CS 422 (Park) **Assignment II** **Due: Feb. 13 (Mon.), 2006**

Submission instructions: Please type your answers and submit electronic copies using `turnin` *by 5pm on the due date. You may use any number of word processing software (e.g., Framemaker, Word, LaTeX), but the final output must be in pdf or ps format that uses standard fonts (a practical test is to check if the pdf/ps file prints on a CS Department printer without problem). For experiments and programming assignments that involve output to terminal, please use* `script` *to record the output and submit the output file. Use* `gnuplot` *to plot graphs. Use* `ps2gif` *to convert a eps/ps plot to gif format (e.g., for inclusion in Word) if there is a need.*

PROBLEM 1 (20 pts)

Read Chapters 5, 6, and 7 from Comer. Solve Problems 5.2, 5.5, 6.5, and 6.6.

PROBLEM 2 (20 pts)

Assume you are the new kid in the digital radio market. Your company's name is *Tedious*. Being averse to multiple sine waves (you can tolerate at most one), you have decided to use baseband, not broadband, to broadcast 100 channels of digital radio. Half of the channels carry super-quality audio in the 0–20000 Hz range for your music junkies. The other half carry voice-quality audio in the 0–4000 Hz range for talk shows and such. Each digitized super-quality audio sample (its magnitude) is encoded using 32 bits; 16 bits per sample are used in the case of voice-quality audio.

• Using Nyquist's sampling criterion, determine how many digitized samples you must pack in a 1-second time window per super-quality audio channel. Do the same for each voice-quality audio channel. In total, how many 32- and 16-bit audio samples must you squeeze in a 1-second time interval for the 100 digital radio channels? What is the data rate (in bps) of each super-quality audio channel following the above quality specs? What is the data rate of each voice-quality channel? What is the data rate of the total 100-channel system?

• Since you're stuck with using a single sine curve to represent bits given your aversion to broadband—say, one (sine) period per bit where high amplitude means 1 and low amplitude means 0—what is the frequency f that you must request from the FCC so that you can broadcast your 100 digital radio channels at the bit rate (bps) computed above? Draw a nice-looking frame (i.e., packet) layout where the 100 radio channels are orderly organized (you may take inspiration from the T1 frame standard or do something completely different) so that your EE VP can go to work producing the transmission hardware.

• Suppose the FCC has already allocated the frequency that you requested to another company but has an unallocated frequency available that is 8 times slower than the original requested frequency (i.e., $f/8$). Supposing your hardware VP is pretty good at amplitude modulation but lousy at phase modulation, how would you use the 8-fold slower frequency to accommodate the data rate required by your 100-channel digital radio system?

PROBLEM 3 (25 pts)

After operating the 100-channel digital radio network for a few years in the West Lafayette area, business is good and you aim to expand the number of channels provided to 1000. You also contemplate on going national, shooting a couple of satellites into geostationary orbit around 22200 miles up in the sky to cover the midwest, your home base. Sirius and XM are shaking in their boots. Upon discussing the technical issues associated with the expansion with your Senior VP in charge of hardware systems, you find that requesting a 10-fold faster frequency from the FCC is not an option for packing 10 times more bits (you will now support 500 super-quality and 500 voice-quality audio channels), as your hardware VP is not only lousy with phase modulation but is weak when it comes to fast clocks. Playing around with amplitude modulation (beyond two levels for representing 0 and 1) is not an option either as you are planning on broadcasting from way up in the sky where sine amplitudes can get easily distorted before they hit the ground. So the only choice left is the dreaded FDM, i.e., broadband.

• Supposing the FCC happens to have 9 unallocated frequencies at $f + \Delta$, $f + 2\Delta$, ..., $f + 9\Delta$ where $\Delta = 10$ KHz is a sufficient guard band for keeping interference between the adjacent carrier frequencies in check, design a FDM

based system for transmitting 1000-channel digital radio following the technical specs of Problem 2 for audio quality. In so doing, you should specify what channels are carried on each of the 10 carrier frequencies, their individual bit rates, the new frame formats, and their total data rate.

• The newly designed system may be viewed as a hybrid system that mixes FDM with TDM. (Note, TDM, by default, uses square waves for transmitting bits, but it may also use sine waves with amplitude modulation as in our baseband example.) How would a "pure" FDM version of the 1000-channel digital radio system look like? Is your system "more" FDM or TDM? Explain your reasoning. What might be some reasons for designing hybrid FDM/TDM systems in place pure FDM or pure TDM systems?

• A key concern when deploying new networking technology is backward compatibility, also called legacy compatibility, with existing systems (hardware and software). Will your old digital radio customers in the West Lafayette region continue to be able to use their legacy receivers (for the baseband system transmitting digital radio operating at f carrier frequency) when listening to the new satellite based system? You may assume that broadcasting from satellites has no bearing on the problem. Discuss why or why not (it will depend on your new design). If the answer is no, can you redesign your 1000-channel system so that some measure of backward compatibility is achieved so that your legacy customers need not throw away their $200 receivers?

PROBLEM 4 (25 pts)

Give a 3-page summary overview of the following aspects of UNIX/Linux system programming: signals (`sigaction`, `kill`, `alarm`, `ualarm`, `sleep`, `waitpid`, `pause`), interprocess communication (`pipe`, `mkfifo`, `semget`, `shmctl`), process creation (`fork`, `vfork`, `execv`, `execve`, `exit`, `_exit`), and file I/O (`read`, `write`, `dup`, `fcntl`). Explain the technical meaning of *atomicity*, *blocking/non-blocking*, *synchronous/asynchronous I/O*, *mutual exclusion*, *semaphores*, and *interrupts* in the context of the above system calls. First state what the terms generally mean, then indicate in which system calls listed above they play a role. The `man` pages should be the primary source of reference. What is meant when one says that "everything in UNIX is a file"? How does `fork` work? What does a child inherit from its parent upon a `fork` system call? What are some features of the parent that a child does not inherent? In some operating systems, `fork` and `exec` are combined into a single system call, generically referred to as "spawning." What benefit, if any, is there in splitting process creation (`fork`) from binary execution (`exec`)? How is a *stdio* library call such as `printf` different from a `read` system call? What are the benefits introduced by *stdio*? All else being equal, is it better to invoke more system calls or less system calls? Explain your reasoning.

PROBLEM 5 (20 pts)

You will find a client/server application under ~park/pub/cs422 (or /.xinuserver/u9/park/pub/cs422); the client program is `client.c` and the server program is `server.c`. Reverse-engineer the code and explain what the server and client are doing. Compile, run, and check its behavior. Why is it important to always inspect the return value of a system call (despite the initial hassle and overhead)? In the server code, what is the role of the `dup2` system call? Can the service provided by this particular server be coded without the help of the `dup2` system call? Explain your reasoning. What is the role of `waitpid`? Is it an essential system call? What would happen if `waitpid` were omitted in the server code?