**Lab 3: Packet Transmission Using USRP**

|  |  |
| --- | --- |
| **Author** | Name： 吉辰卿 Student ID: 11911303 |
| **Introduction**  Software-Defined Radio (SDR) is a revolutionary technology in the field of communication. Its important value lies in that it breaks the limitation that communication function only depends on hardware development and realizes a new system that communication function is defined by software. Therefore, as a student of communication subject, it is necessary to have a comprehensive study of software radio. In this experiment, we focus on understanding three typical structures of software radio, and further simulate the working process of software radio through MATLAB programming. In the experiment part, we import the USRP pre-recorded data into MATLAB program, and restore the text of 4QAM and 16QAM pre-recorded data respectively.  **Lab results & Analysis**：  **Theoretical Analysis:**   1. **Software****-Defined Radio (SDR)**   Software-Defined Radio, whose key idea is to construct an open, standardized and modular general hardware platform, various functions, such as working frequency band, modulation and demodulation type, data format, encryption mode, communication protocol, etc., are completed by software. At the same time, the broadband A/D and D/A converter as close as possible to the antenna in order to develop a highly flexible and open new generation wireless communication system. It can be said that the platform can be controlled and redefined by software, different software modules can be used to achieve different functions, and the software can be updated. Like a computer, its hardware can be constantly updated and upgraded. Since various functions of software radio are realized by software, it is only necessary to add a new software module to realize new service or modulation mode. At the same time, it can form a variety of modulation waveform and communication protocol, which makes it also communicate with the old system of various radio, greatly prolong the use of the radio cycle, but also save the cost.  A standard software radio station consists of broadband antenna, front-end receiver, broadband digital-to-analog converter, universal digital signal processor and so on. Its functionality and required interfaces are shown in Figure 1. Mobile user units provide interfaces such as voice, data, facsimile and multimedia through analog interfaces (narrowband A/D and D/A) and optional mixed source encoding. The quasi-real-time and real-time software performs data analysis, processing and transformation between narrowband and wideband data through A programmable processor, and then the transformation between wideband A/D/A and RF data is completed. The base station provides the PSTN digital interface. On the other hand, it provides its own operation and maintenance through all kinds of equipment. Through the business development workstation, it uses the offline software analysis tools to provide the business development environment support for researchers and developers. The placement of broadband A/D/A converters and the degree of software definition of radio functions are important indicators to measure the quality of software radio. The closer the position of A/D and D/A is to the antenna, the more parts of the radio can be realized by software, and the higher the degree of software will be. The ideal goal of software radio is to place the digital to analog conversion device directly behind the broadband antenna and convert the signal directly to digital signal at rf. In this way, all other parts of the radio station can be completed by software, so as to realize the full software of the communication radio station.  20220311021413  **Figure 1 The structure of SDR**  For software radio, we mainly study three kinds of structure: **Low-If,** **Heterodyne and Direct-Conversion(Zero-If).**   1. **Heterodyne Receiver**   The Heterodyne Receiver has been the main structure of receiver design since its first appearance in 1917. Until 2000, Zero-If Receiver appeared, which is suitable for fully integrated implementation.  20220311023059  **Figure 2 The structure of** **Heterodyne Receiver**  In this structure, input bandpass filters are usually used to suppress out-of-band interference signals and prevent strong out-of-band interference signals from blocking low-noise amplifiers. Generally, the input bandpass filter has a wide bandwidth and consists of multiple channels. The image filter is used to suppress the image frequency. The IF bandpass filter after mixing determines the channel selectivity of the receiver and is used to suppress the adjacent signal power. At the same time, the IF bandpass filter is usually used as the anti-aliasing filter at the front end of AD.   1. **Direct-Conversion Receiver**   The main problem to be solved in Heterodyne Receiver is the problem of image frequency suppression. The Direct-Conversion receiver overcomes the problem of image suppression by converting the signal directly to the baseband (0Hz). Its structure is as follows:  20220311023602  **Figure 3 The structure of** **Direct-Conversion Receiver**    The local oscillator frequency (LO) of the zero if receiver is equal to the radio frequency signal frequency (RF), and the mirror frequency is the signal frequency itself. There is no problem of image frequency interference and the image suppression filter and intermediate frequency filter in the original Heterodyne Receiver structure can be omitted. On the one hand, the elimination of external components is beneficial to the single-chip implementation of the system.  As shown in Figure 3, behind the mixer is an analog low-pass filter that acts as a channel selection filter and an anti-aliasing filter for the AD front end. If the channel selectivity of the receiver is completely realized by the filter, the cutoff frequency of the filter is required to be half of the signal bandwidth to effectively suppress the channel interference in the adjacent channel and the farther end. Since the filter works at low frequency, it can be implemented with an active analog filter. Note the amplitude response matching of the upper and lower branches. Active analog filters have limited dynamic range and limited stopband attenuation relative to passive if filters in Heterodyne Receiver.  However, although the Direct-Conversion Receiver structure reduces the problem of image signal suppression, it also brings other problems. These problems are mainly due to the input signal amplification group to be concentrated in the baseband.   1. **Low-If Receiver**   The Low-If Receiver tries to solve the DC bias and Flicker noise problems of the Zero-If Receiver while maintaining the high integration of the Zero-If Receiver. Many wireless standards require that the suppression of adjacent channel interference be relatively weak compared to other channels. The Low-If Receiver makes full use of this stipulation and selects the appropriate if frequency to take the adjacent channel signal as its mirror signal. The main structure of Low-If Receiver is as follows:  **20220311022557**  **Figure 4 The structure of Low-If Receiver**   1. **The Process that USRP realizes the flow of this simulation experiment**   We needs to program transmitter and receiver on MATLAB to realize this simulation experiment by USRP. The process is shown below:  20220311120303  **Figure 5 Transmitter terminal MATLAB programming process**  Through the transmitter programming process, we can see that at the beginning of programming we need to configure the parameters of the SDR. Then we create the object of the SDR as our transmitter. Then we begin our transmission process, which is read and write in a loop through the object of the SDR. Finally, when the transmission is complete (when the loop exceeds a certain limit), we can exit the loop and free the corresponding software radio object to save storage space. The receiver programming process of software radio is as follows:  20220311121017  **Figure 6 Receiver terminal MATLAB programming process**  As can be seen from the figure above, the programming flow of a receiver is similar to that of the transmitter. First we still need to configure the parameters of the receiver, after we can create and configure the receiver objects. Then, as the same way with the transmitter, we keep the receiver object receiving the data in a loop. Finally, when we finish receiving, that is, when the loop has gone a certain number of times, we exit the loop and free the receiver object to save our memory space.  **Lab results & Analysis**：  **Task 1: Text recover with Pre-Recorded data in 4QAM**  **Result:**  20220311025435  **Figure 7 The Spectrum diagram of the received signal (4QAM)**  20220311025100  **Figure 8 Output from the command line window (4QAM)**  **Analysis：**  Since our experiment is to restore the pre-recorded data on USRP, we need to delete some information related to USRP configuration from the source program provided by Dr.Wu, so as to make our program run accurately. The annotated information and important parameters are shown as follows:  20220311030236  20220311030336  20220311030248  **Figure 9 The content of the source program that needs to be** **annotated**  20220311150844  **Figure 10 The important parameters in 4QAM text recovery**  From our experimental results, we can see that the text information in our 4QAM pre-recorded data is successfully recovered in the command line window with almost no garble and a small bit error rate.  **Task 2: Text recover with Pre-Recorded data in 16QAM**  **Result:20220311030747**  **Figure 11 The Spectrum diagram of the received signal (16 QAM)**  **20220311030733**  **Figure 12 Output from the command line window (16QAM)**  **Analysis：**  First, let me show the important parameters in 16QAM text recovery. The annotated content is the same as in 4QAM above.  **20220311144336**  **Figure 13 The important parameters in 16QAM text recovery**  Through the reading and recovery of 16QAM pre-recorded data, we can find that: In each cycle, there will be some error code in the recovery of pre-recorded data, and the bit error rate is slightly higher than that of 4QAM data. In fact, we have analyzed this phenomenon in lab2, that is,16QAM sacrifices some transmission accuracy while improving transmission rate.  In addition, during the implementation of this task, I also found two interesting problems:   1. **As the number of cycles increases, the bit error rate decreases.**   **20220311101323**  **Figure 14 The bit error rate of the first loop**  We can find that there are many error codes in the first cycle, and the bit error rate reaches about 0.13. After several cycles, the error code generated in each cycle has been fixed, and the bit error rate finally stabilizes at about 0.056. I think the reason for this thing is that the receiver will constantly adjust some of its parameters in the loop to reduce the bit error rate of the received content (since every loop is the same content, the receiver can adjust according to the feedback from the last loop).   1. **The receiver's Desired Amplitude value affects the number of errors received.**   In this experiment, we also found that the Desired Amplitude value of the receiver would affect the received error code content. As shown below:  20220311102442  **Figure 15 Receiver parameter Desired Amplitude**  20220311102608  **Figure 16 Error code content in the command line window (Desired Amplitude is 1)**  20220311104537  **Figure 17 Error code content in the command line window (Desired Amplitude is 0.96)**  20220311104941  **Figure 18 Error code content in the command line window (Desired Amplitude is 0.93)**  20220311105629  **Figure 19 Error code content in the command line window (Desired Amplitude is 0.9)**  20220311105713  **Figure 20 Error code content in the command line window (Desired Amplitude is 0.8)**  20220311105758  **Figure 21 Error code content in the command line window (Desired Amplitude is 0.6)**  According to the above results, we found that the value of Desired Amplitude could affect the content of error code. In the range of 0.9-1, different Desired Amplitude also generated different error codes, and the value of Desired Amplitude can also reflect the quality of the received content. It can be found that when the value of Desired Amplitude dropped to 0.6, the received content could no longer be identified with a high bit error rate. Therefore, Desired Amplitude reflects the amplitude gain of the receiver, and its value will greatly affect the received content at a low value. However, in a suitable range (0.9-1), Desired Amplitude only affects the content of local error codes, which is the most important discovery in this experiment. | |
| **Experience**  In the simulation of the experiment, we use MATLAB programming to recover the pre-recorded 4QAM and 16QAM data from USRP. The experimental results show that the bit error rate of 4QAM is smaller than that of 16QAM, which is completely consistent with our theoretical analysis. At the same time, we were able to recover text content in both 4QAM and 16QAM cases very well, based on the fact that we were able to correctly annotate some of the USRP-Related content in the program. Finally, we also found two interesting phenomena during the experiment and analyzed them. All in all, although we didn't use USRP to implement this experiment in the classroom, we still learned a lot from it and felt that the experiment was meaningful. | |
| **Score** |  |