Application Note

Generating DTMF Tones Using Z8 Encore!® XP F64xx Series MCUs

AN024803-0911

Abstract

This application note describes how Zilog's Z8 Encore! XP F64xx Series MCUs can be used as Dual-Tone Multi-Frequency (DTMF) signal encoders to generate DTMF tones. Z8 Encore! XP microcontrollers feature built-in timers and direct memory access (DMA) with three independent channels (DMA0, DMA1 and DMA2). The timers offer Pulse-Width Modulation (PWM) capability and the PWM Register values are loaded using the DMA. These Z8 Encore! XP MCU features are used to generate DTMF signals with less processor overhead.

Note: The source code associated with this application note, *Z8 Encore! Applications Code Library*, is available for download from the Application Sample Libraries page on the Zilog website.

Z8 Encore! XP Microcontrollers

Z8 Encore! XP products are based on the eZ8™ CPU and introduce Flash memory to Zilog's extensive line of 8-bit microcontrollers. The Flash memory in-circuit programming capability of the Z8 Encore! XP family of MCUs allows 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! XP MCUs combine a 20MHz core with Flash memory, linear-register static random access memory (SRAM), and an extensive array of on-chip peripherals. These peripherals make the Z8 Encore! XP MCU suitable for a variety of applications including motor control, security systems, home appliances, personal electronic devices, and sensors.

Discussion

DTMF is an international signaling standard for touch-tone telephones. A DTMF encoder converts standard telephone digits into the sum of a sinusoid frequency pair that corresponds to a value in a frequency table. The DTMF decoder takes a digital signal as input and converts it to a single digit.

The DTMF standard developed by Bell Laboratories is used in touch-tone telephones and voicemail systems. Each telephone digit corresponds to a combination of high and low tones that is transmitted simultaneously. Four frequencies in this high and low grouping provide 16 possible combinations: 12 digits are used in most telephones; the additional 4 digits are used in military applications. A DTMF detector must conform to the International Telecommunication Union (ITU) standard, which was formerly known as the Inter-

national Telegraph and Telephone Consultative Committee (CCITT) recommendations. Table 1 lists the frequencies corresponding to each digit.

Table 1. DTMF Frequency Table

		High-Group Frequencies (Hz)			
		1209	1336	1477	1633
Low-Group Frequencies (Hz)	697	1	2	3	A
	770	4	5	6	B
	852	7	8	9	C
	941	* (E)	0	# (F)	D

Theory of Operation

A DTMF tone is a combination of two sinusoids of differing frequencies. The pulse-width modulation (PWM) method is the easiest and best way to generate a sinusoidal (sine) wave from a microcontroller.

Sine Wave Generation Using PWM

In PWM method, the duty cycle determines the amplitude of the sine wave and the number of samples determines the frequency of the sine wave. A sine wave is generated when the PWM output is passed through a low pass filter (LPF). To generate a DTMF tone, two timers in PWM mode are used for each frequency and their outputs are together passed through the LPF. Figure 1 displays the relation between signal frequency (1/T), sampling period (Ts), and number of samples (N).

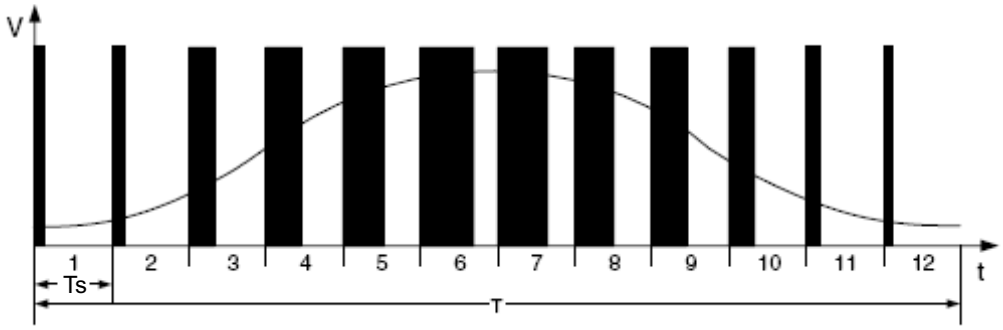

Figure 1. Sine Wave Generation Using PWM

The following formula is used to generate the chart shown in Figure 1:

$$(\text{Number of samples (N)}) \times (\text{Sampling Period (Ts)}) = \text{Signal Period (T)}$$

Computation of the formula above yields the results shown in Equation 1.

Equation 1.

$$N/Fs = 1/F$$

Equation 2.

$$Fs = NF$$

In Equation 2, Fs is the sampling frequency and F is the signal frequency.

The PWM Duty Cycle registers are loaded using either of the following methods:

- The Timer Interrupt method
- The Direct Memory Access method

Timer Interrupt

With the Timer Interrupt method, the PWM registers are loaded at every timer interrupt (or as determined by the number of samples). The timer interrupts occur at the rate of the sampling frequency, thereby causing the servicing of these interrupts to keep the processor inordinately busy.

Direct Memory Access

With the Direct Memory Access (DMA) method, the DMA block is used to load the PWM Duty Cycle Register. The use of the DMA reduces processor overhead; the user interface provided with this application offers a feature to calculate this processor overhead (for details, see the Testing section on page 7). This user interface seamlessly integrates with the DTMF generation application because the DMA allows ample time to process other applications.

Lookup Table Formation

A lookup table is used to load the PWM Timer Reload and Duty Cycle registers. The PWM Duty Cycle registers are loaded at the rate of the sampling frequency. For DTMF generation, sinusoids of different frequencies are required. A single lookup table is used for this purpose, and its values are scaled to achieve an appropriate output frequency.

Calculating Timer Reload Value

The following example describes a method of obtaining PWM timer reload values for different frequencies. In this example, the base frequency is 697 Hz, the system clock is 18.432 MHz and the samples per cycle is 104 s.

The time period is calculated as:

$$Tbase = 1/697 = 1.43472 \text{ ms}$$

To compute the signal period, we reiterate Equation 1:

$$N \times Ts = T$$

The resulting signal period is computed as:

$$\begin{aligned} Ts &= T/N \\ &= 1.43472/104 \\ &= 13.7953\ \mu s \end{aligned}$$

In the computation above, Ts is the time period of the PWM. Therefore, the timer reload register value can be calculated as:

$$\begin{aligned} Reload &= (Ts \times System\ Clock/Prescalar) \\ &= 13.7953 \times 18.432/1 \\ &= 254.27 \\ &= 254 \end{aligned}$$

$$Reload = 0 \times FE \text{ (hexadecimal representation)}$$

The above result corresponds to a sampling frequency of 72.488kHz (1/Ts).

The time periods for all other frequencies are expressed as multiples of the time period of the base frequency. These multiplier values are scaled by 256 to avoid floating point operations. When using these multiplier values, the timer reload values for other frequencies can be calculated.

Consider a 770Hz signal as an example:

$$Time\ period = 1/770 = 1.2987\ ms$$

In terms of the base signal, the multiplier value can be calculated as:

$$\begin{aligned} Multiplier\ value &= 1.2987/Tbase \\ &= 1.2987/1.43472 \\ &= 0.90519 \end{aligned}$$

This multiplier value is then scaled by 256. Therefore, the multiplier value becomes 232, as shown in the following equation.

$$\begin{aligned} Multiplier\ value &= 0.90519 \times 256 \\ &= 231.728 \\ &= 232 \end{aligned}$$

The timer reload value for this signal, expressed as a multiple of the base frequency, is computed as:

$$Timer\ reload\ value = (232 \times 254)/256 = 230$$

Therefore, the frequency is computed as:

$$\begin{aligned} F &= \text{System clock} / (\text{Number of samples} \times \text{Reload}) \\ &= 18432000 / (104 \times 23) \\ &= 770.568 \text{ Hz} \end{aligned}$$

$$\begin{aligned} \text{Percentage Error} &= (770.568 - 770) / 770 \times 100 \\ &= 0.073\% \end{aligned}$$

Calculating PWM Values

The sine value corresponding to the samples is computed as:

$$\sin(n) = \sin(2 \times \pi \times n / N)$$

To use a value of sin(10) as an example the computation becomes:

$$\sin(2 \times \pi \times n / N) = \sin(2 \times \pi \times 10 / 104) = 0.56806$$

Because negative PWM values are not acceptable, the integer 1 is added to all computed sine values to make these sine values positive, as shown in the following equation.

$$\sin(2 \times \pi \times n / N) = 0.56806 + 1 = 1.56806$$

To use a minimum PWM value of 15 as an example, the corresponding PWM value becomes:

$$\begin{aligned} \text{PWM value} &= 1.56806(\text{Timer reload value} - \text{Minimum PWM value}) / 2 \\ &+ \text{Minimum PWM value} \\ &= 1.56806(254 - 15) / 2 + 15 \\ &= 202 \end{aligned}$$

PWM values can also be scaled according to the multiplier before loading to the PWM registers. Referencing the earlier example of 770Hz, yields a multiplier value of 232. Therefore, the PWM value corresponding to 770Hz is computed as:

$$\begin{aligned} \text{PWM value} &= 202 \times 232 / 256 \\ &= 183 \end{aligned}$$

Note: For detailed calculations, refer to the spreadsheet provided in the *Z8 Encore! Applications Code Library* on the Zilog website. This spreadsheet automatically calculates the necessary system clock and timer prescale values for a given number of samples.

Developing the DTMF Encoder

Building a DTMF encoder first requires a filter to pass all low-frequency signals but attenuate (reduce the amplitude of) those signals with frequencies higher than the *cutoff frequency*.

Low Pass Filter Design

The PWM outputs are passed through a Low Pass Filter to demodulate and obtain a DTMF signal; a simple resistor-capacitor (RC) filter is used for this purpose. Assuming a maximum frequency of 1633 Hz for a standard DTMF signaling application, the cut-off frequency (Fo) of the resistor-capacitor (RC) filter can be calculated as:

$$Fo = \frac{1}{2 * \pi * RC}$$

As an example, if this cut-off frequency is 3000 Hz, and if capacitance (C) = 0.2 µF, then the following equation can be calculated to compute resistance (R):

$$R = \frac{1}{2 * \pi * 3000 * 200 * 10^{-9}}$$

$$= 265.25$$

$$= 270\,\Omega$$

Figure 2 shows a schematic diagram of the low pass filter.

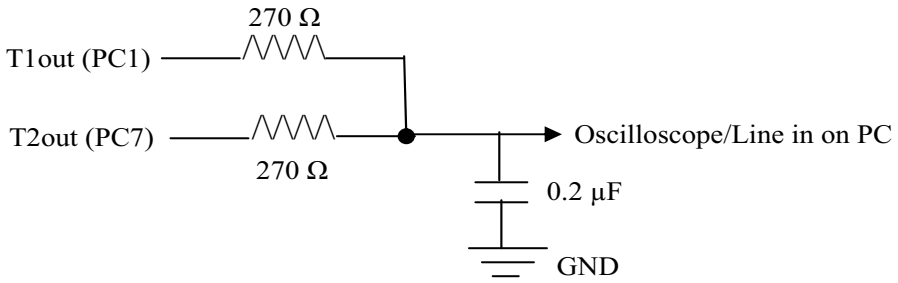

Figure 2. Low Pass Filter

Software Implementation

The software required to generate DTMF signals makes use of the Z8 Encore! XP MCU's timer and DMA functions. The input to the application is a telephone keypad digit for which a DTMF signal is required. The index to the specific DTMF frequencies is obtained from the DTMF frequency table defined in the `sine_table.asm` file contained in the *Z8 Encore! Applications Code Library*. Each index refers to the multiplication factors for that frequency. The first value in the `_SineTable` is the timer compare value of the base frequency. The remaining values represent the first half of the sine wave. The second half is generated by the software as a mirror image of the first half. During operation, these val-

ues are transferred to the Timer PWM Low Register via the DMA. The timer reload value is calculated by multiplying the base timer compare value with the multiplication factor. Only the most significant byte (MSB) is considered. For more information about the software flow, see Appendix B. Flowcharts on page 11.

Loading the PWM Register Using the DMA

The DMA block of the Z8 Encore! XP MCU is used to load the PWM registers. The F64xx Series' DMA controller consists of three independent DMA channels (DMA0, DMA1, and DMA_ADC). DMA0 and DMA1 transfer data between the on-chip peripherals and the register file. The third channel, DMA_ADC, controls Analog-to-Digital Converter (ADC) operation and transfers ADC output data in SINGLE-SHOT Mode to the register file.

In this application, Timer2 is associated with DMA0 to generate low tones and Timer1 is tied to DMA1 to generate high tones. The source (starting address) for the DMAs is the starting address of the corresponding sine table. The sine tables thus formed are scaled and mirrored. The destination (peripheral) address of DMA0 and DMA1 are the PWM Low Byte registers of Timer2 and Timer1, respectively.

Testing

For purposes of demonstration, the source code contained in the Application Sample Library consists of a console interface.

Setup

Figure 3 displays the setup for the generation of DTMF tones using a Z8 Encore! XP F64xx Series MCU.

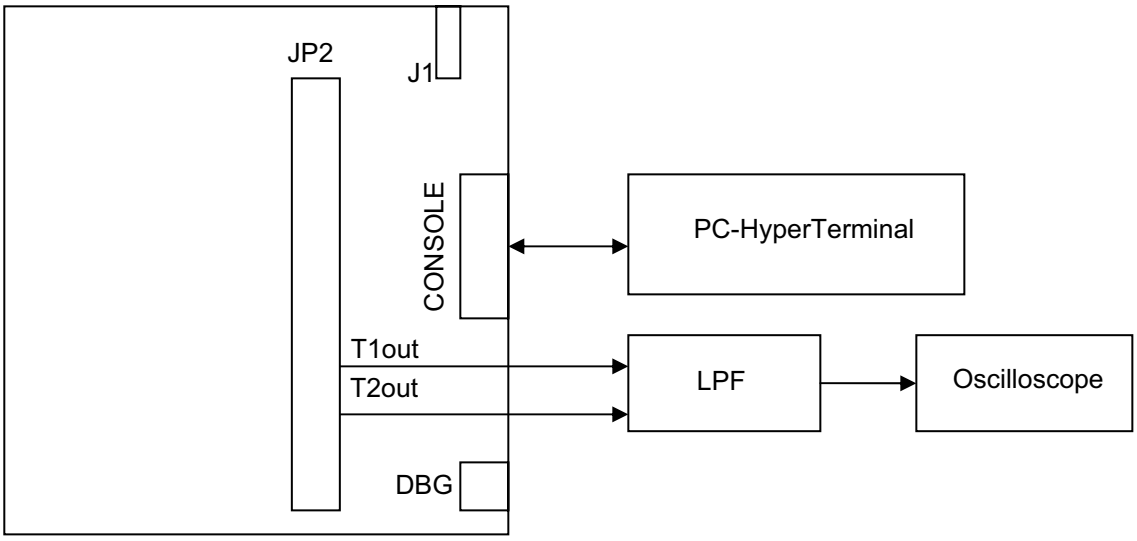

Figure 3. Setup for the Generation of DTMF Tones Using the Z8 Encore! XP F64xx Series Development Board

Equipment Used

The following equipment is used for the setup:

- Z8 Encore! XP F64xx Series Development Kit featuring the F64xx Series Development Board
- ZDSII IDE – Z8 Encore!
- Oscilloscope
- A PC equipped with HyperTerminal configured to the following settings:
  - 57600 baud rate
  - 8 data bits
  - No parity
  - One stop bit
  - No flow control

Procedure to Test the DTMF Application

Observe the following procedure to test the DTMF application.

1. Connect the external LPF to the timer outputs as shown in Figure 3.
2. Connect the console port output to the COM port configured with the HyperTerminal.
3. Install the *Z8 Encore! Applications Code Library,* available from the Application Sample Libraries page on the Zilog website.
4. Launch ZDSII and open the `DTMF.zdsproj` project file, which is located in the source folder.
5. Build and download the code to the F64xx Series Development Board.
6. Execute the code. The user interface routine prints a list of menu items in the HyperTerminal window.
7. Select the appropriate menu item and follow the instructions to generate the DTMF signals. Relevant APIs are called automatically, and the DTMF signal can be viewed on the oscilloscope. For demonstration purposes, the DTMF signal can also be fed to the *Line in* port on a PC and listened for through the speaker output.

Result

The DTMF signals, as well as the individual sine wave outputs, are captured on the oscilloscope. Table 2 lists the percentage error in the output frequencies.

Table 2. Percentage of Error in the Output Frequencies

Observed Low-Tone Frequency (Hz)	Expected Low-Tone Frequency (Hz)	Low-Tone Frequency Difference (Hz)	% Error in Low-Tone Frequency	Observed High-Tone Frequency (Hz)	Expected High-Tone Frequency (Hz)	High-Tone Frequency Difference (Hz)	% Error in High-Tone Frequency
942	941	1	0.11	1339	1336	3	0.22
702	697	5	0.72	1208	1209	–1	–0.08
769	770	–1	–0.13	1488	1477	11	0.74
851	852	–1	–0.12	1642	1633	9	0.55

Summary

This application note provides a quick solution for generating DTMF tones using the DMA and timer feature sets of Zilog's Z8 Encore! XP F64xx Series MCUs. The PWM mode of the timers, combined with the DMA, offers flexibility toward integrating applications that use tone generation as a subset. This method greatly reduces processor overhead compared to other microcontrollers that employ interrupts.

References

The following documents describe functional specifications and/or otherwise support this application note. Each is available for download from the Zilog website.

- eZ8 CPU Core User Manual (UM0128)
- Z8 Encore! XP F64xx Series Product Specification (PS0199)

Appendix A. Glossary

Table 3 lists the abbreviations and definitions used in this Application Note.

Table 3. List of Abbreviations

Abbreviation	Definition
API	Application Programming Interface
DBG	Debug pin on Development Board
DMA	Direct Memory Access
DTMF	Dual-Tone Multi-Frequency
LPF	Low Pass Filter
MCU	Microcontroller Unit
PWM	Pulse-Width Modulation
SRAM	Static Random Access Memory

Appendix B. Flowcharts

Figures 4 and 5 show the sequences for loading the sine table and PWM registers using the DMA. These sequences for loading the sine table in Figure 4 are the same as those for loading the low and high tones table.

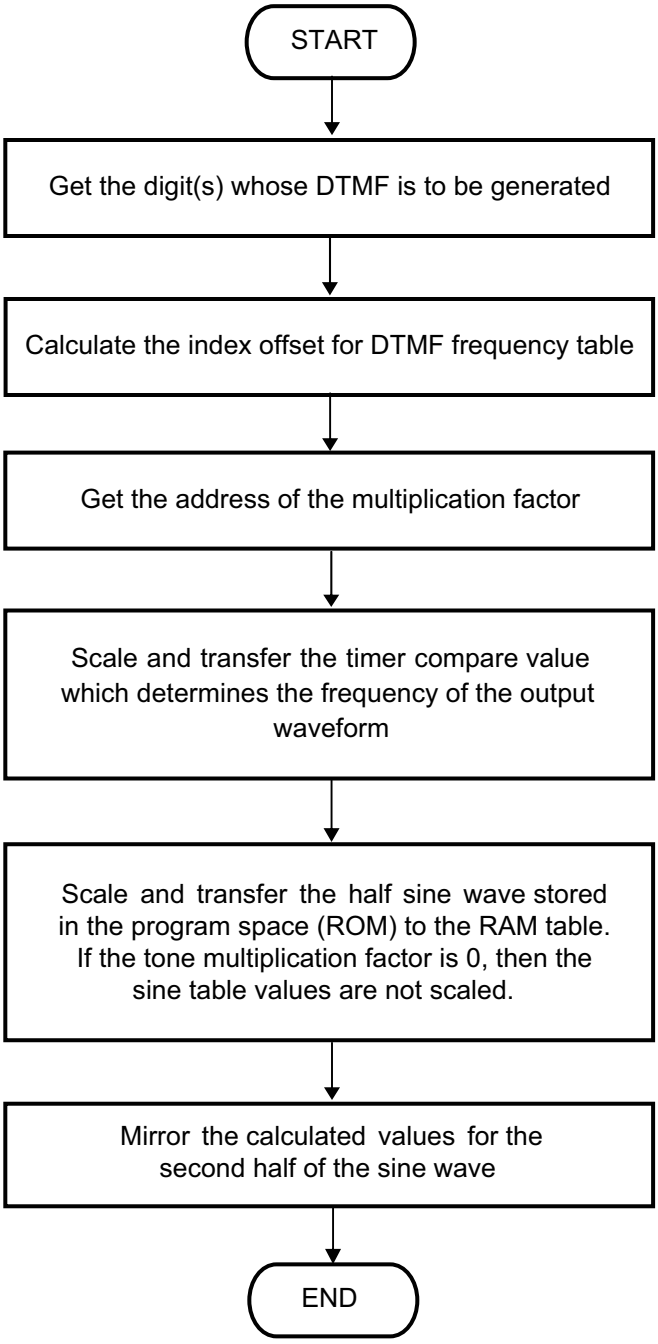

Figure 4. Flowchart for Loading the Sine Table

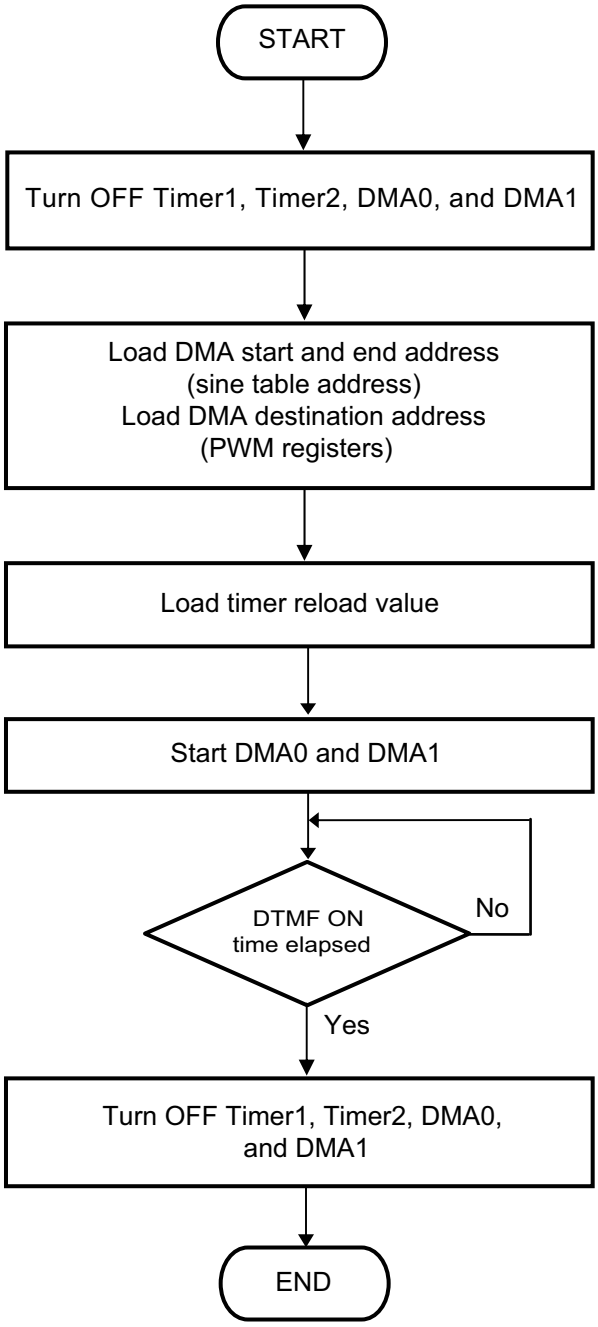

Figure 5. Flowchart for Loading the PWM Registers Using DMA

Customer Support

To share comments, get your technical questions answered, or report issues you may be experiencing with our products, please visit Zilog's Technical Support page at http://support.zilog.com.

To learn more about this product, find additional documentation, or to discover other facets about Zilog product offerings, please visit the Zilog Knowledge Base at http://zilog.com/kb or consider participating in the Zilog Forum at http://zilog.com/forum.

This publication is subject to replacement by a later edition. To determine whether a later edition exists, please visit the Zilog website at http://www.zilog.com.

Warning: DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

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

 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.