***Rapid Embedded Systems***

***Design and Programming Course***

**LAB 7**

**Final Lab**

**Issue 1.0**

Contents

[1 Introduction 1](#_Toc15978193)

[1.1 Lab overview 1](#_Toc15978194)

[2 Requirements 1](#_Toc15978195)

[3 Hardware Setup 2](#_Toc15978196)

[4 Final Lab 3](#_Toc15978197)

[4.1 Song\_def.h 3](#_Toc15978198)

[4.2 Timeout Vs Ticker 3](#_Toc15978199)

[4.3 Your Application Code 4](#_Toc15978199)

# Introduction

## Lab overview

In this lab, we will remake the audio player with the following features:

-Users will be able play 8 different songs using 3 switches and 1 push button

-Users will be able to stop and replay the song from the audio player

-Users will be able to display the name of a song that is being played in the LCD

-Users will be able to display a confirmation message when choosing the song in the LCD

-Users will have 5 seconds to confirm the song that they want to play.

-Users will be able to see the state of the audio player using RGB LEDs (playing song, waiting for next song, no playing song)

-Users will display the instructions for using the audio player via UART

-Users will be able to adjust the volume of a song

-Users will be able to execute these features with the help of interrupts.

By the end of this lab you will have built your own “MP3 player” and you should feel confident about the techniques and concepts that you have learned during this course.

# Requirements

## Software and Hardware

In this lab, we will be using the following hardware and software:

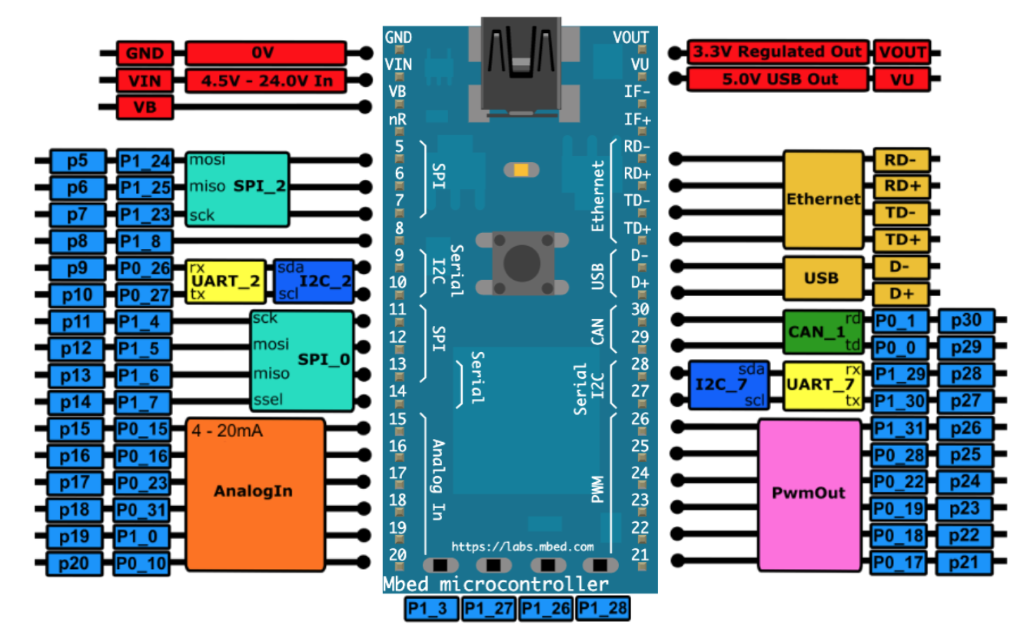
* **Visual Studio,**or any other text editor.
* **Mbed Simulator**, an offline simulator of an mbed microcontroller and hardware components.

The code skeleton, which includes some support for implementing the task in 4 should be found in the same folder as this manual.

# Hardware Setup

## Pin Layout

In this experiment, we are going to use the Mbed Simulator target, for which pin descriptions can be found below:



BUTTON1

*Figure 1: The Mbed Simulator board pin descriptions*

Connect the LEDs and the pins on the circuits to their relative pin which are defined in the table below.

|  |  |
| --- | --- |
| **Pin** | **Pin name in Mbed API** |
| THERMISTOR | p15 |
| PWM SPEAKER | p21 |
| PUSH BUTTON | p22 |
| SWITCH 1 | p23 |
| SWITCH 2 | P24 |
| SWITCH 3 | P25 |
| RED LED | p26 |
| YELLOW LED | p27 |
| BLUE LED | P28 |
| C12832 LCD display SPI MOSI | p5 |
| C12832 LCD display SPI MISO | p6 |
| C12832 LCD display SPI SCK | p7 |
| USB UART TX | p13 |
| USB UART RX | p14 |

# Final Lab

In this lab, we will build our own “MP3 player”, for the sake of this lab, students do not need to generate the songs like they did in lab 8. Students should download the following files: song.h ,song\_def.h and main.cpp. These files should be placed in the same directory.

## Song\_def.h

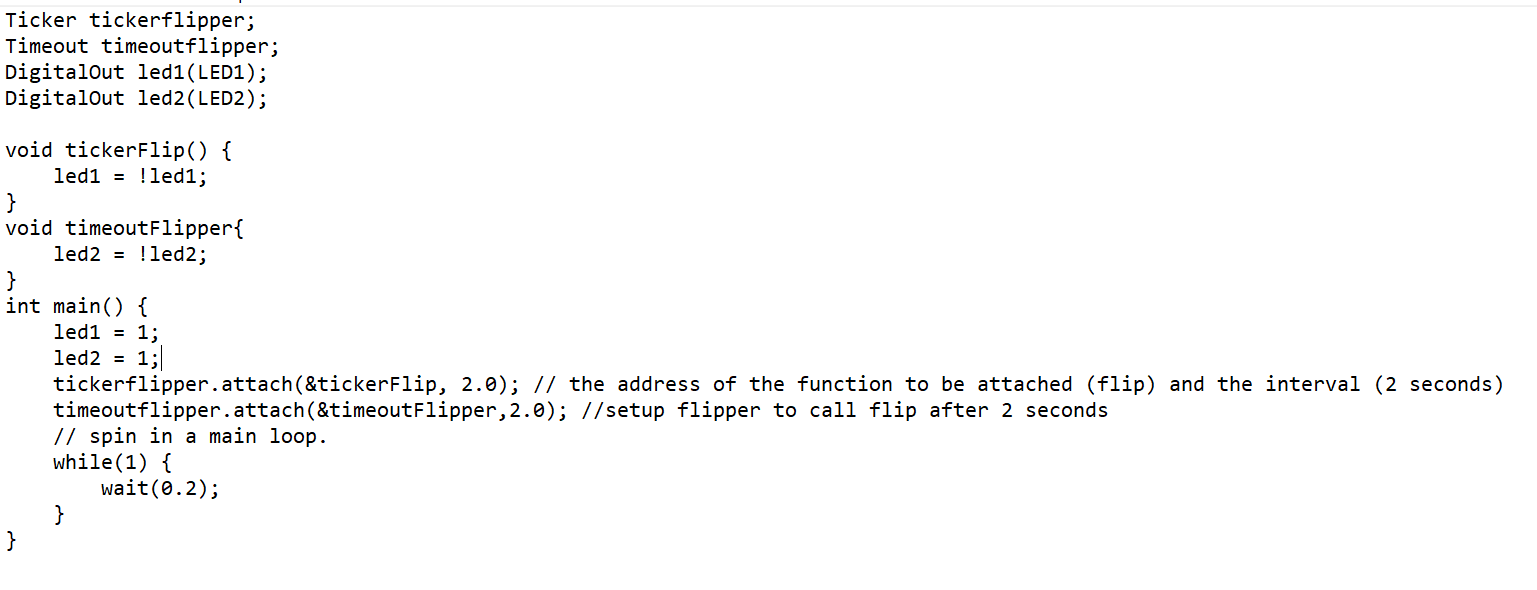
There is in total 10 different songs that have been provided for this lab, each of these songs have their own name (that has been split into two attributes name1 and name2), notes, beats, tempos or speed, and length. In order to access these attributes, you can:

**<SONG>.note[noteIndex]** or \***(<SONG>.note + noteIndex)**

*Note: There are* ***no notes*** *in the songs provided, please take care when dealing with them. It might be useful to review the module 8 lab solution to remember how to play a song.*

## Timeout vs Ticker

The Ticker interface is used to setup a **recurring interrupt to repeatedly call a function** at a specified rate whereas Timeout setup **an interrupt to a call function** after a specified delay. For instance, if we have the following code:



After two seconds **led1 and led2** will turn off, but after another two more seconds **just led1** will turn on because the tickerFlip function will execute as long as the program continues while the timeoutFlipper will just execute once, **unless is setup again**.

## Your Application Code

In this exercise, you are required to complete the main.cpp provided and fulfil the following requirements:

* Define inputs and outputs that are missing.
* Complete the functions update\_lcd\_leds (), adjust\_volume() , pause\_button\_handler() , and get\_input\_handler().
* Use switches 1 – 3 to insert a song in binary form from the **array of songs**. For instance, if all the switches are off or zero then we would end up playing the first song.
* Once you have chosen the number, press the push button once.
* Press the push button again to confirm your choice, if you don’t anything for 5 seconds, everything will continue normally.
* Display the name of the song on the LCD when playing and a confirmation message when changing the song.
* Use the timeout and ticker interface to wait for the 5 seconds of confirmation and play music respectively.
* Use BUTTON1 to stop/play a song.
* When a song is played the blue LED should be on
* When a song is paused the red LED should be on
* When changing a song, the yellow LED should be on
* Create two interrupts for:
  1. Playing/stopping
  2. Getting\_input
* Note that:
  + Playing/stopping will be triggered when BUTTON1 is pushed
  + Playing/stopping interrupt will call the function pause\_button\_handler()
  + Getting\_input will be triggered when the push button is pressed
  + Getting input interrupt will call the function get\_input\_handler()
* In the main program:
  + Clear the LCD display
  + Start all the interrupts
  + Send the instruction of how to use the audio player via UART
  + Update the LCD and volume in the while loop.