**Any Point DFT综述**

**介绍**

硬件实现任意点数DFT的算法描述以及Verilog HDL实现。  
使用一套高效简洁的设备实现任意点数的DFT，包括非2^n点的DFT和2^n点的FFT，结果完美逼近理论值。

**原理说明**

2^n点数的DFT可以通过FFT快速实现，不必赘述。  
非2^n点数DFT的实现，首先通过上采样将非2^n点数的输入数据变成合适的2^n点数的输入数据，然后可以使用2^n点FFT实现快速变换，FFT变换结果经过裁剪和频域补偿得到最终的非2^n点数的DFT变换结果。  
输入数据的上采样也是一个低通滤波过程，需要设计合适的低通滤波器使其拥有良好的通带/过渡带/阻带特性用于上采样过程，并具体实现为一个多相滤波器。由于从非2^n点数到2^n点数的上采样不是整数倍数，因此上采样过程中还要用到插值，通常简单的线性插值即可。DFT本质上是循环卷积，所以在上采样过程中非2^n点数的输入结束后需要进行一段循环重构继续进行上采样过程，而不能是输入数据结束后直接填充零。  
经过2^n点FFT变换的结果，对于非2^n点的DFT而言，多了一些频带之外的数据，这些数据丢弃即可。上采样滤波和插值过程会对FFT得到的频域数据造成变形，需要补偿消除掉。时域的卷积对应于频域的相乘，通过预先计算可以得到上采样过程的频域响应，对FFT变换的结果乘以上采样过程的频域响应的倒数来补偿上采样过程造成的频域数据的变形，从而得到接近完美的非2^n点DFT变换结果。

**设计方法说明**

设计按以下步骤进行：

1. 首先进行算法上的设计，这个过程通常使用Matlab或者Python等工具进行，本项目使用Matlab。使用Matlab设计算法是非常方便的，可以在其中变换各种参数和方法，快速得到运算结果。  
   算法设计成功的标志是按照本算法构造的任意点数的DFT计算出的结果和Matlab内置的fft函数计算出的结果一致（相对误差小于某个设定值，比如-60dB），并且该结构容易用硬件实现。  
   算法设计中的一个重要步骤是用于上采样的FIR低通滤波器的设计，这个低通滤波器需要有比较理想的频域响应特性，同时滤波器的抽头系数不能太多以便减小硬件实现难度。上采样过程的频域响应可以使用Matlab计算出来并由此得到频域补偿系数。  
   算法设计结束之后，需要将低通滤波器的抽头系数和频域补偿系数保存在文件中，以供后面的C-model设计使用。
2. C-model的设计，这一步在Matlab算法设计完成之后的基础上，将各个功能模块用C/C++语言实现，这些功能模块的结构需要和硬件实现相对应，作为Verilog RTL实现的参考模型。  
   先完成浮点C-model，其运算结果需要和Matlab模型保持一致。在浮点C-model和Matlab模型一致的基础上，对其进行定点化得到定点C-model。定点C-model的运算结果需要和浮点C-model运算结果保持一致，定点C-model将作为Verilog RTL实现的黄金参考模型。
3. Verilog HDL实现，包括RTL和Testbench, 以定点C-model为参考模型。

**使用说明**

1. 项目根目录下面有algorithm和lte\_dft两个主要目录。algorithm目录下的内容使用Matlab，探索试验可用硬件实现任意点DFT的各种算法，挑选出最优的方案；lte\_dft目录则是使用本项目所描述的方法，从Matlab算法，C-model一直到Verilog RTL，实现LTE上行链路SC-FDMA所用到的DFT。
2. algorithm目录主要内容如下:  
   algorithm  
   ├── doc  文档目录  
   └── matlab 算法源文件所在目录  
   │ ├── amp\_func.m 频域补偿幅度拟合函数  
   │ ├── ang\_func.m 频域补偿相位拟合函数  
   │ ├── comp\_factor\_fit.m 计算频域补偿的拟合系数并输出到文件  
   │ ├── cooley\_tukey coole-tukey算法的验证，用于3\*2^n点数FFT算法设计  
   │ │   ├── cooley\_tukey\_1st.m  
   │ │   ├── cooley\_tukey.m  
   │ │   └── cooley\_tukey\_main.m  
   │ ├── dft\_fit\_main.m 对频域补偿系数进行拟合对比  
   │ ├── dft\_freq\_fit.m 频域补偿拟合函数  
   │ ├── dft\_main\_lut.m 现场设计滤波器，计算任意点的DFT，频域补偿系数直接由查找表得到  
   │ ├── filter\_design.m 用于上采样的低通滤波器设计  
   │ ├── fit\_any\_point\_dft.m 使用已经设计好的滤波器，计算任意点DFT，频域补偿系数由拟合得到  
   │ ├── lut\_any\_point\_dft.m 使用已经设计好的滤波器，计算任意点DFT，频域补偿系数由查找表LUT得到  
   │ ├── make\_comp\_lut.m 生成频域补偿系数查找表  
   │ ├── preproc\_fft\_postproc.m 任意点DFT的主函数包括采样率转换, 2^n点FFT和频域数据裁剪后输出  
   │ ├── reference 参考目录，其下子目录包含了对各种情况的Matlab算法尝试  
   │ | |\_...  
   │ |  
   │ ├── sample\_rate\_conv.m 采样率转换，用于将非2^n点的输入上采样为2^n点输入  
   | ├── src\_freq\_response.m 尝试计算上采样过程的频域响应

特别提示：  
(1) 刚开始着手时，对有关使用拟合方法计算频域补偿系数的内容可以暂时忽略，重点关注preproc\_fft\_postproc.m，sample\_rate\_conv.m，以及dft\_main\_lut.m这几个文件即可。  
(2) 计算过程模拟了OFDM信号发送/接收的通信过程， 非2^n点频域数据-->使用Matlab内置的ifft函数进行IDFT变换(调制)-->得到非2^n点时域数据-->使用自己设计的DFT算法进行变换(解调)-->还原得到非2^n点频域数据，输入的频域数据和还原后的频域数据一致则说明设计的DFT算法有效。读者如果有兴趣可以直接调用matlab内置的fft函数对得到的非2^n点时域数据做DFT变换，还原输入的非2^n点频域数据，确认它们其实是一致的。  
(3) 对于数字滤波器设计而言，频率是一个相对概念，所以设计好一个上采样滤波器便可以用于所有的非2^n点DFT计算，不必每个点数都设计一个，从而使采样率转换电路的结构设计简单化。

1. lte\_dft目录主要内容如下：  
   lte\_dft  
   ├── c\_model C/C++语言参考模型目录  
   │   ├── fft\_r22 2^n点FFT参考模型目录  
   │   │   ├── cpp C/C++源代码目录  
   │   │   │   ├── fixed\_point 定点模型源代码目录  
   │   │   │   └── float\_point 浮点模型源代码目录  
   │   │   ├── data 数据保存目录  
   │   │   └── matlab 用于对比的Matlab源文件目录，用来产生输入数据并将C-Model产生的结果和Matlab内置函数产生的结果相比较  
   │   │    
     
   │   ├── lte\_dft LTE DFT参考模型目录  
   │   │   ├── cpp C/C++源代码目录  
   │   │   │   ├── fixed\_point 定点模型源代码目录  
   │   │   │   └── float\_point 浮点模型源代码目录  
   │   │   ├── data 数据保存目录  
   │   │   └── matlab 用于对比的Matlab源文件目录，用来产生输入数据并将C-Model产生的结果和Matlab DFT模型产生的结果相比较  
   │   └── reference 参考目录  
   │  
       
   ├── doc 参考文档目录  
   │  
      ├── hdl Verilog HDL设计目录  
   │   ├── rtl RTL设计目录  
   │   │   ├── fft 2^n点FFT RTL实现目录  
   │   │   ├── header 头文件目录，定义相关宏  
   │   │   ├── postproc 后处理单元目录，用于频域数据裁剪和补偿得到最终DFT变换结果  
   │   │   ├── preproc 前处理单元目录，用于采样率转换  
   │   │   └── top 顶层模块目录  
   │   │  
       
   │   ├── sim 仿真目录  
   │   │   ├── dft 用于lte DFT的仿真  
   │   │   └── fft 用于2^n点FFT的仿真  
   │   |  
   │   └── tb 测试平台目录  
   │   ├── dft 用于lte DFT的测试平台  
   │   └── fft 用于2^n点FFT的测试平台  
   │  
       
   └── matlab LTE DFT的Matlab参考模型  
     
     
   由于采用模块化的设计方法，为了搭建下层模块设计的验证环境，项目中不少同名matlab文件在不同目录中反复出现，请知晓。  
   只是对算法设计感兴趣，研究algorithm目录就可以了。  
   如果想要了解从Matlab到RTL的整个设计实现过程，则需要深入了解lte\_dft目录下的内容。

#### 免责声明

本设计可以自由使用，作者不索取任何费用。  
作者对使用结果不做任何承诺也不承担其产生的任何法律责任。  
使用者须知晓并同意上述申明，如不同意则不要使用。