



Two Channel Transmitter and Receiver

EEC 687 Mobile Computing

Project Presentation

By,
Robert Fiske
Malav Shah

EE 687 1 1 2008



Outline

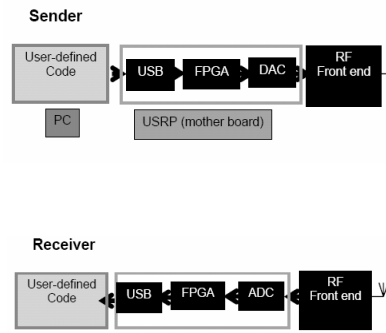
- Introduction
- Project Goal
- Receiver
- Transmitter
- Progress, Problems and Solutions
- References
- Questions



Introduction

USRP : Universal Software Radio Peripheral

- RF Front End Daughter Boards
- ADC-DAC
- User Programmable FPGA
- Programmable USB 2.0 Controller



Introduction (Continue..)

- GNU Radio
 - Open source software toolkit for building software radios
 - Defines the transmission waveform and demodulates received signal
- It contains of
 - C++ classes which implement different signal processing functions
 - Python implement main application programming
 - SWIG works as glue between C++ classes and Python language. It is a Linux package that converts the C++ classes into Python compatible classes



Why SDR?

- All modulation and demodulation is done by software (complex signal processing) instead of electronic circuit so turns all hardware problems to software
- Hardware needs to do only transmission reception of signal
- Software of GNU Radio is free and easily obtainable on the internet
- Only purchased equipment needed is USRP and its daughter boards

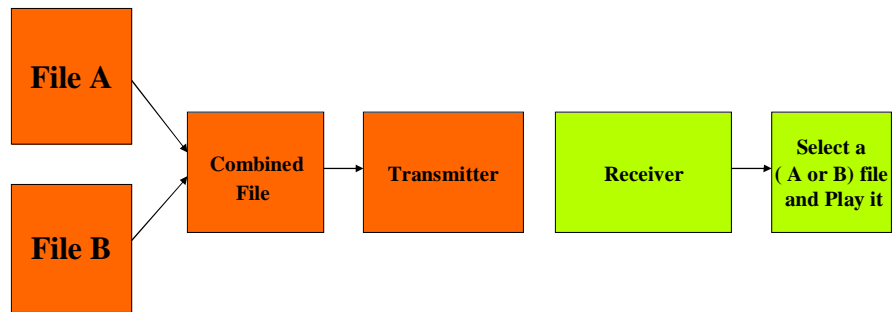


Project Goal

- Transmission of radio signal from server to client
- Server sends two Radio Signals to client
- Radio signal consists of an audio file
- On Receiver side Server will select one of them and play it



Block Diagram



Tasks

- Modify Receive file from USRP according to our needs
- Add a C++ functionality to transmit signal
- Find Decoding scheme to transmit 2 audio files and implement it
- Capture that signal on receiver, select a file play that file



Decoding Scheme for Transmission of 2 files

- We are going to transmit 2 files and on receiver side we choose one of them
- To transmit them we are going to combine these files bit by bit
- For Ex. File A = First nibble A1 A2 A3 A4
- File B = First nibble B1 B2 B3 B4
- Combined Transmission file say File C
- First byte of File C would be A1 B1 A2 B2 A3 B3 A4 B4



Receiver

- Copy and modify:
- originally: `usrp_wfm_rcv.py`
- Changed: `benchmark_rx.py`
 - This performs basic needed functionality.
 - Modify to perform in desired frequency range
 - Modify GUI to suit our needs.



Transmitter

- First attempt transmitting basic tones using demo programs
- Combine benchmark_tx.py and C code using swig to read from file/encode the data and transmit.



Functions

- | | |
|---------------------------|-------------------------------|
| ■ int combine_data | int get_Alimit |
| ■ char* get_combined_data | int get_Blimit |
| ■ void split_data | int get_read_dataA |
| ■ void init_send | int get_file_sizechar checken |
| ■ void init_recv | int get_filesizeR |
| ■ char* get_resultA | char checkend |
| ■ char* get_resultB | char set_end |
| ■ void zero_results | char clear_end |
| ■ void zero_comb | char packet_done |
| ■ void zero_results | char set_packet_done |
| ■ void zero_comb | char clear_packet_done |



Progress

- Transmitter Performs as expected.
- Receiver Switched from output to audio device to File output.



Problem faced and their solution

1. Speed problem with Audio file transfer
 - Switched file based operation.
2. Gnuradio: Dynamic call order
3. Learning to use SWIG and C/Python data types.
4. Problem with transfer of continuous 0s and 1s as python uses strings so 0 data value is end of data

Used 01 as a control flag.



References

- http://academic.csuohio.edu/yuc/mobile08/08_week04_USRP.pdf
- <http://spectrum.ieee.org/oct06/4654>
- <http://www.ettus.com/>
- <http://www.wu.ece.ufl.edu/projects/softwareRadio/#Project%20paper>



Questions



Thank You
