THE CITY COLLEGE OF NEW YORK
Department of Electrical Engineering

EE425 Computer Engineering Laboratory (1XB) – Summer 2016

Experiment 4 – Digital Signal Processing: FIR Filter Implementation

Objective: This experiment is designed to show the digital signal processing capabilities of the PIC18F4520 microcontroller and implement a Finite Impulse Response (FIR) low-pass filter.

Specific Tasks: Program the microcontroller to implement the following transfer function:

$$H(z) = \frac{1 + z^{-1} + z^{-2} + z^{-3}}{4}$$

Part 1 (preparation):

1. Use the *freqz* function from MATLAB to plot the frequency response of the transfer function H(z). For this you may use the sampling frequency $fs = 2*f_{NYQUIST}$, where $f_{NYQUIST}$ is the Nyquist frequency you found in Exp. 2.
2. Use MATLAB to simulate the time-domain response of the filter to a sinusoidal input within a range of frequencies (0 Hz→ fs/2 Hz) and verify the frequency-domain response of the filter obtained in numeral 1.

Part 2:

1. Write the assembly language program to implement the FIR low-pass filter. All arithmetic operations must use the full 10-bit value from the A/D conversion to compute the filtered binary number (use 16-bit arithmetic). Then, only the 8 MSB of the 10-bit result must be sent to the D/A conversion process.
2. Input a sinusoidal signal from the Function Generator to one of the ports of the microcontroller and display the filtered signal in the oscilloscope. Also display the original input in the oscilloscope for comparison.
3. Sweep the frequency of the input signal from 0 Hz to fs/2 Hz and record observations from the oscilloscope. Save all important oscilloscope images and discuss them in the report.
4. Verify the frequency response of the filter obtained in numeral 1 of Part 1.
5. Compare the real-time time-domain response of the filter to a sinusoidal input shown in the oscilloscope to the response obtained in numeral 2 of Part 1 in the frequency range 0 Hz< f < fs/2 Hz.

The following are some steps that you need to consider:

- Use your A/D-D/A conversion program from Exp. 2 as a starting point and modify it to **sample as fast as possible**.
- Draw a flow diagram of the filtering process before coding.
- Write your code in a modular fashion and use the subroutine structure to facilitate coding and make your program transparent and easier to debug.
- Establish a constant loop-timing: the number of clock cycles it takes to return to the execution of any instruction inside your processing loop must be the same.
- Consider using the *nop* instruction.
- Avoid conditional branching as much as possible.
- Verify that each part of your code works before proceeding.
- Use the Sync output of the Function Generator to externally trigger the signals in the oscilloscope.