

## Lab 4i Digital Scope and Spectrum Analyzer using a USB link

This laboratory assignment accompanies the book, Embedded Microcomputer Systems: Real Time Interfacing, by Jonathan W. Valvano, published by Thomsen Publishing, copyright © 2006.

- Goals**
- Build a low frequency spectrum analyzer,
  - Create a high-speed data link between the 9S12 and the PC using USB,
  - Write a PC application that displays the data in both the time domain and the frequency domain.

- Review**
- Operation of the ADC system in the 9S12 data sheet,
  - Data sheets on the Texas Instruments TLC2272 dual op amp IC,
  - Valvano Chapters 11,12,14 on analog amplifiers, analog low pass filters, ADC's, noise, data acquisition systems, and USB.
  - Product information on the DLP design UM245R module

<http://www.ftdichip.com/Documents/ProgramGuides/D2XXPG34.pdf>

- Starter files**
- Lab 2, software in View09 and View10 lecture notes

### Background

This lab builds upon the audio recording system developing in Lab 2. Figure 4.1 shows the data-flow graph of the digital scope and spectrum analyzer system. The electret microphone is a low-cost low-fidelity transducer appropriate for voice recording. An analog band-pass filter will be used to improve signal to noise ratio and to prevent aliasing. You will use an ADC converter on the 9S12 to convert the analog signal into digital form. You are free to choose any sampling rate ( $f_s$ ) you wish, as long as the sampling is performed in real time. The periodic ISR creates the real time sampling, and data is written into a *Fifo* queue located in the DLP Designs UM245R module. This module is based on the chip FT245R made by FTDI (Future Technology Devices International Ltd.). The data is then transmitted to the PC using a USB interface. Windows, a device driver (FTD2XX.sys) and an application programmer interface (FTD2XX.DLL) are provided. You will write an application that runs on the PC displaying the data. Start and stop commands can be sent from your application on the PC to the 9S12 allowing the PC to control activity on the microcontroller. The measurements are displayed on the PC in both time-domain and frequency-domain formats.

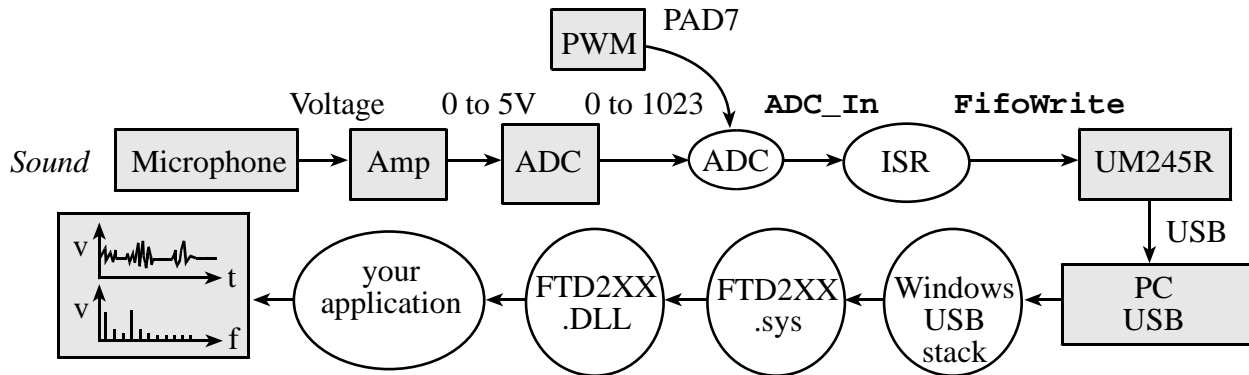


Figure 4.1. Data-flow graph of the digital scope and spectrum analyzer.

**Nyquist Theorem:** If  $f_{max}$  is the largest frequency component of the analog signal, then you must sample more than twice  $f_{max}$  in order to faithfully represent the signal in the digital samples. For example, if the analog signal is

$$A + B \sin(2\pi ft + \phi)$$

and the sampling rate is greater than  $2f$ , you will be able to determine  $A$ ,  $B$ ,  $f$ , and  $\phi$  from the digital samples.

**Valvano Postulate:** If  $f_{max}$  is the largest frequency component of the analog signal, then you must sample more than ten times  $f_{max}$  in order for the reconstructed digital samples to look like the original signal when plotted on a voltage versus time graph.

**Preparation 1 (do this before your Monday/Tuesday lab period)**

1. Design an interface between the 12 digital I/O signals on the UM245R. Design a device driver (a set of software functions in USB.h and USB.c) for the interface. You will need functions that

- 1) check read status (to see if data are available for reading),
- 2) check write status (to see if there is room in the *Fifo* for writing),
- 3) read a byte using busy-wait synchronization, and
- 4) write a byte using busy-wait synchronization.

Write a main program that transmits a simple data sequence (0,1,2,3,...,254,255,0,1,2,...). This main program should also read a byte and toggle the LEDs after each byte received. Place prototypes for public functions in the USB.h header file.

2. Write a main program that implements the data flow as described in Figure 4.1. Add debugging instruments that you can use to verify the sampling is performed in real time. In particular quantitatively measure the sampling jitter of your system.

3. Find a computer onto which you can install the D2XX drivers (FTD2XX.sys and FTD2XX.DLL). These drivers should also be installed on the lab computers. Do not use the virtual com port (VCP).

<http://www.ftdichip.com/Drivers/D2XX.htm>

[http://www.ftdichip.com/Documents/InstallGuides/Windows\\_XP\\_Installation\\_Guide.pdf](http://www.ftdichip.com/Documents/InstallGuides/Windows_XP_Installation_Guide.pdf)

Also install the test application dlptest10c.zip (freeware version 1.0)

<http://www.dlpdesign.com/test.shtml>

**Preparation 2 (do this before your Wednesday/Thursday lab period)**

4. Work through the example shown in lecture notes View09 and View10. Watch the 30 minute video of this process, which is posted on Blackboard. In particular, you will write an application that opens the USB device and accepts the real-time sampled data. You can write this application in whatever platform you wish, but the example programs show the syntax for C++ Builder, C#, Delphi, LabVIEW, Visual Basic and Visual C++

<http://www.ftdichip.com/Projects/CodeExamples.htm>

<http://www.dlpdesign.com/drivers/D2XXPG21.pdf>

For the preparation, your application should add the functionality described in lecture notes View09.

**Procedure (do this during your lab period)**

1. *USB interface test.* Use the simple program written in preparation 1 to test the USB link. On the PC you can run the DLPtest10c.zip program. Verify that data can be successfully exchanged between the 9S12 to the PC.

2. *System checkout:* Run your hardware/software system to verify operation. Connect the microphone to the input and verify the sound patterns can be displayed on the PC. Again, compare qualitatively the collected data to the original signal as observed using an oscilloscope on the ADC input. Print a typical waveform (time domain and frequency domain) for one spoken word. Your digital scope system has these requirements:

You may choose 8-bit or 10-bit sampling, any sampling rate at or above 1 kHz

You may choose any FFT buffer size greater than or equal to 1024 (make it a power of 2)

Continuously collect, transmit, and display data in both the time-domain and in the frequency domain.

Collect one buffer of data, perform a FFT calculation, convert output to dB of full scale, and then plot it.

Both time-domain and frequency domain plots should have axes with numbers, labels, and units.

You will need to add commands to start and stop the transmission (menu commands).

Ability to print the current data on a printer (see lecture notes View10)

Functionality that is nice, but not required

You do not need to calculate a new FFT after each data is received.

You do not need to save data as files on the PC.

You do not need to allow the user to adjust the sampling rate and FFT buffer size

You do not need to implement dynamic zooming, or scaling (i.e., the axes can be fixed)

You do not need to have the commands executable from the ToolBar.

You do not need to have the commands executable from function key (accelerator).

You do not need to modify icons for application or data files

You do not need to modify help files (but you should create the help system and observe how to works)

You do not need to implement FFT data windowing (Hamming window etc.)

3. *Determine the bandwidth.* Increase the sampling rate until data becomes lost. Use debugging instruments to detect lost data and to determine which module limits the bandwidth. Redesign the system so it operates from 50% to 75% of this experimentally determined maximum.

4. *Basic understanding:* For this section you will use the entire system except the microphone. The purpose of this section is to verify the **Nyquist Theorem** and the **Valvano Postulate**. Again, use a sinusoidal waveform (approximately  $2.5+2*\sin(2\pi ft)$  at the ADC) with an adjustable frequency,  $f$ . Collect data on the PC at about four different frequencies for the report. Describe the concepts of Nyquist Theorem, Valvano Postulate, and aliasing using this specific data. Be prepared to explain your results during checkout.

### **Deliverables (exact components of the lab report)**

A) Objectives (1/2 page maximum)

B) Hardware Design

Circuit diagram of the microphone and USB interfaces (preparation 1)

C) Software Design (printout of these software components)

1) Low level USB interface (**USB.c** and **USB.h** files) (preparation 1)

2) Main program used to test the USB link (preparation 1)

3) Main program used to test the system (preparation 2)

4) PC application used to display recorded data (preparation 4)

D) Measurement Data

1) Dynamic circuit performance (procedure 2)

2) Voice data (time domain and frequency domain) collected in the 9S12 (procedure 2)

3) Voice data (time domain and frequency domain) displayed on the PC (procedure 2)

4) Four sets of data (time domain and frequency domain) displayed on the PC (procedure 4)

E) Analysis and Discussion (1 page maximum)

### **Partial Checkout (Wednesday or Thursday during your lab period)(show this to the TA) (20/50)**

Demonstrate preparation 1 and procedure 1. Install drivers on PC (execute CDM 2.02.04.exe), run dlptest10c.exe, and demonstrate you can send and receive individual bytes between the PC and the 9S12.

### **Checkout (show this to the TA) (30/50)**

You should be able to demonstrate the proper operation similar to procedures 2 and 4. Be prepared to discuss what the FFT data means. The TA may ask you to discuss the various factors that limit bandwidth on this system.

### **Hints**

1) This is a long lab with many parts, so start early.

2) Don't try to complete the experiment in one fell swoop. Debug the system in an analytical, step-wise manner.

3) It is OK to download software from the web (not from other EE345M students of course), as long as you reference the source of the software and follow all legal procedures. If you find yourself erasing someone else's name off the source code, you are probably doing something wrong.