Lab 5 Requirements Document
Justin Nguyen and Trevor Murdock

1. Overview

1.1. Objectives:

The objectives of this project are to design, build and test a music player. Educationally, **we** are learning how to interface a DAC, how to design a speaker amplifier, how to store digital music in ROM, and how to perform DAC output in the background. **Our** goal is to play **Pirates of the Carribean.**

1.2. Process: How will the project be developed?

The project will be developed using the TM4C123 board. There will be **two** switches that the operator will use to control the music player. The system will be built on a solderless breadboard and run on the usual USB power. The system may use the on board switches or off-board switches. A hardware/software interface will be designed that allows software to control the player. There will be **three** hardware/software modules: switch input, DAC output, and the music player. The process will be to design and test each module independently from the other modules. After each module is tested, the system will be built and tested.

1.3. Roles and Responsibilities: Who will do what? Who are the clients?

Trevor and Justin are the engineers and **Dylan Zika** is the client. **We will both write the requirements document. Trevor will write the program skeleton. Justin will draw the PCB layout. We will both work on the modules.**

1.4. Interactions with Existing Systems: How will it fit in?

The system will use the TM4C123 board, a solderless breadboard, and the speaker as shown in Figure 5.1. It will be powered using the USB cable.

1.5. Terminology: Define terms used in the document.

SSI – (Synchronous Serial Interface) is different from UART because two devices use the same clock. SSI protocol consists of four I/O lines (SSI0Fss – a control signal), (SCK – a clock generated by the master), (SSI0Tx – driven by master), (SSI0Rx – driven by slave). Data is serially shifted, clocked out on one edge, and received on the other edge.

Linearity – a measure of the straightness of the static calibration curve. A DAC's resolution is linear if f(n + 1) – f(n) = f(m + 1) – f(m) where m & n are digital inputs.

Frequency Responses –the quantitative measure of the output spectrum of a system or device in response to a stimulus. It is a measure of magnitude and phase of the output as a function of frequency. X(t) = Asin(wt) and Y(t) = Bsin(wt + phi)

Loudness – dependent on the voltage given to the DAC. The DAC has a max output volume, but one can mimic loudness by setting the max output volume to the DAC's max output and scaling it down from there.

Pitch – the sine wave's frequency. Different pitches are equivalent to different notes/sounds.

Instrument – one can mimic an instrument through the DAC by using an envelope tailored to that instrument. Brass instruments have much louder and sharper notes as opposed to a wind instrument's softer and more fluid sound. Basically just multiply the correct envelope by your sine wave table and voila, you can mimic an instrument on the DAC.

Melody & Harmony – harmony complements the melody

1.6. Security: How will intellectual property be managed?

The system may include software from StellarisWare and from the book. No software written for this project may be transmitted, viewed, or communicated with any other EE445L student past, present, or future. It is the responsibility of the team to keep its EE445L lab solutions secure.

2. Function Description

2.1. Functionality: What will the system do precisely?

If the operator presses the play/pause button once the music should pause. Hitting the play/pause again causes music to continue. The play/pause button does not restart from the beginning, rather it continues from the position it was paused. If the rewind button is pressed, the music stops and the next play operation will start from the beginning. There is a mode switch that allows the operator to control some aspect of the player. Possibilities include instrument, envelope or tempo. There is a C data structure to hold the music. There must be a music driver that plays songs. The length of the song should be at least 30 seconds and comprise of at least 8 different frequencies. Although we are playing only one song, the song data itself will be stored in a separate module and be easy to change. The player runs in the background using interrupts. The foreground (main) initializes the player, then executes for(;;){} do nothing loop. If you wish to include LCD output, this output should occur in the foreground. The maximum time to execute one instance of the ISR is **.5 ms**. A background thread implemented with output compare will fetch data out of the music structure and send them to the DAC. There is a C data structure to store the sound waveform and instrument. The generated music must sound beautiful utilizing the SNR of the DAC. **It will** be easy to change instruments.

2.2. Scope: List the phases and what will be delivered in each phase.

Phase 1 is the preparation; phase 2 is the demonstration; and phase 3 is the lab report. Details can be found in the lab manual.

2.3. Prototypes: How will intermediate progress be demonstrated?

A prototype system running on the TM4C123 board and solderless breadboard will be demonstrated. Progress will be judged by the preparation, demonstration and lab report.

2.4. Performance: Define the measures and describe how they will be determined.

The system will be judged by three qualitative measures. First, the software modules must be easy to understand and well-organized. Second, the system must employ an abstract data structures to hold the sound and the music. There should be a clear and obvious translation from sheet music to the data structure. Backward jumps in the ISR are not allowed. Waiting for SSI output to complete is an acceptable backwards jump. Third, all software will be judged according to style guidelines. Software **will follow the style described in Section 3.3 of the book** . There are three quantitative measures. First, the SNR of the DAC output of a sine wave should be measured. Second, the maximum time to run one instance of the ISR will be recorded. Third, you will measure power supply current to run the system. There is no particular need to optimize any of these quantitative measures in this system.

2.5. Usability: Describe the interfaces. Be quantitative if possible.

There will be **two** switch inputs. The DAC will be interfaced to a 32-ohm speaker.

2.6. Safety: Explain any safety requirements and how they will be measured.

If you are using headphones, please verify the sound it not too loud before placing the phones next to your ears.

3. Deliverables

3.1. Reports: How will the system be described?

A lab report described below is due by the due date listed in the syllabus. This report includes the final requirements document.

3.2. Audits: How will the clients evaluate progress?

The preparation is due at the beginning of the lab period on the date listed in the syllabus.

3.3. Outcomes: What are the deliverables? How do we know when it is done?

There are three deliverables: preparation, demonstration, and report.