**虚拟现实（VR）无线视频传输设计与实现**

摘 要

随着虚拟现实技术的发展，与之相关的应用也越来越丰富。虚拟现实的应用对无线视频传输技术提出了更高的要求。它的显著特点就是大数据量和低延时。传统数字传输经过压缩量化过程，当信噪比（SNR）低于门阀值时，会出现悬崖效应。因此，在一定的信道条件下，传统的无线视频传输方案往往很难满足虚拟现实视频数据的传输要求，同时在广播的情况下也难以使所有接收端都能获得最佳的用户体验。

SoftCast无线视频传输方案通过信源信道联合编码的伪模拟传输方式，不经过FEC和传统的调制过程，将视频的原始数据经过DCT变换和功率重分配后直接在OFDM信道上传输，避免了悬崖效应，也确保在广播的条件下每个接收端都能解码出在它们信道条件下的质量最好的视频。在与虚拟现实相关的应用下有良好的表现。

本毕业设计完成了一个基于SoftCast的虚拟现实无线视频传输系统的部分模块设计与实现。其中包括SoftCast发送端和接收端的结构以及相应的算法分析，硬件实现及测试。

**关键词：**SoftCast，无线视频传输，广播传输，功率重分配

**Design and Implementation of Wireless Video Transmission in Virtual Reality**

**ABSTRACT**

With the development of virtual reality technology, the related application is also increasingly rich. The application of virtual reality puts forward higher requirements for wireless video transmission technology. Its salient features are large data and low latency. Traditional digital transmission through the compression and quantization process, when the signal to noise ratio (SNR) below the threshold, there will be cliff effect. Therefore, under certain channel conditions, the traditional wireless video transmission scheme is often difficult to meet the low latency and high throughput requirements. In the case of broadcast transmission,it is also difficult to make all the receiver can get the best user experience.

SoftCast wireless video transmission scheme through the source channel joint encoding pseudo-analog transmission, without FEC and the traditional modulation process, the video of the original data through DCT conversion and power redistribution directly on the OFDM channel transmission. This scheme avoids the cliff effect, It is also ensured that each receiver at the broadcast conditions can decode the best quality video under their channel conditions. SoftCast has a good performance In the applications that related to virtual reality.

The graduation design completes the design and implementation of a virtual reality wireless video transmission system which based on SoftCast. Including the structure of SoftCast sender and receiver, as well as the corresponding algorithm analysis, hardware implementation and testing.

**Key words：**SoftCast, Wireless Video Transmission, Broadcast transmission, Power redistribution

目 录

[1 引 言 1](#_Toc484447188)

[1.1 课题背景 1](#_Toc484447189)

[1.2基于H.264/AVC的无线视频传输方案 1](#_Toc484447190)

[1.3 SoftCast无线视频传输方案 2](#_Toc484447191)

[1.4系统规格 3](#_Toc484447192)

[1.5本文所做工作 4](#_Toc484447193)

[2 SoftCast各模块算法分析及介绍 5](#_Toc484447194)

[2.1 DCT及其逆变换 5](#_Toc484447195)

[2.1.1 DCT算法分析 5](#_Toc484447196)

[2.1.2 DCT的逆变换 5](#_Toc484447197)

[2.2 SoftCast过程与LLSE过程 6](#_Toc484447198)

[2.2.1 块平均功率 6](#_Toc484447199)

[2.2.2 功率重分配 6](#_Toc484447200)

[2.2.3 Cordic算法 7](#_Toc484447201)

[2.2.4 CRC校验 8](#_Toc484447202)

[2.2.5 Turbo码的编码与译码 8](#_Toc484447203)

[2.2.6调制与软解调 9](#_Toc484447204)

[2.2.7 LLSE过程 10](#_Toc484447205)

[2.3 OFDM信道 10](#_Toc484447206)

[2.4 SoftCast无线帧结构 10](#_Toc484447207)

[2.5同步算法 12](#_Toc484447208)

[2.6信道估计算法 12](#_Toc484447209)

[3 发送端部分模块RTL代码的设计 14](#_Toc484447210)

[3.1发送端系统设计与基本框架 14](#_Toc484447211)

[3.2 HDMI数据缓存模块的设计 14](#_Toc484447212)

[3.3 DCT模块 15](#_Toc484447213)

[3.3.1 DCT定点化方案及输入输出信号 15](#_Toc484447214)

[3.3.2 DCT模块的设计 16](#_Toc484447215)

[3.4 SoftCast模块 17](#_Toc484447216)

[3.4.1 SoftCast模块的定点化方案及输入输出信号 17](#_Toc484447217)

[3.4.2 SoftCast模块的设计 18](#_Toc484447218)

[3.5组帧模块的设计 19](#_Toc484447219)

[4 接收端部分模块RTL代码设计 22](#_Toc484447220)

[4.1接收端系统设计与基本框架 22](#_Toc484447221)

[4.2解帧模块的设计 22](#_Toc484447222)

[4.3 LLSE模块 23](#_Toc484447223)

[4.3.1 LLSE模块定点化方案及输入输出信号 23](#_Toc484447224)

[4.3.2 LLSE模块的设计 24](#_Toc484447225)

[4.4 IDCT模块 24](#_Toc484447226)

[4.4.1 IDCT模块定点化方案及输入输出信号 24](#_Toc484447227)

[4.4.2 IDCT模块的设计 25](#_Toc484447228)

[5 系统测试 26](#_Toc484447229)

[5.1开发环境 26](#_Toc484447230)

[5.2系统性能仿真 26](#_Toc484447231)

[5.2.1峰值信噪比PSNR 26](#_Toc484447232)

[5.2.2仿真平台性能评估 26](#_Toc484447233)

[5.3硬件仿真与软件仿真结果对比 27](#_Toc484447234)

[5.3.1 DCT模块的硬件仿真 27](#_Toc484447235)

[5.3.2 SoftCast模块的硬件仿真 28](#_Toc484447236)

[5.3.3 组帧模块的硬件仿真 30](#_Toc484447237)

[5.3.4解帧模块的硬件仿真 31](#_Toc484447238)

[5.3.5 LLSE模块的硬件仿真 33](#_Toc484447239)

[5.3.6 IDCT模块的硬件仿真 34](#_Toc484447240)

[5.4 各模块的ChipScope捕获波形 35](#_Toc484447241)

[5.4.1 DCT模块 35](#_Toc484447242)

[5.4.2 SoftCast模块 36](#_Toc484447243)

[5.4.3组帧模块 37](#_Toc484447244)

[5.4.4解帧模块 37](#_Toc484447245)

[5.4.5 LLSE模块 37](#_Toc484447246)

[5.4.6 IDCT模块 38](#_Toc484447247)

[6 结论和展望 39](#_Toc484447248)

[6.1 结论 39](#_Toc484447249)

[6.2 展望 39](#_Toc484447250)

[参考文献 40](#_Toc484447251)

[谢 辞 41](#_Toc484447252)

# 1 引 言

## 课题背景

近年来，虚拟现实技术得到了飞速发展。与虚拟现实有关的应用也渐渐为人们所熟知。VR头戴式眼镜等基于虚拟现实技术的产品也越来越丰富了人们的生活。从技术上来说，虚拟现实技术是一种利用计算机技术生成的一种模拟环境，通过将多源信息的融合和实体行为的系统仿真使用户沉浸到该环境中。虚拟现实技术也是多种技术的综合，包括了计算机图形技术、图像渲染技术、交互技术和网络传输技术等。目前绝大部分的VR眼镜都是通过有线的方式和计算机连接，这限制了使用者的活动范围，也带来了诸多不便，因此，虚拟现实数据的无线视频传输是一个值得研究的课题。

对于虚拟现实的无线视频数据传输来说，和传统视频传输的一个显著的区别就是虚拟现实技术为了让用户有立体感的体验，需要对一幅原始图像处理成大小相当于其两倍的图像数据，因此前者所传输的数据量较大。除此之外，由于虚拟现实技术关注于用户的交互体验，在保证传输准确性的条件下，也要求视频传输系统有较低的延时。随着虚拟现实技术的成熟和与虚拟现实有关的业务的蓬勃发展，传统的无线视频传输方案也渐渐难以满足虚拟现实相关的业务日渐增长的需求。同时，虚拟现实技术也被常常用于教学这样需要广播的场景中。因此，一种既能满足虚拟现实视频传输的大数据量和低延时需求，也能高效地利用广播情况下各个优质信道的信道条件的无线视频传输方案是十分有价值的。本课题基于通信与智能计算实验室的“云环境下的图像视频群体协同表达与处理”国家自然科学基金重大项目中的课题三“视觉失真可容忍的群体化图像视频传输理论与方法”，研究一套针对VR广播的无线视频传输方案。

## 1.2基于H.264/AVC的无线视频传输方案

H.264/AVC是目前被广泛应用的一种数字视频压缩格式。它的主要技术特点包括帧内预测，帧间预测，整数变换、量化和熵编码等。帧内预测是指在一个指定大小的块内，通过目标像素点周围的像素值来预测目标像素点的像素值。帧间预测是通过运动估计和运动补偿消除时间冗余的过程。整数变换意即采用整数DCT变换，这种变换仅通过加减和移位操作即可实现。H.264提供52种从精细到粗糙的不同的量化步长，以满足不同的需求。H.264的优点在于它的压缩率高，在相同的通信条件下能传输更多的数据。

但是H.264也有它的局限性，这也是数字传输方式所共有的局限性，即量化过程是一个不可逆过程。采用H.264的信源编码方式，发送端通常需要对视频数据做压缩编码处理，并且在进行DCT变换之后，对其进行量化。由于量化过程是不可逆过程，因此对DCT系数量化的过程导致了基于H.264的无线视频传输方案在视频发送前就已经决定了被发送的视频的质量。同时，量化过程的特性导致了当信道的信噪比低于某一个门限时，接收端接收到的数据会急剧下降，这被称为数字传输方式的“悬崖效应”[[1]](#endnote-1)[1]。

“悬崖效应”从根本上来说，是因为对信源进行压缩编码和引入数字量化造成的。在数字传输的过程中，不可避免地带入量化误差，因为这些误差在发送端已经发生了，因此即使在最佳信道条件下，在接收端只能恢复出量化误差后的数据。其次，因为信道编码的纠错能力存在不足，当信道条件低于某一阈值的时候，接受的噪声会突然增大，对数据源无法有效恢复。

除此之外，基于H.264的无线视频传输方案的发送端需要预知接收端的信道条件，从而选择能使得接收效果最好的编码调制方案。在网上观看视频时，用户需要选择“标清”、“高清”、“超清”，这些选择会被告知发送端，这就是发送端预知接收端的信道条件的体现。但是在广播的情况下，每个用户需要接收到相同的视频数据，假设有些用户处于很好的信道条件下，而另一些用户处于较差的信道条件下。对于那些处于较好信道条件下的用户来说，由于发送端只能发送同样的数据且要保证每个接收端都能接收到视频数据，所以他们也只能和其他信道条件不好的用户接收同样的、经过严重压缩的视频，从理论上来说，信道资源不能得到充分的利用。从实际上来说，用户的信道条件原本能够接收“超清”的视频，却由于接收和信道条件较差的其他用户处于的同一广播源的压缩率高的视频，只能观看“标清”的视频，用户体验也不好。因此广播传输也是基于H.264的无线视频传输方案的短板。

## 1.3 SoftCast无线视频传输方案

SoftCast是一种与基于H.264信源编码的无线视频传输方案不同的无线视频传输方案。在广播传输的情况下，SoftCast方案不要求视频的发送端预先知道接收端的信道条件，而是让每一个接收端通过自己的信道质量来解码相同的一份视频数据。所以，如果接收端处在一个较好的信道条件中，即信道条件在数学上表现为高信噪比，那么接收端的用户就可以从接收到的视频数据中提取出更多的数据，并因此获得较好的视频质量。反之，若接收端处在一个较差的信道条件中，信道具有低信噪比，用户就只能以较低的质量观看视频[2]。

由于SoftCast方案中没有量化过程，对视频数据进行的都是线性变换，因此在物理层传输的数据和实际图像像素值之间的差值是线性相关的。在保证这一个基本原则的条件下，一个在信道中的小扰动只会在实际的视频数据中产生一个小的扰动。所以一个无线信道具有高信噪比的接收器接收的数据接近发送的数据，对其进行解码的结果接近原始的像素值。一个无线信道具有低信噪比的接收器接收的数据更远离发送的数据，对其进行解码的结果远离原始的像素值，得到较低保真度的图像。

SoftCast传输的过程是先经过SoftCast联合编码解码器，再将得到的结果通过OFDM传输。前者不同于当前传统的无线视频传输方案中用于压缩的视频编码和用于错误保护的物理层编码，它是一种同时提供压缩和错误保护的编码解码器，并且在字段上是线性的实数，从而确保编码前后两者之间的差异是线性相关的。后者反映在SoftCast方案的设计上，使得802.11的物理层可以直接绕过卷积编码和编码形式的802.11比特率的QAM调制等模块，直接通过OFDM传输SoftCast视频。通过上述两个设计，SoftCast方案不追求某一个单一的用户体验达到最佳，而追求绝大部分用户都能得到一个基于用户条件下一个较好的体验质量的效果。因此能在广播的情况下，让各个接收端都能解码出较为满意的质量的视频。

SoftCast对DCT变换后的系数做了功率重分配，这一过程包括DCT变换均为线性变换，保证了信道中实际传输的信号和原始视频像素值呈线性关系，因此视频的接收质量只取决于当前用户自身的信道条件，一个在信道中的小扰动只会在视频中产生一个小的扰动，从而避免了传统的无线视频传输方式在信道条件逐渐变差的临界点上产生的悬崖效应。即在信噪比逐渐降低的条件下，传统的无线视频传输方式会在某一个临界点点上性能断崖式下跌，SoftCast的视频传输性能则始终与信噪比成线性关系[3]。

SoftCast的系统在发送端的做法是：先对原始的视频信号进行DCT变换，在DCT变换的过程中，将低频分量集中到左上角，以便于接下来做功率重分配。接着将变换后得到的DCT系数划分为i块视频块，接着对每一块的DCT的系数，计算平均功率，通过各个块的平均功率，计算出功率分配系数，对于平均系数较大的视频块，功率分配系数要小一些，对于平均系数较小的视频块，功率分配系数要大一些。这也是SoftCast对于功率调节的原则。在功率重分配完成后，经过哈达马变换和稠密星座映射的过程，不经过FEC编码和调制，直接通过IFFT变换将频域信号转换为时域信号，在OFDM信道上传输。

在接收端进行的是上述过程的逆过程。在接收端通过FFT变换，得到原始的输入信号，再利用帧结构中的同步信号实现OFDM信号同步，通过导频信号来完成信道估计的功能。经过这些过程以后，由于DCT系数在发送端已经进行了能量重分配，所以在接收端也要进行与之相反的过程。解帧将数据分为模拟部分和数字部分之后需要进行LLSE过程，LLSE即最小线性均方误差算法。过程的结果是输出解码之后的DCT系数矩阵，对这个矩阵做一次IDCT变换，就得到了原始的视频信号。

上述设计思想也是本系统的主要指导思想。之后国内许多学者也有对该传输方案和具体实现有不少的研究成果，从各个角度分析了SoftCast传输方案和传统方案的性能对比以及提出了一些自己的优化等，这些都值得借鉴和学习，并以此为基础在本系统上实现创新。

## 1.4系统规格

从HDMI接口采集到的视频分辨率为1280\*720，帧率为30fps。在基带系统中传输的数据已经从视频的原始的RGB数据转换为了4:2:0的YUV数据。其中原始数据量与Y分量的数量比为1:1。

基带系统中各模块若无特别说明均以一行为缓存单位，例如对HDMI缓存模块来说，当摄像头输入模块的数据足够一行的时候，模块会给下一个模块使能信号以及有效数据。这种缓存方式也是无线视频传输系统低延时的保证。传统的无线视频传输系统大都缓存一个视频帧或几个视频帧，由于SoftCast基带系统处理单位可以小到一行，因此数据在缓存中的滞留时间将会大大降低。

基带系统即除去射频的发送端和接收端的硬件时钟频率保持和射频时钟频率相同，为30.72Mhz。这种做法也是为了减少不同频率对系统带来的设计复杂性。对于基带系统来说，它需要均衡摄像头数据频率和射频板数据频率，目前系统的设计是基带系统保持和射频板数据频率一致，并且在摄像头HDMI接口到基带之间设置了暂存一行数据的缓存。

每个OFDM符号包含4096个子载波和128个样点的循环前缀。因此符号长度为

在OFDM信道上传输的基本单位是子载波，但是子载波是按无线帧的结构排列好的。每个无线帧包括30个OFDM符号，其中为了接收端能完成信道估计过程，需要在特定的符号的特定子载波的位置上加入导频。同时，在无线帧的第二个符号和第三个符号的第1~257和第3841~4096个子载波处放置固定的同步序列。这些数据将会被用于计算出接收端连续不断的数据中的每一个无线帧的具体位置。其余关于帧结构的具体介绍，将会在2.3.1节给出。

## 1.5本文所做工作

第一章：介绍虚拟现实视频数据的特点，介绍了基于H.264/AVC的无线视频传输方案和SoftCast方案的比较，介绍了系统规格。

第二章：介绍了SoftCast系统中所用到的各个算法，整体剖析了SoftCast过程。

第三章、第四章：介绍了SoftCast无线视频传输系统发送端和接收端的RTL实现。

第五章：介绍了仿真平台的测试结果和RTL代码的测试结果。

# 2 SoftCast各模块算法分析及介绍

## 2.1 DCT及其逆变换

### 2.1.1 DCT算法分析

本系统中采用的是二维的整数DCT变换。DCT变换即离散余弦变换，它是与傅里叶变换相关的一种变换，它类似于离散傅里叶变换（DFT），但是只使用实数，由于DFT变换需要进行复数运算，因此在实际的通信系统中，它很少被使用，而是更多使用DCT变换作为图像处理的一个重要步骤。对DCT变换来说，图像数据可以视为一个二维的系数矩阵，不论是变换前还是变换后。输入给DCT模块的数据是摄像头采集来的图像时域信号，经过DCT变换，得到按频率分布的频域信号。DCT变换后，一块中的低频分量都集中在块的左上角，而高频分量都集中在块的右下角。对于一幅图像来说，主要能量都集中在低频部分，而高频部分大量的系数都接近于0。从人眼的视觉效果上来说，低频部分的数据存储了一幅图的主要信息[4]。最左上角的DCT系数称为直流系数（DC），因为这一点对应频率为0。

DCT变换通常的思路是先将图像分成8\*8,16\*16或32\*32的子块，对每个子块进行单独的DCT变换。因为虽然从原理上来讲可以对整幅图进行DCT变换，但是由于对一整幅图来说，即使对整体进行DCT变换可以获得更好的去相关效果，但是一整幅图的各个部分之间本身的相关性未必很高，因此变换的效果也不明显，更为重要的是这样会急剧增加计算的复杂度。用DCT变换可以表示为

其中

将上面的公式表达为矩阵乘的方式会更为简单，也便于通过硬件实现[5]。设原始的图像矩阵为f，经过DCT变换后的图像矩阵为F，则DCT变换可以表示为

其中A为DCT变换系数矩阵，且满足

### 2.1.2 DCT的逆变换

由于DCT变换的可逆性，IDCT的过程和DCT基本相同。对于公式(2.3)来说，假设可逆，在等式的两边同时左乘变换系数矩阵的逆且右乘变换系数矩阵的转置矩阵的逆，则变化前的图像矩阵可以表示为

因为DCT变换过程使用的变换系数矩阵是已知的，且对同样大小的DCT变换来说，它们的变换系数矩阵都相同。因此借助MATLAB等工具可以很轻松计算出是否可逆以及若满足可逆条件下，的逆矩阵和的逆矩阵。通过这些已知量，可以使用与DCT变换相同的计算方法，计算出DCT的逆变换即IDCT变换的结果。

## 2.2 SoftCast过程与LLSE过程

基带系统中的SoftCast过程包含了在发送端分别计算出数字部分和模拟部分数据的整个过程，在硬件实现上被包括在同一个模块中。这也是本系统中最能体现SoftCast思想的部分。在接收端，完成与SoftCast过程相反的LLSE过程。SoftCast过程包括求块平均功率，功率重分配，数据拼接与CRC校验，Turbo编码与16QAM调制几个步骤。与之相对的，LLSE过程包括软解调，Turbo译码，CRC校验与LLSE译码几个步骤。

### 2.2.1 块平均功率

DCT变换将低频分量聚集到系数矩阵的左上角，高频分量聚集到系数矩阵的右下角，依据SoftCast的功率重分配思想，每一个32\*32的DCT系数矩阵都可以被分为许多块，这些块内部的系数都有着某种相似的性质。在这里表现为它们的值较为接近。块的划分方式在系统设计时就已经定好，设为每一块的平均功率，其计算公式为

其中为每个块中的DCT系数。即对每个对应块内的所有元素平方的和在块内求平均。

### 2.2.2 功率重分配

功率重分配思想是SoftCast方案的核心思想之一，其主要的目的是通过改变一个DCT系数矩阵中各系数所占总功率的比例，来使得在有误码的信道条件下接收端处接收到的错误最小。例如某个信道中噪声的扰动范围为，一个幅值为2.5的信号直接在此信道上传输时，接收信号的值的范围为。考虑将这个信号在发送前放大十倍，则接收到的信号值的范围为，除以放大倍数之后得到，因此在发送前放大信号幅值可以减少信道噪声对该信号的影响[6]。

在求出了每个块内的所有15个以后，可以依此计算功率重分配系数。对于整个系统来说，功率的总量是恒定的。但是可以通过放大其中的一部分信号，来提高这些信号在信道上的抗干扰能力。虽然这种做法也会使得其他信号得到的功率少一些，但是总可以对每一个块找到一个最合适的功率重分配系数矩阵，来使得整个块在信道条件下在接收端LLSE解码时的均方误差最小。假设对第i个块的所有DCT系数，经过功率重分配之后的，被传输的值为

在接收端经过高斯白噪声信道，接收到的值为

其中n是均值为0，方差为的随机变量。则在接收端LLSE译码时均方误差为[7]

令，则需要求得以下表达式的最小值

其中P为总功率。一个DCT系数块共1024个点，去掉DC系数，共有1023个点需要进行功率分配，即P=1023。通过拉格朗日算子

可以解决这个优化问题。对与分别求偏微分并令其为0，最终可以求得

对于每个块，求出对应的的值，将每个块中的DCT系数和该块的功率重分配系数相乘，得到功率重分配以后的模拟部分数据。数字部分的数据是取DCT系数中的直流系数和每个块的进行处理。

### 2.2.3 Cordic算法

第2.2.2节已经给出每个块的功率分配系数的值通过总功率P和各块的平均功率求得的方法，公式可以化简为

要求得，必须经过开方运算。系统中使用了Cordic算法用于计算平方根。Cordic算法即坐标旋转数字计算方法，和其他传统的开平方算法不同的是，这种算法在迭代过程中只涉及到加减和移位运算，适合用硬件实现。它的基本原理是向量的旋转[8]。

假设在一个直角坐标系中，坐标逆时针旋转角度后得到,则它们满足关系

由于三角函数的值取值可以为所有实数，令,则乘上的操作等同于移位操作。在限定旋转角度的条件下，第i+1次旋转可以表示为：

旋转角度为

其中，。令旋转角度为，并且满足条件时迭代停止，此时的各个值为

,

,

因此，若要求的平方根，只需将，分别赋值和，代入即可。硬件实现中的Cordic算法共使用16次迭代求得平方根。求的过程可以通过复用两次Cordic算法完成。

### 2.2.4 CRC校验

CRC即循环冗余校验码，它的算法是：通过将位二进制数作为检查附加，来编码位二进制数字的消息，并发送位信息位和位校验位。所以这种编码也称为码，将二进制数字的稀疏化作为变量中的多项式的系数是十分方便的。例如，消息110101由多项式表示。任意一个由二进制位串组成的代码都可以和一个系数仅为‘0’和‘1’取值的多项式一一对应。系统的CRC校验使用的生成多项式为

加CRC校验的过程是对输入的位数据先添加个0，然后用这个新的数据模2除以生成多项式，所得的余数即n位CRC校验码。

在接收端，完成的是CRC校验的检验过程。这和添加CRC校验码的过程类似，是用接收到的比特序列模2除以生成多项式，若所得余数为0，则表示CRC校验正确，否则CRC校验出错，数据有误。CRC是一种检错码，并不具有纠错能力。但是由于在实际系统中，经过纠错码Turbo的译码过程并不一定能完全纠正所有错误，因此在Turbo码过程后的CRC校验能检验出绝大部分没有被Turbo码纠正的错误。在实际系统中，若经过CRC校验发现了错误，则丢弃这些数据，在图像的相应位置使用上一帧的数据。若CRC校验正确，就可以将数字部分的各DCT块的DC系数和行平均的15个正确解出。

### 2.2.5 Turbo码的编码与译码

由于数字部分保存着最为重要的DC系数和接收端LLSE解码过程所需要解出的一行中的各个，因此这部分数据应进行纠错编码，以保证接收端在可接受的误码情况下能还原出添加完CRC校验位后，数字部分的数据将会进行Turbo编码过程[9]。

Turbo码是又称为并行级联卷积码，是一种性能非常好的纠错码。它采用的是随机编码的思想，并且结合了随机交织器和卷积码。模拟结果表明，采用大小为65535的随机交织器进行Turbo编码，迭代次数为18时，在高斯信道下码率为1/2的Turbo码的误码率，这个性能接近了Shannon限[10]。

Turbo码的编码器由两个递归系统卷积编码器和一个随机交织器连接，这两个编码器分别输出相应的校验位。待编码的数字序列一方面直接进入第一个编码器，另一方面通过随机交织器进入第二个编码器。这个随机交织器的作用是将输入的数字序列转换为比特长度相同但各比特位重新排列的序列。编码后的校验位经过删余过程，从两个校验序列中周期性删除一些校验位，以提高码率。

译码部分，Turbo码的译码器译码器由两个软输入软输出译码器，加上一个和编码器一样的交织器组成。两个译码器的工作不同。第一个译码器对第一部分的分量码进行最佳译码，得到每一比特的最大似然信息，这些信息经过交织器后被作为先验信息在第二个译码器对第二部分的分量码进行最佳译码，这一结果又经过解交织送回给第一个译码器，经过多次迭代后，两个译码器的信息趋于稳定，就可以对接近于整个码的最大似然译码的似然比进行硬判决，就能得到信息序列的最佳估计值。

### 2.2.6调制与软解调

经过Turbo编码后，所得的数字部分码块按归一化后的能量进行16QAM调制，星座点采用格雷码方式映射。

16QAM调制即16个星座点的正交幅度调制，调制的过程是将输入的4位二进制数据映射到星座图对应的点上。星座图是一个复平面，其上的每一个点都代表一个复数，如图所示[11]。

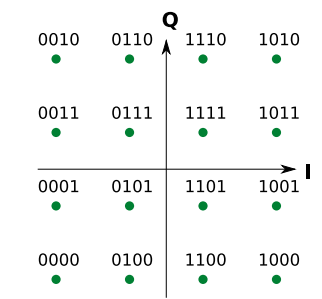


图2.1 16QAM星座图

映射到对应的点后，形成复数调制符号，符号的I分量和Q分量共同表示出这一个点在星座图上的位置，经过调制，信号变得能够在OFDM信道上传输。调制过后的数字部分数据和之前经过功率重分配计算得出的模拟部分数据一起组成无线帧。

软解调是调制的逆过程。由于系统假设的信道条件是衰落信道，因此需要在软解调的过程中将乘性噪声系数H作为一个参数代入计算，解调得到的数字部分的LLR值。软解调的输入包括解帧得出的经过信道补偿的X值的数字部分和乘性噪声系数H值的数字部分。经过16QAM调制的信号经过信道传输，由于信道噪声等因素，其值未必是发送端经过信道之前的精确值。如果通过解调的过程能得到与发送端调制过程相同的星座点的坐标，也就能还原出发送端经过调制之前的数据。解调分为硬判决解调和软判决解调。硬判决就是指在不确定对应哪个星座点的条件下勉强做出判决，软判决就是指通过码元的后验概率或似然函数判断星座点的位置[12]。SoftCast系统中的解调方式使用的是软判决解调方式。即软解调不给出准确的星座点，而是给出一些已知的概率信息，对这些信息而不是确定的星座点进行Turbo译码。由于Turbo译码也是一个通过概率确定结果的过程，可以将软判决的结果概率作为输入再进行译码。经过软解调和Turbo译码过程后的数据被还原为原本在发送端未经过Turbo编码的数据。

### 2.2.7 LLSE过程

在接收端，通过将数字部分的数据进行一系列处理，可以分离得到各行的15个。参照发送端通过求得功率分配系数的过程，称为LLSE解码过程。通过式求得，在衰落信道的条件下，实际的DCT系数通过以下公式还原：

其中，是通过数字部分的数据分离得到的，为信道噪声的平均功率，是信道估计模块给出的，为接收到的DCT系数。

## 2.3 OFDM信道

随着通信技术的发展，人们对无线通信资源的使用逐渐从尽可能的提高覆盖率转变到提高频谱利用率上。传统的FDM技术为了避免各子载波之间的干扰，必须在相邻的子载波之间保留较大的间隔。在这种情况下，频谱利用效率较低。随着数字调制技术FFT的出现，FDM技术有了突破性的进展。FFT允许在保持子载波之间的正交性的条件下，将FDM的各个子载波重叠排列。由于OFDM技术中的子载波排列是部分重叠的，所以相同带宽下可以排列更多的子载波。OFDM技术的实现方法为将需要传送的信号分割为N个子信号，然后分别调制到N个相互正交的子载波上，因为子载波的频谱是相互部分重叠的，所以该技术的频谱效率高。

OFDM信道的优势在于在发送端进行了数据符号的串并变换，这种变换使得每个子载波符号周期相对变长，从而减轻了无线信道的多径时延的时间弥散性对系统的影响，避免了频率选择性衰落。同时子载波的相互正交也提高了系统容量。为了避免OFDM符号之间的干扰，需要在符号之间加入循环前缀。

OFDM基带信号要通过IFFT变换调制到无线信道中，接收端经过载波下变频通过FFT变换实现OFDM信号的解调。

## 2.4 SoftCast无线帧结构

SoftCast的无线帧结构如下图所示。

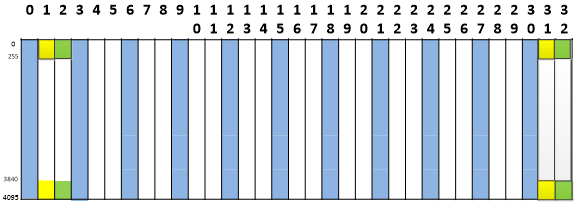


图2.2 SoftCast无线帧结构图

图中，最上方的数字表示的是OFDM符号编号。一个SoftCast无线帧包含30个OFDM符号，每个OFDM符号包含4096个子载波以及128样点的循环前缀。每一个子载波对应射频一个时钟周期可以传输的数据，包含了32bit的数据。一个无线帧包含4096\*30个子载波。因此，在图中，序号为30的OFDM符号是下一个无线帧的起始位置。

由于经过IFFT的数据在射频板上必须连续发送，而在接收端却无法判断出什么位置才是一帧的起始位置，因此在每个帧结构中需要加入长度总数为1024子载波的同步序列，这些数据用于同步算法。第2和第3个符号的第2~257，第3841~4096个子载波保存的就是这样固定的同步数据，在图上用特别的方框表示出来。

同时，在某些符号的特定位置插入了导频数据，这些数据用于接收端的信道估计，以便在接收端接收到受到噪声影响的数据的情况下还原发送端发送出去的子载波。

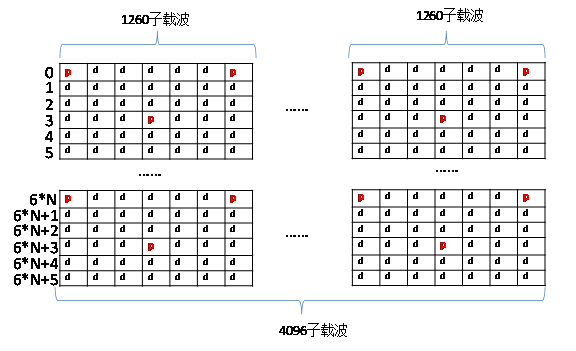


图2.3 导频子载波和数据子载波所处位置

如上图所示，每一行是一个OFDM符号，图中的d代表数据子载波，p代表导频子载波。用数学语言描述，符号序号记为symbol，取值为1~30，子载波号设为number，取值为1~4096。其中所有符号的第1261~2836个子载波均空闲置0。即符合条件mod(symbol,6)==1的符号中符合条件mod(number,6)==1的子载波，这些位置包含的是第一类的导频数据。符合条件mod(symbol,6)==4的符号中符合条件mod(number,6)==4的子载波，这些位置包含的是第二类的导频数据。这些导频数据都是固定的。

## 2.5同步算法

因为接收端从天线上接收来的数据是连续不断的，因此需要一段特定的数据（称为A组数据），这些数据的特性使得它们经过确定的另一组数据（称为B组数据）进行相关操作以后表现出和其他数据较为明显的差异。具体表现为A组数据和B组数据做相关操作的结果，比其他正常视频数据通过OFDM信道传输来的结果和B组数据做相关的结果的值有一个显著的大小差距，这里将A组数据和B组数据的相关结果的值称为峰值。

在组成帧结构的过程中，将A组数据固定的保存在每一个无线帧第二和第三个符号的第1~257和第3841~4096个子载波处。四组同步数据共占用1024个子载波，并且这四组数据对于每一个无线帧都是相同的。B组数据保存在本地，通过它们进行相关操作，就可以找到从天线上接收来的连续不断的数据中，那些特定的数据的位置。又根据A组数据在无线帧中占据着固定的位置的特性，就可以得知一个个无线帧结构在接收到的数据流中的位置。由此可以确定FFT操作该从哪些数据开始。

同步模块的执行过程是，模块对不断接收来的数据做FFT，并与本地预存好的B组数据的FFT结果做相关操作，即接收到的数据实部与本地预存的同步数据FFT结果的实部，虚部与虚部做乘和累加操作，并将所得到的结果送入IFFT模块做IFFT，对IFFT的结果，错位2048个数据累加，得到结果的峰值的位置即对应下一个帧头在序列的相对位置。再将这个位置传送给FFT，以便FFT变换和去CP前缀。

## 2.6信道估计算法

无线通信系统的性能和正确性很大程度上会受到无线通信信道的影响，这和有线通信有很大的不同。数据在无线通信信道上传输时会收到噪声的干扰，与原来的值表现出偏差。需要通过信道估计来在接收端恢复出发送端的发射信号，而信道估计的过程就是通过导频的变化值估计其他有效数据的变化值的过程。

帧结构中有固定位置的导频，这些导频不论在发送端还是接收端都是已知的。系统假设无线通信的信道是衰落信道，这种信道的噪声可以表示为：

其中y为接收端接收到的信号，H为乘性噪声系数，x为发送端发送的原始信号，n为加性噪声系数。SoftCast无线通信系统采用的是MMSE信道估计算法，信道估计的过程就是通过利用在发送端和接收端都已经知道原来的x值。在接收端通过接收到的实际y值和已知的x值，可以计算出导频点的H值，再利用这些点的H值进行维纳滤波，得到临近点的H值。

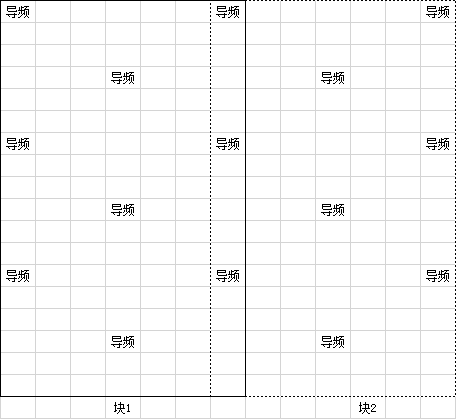


图2.4 信道估计所用的导频位置及其影响的结果

如上图所示，实线部分的为块1，块1内的即实线部分不包括虚线部分的18\*6=108个点由实线框内的9个导频数据通过维纳滤波计算得出。同理块2内的108个点也由块2的9个导频计算得出。最后，信道估计模块对每一个子载波，输出经过信道补偿的X值和乘性噪声系数H值。

# 3 发送端部分模块RTL代码的设计

## 3.1发送端系统设计与基本框架

在发送端，视频数据从一个摄像头读入，通过HDMI接口，传输到基带系统中，经过发送端的一系列处理，通过射频板发射出去。在接收端通过一系列处理过程，最终像摄像头采集到的图像在VR眼镜上显示出来。

系统发送端的模块微架构如下图所示：



图3.1 SoftCast发送端模块微架构

在发送端，从HDMI采集的数据经过32\*32矩阵大小的2D-DCT变换，得到DCT系数矩阵，按照不均等的划分方式，将32\*32的DCT系数矩阵划分为最左上角的DC系数（直流系数）和其余的15个反L型块。

对于i的取值从1到15的每一个块，计算该块的平均功率，同时利用一行中所有块的平均功率，计算出一行DCT系数的15个平均功率，再用这15个平均功率计算出这一行DCT系数对应块的15个功率重分配系数。

此时数据流分为两部分，即模拟部分和数字部分。模拟部分即图中所示的SoftCast过程，把之前计算得到的功率重分配系数乘上对应L型块的每个DCT系数作为模拟部分的数据。在这部分处理后，模拟部分的值就已经是经过功率重分配过程的数据。数字部分的数据的来源是左上角的DC系数和每个L型块的。这些数据按预设的码块长度拼接后加CRC校验码，按照固定的长度进行Turbo编码过程，Turbo编码的结果经过16QAM调制作为数字部分的数据。数字部分和模拟部分的数据到组帧过程会进行一部分的交织，用输入的有效数据组成无线帧的结构，以OFDM符号为单位开始IFFT过程，并且加上循环前缀CP。IFFT得到的时域数据将通过射频板发送到接收端。

## 3.2 HDMI数据缓存模块的设计

摄像头的扫描方式是按行扫描的。这也十分易于理解，对于一幅1280\*960的图片，按行扫描是指从最左上角的第一个像素点开始扫描，得到图像的像素值，一直扫描到第一行的最后一个像素点，即整幅图的最右上角的那个像素点。此时共扫描了1280个像素点。基带系统支持的分辨率为1280\*720，因此HDMI缓存模块需要截取原始图像中左上角1280\*720个像素点，其余的像素点的数据丢弃。这样就可以在数据进入DCT变换之前截取有效数据，多余的数据就不需要进行以后的变换过程。

同时，这里还需要一个行转块的过程。由于DCT变换的单位是32\*32的块。而摄像头扫描的方式是按行扫描，通过这种方式的扫描并不能直接得到一块一块的图像原始数据，如图所示。



图3.2 HDMI缓存模块示意图

一个合理的解决办法就是将摄像头采集来的数据先进行缓存。缓存32行以后，在HDMI接口模块的缓存处得到一幅1280\*32的部分图像，对应实际要传输的图像的最上方32行数据。当缓存的数据达到32行时，在HDMI接口模块处将这些数据进行按块分离。按块间从左到右，块内从左到右，从上到下的方式，将这些数据传送给DCT模块做DCT变换。

## 3.3 DCT模块

### 3.3.1 DCT定点化方案及输入输出信号

DCT模块的输入为32\*32的矩阵，矩阵的每个元素的位宽为8bit，范围[0,255]的无符号数。定点表示为x(u,8,0)，即含0位小数位的8位无符号数。变换矩阵的定点表示为A(s,8,6)，即含有6位小数位的8位有符号数。x进行两次矩阵乘得到结果y(s,34,12)，最后截掉小数位并右移位，同时饱和截位得到16bit的有符号数。

因此，DCT模块的输出为32\*32的矩阵，矩阵的每个元素的位宽为16bit，范围[]的有符号数，记为(s,16,0)。

DCT模块的输入信号：

input clk; //系统时钟，30.72Mhz

input rst\_n; //异步复位信号，低电平有效

input enable; //使能信号，提前有效数据一个周期拉高

input sync\_input\_line; //行同步信号

input [7:0] data\_in; //输入8位有效数据

DCT模块的输出信号：

output [15:0] data\_out; //输出16位DCT系数

output reg sync\_o; //输出有效信号，拉高表示下一周期的data\_out为有效输出

output reg sync\_output\_line; //输出行有效信号，输出一行的开始时拉高

### 3.3.2 DCT模块的设计

DCT模块的微架构如图所示。



图3.3 DCT模块微架构

上一节已经提到，系统中采用整数DCT进行近似计算。在具体实现中，直接使用矩阵向量相乘再累加的方式完成运算。由于矩阵相乘满足结合律，因此可以先计算后两个矩阵相乘的结果，即先计算。先对通过流水输入的图像原始数据缓存，等缓存满32个8位无符号数时（一个DCT行），一次将这些数取出来送入向量乘法器和DCT变换系数矩阵中对应的一列进行乘法计算，即同时计算中的一行乘以中的一列的结果。仔细观察DCT系数矩阵，可以发现它们在（从1行开始计数）奇数行是左右对称的。以i作为矩阵元素的下标，在偶数行，和互为相反数。利用这一点，在计算过程中可以通过判断当前行数是奇还是偶，来提前计算好或，将这个结果作为临时变量拿去做乘法，用这种方式，可以少做一半乘法，节约了硬件资源。因此，在模块中每一周期只做16次乘法，而通过在做乘法前预先将对应的变量加减来确保正确性。这些乘法的结果将会在接下来的每一个流水周期内两两相加，而每一个流水周期都可以进行一个行向量和列向量的相乘。设当前时钟为起始时钟，根据矩阵乘法的性质，的第一行的每一列的数都是矩阵f第一行的所有数和中该列的所有数对应位置的乘积之和，则第一个时钟周期模块做16次乘法，对所有求得的结果，并用16个寄存器将其保存下来。

由于做一次相同位数的加法恰好可以扩大一位，并且在同一周期做太多次运算会造成路径过长而未必能综合出正确的结果，因此在模块中，下一周期，这16个寄存器的值两两相加得到8个新的临时变量。也将这些值存储到相应的寄存器变量中。这一周期（即完成乘法运算的下一个周期）实际上16个乘法结果的寄存器的值已经没有意义了，因此可以用于存储下一个向量点乘的结果即对所有求得的结果。这样在计算16个寄存器的周期值两两相加这个周期里，也没有闲置硬件资源，这就是流水线设计。[13]

因此，实际实现方式是，对输入矩阵来说，每一行的32个数都需要遍历所有的变换32列DCT变换系数。系数矩阵以变量的方式存储在程序中，每隔1个时钟周期，表示当前系数矩阵的变量就按照之前预设的值改变一次。这些数作为一个列向量与之前保存下来的矩阵f的行向量做向量运算，同时在中间存储器的相应位置中保存运算的结果。计算完之后，相当于完成了一次一维DCT变换。

由于经过一维DCT变换的数据的DC分量都集中在了第一行，转置后则集中在最左一列。此时如果再经过一次一维DCT，就可以保证DC分量都集中在最左上角的那一个点。这样的临时存储器需要两个，因为这种做法相当于在写入存储器的同时进行了矩阵行列转置。因此在一个存储器处于写入状态的时候，程序可以读取另一个存储器，所以需要两个相同的存储器交替存储。这种存储方式称为乒乓存储。这种存储方式最简单的实现方式是在其中一块存储器刚好存满的时候，控制输入转向另一块存储器，同时这个已经存满的存储器开始输出。这样，在1024个周期后，存满的存储器又可以进行一次交换，再换回刚输出完数据的存储器接收数据，依次循环。

## 3.4 SoftCast模块

### 3.4.1 SoftCast模块的定点化方案及输入输出信号

经过DCT变换之后的数据按块送入SoftCast模块中。SoftCast模块将每个32\*32的DCT块分为16个部分，包括DC系数和15个反L型块，块的具体划分如图所示。

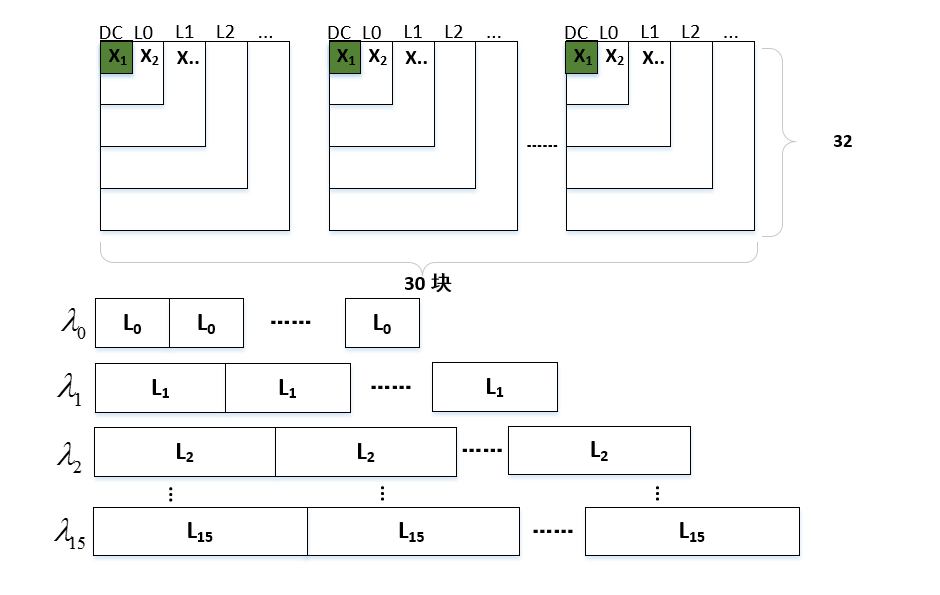


图3.4 SoftCast过程中块的划分

其中，每个块的行、列分界点可用向量[1,2:2:29,30:32]表示。每个块的第1行第1列即Y(1,1)为DC分量，Y(2:3,2:3)元素组成的L型块用于计算；Y(4:5,4:5)元素组成的L型块用于计算，依此类推。最后一个块由Y(30:32,30:32)组成，用于计算。为了减少数字部分的数据量，对每行30个DCT块都使用相同的，这个是该行30个DCT块中对应块的平均值。

SoftCast模块的输入为DCT变换后的DCT系数(s,16,0)，求功率需要平方，得到(u,32,0)。每一个块中元素相加的个数最多为183个，这个数字小于256即2的8次方，因此总功率之和定为(u,40,0)，加和结果求平均需要除以183，转化为乘2865后右移19bit，饱和得到(u,24,0)。再将30个块的15个分别求平均结果依然为(u,24,0)。根据式(2.13)，代入P=1023，可以得出功率重分配系数的定点为(u,16,16)，这些系数乘以所有的DCT系数y(s,16,0)并饱和截位到(s,16,9)，作为模拟部分的数据。

包括DC系数和每一行的15个在内的这16个数据先进行位宽压缩和拼接，通过仿真平台上已经测试的多组实际视频，使用matlab仿真统计这一部分的数值范围，据此将15个的数据位宽分别设为[18 15 14 13 14 15 13 12 13 14 16 12 11 10 10]，DC系数位宽设为13bit。这样整个一行的数字部分拼接后的数据位宽为200+30\*13=590bit。这些数据补10个0到为600bit后分割成3个等长的码块，每个码块长度为200bit。由于这部分数据非常重要，同时也为了满足Turbo编码的长度要求，在进行Turbo编码之前，需要再对每个码块添加CRC校验。对每一行的数字部分来说，在每个码块后添加16bit的CRC校验位后得到3个216bit码块。因为一个216bit大小的码块恰好可以进行一次Turbo编码。1/3码率的turbo编码输出编码后码块长度为216\*3+12（尾比特）= 660bits，调制后每个码块占用660/4=165子载波。因此数字部分共占用165\*3=495个子载波。

SoftCast模块的输入信号：

input clk; //系统时钟，30.72Mhz

input rst\_N; //异步复位信号，低电平有效

input enable; //使能信号，提前有效数据一个周期拉高

input signed [15:0]data\_in; //输入数据(s,16,0)

input input\_line\_sync; //行同步信号

SoftCast模块的输出信号：

output sync\_digital; //数字部分输出同步信号

output [15:0]data\_out\_digital\_real; //数字部分输出信号real

output [15:0]data\_out\_digital\_image; //数字部分输出信号image

output output\_line\_sync\_digital; //数字部分输出帧同步信号

output sync\_analog; //模拟部分输出同步信号

output [15:0]data\_out\_analog\_real; //模拟部分输出数据real

output [15:0]data\_out\_analog\_image; //模拟部分输出数据image

output output\_line\_sync\_analog; //模拟部分输出帧同步信号

### 3.4.2 SoftCast模块的设计

由于输入和输出数据数量不匹配，即每输入一行的DCT系数为30720个，但不仅需要输出经过模拟部分即功率重分配后的这30720个数，还需要输出数字部分即DC系数和15个，经过调制后的结果。若输入输出位宽相同，则无法平衡输入与输出。因此将相邻的两个16位的DCT系数拼接成一个32位数，一次输出一个32位数。SoftCast模块微架构如图所示。



图3.5 SoftCast过程模块微架构

SoftCast模块对输入的每一个DCT系数先求平方，并且通过一个寄存器变量对其进行累加。这是为了求块平均功率所必要的步骤。这里通过一个地址表来表明目前输入的系数序号是属于哪一个块。



图3.6 用于求的地址表内容

地址表给出当前的地址对应属于哪一个块，就可以判断将计算出的功率值加到哪一个块的变量上。计算出每块中全部的功率之和之后，需要将这个数除以该块的数据个数以求平均。

做除法在硬件上实现需要消耗较大的资源，因此在误差允许的范围之内，将除法操作改为乘法和移位操作，因为乘法和移位操作通过硬件实现较为方便。以第四块为例，第四块数据个数为40，要将求得的总功率除以40，可以通过乘13107再右移19位实现，即相当于除以了/13107=40.000610。

求出了一行中各个块的平均功率之后，对一行内的各个对应求平均，再参考式(2.13)，将各个块的平均功率送入cordic 模块进行开方运算。在开方运算完毕之后可以求得和，但是这里需要对再开一次根号。所以需要复用一次cordic模块。在求得之后，需要用这个数去除1，这里没有更好的解决办法，只能使用流水线除法模块，得到结果。流水线除法和第二次开根号可以并行执行，提高执行效率，最后将所有的结果全部乘起来得到，用之前缓存好的DCT系数乘以块对应的得到模拟部分的数据，这些数据也被缓存起来。

在对做cordic开方运算的同时，一行的15个和所有的DC系数也被处理作为数字部分的数据。这些数据先进行拼接后补零，成为3块200bit的数据块用于添加CRC校验码。添加16bit的CRC校验位后送入Turbo模块进行编码。经过Turbo编码得到的数据进行调制后作为数字部分的数据。CRC校验模块、Turbo编码的模块和调制模块都已有现成的代码可供调用。

## 3.5组帧模块的设计

组帧模块微架构如图所示。



图3.7 组帧模块微架构

组帧模块以行为单位接收到经过SoftCast过程的数字部分数据和模拟部分数据后，将这些数据存放在两块交替使用的PQ RAM中。由于接收端需要通过导频值进行信道估计以确定信道补偿后的数据，因此需要在固定位置加入导频数据，同时为了接收端能进行同步操作找到OFDM符号的起始位置，也需要在帧的固定位置加上同步数据。在输入4行图像数据之后，组帧模块会输出一个完整的无线帧，组帧输出的时候，需要在符号与符号之间加入一段间隔，以便于IFFT过程加入循环前缀，即CP，用来保证有时延的OFDM信号在FFT积分周期内总是具有整数倍周期。系统中的CP长度为128个子载波。

在组帧模块中，每2个伪模拟数据或调制后的数字数据组成一个复数信号占用一个OFDM子载波。一行30个32\*32的DCT块的伪模拟数据共占用30\*512=15360个子载波。调制后的数字部分共有3码块，每码块占165个子载波，与伪模拟部分拼接后共占15855个子载波。输入顺序是先输入数字部分的495个子载波，后输入模拟部分的15360个子载波。

同时，为了数字部分码块对抗频率选择性衰落，拼接后的前2520子载波进行一次行列交织，交织行列分别选择为15行和168列。

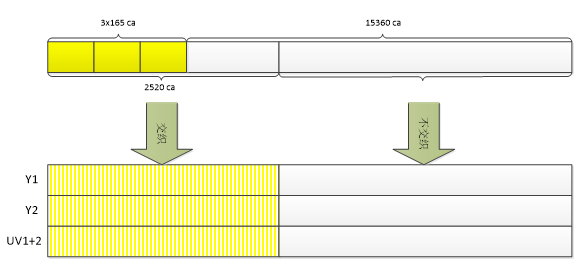


图3.8 组帧过程中的行列交织

一种最简单的实现方式是，行列交换就是先将数据存入矩阵再读出矩阵，但是这个过程是按列存储，按行读出。经过这一个矩阵，数据的顺序就发生了变化，因此，也就完成了行列交织。而对于序号在2520之后的数据，只要顺序读入顺序读出即可，注意在组成无线帧的时候，需要在指定的地方加入导频子载波和同步序列。而系统中的硬件实现方法是先用MATLAB生成一个地址映射表，然后以ROM的形式导入硬件代码中，再在输入特定序号的数据时，查询地址映射表得到该数据所应该存储到的缓存地址。利用高级语言生成一个二维矩阵很容易，并且若改变系统设计的要求，这种方式也不用改动硬件代码，易于维护。

在模块的具体实现中，使用block memory的IP core ROM保存导频数据和同步数据，使用IP core RAM保存输入的有效数据。在程序执行时设置一个变量用于计数当前输出的位置。对照地址表，若当前输出的位置为同步或导频数据，就将保存导频数据或同步数据的临时变量取出，并且将其对应的地址指针自增，若当前输出的位置对应的是有效数据，则从RAM中读出有效数据，同样地址指针自增。

# 4 接收端部分模块RTL代码设计

## 4.1接收端系统设计与基本框架

系统接收端的模块微架构如下图所示：

******

图4.1 SoftCast接收端模块微架构

在接收端，从天线接收的数据先经过同步模块获取帧同步定时和频偏。同步模块获取每一个无线帧的帧头的位置信息，将这个信息给FFT模块，以保证开始做FFT变换的数据都是完整的一个OFDM符号。FFT模块根据同步模块获取的帧同步定时对接收天线数据进行4096点的FFT变换并去除循环前缀。信道估计利用FFT得到的结果中的导频数据，获得信道补偿后的每一个子载波的频域数据X和衰落信道下的乘性噪声系数H。解帧模块根据已知的导频和同步符号在帧结构中的位置对信道补偿后的频域数据以及按帧结构有序排列的乘性噪声系数进行解交织，分离出数字部分和伪模拟部分后，分别送入不同的后续模块。经过信道补偿后的数字部分加上乘性噪声系数的数字部分一起作为软解调的输入。经过软解调和Turbo译码后得到带CRC校验位的每行15个平均功率，即每一行的15个，以及所有块的平均DC系数。解帧后的伪模拟数据根据数字部分译码得到的值进行LLSE译码得到发送端DCT系数的估计值，同数字部分译码得到的DC系数组合成完整的DCT系数矩阵，一同送入IDCT模块，IDCT模块对32\*32的DCT系数矩阵进行IDCT变换，还原成原始的图像数据通过HDMI口输出到VR眼镜上。

## 4.2解帧模块的设计

解帧的过程其实就是组帧过程的反过程。输入给解帧模块的数据单位是OFDM子载波，以无线帧的格式按顺序输入。由于帧结构是已知的，因此解帧操作的过程比较简单，值得注意的是需要进行解除有效数据行列交织的过程。经过信道估计的OFDM符号只剩下2520个子载波，因为其中的1576个无效数据已经被去除，留下了符号头部和符号尾部的有效数据。

由于信道估计有经过信道补偿的X值和乘性噪声系数H值两路输出，且都是完整的帧结构，只是其内部数据的意义不同。因此在实际系统中，也需要包含两个并行执行的解帧模块，分别处理H部分和X部分。

解帧模块微架构如图所示。



图4.2 解帧模块微架构

为了便于将来设计方案有可能发生的修改，解帧模块将整个无线帧的帧结构以coe文件的形式存入IP core中。用MATLAB根据帧结构生成地址表和数模对照表。在模块中，使用各两块以乒乓存储方式工作的用于存储数字部分数据和模拟部分数据的RAM。其中地址表指示的是某一个子载波对应的数据应该被保存在上述存储器中的位置，数模对照表表示某一个子载波对应的数据的去向，包括数字部分RAM，模拟部分RAM，丢弃（导频、同步）等。对一整行数据都给出他们的去向并存放好之后，就可以准备输出。由于LLSE模块控制的需要，要求数字部分必须在当行的模拟部分之前输出，且不能同时输出，即使它们对应着不同的输出变量。因此若每当一块RAM存满就将其输出的做法是不可取的。目前采用的的做法是，另加一块RAM缓存，用于存储当前存满一行的数字部分数据，若判断上一行的模拟部分输出完毕，则这些数据开始输出，否则留在缓存中等待。

## 4.3 LLSE模块

### 4.3.1 LLSE模块定点化方案及输入输出信号

解帧模块一个周期的输出是32bit的数据，不论是补偿后的数据X路，还是信道估计值H路，定点方案均可以表示为(s,20,10)，即含有10位小数位的20位有符号数。输出为DCT系数，这里由于LLSE部分解码出g(u,16,16)，(u,18,0)，经过公式(2.21)的过程，得到DCT系数的表示为(s,32,16)。

输入部分：

input clk; //系统时钟，30.72Mhz

input rst\_n; //复位信号，低电平有效

input input\_line\_sync; //行同步信号

input frame\_analog\_enable; //模拟部分数据使能

input [31:0] frame\_analog\_data; //输入X路模拟部分数据

input frame\_digital\_enable; //数字部分数据使能

input [31:0] frame\_digital\_data0; //输入X路数字部分数据

input [23:0] frame\_digital\_data1; //输入H路数字部分数据

input [7:0] sigma; //信道估计根据信道信噪比给出的值

输出部分：

output output\_line\_sync; //行同步信号

output enable\_o; //输出使能信号

output [31:0] data\_out\_idct0; //llse输出0路分量

output [31:0] data\_out\_idct1; //llse输出1路分量

### 4.3.2 LLSE模块的设计

LLSE模块微架构如图所示。

图4.2 LLSE模块微架构

LLSE模块的绝大部分都是对SoftCast模块执行的逆过程。解帧得到的H路和X路的数字部分首先被送入LLSE模块进行软解调。每4个数字部分数据可以得到并行的4个LLR值。软解调模块被封装成一个独立的模块，在输入一行有效数据后输出495组4个LLR值。每个LLR值的数据宽度为16bit，这样一组LLR值就含有64bit的数据。

这些数据经过重新拼接组合，成为660组48bit的数据送入Turbo译码模块进行译码。由于Turbo译码需要较长的时钟周期，因此在接收端复用了3个Turbo译码模块进行并行计算。每个Turbo译码模块接收220组48bit的数据，解码出216bit的有效数据之后，将这些数据送入CRC校验模块进行CRC校验，CRC校验通过后，经过数据分离等过程，可以得到一行的所有15个平均值和一行每个DCT块的所有DC系数。

利用公式(2.13)，采用和发送端SoftCast模块相同的过程求得功率分配系数之后，将所有的和代入LLSE公式(2.19)，将之前缓存好的X路的模拟部分数据作为，利用对应的和，求得对应位置的原本DCT系数，缓存好一行的DCT系数后，分两路同时输出。

## 4.4 IDCT模块

### 4.4.1 IDCT模块定点化方案及输入输出信号

LLSE模块输出的是两路DCT系数，表示为x(s,32,16)。变换系数矩阵的定点表示为A(s,8,6)，是DCT变换系数矩阵的逆矩阵，这个逆矩阵通过MATLAB求出。x进行两次矩阵乘得到结果y(s,58,28)。最后截掉小数位并右移5位，同时饱和截位得到8bit的无符号数。

因此，IDCT模块的输出为32\*32的矩阵，矩阵的每个元素为(u,8,0)。

IDCT模块的输入信号：

input clk; //系统时钟，30.72Mhz

input rst\_n; //异步复位信号，低电平有效

input enable; //使能信号，提前有效数据一个周期拉高

input sync\_input\_line; //行同步信号

input [31:0] data\_in; //输入31位DCT系数

IDCT模块的输出信号：

output [7:0] data\_out; //输出8位原始图像数据

output reg sync\_o; //输出有效信号，拉高表示下一周期的data\_out为有效输出

output reg sync\_output\_line; //输出行有效信号，输出一行的开始时拉高

### 4.4.2 IDCT模块的设计

经过LLSE过程得到的数据是连续的32\*32的DCT系数，这些数据需要进行DCT变换的逆变换即IDCT变换来得到视频的原始图像数据。由于LLSE模块中输出结果的机制比较特殊，将会同时输出两路即64bit的数据，如下图所示。



图4.4 IDCT的并行输入

因此需要进行两路的并行IDCT变换，即先将LLSE解码的结果按块存放，再按系数分发给两个IDCT模块做IDCT变换，得到的结果再重新组合，拼接成原始的图像数据。IDCT模块的并行输出方式如下图所示。



图4.5 IDCT的并行输出

因此，在系统实现上的做法是，重用DCT模块的代码，在这个基础上调整输入输出的位宽和已经保存在代码中的变换系数矩阵的值。同时加入一个并行控制模块，控制两个IDCT的输入和输出。其余执行过程和DCT模块类似。

# 5 系统测试

## 5.1开发环境

操作系统：Windows 10

MATLAB版本：R2016a

基带Verilog开发环境：ISE 14.7

Verilog代码仿真环境：Isim 14.7

FPGA：Xilinx V7

## 5.2系统性能仿真

### 5.2.1峰值信噪比PSNR

峰值信噪比（PSNR）经常用作图像压缩等领域中信号重建质量的测量方法，其主要物理意义是影像信号的最大值与影像中杂讯的比值。其数学表达式为：

其中，MSE为原图像与处理图像之间的均方误差。

### 5.2.2仿真平台性能评估

在MATLAB仿真平台上，模拟信道条件是高斯白噪声信道和衰落信道条件下，随着信道的信噪比SNR的变化时时，结果的PSNR指标如图所示。

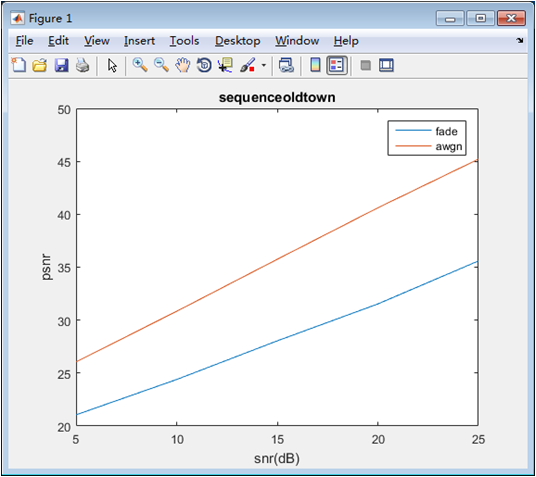


图5.1 信道条件变化时PSNR的变化

从图中可以看出，SoftCast仿真平台给出的结果是随着信道条件SNR的变化，PSNR的变化是线性的。对于采用H.264的信源编码，传统的调制方式的点对点传输来说，PSNR无法随着SNR的变化线性变化，因为在SNR低于某一门限值会产生悬崖效应，而SoftCast方案则能避免这样的结果。

Turbo码的误比特率随SNR的变化如图所示。

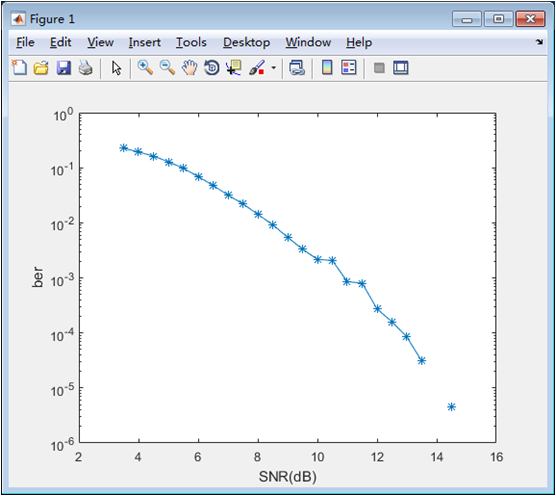


图5.2 Turbo码的误比特率随SNR的变化

从图中可以看出，Turbo码具有优良的性能，即使处于较低SNR的条件下，误比特率也能控制在一个可以接受的水平。由于Turbo码解码后还有CRC校验这一阶段，又能控制一些错误。同时，Turbo码的误比特率也随SNR线性变化。

## 5.3硬件仿真与软件仿真结果对比

通过使用实际视频在MATLAB仿真平台上测试，得到了在各个模块连接处的测试数据。测试数据包含10个视频帧。硬件仿真通过使用Isim等工具，查看硬件代码运行时的实际波形，并且输出结果与仿真平台的测试数据进行对比。

### 5.3.1 DCT模块的硬件仿真

将测试数据dct\_in\_y\_frame1.txt以coe文件的形式导入IP core ROM，作为模块的输入。DCT模块的输入波形如图所示：

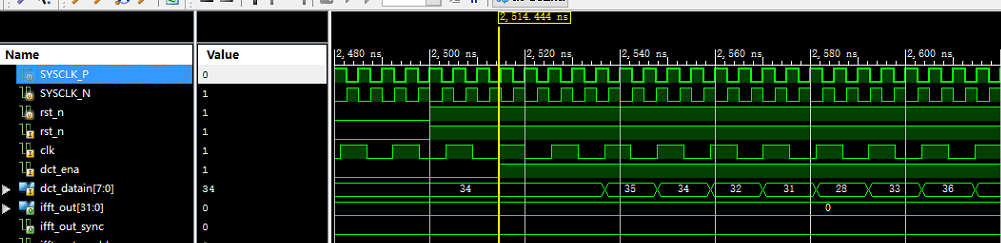


图5.3 DCT模块仿真的输入波形

enable拉高的位置恰好在输入数据34之前一个周期，因此34为输入的第一个有效数据。输出波形如图所示：

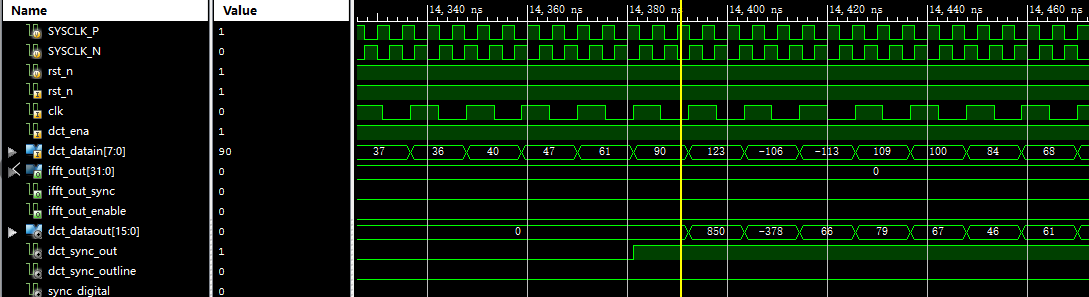


图5.4 DCT模块仿真的输出波形

enable拉高的位置恰好在输出数据850之前一个周期，因此850为输出的第一个有效数据。将这些输出数据输出到TXT文件中，通过MATLAB和dct\_out\_y\_frame1.txt的数据相减对比，得到结果。

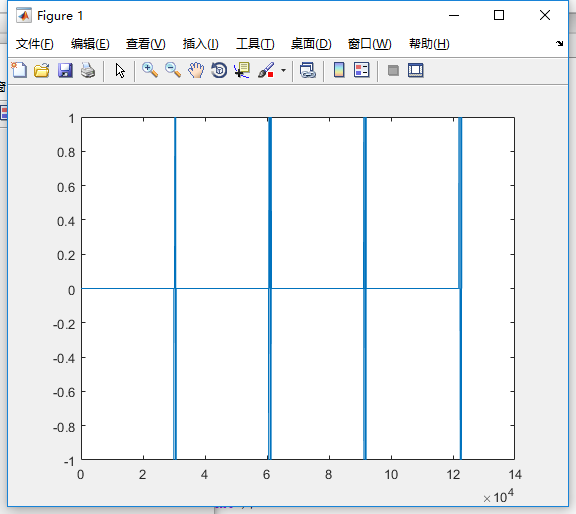


图5.5 DCT模块软件仿真结果与硬件仿真结果对比

从结果中可以看出，在左右的数据个数下，模块输出的结果与软件仿真下的结果误差在，即误差的绝对值不超过1，且误差个数非常有限，因此结果是理想的。这里产生的误差原因是MATLAB和硬件Verilog四舍五入的方式有所差异。例如对于-8.5这个数，MATLAB四舍五入得到的结果是-9，而按Verilog代码实现得到的结果则是-8。

### 5.3.2 SoftCast模块的硬件仿真

将测试数据dct\_out\_y\_frame1.txt以coe文件的形式导入IP core ROM，作为模块的输入。DCT模块的输入波形如图所示：

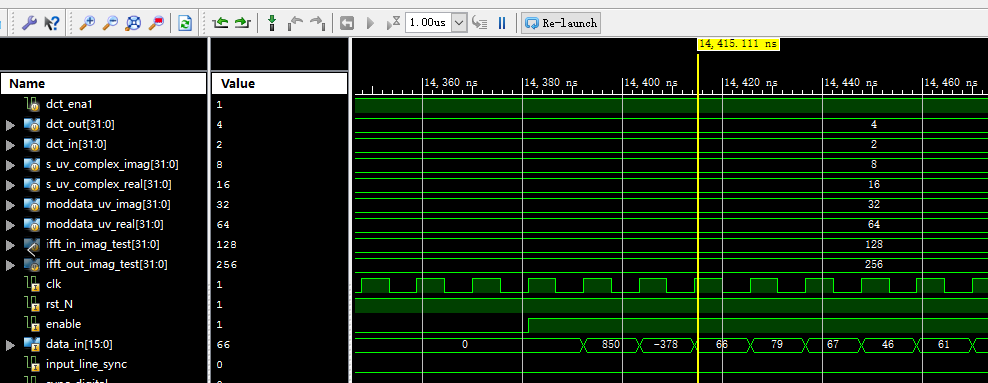


图5.6 SoftCast模块仿真的输入波形

enable拉高的位置恰好在输入数据850之前一个周期，因此850为输入的第一个有效数据。输出数字部分波形如图所示：

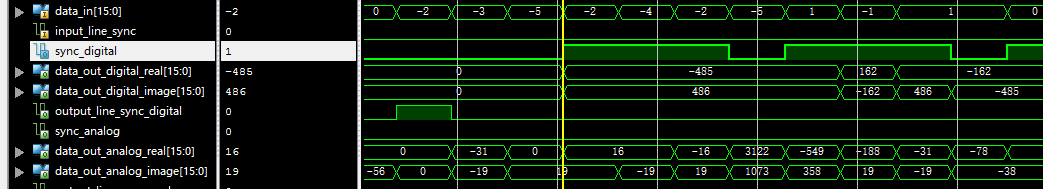


图5.7 SoftCast模块数字部分的输出波形

sync\_digital拉高的位置恰好在输出数据(-485,486)之前一个周期，因此(-485,486)为输出的第一组有效数据。将这些输出数据输出到TXT文件中，通过MATLAB和moddata\_y\_imag\_frame1.txt和moddata\_y\_real\_frame1.txt组合的数据相减对比，得到结果。

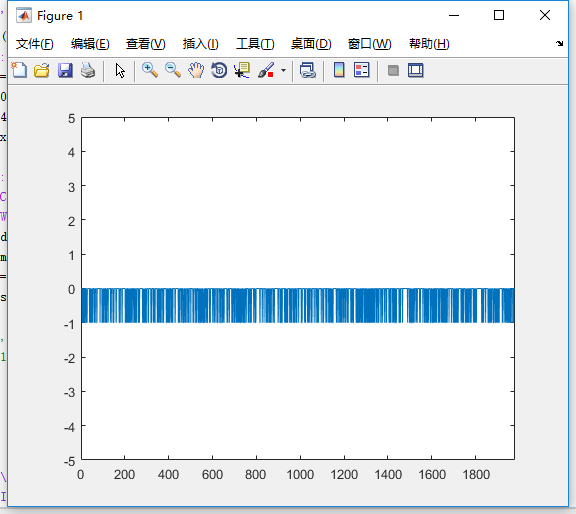


图5.8 SoftCast模块数字部分软件仿真结果与硬件仿真结果对比

输入数据1980个即4行数据，约占一整幅图的1/5。数字部分的数据由于采用16QAM调制，实部或虚部固定取值为4个值：±162，±486。这里由于负数四舍五入的误差，所有的-486都在输出时变为-485，但是这个误差对接收端的软解调过程不产生影响，因为接收端能判定-485的值以极大的概率就是-486。因此该误差对系统的运行也没有影响。结果也是理想的。模拟部分的输出在当行的数字部分之后，波形如图所示：

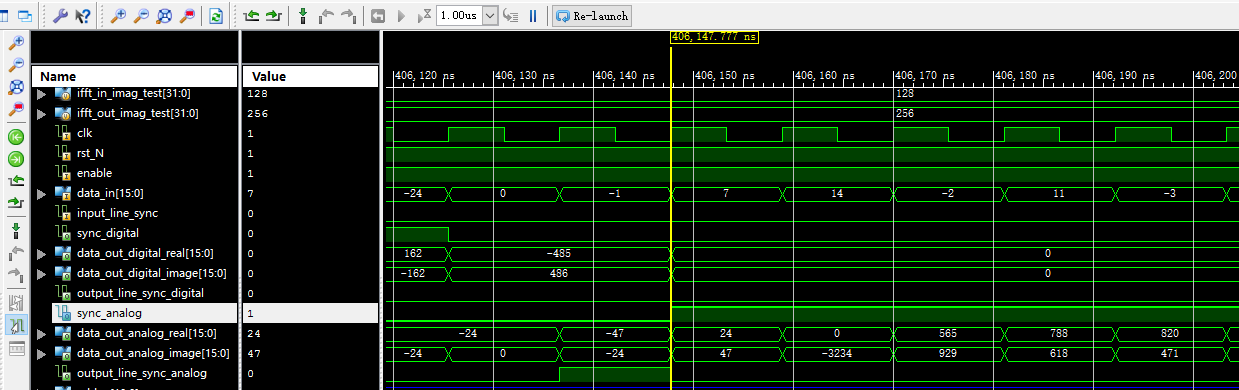


图5.9 SoftCast模块模拟部分的输出波形

sync\_analog拉高的位置恰好在输出数据(0,-3234)之前一个周期，因此(0,-3234)为输出的第一组有效数据。0表示DC系数，由于DC系数太大，已经通过数字部分调制后传输了，模拟部分就无需再传一次。将这些输出数据输出到TXT文件中，通过MATLAB和s\_y\_complex\_imag\_frame1.txt以及s\_y\_complex\_real\_frame1.txt数据相减对比，得到结果。

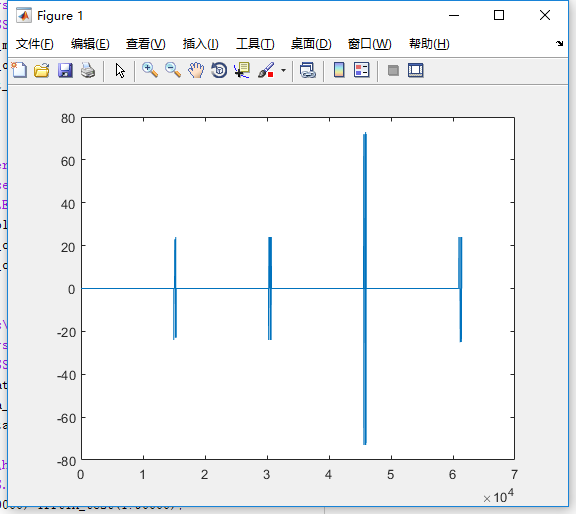


图5.10 SoftCast模块数字部分软件仿真结果与硬件仿真结果对比

数据在第个和第个左右有明显的波动。经查，这部分数据波动的原因在于cordic算法开根号的误差导致。

### 5.3.3 组帧模块的硬件仿真

将测试数据moddata\_y\_real\_frame1.txt、moddata\_y\_imag\_frame1.txt、moddata\_uv\_real\_frame1.txt、moddata\_uv\_imag\_frame1.txt、s\_y\_complex\_imag\_frame1.txt、s\_y\_complex\_real\_frame1.txt、s\_uv\_complex\_imag\_frame1.txt、s\_uv\_complex\_imag\_frame1.txt按两行y分量，一行uv分量以及实部虚部匹配的方式组合好，以coe文件的形式导入IP core ROM，作为模块的输入。组帧模块的输入波形和SoftCast模块的输出波形相同，组帧的输出波形如图所示：

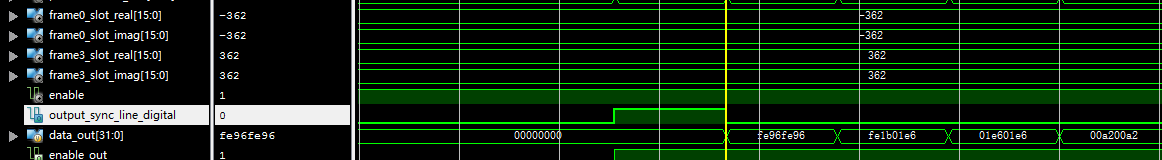


图5.11 组帧模块的输出波形

output\_sync\_line\_digital信号指示一个符号的起始，enable\_out信号拉高的位置恰好是第一个数据fe96fe96(分实部虚部)的前一个周期，因此第一个输出数据为fe96fe96。将这些输出数据输出到TXT文件中，通过MATLAB和ifft\_in\_imag\_frame1.txt数据相减对比，得到结果。

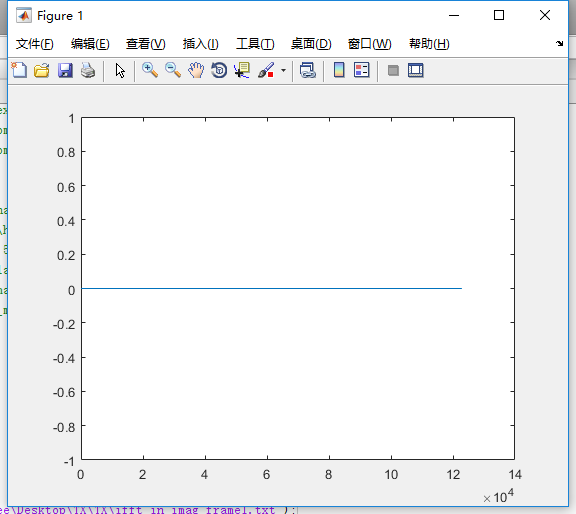


图5.12 组帧模块软件仿真结果与硬件仿真结果对比

由于组帧模块并没有运算单元，执行的都是数据搬运的操作，因此在功能正确的情况下，不会出现误差。

### 5.3.4解帧模块的硬件仿真

对于解帧模块来说，X分量和H分量虽然数据不同，但是它们的组成结构是相同的，因此只需要测试对X分量的解帧操作正确性即可。将测试数据x\_compensate\_imag\_frame1.txt和x\_compensate\_real\_frame1.txt按实部虚部匹配的方式组合好，以coe文件的形式导入IP core ROM，作为模块的输入。解帧模块的输入波形如图所示：

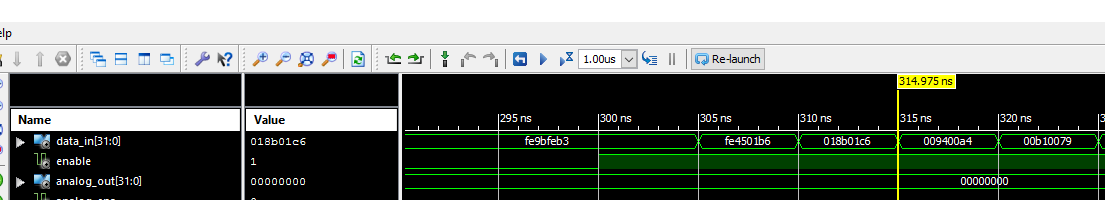


图5.13 解帧模块的输入波形

解帧模块的enable信号在fff8ff90前一个周期拉高，即对应fff8(实部)ff90(虚部)为第一个输入数据。输出数字部分的波形为：

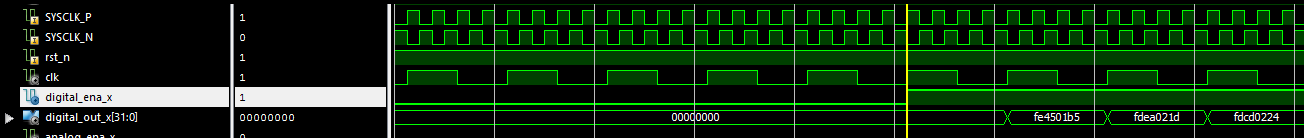


图5.14 解帧模块数字部分的的输出波形

对数字部分，输出使能信号为digital\_ena5，对应第一个输出的值为fe45(实部)01b6(虚部)。模拟部分输出接在数字部分输出之后。

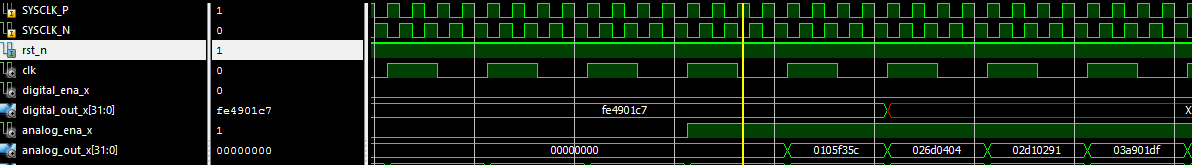


图5.15 解帧模块模拟部分的的输出波形

对模拟部分，输出信号使能为analog\_ena，对应第一个输出的值为0106(实部)f35c(虚部)。将这些输出数据输出到TXT文件中，通过MATLAB和rS\_complex\_imag\_frame1.txt、rS\_complex\_real\_frame1.txt、rmodSignal\_imag\_frame1.txt、rmodSignal\_real\_frame1.txt实部虚部组合得到的数据相减对比，得到结果。

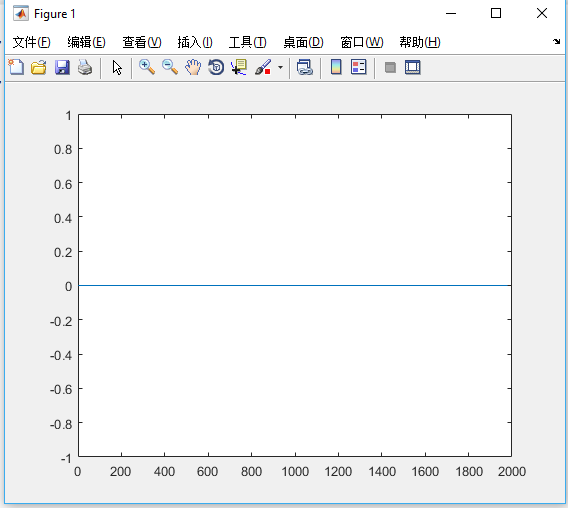
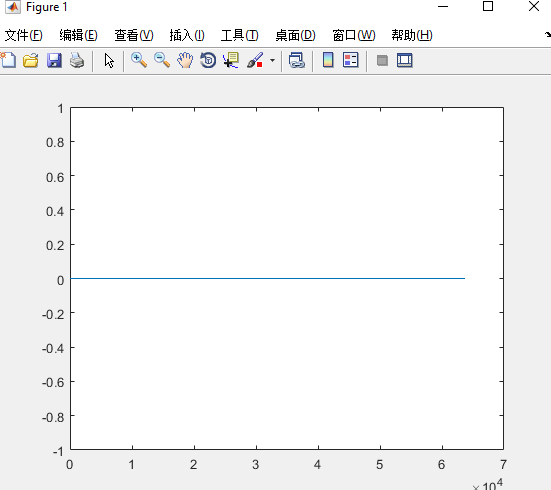


图5.16 解帧模块的软件仿真结果与硬件仿真结果对比

和组帧模块一样，解帧模块不经过运算，若功能正确则输出的结果一定是正确的。

### 5.3.5 LLSE模块的硬件仿真

将测试数据X部分的rS\_complex\_imag\_frame1.txt、rS\_complex\_real\_frame1.txt、rmodSignal\_imag\_frame1.txt、rmodSignal\_real\_frame1.txt，H部分的rmodH\_imag\_frame1.txt、rmodH\_real\_frame1.txt按实部虚部匹配的方式组合好，以coe文件的形式导入IP core ROM，作为模块的输入。LLSE模块的输入波形如图所示：

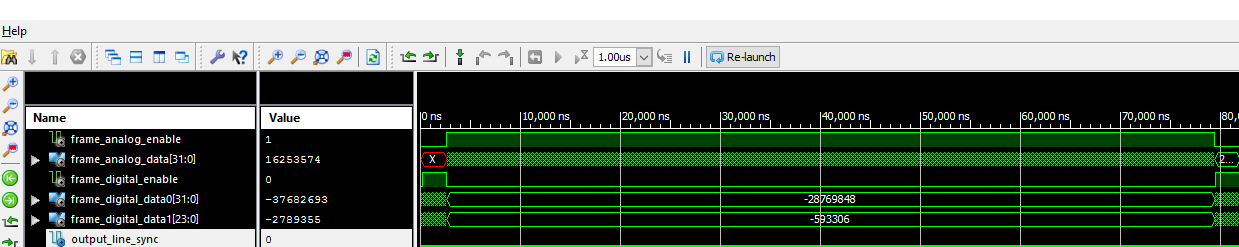


图5.17 LLSE模块的输入波形

LLSE把解帧得到的结果的H部分的数字部分，和X部分的数字和模拟部分作为输入。对应使能信号为frame\_analog\_enable和frame\_digital\_enable。

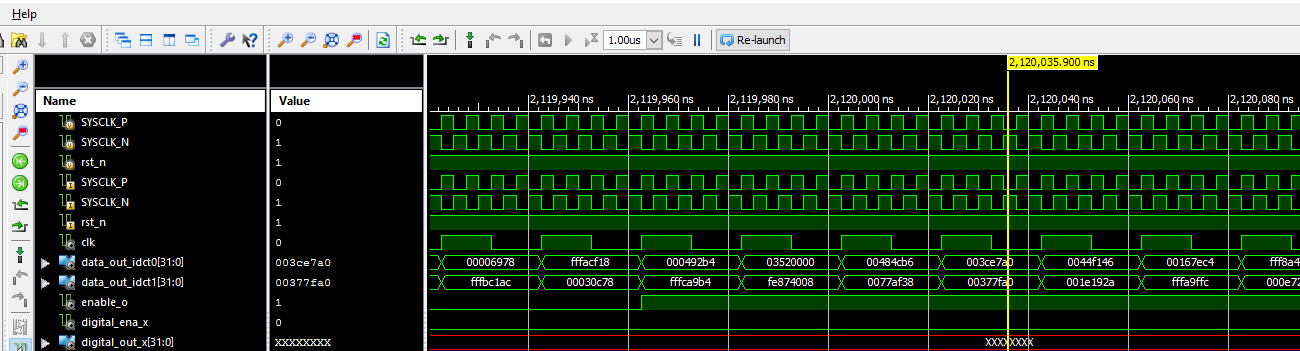


图5.18 LLSE模块的输出波形

和其他模块一样，输出信号enable\_o拉高的下一个周期对应有效数据，即55705600和-24690680。由于LLSE分两路输出，无法直接和MATLAB的仿真结果对比，因此对比LLSE模块中求出的功率分配系数的值，因为最后一步是通过简单还原DCT系数的过程，若的值正确了，LLSE的输出也正确了。

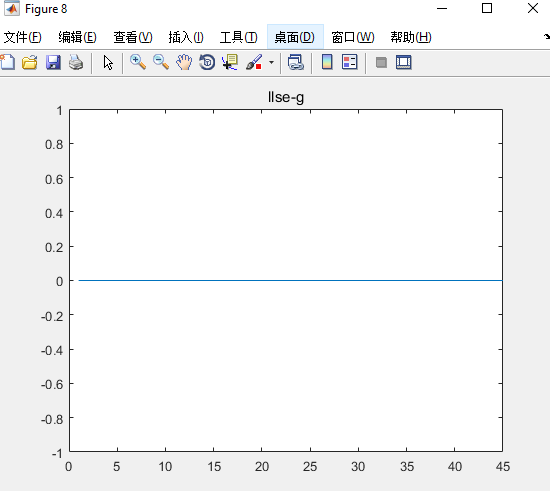


图5.19 LLSE模块功率分配系数的软件仿真结果与硬件仿真结果对比

### 5.3.6 IDCT模块的硬件仿真

将测试数据idct\_in\_frame1.txt以1024个数为单位按块依次分离，组成两个文件，再将其分别以coe的形式导入IP core ROM，作为模块的输入,一次输入两个数。IDCT模块的输入波形如图所示。

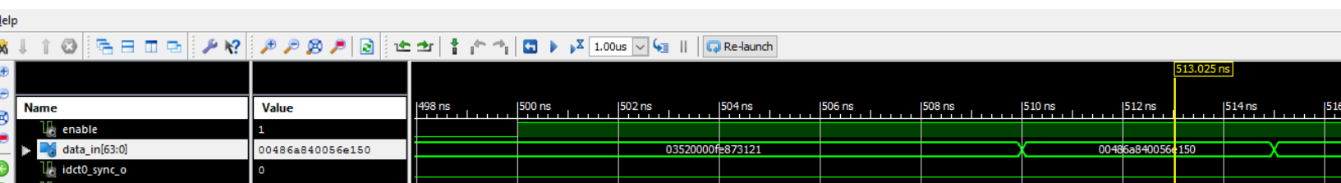


图5.20 IDCT模块的输入波形

enable为使能信号，在有效数据前一周期拉高。IDCT的输出分为两路输出，每一路都输出一块完整的视频原始数据。

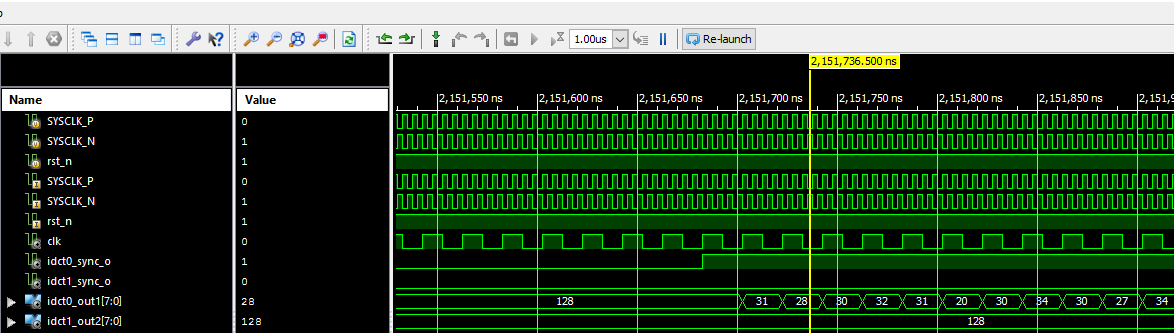


图5.21 IDCT模块第一路的输出波形

如图所示，每一路的输出都有专门的idct\_sync\_o来控制。两路输出间隔1024个周期，用MATLAB编写一个脚本将两路输出组合起来并和idct\_out\_frame1.txt进行比较。结果表明模块功能正确性得到了验证。

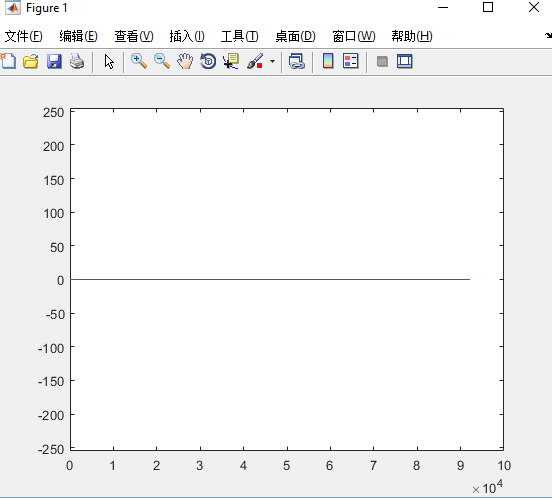


图5.22 IDCT模块的软件仿真结果与硬件仿真结果对比

## 5.4 各模块的ChipScope捕获波形

在需要观察的变量前添加(\* KEEP = "TRUE"\*)防止该信号被综合优化掉，在ChipScope的界面中设置好触发条件，在FPGA上烧入工程文件，拨动RESET按钮后即可在ChipScope中观察捕获到的波形。

### 5.4.1 DCT模块

设置触发变量为dct\_sync\_out，观察变量为dct\_dataout。ChipScope波形如图所示。

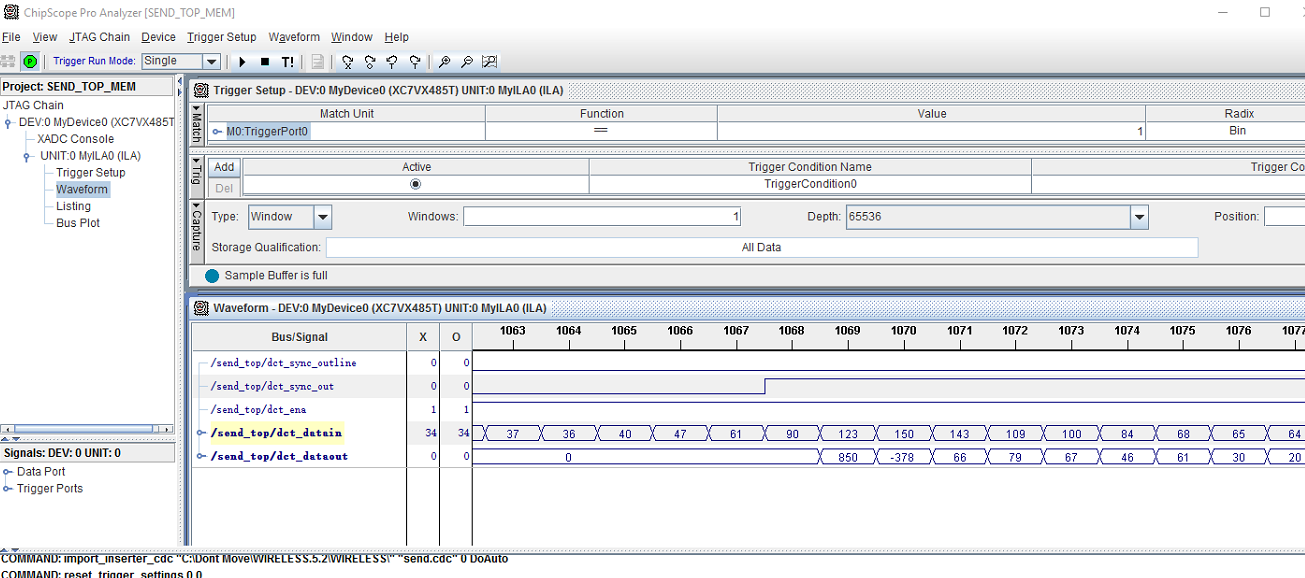


图5.23 DCT模块的ChipScope波形

经过比对，波形形状和数据都和硬件仿真的结果相同。

### 5.4.2 SoftCast模块

设置触发变量为sync\_digital，观察变量为data\_out\_digital\_real、data\_out\_digital\_imag、data\_out\_analog\_real、data\_out\_digital\_imag。ChipScope波形如图所示。

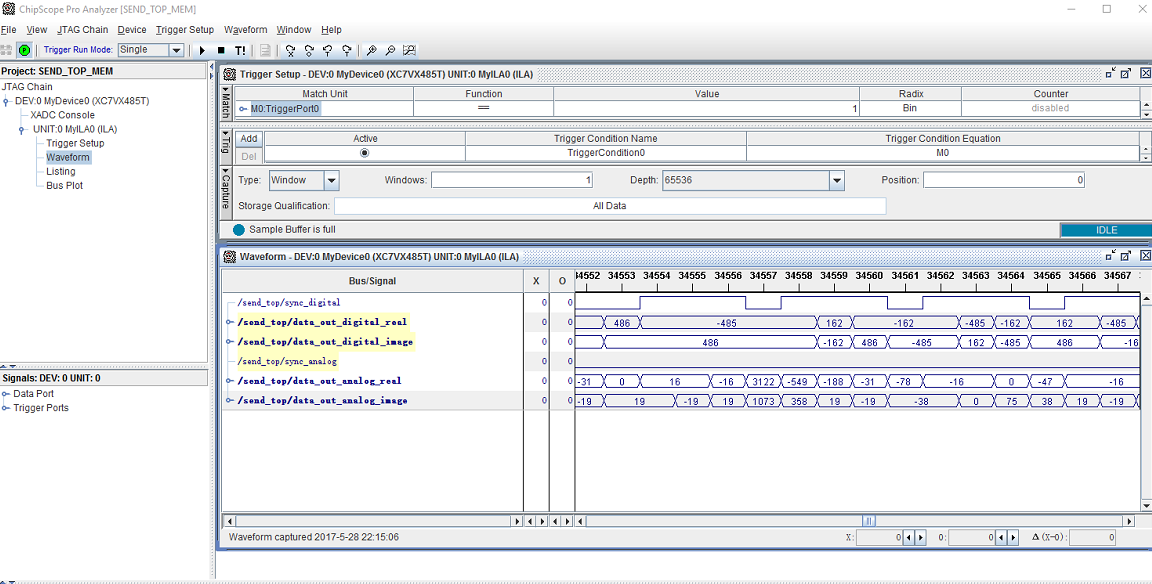


图5.24 SoftCast模块的数字部分ChipScope波形

由于模拟部分的输出总是在数字部分之后，因此设置触发变量为sync\_digital的同时可以在波形的后部分观察到模拟部分的数据。如图所示。

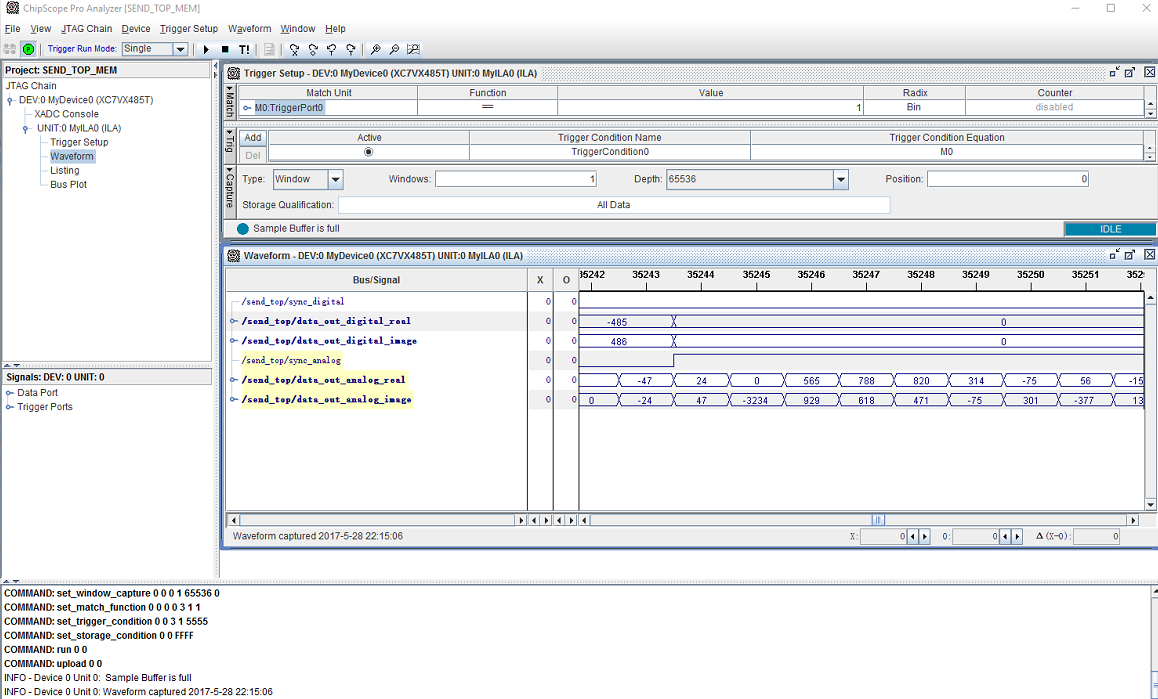


图5.25 SoftCast模块的模拟部分ChipScope波形

经过比对，波形形状和数据都和硬件仿真的结果相同。

### 5.4.3组帧模块

设置触发变量为trs\_enableout，观察变量为ifft\_in\_real、ifft\_in\_imag。ChipScope波形如图所示。

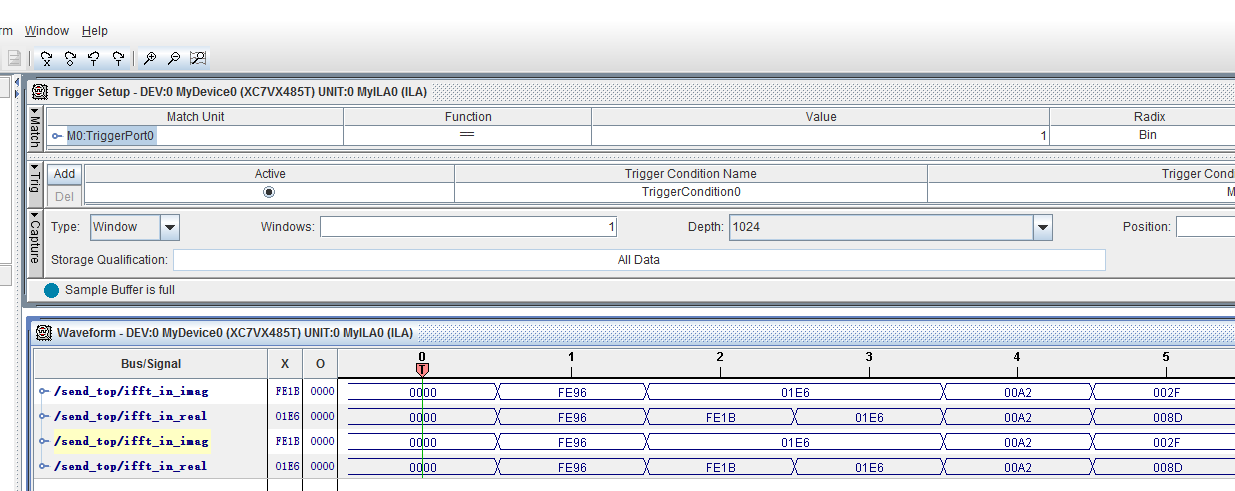


图5.26 组帧模块的ChipScope波形

经过比对，波形形状和数据都和硬件仿真的结果相同。

### 5.4.4解帧模块

数字部分，设置触发变量为digital\_ena\_x，观察变量为digital\_out\_x。ChipScope波形如图所示。

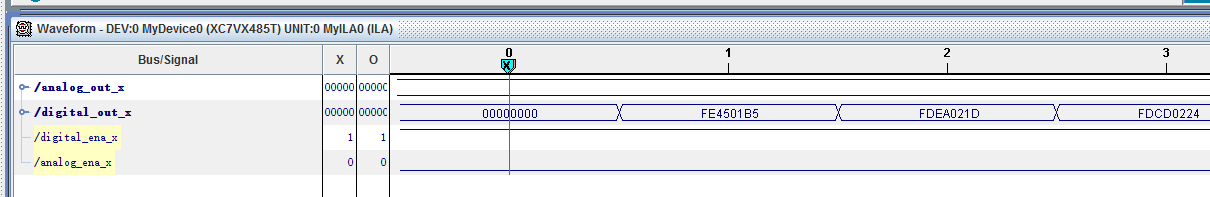


图5.27 解帧模块数字部分的ChipScope波形

模拟部分，设置触发变量为analog\_ena\_x，观察变量为analog\_out\_x。ChipScope波形如图所示。

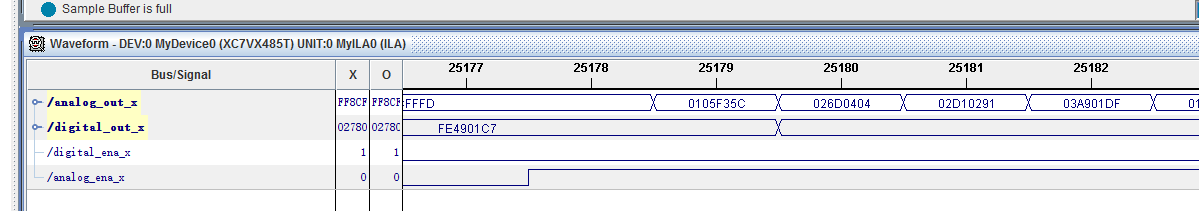


图5.28 解帧模块模拟部分的ChipScope波形

经过比对，波形形状和数据都和硬件仿真的结果相同。

### 5.4.5 LLSE模块

设置触发变量为enable\_o，观察变量为data\_out\_idct0与data\_out\_idct1。ChipScope波形如图所示。

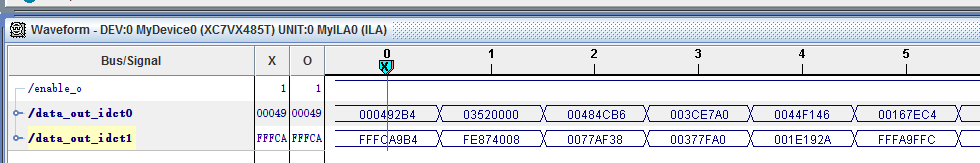


图5.29 LLSE模块输出的ChipScope波形

经过比对，波形形状和数据都和硬件仿真的结果相同。

### 5.4.6 IDCT模块

设置触发变量为idct0\_sync\_o，观察变量为idct0\_out1。ChipScope波形如图所示。

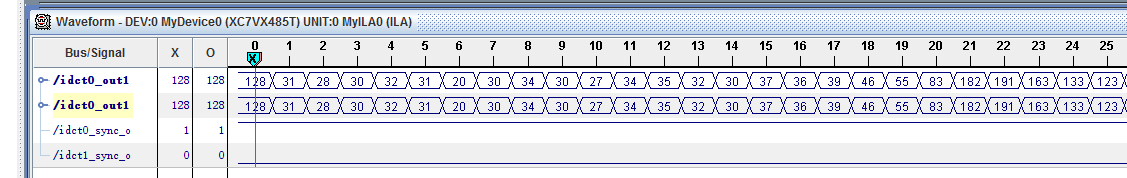


图5.30 IDCT模块输出的ChipScope波形

这里由于idct0\_sync\_o提前一个周期拉高，所以数据应该从第二个即31开始比对。经过比对，波形形状和数据都和硬件仿真的结果相同。

# 6 结论和展望

## 6.1 结论

1、根据SoftCast算法设计了无线视频传输系统，发挥SoftCast 信源信道联合编码的优势，在广播传输的情况下系统能使得每个发送端都能解码出一份较满意效果的视频。

2、利用SoftCast仿真平台得到SoftCast的PSNR性能图，分析SoftCast算法的性能曲线。在低于信噪比门限时具有数字传输所不具有的性能优势。

3、完成了SoftCast系统中DCT/IDCT模块的设计和RTL仿真的正确性验证。在可允许的误差范围内，模块设计正确。

4、完成了SoftCast系统中SoftCast、LLSE过程模块的设计和RTL仿真的正确性验证。出现部分误差，模块功能设计正确。

5、完成了SoftCast系统中解帧、组帧模块的设计和RTL仿真的正确性验证。模块设计正确。

## 6.2 展望

1、继续完善系统，连接射频板以及从摄像头HDMI采集数据，这样真正可以实现从摄像头采集数据，通过FPGA至射频发送数据，接收端通过射频接受然后传给FPGA处理后从HDMI接口输出到电脑或者FPV。

2、将搭建完成的系统应用于VR视频传输。

# 参考文献

[1] 王莉莎.基于SoftCast的无线视频传输优化技术研究[D]. 北京邮电大学, 2015.

[2] Jakubczak S, Rahul H, Katabi D. One-Size-Fits-All Wireless Video[C]//HotNets. 2009.

[3] 孔鑫柱. 无线数模混合传输中功率分配优化技术研究[D].中国科学技术大学,2016.

[4] 武瑛. DCT变换在图像压缩中的应用[J]. 计算机与现代化,2013,(04):103-106.

[5] 陈宝远,吴孟泽,张清喜,等. H.264标准中整数DCT图像压缩算法的优化及实现[J]. 哈尔滨理工大学学报, 2009, 14(5): 27-30.

[6] 赵孝. 无线网络中自适应混合数模视频传输技术研究[D]. 中国科学技术大学, 2015.

[7] Jakubczak S, Rahul H, Katabi D. Softcast: One video to serve all wireless receivers[J]. MIT Technical Report, MIT-CSAIL-TR-2009–005, 2009.

[8] 邓强. 利用CORDIC算法计算平方根及其FPGA实现[J]. 通信技术, 2013 (7): 129-131.

[9] 白宝明. Turbo码理论及其应用的研究[D]. 西安电子科技大学, 1999.

[10]Berrou C, Glavieux A, Thitimajshima P. Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1[C]//Communications, 1993. ICC'93 Geneva. Technical Program, Conference Record, IEEE International Conference on. IEEE, 1993, 2: 1064-1070.

[11]侯立正. 基于软件无线电的16QAM调制解调器设计与FPGA实现[D]. 西安: 西安电子科技大学, 2008.

[12]孔艳芳. 16QAM解调模块的设计实现与性能验证[D]. 重庆大学, 2011.

[13]司马苗,周源华. 基于FPGA的二维DCT变换的实现[J]. 红外与激光工程, 2003, 32(4): 436-439.

# 谢 辞

通过这几个月来的努力，我的毕业论文《虚拟现实（VR）无线视频传输设计与实现》终于完成了，也意味着我的大学生活即将结束。在大学里的四年，我在学习上和思想上都得到了全面的提升，我的进步除了自身的努力之外，更离不开各位老师、同学对我的关心与支持。

我要感谢我的导师吴俊老师。在过去的几个月，吴老师不仅在学业上给我以精心指导，同时还在思想、生活上给我以无微不至的关怀，给与了我很大的帮助，在此我向吴老师致以诚挚的谢意和崇高的敬意。吴老师在我完成毕业论文的每一个阶段，从选题到课题调研，系统的设计，论文的构思和格式调整等各个环节他们都严格把关，给予了我悉心的指导。

另外，我还要感谢实验室项目组的学长学姐们，我在项目中遇到了许多问题都从项目组的学长学姐处得到了详尽的指导，也正是由于他们在学习和生活上对我的关心和支持，我才能够克服重重困难和疑惑，直至本论文顺利完成。

最后，再次对关心、帮助我的老师和同学表示衷心地感谢！在未来的日子里，我会更加努力的学习，不辜负他们对我的帮助和期望。

1. [↑](#endnote-ref-1)