Experiment # 3

Baseband Pulse Transmission - Part II

1 Purpose

In the previous experiment, you observed the effects of inter-symbol interference on baseband pulse transmission. By varying the bandwidth of the channel (represented by a filter), you increased or decreased the effect of ISI. This time your task will be to design both the transmit and the matching receiver filter for a given noisy channel. Then you will vary the amount of noise in the system and observe the consequence of that in the eye-diagram. This experiment might *s*eem long, if your knowledge of `Matlab` is a little limited. Hang in there.

At the end of this experiment, you should have a better understanding of:

- The choices for shaping a pulse with the transmitter filter;
- How to design a matching receiver filter;
- The effects of noise on a baseband pulse transmission;

2 Background Reading and Preparation

The relevant reading for this experiment can be found in ([1] and [2]). Some extra details can be found in the course notes or the lectures.

Before coming to the lab, complete the **lab preparation** and hand it to the T.A.. At this time you should be comfortable enough with the tools utilized in the lab. You are welcome to come to the lab anytime during regular office hours and build models of your own communication systems.

3 Equipment

Hardware:

1. One Signal Generator;
2. One Two-Channel Oscilloscope;
3. One TMS320C6711DSK Module (the daughtercard is **not** used in this experiment);
4. One notebook workstation;

5. Two Coaxial cables BNC-to-BNC;

Software:

1. `Matlab` Release 12
2. `Simulink` with ECE417 Toolbox
3. `Code Composer Studio`, v.2.1

4 Experiment

The approach taken will be the usual "design, simulate and build". This time, however, you will be free to make decisions related to the design of the different parts of your system. Your decisions **must be** backed up by theoretical concepts. Since you should already be familiar with `Matlab` and `Simulink`, this outline will not provide you with the step-by-step directions. Like in the last experiment, you will **not** utilize the audio daughtercard. The CODEC utilized will be the AD535 present on the DSP main board, which utilizes a sampling frequency of 8KHz. The source of bits is done by means of a sequence generator on the DSP target.

All results are to be reported in the spaces provided in this outline. Make sure that the T.A. verifies every result you record.

4.1 Simulation of Baseband Pulse Transmission

Remember, you will *only* utilize the blocks found on the ECE417 library. The bit generator utilized before has been expanded to generate a longer (still periodic) sequence of bits. In this experiment you will utilize upsampling factors of 4 and 10.

On your new model, put together a communication system as in the previous experiment. Remember that you have to modify your data from a sequence of 1 and 0 to 1 and -1 (or any mapping that you desire – remember, you will have to justify your results). You are to utilize sampling rate of 1/800 where necessary, and an upsampling factor of 10. The filters come from the "FDA Tool" block, provided by `Simulink`. Later you will be asked to add noise to your system. This will be by inserting a "Gaussian Noise Generator" to your model. The generator is found on the ECE417 blockset.

Set the low-pass filter representing the channel to a cutoff frequency of 3900Hz. You can choose the order of the filter representing the channel as you please, just make sure you write down the details of the filter (cutoff, type, order) when you submit your report to the T.A.. Place scopes before and after the transmitter filter, after the channel and after the receiver filter. You will be asked to compare the "original" data (i.e., before the transmitter filter) with the received data (which is after the receiver filter). This is not to be done visually; rather, you will use `Matlab` to display your data for comparison. In order for you to have the data sent to `Matlab` from your model, you need to place a "sink" called "to workspace" (look for it on the ECE417 blockset) at different points in your model. Place one after the upsampler (or before the transmitter filter) and one after the receiver filter. Double click on the blocks to set the parameters properly: use a type "array" and use a variable name of your choice (they will be reshaped and renamed, so use a bogus name for now).

You might have noticed that there is no mentioning so far of the details about the transmitter and receiver filters. There is also no reference to noise. That is because your task will be to design both filters as to a) shape the pulse properly for transmission and b) match the receiver filter to optimize the reception in the presence of noise. For now, however, make sure that your system is working as it should, and that the variables are being passed to the workspace as they should. Place any low-pass filter (say, of order 20, with cutoff frequency around 3500Hz and run the system. Now go into the `Matlab` space and type "`who`". You should see the names of the variables you have created. You should also check for their size, by typing "`size(bogus_name)`". You do not need to use all the points that were sent to `Matlab` in this format "array". You need to reformat them appropriately for the rest of this part of the experiment. Type:

```
good_name=bogus_name(1,1:10000);
```

This creates a 1 by 10000 matrix (actually, a vector) named with the good name you chose for your variables, which holds 10000 points of your bogus_name variables. If you want to plot the 100 first elements of the two variables on the same plot (the second one in red), type

```
plot(first_variable(1:100)), hold, plot(second_variable(1:100),'r')
```

One last detail: the "FDA Tool" in `Simulink` allows you to import coefficients for an arbitrary filter that you have designed. In order for you to do that, you must go under the menu "file/import" and input the name of your variables there. Either that or you can type in the values on the blank spaces provided.**Do not forget to give unique names to everything you design, so that during the experiment they can be called easily into play**. Some more `Matlab` tips will be given when you need them; some will be given on the chalkboard at the time of the experiment.**Here we go.**

Be prepared to answer questions asked by the T.A. You will be marked on these answers as well.

- *Choose a transmitter filter to shape your pulse appropriately. Describe the impulse response of your filter below, and justify the reason for your choice.Do not forget to store it in* `Matlab`.

- *Based on your choice for a transmitter filter, and your knowledge of the channel, design the matched receiver filter. Detail the procedure below and present the impulse response of the filter designed. Store it in* `Matlab`*.*

- *Run it after you deem your design acceptable. Now go into* `Matlab` *and plot the two variables. Save the plot. Draw below what you see (or show the plot on the screen to the T.A.). From what you are receiving, can you recover the data you have sent?*

- *Now add noise to your channel, and give some attenuation to the noise added (put a gain block with a gain of 0.1 right after the noise source). Run the simulation again, go into* `Matlab` *and plot the two new variables. Save the plot. Draw what you see (or show it to the T.A.). Can you still recover the data? Change the noise gain to 0.5, run and compare the results.*

4.2 Downloading the System

Now the complicated part is done. Prepare your system to be built and downloaded into the DSP target. Start by changing the noise gain to 0.01 and removing all scopes and "to workspace" blocks. Make sure that the sampling rates are compatible with that used by the board. Now place the c6711DAC after the receiver filter. Remember that you are using the CODEC that is located on the main DSP board, not the one on the daughtercard. You know how to proceed from here.

Continuing the experiment:

- *Run your system and display on the oscilloscope an eye-diagram, as in the previous experiment. Draw below the diagram, explaining the main features observed*

- *Change the noise gain to 0.1, and download it again (if you are a hard-core type, go into the C code, change the parameter there, recompile and voilá). Is there any change to the eye-diagram? Specify the changes below.*

5 Going The Extra Mile

Now you have seen ISI and have dealt with noisy channels. The real part that has not come into your model is the equalizer. You can try to design another filter that "compensates" for the deterioration (in terms of frequency, not noise) introduced by the channel. If you know the channel, you will try to design a filter that "flattens-out" the channel within the band of interest. If you don't know the channel, you will try to do it "adaptively". Say, then, that you have a channel that introduces a large attenuation right in the middle of your band of interest. What should you do to correct that? Now what if the attenuation that the channel introduces varies with time?

6 Conclusion

In this experiment, you excercised your knowledge of the theory by choosing some of the parameters of your communication system. You gave your optimum shape to the transmitted pulse, you designed the receiver filter to match your choice for the transmitter filter given a known channel and you observed the effects of a noisy channel on baseband pulse transmission.

References

[1] B. P. Lathi, *Modern Digital and Analog Communication Systems*, 3rd Edition. New York: Oxford University Press, 1998.

[2] S. Haykin *Communication Systems*, 4th Edition. Toronto: John Wiley & Sons, Inc., 2001.