Team Contributions Document

Team member	Student Number
Ilia Sosner	49425259
Cole Shanks	54950860
Guoyu Zhao	65442865
Reuben Singh Joginder	89291884

Team Number: 8
Instructors: Dr. Paul Lusina and Dr. Cristian Grecu
Teaching Assistants: Hamed Noori and Fredy Alves
Course: ELEC 391
Document: Team Contributions
Total Number of Pages: 10
Date: 17th June, 2020

Table of Contents

Team Contributions

Ilia Sosner……………………………………………………………………………… 3

Reuben Singh Joginder ………………………………………………………………… 5

Cole Shanks ……………………………………………………………………………… 7

Guoyu Zhao……………………………………………………………………………… 9

Ilia Sosner

Individual contributions

For the earlier stages of the project that involved simulation I did quite a bit of work to simulate and design the system in Matlab and simulink. There contributions include:

> -Creating square, triangular and gaussian waveforms in matlab and doing the frequency analysis to figure out maximum allowable pulse rates in relation to the spectral power mask. See fft_xx.m files.
> -Generated gaussian pulse waveform data in matlab for use in the transmitter and matched filter, both in simulink and the fpga implementation. See mag.mem, gause_wave.m files.
> -I implemented and tested most of the simulink modules for our simulation including the audio source/processing , text source/processing, transmitter, multipath channel, receiver (with multipath recovery), A-law compression, text and audio sinks.
> - I spent a considerable amount of time trying to model the channel accurately. I implemented a lot of energy calculations for the signal and noise to ensure that I was getting an accurate SNR for the BER analysis.
> -For the Tx/Rx pair I chose to use my own generated waveforms and design instead of built-in simulink blocks.

For the initial design and verification report I wrote the sections on objectives/requirements, overall system design, source modeling, A-law compression design, Modulation design, Tx/Rx design, system verification, source and sink verification, A-law verification, modulation verification, and Tx/Rx verification. I also built, tested and demonstrated the overall system model in simulink for demo-1 and used that as a basis for the FPGA design.

For the FPGA design and implementation portion I completed the following modules and tasks:

-Made sure that what we were implementing on the fpga was representative of the system we were trying to model. This included figuring how many samples we needed to represent our transmission pulse and matched filter and how many clock cycles they should be present for to attain our real-world desired pulse rate.
- Did research on audio compression algorithms and suggested implementing A-law compression.

-Designed a transmitter and multipath receiver module from scratch using pre generated waveform data.
-Designed an accurate channel that included multipath, attenuation and an open source AWGN module.
-Exported modelsim waveforms into to excel and did energy calculations to figure out how much to scale our noise channel by to achieve various SNR levels accurately
-Designed a set of transmit and receiver buffers with serializers/deserializers.
-Generated RAM blocks for various modules.
- Wrote about 10 testbenches of various complexities, from individual modules to full end to end transmission. Extensive system level testing.
-Wrote a BER module and conducted BER tests on the FPGA system.
-Wrote all the top level modules and spent a lot of time figuring out how to synchronize all the parts of the system.
-Synthesized and debugged the full system on the DE1 using signal-tap and the memory viewer.

For this final report I wrote the design and verification sections for overall system, Tx/Rx, channel, and source/sink modeling.

Team Effectiveness

Overall I think our team has been pretty effective at getting the job done. Although the work is not always distributed equally, *most* of the members make a strong effort to be present, participate in discussions, and generally contribute. I think we have maintained largely the same dynamic since module 1.

Other Comments

Distribution and integration of tasks was made fairly easy by the fact that we stayed in touch almost daily and had official scrums twice per week. One of the hardest parts of this project was the huge amount of information to be learned about communications system design and analysis. I overcame this challenge by simply putting every spare hour into this course and trying to really learn the fundamental material. I’ve always been interested in radio communications and spent a year designing and building avionics systems for UBC Rocket so I found this course really useful in cementing some of the more “loose” knowledge I had around this topic. Overall a very rewarding experience.

Reuben Singh Joginder

Individual Contributions

For the early stages of the project after the individual assignment, after contribution to the Communication System Proposal (including attending all scrum meetings, and scheduled events), I was assigned as the lead for Modulation/Demodulation and Compression/ Decompression, which I focused on mostly throughout the course, but also made an effort to understand other modules. For the simulink portion, the Modulation/Demodulation module was fairly easy to implement as it was carried over from the individual assignment, but I carried out some investigations to see if the parameters were still correct since modules got added to the system drastically. The compression/decompression aspect took the majority of my time for this assignment. After consultations on what would be a suitable algorithm to implement, I started studying the Lempel-Ziv algorithm, but had issues with implementing it in simulink (as I was not yet aware of the Matlab Level-2 S function block), which I initially thought was a gap in my knowledge. Hence I started to read various research papers, textbooks, internet resources etc. to understand the implementation, looked at alternative algorithms (DPCM, run length, shannon-fano, LPC, Huffman, etc). Meanwhile, I tried to use existing simulink blocks, such as mu-law compressor and expander to implement a simple working audio compression module, but I wasn’t too confident on the theoretical aspect of the block, hence reached out to my teammates for help. My team member, illia sosner helped me out in understanding A-law and we went with the A-law configuration which was also similar to my mu-law module. This sorted out the audio compression aspect, however text compression still remained. Upon clarifying some of my doubts during an office hour and understanding that a Level-2 Matlab S-function block could be used and how it could be leveraged, I was able to implement run-length encoding in simulink, written from scratch, and this was included in our submissions. For the initial design report, I worked on the overall report design and proof reading, also worked on text compression algorithm design, improved the A-law compression design and verification, and the modulation/ demodulation design verification, Run-length compression for text files verification.

For the FPGA portion of the course, I received feedback from Dr. Paul Lusina that Huffman Coding is probably a better alternative, hence put all my remaining time into perfecting the modules. For the FPGA implementation, I was contributed to:

- Building the BPSK modulation module, including an extensive testbench covering all cases.

- Building a BPSK demodulation module, including an extensive testbench covering all cases.
- Initially I had a run-length encoding prototype that I built after implementing the Simulink model, however this was soon discarded.
- Building an A-law Compression Module, including an extensive testbench.
- Building an A-law Expansion Module including an extensive testbench.
- Building a Huffman Encoder module (which is counted towards our stretch goals), including an extensive testbench.
- Building testbenches that utilize memory for verification, and simulating transmission of actual data.
- Building a Huffman Decoder Module, including an extensive testbench.
- Modifying an open-source low pass filter.

For the second presentation slides, I've worked on Objectives and Requirements, Compassion and Modulation/demodulation.
For the second report, I've worked on thoroughly proof reading the entire report, and consistent formatting (including all footnotes, proper labelling, table of contents, etc). mentioned in feedback. I've also worked on the following sections: Stretch goals, Compression Encoding/Decoding (audio and text), improving modulation/demodulation, Appendices. I've also worked on Compression/Decompression encoding/decoding verification and modulation/demodulation verification.

Team Effectiveness

I think everyone in the team is effective, everyone contributes, wants to learn and works hard. In my opinion, if a Team lead only focuses on their modules, the work will be distributed more evenly, and would greatly reduce the stress on individual members. Also, I think there needs to be more clear communication, for example when a teammate is asking for help, they aren't necessarily asking someone to do their work for them, and are looking for guidance on how to solve a particular problem. The team dynamic was consistent throughout. Overall, the team members were really supportive and a joy to work with.

Other comments

We held regular scrum meetings and communicated alot throughout the term. There have been some miscommunications which were hurdles. Working in a team is always challenging and I think looking at things from other people's perspective was helpful. I've learnt to be more clear with what I want, and learnt alot about communication systems. Issues with team dynamic were usually what you get when working in a team, nothing too serious.

Cole Shanks

Individual Contributions:

- <u>Error Correction Encoding / Decoding Simulink:</u>

For the first part of the project design, I worked on the modelling of the Error Correction Modules in Simulink. As this was the start of the project, we had not yet decided upon what method would best suit the design goals and requirements that we had to meet. First, I tried to implement the algorithm manually by using a simple parity bit check per byte. This was one of the first methods suggested to us in the course and seemed like a good and simple method. I tried building up the algorithm using logic blocks and math functions in Simulink and made a fair amount of progress. However, I then discovered pre-existing blocks for error encoding existing within Simulink and this immediately became clear as the better alternative. The methods I looked at specifically were Hamming Codes, BCH Codes, and Reed Solomon Codes. After some experimenting and research into the methods I concluded that Reed Solomon would be a good choice for our design. I built a simple test environment with a Random Integer Generator, Reed Solomon encoder/decoder, a Gain block to introduced noise into the channel, and an Error Rate Calculation Block. Next, I experimented with inputs and noise levels to see what kind of error correction that we could accomplish with the parameters that we had chosen. I chose N = 256, K = 223 as our parameters. This meant that we would have 32 redundant bits of data or a Code Rate of 12.5%. With these parameters we would be able to correct up to 16 symbol errors anywhere in our message. This seemed like an acceptable code rate and the error correction appeared very good although I would later discover that we could meet our design goals with a method that can correct far fewer errors.

- <u>Error Correction Encoding / Decoding FPGA:</u>

After Demo 1, we shifted our focus onto FPGA implementation. I began work on the error correcting mechanism and started thinking about what additional challenges would be present on the physical board. I started with the intention of using Reed Solomon encoding again as our mechanism of error correction. The Quartus library includes two IP Cores relating to Reed Solomon encoding. The one I began work on was Reed Solomon II. I did a fair amount of testing and ensured that the timing of the input signals was correct in order to encode the data. I generated both modules and began simple testing in ModelSim. Eventually I found out that I would need a license in order to synthesize these modules in Quartus. There seemed to be a path to getting a license for free but I then discovered another option which was the ALTECC IP Core. This was another Core found within the Quartus library but this did not require a license.

It used a Hamming scheme to encode the data and was not a black box like Reed Solomon II. I could actually peer in and see the algorithm which helped me a lot in understanding how the module was functioning. I decided to use a Hamming (72, 64) scheme. This gave us a Code rate of 11% which was suitable for our transmission of data. The block was also combinational which yielded minimal latency. I constructed a simple test system and simulated in ModelSim while introducing different amounts of noise into the channel. After I confirmed that single-bit errors could be effectively corrected, I moved into integrating these blocks into the overall system. This required buffering at the input of the encoder and the output of the decoder in order to meet data rates of the system. I did this in conjunction with Ilia who added them to the overall system and adjusted them such that they met the required rates of data transmission.

Team Effectiveness:

- What we did well:

  I think our team performed effectively because we all shared the common goal of striving to do well in this course and create a design that we were proud of. Our group involved people with various situations. Some on co-op, some taking multiple courses, and some in different time-zones. However, we were able to perform well as a cohesive unit because everyone was on the same page with regards to what we sought to achieve.

- Suggestions for Improvement:

  There is always room for improvement. If I was to start the course again from Day 1, I think our team would have benefited from each member being more actively involved in each other's activities. We always collaborated and brought everything together as a whole, however there were certain specific aspects of each person's tasks that truly only that person understood on a fundamental level. Division of labour is important but I believe we would have benefited from more active communication with the specific details of each person's tasks.

- Changes since the beginning:

  I think one thing all of us learned since Module 1 is the importance and usefulness of Scrum meetings. At the beginning of the course, we conducted these more as a formality and to submit the required weekly scrum info. As the course progressed, I think all of us began to see the value in scheduled weekly meetings. We maintained active communication over discord and met over voice chat on many occasions. However, having the additional weekly meetings at the same time where we would exchange information proved highly useful. It is a practice I intend to continue on projects I undertake in the future.

Other Comments:

- Breaking up test cases into small chunks rather than trying to simulate the top-level module from the start is a good way to ensure the modules are working properly. If they aren't, it's easier to pinpoint the problem.
- Maintaining a weekly schedule and setting goals and milestones. The break between Demo 1 and Demo 2 was large. It was important to maintain a steady workflow such that we didn't have an insurmountable amount of work at the end.
- Active communication with team-members. Communication on a daily basis was critical to our success.
- Come to TUT's and Office Hours with questions. The feedback in the Office hours in the second half of the course proved very useful.
- I learned a lot about Error Encoding. I learned a lot about communication systems as a whole. However, error correction was the task I was dedicated to and so most of my time was devoted there. I found the concept that you could allocate a small number of redundant bits and correct errors over a large array of data bits really cool. The important differences between the methods was also interesting and this is an area of digital design I would like to do more of in the future.

Guoyu Zhao

For the earlier stage of the project after the individual assignment of Matlab/FPGA, I was assigned to work on the Transmitter/Receiver of our project, which I took efforts into. Since we chose Gaussian pulse for our carrier wave for better transmitting quality, it also brought some difficulties on both Simulink and FPGA compared with other pulse shapes.I spent a lot of time researching information online about signal transmission and reviewing the tutorials because I had less understanding of this field.

Individual contributions

Transmitter/Receiver implementation on Simulink
The implementation of Transmitter on Simulink is not challenging. I just multiplied the signal output from the BPSK with the gaussian pulse that we generated by

matlab code. For the receiver, I used the matched filter to remove the noise and use the block integral & dump for calculating the result of convolution. But the most difficult part is removing multipath signals in the receiver. My initial design is adding one more output of the channel, so that the signals can be minused and get the original signal perfectly. But my teammate Ilia said only one output of the channel should be set. I tried to add delays in the receiver and fixed the error when dealing with the multipath signals, but the result was still not good, because I failed to find a proper way to add the delay block. Finally, Ilia found out a method of adding the delay so that we could get the result with little BER.

Transmitter/Receiver implementation on FPGA

For the receiver part I transferred the matlab code into HDL code for generating the gaussian pulse, and multiplied with the modulation output(BPSK signal). The implementation of the receiver is more challenging. I stored the 60 samples (60*12 clk) into FIFO and then calculated the convolution results by multiplying certain coefficients for each sample. However, the biggest challenge for me is to synchronize all the clk, when removing the multipath signal, and FSM of counting clk when storing 60 samples in advance before adding the delays. Although the multipath signal can be partially removed, the results seem not good enough. So we finally chose Ilia's version of the receiver for better matching with other modules.

Team effectiveness

I am really proud of my team because we overcame many challenges together. Reuben has Coop stuff, and I am not in the same time zone with my teammates, we set the scrum meeting time suitable for everyone in the team. And everyone participated actively. My teammates helped me a lot when I faced some difficulties and taught me with patience. Finally we made this whole project done.

Other comments

I learned a lot of useful information about the sigal in this course by taking the courses, searching information subjectively and implementing what we learned into our project. Also, taking the office hour is really useful for solving questions. Teamwork is also important when doing the project. Without my teammates' help I may not understand the stuff well and not implement correctly. Finally I really want to thank my teammates for helping me.