

Lab #9: Scrolling Images and Melodies

Introduction

In order to construct an interactive video game, you will need to provide both audio and visual feedback to the player. This week's lab provides an introductory methodology to simple animations and music playing. The main objectives are to successfully toggle between static images (the arrows you have been displaying) and animations, as well as between short beeps and a full melody.

Melody Overview

For this week's lab, your STM32L100RC microcontroller is required to play a song when the RESET button is pressed. The song is to be played through an 8-Ω speaker driven by the audio driver circuit constructed in last week's lab. The audio signal will be a variable frequency, 50% duty cycle PWM signal. In order to play a song, the frequency of this signal will have to be changed to the corresponding music note. The notes must be played in the correct order and changed at the correct time to play a song. The song you implement on your microcontroller this week is up to you; however, if you would like to perform a simple melody, you may play the Auburn University Fight song. Figure 1 and Table 1 show the musical notation and a simplified "Music for Non-Musicians" version of this melody respectively.



Figure 1: "War Eagle fly down the field!"

Lyric	Music Note	Duration (counts)
War	C	2
Ea-	A	1
Gle	G	1
Fly	F	0.5

Down	F	0.5
(rest)		0.5
The	D	0.5
Field!	C	2

Table 1: Simplified Representation of “War Eagle”

Animation Overview

For this week’s lab, your STM32L100RC microcontroller is required to display a simple animation on the LED matrix you have set up in previous labs. This can be achieved in several ways, but two simple ways will be described for you to implement in your lab. First, this can be achieved by cycling through several simple images in a specified pattern: Figure 2 shows the three images necessary to make Mario run in his NES debut. Additionally, you may implement a simple animation on your microcontroller by displaying a ‘scrolling text’ image. For this lab, you may implement any animation you would like using either methodology; however, if you would like to continue the Auburn University theme, Figure 3 shows one possibility for scrolling text that says ‘AUBURN!’. The image for a scrolling text image can be stored into a 32-bit or 64-bit, which can be scrolled through by masking 8 bits at a time.



Figure 2: Sprites of Mario Running

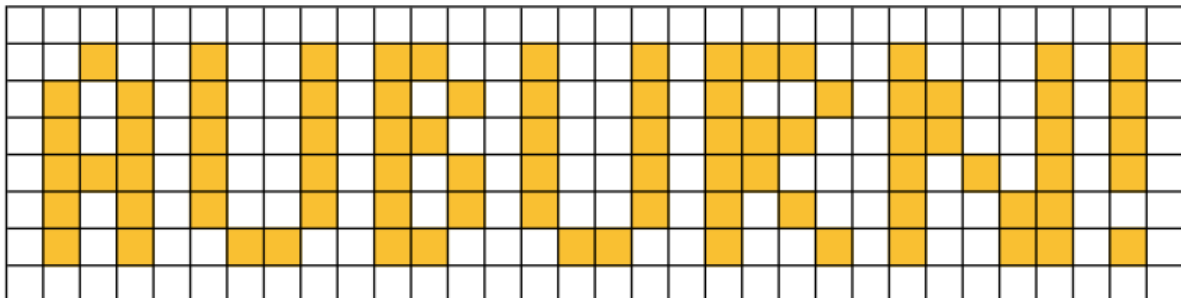


Figure 3: ‘AUBURN!’ Scrolling Text Image

Pre-Lab Assignment

Modify the code from last week's lab to play a song and display an animation when the 'RESET' button is pressed. When any other key is pressed, the static image corresponding to that button should be displayed and a 0.1 second duration beep should be heard. The song played and the animation displayed can be any of your choosing, so be creative!

Things to consider when planning and drafting your code:

1. Can the timer you have configured to display images on the LED matrix be used to change music notes?
2. Will the same strategy work for a multi-image animation (Mario running) as a scrolling text animation?

Laboratory Experiments

1. Record all observations and problems you have in your lab notebook.
2. Check all wiring from prior labs before powering up EEBoard and STM32L100RC
3. Observe the audio signal for your song on the Waveforms digital Oscilloscope; annotate a screenshot to show the different music notes played and their duration
4. Observe the values on the columns of the LED matrix using the Logic Analyzer; can you obtain a screenshot of the animation changing?

Laboratory Report

1. Include all screenshots with annotations
2. Discuss your chosen strategy of animation and why you could or couldn't obtain a screenshot of the animation changing
3. Discuss possible methods for animating the player-controlled motion of a character around the LED matrix.