Problem Set 2 Addendum

Data Sets

The problems make reference to two different data sets. To have the TX MAC generate these data sets, use the following parameters:

Data Set 1	**Data Set 2**

`NUM_MESSAGES = 60`
`MESSAGE_SIZE = 20 (bytes)`
`RATE = 2 (12 Mbps)`
	`NUM_MESSAGES = 60`
	`MESSAGE_SIZE = 20 (bytes)`
	`RATE = 3 (24 Mbps)`

Channel Models

The problem set also makes reference to two different channel models. The parameters for these two models are specified below:

Channel Model A	**Channel Model B**
`AWGN_VARIANCE = 0.005` `DELTA_F = 0`	`AWGN_VARIANCE = 0.02` `DELTA_F = 0`

Do not use the multi-path parameters (`MP_AMP[]` and `MP_DELAY[]`) except where directed in Problem 4.

Getting Started

The code for the behavioral model you will use in the problem set is located in the course locker at:

`/mit/6.973/homework2`

I recommend that you copy this entire directory (using **`cp -R`**) to your home directory so that you can make changes to the code and schematics. To view the modules in Cadence and get them ready to execute, follow these steps:

1. Make sure you have added the commands in the CppSim/Cadence Tutorial to your **`.cshrc.mine`** file and that you have made copies of the **`VppSim`** and **`menus`** directories as directed in the tutorial.
2. In addition, add the following command to your **`.cshrc.mine`** file:

   ```
   setenv LD_LIBRARY_PATH /usr/lib
   ```

3. Edit your **`VppSim/cds/cds.lib`** file and add the following lines:

   ```
   DEFINE 80211a_transmitter [your path to homework2]/homework2/80211a_transmitter
   DEFINE 80211a_receiver [your path to homework2]/homework2/80211a_receiver
   ```

4. Run **`icms &`** from your **`VppSim/cds`** directory (just like in the tutorial)
5. Call up the schematic for the transmitter_harness cell in the 80211a_transmitter library
6. Go to Options → VppSim to bring up the VppSim dialog box
7. Make sure CppSim is selected and click "Netlist Only"
8. Click on "Edit Sim File" to create a test.par for this simulation. Fill in the following parameters:

```
a.  num_sim_steps: 50e5
b.  Ts: 1/10e9
c.  probe: clk
```

You are now ready to compile and run the simulation. To do so, first **`cd`** into your **`VppSim/SimRuns/80211a_transmitter/transmitter_harness`** directory. To complete the first step of the compilation, run **`vppsim -cpp`**. To complete the second step of the compilation and run the simulation, run **make**.

If the simulation is working properly, you should see a long stream of debugging output being generated; you can grep through this output to view it in a more meaningful way as described later. If you want to run the simulation again, run **`./test`**. If you change some of the module code, you will need to recompile with **`vppsim -cpp`** and **`make`**, and if you change anything in any of the schematics, you will need to re-netlist from the Cadence VppSim dialog box as well as recompile.

Viewing and Editing the CppSim Code

The CppSim code for each module is located in:

`[your path to homework2]/80211a_[library name]/[module name]/cppsim/text.txt`

where **`[library name]`** is either **transmitter** or **receiver**, depending on which high-level block the module resides in. For an explanation of the different sections in CppSim module code, see the CppSim/Cadence Tutorial or the CppSim documentation.

Grepping the Simulator Output

To make any sense of the simulator's output, you will need to grep it for the modules you are interested in. Each module will generate its debugging information for each cycle on a line preceded by it's debug code. The debug codes for the modules are listed in the following table:

Module	Code
TX MAC	A
Analog TX	B
Channel Model	C
Analog RX	D
RX MAC	E
TX Controller	F
Scrambler	G
Encoder	H
Interleaver	I
Mapper	J
IFFT	K
Cyclic Extend	L
Synchronizer	M
Serial to Parallel	N
FFT	O
Detector	P
Viterbi	Q
RX Controller	R
Descrambler	S

To grep for the output of a module or set of modules, follow your **`make`** or **`./test`** command with a grep statement that will match lines with a first letter corresponding to the debug code of the modules you are interested in. For example:

```
./test | grep "^[GH]"
```

or

```
make | grep "^[GH]"
```

will output the debugging info for the Scrambler and Convolutional Encoder modules, and:

```
./test | grep "^[^BCD]"
```

will output all debugging information except that of the Analog TX, Channel Model, and Analog RX modules.

Useful Tools

Two programs have been provided to aid the debugging you will have to do in Problem 1. The first is a compiled version of the simulator before the three errors were introduced into the transmitter. This program is called **golden** and is provided in the **homework2** directory. **golden** can be run just like the **test** executable you will be creating and can be grepped in like fashion.

The other program, called **bit_errors**, will compare two binary files and output the number of bits in which these files differ. During the course of a simulation, the TX MAC and RX MAC modules will generate output files, called **tx_mac_output** and **rx_mac_output**, that contain all of the data sent by the transmitter and received by the receiver, respectively. **bit_errors** was designed to compare these files and give you the number of bit errors incurred during transmission. You can use this program (along with the size of the files) to calculate bit error rate for Problems 2 and 4, and to verify that you have corrected the errors in the transmitter in Problem 1. Running **bit_errors** on the output files of **golden** should report that there are zero bit errors.

Remote Login Servers

Two remote login servers have been set up for our class to share with 6.375. The servers are Athena Linux machines so they will be able to run all of the tools we will be using in 6.973 this semester, however, if you want to use graphical tools such as icms and are running Windows, you will need to have an X server running on your local machine (such as Exceed or XWin32). The two servers are called:

```
vlsifarm-01.mit.edu
```

and

```
vlsifarm-02.mit.edu
```

and you should all have remote login access to them. These servers have dual XEON 3.0 GHz processors and 1 GB of RAM each, and should be faster than the machines in 38-301, but they may become heavily loaded close to the due dates of 6.973 and 6.375 assignments.