多模态FFT处理器的设计

|  |  |
| --- | --- |
| 姓名： | 刘金硕 |
| 学号： | 202428015926051 |

# 目录

[目录 1](#_Toc150846839)

[1 设计规格与要求 1](#_Toc150846840)

[2 算法原理与算法设计 1](#_Toc150846841)

[3 实现架构 2](#_Toc150846842)

[4 仿真结果 5](#_Toc150846843)

[5 综合结果 7](#_Toc150846844)

# 设计规格与要求

FFT用于快速计算离散傅立叶变换（DFT）。长为N的序列x(n)的DFT定义为：

相应的序列X(k)的IDFT定义为

这里DFT和IDFT定义均忽略前面常数因子。

设计一个多模FFT处理器时序逻辑电路，支持计算64/128/256/512点FFT和IFFT。模块整体采用流水线结构实现，能够处理连续多组输入数据。

设计要求：

* Verilog实现代码可综合，给出详细设计文档、综合以及仿真结果。
* 在每组数据处理过程中，inv和np信号值保持不变。
* 支持输入多组数据不同模式（FFT/IFFT、不同点数）的切换。
* 计算过程进行适当精度控制，保证输出结果精确度，输出定点格式（精度范围）可以根据需要进行调整，需要对计算结果进行误差分析。

# 算法原理与算法设计

**FFT算法原理**

快速傅里叶变换（FFT）利用了虚指数项的对称性等特点，对离散傅里叶变换（DFT）的计算实现了简化，从而提高计算机的求解速度，其计算结果与离散傅里叶变换（DFT）是完全一致的。

DFT变换公式为：

其中

将离散傅里叶变换中的奇数项和偶数项分离，前N/2个点可以表示为：

根据虚数单位圆的对称性，后N/2个点可以表示为

由此可见，后N/2的点的值可以通过计算前N/2的点的中间过程值确定。对和继续进行奇偶分解，就可以减少一半的计算量，通过这样分治的计算可以将DFT的算法复杂度从减小到。

# 实现架构

**模块端口说明**

**top module：**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **方向** | **位宽** | **描述** |
| clk | I | 1 | 系统时钟 |
| rst\_n | I | 1 | 系统异步复位，低电平有效 |
| inv | I | 1 | 模式控制，0表示FFT运算，1表示IFFT运算 |
| np | I | 2 | FFT/IFFT点数：  0表示64点  1表示128点  2表示256点  3表示512点 |
| valid\_in | I | 1 | 输入数据有效指示，高电平有效 |
| sop\_in | I | 1 | 每组输入数据（64个数）第一个有效数据指示，高电平有效 |
| x\_re | I | 16 | 输入数据实部，二进制补码定点格式 |
| x\_im | I | 16 | 输入数据虚部，二进制补码定点格式 |
| valid\_out | O | 1 | 输出数据有效指示，高电平有效 |
| sop\_out | O | 1 | 每组输出数据（64个数）第一个有效数据指示，高电平有效 |
| y\_re | O | 16 | 输出数据实部，二进制补码定点格式 |
| y\_im | O | 16 | 输出数据虚部，二进制补码定点格式 |

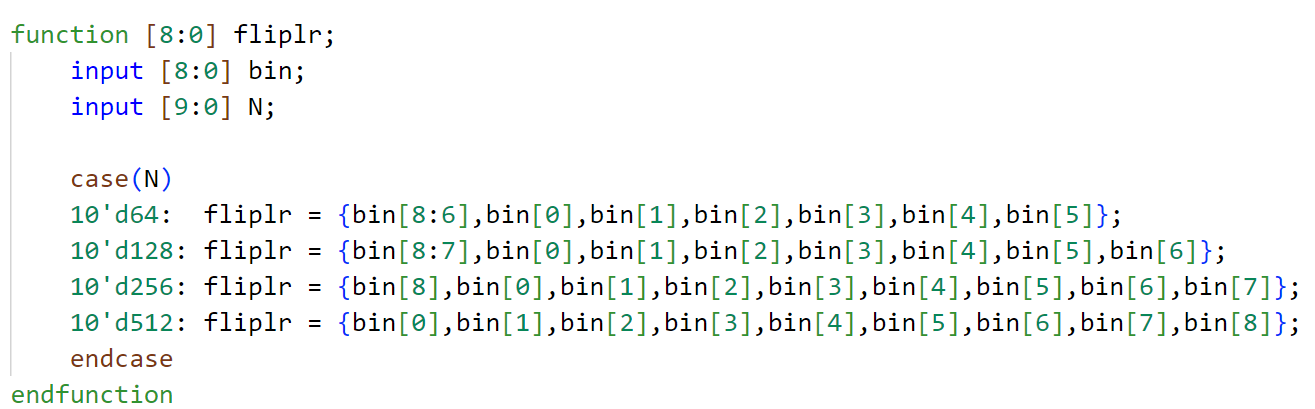
**模块架构**

本设计模块架构如下图所示，采用串行流水线方式计算FFT，即只使用一个蝶形运算单元，逐个进行FFT所需要的每一个蝶形运算。输入序列按时间抽取重新排列存入RAM中，旋转因子定点化后存入ROM。由控制器产生相应控制信号与地址信号，读取RAM中的数据以及ROM中的旋转因子送入蝶形运算单元进行计算，计算后的结果再存入RAM中。全部计算完成后输出RAM缓存区中的结果。



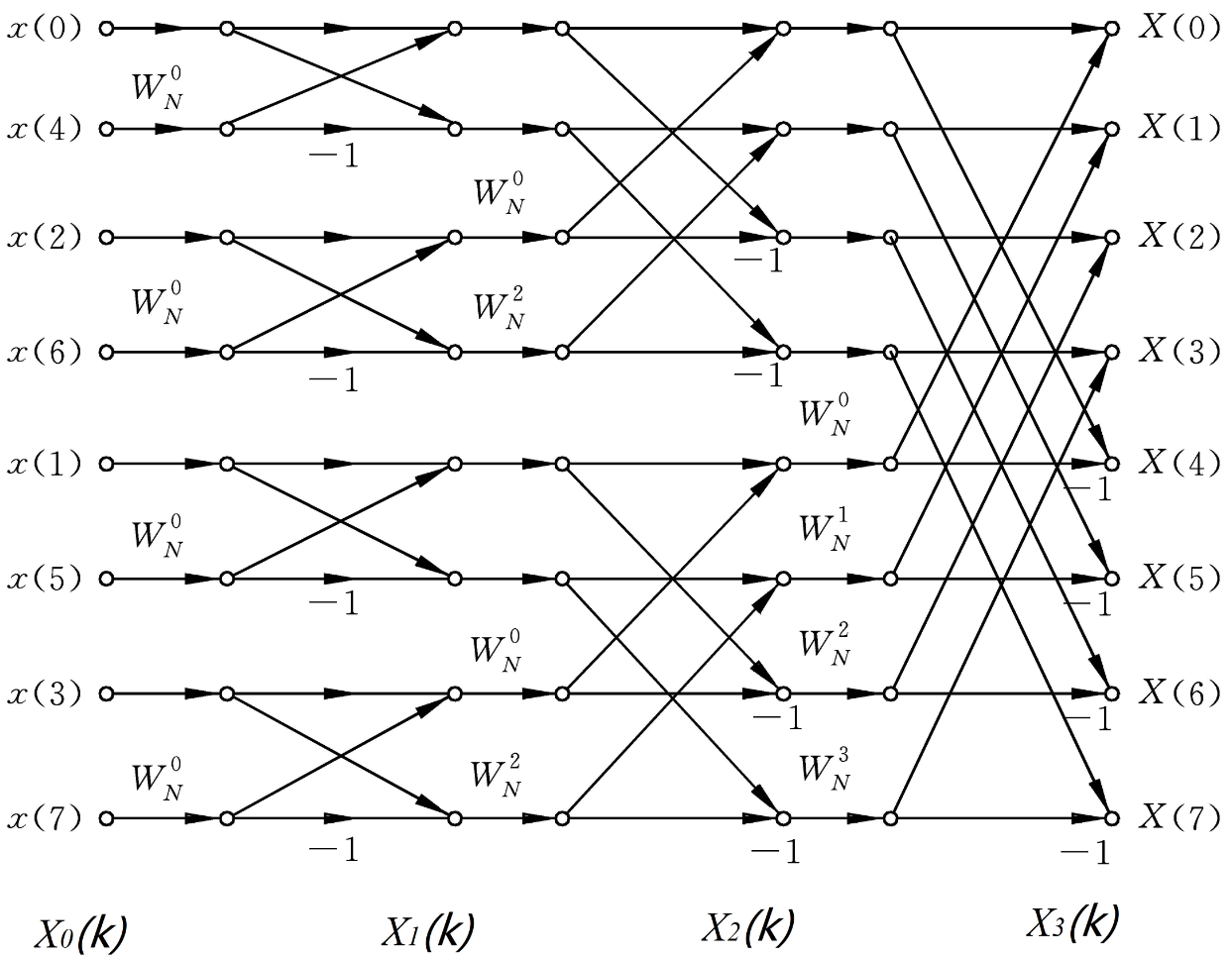
**重排序单元**

以8点FFT为例，对于 N=8 点的 FFT 计算，X(0) ~ X(7) 位置对应的 2 进制码为：X(000), X(001), X(010), X(011), X(100), X(101), X(110), X(111)。将其位置的 2 进制码进行翻转：X(000), X(100), X(010), X(110), X(001), X(101), X(011), X(111)。此时位置对应的 10 进制为：X(0), X(4), X(2), X(6), X(1), X(5), X(3), X(7)。恰好对应 FFT 第一级输入数据的顺序。因此对于输入数据，需要使用重排序单元，产生重排序后的地址存入RAM中进行缓存。产生重排序地址的核心函数如下：

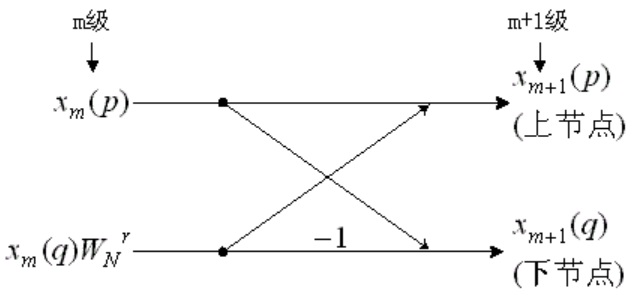


**蝶形计算单元**

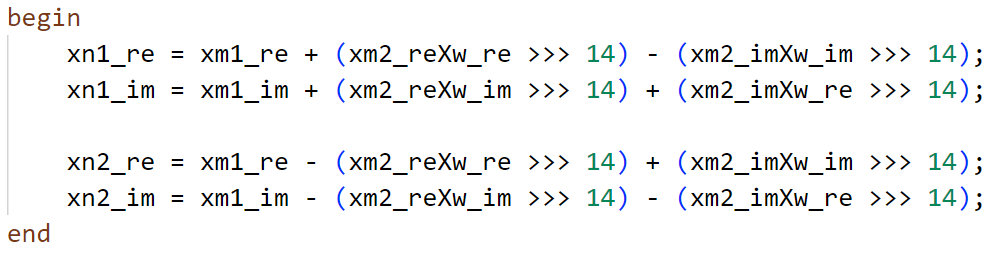
蝶形运算单元是FFT的计算核心单元。下面以8点FFT为例分析蝶形计算的原理与过程。如下图所示，8点FFT分三级计算完成，每一级的运算由若干蝶形计算单元完成，每一级均有N/2个蝶形单元。

**

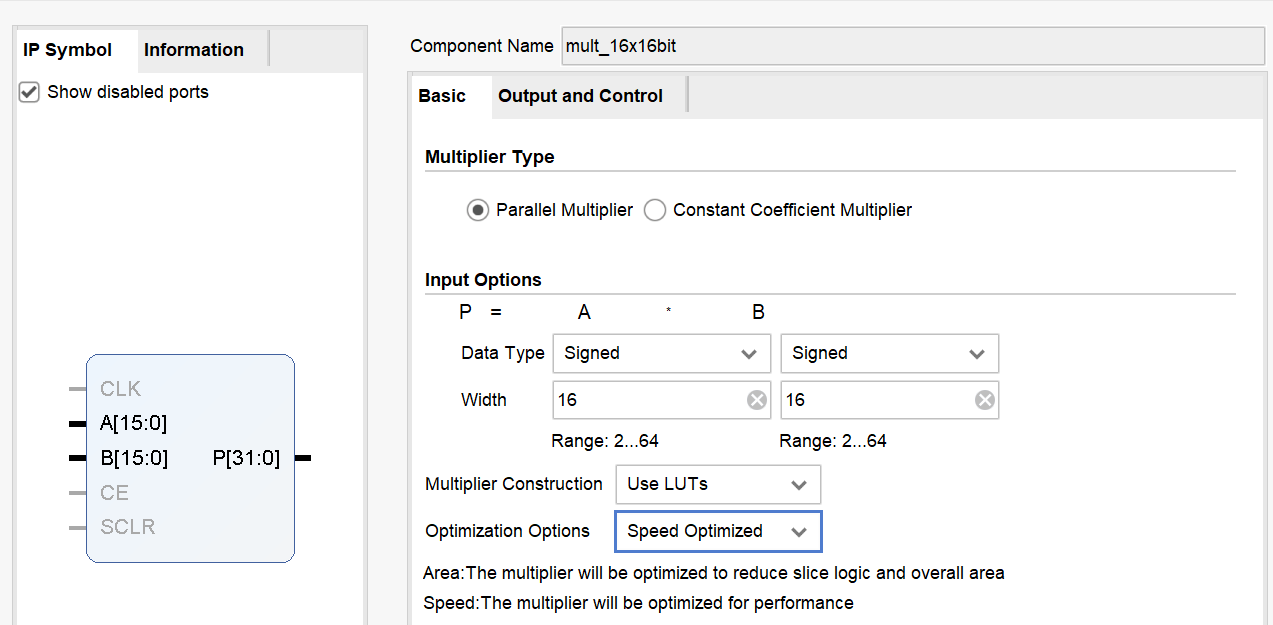
单个蝶形单元结构如下图。



蝶形运算单元输入为xm1、xm2和旋转因子w的实部与虚部，输出为xn1、xn2的实部与虚部。在verilog中实现如下图，由于旋转因子在定点化的过程中扩大了倍，因此每一次的蝶形运算时需要将与旋转因子相乘的结果右移14位，从而抵消旋转因子扩大倍数带来的影响。



与旋转因子相乘使用乘法器ip，设置如下：

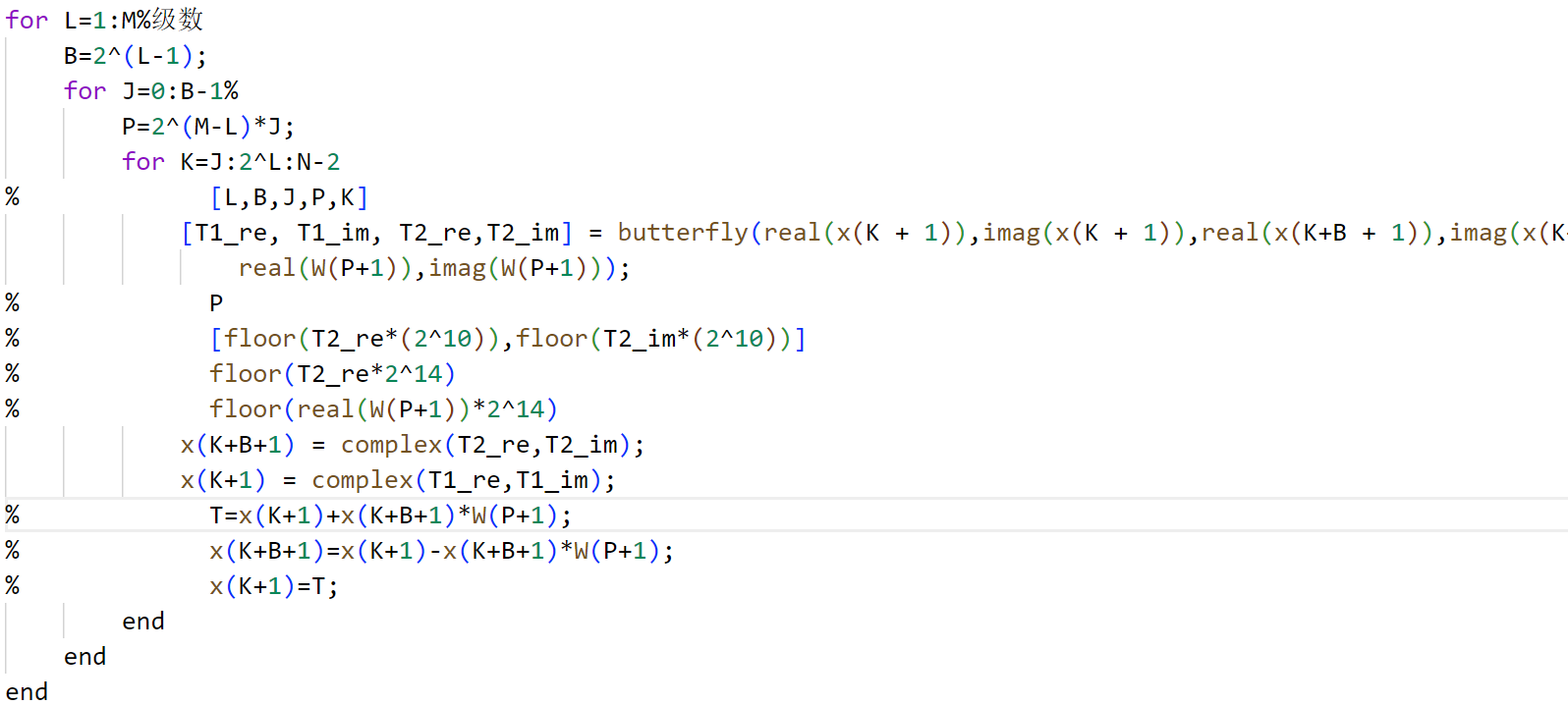


**蝶形计算控制器**

本控制单元用于产生控制信号控制蝶形计算。具体为生成蝶形运算输入端数据读取、输出端数据写入地址，并产生读写控制信号。主要的控制信号如下：

|  |  |
| --- | --- |
| 控制信号 | 描述 |
| L | 级数计数器 |
| J | 每一级下蝶形单元组数计数器 |
| P | 旋转因子计数器，用于生成旋转因子的寻址地址 |
| K | 位置计数器，蝶形运算输入端第一个输入所在的地址。 |
| B | 蝶形运算输入端两信号间隔 |

以上参数的产生算法可以参见下面的matlab代码：



**旋转因子生成与存储**

旋转因子由matlab软件产生并定点化，存储到ROM中，随后由蝶形运算控制器控制读取。本设计中，生成了512点的旋转因子w，将其扩大 512倍后取整存入ROM，完成定点化。生成旋转因子的matlab代码如下：

N = 512;

for m=0:N/2-1

W(m+1)=complex(cos(2\*pi\*m/N),-sin(2\*pi\*m/N));

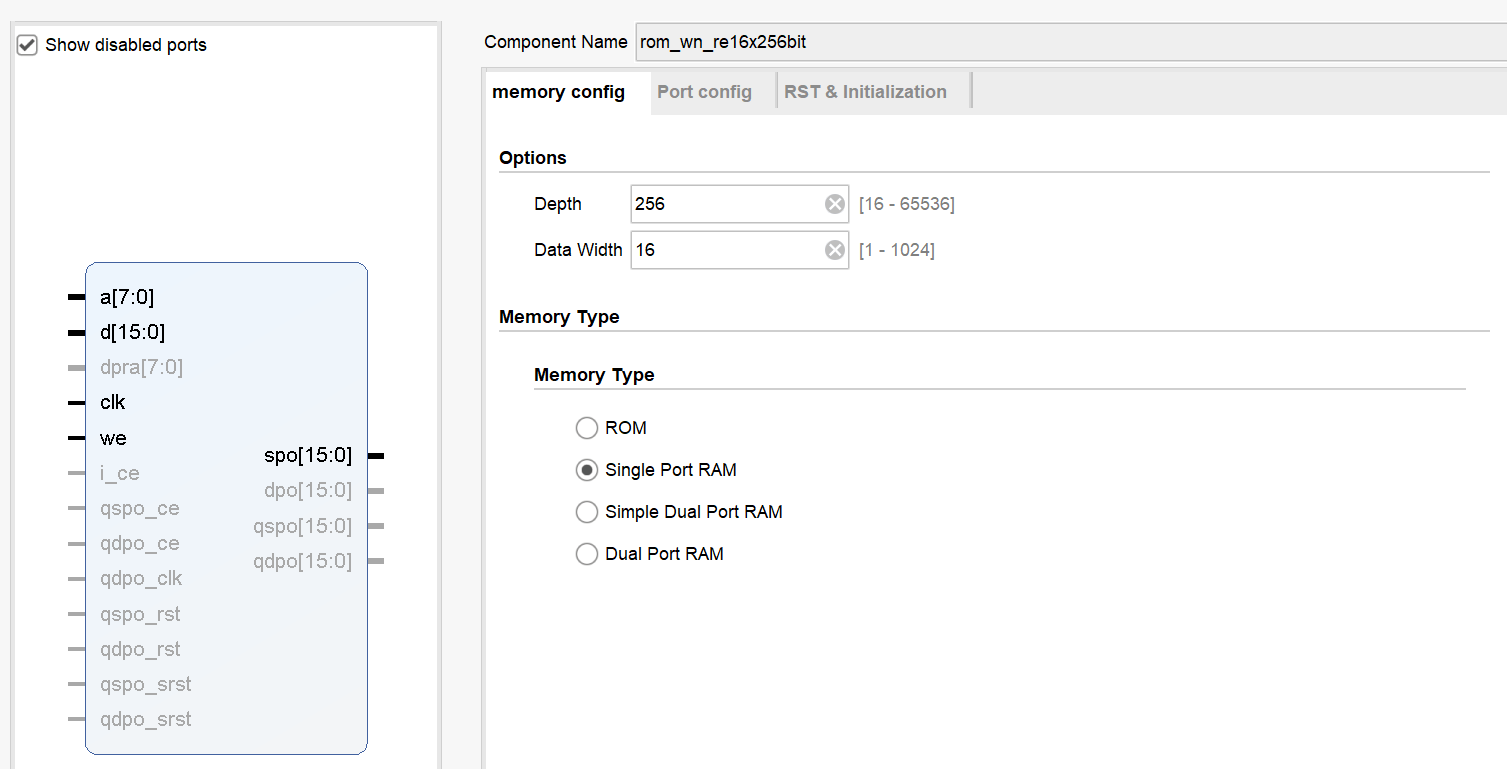
end

r = floor(real(W')\*(2^14));

im = floor(imag(W')\*(2^14));

生成512点的旋转因子同时包含了64点、128点和256点用到的所有旋转因子值。将旋转因子看作是在Z平面单位圆上等间隔的采样，那么从512点旋转因子到64点、128点和256点是一个降采样的过程，生成的512点旋转因子也包含了其他更小点数的旋转因子，因此可以满足多模态计算的要求。64点、128点、256点和512点的旋转因子在ROM中的寻址地址为计数器P的计数值的8倍、4倍、2倍和1倍。

将定点化后的旋转因子写入coe文件，初始化ROM ip核。其中，ROM配置为深度256，数据位宽16bit，无输出寄存器。

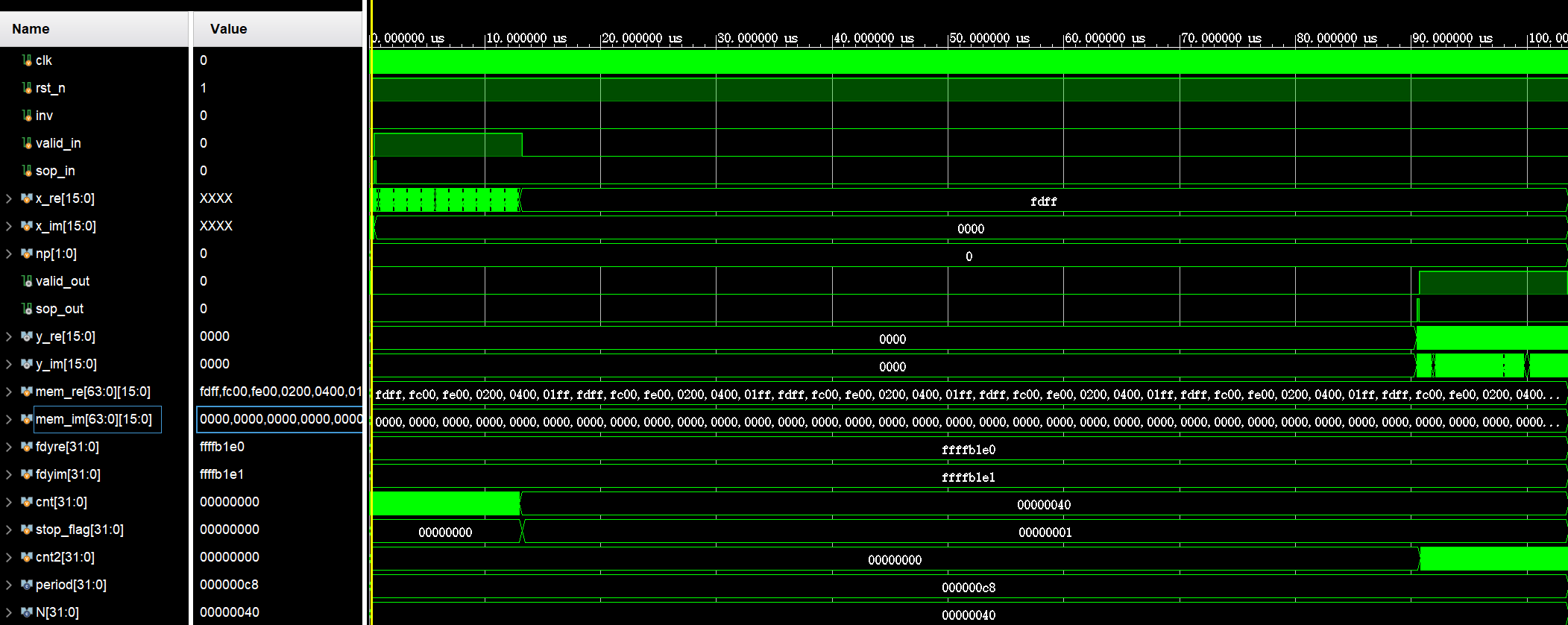


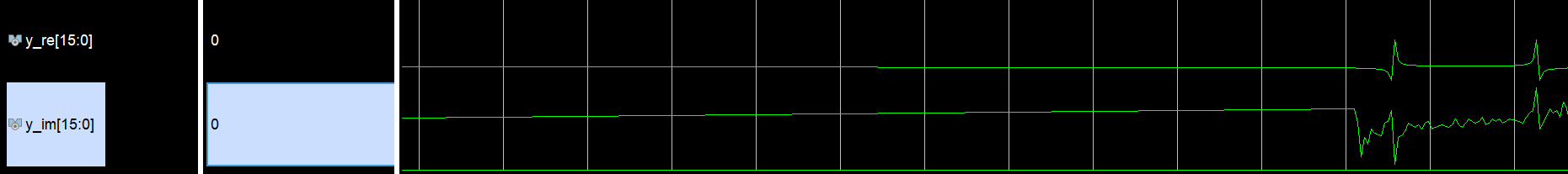
# 仿真结果

分别对64，128，256，512点进行仿真testbench中载入matlab生成的输入数据，将FFT计算结果写入文件，在matlab中进行结果比对。

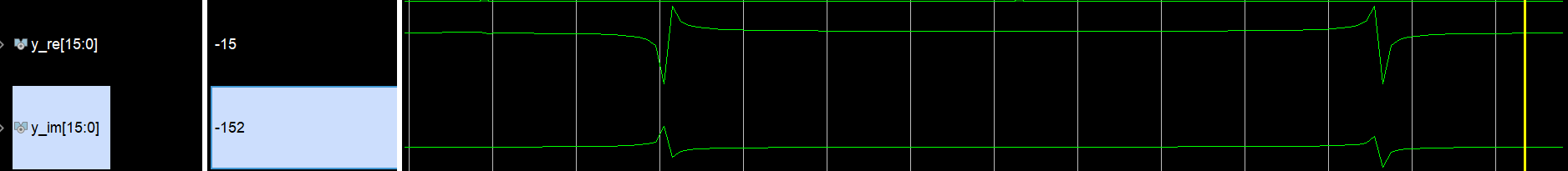
**仿真波形**

64

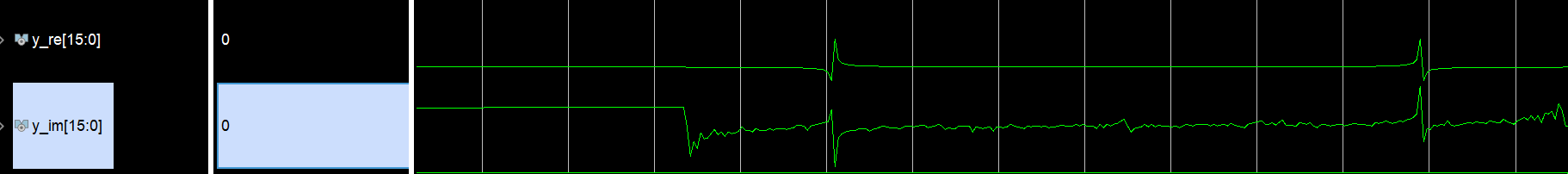




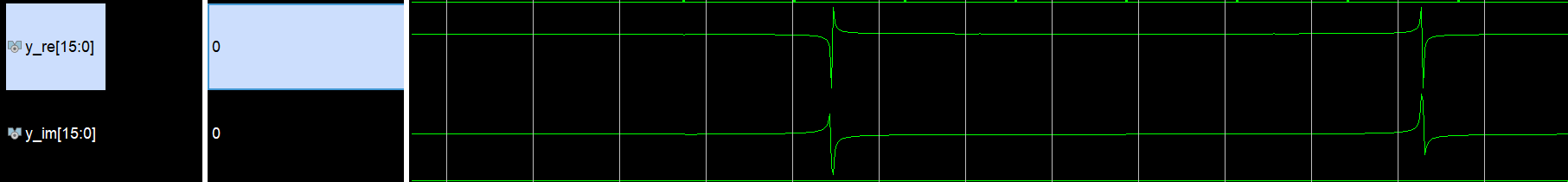
128



256



512



**MATLAB中结果验证**

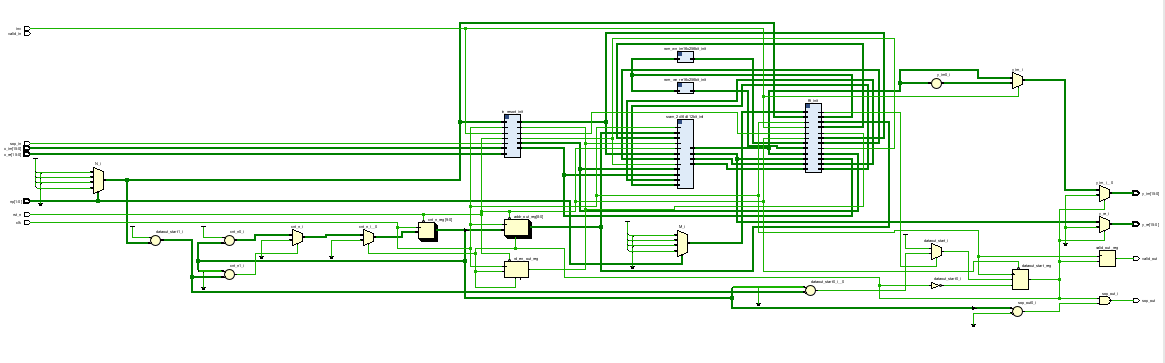
|  |  |
| --- | --- |
|  |  |
|  |  |

可以看到，FFT模块产生的数据与MATLAB算法产生的数据一致，证明模块功能的正确性。

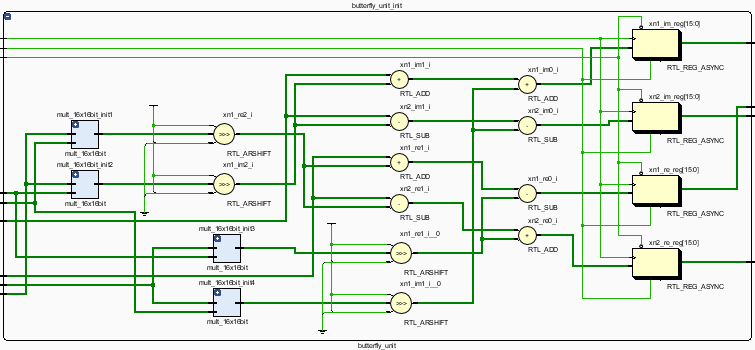
# 综合结果

**RTL SCHEMATIC**

顶层模块



蝶形运算模块



**FPGA资源使用**

