Electronic Basics #7: 7 Segment Display

Introduction

A **segmented display** is an electronic display device used to show numerical digits or characters. It consists of an array of individual segments, typically **LEDs**, that can be turned on or off to form different numbers or letters. The most common type of segmented display is the **7-segment display**, which is used in many electronic devices like clocks, calculators, and appliances.

Basic Structure

A 7-segment display contains **seven LEDs** arranged in a figure-eight pattern, where each segment is labeled from 'a' to 'g'. These segments can be turned on or off to form digits from **0 to 9**. Some displays also include an **eighth segment** for a decimal point, useful for displaying decimal numbers.

The segments are usually controlled through two types of connections:

- **Anode Configuration**: The anodes of all LEDs are connected together, and individual segments are controlled by grounding the corresponding cathode pin.
- **Cathode Configuration**: The cathodes are connected together, and individual segments are powered by applying voltage to the corresponding anode pin.

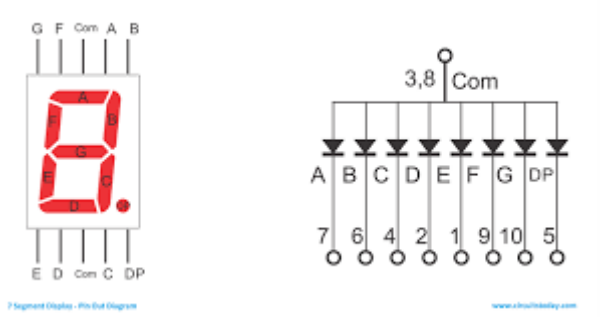

Fig7.1: 1 Digit Segmented Display

Controlling a Single Digit 7-Segment Display

For a single digit, a **7-segment display** consists of **7 individual LEDs** (segments a to g). Each segment can be controlled by sending current through its corresponding pin. To display a number, specific combinations of segments are turned on:

- **0**: Segments a, b, c, e, f, g are lit.
- **1**: Segments c and f are lit.
- **2**: Segments a, c, d, e, g are lit.
- **3**: Segments a, c, d, f, g are lit.
- And so on for the digits 4-9.

To control the 7-segment display with an **Arduino**, each segment is connected to a digital I/O pin on the Arduino. By setting the pins high or low, you can turn on or off each segment to form a number.

Specifications

Some important specifications of a 7-segment display are:

- **Voltage**: Typically **5V** for Arduino-compatible displays.
- **Current**: Usually around **20mA per segment**, which should be taken into account when designing circuits to avoid overloading the display.
- **Size and Color**: 7-segment displays come in various sizes and colors (common colors include red, green, and yellow).
- **Common Anode vs Common Cathode**: Displays may be common anode (all anodes connected) or common cathode (all cathodes connected). The type of display determines whether the segments are lit by providing a HIGH or LOW signal.

Controlling with Arduino

To control a 7-segment display with Arduino, you need to connect each segment (a to g) to a digital I/O pin on the Arduino board. You can use **digitalWrite()** to set each pin HIGH or LOW, depending on whether you are using a common anode or common cathode display.

Example code for a common cathode 7-segment display to display the digit '8':

```
int segments[] = {2, 3, 4, 5, 6, 7, 8};  // Segment pins (a to g)

int digit[10][7] = {
  {1, 1, 1, 1, 1, 1, 0},  // 0
  {0, 1, 1, 0, 0, 0, 0},  // 1
  {1, 1, 0, 1, 1, 0, 1},  // 2
  {1, 1, 1, 1, 0, 0, 1},  // 3
  {0, 1, 1, 1, 0, 1, 1},  // 4
  {1, 0, 1, 1, 0, 1, 1},  // 5
  {1, 0, 1, 1, 1, 1, 1},  // 6
  {1, 1, 1, 0, 0, 0, 0},  // 7
  {1, 1, 1, 1, 1, 1, 1},  // 8
  {1, 1, 1, 1, 0, 1, 1}   // 9
};


void setup() {
  for (int i = 0; i < 7; i++) {
```

```
    pinMode(segments[i], OUTPUT);
  }
}

void loop() {
  for (int i = 0; i < 7; i++) {
    digitalWrite(segments[i], digit[8][i]);  // Display '8'
  }
  delay(1000);  // Wait for 1 second
}
```

Four-Digit Display and Multiplexing

For a **4-digit 7-segment display**, the process of displaying multiple digits requires **multiplexing**. Since each digit in the 4-digit display shares the same set of segments, we can light up one digit at a time, cycling through the digits rapidly. This gives the illusion of all digits being displayed simultaneously.

Multiplexing works by turning on one digit at a time and rapidly switching between them. Each digit is controlled using **transistors or MOSFETs** to sink current. By controlling which digit is active at any given moment and rapidly cycling through them, the display appears to show all digits at once. The Arduino can control the **anode or cathode** of each digit and cycle through them at a high frequency to create a smooth display.

In this setup, **each digit shares the same segment lines**, but only one digit is powered at a time. The switching occurs fast enough (e.g., every few milliseconds) so that the human eye perceives all digits as lit simultaneously.