

FM Exercises

The following sequence of exercises is designed to take you through a process leading to the DSP implementation of an FM modulator and demodulator. The algorithms will first be formulated and tested using a stable software development tool like MATLAB or, at a lower level, C. Once the algorithms have been programmed and tested, you can relatively easily transform them into DSP C and assembly code. You should structure your MATLAB or C programs to be similar to how you would expect to implement them on the DSP.

FM Modulator Design

1. Do a MATLAB or C simulation of a discrete-time FM modulator using floating point arithmetic except for naturally integer computations like with indexes. Let the baseband message be a 100 Hz sinewave and the carrier frequency be 1 kHz. Start with a modulation index of $\beta = 5$. Use a sampling frequency that is high enough to get nice signal plots. Use the C library sine and cosine functions.
2. Plot the modulated FM signal. Try several values of β .
3. Plot the baseband inphase and quadrature components of the FM signal. Explain why they do not have the “accordion” like stretching and shrinking of the passband signal plotted in the previous item.
4. Use the FFT or PSD functions of MATLAB to plot the spectrum of your FM signal and compare the results to the theoretical ones. Experiment with different values of β .
5. Convert your MATLAB program to a C program that runs on the DSP and send the modulated signal samples to the D/A converter. Use a sampling rate of 8 kHz. Observe the FM wave in real-time on the oscilloscope.
6. Now attach a signal generator to the A/D input and use these samples for your message samples, $m(nT)$, rather than generating them internally in your program.
7. Computing $\sin \theta$ and $\cos \theta$ requires a significant sequence of arithmetic operations and is not efficient for DSP implementations. Therefore, modify your modulator program to look up the sines and cosines in a table. A table containing 256 sine samples equally spaced over 360 degrees should be adequate. The angles should then be converted to integers such that 64 corresponds to 90 degrees and the integer angle representation becomes an index into the table. You can use the identity $\cos \theta = \sin(\theta + \pi/2)$ so only a single table is required. You can still use floating point arithmetic for this exercise. Send the FM signal samples to the D/A converter and observe the signal on the oscilloscope. Estimate the speed improvement with table look-up relative to using the library sine and cosine functions by using the Code Composer profiling tools.
8. The TMS320C54x DSP has a 16-bit integer architecture. Therefore, floating point arithmetic is very inefficient. Even 32-bit normal C integer arithmetic is inefficient. Most commercial products use 16-bit integer arithmetic to allow as many functions

as possible to be realized within a single DSP. Modify the DSP program of the previous exercise to use 16-bit (short) integer arithmetic wherever possible. This includes converting the values in the sine table to 16-bit integers.

9. To further increase the computational efficiency of a DSP program, numerically intensive computations should be performed by hand coded assembly routines that are called from C. The overall control code is still often implemented in C for ease of generation, testing, and portability. Modify your previous program to call assembly routines to find the modulated signal samples.

Implementing an FM Discriminator

1. First do a MATLAB or C floating point simulation of the discrete-time FM discriminator shown in the FM lecture notes. You can use the program REMEZ.EXE in the directory C:\DIGFIL\REMEZ to design the Hilbert transform and differentiation filters. First use the passband FM signal samples generated by your modulator simulation. Then bypass the demodulation step and use the baseband inphase and quadrature samples generated in your modulator simulation.
2. Convert your simulation into a 16-bit integer realization for the DSP using assembly routines for filtering and any numerically intensive functions. First, generate the signal samples internally within the program and send them to your demodulator within the program.
3. Next, team up with an adjacent group. Use the DSP board in one computer to modulate a message taken from a signal generator and connect the codec output to the codec input of the other computer. Demodulate the signal with the DSP in the second computer and display the result on the oscilloscope.

In the case of the TI EVM, use the passband FM modulated signal. With the TIGER 549/PC boards, first use the single passband modulated signal and then use the complex envelope consisting of the baseband inphase and quadrature components.

4. When your modulator and demodulator for the TIGER 549/PC is working, connect the I and Q outputs of the codec to the I/Q modulator in the WFE up-converter. Observe the ISM-band RF output on the spectrum analyzer. Then reattach the antenna and transmit the signal to the WFE down-converter. Attach the outputs of the I/Q demodulator (which is actually the same as an I/Q modulator) to the stereo codec input of another computer and run your FM discriminator program. Display the demodulated signal on the oscilloscope.
5. As another alternative for demodulation, disconnect the I/Q demodulator from the WFE down-converter and attach the M1-EMSR-2 mixer, A1-02111 amplifier, and F1-L-0020 lowpass filter. Program the down converter oscillators to bring the RF input signal down to a 10 MHz IF output frequency. Then use the SIGTEK ST-114 board which contains a Harris HSP50214 programmable digital downconverter to do the FM demodulation.

6. We also have a Harris HSP50215EVAL board which has four HSP50215 digital up-converters on it. These chips can accept digitized I and Q inputs and combine them to generate an IF signal. However, this board was designed primarily to demonstrate the HSP50215 capabilities and not as one to easily interface with DSP's. A very challenging and useful project would be to design and fabricate an interface between the DSP boards and this board.