata2_实部*Wn_实部  - indata2_虚部*Wn_虚部
// indata1'_虚部  = indata1_虚部  + indata2_实部*Wn_虚部  + indata2_虚部*Wn_实部
// indata2'_实部  = indata1_实部  - indata2_实部*Wn_实部  + indata2_虚部*Wn_虚部
// indata2'_虚部  = indata1_虚部  - indata2_实部*Wn_虚部  - indata2_虚部*Wn_实部

//  陈述关于 signed 变量的几个事实
// 1、两个 signed 变量 bit 相等，带上同样多的符号位，则乘积结果的符号位为两个变量的
符号位数之和，可以通过结果符号位进一步研判是否进位。
```

```
// 2、两个 signed 变量相乘，其非符号位相乘的结果位数是两个变量的非符号位之和。结果位数的剩余都是符号位。见第 5 条说明
// 3、两个 signed 变量相加，其结果符号位数为两个变量符号位之和。
// 4、对 wn 的实部和虚部采用符号位首位，其余位由小于 1 的数字右移 7 位。
// 5、两个 signed 变量，1 个 4 位(符号位 1 位)，1 个 5 位(符号位 2 位)，结果输出设定 9 位，则结果的符号位必定为 3 位。
// 6、不等位符号位的两个 signed 变量相加，其结果的符号位不定，只取决于符号位
// 7、11100，如果两位符号位，则有 11100-->11011-->11100=-4;如果一位符号位，则有 11100-->11011-->10100=-4.

//  几个优化符号位的手段
// -1+(-1) = -2   以二进制补码为例，1111+1111 = 1110
// -7+(-7) = -14  以二进制补码为例，1001+1001 = 10010

module butterfly(
  input clk,
  input rstn,
  input signed [19:0] indata1_real,
  input signed [19:0] indata1_imag,
  input signed [19:0] indata2_real,
  input signed [19:0] indata2_imag,
  input signed [15:0] wn_real,            //  约定只有一位符号位
  input signed [15:0] wn_imag,
  input regular_signal,                   //  避免后续出现竞争
  output signed [19:0] oudata1_real,//  输出只有一位符号位
  output signed [19:0] oudata1_imag,
  output signed [19:0] oudata2_real,
  output signed [19:0] oudata2_imag,
  output over_signal);                    //用于避免后续出现竞争

  // en 信号的右移位寄存器，便于规范流水线
  reg [4:0] regular_register;
  assign over_signal = regular_register[2];
  always @(posedge clk or negedge rstn) begin
      if (!rstn) begin
          regular_register <= 5'b0;
          end
      else begin
          regular_register <= {regular_register[3:0],regular_signal};
          end
      end
 //  以下存储 indata2 和 Wn 相乘的两个实数部分和虚数部分，-----------以下四个变量都是 1 位符号位
  reg signed [35:0] indata2_wn_real1; //两个实部相乘
 reg signed [35:0] indata2_wn_real2; //两个虚部相乘，相乘虽然有负号，但无所谓
 reg signed [35:0] indata2_wn_imag1; //indata2 实部和 Wn 虚部
 reg signed [35:0] indata2_wn_imag2; //indata2 虚部和 Wn 实部
 reg signed [35:0] indata1_real_tmp;
 reg signed [35:0] indata1_imag_tmp;

 always @(posedge clk or negedge rstn) begin
    if (!rstn) begin
       indata2_wn_real1 <= 36'd0;indata2_wn_real2 <= 36'd0;
       indata2_wn_imag1 <= 36'd0;indata2_wn_imag2 <= 36'd0;
       indata1_real_tmp <= 36'd0;indata1_imag_tmp <= 36'd0;
      end
    else if(regular_signal) begin
       indata2_wn_real1 <= indata2_real*wn_real; // rr
       indata2_wn_real2 <= indata2_imag*wn_imag; // ii
       indata2_wn_imag1 <= indata2_real*wn_imag; // ri
       indata2_wn_imag2 <= indata2_imag*wn_real; // ir
       indata1_real_tmp      <=      {{3{indata1_real[19]}},indata1_real[19:0],13'd0};      //
28'b00_000_0000_0000_0000_0001_0000_000
       indata1_imag_tmp     <=     {{3{indata1_imag[19]}},indata1_imag[19:0],13'd0};     //
28'b10_000_0000_0000_0000_0001_0000_000
      end
 end
 //  以下存储实部-虚部对应相乘的结果的相加减项
 reg signed [35:0] indata2_rrii; //此处以 indata1'的实部和虚部作为变量存在的依据
 reg signed [35:0] indata2_riir;
 //很显然需要做的预备工作是扩展 indata1 实部和虚部的位数
 reg signed [35:0] indata1_real_tmp1;
 reg signed [35:0] indata1_imag_tmp1;

  always @(posedge clk or negedge rstn) begin
   if (!rstn) begin
      indata2_rrii <= 36'd0;
      indata2_riir <= 36'd0;
      indata1_imag_tmp1 <= 36'd0;
      indata1_real_tmp1 <= 36'd0;
      end
   else if(regular_register[0]) begin
      indata2_rrii <= indata2_wn_real1 - indata2_wn_real2;
      indata2_riir <= indata2_wn_imag1 + indata2_wn_imag2;
      indata1_real_tmp1 <= indata1_real_tmp;
      indata1_imag_tmp1 <= indata1_imag_tmp;
      end
  end
```

```
// 以下存储最终结果
reg signed [35:0] oudata1_real_tmp;
reg signed [35:0] oudata1_imag_tmp;
reg signed [35:0] oudata2_real_tmp;
reg signed [35:0] oudata2_imag_tmp;

always @(posedge clk or negedge rstn) begin
  if (!rstn) begin
        oudata1_real_tmp <= 36'd0;
        oudata1_imag_tmp <= 36'd0;
        oudata2_real_tmp <= 36'd0;
        oudata2_imag_tmp <= 36'd0;
        end
    else if(regular_register[1]) begin
        oudata1_real_tmp <= indata1_real_tmp1 + indata2_rrii;
        oudata1_imag_tmp <= indata1_imag_tmp1 + indata2_riir;
        oudata2_real_tmp <= indata1_real_tmp1 - indata2_rrii;
        oudata2_imag_tmp <= indata1_imag_tmp1 - indata2_riir;
        end
end
// 以下截断并生成仅有一位符号位的输出结果，后 7 位必然不要
// 8'b1111_1111 = 255 7'b0111_1111 = 127 6'b0011_1111 = 63 5'b0001_1111 = 31
// 4'b0000_1111 = 15    3'b0000_0111 = 7 2'b0000_0011 = 3 1'b0000_0001 = 1
/* assign oudata1_real = (oudata1_real_tmp[28:26]==3'b000 ||
oudata1_real_tmp[28:26]==3'b111) ? {oudata1_real_tmp[28],oudata1_real_tmp[25:7]} :

((oudata1_real_tmp[28:27]==2'b00&&oudata1_real_tmp[26]!=0)||(oudata1_real_tmp[28:27]=
=2'b11&&oudata1_real_tmp[26]!=1)) ? {oudata1_real_tmp[28],oudata1_real_tmp[26:8]} :
oudata1_real_tmp[28:9];
  assign oudata2_real = (oudata2_real_tmp[28:26]==3'b000 ||
oudata2_real_tmp[28:26]==3'b111) ? {oudata2_real_tmp[28],oudata2_real_tmp[25:7]} :

((oudata2_real_tmp[28:27]==2'b00&&oudata2_real_tmp[26]!=0)||(oudata2_real_tmp[28:27]=
=2'b11&&oudata2_real_tmp[26]!=1)) ? {oudata2_real_tmp[28],oudata2_real_tmp[26:8]} :
oudata2_real_tmp[28:9];
  assign oudata1_imag = (oudata1_imag_tmp[28:26]==3'b000 ||
oudata1_imag_tmp[28:26]==3'b111) ? {oudata1_imag_tmp[28],oudata1_imag_tmp[25:7]} :

((oudata1_imag_tmp[28:27]==2'b00&&oudata1_imag_tmp[26]!=0)||(oudata1_imag_tmp[28:
27]==2'b11&&oudata1_imag_tmp[26]!=1)) ?
{oudata1_imag_tmp[28],oudata1_imag_tmp[26:8]} : oudata1_imag_tmp[28:9];
  assign oudata2_imag = (oudata2_imag_tmp[28:26]==3'b000 ||
oudata2_imag_tmp[28:26]==3'b111) ? {oudata2_imag_tmp[28],oudata2_imag_tmp[25:7]} :
```

```
((oudata2_imag_tmp[28:27]==2'b00&&oudata2_imag_tmp[26]!=0)||(oudata2_imag_tmp[28:
27]==2'b11&&oudata2_imag_tmp[26]!=1))                                            ?
{oudata2_imag_tmp[28],oudata2_imag_tmp[26:8]} : oudata2_imag_tmp[28:9];
  */
    assign oudata1_real = {oudata1_real_tmp[35],oudata1_real_tmp[13+18:13]};
    assign oudata1_imag = {oudata1_imag_tmp[35],oudata1_imag_tmp[13+18:13]};
    assign oudata2_real = {oudata2_real_tmp[35],oudata2_real_tmp[13+18:13]};
    assign oudata2_imag = {oudata2_imag_tmp[35],oudata2_imag_tmp[13+18:13]};
endmodule
```

附录五、test code

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 2020/12/20 20:42:32
// Design Name:
// Module Name: fft64_test
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module fft64_test();
  reg clk;
  reg rstn;
  reg input_signal;
  reg signed [7:0] x_real_0; reg signed [7:0] x_imag_0;
  reg signed [7:0] x_real_1; reg signed [7:0] x_imag_1;
  reg signed [7:0] x_real_2; reg signed [7:0] x_imag_2;
  reg signed [7:0] x_real_3; reg signed [7:0] x_imag_3;
  reg signed [7:0] x_real_4; reg signed [7:0] x_imag_4;
  reg signed [7:0] x_real_5; reg signed [7:0] x_imag_5;
  reg signed [7:0] x_real_6; reg signed [7:0] x_imag_6;
  reg signed [7:0] x_real_7; reg signed [7:0] x_imag_7;
  reg signed [7:0] x_real_8; reg signed [7:0] x_imag_8;
  reg signed [7:0] x_real_9; reg signed [7:0] x_imag_9;
  reg signed [7:0] x_real_10; reg signed [7:0] x_imag_10;
  reg signed [7:0] x_real_11; reg signed [7:0] x_imag_11;
  reg signed [7:0] x_real_12; reg signed [7:0] x_imag_12;
  reg signed [7:0] x_real_13; reg signed [7:0] x_imag_13;
  reg signed [7:0] x_real_14; reg signed [7:0] x_imag_14;
  reg signed [7:0] x_real_15; reg signed [7:0] x_imag_15;
  reg signed [7:0] x_real_16; reg signed [7:0] x_imag_16;
  reg signed [7:0] x_real_17; reg signed [7:0] x_imag_17;
  reg signed [7:0] x_real_18; reg signed [7:0] x_imag_18;
  reg signed [7:0] x_real_19; reg signed [7:0] x_imag_19;
  reg signed [7:0] x_real_20; reg signed [7:0] x_imag_20;
  reg signed [7:0] x_real_21; reg signed [7:0] x_imag_21;
  reg signed [7:0] x_real_22; reg signed [7:0] x_imag_22;
  reg signed [7:0] x_real_23; reg signed [7:0] x_imag_23;
  reg signed [7:0] x_real_24; reg signed [7:0] x_imag_24;
  reg signed [7:0] x_real_25; reg signed [7:0] x_imag_25;
  reg signed [7:0] x_real_26; reg signed [7:0] x_imag_26;
  reg signed [7:0] x_real_27; reg signed [7:0] x_imag_27;
  reg signed [7:0] x_real_28; reg signed [7:0] x_imag_28;
  reg signed [7:0] x_real_29; reg signed [7:0] x_imag_29;
  reg signed [7:0] x_real_30; reg signed [7:0] x_imag_30;
  reg signed [7:0] x_real_31; reg signed [7:0] x_imag_31;
  reg signed [7:0] x_real_32; reg signed [7:0] x_imag_32;
  reg signed [7:0] x_real_33; reg signed [7:0] x_imag_33;
  reg signed [7:0] x_real_34; reg signed [7:0] x_imag_34;
  reg signed [7:0] x_real_35; reg signed [7:0] x_imag_35;
  reg signed [7:0] x_real_36; reg signed [7:0] x_imag_36;
  reg signed [7:0] x_real_37; reg signed [7:0] x_imag_37;
  reg signed [7:0] x_real_38; reg signed [7:0] x_imag_38;
  reg signed [7:0] x_real_39; reg signed [7:0] x_imag_39;
  reg signed [7:0] x_real_40; reg signed [7:0] x_imag_40;
  reg signed [7:0] x_real_41; reg signed [7:0] x_imag_41;
  reg signed [7:0] x_real_42; reg signed [7:0] x_imag_42;
  reg signed [7:0] x_real_43; reg signed [7:0] x_imag_43;
  reg signed [7:0] x_real_44; reg signed [7:0] x_imag_44;
  reg signed [7:0] x_real_45; reg signed [7:0] x_imag_45;
  reg signed [7:0] x_real_46; reg signed [7:0] x_imag_46;
  reg signed [7:0] x_real_47; reg signed [7:0] x_imag_47;
  reg signed [7:0] x_real_48; reg signed [7:0] x_imag_48;
  reg signed [7:0] x_real_49; reg signed [7:0] x_imag_49;
  reg signed [7:0] x_real_50; reg signed [7:0] x_imag_50;
  reg signed [7:0] x_real_51; reg signed [7:0] x_imag_51;
  reg signed [7:0] x_real_52; reg signed [7:0] x_imag_52;
  reg signed [7:0] x_real_53; reg signed [7:0] x_imag_53;
  reg signed [7:0] x_real_54; reg signed [7:0] x_imag_54;
  reg signed [7:0] x_real_55; reg signed [7:0] x_imag_55;
  reg signed [7:0] x_real_56; reg signed [7:0] x_imag_56;
  reg signed [7:0] x_real_57; reg signed [7:0] x_imag_57;
  reg signed [7:0] x_real_58; reg signed [7:0] x_imag_58;
  reg signed [7:0] x_real_59; reg signed [7:0] x_imag_59;
  reg signed [7:0] x_real_60; reg signed [7:0] x_imag_60;
  reg signed [7:0] x_real_61; reg signed [7:0] x_imag_61;
  reg signed [7:0] x_real_62; reg signed [7:0] x_imag_62;
  reg signed [7:0] x_real_63; reg signed [7:0] x_imag_63;

  wire signed [19:0] y_real_0; wire signed [19:0] y_imag_0;
  wire signed [19:0] y_real_1; wire signed [19:0] y_imag_1;
  wire signed [19:0] y_real_2; wire signed [19:0] y_imag_2;
  wire signed [19:0] y_real_3; wire signed [19:0] y_imag_3;
  wire signed [19:0] y_real_4; wire signed [19:0] y_imag_4;
  wire signed [19:0] y_real_5; wire signed [19:0] y_imag_5;
  wire signed [19:0] y_real_6; wire signed [19:0] y_imag_6;
  wire signed [19:0] y_real_7; wire signed [19:0] y_imag_7;
  wire signed [19:0] y_real_8; wire signed [19:0] y_imag_8;
  wire signed [19:0] y_real_9; wire signed [19:0] y_imag_9;
  wire signed [19:0] y_real_10; wire signed [19:0] y_imag_10;
  wire signed [19:0] y_real_11; wire signed [19:0] y_imag_11;
  wire signed [19:0] y_real_12; wire signed [19:0] y_imag_12;
  wire signed [19:0] y_real_13; wire signed [19:0] y_imag_13;
  wire signed [19:0] y_real_14; wire signed [19:0] y_imag_14;
  wire signed [19:0] y_real_15; wire signed [19:0] y_imag_15;
  wire signed [19:0] y_real_16; wire signed [19:0] y_imag_16;
  wire signed [19:0] y_real_17; wire signed [19:0] y_imag_17;
  wire signed [19:0] y_real_18; wire signed [19:0] y_imag_18;
  wire signed [19:0] y_real_19; wire signed [19:0] y_imag_19;
  wire signed [19:0] y_real_20; wire signed [19:0] y_imag_20;
  wire signed [19:0] y_real_21; wire signed [19:0] y_imag_21;
  wire signed [19:0] y_real_22; wire signed [19:0] y_imag_22;
  wire signed [19:0] y_real_23; wire signed [19:0] y_imag_23;
  wire signed [19:0] y_real_24; wire signed [19:0] y_imag_24;
  wire signed [19:0] y_real_25; wire signed [19:0] y_imag_25;
  wire signed [19:0] y_real_26; wire signed [19:0] y_imag_26;
  wire signed [19:0] y_real_27; wire signed [19:0] y_imag_27;
  wire signed [19:0] y_real_28; wire signed [19:0] y_imag_28;
  wire signed [19:0] y_real_29; wire signed [19:0] y_imag_29;
  wire signed [19:0] y_real_30; wire signed [19:0] y_imag_30;
  wire signed [19:0] y_real_31; wire signed [19:0] y_imag_31;
  wire signed [19:0] y_real_32; wire signed [19:0] y_imag_32;
  wire signed [19:0] y_real_33; wire signed [19:0] y_imag_33;
  wire signed [19:0] y_real_34; wire signed [19:0] y_imag_34;
  wire signed [19:0] y_real_35; wire signed [19:0] y_imag_35;
  wire signed [19:0] y_real_36; wire signed [19:0] y_imag_36;
  wire signed [19:0] y_real_37; wire signed [19:0] y_imag_37;
  wire signed [19:0] y_real_38; wire signed [19:0] y_imag_38;
  wire signed [19:0] y_real_39; wire signed [19:0] y_imag_39;
  wire signed [19:0] y_real_40; wire signed [19:0] y_imag_40;
  wire signed [19:0] y_real_41; wire signed [19:0] y_imag_41;
  wire signed [19:0] y_real_42; wire signed [19:0] y_imag_42;
  wire signed [19:0] y_real_43; wire signed [19:0] y_imag_43;
  wire signed [19:0] y_real_44; wire signed [19:0] y_imag_44;
  wire signed [19:0] y_real_45; wire signed [19:0] y_imag_45;
  wire signed [19:0] y_real_46; wire signed [19:0] y_imag_46;
  wire signed [19:0] y_real_47; wire signed [19:0] y_imag_47;
  wire signed [19:0] y_real_48; wire signed [19:0] y_imag_48;
  wire signed [19:0] y_real_49; wire signed [19:0] y_imag_49;
  wire signed [19:0] y_real_50; wire signed [19:0] y_imag_50;
  wire signed [19:0] y_real_51; wire signed [19:0] y_imag_51;
  wire signed [19:0] y_real_52; wire signed [19:0] y_imag_52;
  wire signed [19:0] y_real_53; wire signed [19:0] y_imag_53;
  wire signed [19:0] y_real_54; wire signed [19:0] y_imag_54;
  wire signed [19:0] y_real_55; wire signed [19:0] y_imag_55;
  wire signed [19:0] y_real_56; wire signed [19:0] y_imag_56;
  wire signed [19:0] y_real_57; wire signed [19:0] y_imag_57;
  wire signed [19:0] y_real_58; wire signed [19:0] y_imag_58;
  wire signed [19:0] y_real_59; wire signed [19:0] y_imag_59;
  wire signed [19:0] y_real_60; wire signed [19:0] y_imag_60;
  wire signed [19:0] y_real_61; wire signed [19:0] y_imag_61;
  wire signed [19:0] y_real_62; wire signed [19:0] y_imag_62;
  wire signed [19:0] y_real_63; wire signed [19:0] y_imag_63;
  wire output_signal;


  fft64 myfft64(clk,rstn,input_signal,
  x_real_0, x_imag_0, x_real_1, x_imag_1, x_real_2, x_imag_2, x_real_3, x_imag_3,
  x_real_4, x_imag_4, x_real_5, x_imag_5, x_real_6, x_imag_6, x_real_7, x_imag_7,
  x_real_8, x_imag_8, x_real_9, x_imag_9, x_real_10, x_imag_10, x_real_11, x_imag_11,
  x_real_12, x_imag_12, x_real_13, x_imag_13, x_real_14, x_imag_14, x_real_15, x_imag_15,
  x_real_16, x_imag_16, x_real_17, x_imag_17, x_real_18, x_imag_18, x_real_19, x_imag_19,
  x_real_20, x_imag_20, x_real_21, x_imag_21, x_real_22, x_imag_22, x_real_23, x_imag_23,
  x_real_24, x_imag_24, x_real_25, x_imag_25, x_real_26, x_imag_26, x_real_27, x_imag_27,
  x_real_28, x_imag_28, x_real_29, x_imag_29, x_real_30, x_imag_30, x_real_31, x_imag_31,
  x_real_32, x_imag_32, x_real_33, x_imag_33, x_real_34, x_imag_34, x_real_35, x_imag_35,
  x_real_36, x_imag_36, x_real_37, x_imag_37, x_real_38, x_imag_38, x_real_39, x_imag_39,
  x_real_40, x_imag_40, x_real_41, x_imag_41, x_real_42, x_imag_42, x_real_43, x_imag_43,
  x_real_44, x_imag_44, x_real_45, x_imag_45, x_real_46, x_imag_46, x_real_47, x_imag_47,
  x_real_48, x_imag_48, x_real_49, x_imag_49, x_real_50, x_imag_50, x_real_51, x_imag_51,
  x_real_52, x_imag_52, x_real_53, x_imag_53, x_real_54, x_imag_54, x_real_55, x_imag_55,
  x_real_56, x_imag_56, x_real_57, x_imag_57, x_real_58, x_imag_58, x_real_59, x_imag_59,
  x_real_60, x_imag_60, x_real_61, x_imag_61, x_real_62, x_imag_62, x_real_63, x_imag_63,
  y_real_0, y_imag_0, y_real_1, y_imag_1, y_real_2, y_imag_2, y_real_3, y_imag_3,
  y_real_4, y_imag_4, y_real_5, y_imag_5, y_real_6, y_imag_6, y_real_7, y_imag_7,
  y_real_8, y_imag_8, y_real_9, y_imag_9, y_real_10, y_imag_10, y_real_11, y_imag_11,
  y_real_12, y_imag_12, y_real_13, y_imag_13, y_real_14, y_imag_14, y_real_15, y_imag_15,
  y_real_16, y_imag_16, y_real_17, y_imag_17, y_real_18, y_imag_18, y_real_19, y_imag_19,
  y_real_20, y_imag_20, y_real_21, y_imag_21, y_real_22, y_imag_22, y_real_23, y_imag_23,
  y_real_24, y_imag_24, y_real_25, y_imag_25, y_real_26, y_imag_26, y_real_27, y_imag_27,
  y_real_28, y_imag_28, y_real_29, y_imag_29, y_real_30, y_imag_30, y_real_31, y_imag_31,
  y_real_32, y_imag_32, y_real_33, y_imag_33, y_real_34, y_imag_34, y_real_35, y_imag_35,
  y_real_36, y_imag_36, y_real_37, y_imag_37, y_real_38, y_imag_38, y_real_39, y_imag_39,
  y_real_40, y_imag_40, y_real_41, y_imag_41, y_real_42, y_imag_42, y_real_43, y_imag_43,
  y_real_44, y_imag_44, y_real_45, y_imag_45, y_real_46, y_imag_46, y_real_47, y_imag_47,
  y_real_48, y_imag_48, y_real_49, y_imag_49, y_real_50, y_imag_50, y_real_51, y_imag_51,
  y_real_52, y_imag_52, y_real_53, y_imag_53, y_real_54, y_imag_54, y_real_55, y_imag_55,
  y_real_56, y_imag_56, y_real_57, y_imag_57, y_real_58, y_imag_58, y_real_59, y_imag_59,
  y_real_60, y_imag_60, y_real_61, y_imag_61, y_real_62, y_imag_62, y_real_63, y_imag_63,
  output_signal
  );

      initial begin
          clk = 0; //50MHz
          rstn = 0 ;
          #15 rstn = 1;
          forever begin
              #15 clk = ~clk; //50MHz
          end
      end

  initial begin
    input_signal = 0;
    x_real_0 = 8'd1;     x_imag_0 = 8'd1;
    x_real_1 = 8'd2;     x_imag_1 = 8'd2;
    x_real_2 = 8'd3;     x_imag_2 = 8'd1;
    x_real_3 = 8'd4;     x_imag_3 = 8'd2;
    x_real_4 = 8'd5;     x_imag_4 = 8'd1;
```

```
x_real_5 = 8'd1;      x_imag_5 = 8'd2;
x_real_6 = 8'd2;      x_imag_6 = 8'd1;
x_real_7 = 8'd3;      x_imag_7 = 8'd2;
x_real_8 = 8'd4;      x_imag_8 = 8'd1;
x_real_9 = 8'd5;      x_imag_9 = 8'd2;
x_real_10 = 8'd1;      x_imag_10 = 8'd1;
x_real_11 = 8'd2;      x_imag_11 = 8'd2;
x_real_12 = 8'd3;      x_imag_12 = 8'd1;
x_real_13 = 8'd4;      x_imag_13 = 8'd2;
x_real_14 = 8'd5;      x_imag_14 = 8'd1;
x_real_15 = 8'd1;      x_imag_15 = 8'd2;
x_real_16 = 8'd2;      x_imag_16 = 8'd1;
x_real_17 = 8'd3;      x_imag_17 = 8'd2;
x_real_18 = 8'd4;      x_imag_18 = 8'd1;
x_real_19 = 8'd5;      x_imag_19 = 8'd2;
x_real_20 = 8'd1;      x_imag_20 = 8'd1;
x_real_21 = 8'd2;      x_imag_21 = 8'd2;
x_real_22 = 8'd3;      x_imag_22 = 8'd1;
x_real_23 = 8'd4;      x_imag_23 = 8'd2;
x_real_24 = 8'd5;      x_imag_24 = 8'd1;
x_real_25 = 8'd1;      x_imag_25 = 8'd2;
x_real_26 = 8'd2;      x_imag_26 = 8'd1;
x_real_27 = 8'd3;      x_imag_27 = 8'd2;
x_real_28 = 8'd4;      x_imag_28 = 8'd1;
x_real_29 = 8'd5;      x_imag_29 = 8'd2;
x_real_30 = 8'd1;      x_imag_30 = 8'd1;
x_real_31 = 8'd2;      x_imag_31 = 8'd2;
x_real_32 = 8'd3;      x_imag_32 = 8'd1;
x_real_33 = 8'd4;      x_imag_33 = 8'd2;
x_real_34 = 8'd5;      x_imag_34 = 8'd1;
x_real_35 = 8'd1;      x_imag_35 = 8'd2;
x_real_36 = 8'd2;      x_imag_36 = 8'd1;
x_real_37 = 8'd3;      x_imag_37 = 8'd2;
x_real_38 = 8'd4;      x_imag_38 = 8'd1;
x_real_39 = 8'd5;      x_imag_39 = 8'd2;
x_real_40 = 8'd1;      x_imag_40 = 8'd1;
x_real_41 = 8'd2;      x_imag_41 = 8'd2;
x_real_42 = 8'd3;      x_imag_42 = 8'd1;
x_real_43 = 8'd4;      x_imag_43 = 8'd2;
x_real_44 = 8'd5;      x_imag_44 = 8'd1;
x_real_45 = 8'd1;      x_imag_45 = 8'd2;
x_real_46 = 8'd2;      x_imag_46 = 8'd1;
x_real_47 = 8'd3;      x_imag_47 = 8'd2;
x_real_48 = 8'd4;      x_imag_48 = 8'd1;
```

```
        x_real_49 = 8'd5;       x_imag_49 = 8'd2;
        x_real_50 = 8'd1;       x_imag_50 = 8'd1;
        x_real_51 = 8'd2;       x_imag_51 = 8'd2;
        x_real_52 = 8'd3;       x_imag_52 = 8'd1;
        x_real_53 = 8'd4;       x_imag_53 = 8'd2;
        x_real_54 = 8'd5;       x_imag_54 = 8'd1;
        x_real_55 = 8'd1;       x_imag_55 = 8'd2;
        x_real_56 = 8'd2;       x_imag_56 = 8'd1;
        x_real_57 = 8'd3;       x_imag_57 = 8'd2;
        x_real_58 = 8'd4;       x_imag_58 = 8'd1;
        x_real_59 = 8'd5;       x_imag_59 = 8'd2;
        x_real_60 = 8'd1;       x_imag_60 = 8'd1;
        x_real_61 = 8'd2;       x_imag_61 = 8'd2;
        x_real_62 = 8'd3;       x_imag_62 = 8'd1;
        x_real_63 = 8'd4;       x_imag_63 = 8'd2;
        @(negedge clk) ;
        input_signal = 1 ;
      /* forever begin
            @(negedge clk) ;
                x_real_0    = (x_real_0    > 8'hFF) ? 8'h00 : x_real_0    + 1; x_imag_0    =
(x_imag_0    > 8'hFF) ? 8'h00 : x_imag_0    + 4;
                x_real_1    = (x_real_1    > 8'hFF) ? 8'h00 : x_real_1    + 2; x_imag_1    =
(x_imag_1    > 8'hFF) ? 8'h00 : x_imag_1    + 3;
                x_real_2    = (x_real_2    > 8'hFF) ? 8'h00 : x_real_2    + 3; x_imag_2    =
(x_imag_2    > 8'hFF) ? 8'h00 : x_imag_2    + 2;
                x_real_3    = (x_real_3    > 8'hFF) ? 8'h00 : x_real_3    + 4; x_imag_3    =
(x_imag_3    > 8'hFF) ? 8'h00 : x_imag_3    + 1;
                x_real_4    = (x_real_4    > 8'hFF) ? 8'h00 : x_real_4    + 1; x_imag_4    =
(x_imag_4    > 8'hFF) ? 8'h00 : x_imag_4    + 4;
                x_real_5    = (x_real_5    > 8'hFF) ? 8'h00 : x_real_5    + 2; x_imag_5    =
(x_imag_5    > 8'hFF) ? 8'h00 : x_imag_5    + 3;
                x_real_6    = (x_real_6    > 8'hFF) ? 8'h00 : x_real_6    + 3; x_imag_6    =
(x_imag_6    > 8'hFF) ? 8'h00 : x_imag_6    + 2;
                x_real_7    = (x_real_7    > 8'hFF) ? 8'h00 : x_real_7    + 4; x_imag_7    =
(x_imag_7    > 8'hFF) ? 8'h00 : x_imag_7    + 1;
                x_real_8    = (x_real_8    > 8'hFF) ? 8'h00 : x_real_8    + 1; x_imag_8    =
(x_imag_8    > 8'hFF) ? 8'h00 : x_imag_8    + 4;
                x_real_9    = (x_real_9    > 8'hFF) ? 8'h00 : x_real_9    + 2; x_imag_9    =
(x_imag_9    > 8'hFF) ? 8'h00 : x_imag_9    + 3;
                x_real_10    = (x_real_10    > 8'hFF) ? 8'h00 : x_real_10    + 3; x_imag_10    =
(x_imag_10    > 8'hFF) ? 8'h00 : x_imag_10    + 2;
                x_real_11    = (x_real_11    > 8'hFF) ? 8'h00 : x_real_11    + 4; x_imag_11    =
(x_imag_11    > 8'hFF) ? 8'h00 : x_imag_11    + 1;
                x_real_12    = (x_real_12    > 8'hFF) ? 8'h00 : x_real_12    + 1; x_imag_12    =
(x_imag_12  > 8'hFF) ? 8'h00 : x_imag_12  + 4;
            x_real_13  = (x_real_13  > 8'hFF) ? 8'h00 : x_real_13  + 2; x_imag_13  =
(x_imag_13  > 8'hFF) ? 8'h00 : x_imag_13  + 3;
            x_real_14  = (x_real_14  > 8'hFF) ? 8'h00 : x_real_14  + 3; x_imag_14  =
(x_imag_14  > 8'hFF) ? 8'h00 : x_imag_14  + 2;
            x_real_15  = (x_real_15  > 8'hFF) ? 8'h00 : x_real_15  + 4; x_imag_15  =
(x_imag_15  > 8'hFF) ? 8'h00 : x_imag_15  + 1;
            x_real_16  = (x_real_16  > 8'hFF) ? 8'h00 : x_real_16  + 1; x_imag_16  =
(x_imag_16  > 8'hFF) ? 8'h00 : x_imag_16  + 4;
            x_real_17  = (x_real_17  > 8'hFF) ? 8'h00 : x_real_17  + 2; x_imag_17  =
(x_imag_17  > 8'hFF) ? 8'h00 : x_imag_17  + 3;
            x_real_18  = (x_real_18  > 8'hFF) ? 8'h00 : x_real_18  + 3; x_imag_18  =
(x_imag_18  > 8'hFF) ? 8'h00 : x_imag_18  + 2;
            x_real_19  = (x_real_19  > 8'hFF) ? 8'h00 : x_real_19  + 4; x_imag_19  =
(x_imag_19  > 8'hFF) ? 8'h00 : x_imag_19  + 1;
            x_real_20  = (x_real_20  > 8'hFF) ? 8'h00 : x_real_20  + 1; x_imag_20  =
(x_imag_20  > 8'hFF) ? 8'h00 : x_imag_20  + 4;
            x_real_21  = (x_real_21  > 8'hFF) ? 8'h00 : x_real_21  + 2; x_imag_21  =
(x_imag_21  > 8'hFF) ? 8'h00 : x_imag_21  + 3;
            x_real_22  = (x_real_22  > 8'hFF) ? 8'h00 : x_real_22  + 3; x_imag_22  =
(x_imag_22  > 8'hFF) ? 8'h00 : x_imag_22  + 2;
            x_real_23  = (x_real_23  > 8'hFF) ? 8'h00 : x_real_23  + 4; x_imag_23  =
(x_imag_23  > 8'hFF) ? 8'h00 : x_imag_23  + 1;
            x_real_24  = (x_real_24  > 8'hFF) ? 8'h00 : x_real_24  + 1; x_imag_24  =
(x_imag_24  > 8'hFF) ? 8'h00 : x_imag_24  + 4;
            x_real_25  = (x_real_25  > 8'hFF) ? 8'h00 : x_real_25  + 2; x_imag_25  =
(x_imag_25  > 8'hFF) ? 8'h00 : x_imag_25  + 3;
            x_real_26  = (x_real_26  > 8'hFF) ? 8'h00 : x_real_26  + 3; x_imag_26  =
(x_imag_26  > 8'hFF) ? 8'h00 : x_imag_26  + 2;
            x_real_27  = (x_real_27  > 8'hFF) ? 8'h00 : x_real_27  + 4; x_imag_27  =
(x_imag_27  > 8'hFF) ? 8'h00 : x_imag_27  + 1;
            x_real_28  = (x_real_28  > 8'hFF) ? 8'h00 : x_real_28  + 1; x_imag_28  =
(x_imag_28  > 8'hFF) ? 8'h00 : x_imag_28  + 4;
            x_real_29  = (x_real_29  > 8'hFF) ? 8'h00 : x_real_29  + 2; x_imag_29  =
(x_imag_29  > 8'hFF) ? 8'h00 : x_imag_29  + 3;
            x_real_30  = (x_real_30  > 8'hFF) ? 8'h00 : x_real_30  + 3; x_imag_30  =
(x_imag_30  > 8'hFF) ? 8'h00 : x_imag_30  + 2;
            x_real_31  = (x_real_31  > 8'hFF) ? 8'h00 : x_real_31  + 4; x_imag_31  =
(x_imag_31  > 8'hFF) ? 8'h00 : x_imag_31  + 1;
            x_real_32  = (x_real_32  > 8'hFF) ? 8'h00 : x_real_32  + 1; x_imag_32  =
(x_imag_32  > 8'hFF) ? 8'h00 : x_imag_32  + 4;
            x_real_33  = (x_real_33  > 8'hFF) ? 8'h00 : x_real_33  + 2; x_imag_33  =
(x_imag_33  > 8'hFF) ? 8'h00 : x_imag_33  + 3;
            x_real_34  = (x_real_34  > 8'hFF) ? 8'h00 : x_real_34  + 3; x_imag_34  =
(x_imag_34 > 8'hFF) ? 8'h00 : x_imag_34 + 2;
            x_real_35 = (x_real_35 > 8'hFF) ? 8'h00 : x_real_35 + 4; x_imag_35 =
(x_imag_35 > 8'hFF) ? 8'h00 : x_imag_35 + 1;
            x_real_36 = (x_real_36 > 8'hFF) ? 8'h00 : x_real_36 + 1; x_imag_36 =
(x_imag_36 > 8'hFF) ? 8'h00 : x_imag_36 + 4;
            x_real_37 = (x_real_37 > 8'hFF) ? 8'h00 : x_real_37 + 2; x_imag_37 =
(x_imag_37 > 8'hFF) ? 8'h00 : x_imag_37 + 3;
            x_real_38 = (x_real_38 > 8'hFF) ? 8'h00 : x_real_38 + 3; x_imag_38 =
(x_imag_38 > 8'hFF) ? 8'h00 : x_imag_38 + 2;
            x_real_39 = (x_real_39 > 8'hFF) ? 8'h00 : x_real_39 + 4; x_imag_39 =
(x_imag_39 > 8'hFF) ? 8'h00 : x_imag_39 + 1;
            x_real_40 = (x_real_40 > 8'hFF) ? 8'h00 : x_real_40 + 1; x_imag_40 =
(x_imag_40 > 8'hFF) ? 8'h00 : x_imag_40 + 4;
            x_real_41 = (x_real_41 > 8'hFF) ? 8'h00 : x_real_41 + 2; x_imag_41 =
(x_imag_41 > 8'hFF) ? 8'h00 : x_imag_41 + 3;
            x_real_42 = (x_real_42 > 8'hFF) ? 8'h00 : x_real_42 + 3; x_imag_42 =
(x_imag_42 > 8'hFF) ? 8'h00 : x_imag_42 + 2;
            x_real_43 = (x_real_43 > 8'hFF) ? 8'h00 : x_real_43 + 4; x_imag_43 =
(x_imag_43 > 8'hFF) ? 8'h00 : x_imag_43 + 1;
            x_real_44 = (x_real_44 > 8'hFF) ? 8'h00 : x_real_44 + 1; x_imag_44 =
(x_imag_44 > 8'hFF) ? 8'h00 : x_imag_44 + 4;
            x_real_45 = (x_real_45 > 8'hFF) ? 8'h00 : x_real_45 + 2; x_imag_45 =
(x_imag_45 > 8'hFF) ? 8'h00 : x_imag_45 + 3;
            x_real_46 = (x_real_46 > 8'hFF) ? 8'h00 : x_real_46 + 3; x_imag_46 =
(x_imag_46 > 8'hFF) ? 8'h00 : x_imag_46 + 2;
            x_real_47 = (x_real_47 > 8'hFF) ? 8'h00 : x_real_47 + 4; x_imag_47 =
(x_imag_47 > 8'hFF) ? 8'h00 : x_imag_47 + 1;
            x_real_48 = (x_real_48 > 8'hFF) ? 8'h00 : x_real_48 + 1; x_imag_48 =
(x_imag_48 > 8'hFF) ? 8'h00 : x_imag_48 + 4;
            x_real_49 = (x_real_49 > 8'hFF) ? 8'h00 : x_real_49 + 2; x_imag_49 =
(x_imag_49 > 8'hFF) ? 8'h00 : x_imag_49 + 3;
            x_real_50 = (x_real_50 > 8'hFF) ? 8'h00 : x_real_50 + 3; x_imag_50 =
(x_imag_50 > 8'hFF) ? 8'h00 : x_imag_50 + 2;
            x_real_51 = (x_real_51 > 8'hFF) ? 8'h00 : x_real_51 + 4; x_imag_51 =
(x_imag_51 > 8'hFF) ? 8'h00 : x_imag_51 + 1;
            x_real_52 = (x_real_52 > 8'hFF) ? 8'h00 : x_real_52 + 1; x_imag_52 =
(x_imag_52 > 8'hFF) ? 8'h00 : x_imag_52 + 4;
            x_real_53 = (x_real_53 > 8'hFF) ? 8'h00 : x_real_53 + 2; x_imag_53 =
(x_imag_53 > 8'hFF) ? 8'h00 : x_imag_53 + 3;
            x_real_54 = (x_real_54 > 8'hFF) ? 8'h00 : x_real_54 + 3; x_imag_54 =
(x_imag_54 > 8'hFF) ? 8'h00 : x_imag_54 + 2;
            x_real_55 = (x_real_55 > 8'hFF) ? 8'h00 : x_real_55 + 4; x_imag_55 =
(x_imag_55 > 8'hFF) ? 8'h00 : x_imag_55 + 1;
            x_real_56 = (x_real_56 > 8'hFF) ? 8'h00 : x_real_56 + 1; x_imag_56 =
(x_imag_56   > 8'hFF) ? 8'h00 : x_imag_56   + 4;
              x_real_57   = (x_real_57   > 8'hFF) ? 8'h00 : x_real_57   + 2; x_imag_57   =
(x_imag_57   > 8'hFF) ? 8'h00 : x_imag_57   + 3;
              x_real_58   = (x_real_58   > 8'hFF) ? 8'h00 : x_real_58   + 3; x_imag_58   =
(x_imag_58   > 8'hFF) ? 8'h00 : x_imag_58   + 2;
              x_real_59   = (x_real_59   > 8'hFF) ? 8'h00 : x_real_59   + 4; x_imag_59   =
(x_imag_59   > 8'hFF) ? 8'h00 : x_imag_59   + 1;
              x_real_60   = (x_real_60   > 8'hFF) ? 8'h00 : x_real_60   + 1; x_imag_60   =
(x_imag_60   > 8'hFF) ? 8'h00 : x_imag_60   + 4;
              x_real_61   = (x_real_61   > 8'hFF) ? 8'h00 : x_real_61   + 2; x_imag_61   =
(x_imag_61   > 8'hFF) ? 8'h00 : x_imag_61   + 3;
              x_real_62   = (x_real_62   > 8'hFF) ? 8'h00 : x_real_62   + 3; x_imag_62   =
(x_imag_62   > 8'hFF) ? 8'h00 : x_imag_62   + 2;
              x_real_63   = (x_real_63   > 8'hFF) ? 8'h00 : x_real_63   + 4; x_imag_63   =
(x_imag_63   > 8'hFF) ? 8'h00 : x_imag_63   + 1;
         end
         */
 end

endmodule
```

附录六、matlab 生成因子虚部和实部的 code

```
Wnr_real_dec = []; Wnr_real_or = [];
Wnr_imag_dec = []; Wnr_imag_or = [];
Wnr_real_hex = {};
Wnr_imag_hex = {};
for r = 0:31
    Wnr_factor   = cos(pi/32*r) - 1j*sin(pi/32*r) ;
    Wnr_integer = floor(Wnr_factor * 2^13) ;

    if (real(Wnr_integer)<0)
        Wnr_real     = real(Wnr_integer) + 2^16 ; %负数的补码
    else
        Wnr_real     = real(Wnr_integer) ;
    end
    if (imag(Wnr_integer)<0)
        Wnr_imag     = imag(Wnr_integer) + 2^16 ;
    else
        Wnr_imag     = imag(Wnr_integer);
    end
    Wnr_real_or = [Wnr_real_or real(Wnr_factor)];
    Wnr_imag_or = [Wnr_imag_or imag(Wnr_factor)];
    Wnr_real_dec = [Wnr_real_dec real(Wnr_integer)];
    Wnr_imag_dec = [Wnr_imag_dec imag(Wnr_integer)];
    Wnr_real_hex{r+1}   =   dec2hex(Wnr_real,4);    %实部
    Wnr_imag_hex{r+1}   =   dec2hex(Wnr_imag,4);     %虚部
end
%% 打印

fid=fopen('D:\test_sin_cos.txt','wt'); %写的方式打开文件（若不存在，建立文件）；
for i = 1:32
   fprintf(fid,'%s',' assign fac_real[ ');
   fprintf(fid,'%d ',i-1);
   fprintf(fid,'%s',']=');
   fprintf(fid,'%s','16''h');
   fprintf(fid,'%s ',Wnr_real_hex{i});   % %d 表示以整数形式写入数据，这正是我想要的；
   fprintf(fid,'%s',';');
   %fprintf(fid,'\n');
   fprintf(fid,'%s',' assign fac_imag[ ');
   fprintf(fid,'%d ',i-1);
   fprintf(fid,'%s',']=');
   fprintf(fid,'%s','16''h');
   fprintf(fid,'%s ',Wnr_imag_hex{i});   % %d 表示以整数形式写入数据，这正是我想要的；
   fprintf(fid,'%s',';');
   fprintf(fid,'\n');

end
fclose(fid);
```

附录七、matlab 生成倒置输入样本序号的 code

```
clear on;
clear off;
stringbox = [];
for i = 0:1:63
    stringbox = [stringbox ; dec2bin(i,6)];
end
box_inverse =[];
for i = 1:1:64
    tmp = stringbox(i,:);
    tmp_inverse = tmp(end:-1:1);
    box_inverse = [box_inverse bin2dec(tmp_inverse)];
end

fid=fopen('D:\test.txt','wt'); %写的方式打开文件（若不存在，建立文件）；
for i = 1:1:64
   fprintf(fid,'%s',' assign x_real_all [ 0 ][ ');
   fprintf(fid,'%d',i-1);
   fprintf(fid,'%s',' ] = {{12{x_real_');
   fprintf(fid,'%d ',box_inverse(i));   % %d 表示以整数形式写入数据
   fprintf(fid,'%s','[ 7 ]}},x_real_');
```

```
    fprintf(fid,'%d ',box_inverse(i));
    fprintf(fid,'%s','};    ');
    %fprintf(fid,'\n');
    fprintf(fid,'%s',' assign x_imag_all [ 0 ][ ');
    fprintf(fid,'%d',i-1);
    fprintf(fid,'%s',' ] = {{12{x_imag_');
    fprintf(fid,'%d ',box_inverse(i));   % %d  表示以整数形式写入数据
    fprintf(fid,'%s','[ 7 ]}},x_imag_');
    fprintf(fid,'%d ',box_inverse(i));
    fprintf(fid,'%s','};    ');
    fprintf(fid,'\n');
end
fclose(fid);
```

附录八、matlab 生成 6 级流水线 module 调用的 code

```
fid=fopen('D:\test_generate.txt','wt'); %写的方式打开文件（若不存在，建立文件）;
%{
for i = 0:1:31
    fprintf(fid,'%s','butterfly my0_');
    fprintf(fid,'%d',i);
    fprintf(fid,'%s','(clk,rstn,x_real_all[0][');
    fprintf(fid,'%d',2*i);
    fprintf(fid,'%s','],x_imag_all[0][');
    fprintf(fid,'%d',2*i);
    fprintf(fid,'%s','],x_real_all[0][');
    fprintf(fid,'%d',2*i+1);
    fprintf(fid,'%s','],x_imag_all[0][');
    fprintf(fid,'%d',2*i+1);
    fprintf(fid,'%s','],fac_real[0],fac_imag[0],signal_box[');
    fprintf(fid,'%d',i);
    fprintf(fid,'%s','],x_real_all[1][');
    fprintf(fid,'%d',2*i);
    fprintf(fid,'%s','],x_imag_all[1][');
    fprintf(fid,'%d',2*i);
    fprintf(fid,'%s','],x_real_all[1][');
    fprintf(fid,'%d',2*i+1);
    fprintf(fid,'%s','],x_imag_all[1][');
    fprintf(fid,'%d',2*i+1);
    fprintf(fid,'%s','],signal_box[32+');
    fprintf(fid,'%d',i);%;]');
    fprintf(fid,'%s',']); ');
    fprintf(fid,'\n');
end
%}
%{
```

```
for i=0:15
    fprintf(fid,'%s','butterfly my1_');
    fprintf(fid,'%d',2*i);
    fprintf(fid,'%s','(clk,rstn,x_real_all[1][');
    fprintf(fid,'%d',4*i);
    fprintf(fid,'%s','],x_imag_all[1][');
    fprintf(fid,'%d',4*i);
    fprintf(fid,'%s','],x_real_all[1][');
    fprintf(fid,'%d',4*i+2);
    fprintf(fid,'%s','],x_imag_all[1][');
    fprintf(fid,'%d',4*i+2);
    fprintf(fid,'%s','],fac_real[0],fac_imag[0],signal_box[32+');
    fprintf(fid,'%d',2*i);
    fprintf(fid,'%s','],x_real_all[2][');
    fprintf(fid,'%d',4*i);
    fprintf(fid,'%s','],x_imag_all[2][');
    fprintf(fid,'%d',4*i);
    fprintf(fid,'%s','],x_real_all[2][');
    fprintf(fid,'%d',4*i+2);
    fprintf(fid,'%s','],x_imag_all[2][');
    fprintf(fid,'%d',4*i+2);
    fprintf(fid,'%s','],signal_box[2*32+');
    fprintf(fid,'%d',2*i);%;]');
    fprintf(fid,'%s',']); ');
    fprintf(fid,'\n');

    fprintf(fid,'%s','butterfly my1_');
    fprintf(fid,'%d',2*i+1);
    fprintf(fid,'%s','(clk,rstn,x_real_all[1][');
    fprintf(fid,'%d',4*i+1);
    fprintf(fid,'%s','],x_imag_all[1][');
    fprintf(fid,'%d',4*i+1);
    fprintf(fid,'%s','],x_real_all[1][');
    fprintf(fid,'%d',4*i+1+2);
    fprintf(fid,'%s','],x_imag_all[1][');
    fprintf(fid,'%d',4*i+1+2);
    fprintf(fid,'%s','],fac_real[16],fac_imag[16],signal_box[32+');
    fprintf(fid,'%d',2*i+1);
    fprintf(fid,'%s','],x_real_all[2][');
    fprintf(fid,'%d',4*i+1);
    fprintf(fid,'%s','],x_imag_all[2][');
    fprintf(fid,'%d',4*i+1);
    fprintf(fid,'%s','],x_real_all[2][');
    fprintf(fid,'%d',4*i+2+1);
    fprintf(fid,'%s','],x_imag_all[2][');
    fprintf(fid,'%d',4*i+2+1);
    fprintf(fid,'%s','],signal_box[2*32+');
    fprintf(fid,'%d',2*i+1);%;]');
    fprintf(fid,'%s','];  ');
    fprintf(fid,'\n');
end
%}
%{
for i=0:7
    fprintf(fid,'%s','butterfly
my2_');fprintf(fid,'%d',4*i);fprintf(fid,'%s','(clk,rstn,x_real_all[2][');fprintf(fid,'%d',8*i);

fprintf(fid,'%s','],x_imag_all[2][');fprintf(fid,'%d',8*i);fprintf(fid,'%s','],x_real_all[2][');fprintf(fid,'%d',8
*i+4);

fprintf(fid,'%s','],x_imag_all[2][');fprintf(fid,'%d',8*i+4);fprintf(fid,'%s','],fac_real[0],fac_imag[0],sig
nal_box[2*32+');

fprintf(fid,'%d',4*i);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%d',8*i);fprintf(fid,'%s','],x_imag_all[3
][');

fprintf(fid,'%d',8*i);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%d',8*i+4);fprintf(fid,'%s','],x_imag_al
l[3][');
    fprintf(fid,'%d',8*i+4);  fprintf(fid,'%s','],signal_box[3*32+');fprintf(fid,'%d',4*i);fprintf(fid,'%s','];
');fprintf(fid,'\n');

    fprintf(fid,'%s','butterfly
my2_');fprintf(fid,'%d',4*i+1);fprintf(fid,'%s','(clk,rstn,x_real_all[2][');fprintf(fid,'%d',8*i+1);

fprintf(fid,'%s','],x_imag_all[2][');fprintf(fid,'%d',8*i+1);fprintf(fid,'%s','],x_real_all[2][');fprintf(fid,'%
d',8*i+1+4);

fprintf(fid,'%s','],x_imag_all[2][');fprintf(fid,'%d',8*i+1+4);fprintf(fid,'%s','],fac_real[8],fac_imag[8],
signal_box[2*32+');

fprintf(fid,'%d',4*i+1);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%d',8*i+1);fprintf(fid,'%s','],x_imag
_all[3][');

fprintf(fid,'%d',8*i+1);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%d',8*i+1+4);fprintf(fid,'%s','],x_i
mag_all[3][');
    fprintf(fid,'%d',8*i+1+4);
fprintf(fid,'%s','],signal_box[3*32+');fprintf(fid,'%d',4*i+1);fprintf(fid,'%s','];  ');fprintf(fid,'\n');
```

```
    fprintf(fid,'%s','butterfly
my2_');fprintf(fid,'%d',4*i+2);fprintf(fid,'%s','(clk,rstn,x_real_all[2][');fprintf(fid,'%d',8*i+2);

fprintf(fid,'%s','],x_imag_all[2][');fprintf(fid,'%d',8*i+2);fprintf(fid,'%s','],x_real_all[2][');fprintf(fid,'%
d',8*i+2+4);

fprintf(fid,'%s','],x_imag_all[2][');fprintf(fid,'%d',8*i+2+4);fprintf(fid,'%s','],fac_real[16],fac_imag[1
6],signal_box[2*32+');

fprintf(fid,'%d',4*i+2);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%d',8*i+2);fprintf(fid,'%s','],x_imag
_all[3][');

fprintf(fid,'%d',8*i+2);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%d',8*i+2+4);fprintf(fid,'%s','],x_i
mag_all[3][');
    fprintf(fid,'%d',8*i+2+4);
fprintf(fid,'%s','],signal_box[3*32+');fprintf(fid,'%d',4*i+2);fprintf(fid,'%s',']); ');fprintf(fid,'\n');

    fprintf(fid,'%s','butterfly
my2_');fprintf(fid,'%d',4*i+3);fprintf(fid,'%s','(clk,rstn,x_real_all[2][');fprintf(fid,'%d',8*i+3);

fprintf(fid,'%s','],x_imag_all[2][');fprintf(fid,'%d',8*i+3);fprintf(fid,'%s','],x_real_all[2][');fprintf(fid,'%
d',8*i+3+4);

fprintf(fid,'%s','],x_imag_all[2][');fprintf(fid,'%d',8*i+3+4);fprintf(fid,'%s','],fac_real[24],fac_imag[2
4],signal_box[2*32+');

fprintf(fid,'%d',4*i+3);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%d',8*i+3);fprintf(fid,'%s','],x_imag
_all[3][');

fprintf(fid,'%d',8*i+3);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%d',8*i+3+4);fprintf(fid,'%s','],x_i
mag_all[3][');
    fprintf(fid,'%d',8*i+3+4);
fprintf(fid,'%s','],signal_box[3*32+');fprintf(fid,'%d',4*i+3);fprintf(fid,'%s',']); ');fprintf(fid,'\n');

end
%}
%{
for i=0:3
        for j = 0:7
            fprintf(fid,'%s','butterfly
my3_');fprintf(fid,'%d',8*i+j);fprintf(fid,'%s','(clk,rstn,x_real_all[3][');fprintf(fid,'%d',16*i+j);

fprintf(fid,'%s','],x_imag_all[3][');fprintf(fid,'%d',16*i+j);fprintf(fid,'%s','],x_real_all[3][');fprintf(fid,'%
d',16*i+j+8);
```

```
        fprintf(fid,'%s','],x_imag_all[3][');fprintf(fid,'%d',16*i+j+8);
        if j==0
            fprintf(fid,'%s','],fac_real[0],fac_imag[0],signal_box[3*32+');
        elseif j==1
            fprintf(fid,'%s','],fac_real[4],fac_imag[4],signal_box[3*32+');
        elseif j==2
            fprintf(fid,'%s','],fac_real[8],fac_imag[8],signal_box[3*32+');
        elseif j==3
            fprintf(fid,'%s','],fac_real[12],fac_imag[12],signal_box[3*32+');
        elseif j==4
            fprintf(fid,'%s','],fac_real[16],fac_imag[16],signal_box[3*32+');
        elseif j==5
            fprintf(fid,'%s','],fac_real[20],fac_imag[20],signal_box[3*32+');
        elseif j==6
            fprintf(fid,'%s','],fac_real[24],fac_imag[24],signal_box[3*32+');
        else
            fprintf(fid,'%s','],fac_real[28],fac_imag[28],signal_box[3*32+');
        end

fprintf(fid,'%d',8*i+j);fprintf(fid,'%s','],x_real_all[4][');fprintf(fid,'%d',16*i+j);fprintf(fid,'%s','],x_imag_all[4][');

fprintf(fid,'%d',16*i+j);fprintf(fid,'%s','],x_real_all[4][');fprintf(fid,'%d',16*i+j+8);fprintf(fid,'%s','],x_imag_all[4][');
        fprintf(fid,'%d',16*i+j+8);
fprintf(fid,'%s','],signal_box[4*32+');fprintf(fid,'%d',8*i+j);fprintf(fid,'%s',']); ');fprintf(fid,'\n');
    end
end
%}
%{
for i=0:1
    for j = 0:15
        fprintf(fid,'%s','butterfly my4_');fprintf(fid,'%d',32*i+j);fprintf(fid,'%s','(clk,rstn,x_real_all[4][');fprintf(fid,'%d',32*i+j);

fprintf(fid,'%s','],x_imag_all[4][');fprintf(fid,'%d',32*i+j);fprintf(fid,'%s','],x_real_all[4][');fprintf(fid,'%d',32*i+j+16);
        fprintf(fid,'%s','],x_imag_all[4][');fprintf(fid,'%d',32*i+j+16);
        fprintf(fid,'%s','],fac_real[');
        fprintf(fid,'%d',2*j);
        fprintf(fid,'%s','],fac_imag[');
        fprintf(fid,'%d',2*j);
        fprintf(fid,'%s','],signal_box[4*32+');
fprintf(fid,'%d',16*i+j);fprintf(fid,'%s','],x_real_all[5][');fprintf(fid,'%d',32*i+j);fprintf(fid,'%s','],x_ima
g_all[5][');

fprintf(fid,'%d',32*i+j);fprintf(fid,'%s','],x_real_all[5][');fprintf(fid,'%d',32*i+j+16);fprintf(fid,'%s','],x
_imag_all[5][');
        fprintf(fid,'%d',32*i+j+16);
fprintf(fid,'%s','],signal_box[5*32+');fprintf(fid,'%d',16*i+j);fprintf(fid,'%s',']); ');fprintf(fid,'\n');
      end
end
%}
for i=0
      for j = 0:31
        fprintf(fid,'%s','butterfly
my5_');fprintf(fid,'%d',64*i+j);fprintf(fid,'%s','(clk,rstn,x_real_all[5][');fprintf(fid,'%d',64*i+j);

fprintf(fid,'%s','],x_imag_all[5][');fprintf(fid,'%d',64*i+j);fprintf(fid,'%s','],x_real_all[5][');fprintf(fid,'%
d',64*i+j+32);
        fprintf(fid,'%s','],x_imag_all[5][');fprintf(fid,'%d',64*i+j+32);
        fprintf(fid,'%s','],fac_real[');
        fprintf(fid,'%d',1*j);
        fprintf(fid,'%s','],fac_imag[');
        fprintf(fid,'%d',1*j);
        fprintf(fid,'%s','],signal_box[5*32+');

fprintf(fid,'%d',32*i+j);fprintf(fid,'%s','],x_real_all[6][');fprintf(fid,'%d',64*i+j);fprintf(fid,'%s','],x_ima
g_all[6][');

fprintf(fid,'%d',64*i+j);fprintf(fid,'%s','],x_real_all[6][');fprintf(fid,'%d',64*i+j+32);fprintf(fid,'%s','],x
_imag_all[6][');
        fprintf(fid,'%d',64*i+j+32);
fprintf(fid,'%s','],signal_box[6*32+');fprintf(fid,'%d',32*i+j);fprintf(fid,'%s',']); ');fprintf(fid,'\n');
      end
end
fclose(fid);
```

附录九、matlab 调用 fft()函数生成 64 点结果的 code

```
clear all;close all;clc;
x_real_element = [1, 2, 3, 4, 5];
x_real_part1 = repmat(x_real_element,1,12);
x_real = [x_real_part1 1 2 3 4];
x_imag_element = [1j 2j];
x_imag = repmat(x_imag_element,1,32);
x = x_real + x_imag;
step_element = [1+4j 2+3j 3+2j 4+1j];
step        = repmat(step_element,1,16);
```

```
x2 = zeros(6,64);
x2(1,:) = x;
for i =1:5
        if i==1
                x2(1+1,:) = x + step;
        else
                x2(i+1,:) = x2(i,:) + step;
        end
end
%% fft
fft1 = zeros(1,64);
fft1(1,:) = fft(x2(1,:));
fft1_all = zeros(2,64);
for i =1
        fft1_all(2*i-1,:) = real(fft1(i,:));
        fft1_all(2*i,:) = imag(fft1(i,:));
end
```