Visible Light Communication with FPGAs

6.111 Final Project Proposal

Zachary Zumbo | Mehmet Tugrul Savran

Overview

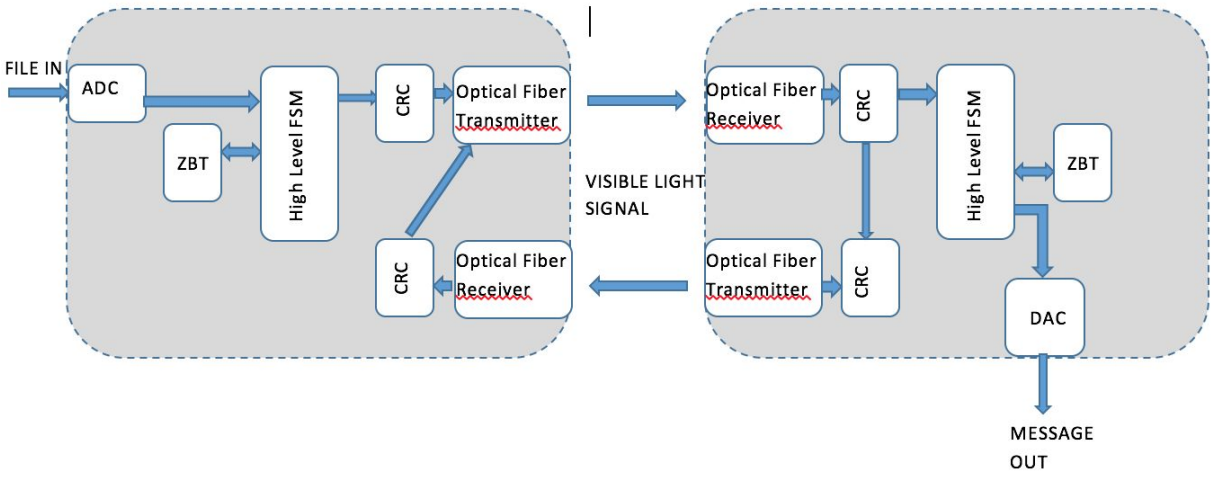
While wireless communication systems have been in use for nearly a century, very few have utilized the visible light spectrum due to its line-of-sight nature and physical constraints. Even with such drawbacks, visible light technology has uses in areas such as underwater technologies and aerial in-flight communication. We seek to circumvent the disadvantages of visible light comms by using a buffering system that will allow streaming data (such as music, radio, video) to be interrupted yet still output in real time. Effectively, the transmitter would take in data via an mp3 player or microphone and transmit to a receiving FPGA with a buffer. The receiving FPGA would begin playback after a certain amount of data has been received. The end effect is seamless visible light communication with interruption control. Our stretch goal is to be able to send a different form of data (e.g. jpeg image).

Design

The transmitting FPGA (tFPGA) will be constantly recording 12-bit audio at 48kHz (half-rate) and sending the data to the receiving FPGA (rFPGA). First, the tFPGA will write the samples from the AC97 module to ZBT memory and then read to the transmitter module. The transmitter module will packetize the data and then convert it to an analog signal using the labkit's built-in DAC.

The rFPGA will then retrieve the data, strip the frames of the packet, run it through a CRC module and store the data in its own ZBT memory. After it has received a critical mass of data, it will begin playing back this audio. The data will be played back with a lag to ensure short gaps do not interfere with playback. Because data can be sent over the fiber optic connection faster than it is being written to the transmitting FPGA, we can constantly be filling the buffer so that a gap of 20 seconds will not affect the playback.

High-Level Block Diagram



Implementation

Synthesizing all the subproblems of the project will be done using modules, each of which are represented in the above block diagram. Each of the labkits will have a copy of all modules and will thus both be able to send and receive data depending on the operating mode.

ADC: This is the analog-to-digital converter that will allow us to sample the incoming audio signal from either a microphone or media player (through line-in). This is implemented in the AC97 module, but we will modify our existing audio lab to sample at 48kHz and at a bit depth of 12 bits. This will allow for values from -2048 to 2047 using a 12 bit signed data bus, and this sampling rate should also cover the entirety frequency range of human hearing (24kHz max). The module will output to the FSM which will store the data in ZBT memory.

DAC: The digital-to-analog converter will convert our internal digitized and packetized audio signal from memory to an analog signal to be used by our transmitter. The input is a 12-bit bus from the fsm and its output is a single wire out, again handled by the ac97 module.

CRC: To determine whether or not we have successfully transmitted a packet of data to the rFPGA, we have a cyclic redundancy check module which will output a data integrity signal to the FSM. If the packet is intact, we send a PACKET_OK signal and then the packet itself is loaded into memory. If the CRC fails, we send a PACKET_ERROR signal and request another packet from the tFPGA. This will be implemented using a separate state in the FSM.

ZBT Memory: This is the storage center for the FPGA that will house our audio data. With two banks of 512kb x 36 bit memory, we will be able to store a maximum of 32 seconds of 48kHz 12-bit audio per bank, more than enough to satisfy the 20 second lag. The first bank will be used to store outgoing data, the second will be used for incoming data. A ram clock module will be implemented that is not shown to satisfy timing constraints of the ZBT memory block. Reading and writing will be done on separate cycles, which is acceptable because we are sampling at such a rate that with a 65MHz clock there are over 1300 cycles between samples from the AC97 module.

Packetizer: This module will act as the intermediary between the FSM and the DAC out to the transmitter. By converting our 12-bit parallel audio signal to a serial signal, we will be able to transmit the data via a fiber optic transmitter. We will use frames to uniquely identify packets and communicate with a standard between FPGAs. We will design different packet types for data and requests so that we can easily determine what state the FSM should be in.

High-level-FSM: The main component of the top-level module is the FSM. This module will determine if we are currently sampling information from the AC97 module and writing to memory, requesting a packet from the tFPGA, reading from memory to play back audio, or transmitting a signal of its own. Essentially this module is the brain that will connect the DAC, ADC, packetizer, and memory to route data in the right direction.

Additional Equipment and Resources

Note that the only additional components to two 6.111 LabKits are:

- AV02 Fiber Optic Transmitter
- AV02 Fiber Optic Receiver

Timeline

The Gant Chart below designates our timeline

Legend: Navy filling correspond to Mehmet Tugrul Savran; Yellow filling correspond to Zachary Zumbo; Red filling corresponds to Group Work.

Development\Week of	11/7	11/14	11/21	11/28
Integration of AV02 Fiber Optic Transmitter and Receiver				
Modification/Testing ADC				
Integration of built-in DAC and simple communication testing				
ZBT Memory Module				
Building/Testing Packetizer				
Implementing CRC				
Development of FSM				
Testing FSM and Communication				
Debugging and More Testing				
Stretch Goals				

Conclusion

Relatively very few means of wireless communications systems exploited the visible light spectrum due to its vulnerabilities to interruption. However, we aim to ameliorate such disadvantage by offering a buffering infrastructure to allow a significant margin to interruption even in streaming data. In addition, being able to see data being transferred “alive” just amazed us, which was our main motivation in pursuing this project idea. Moreover, we will gain substantial exposure to data transmission and communication protocols which we think is must-have knowledge for electrical engineers. In the end, we hope to engineer a bi-directional visible light communication infrastructure with interference control and have loads of fun.