



## *LED Digit Displays - Large Quantities*

**Author:** Luis Espinal C.  
**Associated Project:** Yes  
**Associated Part Family:** CY8C27xxx  
**PSoC Designer Version:** 4.2

### **Abstract**

PSoC™ ICs are capable of handling a large variety of analog and digital signals. But how can we see those large quantities of processed information on standalone projects that need to be readable from a distance? LED displays are a good solution.

This Application Note presents a simple LED solution with a large display for 16 characters, 14 segments plus decimal point and a 16-segment bar graph. It utilizes only seven MCU I/O pins. This solution can be easily extended to handle additional LED digit groups by adding a single data output per group. Also, it is possible to manage larger digit quantities if seven segment displays are used.

### **Introduction**

With RAM increments, Flash introduction, additional modules and peripherals of all kinds, microcontroller units (MCU) have achieved great processing capabilities in recent years. Even so, for each MCU, I/O pin quantity is a limited resource that must be carefully managed.

When information must be displayed on LEDs, the user has to determine how much information should be displayed without sacrificing the efficient usage of the I/O pins.

In most projects, LED displays use four digits or less and tend to be overlooked during design time. Let us examine some of these design issues.

### **Duty Cycle**

To save resources in a MCU, when designing an LED display, it is a common practice, not to say obligatory, to multiplex digits on the display LED. Then, how do we come up with a limitation of how many digits must be multiplexed? Every time we add a single digit, the duty cycle or "on time" for each digit decreases, thus decreasing the luminosity. For example, if we have a two-digit display, each digit has a 50% (or near) duty cycle; but if you have a four-digit display, each digit has a 25% duty cycle, which greatly diminishes luminosity.

### **Refreshing Frequency**

Another variable to evaluate is the refresh frequency or how many times each digit is turned on every second. Contrary to popular thought, a higher refresh frequency is not necessarily better.

With refresh frequencies below 35 cycles per second per digit, flickering is noticeable, which makes it difficult to read. A higher refresh frequency rate may make the flickering disappear but does not necessarily provide better display performance.

For every new digit cycle, the MCU requires a fixed interval time for data digit introduction. This interval time is measured in machine cycles and is dependent on the digit's information introduction program, which temporarily disables the display and the LEDs.

As the refresh frequency increases, this introduction's fixed time begins to use the same amount of time it takes to turn on the LED. Because introduction time is fixed, the turn on time must be decreased in order to carry out with the period time of selected frequency. This situation decreases the effective duty cycle, as previously explained. This explains why a display, for example with two digits, has a duty cycle a bit below 50%.

Since MCUs take their time to introduce the data digit in their 50% assigned digit time and the refresh time is always slow enough, without blinking, a faster MCU doesn't significantly decrease the duty cycle of the LED.

**Hardware**

Let's see a solution when large digit quantities are required.

Keeping the previously discussed ideas in mind, the user can achieve good LED display performance for 16 digits, 14 segments and decimal point and a 16-segment bar graph by splitting it into two parts; one is multiplexed for digits 0 to 7 and the other is multiplexed for digits 8 to 15.

In this way, the LED duty cycle is near to 1/8 (12.5%). On the other hand, if all 16 digits are multiplexed together, the duty cycle is near 1/16 (6.25%), which reduces luminosity.

Each half is multiplexed and parallel, simultaneously turning on 0 and 8 digits, 1 and 9 digits, 2 and 10 digits and so on, by means of 74HC138, with every output pin driven by two transistors that feed the displays' anodes.

Two shift register blocks, with two daisy-chained 74HC164s on every block, provide the output register and current driver for the LED segments (bits 0 to 14). Bit 15 is assigned to the bar graph segment.

This can be seen in the schematics in figures 1, 2 and 3.

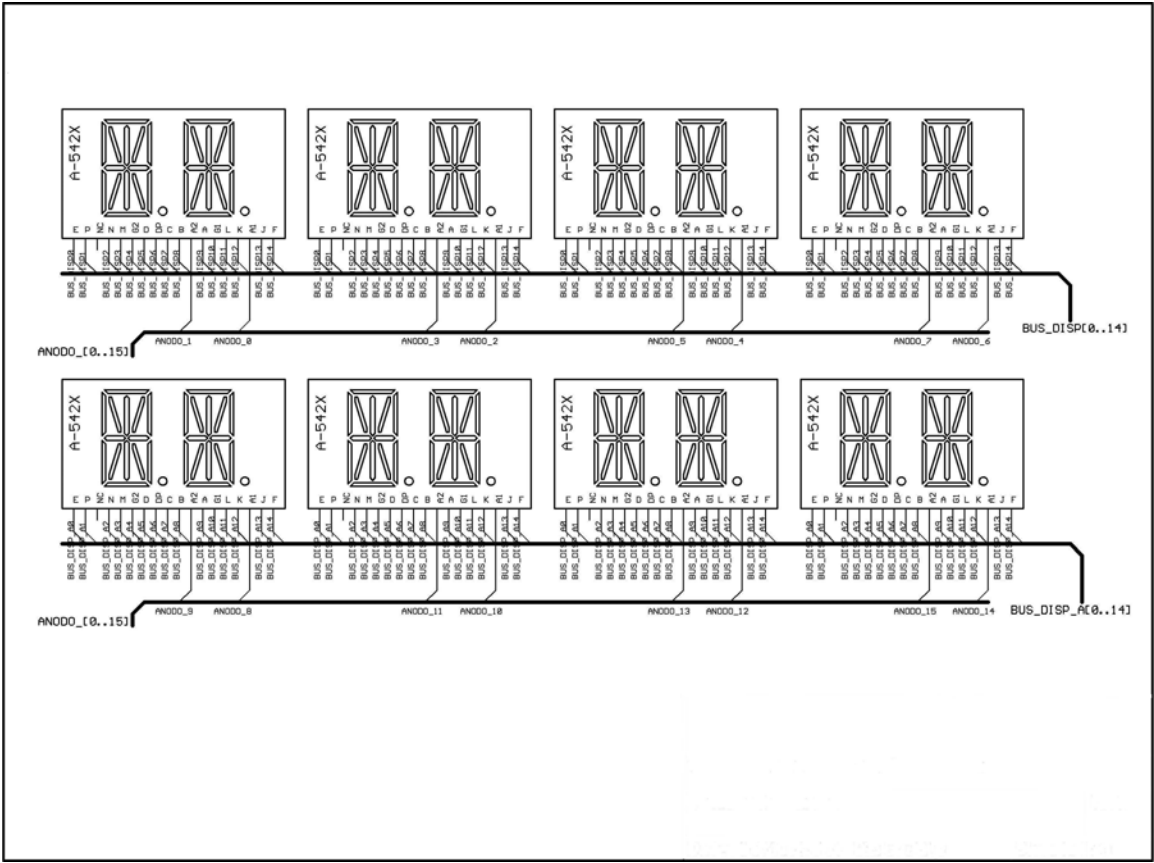


Figure 1. Segment Display

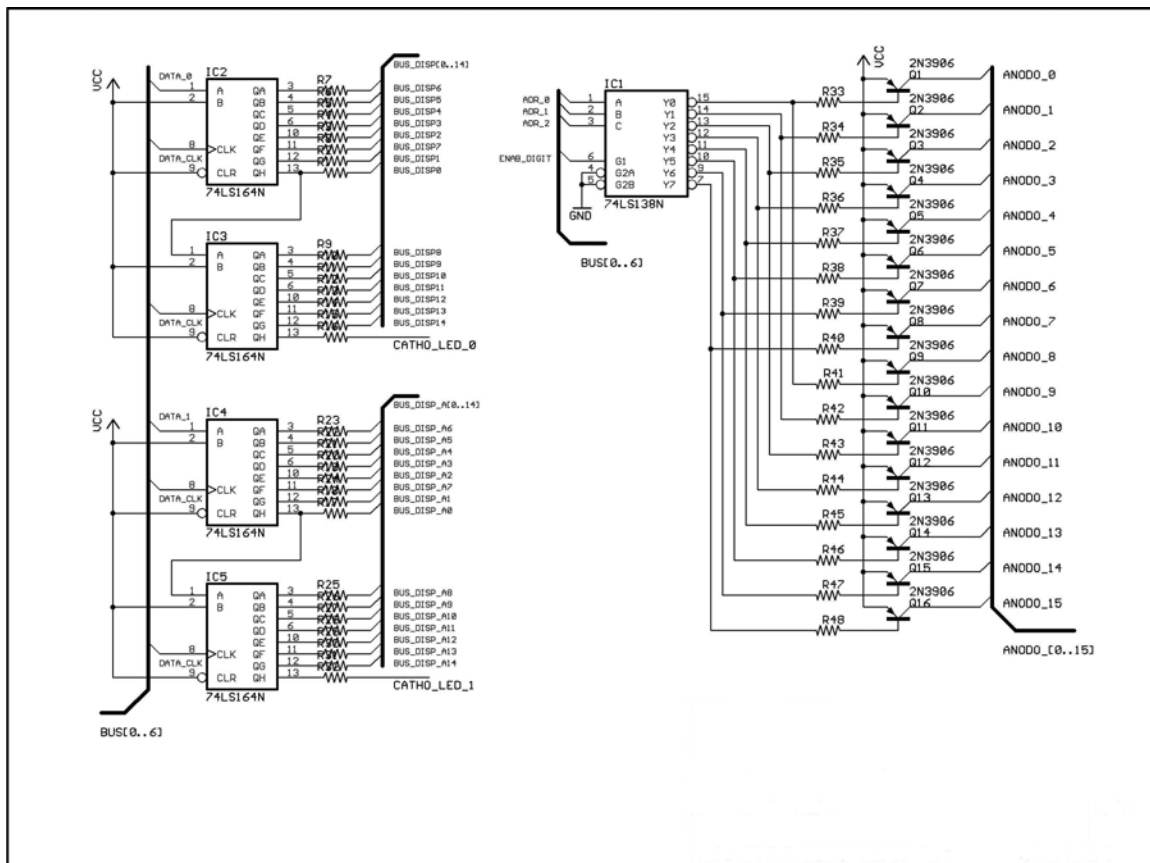


Figure 2. Segment Display SS

The display is controlled by seven I/O MCU lines:

- Two data I/Os (Data\_0, Data\_1)
- One data clock I/O (Data\_clk)
- Three address I/Os (Adr\_0, Adr\_1, Adr\_2)
- One digit-enabling I/O (enab\_digit)

Data\_0 I/O provides information for 0 to 7 digits, and Data\_1 provides information for 8 to 15 digits. Both are transferred at the same clock cycle of Data\_clk.

The PSoC device provides a 16-bit timer, which has 320 interrupts/second cycles.

## Software

Program performance is based on a periodic interrupt from Timer\_16. The interrupt service routine includes the display manager routine, *message\_to\_display()*.

Interrupts are generated 320 times per second. Each digit is refreshed 40 times per second (320 cycles/8 digits), which is a slightly above the blinking threshold.

The *message\_to\_display()* routine reads data from *message[]*, and has the text stored in ASCII code from 32 to 90. This is a natural limitation, due to using the 14-segment design display.

The decimal points are stored on a separate array called *dp\_pos[]*. This is for programming ease of *last\_pos* index, because digits are enabled simultaneously both between 0 to 7 and 8 to 15 in the position corresponding with *message[]*, and the character and decimal point are displayed together in one digit.

The *bar\_graph* global variable stores a value, between 0 and 15, which turns on the desired segments of the bar graph.

You can run the associated project in PSoC Designer to view the display routines.

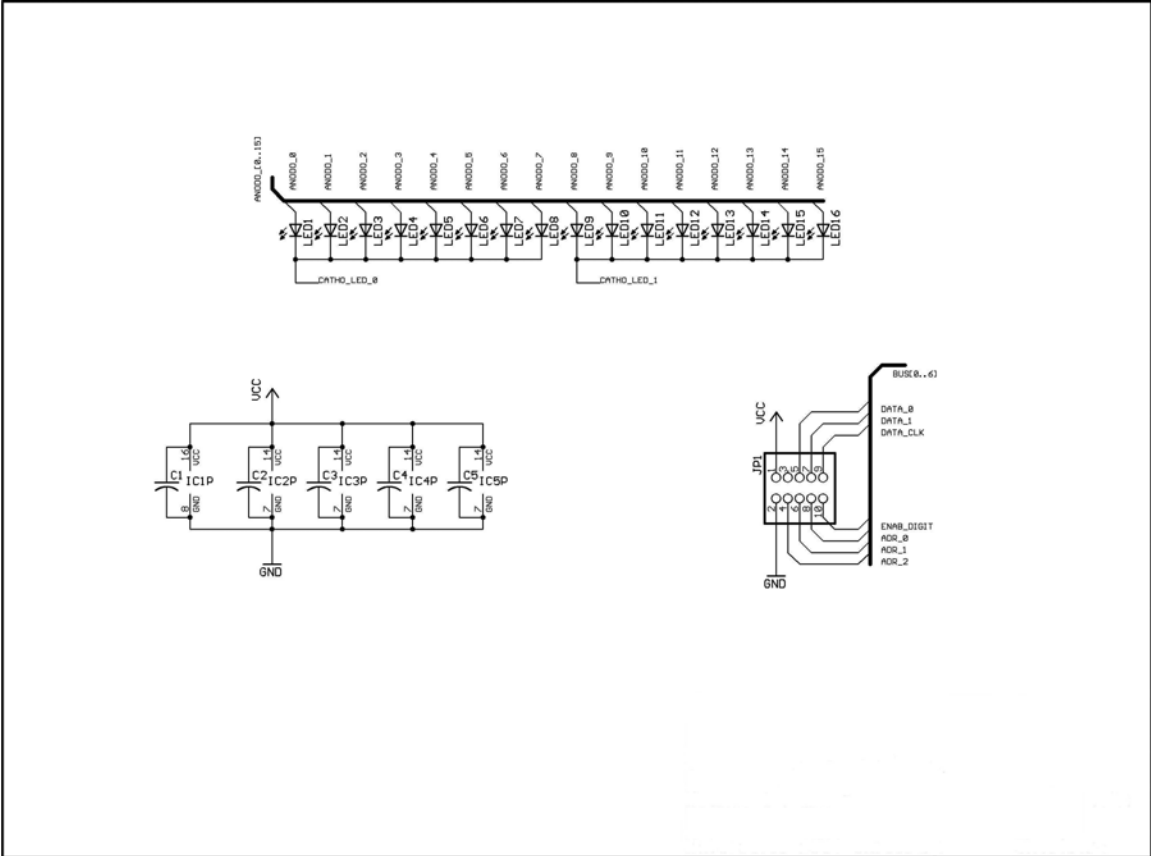


Figure 3. Segment Display SS

---

## About the Author

**Name:** Luis Espinal C.  
**Title:** Electronics Engineer  
**Background:** Consultant in electronics for embedded designs, with more than 10 years experience working with MCUs.  
**Contact:** [espinalc@hotmail.com](mailto:espinalc@hotmail.com)  
Use subject: **MCU** – your subject

---

Cypress Semiconductor  
2700 162<sup>nd</sup> Street SW, Building D  
Lynnwood, WA 98037  
Phone: 800.669.0557  
Fax: 425.787.4641

<http://www.cypress.com/> / <http://www.cypress.com/support/mysupport.cfm>

Copyright © 2005 Cypress Semiconductor Corp. All rights reserved.

PSoC™, Programmable System-on-Chip™, and PSoC Designer™ are PSoC-related trademarks of Cypress.

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

The information contained herein is subject to change without notice. Made in the U.S.A.