

ELEC3730 - Assignment # 1

- Your answers to this assignment should consist of source code printouts accompanied by any *brief* written explanation of the solutions strategies you have employed that you feel would be useful. It is recommended that you include sufficient commenting in your code to make the need for this explanation as non-existent as possible.
- Your assessment for this assignment will be based on this written report. You should be aware that you may be required to demonstrate your solutions (compile and run them) should your tutor require it.
- You are expected to attempt all the following questions.

1. Programming in C

- (a) **(15 marks)** Consider a signal $u(t)$ that is sampled once every Δ seconds by an analog to digital (A/D) converter to yield a stream of samples $\{u_k\}$ given by

$$u_k = u(t)|_{t=k\Delta} = u(k\Delta). \quad (1)$$

Your task is to filter these samples to provide a new set of samples $\{y_k\}$ according to the following formula

$$y_k = \sum_{n=0}^{M-1} h_n u_{k-n} \quad (2)$$

where $\{h_n\}$ are a set of fixed numbers. For example, if $h_0 = 1, h_1 = 0.5, h_2 = -0.2$ and $M = 3$ then equation (2) says that a filtered output sample y_k is generated from three input samples according to

$$y_k = u_k + 0.5u_{k-1} - 0.2u_{k-2}. \quad (3)$$

This is called Finite Impulse Response (FIR) filtering, and your mobile phone uses it extensively.

Write a function in C that will implement an arbitrary FIR filter computation. Its prototype should be

`void FIR(float *y, float *u, float *h, int M, int N)`

where `y` and `u` are (respectively) pointers arrays of output and input samples of length `N`, and `h` is a pointer to an array of FIR filter co-efficients of length `M`. In implementing (2), you may assume that $u_k = 0$ for $k < 0$.

- (b) **(15 marks)** Write a program that generates $N = 400$ samples of a signal which is the square wave

$$u_k = \begin{cases} 1 & ; (k/100 \bmod 2) < 1 \\ -1 & ; (k/100 \bmod 2) \geq 1 \end{cases} \quad (4)$$

saves it to a file (called `data.txt`) as ascii text, one line per data point, calls the `FIR` function from the preceding question with

$$h_k = \frac{1}{\alpha} \times 0.95^k, \quad M = 100; \quad \alpha = \sum_{k=0}^{M-1} 0.95^k$$

then prints the result to a file `output.txt`.

Compare to the result computed by MATLAB. This can be found by loading the `data.txt`, `output.txt` files using the MATLAB `load` command, followed by the `filter` command. That is, compare the results from your C program to MATLAB's results which can be computed by

```
>> k=0:2*pi/399:2*pi;
>> u=sign(sin(2*k));
>> h=0.95.^(0:1:99); h = h/sum(h);
>> y=filter(h,1,u);
```

- (c) **(15 marks)** Another method of digital filtering is the Infinite Impulse Response (IIR) technique, which implements the following relationship between input samples $\{u_k\}$ and output samples $\{y_k\}$

$$y_k = \sum_{n=0}^{M_b-1} b_n u_{k-n} - \sum_{n=1}^{M_a-1} a_n y_{k-n} \quad (5)$$

where now there are two sets $\{b_n\}$ and $\{a_n\}$ of fixed numbers which define the filter.

For example, if $b_0 = 1, b_1 = 0.5, b_2 = -0.2, a_1 = 0.5, a_2 = 0.25, M_b = M_a = 3$ then equation (5) says that a filtered output sample y_k is generated from three input samples *and two past output* samples according to

$$y_k = u_k + 0.5u_{k-1} - 0.2u_{k-2} - 0.5y_{k-1} - 0.25y_{k-2}. \quad (6)$$

Write a function in C that will implement an arbitrary IIR filter computation. Its prototype should be
void IIR(float *y, float *u, float *a, float *b, int Ma, int Mb, int N)

In implementing (5), you may assume that $u_k = 0, y_k = 0$ for $k < 0$.

- (d) **(10 marks)** Repeat question 1b but now using the above IIR filter with $M_a = M_b = 3$ and

$$a_1 = -1.908636, \quad a_2 = 0.914361, \quad b_0 = 0.00135109, \quad b_1 = 0.00270219, \quad b_2 = 0.00135109.$$

The MATLAB component will become

```
>> k=0:2*pi/399:2*pi;
>> u=sign(sin(2*k));
>> a=[1,-1.908636,0.914361]; b=[0.00135109,0.00270219,0.00135109];
>> y=filter(b,a,u);
```

2. Bit-Wise programming in C

- (a) **(10 marks)** A project requires a time-of-day subroutine. The time of day is recorded in a structure called TIME that has three eight bit members called hrs, mins and secs. Write a subroutine that accepts a pointer to this structure and increments the time by one second. Write a program that calls this subroutine in a manner to illustrate its correct operation via printing results to the screen.
- (b) **(10 marks)** Write a code-snippet that will call a subroutine process if the bits in an 8-bit variable status are as follows (do not change the value of status or use structure bit fields)
- bit3 \oplus bit1=0
 - bit2 + bit0=0