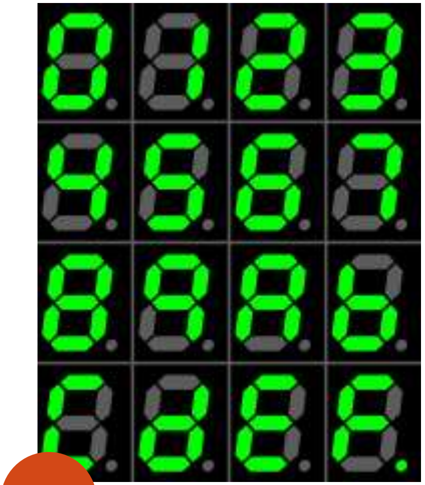
IoT

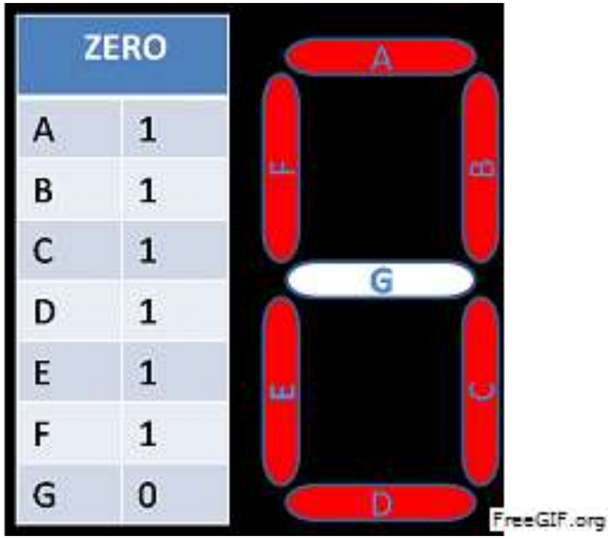
Arduino-Seven Segment, Buzzer Interfacing

Dr. Sarwan Singh
Deputy Director
NIELIT Chandigarh

Agenda

- LED vs Seven Segment
- Basic principle
- Seven Segment Interfacing
- Buzzer basic, interfacing
- Practical Example/Experiment





ZERO	
A	1
B	1
C	1
D	1
E	1
F	1
G	0

Seven Segment Displays

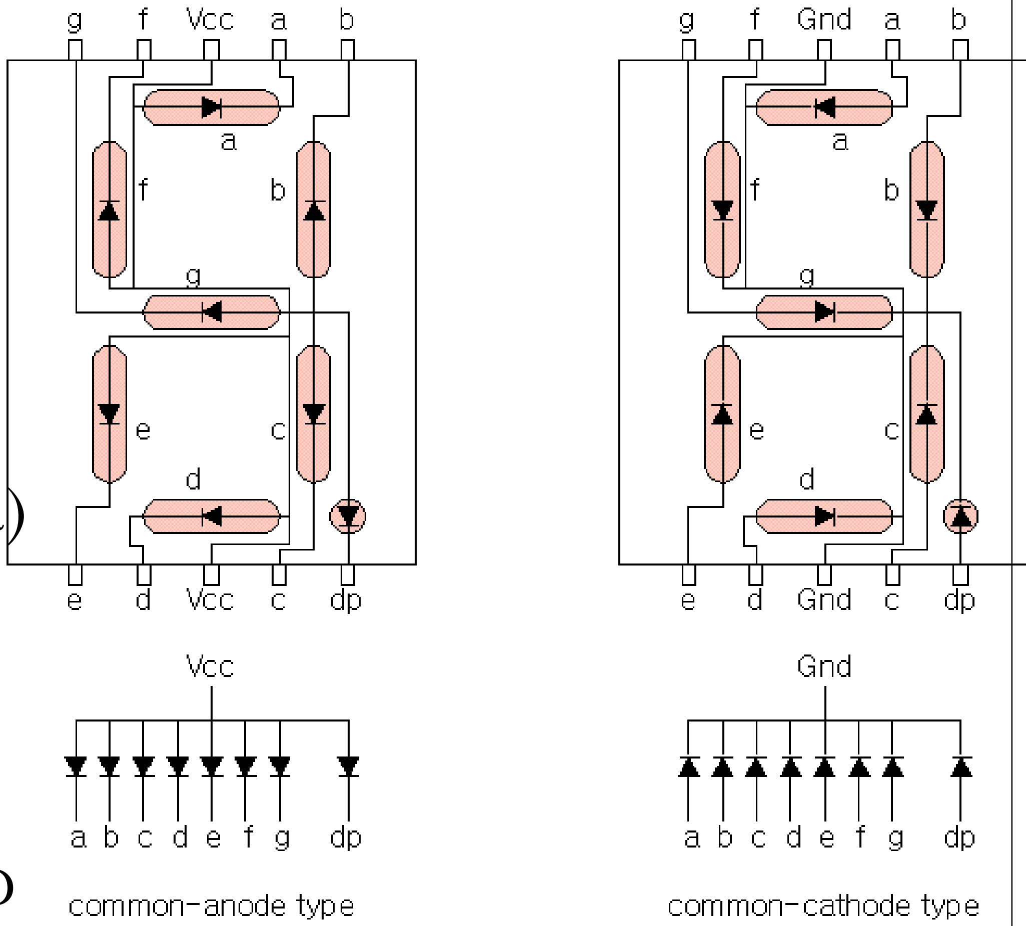
This presentation will demonstrate how

- A seven-segment display can be used to display the decimal numbers 0-9 and some alpha characters.
- A common anode seven-segment display works.
- A common cathode seven-segment display works.
- To select the resistor value for a seven-segment display.

Retro
LED Watch
(Circa 1970s)

Type of seven segment

- A Seven-Segment Display (SSD) is simply a figure eight grouping of LEDs {some include a decimal point (DP)}.
- Each Segment is labeled (a) thru (g).
- SSDs are available in two configurations
  - Common Cathode (all LED cathodes are connected)
  - Common Anode (all LED anodes are connected)

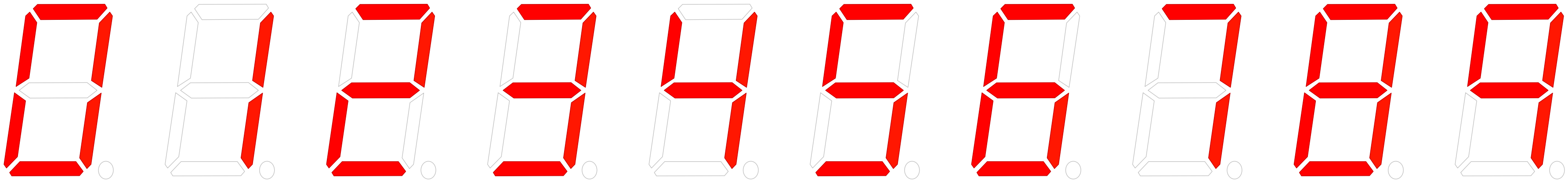


common-anode type

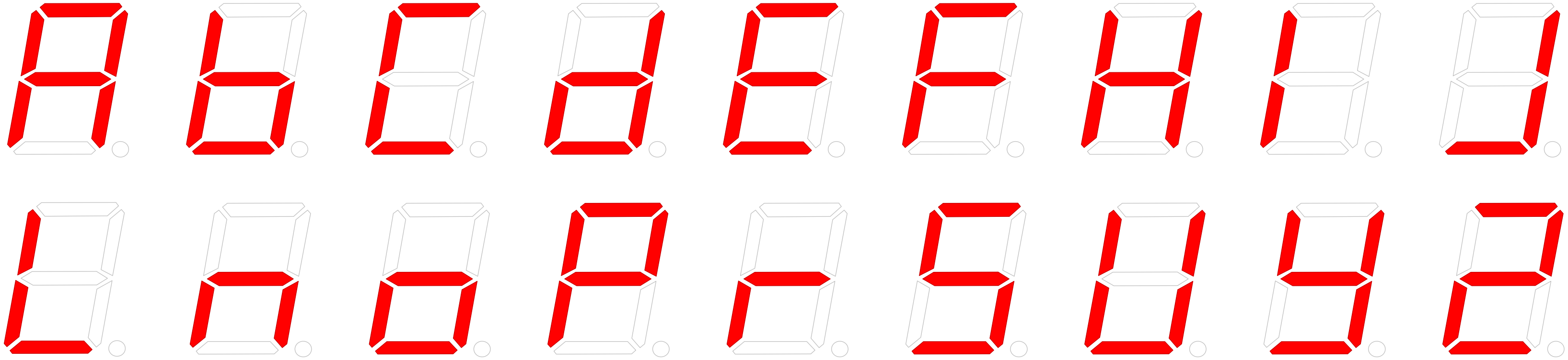
common-cathode type

Display Possibilities

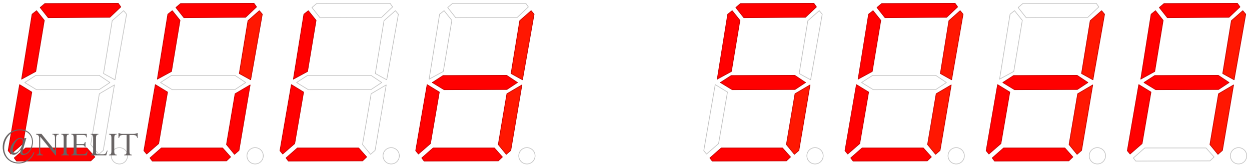
Decimal Digits 0-9



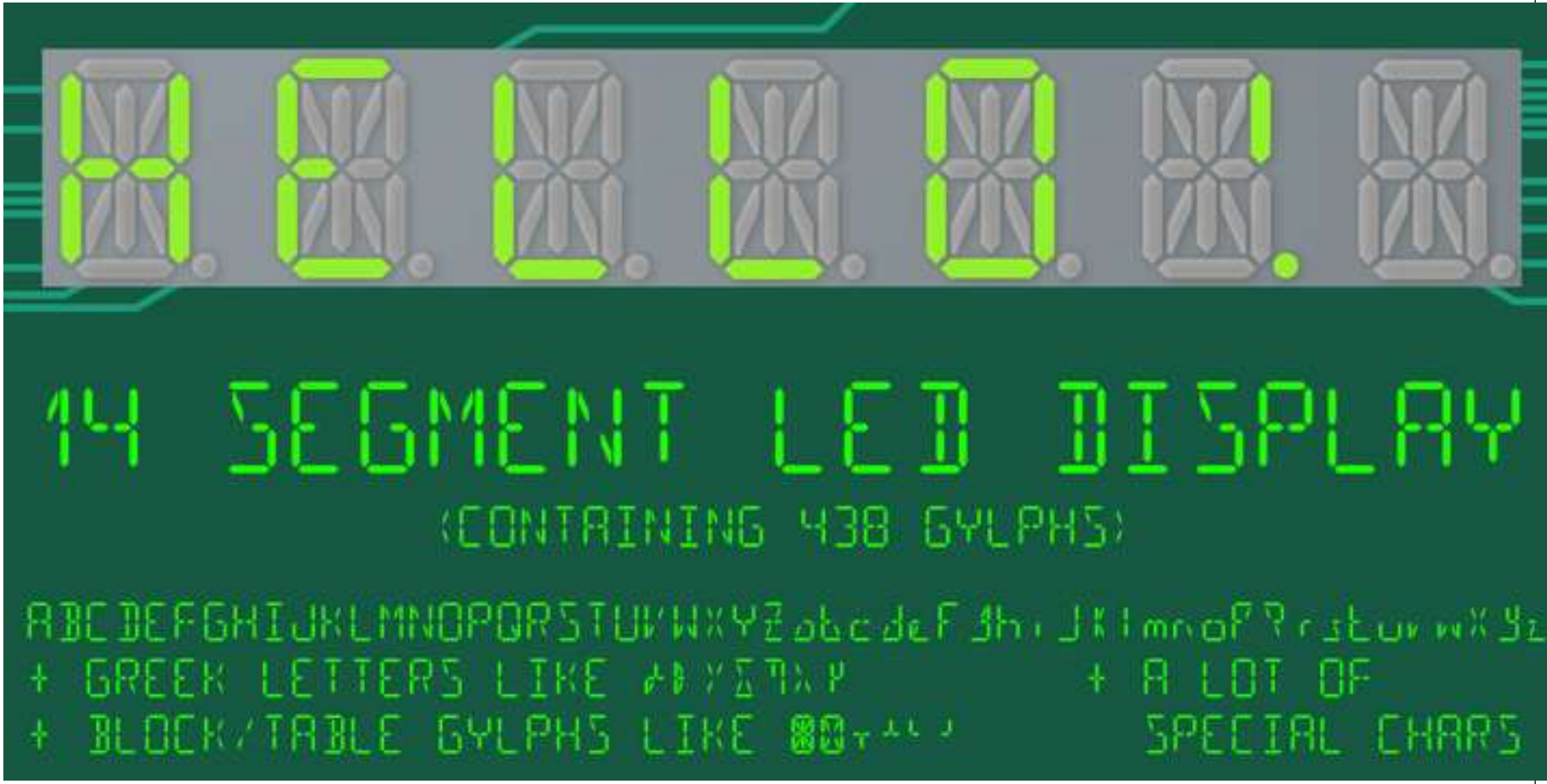
Select Alpha Characters

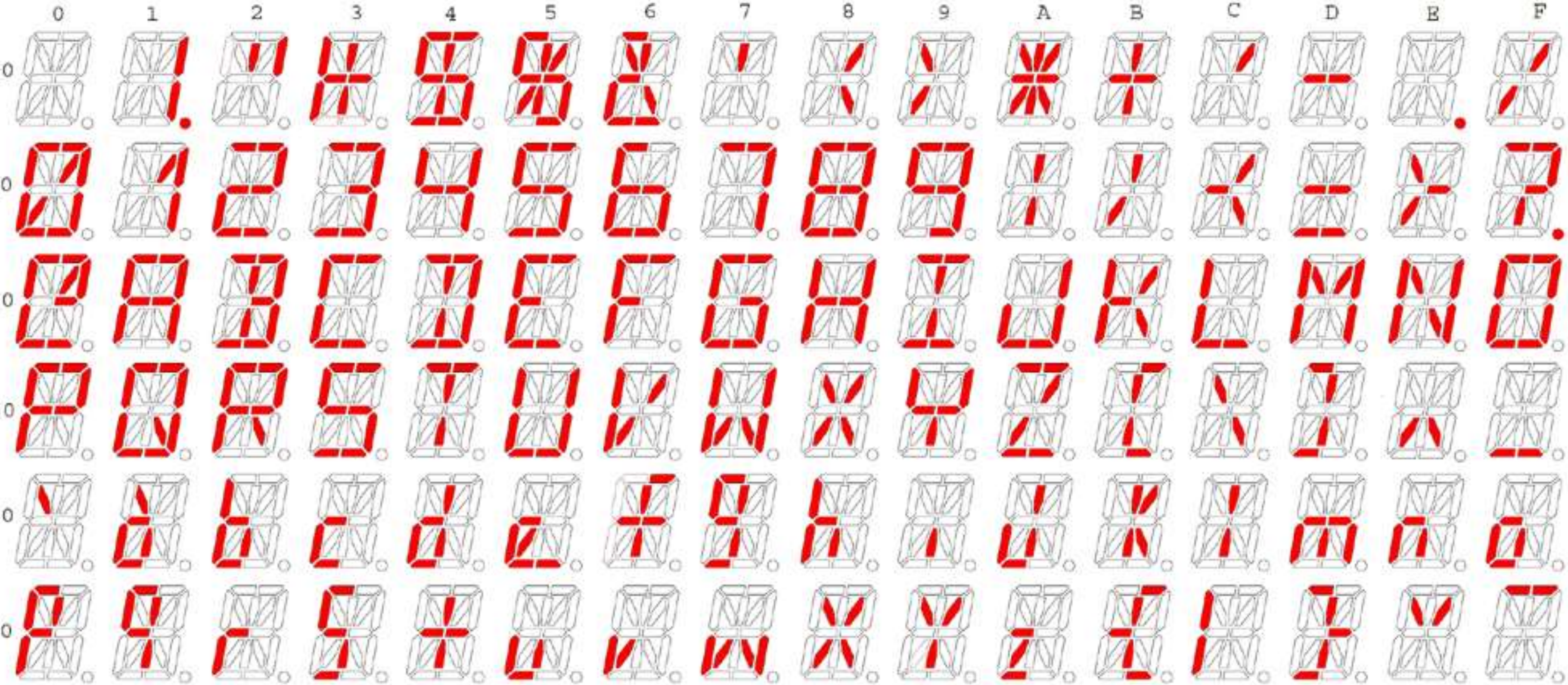


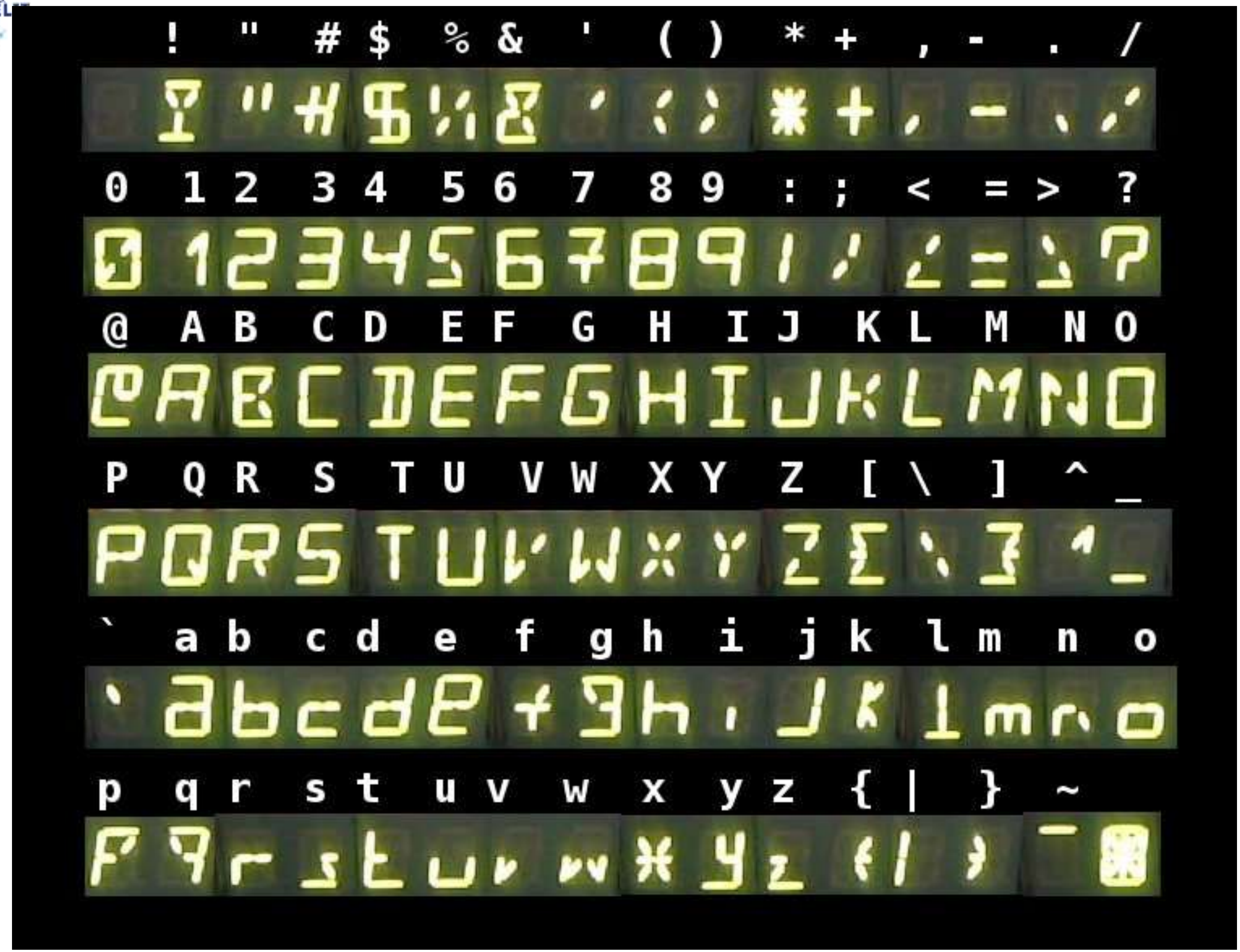
Simple Messages



14 segment LED display

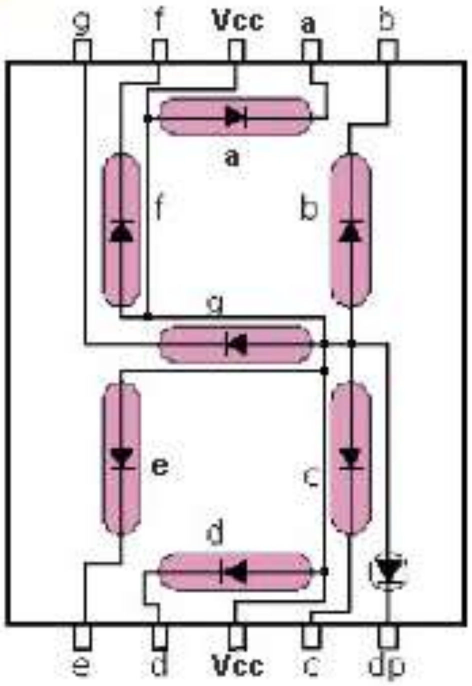



0
1
2
3
4
5
6
7
8
9
A
B
C
D
E
F
20
30
40
50
60
70


! " # $ % & ' ( ) * + , - . /
0 1 2 3 4 5 6 7 8 9 : ; < = > ?
@ 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 [\ ] ^ _
` 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 { | } ~

Display	g	f	e	d	c	b	a
0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	1
2	0	1	0	0	1	0	0
3	0	1	1	0	0	0	0
4	0	0	1	1	0	0	1
5	0	0	1	0	0	1	0
6	0	0	0	0	0	1	0
7	1	1	1	1	0	0	0
8	0	0	0	0	0	0	0
9	0	0	1	0	0	0	0

Common Anode



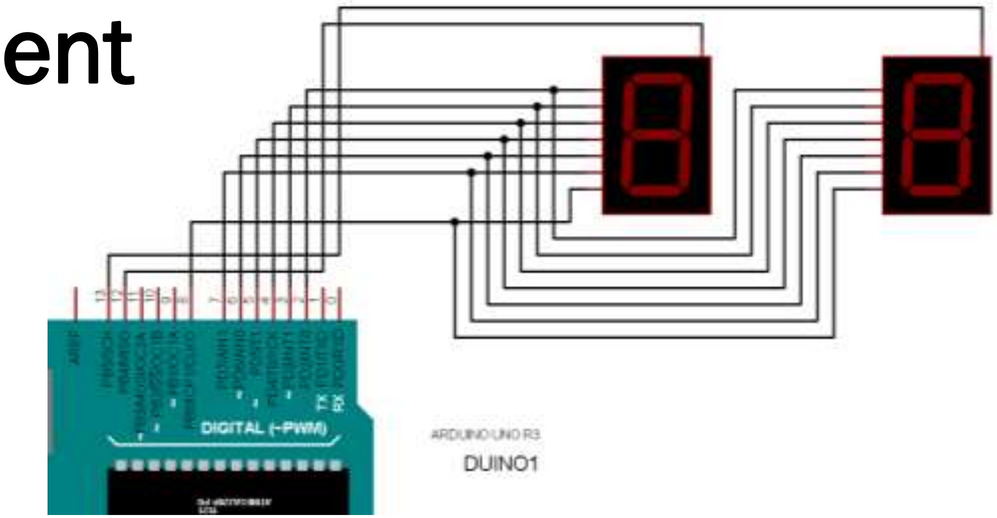
Type of seven segment

```
void setup()
{  // define pin modes
    pinMode(2,OUTPUT);
    pinMode(3,OUTPUT);
    pinMode(4,OUTPUT);
    pinMode(5,OUTPUT);
    pinMode(6,OUTPUT);
    pinMode(7,OUTPUT);
    pinMode(8,OUTPUT);
}
```

```
void loop()
{   // loop to turn leds of seven seg ON
   for(int i=2;i<9;i++)
     {    digitalWrite(i,HIGH);
           delay(600);
     }
     // loop to turn leds of seven seg OFF
     for(int i=2;i<9;i++)
     {    digitalWrite(i,LOW);
          delay(600);
     }
       delay(1000);
}
```

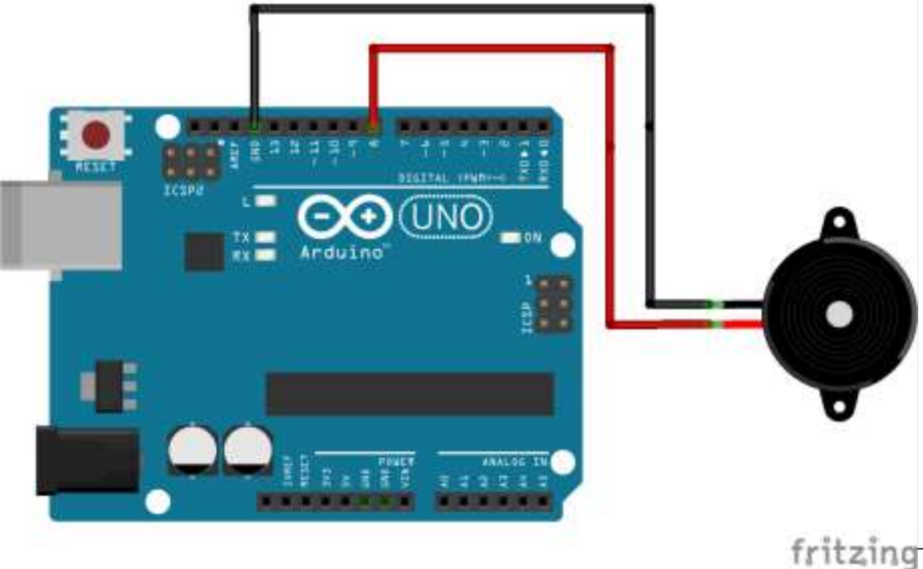
```
int digits[][7] = {
      {0,0,0,0,0,0,1}, // zero
      {1,0,0,1,1,1,1}, //one
      {0,0,1,0,0,1,0}, //two
      {0,0,0,0,1,1,0} // three
};
void show( int dgt ){
      for (int i=2,j=0; i<9 ; i++,j++){
              digitalWrite(i, digits[dgt][j]);
      }
}
void loop() {      show (3);          }
```

Two seven segment



```
int c=0;
 void loop()
{
     t=c%10;        p=c/10;
     unsigned long startTime = millis();
     for (unsigned long elapsed=0; elapsed < 600; elapsed = millis() - startTime)
      {
          digitalWrite(seg1,HIGH);        digitalWrite(seg2,LOW);
          show(t);        delay(5);
          digitalWrite(seg2,HIGH);       digitalWrite(seg1,LOW);
          show(p);delay(5);
     }
}
```

Piezo Buzzer Interfacing

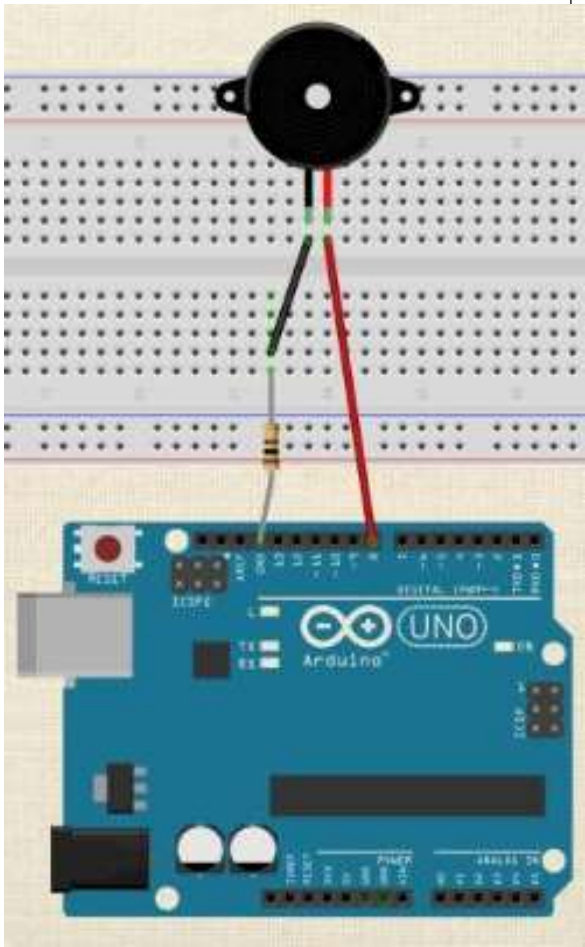


Introduction

- A **piezo buzzer** is generally used to signal user in the form of tone or **beep**.
- This type of **buzzer** widely used in alarm, domestic gadgets or in embedded systems product to provide some kind of indication or alert.
- Buy designing a small circuit we can **interface** it easily with**Arduino**.

Buzzer Interfacing

```
const int buzzerPin = 9;
void setup() {
        pinMode(buzzerPin, OUTPUT);
}
void loop() {
    tone(buzzerPin, 50);
    delay(50);
    noTone(buzzerPin);
    delay(100);
}
```



Tone function

tone(pin number, frequency in hertz);

tone(pin number, frequency in hertz, duration in milliseconds);

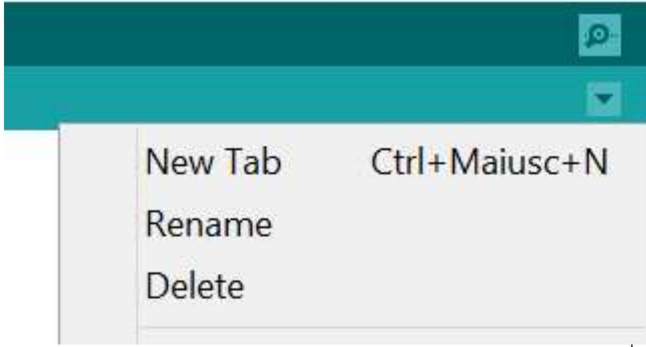
- The pin number that you will use on the Arduino.
- The frequency specified in hertz. Hertz are cycles per second.
- The frequency is an unsigned integer and can take a value up to 65,535 – but if you are trying to make tones for the human ear, then values between 2,000 and 5,000 are where our ears are most tuned.

```
//Specify digital pin on the Arduino that the positive lead of piezo buzzer
is attached.
int piezoPin = 8;
void setup() { }//close setup
void loop() {
        /*Tone needs 2 arguments, but can take three
          1) Pin#
          2) Frequency - this is in hertz (cycles per second) which
        determines the pitch of the noise made
          3) Duration - how long teh tone plays    */
 tone(piezoPin, 1000, 500);
}
```

Play a Melody using the tone() function

```
#include "pitches.h"
// notes in the melody:
int melody[] = {   NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4   };
// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {    4, 8, 8, 4, 4, 4, 4, 4       };
void setup() {
 // iterate over the notes of the melody:
 for (int thisNote = 0; thisNote < 8; thisNote++) {
  // to calculate the note duration, take one second divided by the note type.
  //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
  int noteDuration = 1000 / noteDurations[thisNote];
   tone(8, melody[thisNote], noteDuration);
  // to distinguish the notes, set a minimum time between them.
  // the note's duration + 30% seems to work well:
  int pauseBetweenNotes = noteDuration * 1.30;
   delay(pauseBetweenNotes);
  // stop the tone playing:
   noTone(8);
 }
}
void loop() { }
 // no need to repeat the melody.
```

- To make the pitches.h file, either click on the button just below the serial monitor icon and choose "New Tab", or use Ctrl+Shift+N.



```
/*************************************************
 * Public Constants
 *************************************************/

#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
```

Source: https://www.arduino.cc/en/Tutorial/ToneMelody?from=Tutorial.Tone