zilog **Application Note**

# How to Interface a Seven-Segment Display with the Z8 Encore!® MCU

**AN018902-0708**

## Abstract

This Application Note demonstrates how to multiplex and interface four 7-segment LEDs with the Z8 Encore!® MCU to form a display for a digital clock.

A GPIO port on the Z8 Encore!® MCU is used to drive the seven segments of the display. Four additional GPIO ports are used to select the individual digits. In this application, real time is displayed using the seven-segment display with a feature to adjust and set the time.

## Z8 Encore!® Flash Microcontrollers Overview

Zilog's Z8 Encore!® products are based on the eZ8 CPU and introduce Flash memory to Zilog's extensive line of 8-bit microcontrollers. Flash memory in-circuit programming capability allows for faster development time and program changes in the field. The high-performance register-to-register based architecture of the eZ8 core maintains backward compatibility with Zilog's popular Z8 MCU.

Z8 Encore!® MCUs combine a 20 MHz core with Flash memory, linear-register SRAM, and an extensive array of on-chip peripherals. These peripherals make the Z8 Encore!® MCU suitable for a variety of applications including motor control, security systems, home appliances, personal electronic devices, and sensors.

## Discussion

Light Emitting Diodes (LED), as the name implies, are diodes that emit light when forward-biased. LEDs are used to display information. Commonly-used LED displays are offered as seven-segment displays and dot-matrix displays. The seven-segment display is arranged in the form of the digit 8 and can display numbers and hexadecimal characters (0–F).

## Seven-Segment Display Construction

A seven-segment display is a group of eight LEDs arranged in segments. These segments are generally used to display the numbers 0 through 9 and the letters A through F. The display features a common-anode or common-cathode orientation as shown in Figure 1.

In this Application Note, a GPIO port on the Z8 Encore!® MCU transmits 8 bits of data. These bits are applied to the seven-segment display to cause it to illuminate the appropriate segments to display the proper number or character.

The seven-segment display used in this Application Note is of the common-cathode type. The common cathode is connected to a transistor, and the anode terminals of the LEDs are connected to Port E[0:7] through current limit resistors.

Segment intensity is dependent on the current flow and should not exceed the limit of the segment.

Copyright ©2008 by Zilog®, Inc. All rights reserved.
www.zilog.com

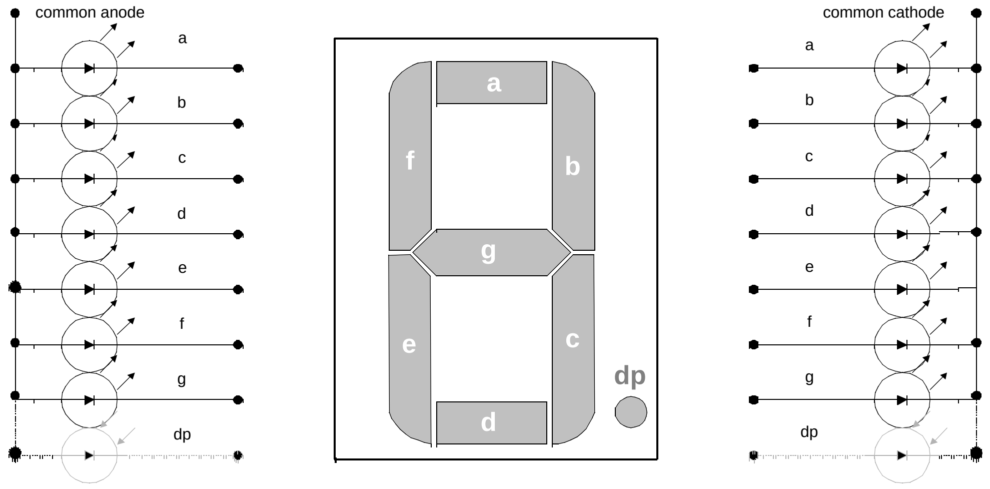

Figure 1. A Seven-Segment Display

The seven-segment unit illustrated in Figure 1 is used to display a single number (0-–) or a letter (A–F). For other applications such as digital timers, clocks, and other displays, several units can be aligned next to each other to form a larger panel.

## The Hardware Connection

For the seven-segment display, one GPIO pin is assigned per LED. Some examples of character display are shown in Figure 2.

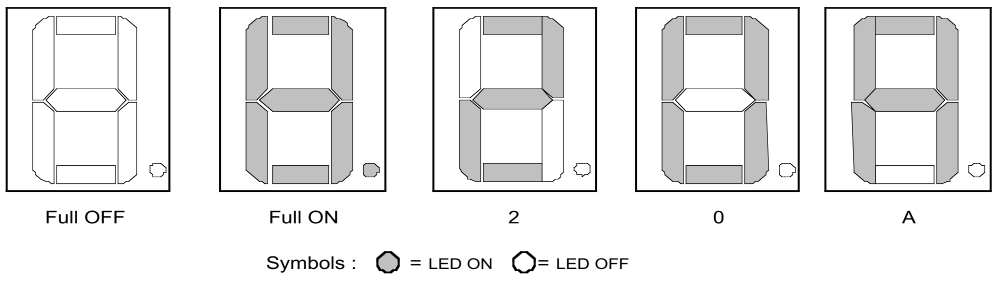

Figure 2. Displayed Characters

For a certain character, a combination of LED ON and LED OFF is generated to display the character for a short period of time. The pattern is loaded alternately to display other characters. For example, to display the number 2, LEDs **a**, **b**, **d**, **e**, and **g** are illuminated. Table 1 provides the display pattern for numbers and characters A through F.

**Table 1. Display Pattern for Characters 0-F**

| Characters | DP | G | F | E | D | C | B | A | Hexadecimal |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 3F |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 06 |
| 2 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 5B |
| 3 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 4F |
| 4 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 66 |
| 5 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 6D |
| 6 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 7D |
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 07 |
| 8 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 7F |
| 9 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 6F |
| A | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 77 |
| B | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 7C |
| C | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 39 |
| D | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 5E |
| E | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 79 |
| F | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 71 |

To control four 7-segment displays, multiplexing can reduce the number of GPIO pins required.

In this setup, the four multiplexed seven-segment displays are turned on one at a time to output the appropriate display. Because of the visual phenomenon known as *persistence of vision*, rapid switching of the seven-segment display can appear as if all four displays are turned on.

Figure 3 illustrates the procedure for displaying a four-digit value in the multiplexed seven-segment display. In this example, to be able to display the value 1258, number 8 is displayed first by inputting the pattern of number 8. Next, the NPN transistor connected to the corresponding seven-segment display is switched on in a short period of time while the other transistors remain turned off. Next, the display is turned off, and the following pattern is input to turn on the second transistor. This pattern repeats until the four digits alternately display. Rapid switching produces the illusion that all four 7-segment displays are turned on.

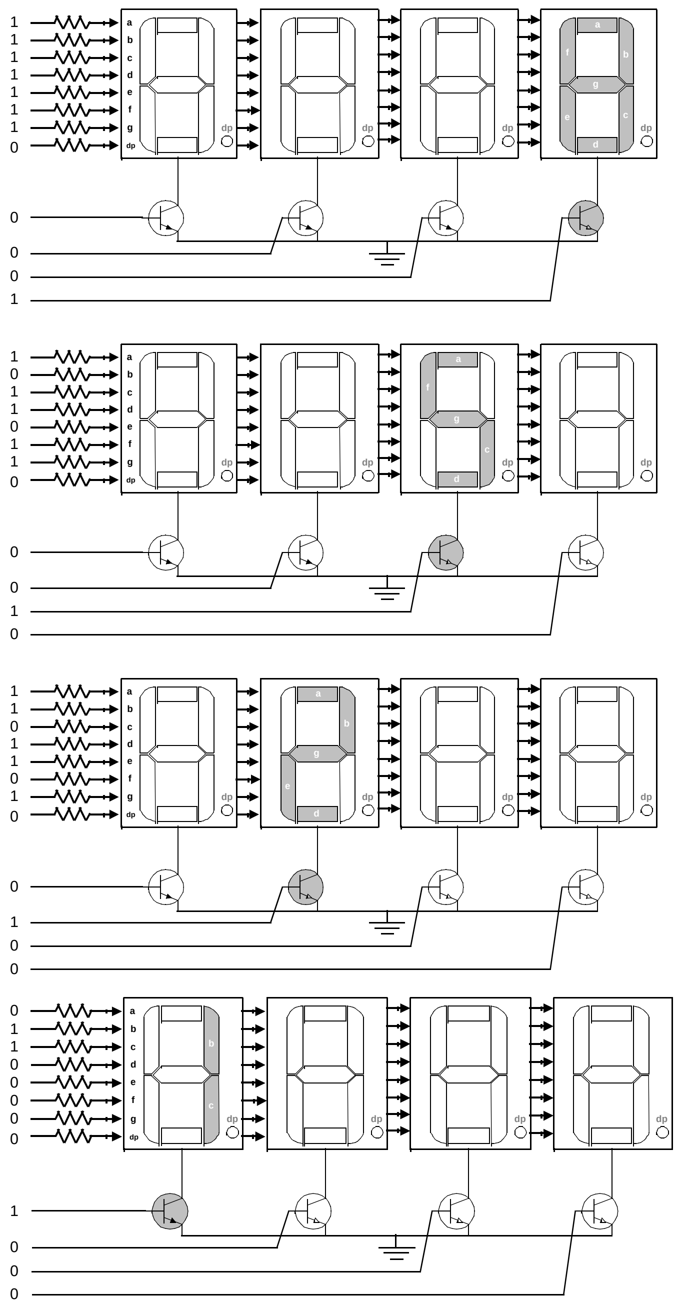

Figure 3. Displaying a Four-Digit Number in a Multiplexed Segment

## Interfacing Four 7-Segment Displays with the Z8 Encore!® MCU

The following sections describe the hardware and software components required to build the keypad routine.

### Hardware Architecture

Figure 4 illustrates a block diagram of the hardware architecture featuring the Z8 Encore!® MCU and four 7-segment displays.

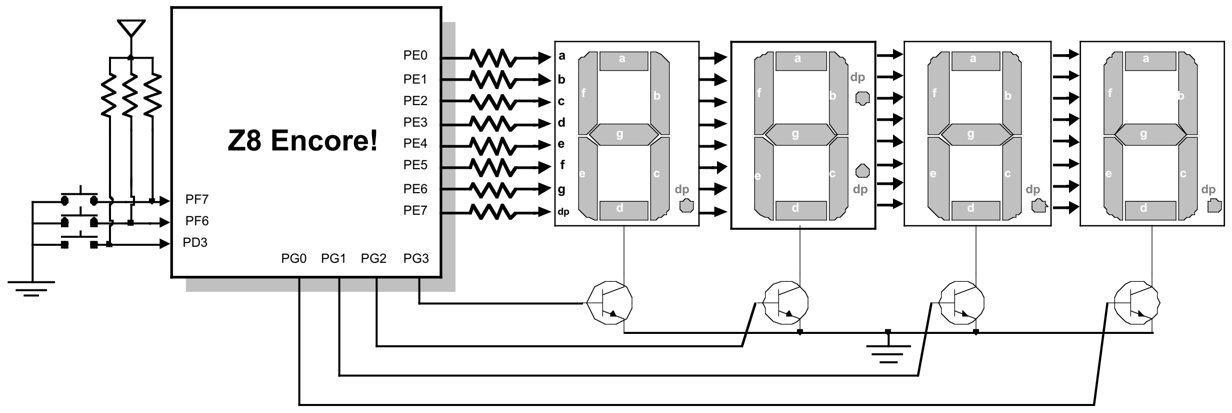

Figure 4. Block Diagram of Seven-Segment Displays with the Z8 Encore!® MCU

GPIO Port E [0:7] is used to drive a segment LED in the four display units in a multiplexed fashion. Four NPN transistors are used to turn on or off the display units individually. Three GPIOs—PD3, PF6, and PF7—are used to adjust the time as input pins and serve as a source of external interrupt. A pull-up resistor and a normally open pushbutton switch are connected at each of these pins as shown in Figure 4.

### Software Implementation

In this application, the following Z8 Encore!® GPIO pins are used —8 Port E pins for driving the segments of the display, 4 Port G pins to alternately turn on or off each seven-segment display, and Port D Bit 3 as an interrupt source to select a mode that allows an adjustment of hours or minutes. Port E [0:7] outputs either 1 or 0 depending on the defined pattern to display the required character. PF6 is used to decrement the current display of hour or minutes, while PF7 is used to increment the value.

Port E and Port G are initialized as outputs and Port D Bit 3 is configured as an interrupt source and set as a high priority.

Timer 0 is programmed in to operate in a continuous mode to generate a timer output of 100ms. The reload value is computed based on the following formula:

Continuous Mode Time-Out(s) = (Reload value x Prescale value) ÷ System Clock (Hz)

A Reload Value High and Low Byte equal to `38h` and `40h`, respectively, are used with a prescale value of `128` for the system running on an 18.432MHz system clock.

To produce accurate timing, a time-out of one second is required to update the display. To achieve this one-second timing, Timer 0 produces a time-out of 100ms duration, and a timer interrupt counter is used to count from 0 to 10 before it updates the display. After every change in value, the ones and tens values of the minutes and hours are extracted to get the data to be displayed. The extracted value is transferred to a register to change or update the data during every scan. After each scan, the values are updated and displayed by getting the pattern in the provided array.

Initially, the display is set to zero and waits for a falling edge interrupt in PD3. Pressing SW1 (PD3) changes the mode of the real-time clock (RTC). Three modes allow the user to configure the RTC and adjust for the correct time. Mode 0 displays the current time. Mode 1 allows the user to either increment or decrement the value of minutes displayed. The value of the hour can be modified by entering Mode 2. Pressing SW1 (PD3) three times resets the mode and returns to normal operation.

To adjust the value at any time, Timer 1 is used to monitor whether switches SW2 and SW3 (PF6 & PF7) are pressed. Timer 1 also operates in continuous mode and has a time-out of 100ms. The formula for Timer 0 above is used. Port F bits 6 and 7 are initialized as inputs, and these bits are also used as interrupt sources. The MCU receives data from Port F inputs every two interrupts and then processes the data. Pressing the switch connected to PF6 decrements the displayed value while SW3 (PF7) increments the value.

Functions processed by the two timers work independently in the background. The main program continually refreshes and updates the display depending on the data transmitted by the two timers.

The **display()** routine called in the **main** program refreshes the display by performing the following tasks:

- Increments a scan counter that provides the timing required to shift the digits to be displayed, and resets the counter when it reaches the terminal count
- Calls the **get_data** API (extracts the digit from the hour and minute variables)
- Calls the **transfer_data** API (transfers the extracted digits to a register)
- Calls the **update()** API (gets the pattern from an array and outputs the data)

The **count()** routine is called every one second and is generated by a Timer 0 interrupt. It updates the seconds, minutes, and hours values.

The **adjust_time()** routine allows the user to adjust the current time by pressing control buttons.

**Note:** *The parameters and returns of each of the above functions is void.*

## Testing

The following sections describe the equipment and procedure required to test the seven-segment display interface with the Z8 Encore!® MCU.

### Equipment Used

The following equipment was used for testing:

- Z8 Encore!® Development Kit (Z8ENCORE000ZCO)
- ZDS II IDE–Z8 Encore!®
- Four SG511 7-segment displays
- NPN transistors

### Test Setup

To build the application, observe the following instructions.

1. Set up the connection between the Z8 Encore!® Development Board and the seven-segment display as shown in Figure 4.
2. Run the ZDS II IDE for Z8 Encore!®. Apply power to the Z8 Encore!® Development Board and download the source code.
3. Run the program.
4. The display is initially set to zero. Press SW1 (PD3) to adjust the display. To adjust to the correct minutes value, press switches PF6 and PF7.

5. Press SW1 a second time to enter the hours mode. To adjust to the correct hours value, press switches PF6 and PF7.
6. Press SW1 a third time to resume normal operation.

**Note:** *The seven-segment display used in this application is a common cathode. If you are using a common anode, please refer to the source code for some modifications in the program.*

### Test Results

The results were as expected, as accurate timing was achieved in displaying the time on the four 7-segment displays. The time was entered and adjusted correctly using the provided control switches.

## Summary

This Application Note explains how to control a multiplexed seven-segment display along with a feature to set the time of the clock using the GPIO ports, the built-in timers, and the peripherals of the Z8 Encore!® MCU with minimal external circuitry.

## Appendix A—Reference

Related documents that may be useful are listed in Table 2.

**Table 2. Related Document References**

| Topic | Document |
|---|---|
| eZ8 CPU | eZ8 CPU User Manual (UM0128) |
| Z8 Encore!® CPU | Z8 Encore!® Microcontrollers with Flash Memory and 10-Bit A/D Converter Product Specification (PS0176) |
| ELS-511 Display | S511GWA.pdf from www.everlight.com |

## Appendix B—Flowcharts

Figure 5 illustrates the flow of the main routine that interfaces a seven-segment display to the Z8 Encore!® MCU's GPIO ports, which are initialized to output the seven-segment display. Timer 0 is configured in continuous mode to set the timing of the clock. Port D is also initialized to detect a falling edge interrupt to set the mode of the real-time clock. Port F inputs are used to enter the correct time.Timer 1 also operates in continuous mode to monitor whether the switches are pressed to adjust the time. The display is updated at one second intervals.

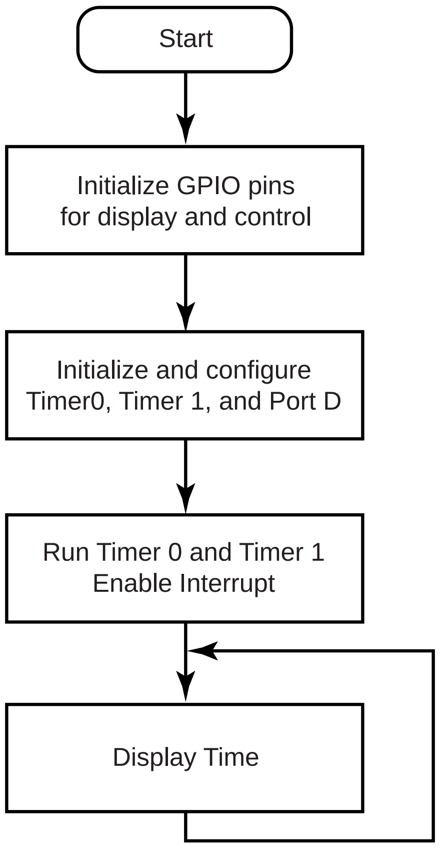
Start
Initialize GPIO pins for display and control
Initialize and configure Timer0, Timer 1, and Port D
Run Timer 0 and Timer 1 Enable Interrupt
Display Time

**Figure 5. Main Routine Flow**

Figure 6 illustrates the flow of the Port D interrupt. When a falling edge is detected, the mode is incremented. On the third interrupt, the mode resets to zero.

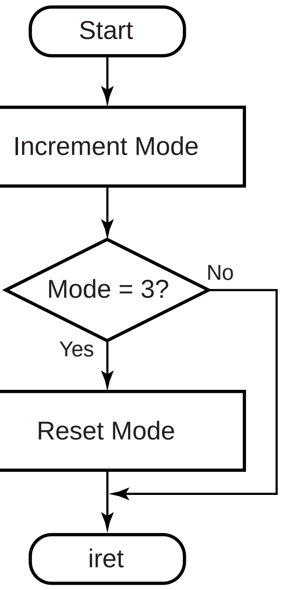

**Figure 6. Port D3 Interrupt Flow**

Figure 7 illustrates the flow of the Timer 0 routine. Timer 0 operates in continuous mode to produce accurate clock timing. A counter is incremented every 100 ms, and counts 10 occurrences to result in a one-second time-out. Next, it increments the seconds counter. Every 5 counts, the **blink_colon** flag is toggled to produced a blinking display. When the seconds counter reaches 60 counts, the minute counter is incremented. On the 60th count, the hour counter is incremented until it reaches 24 counts and the cycle continues.

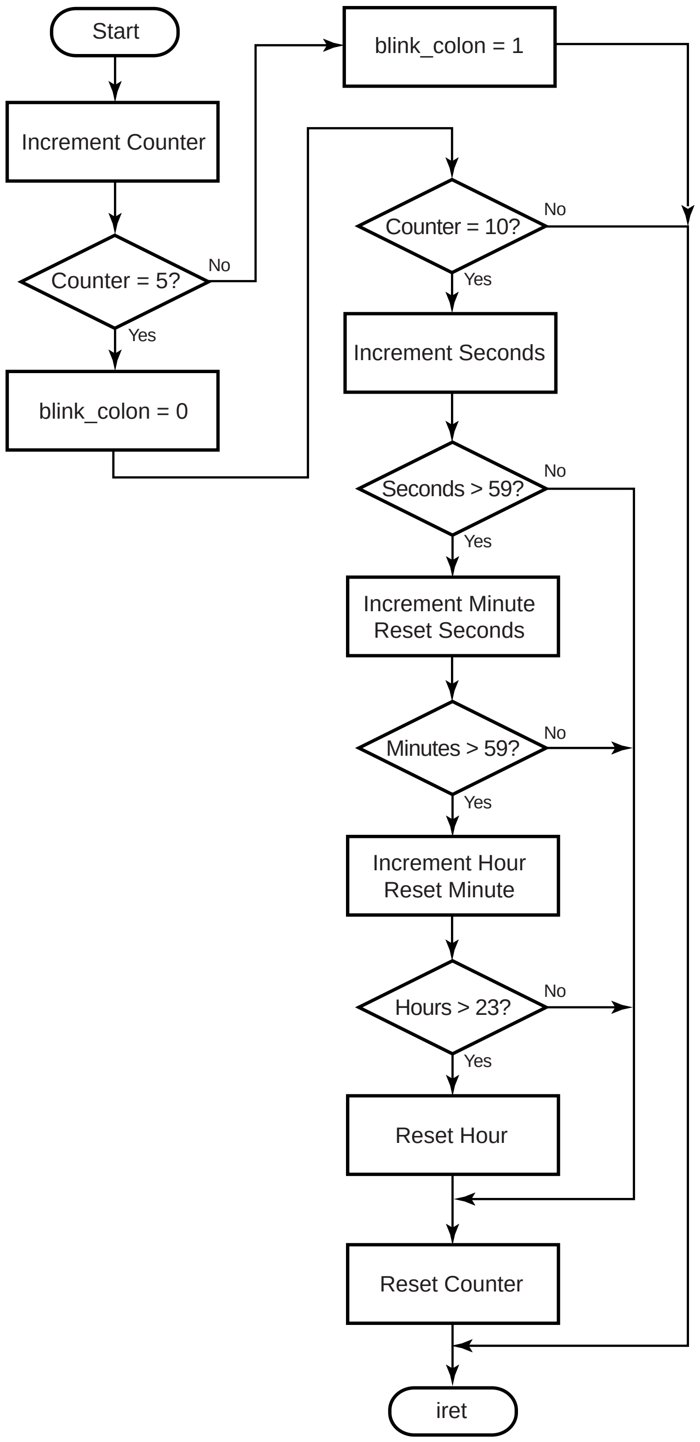

**Figure 7. Timer 0 Flow**

Figure 8 illustrates the flow of the Timer 1 routine. Timer 1 operates in continuous mode with a time-out value of 100 ms. For every two interrupts, it checks the mode of the real-time clock. The displayed time can be either incremented or decremented using the switches.

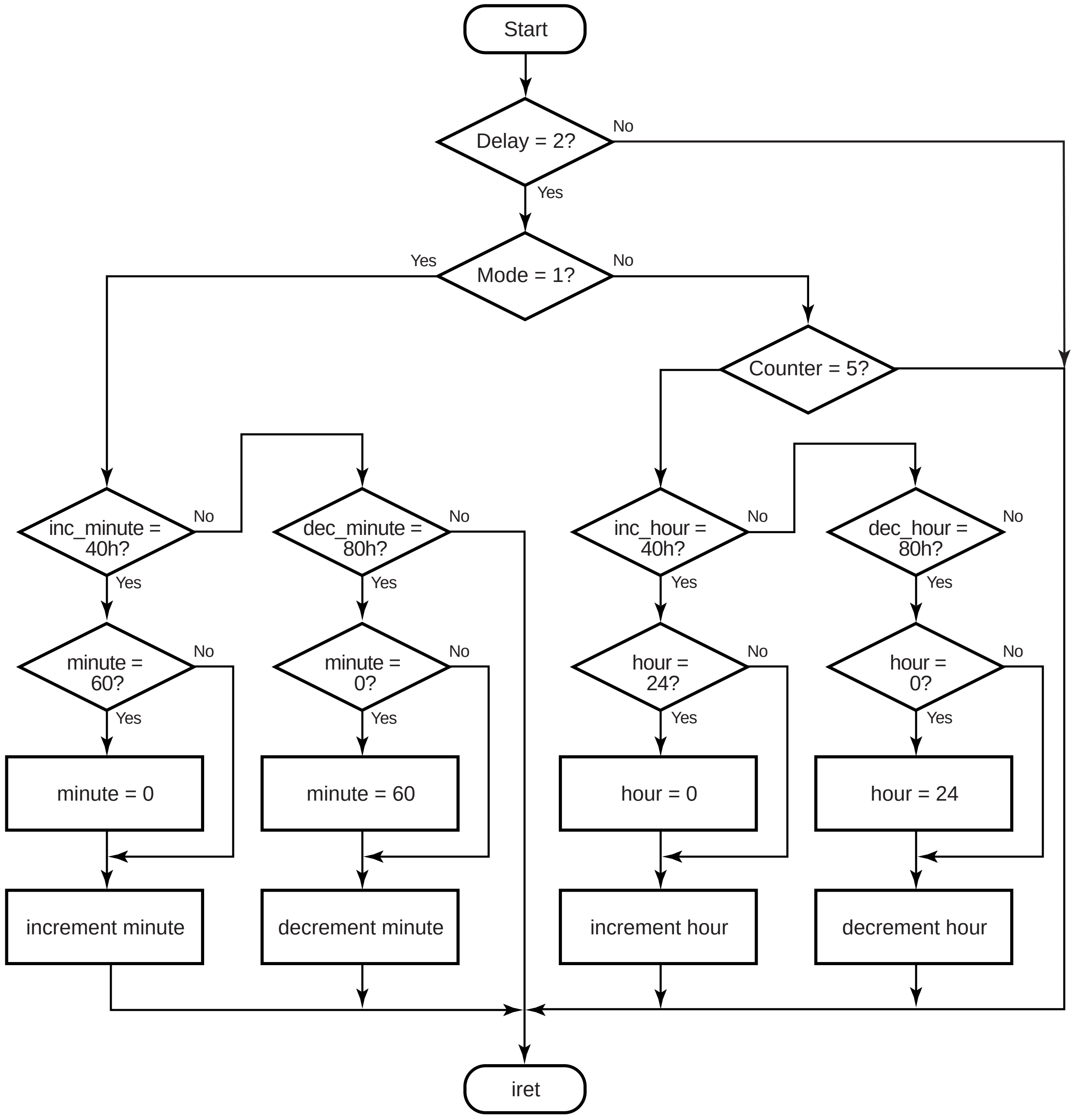

Figure 8. Timer 1 Flow

**Warning:** DO NOT USE IN LIFE SUPPORT

## LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

## As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

## Document Disclaimer

©2008 by Zilog, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

Z8, Z8 Encore!, and Z8 Encore! XP are registered trademarks of Zilog, Inc. All other product or service names are the property of their respective owners.