



# CPEN 291

## Computer Engineering Design Studio I

### Lab 2: Morse Code with Arduino (Make Arduino Talk)

ECE – UBC

2016 W2

© *Farshid Agharebparast*

January 2017

Electrical and Computer Engineering  
University of British Columbia

# Lab 1 Review Discussion

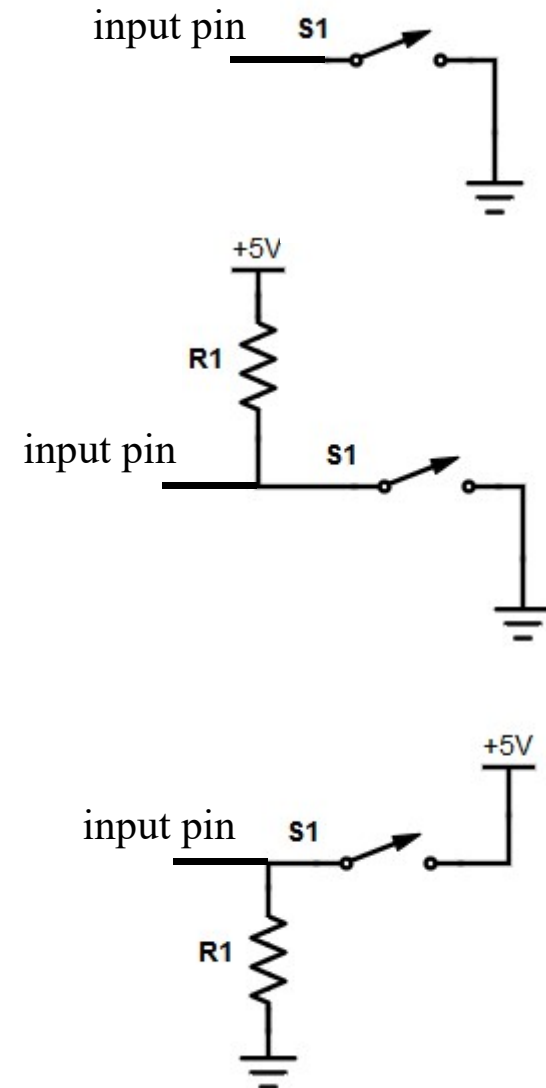
## ❑ Pull-up or pull-down resistors

❖ See the incorrect circuit on the right.

- If you do not use a pull-up resistor, when the switch is open, the pin is not connected to anything, while we want to ensure that the logic is high.

❖ With a pull-up resistor, it is guaranteed that the logic is high (5V) when the switch is open. When the switch is closed, the pin is connected to ground. Choose R1 to limit the current.


❖ The logic will be reversed when we use a pull-down resistor.



## Objective

- ❑ Embedded system design with basic audio and display interface;
  - ❖ Also engineering skills like documentation, group work, ...
- ❑ Application: To create a Morse code generator (audible and visual).
  - ❖ We will use the basic Piezo buzzer (audible) and a 7-seg LED (visual) as the output devices
  - ❖ Components: Arduino UNO, proto-shield/mini breadboard, Piezo buzzer, 7 segment LED, LED, other (resistors, ...)
  - ❖ You must code all functionalities yourself (that is do not use any not-built-in Arduino libraries), if in doubt ask.
  - ❖ Do not use the Piezo buzzer until you fully test your program with the LED. It would be loud and bothersome to others.

## Morse Code

- ❑ Morse code is a method of transmitting text information as a series of on-off tones, lights, or clicks that can be directly understood by a skilled listener or observer without special equipment.
- ❑ International Morse Code encode alphabet as standardized sequences of short and long signals called "dots" and "dashes" (or "dits" and "dahs").
- ❑ Each character (letter or numeral) is represented by a unique sequence of dots and dashes.
  - ❖ For example, letter A is encoded as one dot followed by one dash:  

  - ❖ The dot duration is the basic unit of time measurement in code transmission.

# International Morse Code

- We are going to consider English letters, digits, punctuations (only the four shown below) and error.

## International Morse Code

1. The length of a dot is one unit.
2. A dash is three units.
3. The space between parts of the same letter is one unit.
4. The space between letters is three units.
5. The space between words is seven units.

A • —  
B — • • •  
C — • — •  
D — • •  
E •  
F • • — •  
G — — •  
H • • • •  
I • •  
J • — — —  
K — • —  
L • — • •  
M — —  
N — •  
O — — —  
P • — — •  
Q — — • —  
R • — •  
S • • •  
T —

U • • —  
V • • • —  
W • — —  
X — • • —  
Y — • — —  
Z — — • •

1 • — — — —  
2 • • — — —  
3 • • • — —  
4 • • • • —  
5 • • • • •  
6 — • • • •  
7 — — • • •  
8 — — — • •  
9 — — — — •  
0 — — — — —

Full Stop (.)

• — • — • —

Comma (,)

— — — • — — —

Colon (:)

— — — — • •

Hyphen or Dash (-)

— • • • • —

Erase (or Error)

• • • • • • •

Source: [https://en.wikipedia.org/wiki/Morse\\_code](https://en.wikipedia.org/wiki/Morse_code)

## Morse Code

- Each character (e.g. letter or numeral) is represented by a unique sequence of dots and dashes.
  - ❖ For efficiency, the length of each character in Morse is approximately inversely proportional to its frequency of occurrence in English. Thus, the most common letter in English, the letter "E," has the shortest code, a single dot.
- The dot is the basic element in Morse code. All other elements are defined in terms of multiples of the dot length.
  - ❖ The duration of a dash is three times the duration of a dot.
  - ❖ Each dot or dash is followed by a short silence, equal to the dot duration.
  - ❖ The letters of a word are separated by a space equal to three dots (one dash), and the words are separated by a space equal to seven dots.

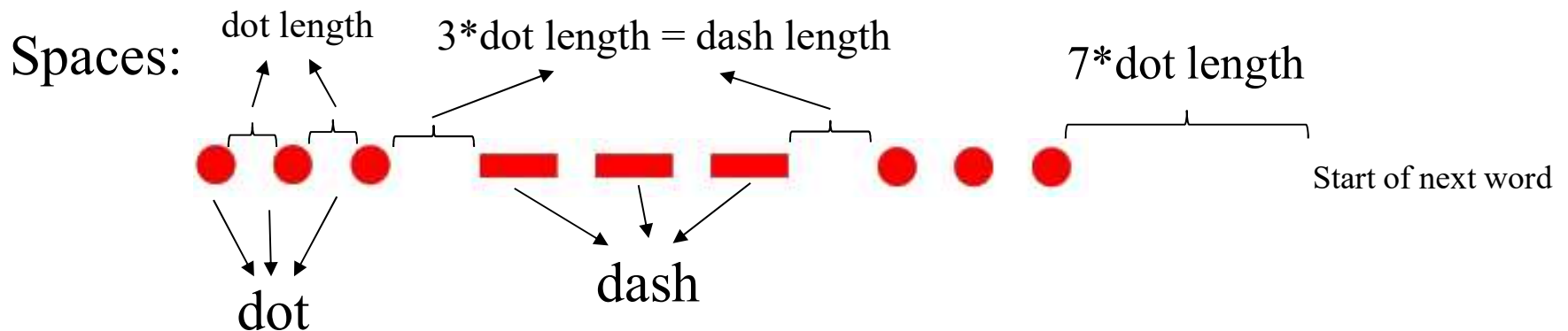
## Morse Code - Example

- The international SOS distress signal will be:

(<http://en.wikipedia.org/wiki/SOS>)



- In order to transmit SOS, we have the following lengths:



- The Morse code speed is measured in WPM (words per minute). Assuming a standard word duration of 50 dots (using PARIS as a typical word), we have:

$$Speed (WPM) = 1200 / dot\_length\_in\_milliseconds$$

## Piezo Buzzer

- ❑ Piezo functions based on the concept that some crystals make a spark when squeezed. The reverse effect also works: spark that crystal and it flexes.

❖ Piezo is from the Greek work *piezein* meaning squeeze.

Image of a  
Piezo buzzer:



Image of an actual  
Piezo disk used  
inside the package:



- ❖ Piezo is used to make sound: we cause it to flex back and forth which moves air to make sound. It can also be used as a sensor, but not in this lab.
- ❖ The fireplace lighter is an application of Piezo.





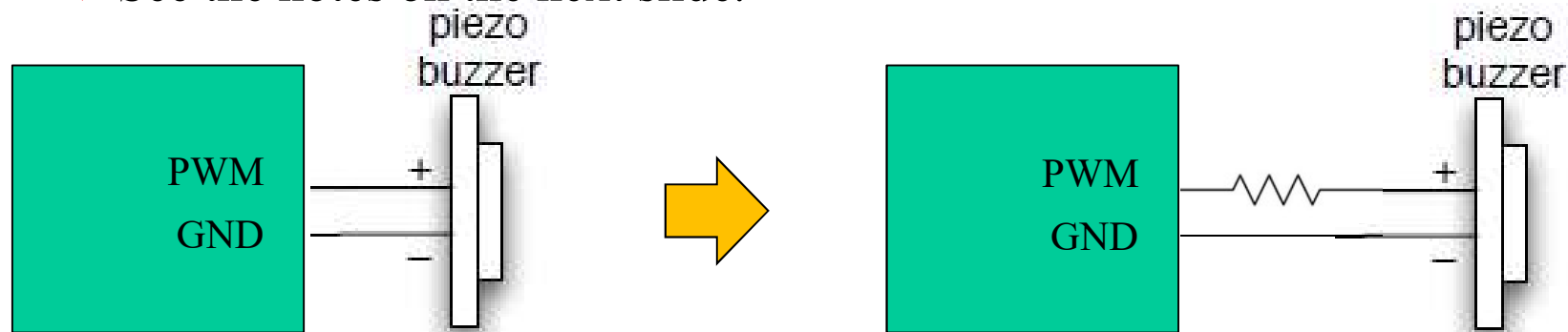
## Using Piezo Buzzer

- ❑ Our Piezo is a plain piezo. It works when a square-wave is applied to it.
  - ❖ Some other Piezo buzzers on the market have a driver circuit inside that would allow DC for excitation, but it generates a fixed frequency buzz (e.g. 2kHz).
  - ❖ Based on our Piezo's datasheet: "Do not apply DC bias to the piezoelectric buzzer; otherwise insulation resistance may become low and affect the performance."
- ❑ So connect the Piezo buzzer to a free PWM pin.
  - ❖ The generated sound might bother others. Limit the volume by the implemented PWM duty cycle.
  - ❖ In theory, the added resistor in series with the Piezo buzzer (like 10k ohms or larger) is another option. There is no need here, but if we wanted variable volume control, we could use a POT (variable resistor) .

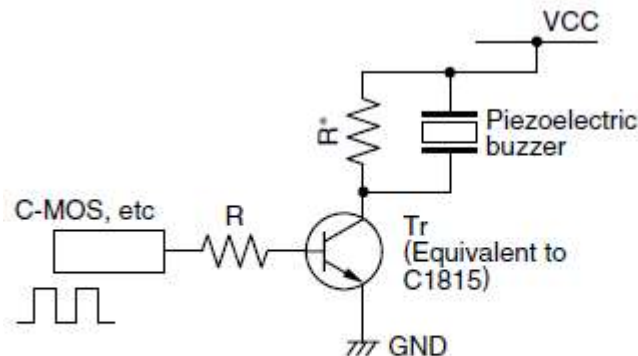
## Piezo Buzzer Circuit

- The simplest circuit is to connect the Piezo buzzer to a free PWM pin and the other pin to GND (or with a resistor in series).

❖ See the notes on the next slide.



- Sometimes a Zener diode is used for the IO pin protection.
- The datasheet's recommended operating circuit example is:



\* Resistor to do charging and discharging to a piezoelectric element (Value of about  $1k\Omega$  is good efficiency).

## Piezo Buzzer: Arduino Functions

□ In order to write to Piezo, you have two options:

❖ Using `analogWrite()`

see <http://arduino.cc/en/Tutorial/PWM> and  
<http://arduino.cc/en/Reference/AnalogWrite>

Note that PWM wave will continue once `analogWrite()` is called, until the next call to `analogWrite()` on that pin.

❖ Using `tone()`

see <http://arduino.cc/en/reference/tone>

`tone()` produces a wave of specific frequency and duration.

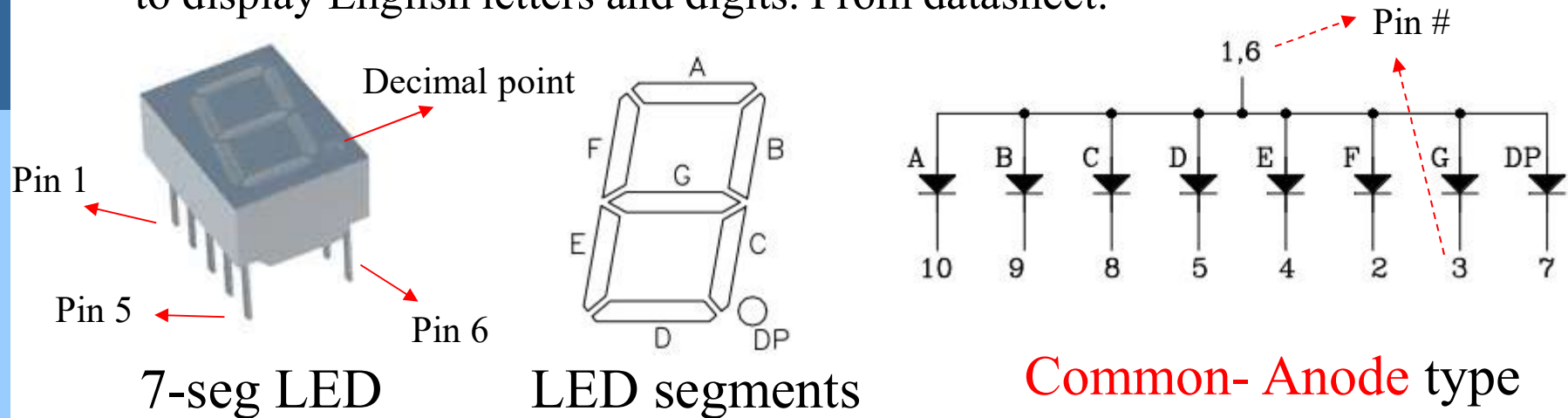
See the above reference webpage for the limitation of `tone()`.

## Piezo Buzzer Notes

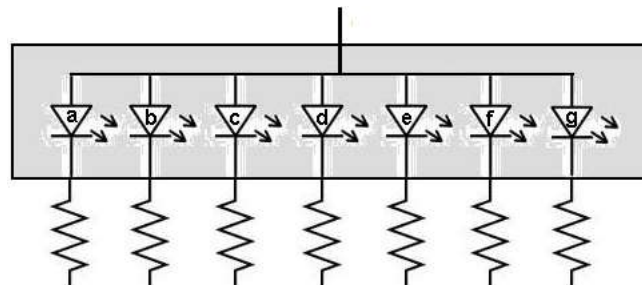
- ❑ Piezo sounder is normally designed to produce sound at a specific stable frequency, but we can use them at other frequencies as well and even use them to create melody.
  - ❖ This would not be the case if the Piezo component is designed to work with DC input.
- ❑ Normally Piezo disk (used in the buzzer) has electrical polarity, but for some buzzer packages one can just ignore it.
- ❑ Since Piezo can work as a sensor, be careful when you are using it. Do not keep it connected to Arduino when you are not using it, and make sure you do not squeeze its disk or create loud sound close to it.
- ❑ See the posted datasheet's "precautions for use"

## 7-segment LED

- ❑ You are somewhat familiar with the 7-segment LEDs from APSC 160. In this lab, you first connect one **7-segment LED** to Arduino and then write the code to display English letters and digits. From datasheet:

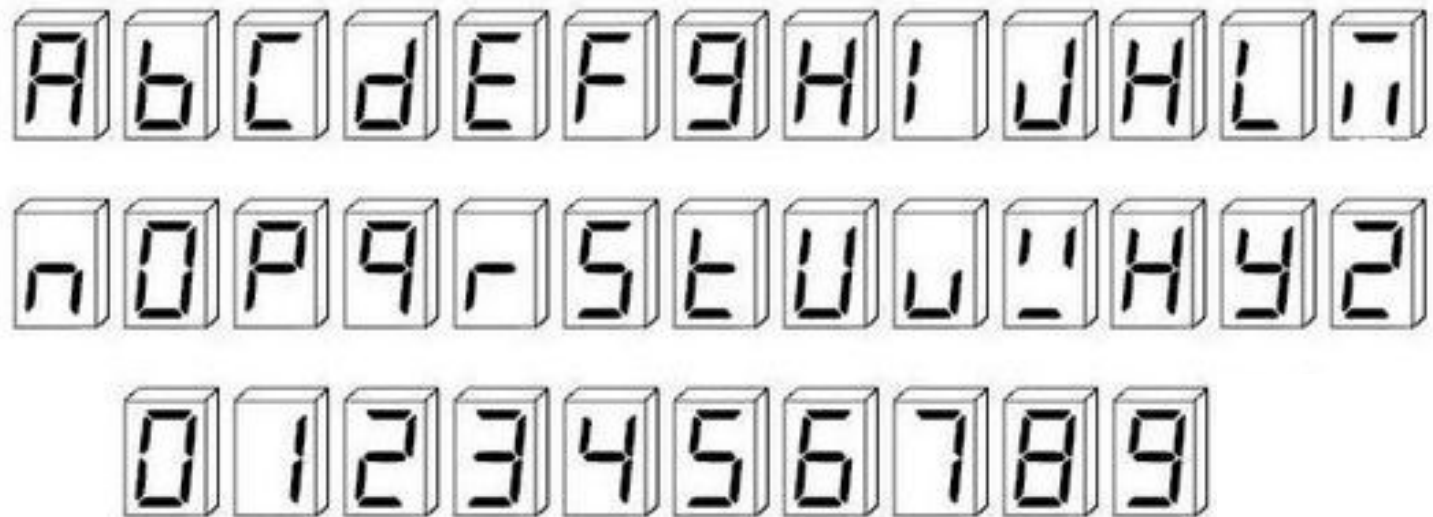


- ❑ Pin 1 (and 6) is connected to the anode of all LED segments, and it is connected to 5V. The Cathode of each LED segment is connected through one resistor to one digital IO pin of Arduino (use external resistors as shown here):



## 7-segment LED

- ❑ Be careful when you wire the 7 segment LED to not burn out its segments.
- ❑ Since we do not have an exact representation for every letter on the 7 segment LED, use the following convention (again case-insensitive) and use decimal point for period (the only punctuation displayable):



- ❑ For example, to display A, you will need to turn on all LED segments except segments *D* and *DP* (decimal point).

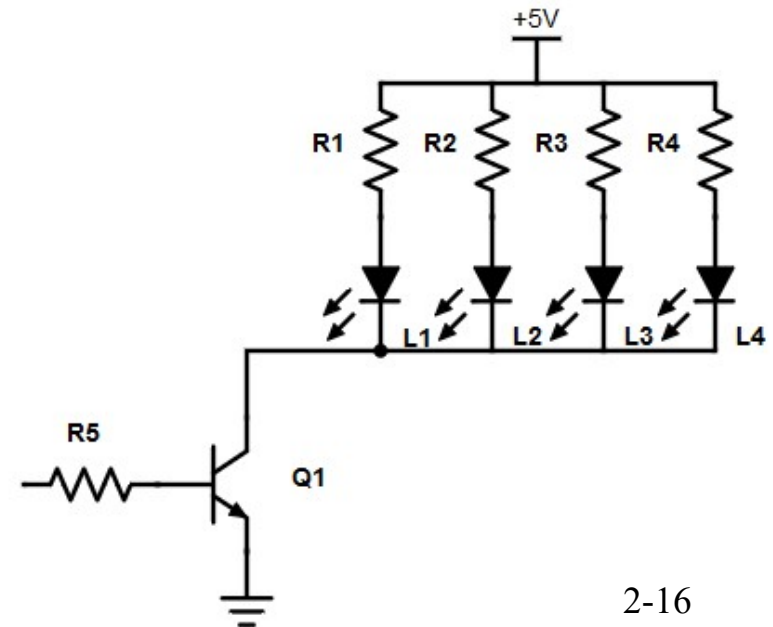
## 7-segment LED Programming

- ❑ At the lowest level, you will need to use `digitalWrite()` to control each segment of the LED.
- ❑ Use programmer-defined functions, arrays, and lookup tables.
- ❑ Your program should have a good structure by using functions and arrays.
  - ❖ Do not include unnecessary or repetitive statements.

## Additional Discussions:

### Attaching Devices That Needs More Current

- ❑ An Arduino pin can deliver 40mA of current to an output device.
- ❑ If an output device that you want to power requires more current, then you need some form of amplification.
- ❑ A simple way to do so is to use transistors. The example below show how you can connect 4 LEDs (all controlled to behave exactly the same) to an Arduino output pin.
- ❑ Some possible values to select:
  - ❖  $R1=R2=R3=R4= 270 \text{ ohms}$
  - ❖  $R5 = 1K \text{ ohms}$
  - ❖ Transistor: 2N3904 or equivalent





## References

❑ Arduino Reference:

<http://arduino.cc/en/Reference/HomePage>

❑ Arduino: <http://arduino.cc/>

❑ Datasheets:

❖ See the datasheets linked/posted on Connect