EECS 452 Midterm — February 14, 2008

The written part of your midterm can either be submitted in hard copy form or via CTools in the form of a PDF file.

Hard copy should be single sided. I plan to be in the office from 10PM until midnight on Friday, February 22 to accept submissions. If I am not around prior to that, and you are, put your submission under my office door.

If you wish to submit a PDF version, and you do not know how to generate a PDF file, you are encouraged to contact a CAEN counselor to learn how to do this. Hard copy is the preferred form and I will print hard copies of the PDF submissions for use in grading.

C and VHDL program files should be collected together into a single zip file and submitted via CTools . Include your name in the comments embedded in the files. No name, no way to assign credit. Also, it is very difficult to *give* partial credit if one's work is not included.

This exam is given under the College of Engineering's Honor Code. **It is to be individual effort.** You are allowed to use any printed or web based references that you wish (including class handouts, etc). You may consult with either of the course GSIs or the instructor. Email questions will not be responded to. However, emails requesting appointments will be answered. Useful help will cost in terms of partial credit. For example, if you want to know if your answer to a particular question is correct or not and one of us tells you there will be a charge. There won't be any charge for answers to minor matters. Which are minor and which are not are part of your gamble. If you are the first to find a glaring error on the instructor's part you probably will get extra credit.

Include the honor pledge at the beginning of your PDF file and type your name as affirmation.

The honor pledge is " I have neither given nor received unauthorized aid on this examination, nor have I concealed any violations of the Honor Code."

"Instructors are not required to grade tests in which the signed Honor Pledge does not appear. The Honor Code remains enforced whether or not the student signs the Pledge."

www.engin.umich.edu/students/honorcode/code/interp.html

There is a significant amount of white space in this document. This is because it is both the test and the answer key with the answers turned off.

Your submission is due no later than midnight, Friday, February 22, 2008. Submissions can always turned in earlier.

1 Basic background

1. (5 pts) Write down the equations for the forward and inverse Discrete Fourier Transform. Indicate which is which. Use the "standard" definitions.

2. (5 pts)

   Relate the frequency spacing Δf between FFT values to: the sample rate f_s, the number of samples N in a data set, and the sample set duration T?

   ____

3. (5 pts) In a *few* sentences discuss the benefits that linearity (hence linear systems) provide. If you use an equation explain it's significance.

4. (10 pts) In lecture, three methods of using the FFT to take an inverse FFT (IFFT) were discussed. The first two methods were very practical while the third was more of theoretical interest. Briefly describe (using equations and words) the first two methods:

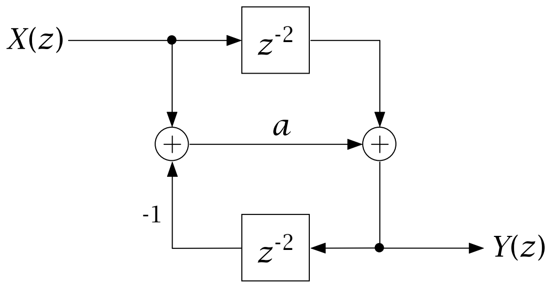

Figure 1: Interesting filter stage block diagram. The -1 indicates that the associated signal is multiplied by -1.

5. (10 pts) For the filter section shown in Figure 1 derive the transfer function

$$H(z) = \frac{Y(z)}{X(z)}.$$

6. (10 pts) We will be making use of the Figure 1 filter section later in the midterm. In order to able to scale out values properly to avoid overflow we need to know the two following input, X, to delay stage output transfer functions:

   (a) From the input X to the output, W_1, of the top z^{-2} box (actually a cascade of two single sample delays).

   $$\frac{W_1(z)}{X(z)} =$$

   (b) From the input X to the output, W_2 of the bottom z^{-2} box. As with the TDF2 it would be adequate to relate W_2/X to Y/X.

   $$\frac{W_2(z)}{X(z)} = \qquad\qquad \frac{Y(z)}{X(z)}$$

7. (10 pts) For the filter section shown in Figure 1 draw the transposed form.

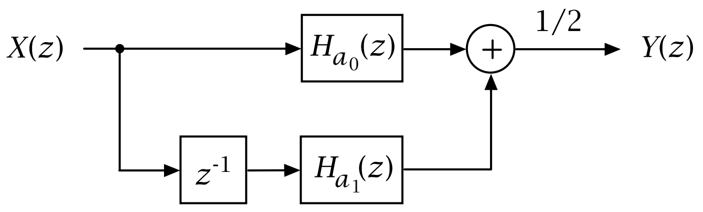

Figure 2: Low pass filter using two interesting filter sections (Figure 1). The boxes labeled $H_{a_0}(z)$ and $H_{a_1}(z)$ correspond to individual interesting filter sections using the indicated a values.

8. (25 pts) Figure 2 uses two of sections shown in Figure 1 to implement a low pass filter. One section uses a value of $a = a_0 = 1/8$ and the other uses a value of $a = a_1 = 9/16$. Use MATLAB and the subplot function to generate a sheet containing four plots (going top down):

   (a) Magnitude of the transfer function of a single section as shown in Figure 1 with $a = 1/8$. `subplot(4,1,1)`.

   (b) Magnitude of the transfer function of a single section as shown in Figure 1 with $a = 9/16$. `subplot(4,1,2)`.

   (c) The magnitude of the transfer function of a low pass filter as shown in Figure 2 with the top section using $a = a_0 = 1/8$ and the lower section $a = a_1 = 9/16$. `subplot(4,1,3)`.

(d) The above but plotted using dB over the range -80 dB to 2 dB. `subplot(4,1,4)`.

(e) A two stage cascade of the Figure 2 filter finds use as a half-band filter in high decimation sample rate conversion systems. These filters are used in the early stages when dropping sample rates in steps of two. The response of a two stage cascade is easily modeled by simply multiplying the dB values used generate the above `subplot(4,1,4)` plot. Generate this plot with the y-axis ranging from -160 dB to 2 dB.

Use a value of $f_s = 48000$ Hz and 1000 sample points going between 0 Hz and $f_s/2$. Hint: `freqz` should be very useful. Label your plots.

Include your sheet of plots and a copy of your MATLAB script.

Note: in testing I had some problems getting MATLAB to print nicely proportioned plots. If you run into this, consider putting two plots per page instead of four. I also had success by elongating the Figure prior to printing it. I did not have any problems generating eps plots using `print -deps xxx.eps`. See what works for you.

9. (10 pts) Assume a $f_S = 48000$ Hz sampled data set containing four unit amplitude sinewaves. The frequencies are 10 kHz, 15 kHz 20 kHz 24 kHz. If we drop the sample rate by a factor of two by simply discarding every second sample where in the spectrum does the energy associated with these

lines now appear?

Originally at	After resamling
10 kHz	
15 kHz	
20 kHz	
24 kHz	

10. (5 pts) If we were to filter the data used in the previous question using the two stage (two section) filter which of the lines would have negligible amplitude (< -100 dB) prior to re-sampling?

2 The TI C5510

11. (30 pts) We want to simulate a semi-secure communications system using the TI `cfft32_noscale` FFT. The system encodes information into a series of 12-bit complex numbers, takes the FFT and then sends the FFT values over an unsecure channel to a receiver. At the receiver an inverse FFT is used to recover the original data.

    Our present goal is to test the IFFT back to data processing. If we can't do this correctly it makes no sense to go on. Two options being considered here are the use of either the scaling or the non-scaling version of the cfft32 FFT to implement the inverse FFT. Probably one of the two versions will be easier to work with. We want you to investigate this and report on what you decided and why.

    You are to write a C program for the C5510 that reads two files, one containing the original data values and one containing the corresponding FFT values. These files should be generated using MATLAB. The values are to be integer. You are to use your choice of the scaling or non-scaling cfft32 function to inverse transform the FFT values. The program is to also compare the result with the original data. Print out the maximum error magnitude and the RMS error (you can use floating point for this calculation). Don't forget about the possible need to bit-reverse the data order.

    Your implementation will be tested using an instructor generated data set. If you feel information is lacking in the problem description, make reasonable assumptions and note them in you submission along with a short discussion about why you made the choices that you made.

- Include in your written submission a short discussion on the issues associated with the choice of the scaling vs the non-scaling and why you felt one version might be an easier choice to work with (or not).
- Include a listing of your test set MATLABscript in your written submission.
- Also include a brief discussion of how your data set was generated, how you approached the taking of the inverse FFT and how well the processing worked.
- Include the C5510 C source file in the collection of files to be used for testing.

12. (60 pts) Program the two section filter shown in Figure 2. Use $a_0 = 1/8$ and $a_1 = 9/16$. It should be possible to use only shift and add instructions in place of multiplications. Indeed these values were chosen with the intention of making this so. Include the C code for your filter function. A machine readable form is desired but we have not yet determined how to accomplish uploads using CTools. The name of the file should be `TwoSection.c`. Include your name in the comments.

    Use the calling sequence:

    ```
    void TwoSection(int *x, int *y, int nx);
    ```

    The first argument points to an array of `nx` input samples, the second argument points to an array of `nx` output samples, and `nx` is the number of samples to be processed per call.

    Any needed data buffers are to be included in your file. They are yours to define, organize and use. Points are given for using two's complement rounding when appropriate. You might consider using Q18 values when working in 1/8ths and Q19 values when working in 1/16ths or you might simply use Q19 values all around. The delay stages are to be 16-bit integers.

    (a) Modify your Exercise 5 Filter Timer code and determine the execution time. This would be a good time at which to use an oscillator to check out the transfer function and verify whether or not your code is working as expected. Include your measured time and any comments about any problems that you might have encountered.

    (b) The transfer function measurement program used in lab Exercise 5 has been modified for use in testing the `TwoSection.c` codes. Use it to measure the magnitude and phase response of your filter code.

If you will be submitting a PDF file for the exam include the post.ps file for your filter. If you are turning in hard copy, print the file and include the print version in your submission. This plot serves as the answer to this part of the question.

Ok, now for the gotchas. The following files (and I think only these) have been modified and replace like named versions:

- `TFMark2b.c`
- `DisplayTest01.c`
- `fir4testing.c` ... replaces fir function.

These will be available in a zipped collection available on the course handouts web page.

The `fir4testing.c` file replaces `fir.asm/c`. Specifying the FIR filter to TFMark2b will cause a call to the `TwoSection` function instead. This keeps the number of changes that needed to be made minimal. You will have to add your `TwoSection.c` file to the project. Everything should work. Specify the FIR filter and use a one bit shift, direct. The intent of this run is to verify that your filter's transfer function reasonably closely matches measurement. If not, either one or the other or both are likely wrong.

About the measurement. Slightly noisy plots maybe with a small spike or two have been seen. For a number of runs I got a moderate spike in the response around 19 kHz. From about 19 kHz to 24 kHz things get noisy. What bothers me is my first measurements had a moderate noise spike around 19 kHz and after trying to understand why (the filter code or the measurement code) the spike when away. If your plot looks reasonably like prediction your filter is probably good. Ask if you are uncertain, there is one free judgment call available.

The submitted files will be tested and evaluated on the basis of execution time, design, and efficient use of resources. The latter will be judgement calls by the instructor. In general, if the submitted function works properly and is reasonably well written full credit will be given. Overflow protection is not being asked for. It is expected that for very large input levels (almost "rail-to-rail") that the filter output will "break up".

You might find it helpful to sketch out the full filter showing all of the individual delay stages prior to starting to program. A picture is worth ....

Summarizing:

(a) Include a listing of your function in the PDF file along with your `FilterTimer` timing (as hex) value. You can also include any comments that will the draw the grader's attention to special aspects of your code.

(b) Include the filter transfer function plot. It might necessary to use a one bit shift in order to avoid overflow.

(c) upload your source file to CTools. This will be used in grading to check proper filter operation and to verify the timing.

3 The Spartan-3 Starter Board

13. (60 pts) As the final part of the MidTerm you are to use VHDL to implement the filter that you just programmed in C and test in order to verify operation.

    You are provided the C test code, the basic top level VHDL support and a skeleton filter VHDL entity where you will add your VHDL code. The only portion of the task that has significant difficulty is the multiplication by 1/8 and 9/16. Implementation of rounding is not required (and I didn't in my test code).

    I divided my task into register updating in a state machine (provided) and doing the required arithmetic outside of the main process using the VHDL + and - operators (no bit serial arithmetic required here).

    I decided to do the arithmetic using 1/16ths both for the a_0 section (1/8 = 2/16) and for the a_1 section (9/16). I used 16 bit registers but extended the signals to 21 bits. This provides one extra bit to catch the carry out from summing the outputs from the two sections prior to division by 2. The division by 2 is easily accomplished by loading only the sum bits (20 downto 5) into the register used to hold the y output.

    To convert a 16 bit Q15 signal, `w1`, into a 21 bit Q19 signal one can use

    ```
    (w1(15) & w1 & "0000")
    ```

    The `w1(15)` bit sign extends and the `"0000"` is what gets appended to account for what amounts to a 4 bit left shift.

    To create a 21 bit 3/16 Q19 version of a 16-bit Q15 signal one might do the following:

    ```
    w3_16 <= (w1(15) & w1(15) & w1(15) & w1(15) & w1 & '0')
           + (w1(15) & w1(15) & w1(15) & w1(15) & w1(15) & w1);
    ```

    The extension to other multiples of 16 should be apparent.

    Other than doing the arithmetic the implementation should be easy. Again, implementing rounding is not required for this effort.

A detailed block diagram with each delay stage and some of the internal signals named proved extremely useful when I did this part of the exercise. When adding three signals together I did this using a temporary signal for one sum and then adding the third signal to obtain the sum of the three signals.

The primary C and VHDL files will be packaged into a zip file and be placed onto the EECS 452 Course Handouts Page. Any remaining files can be found in the lab experiment support directories. The files used in this part of the MidTerm were based on those used in the metastability demonstration. If you have a choice as whether to use an earlier file or the version supplied, you should choose the version supplied.

The following TI files will be used:

- `AIC23Support01.c`
- `MemIncDec.asm`
- `MidTermMain.c`
- `setup_codec_meta.c`
- `EECS452.cmd`
- `rts55.lib`
- `C5510.h`
- `McBSP_452.h`

The C5510 left channel displays the left channel input. Maybe the oscillator? The C5510 right channel displays filter output of the right channel input signal. Maybe the oscillator?

The following VHDL files will be used:

- `TwoSectionTop.vhd`
- `MidTermFilter.vhd`...the file you add to.
- `McBSPS3slave.vhd`
- `SSD02.vhd`
- `S3_metastable.ucf`

You can add your code to `MidTermFilter.vhd` or start over creating your equivalent.

As with the C code filter version over driving the input will cause the output to break up. This is only near the maximum input level swing.

Submit:

- as part of the hard copy or PDF file the VHDL listing.
- any comments about your code and testing that you want the grader to be aware of.
- the Filter VHDL file. This will be used to test and evaluate your solution.

Answers to minor VHDL syntax questions will be charged for lightly if at all.