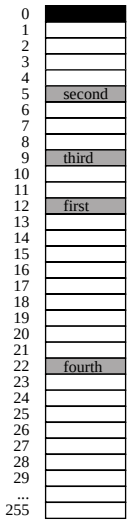
Jonathan W. Valvano First Name: _____ Last Name:_____
February 15, 2008, 10:00 to 10:50am

Open book, open notes, calculator (no laptops, phones, devices with screens larger than a TI-89 calculator, devices with wireless communication). You may put answers on the backs of the pages, but please don't turn in any extra sheets.

(30) Question 1. Consider a file system that uses a **file translation table** (**FTT**) to define the set of blocks allocated to each file. There are 65536 bytes on this disk, made up of 256 blocks, where each block is 256 bytes. Block 0 contains the directory and not available for data. Each file has its own **FTT**, which is a null-terminated list of block numbers assigned to the file. The example in the figure shows a file with 4 allocated blocks, with the first block at 12, and the last block at 22. The directory entry includes the file name, the total number of bytes and the block number of its **FTT**. All 256 bytes of each data block can contain data for the file. For example, the figure shows a file with 1024 bytes of data, stored in 5 blocks (**FTT** and 4 data blocks).

File Translation Table

0	12
1	5
2	9
3	22
4	0
5	0
...	
255	0



Disk
256 blocks
256 bytes/block

(10) Part a) Does this file system have any external fragmentation? Justify your answer.

(5) Part b) Assume a file has **n** data blocks. It takes one *block read* to fetch the **FTT**. On **average** how many more *block reads* does it take to read a single byte at a random position in the file? What is the **maximum** number of additional *block reads* that it takes to read a single byte in the file (worst case)?

(5) Part c) Consider the linked allocation scheme described in Lab 25. Assume the directory is in memory and the file has **n** data blocks. On **average** how many *block reads* does it take to read a single byte at a random position in the file? What is the **maximum** number of *block reads* that it takes to read a single byte in the file (worst case)?

(10) Part d) Assume you are given the following function that reads a 256-byte block from disk

```
int eDisk_ReadBlock(unsigned char *pt,    // result returned by reference
  unsigned char blockNum);                // which block to read
```

Write a C function that returns a byte from a file at a random location. Do not worry about error handling (e.g., **`eDisk_ReadBlock`** error or address too big). The inputs to the function are **`numFTT`** (the block number of the file's **FTT**) and **`address`** (the byte address, where 0 is the first byte, 1 means second byte etc.). You can use two buffers.

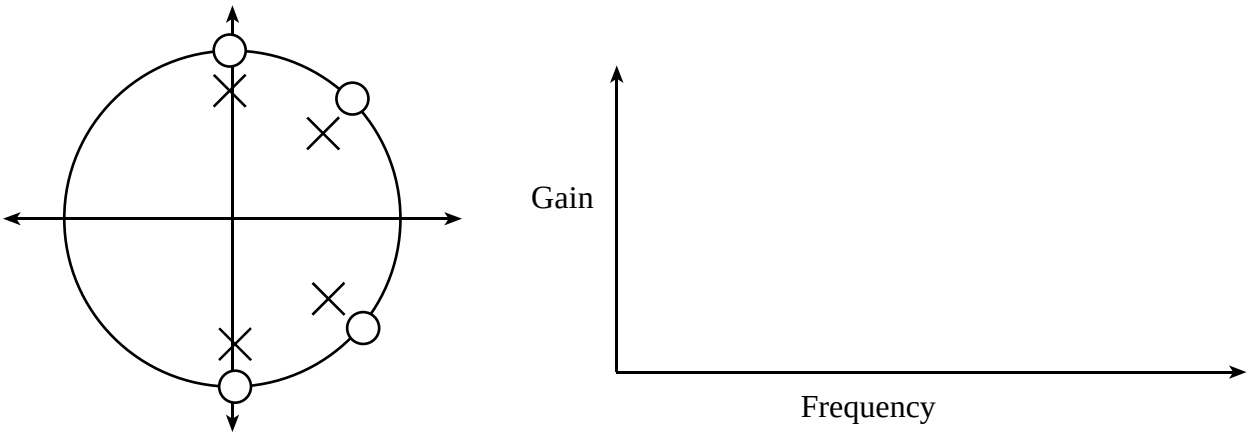
```
unsigned char FTTbuf[256];  // place to store FTT
unsigned char Databuf[256]; // place to store data
```

The prototype of the C function you have to write is

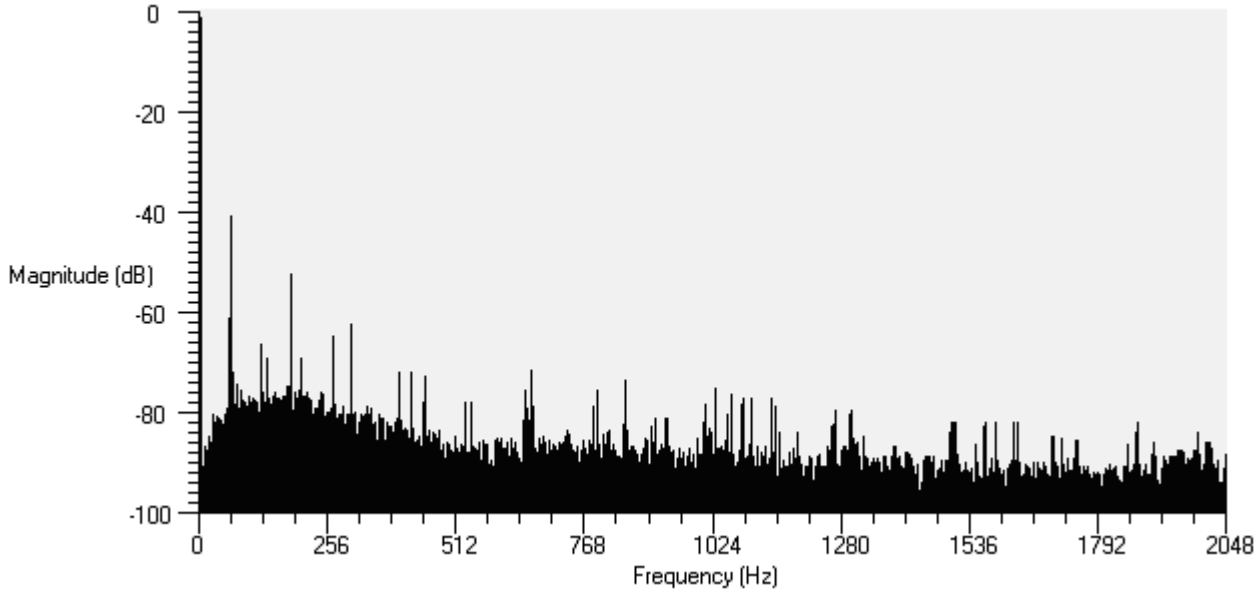
```
unsigned char eFile_Read(unsigned char numFTT, unsigned short address);
```

(20) Question 2. Consider a file system that manages a 16 Megabyte (2^{24} bytes) EEPROM storage for a battery-powered embedded system. You are free to select from a range of EEPROM chips with different block sizes. The block size can be any power of 2 from 1 to 2^{24} bytes. **Chip$_n$** has a total of 16 Megabytes with block size 2^n bytes. **Chip$_n$** can perform a 2^n byte block-write operation in 1 ms regardless of block size. For bandwidth reasons, therefore, you wish to choose a large block size. A block will be completed allocated to a file (you are not allowed to split one block between two files.) 16 bytes of each block are used by the file system to manage pointers, type, size, and free space. However, if the file were to contain 1 byte of data, an entire block would be allocated, and the remaining 2^n-17 bytes would be wasted. File sizes in this system are uniformly distributed from 50,000 to 150,000 bytes (this means any file size from 50,000 to 150,000 bytes is equally likely with an average size of 100,000 bytes). **You are asked to choose the largest block size with the constraint that the average internal fragmentation be less 5% of the total number of bytes stored.** Show your work.

(10) Question 3. The sampling rate of a real-time data acquisition system is 4000 Hz. Make a rough sketch of the gain versus frequency response of a digital filter with this pole-zero plot. You should label the **Frequency** axis with specific values like 1000, 2000, but you need not label the **Gain** axis.



(15) Question 4. In order to measure noise, the sensor on a data acquisition system is removed, and the inputs are grounded. The 10-bit 9S12 ADC is sampled at 4096 Hz, and the collected data are converted to the frequency domain by calculating the FFT. On this scale, -54 db is the same as about 5mV when converted to equivalent voltage. Similarly, -80 db is the equivalent of 250 μV.



Part a) What is the most likely cause of these errors.

Part b) How would to suggest we redesign the system to improve signal to noise ratio?

(25) Question 5. The objective of this question is to design the analog electronics to interface a vibration sensor to the 0 to +5V built-in ADC of the 9S12. The transducer output, $\mathbf{V_t}$, is a single voltage (relative to analog ground, not differential), with a range of -0.5 to +0.5 volts. The vibration signal has frequencies of interest from 1 to 200 Hz. The sampling rate is 400 Hz. Use analog filters to remove noise outside this frequency range. The two op amps operate on a single +5V supply and have rail-to-rail inputs and outputs. Build this interface (passive HPF, gain using one op amp, Butterworth LPF using a second op amp). REF03 is a 2.50V analog reference. You do not need to show the power connections.