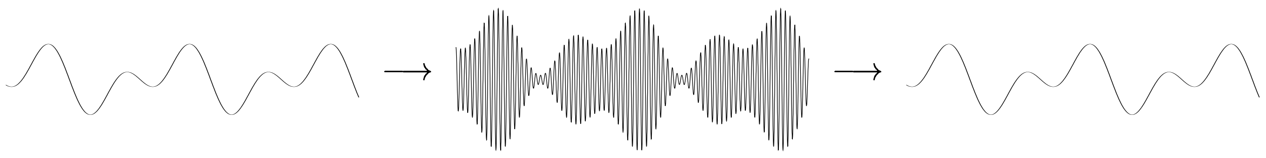
Experiment # 3

Amplitude Modulation and Demodulation



1 Purpose

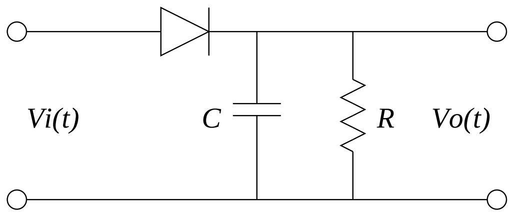
Amplitude modulation (AM), still widely used in commercial radio today, is one of the simplest ways that a message signal can modulate a sinusoidal carrier wave. The purpose of this lab is for you to gain familiarity with the concepts of amplitude modulation and demodulation. This will be done in three main steps:

- First, an amplitude modulation system will be created and simulated using `Simulink`.
- Second, this system will be implemented on a TMS320C6711 DSP platform. Similarly to an AM radio station, the modulating signal will be an audio signal.
- Third, the AM signal will be demodulated by an envelope detector and low-pass filter combination, which you will implement with discrete analog components on a prototype-board.

At each of these three steps, you will be required to change various system parameters and observe the consequences of the changes.

2 Background Reading and Preparation

Amplitude modulation is one type of continuous-wave modulation, covered in [1, Ch. 2]. *Before* coming to the lab, you are encouraged to read [1, Section 2.2], and also the principles of operation of the diode envelope detector circuit shown below (see, e.g., [2, pp. 168–169]).



A diode envelope detector

Before coming to the lab, you should be able to:

1. write an equation for an AM-modulated signal, and understand the significance of all constants in the formula;
2. understand the concept of overmodulation;
3. understand how to choose the values of R and C in the envelope detector to achieve an effective AM demodulator;
4. understand how the AM-modulated signal is described in the frequency domain.

In order to perform this experiment effectively, a good understanding of `Simulink` and `Matlab` is required.

3 Equipment

Hardware:

1. One Signal Generator
2. One Two-Channel Oscilloscope
3. One TMS320C6711DSK Module (with audio daughtercard) attached to a workstation
4. Four Coaxial cables; two with BNC terminations and two with BNC/probe terminations (for use with Prototype board)
5. One prototype board
6. One diode (2n4148?), One Capacitor (?), One Resistor (1KOhm - black, brown, red)

Software:

1. `Matlab` Release 12
2. `Simulink` with ECE4xx Toolbox
3. `Code Composer Studio`, v.2.1

4 Experiment

The experiment is divided in three parts: the design and simulation of an AM modulator in `Simulink`; he download and test of the designed system on a DSP platfom; and the demodulation of the signal with analog, discrete, components.

4.1 Designing and Simulating an AM Modulator

Upon running `Matlab`, click on the `Simulink` icon. This will open the `Simulink` Library. In this library, you will find the ECE416 blockset and all the blocks needed to perform the experiment. Your task is to identify them, and to change the parameters throughout the experiments.

Based on the block diagram that you have reviewed for an AM modulator, build your model. For the initial simulation, use a discrete-time sine wave generator as input, and two oscilloscopes (not floating oscilloscopes) for output. One oscilloscope will be a time-domain scope and one an FFT-based scope. Use an input signal of 1KHz. The input signal must be added to a DC component and this combined signal will be multiplied by the carrier frequency. Use a DC component of 3. The carrier frequency is generated by another discrete-time sine wave generator, this time generating a much higher frequency (use 12KHz – this number is compatible with the sampling frequency used by the DSP board).

Run your simulation, and observe the result on the `Simulink` oscilloscopes.You are required to adjust the simulation parameters to "discrete" prior to running the simulation of your model. You may be required to resolve some issues regarding signal shape and how it is handled by `Simulink`.

You should observe an AM signal similar to that presented in [1, pp. 89 and 91]. Observe and report the amplitude of the modulating signal and the carrier signal. Modify your system to simulate a modulating index greater than 1. Report how you did it and what it means, based on the theory reviewed. Observe and report the signal on the FFT-based scope. Interpret the results based on the theory. Vary the input signal (triangular wave, square wave), observe and report the results.

4.2 Building and Running Your Model

In this part, start by substituting the input and output blocks, respectively, by the ADC and DAC blocks (for the C6711) found on the ECE416 Toolbox. The sampling frequency for the board you are using is fixed at 48KHz, and all input and output parameters are pre-set, so you do not have to change anything on the ADC and DAC blocks. The DAC will be your time-domain output, and you must remove the FFT-based oscilloscope. The frequency domain representation of the output signal will be observed through `Matlab`.

Set the parameters on the original blocks (DC, Carrier sine wave generator) to the original values of the previous section, since the frequency-domain output will be done through a real-time data exchange channel. Also, since you will be dealing with real electric signals now, connect an oscilloscope to an output port (an output BNC connector on your target hardware) and a signal generator to an input port. Ideally, you should connect the output signal from the target to one channel on the scope, and split the input signal to have it going to the target as well as to the second channel on the scope. By doing this, you can have both your input and your output signals displayed on the oscilloscope. Use as a standard input signal a 1Vpp, 1KHz sine wave.

Make sure you modify the `Simulink` build option parameters to make your model compatible with the target hardware. The parameters to be set are found under tools/real-time workshop/options, found on the command bar of your `Simulink` model. By selecting "options", a window will appear to you. Follow this procedure:

- Select the "Solver" tab. Click on "Solver Options" and select type "Fixed-Step";
- Next, select the "Real-Time Workshop" tab. Click on the "Browse" button on the Configuration area. A new window will appear, with a list of "`.tlc`" files. Select the "`ti_c6000.tlc`" file and click on the OK button.
- Under the same "Real-Time Worshop" tab, go under Category and look under "TI C6000 Target Selection". You should have the C6711DSK selected, and must not change any other selections.
- Click on OK and you are ready to download your model onto the DSP platform.

Build the model. You can do this either by going under Tools/Real-Time Workshop/Build or by pressing `Ctrl-B`. At this stage, progress messages will appear on the `Matlab` Command Window. If any error occurs, you will be showed a new window with the specific details of the error. Try to resolve it, and if you cannot proceed, ask for assistance from your Laboratory T.A..

After the code for your model has been generated, `Matlab` will load it onto Texas Instrument's DSP programming environment, called Code Composer Studio. This program will be opened in a new window, and progress messages will appear indicating that your model has been turned into a "Project" and that the compiled and assembled project is being loaded onto the DSP platform. The program will run automatically. A window will appear within CCS, which is the "Disassembled" code that is running on the platform.

Now that the project is visible in Code Composer Studio, explore the files avaliable on the project tree (left-hand side). Take a careful look at the code, in both C files and H files. From this brief survey, you should have a picture of how the project was generated. You can run or halt the program as you wish. Try to identify the routines in which the many parts of your model are generated in software, such as the carrier sinusoid, the DC component, etc.

1. **Time Domain Results**

   Using the oscilloscope, measure and report the characteristics of the output signal. Make a comparison with the signal observed during the simulation stage. Using the signal generator, vary the amplitude of the input signal, observe and report the changes in the output. Return to a 1Vpp, 1KHz sine wave input.

   Now go to your `Simulink` model and alter the DC constant added to the input signal. From a constant value of 3, change it to 1 and then to 0. Every time you change the DC constant on your model, you must re-build it. Follow the same procedure as explained above (i.e., press `Ctrl-B`) and wait for the process to be finished. Run your project after it is done compiling and assembling (it will run automatically). Observe and report the results after each change. Explain the observed changes on the output signal according to the theory.

2. **Frequency Domain Results**

   In the simulation-only part of this experiment, you included an FFT-based scope provided by `Simulink` to see the frequency domain representation of the output signal. Since now you have a system generating actual signals, you have two options to visualize the output signal in the frequency domain: one is to attach a spectrum analyzer to the output of the target board; the second is to use the digital signal processor to send data back into the `Matlab` workspace, and this data will be the same data sent to the DAC, which eventually will be seen in the time domain on the oscilloscope used on part (a) above. In this part you will request that data be read from the DSP memory, and have that data written into a `Matlab` variable. Since the data come from a limited space in memory, the `Matlab` variable will be a vector of limited length. In your case, the length is pre-determined to be 1024 samples in length. When that is done, you can manipulate it mathematically to visualize it. You can also repeat the procedure as many times as you wish, to visualize 1024 points of data as time goes.

   The procedure to be followed is presented below:

   - Create a "`.m`" file with the following contents:

     ```
     cc=CCS_Obj;
     x=read(cc,address(cc,'storage_array'),'int32',1024);
     y=double(x);
     z=fft(y,1024);
     subplot(2,1,1)
     semilogx(abs(z))
     axis([0 512 0 5e11])
     subplot(2,1,2)
     plot(y)
     axis([0 150 -1e9 1e9])
     ```

   - Run the file
   - Modify the file and run again as you wish, to visualize the data in the frequency domain.

You may notice that the magnitude of the numbers read is very large. Remember that they are being read from memory, and not only are a result of some computations, but must be scaled properly to be sent to the D/A converter. The numbers seen obviously do not represent voltage; they must be scaled down. You can find the scaling factor by looking at them in the time domain (in `Matlab`) and comparing the numbers seen with those read on the oscilloscope (that is, after passing through the D/A converter). You can then correct the program to represent the actual voltage values.

On your report, include a `Matlab` plot of the signal, showing the distinctive characteristics of the AM signal, for every change you make to the model. Run initially with the original values of the model (i.e.: a DC constant of 3, a carrier frequency of 12KHz and amplitude 1 and an input sinusoid of 1Vpp, 1KHz). Collect data into `Matlab`, plot the Frequency Domain data and report your results, highlighting the signal characteristics. Repeat the procedure for a carrier frequency of 8KHz. Finally, change your model for a modulation index greater than 1 and repeat the procedure. On your report, explain how you have achieved such modulation index and present results, highlighting the differences relative to the previous results.

4.3 Demodulating the AM Signal

Your modulator runs fine, but your intention is to transmit the desired signal and receive it after it goes through modulator, amplifiers, antenna, channel and receiving antenna. You must demodulate the received signal to extract the modulating signal, which is the signal containing the information (in your case, this information is a 1KHz, 1Vpp tone).

You are to build and test an AM demodulator which is made of an envelope detector. The envelope detector is comprised of a diode and an RC low-pass filter, as explained in [2, pp. 168–169]. The calculation of the components should be done prior to the experiment. Your task is to attach the output signal from the target onto the input of your envelope detector, and the output of your envelope detector to the oscilloscope. Include in your report the schematics of the detector, as well as the calculation of the components. Explain the result observed on the oscilloscope.

5 Going The Extra Mile

Another extra adventure you may want to try is to modify the C code of the generated project and make your own project. This will help you to understand many details of writing code for real-time applications. You will notice that you do not need to go all the way back to the model every time you need to modify a parameter. However, sometimes looking at the code and modifying it is not as straight-forward as looking at a block diagram, changing a number and regenerating the code through the Real-Time Workshop. As you become familiar with coding in C for the target hardware, you may not even need to model it first on `Simulink`.

Code that is automatically generated can be very hard to read, particularly if the automatic code optimization feature is used.

6 Conclusion

In this sample experiment, you were presented with the key issues involved in designing, simulating and implementing an AM modulator and demodulator. This was done by using a simulation model, a DSP platform in which the model was implemented and an envelope detector for the demodulation of the signal. The experiment intended to guide you through the steps necessary to achieve a practical understanding of the concepts studied not only in the theory, but also in other courses of the curriculum.

References

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

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

[3] The Mathworks, Inc., *Using Simulink.*

[4] The Mathworks, Inc., *Real-Time Workshop.*