QUT Science and Engineering

EGB240 Electronic Design

Assessment 1: Siren Design (Part B)

Semester 1, 2020

Deliverable Summary

Microcontroller code (10%)	11:59pm Friday 15th May

Assessment 1: Siren Design (Part B)

Assessment 1 is split into two parts, in which you will design a two-tone or time-varying siren via two different methods. In Part B you will synthesise a siren via software running on a microcontroller.

Task

Your task is to design and implement a swept-frequency siren in software using a microcontroller. This project is to be completed individually. You must submit your working microcontroller code (both source and compiled) electronically via Blackboard. Your design will be evaluated in a test fixture against the target specification according to the assessment criteria and standards.

Specification

Your design must conform to the following specification:

- Implemented using the TeensyBOB development board
- Piezoelectric buzzer to be connected from JOUT to GND
- Buzzer excitation to be achieved using 50% duty cycle PWM
- One second period for the frequency sweep
- Minimum and maximum frequencies adjustable up to ±500 Hz
- Controlled via pushbuttons S1-S4

The parameters of the frequency sweep for your siren design are determined by the last four digits of your student number according to Figure 1 and Table 1 on the following page.

On initialisation, f_{max} and f_{min} should be set to their nominal values as listed in the table. When in the corresponding adjustment mode (see below), f_{max} and f_{min} should be adjustable up to ±500 Hz of their nominal values (but not beyond this range). Adjustments to f_{max} and f_{min} should be in discrete increments or decrements of f_{inc}.

The siren must be operable by a user via pushbuttons S1 through S4, as defined below:

Pushbutton S1: Cycles through modes. RUN → ADJUST FMAX → ADJUST FMIN → RUN

Pushbutton S2: Increases frequency f_{max} or f_{min} by f_{inc} when in ADJUST FMAX or ADJUST FMIN modes respectively.

Pushbutton S3: Decreases frequency f_{max} or f_{min} by f_{inc} when in ADJUST FMAX or ADJUST FMIN modes respectively.

Pushbutton S4: Resets mode to RUN and f_{max} and f_{min} to their nominal frequencies.

All buttons should be de-bounced and perform only a single action for each press-and-release cycle by a user.

When in mode RUN the output should be frequency-modulated according the siren frequency sweep specification. When in modes ADJUST FMAX or ADJUST FMIN, the output frequency should be held constant at f_{max} or f_{min} respectively. On initialisation, the mode should be set to RUN.

Siren Frequency Sweep Specification

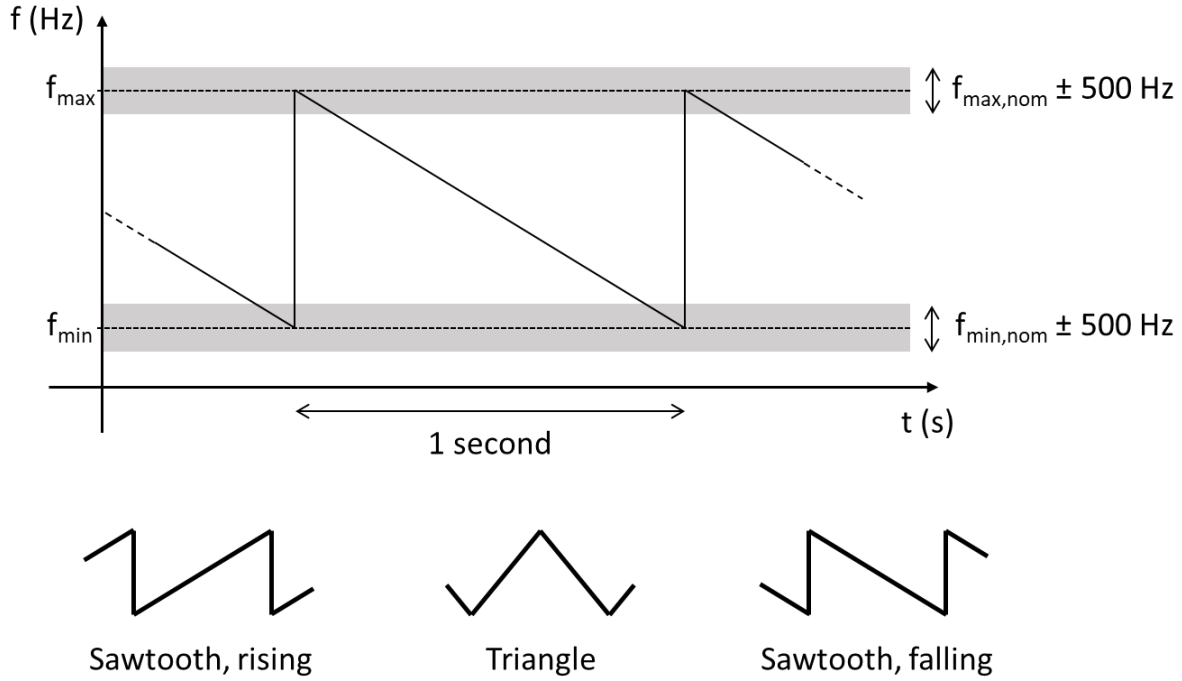

Figure 1: Siren frequency sweep specification.

Student number: nXXXXABCD

A	Waveform
0	Sawtooth, rising
1	
2	
3	
4	Sawtooth, falling
5	
6	
7	Triangle
8	
9	

B	f_{min}, nominal
0	2300
1	
2	2400
3	
4	2500
5	
6	2600
7	
8	2700
9	

C	f_{max}, nominal
0	4600
1	4700
2	4800
3	4900
4	5000
5	5100
6	5200
7	5300
8	5400
9	5500

D	f_{inc}
0	20
1	25
2	
3	50
4	
5	100
6	
7	125
8	
9	250

Table 1: Mapping of student number to siren frequency sweep parameters.

Resources

Components

You will be supplied with a kit of components to use in your microcontroller siren design. This kit can be collected from the Store from the beginning of Week 6, and will contain:

- 1 x Development board w/ breadboard and USB cable (also used for DVR project)
- 1 x Piezoelectric buzzer

Datasheets

A link to the datasheet for the piezoelectric buzzer supplied in the project kit will be made available on Blackboard. The schematic for the QUT-designed development board will also be made available on Blackboard, along with a link to the datasheet for the ATmega32U4 microcontroller used on the Teensy 2.0 board. Further documentation for the Teensy 2.0 board, including the schematic, can be found on the Teensy webpage.

Template project and skeleton code

A template project for PlatformIO will be provided on Blackboard, including skeleton code which initialises the ATmega32U4 microcontroller. Use of this template project and skeleton code is recommended, but not compulsory. An example makefile will be provided for users of other toolchains, however, please note that the EGB240 teaching team will not provide support for development environments other than PlatformIO.

Software

We recommend the use of the following software packages for the completion of this project. All packages are available for you, as QUT students, to install and/or use on both QUT workstations and personal computers at no cost.

Microcontroller programming: **PlatformIO IDE for VSCode**

Available for download from: http://bit.ly/EGB240-PlatformIO

Teensy bootloader: **Teensy Loader**

Available for download from: http://bit.ly/EGB240-Teensy

Note: Teensy Loader is automatically installed when using PlatformIO IDE for VSCode.

Deliverables

Microcontroller code (10%) — Due: 11:59pm Friday 15th May

You are required to submit your working microcontroller code implementing a siren as an electronic submission via Blackboard. You must include both a compiled HEX file, and your C source code. Your programme will be tested for functionality by the teaching team and graded against the assessment criteria.