**EE 445S Real-Time Digital Signal Processing Laboratory – Prof. Brian L. Evans**

**Lab 6 Instructions – Part 1 – QAM Transmitter**

**1. winDSK demonstration**

Connect your DSP board with the PC through the serial cable. Run winDSK8 on the PC, and then follow the instruction on page 302-306 in the real-time DSP book to run the “CommDSK” application. Check the result for root raised cosine pulse shape filtering.

**2. Getting the raised cosine filter coefficients using MATLAB**

Design your pulse shaping filter using ‘rcosine’ command in MATLAB according to the following specifications:

* + - Input symbol rate: 2400 Hz (fsym, Fd in Matlab)
		- Output sample rate: 48 kHz (fs, Fs in Matlab)
		- Type of filter: fir/normal
		- Alpha (the excess bandwidth factor): 0.8 (, r in Matlab)
		- Truncation limit: 4 symbol periods (delay in Matlab)

**3. DSP implementation (Code Composer):**

**Task a):**

* Create a new project with the source files in C:/CD/code/chapter\_18/ccs/QPSK\_Tx
* Use the file “impulseModulatedQPSK\_ISRs.c”. (Exclude the file “impulseModulatedQPSK\_ISRs\_revA.c” from build for this task).
* Run this program and Check the result using oscilloscope.
* Understand the codes in the file “impulseModulatedBPSK\_ISRs.c” based on the explanation in section 18.4.1 of the textbook. Especially understand how the in-phase and quadrature phase components are generated.
* Show the screenshot in your lab report.
* Now, try to benchmark this code. Look at the number of clock cycles required to generate the output by inserting two breakpoints.

**Task b):**

Now we will modify the code a little bit:

* In the original code the symbols are generated using the “rand()” function. Now try to generate the random symbols using the scrambler that you created in Lab 4. This time, we will be using the polynomial **1 + D18 + D23**. The **initial state** of the scrambler is **5**.
* Design your own raised cosine filters follow the specifications given in task 2. Export the coefficients into the source file “coeff.c” in the CCS project. **Note that the length of the filter is different from what we have in the original code. So please modify the necessary code to make it compatible with the new filters.**
* Run the project, show the screenshots.

**Task c):**

* Create a new project with the source files in C:/CD/code/chapter\_18/ccs/QPSK\_Tx
* Use the file “impulseModulatedQPSK\_ISRs\_revA.c” instead of “impulseModulatedQPSK\_ISRs.c”.
* Run this program and Check the result using oscilloscope.
* Understand the codes in the file “impulseModulatedBPSK\_ISRs\_revA.c” based on the explanation in section 18.4.2 of the textbook.
* Show the screenshot in your lab report.
* Now, try to benchmark this code. Look at the number of clock cycles required to generate the output by inserting two breakpoints.
* Compare this value with the one which you got in Task a).

**Task d):**

Repeat Task a) for 16 QAM. In this case you need to generate 4 random bits, instead of 2. Additionally, you need to write your own look up table for the symbol mapper, analogous to ‘const float QPSK\_LUT[4][2]’ used in QPSK. Please refer to Prof. Steven Tretter’s slides at <http://users.ece.utexas.edu/~bevans/courses/realtime/lectures/laboratory/c6748winDSK/lab3/index.html> to take a look at the constellation diagram of 16 QAM and the associated Gray Coding.

**Task e):**

Repeat Task b) for 16 QAM. You need to call the PN sequence generator function 4 times to generate 4 random bits.