Application Note

Using a PWM as a Digital-To-Analog Converter

AN035701-0915

Abstract

Several microcontroller (MCU) applications utilize a Digital-to-Analog Converter (DAC). Though many of these applications require a complex DAC, some may only need a simple single-channel DAC. Because most MCUs do not have a built-in DAC, and an external DAC is often expensive, an alternative method is used for implementing a single-channel DAC. This method is the generation of Pulse Width Modulation (PWM), which is later integrated by a passive RC low pass filter. The advantage of this method is that the DAC resolution can be configured by software. However, the basic principle of this type of DAC is the generation of PWM. This application note describes the use of a Zilog ZNEO MCU PWM peripheral as a digital-to-analog converter.

There are multiple ways of generating a PWM. The simplest method involves generating the PWM directly from the dedicated PWM timer on the MCU. Although a built-in PWM timer is not a feature of every MCU, the Zilog ZNEO includes PWM timers. In many circuit applications, it is useful to have a technique to generate an arbitrary voltage or analog signal. Using a Zilog MCU to generate a programmable level or have the ability to adjust the level in response to external events increases the range of possibilities for a system designer. This application note demonstrates several methods to utilize Zilog MCU peripherals to generate a PWM signal, and therefore, a DAC. This document also discusses basic concepts of PWMs and how to filter their outputs to use a PWM as a low-cost DAC.

Note: The source code file associated with this application note, AN0357-SC01, is available free for download from the Zilog website. This source code has been tested using Zilog's ZDS II ZNEO 5.2.0.

ZNEO Flash Microcontroller Features

The ZNEO family of products includes the following features:

- 20 MHz ZNEO CPU
- 128 KB internal Flash memory with 16-bit access and In-Circuit Programming
- 4 KB internal RAM with 16-bit access
- External interface allows seamless connection to external data memory and peripheral with:
  - Six chip selects with programmable Wait states
  - 24-bit address bus supports 16 MB
  - Selectable 8-bit or 16-bit data bus widths
  - Programmable chip select signal polarity
  - ISA-compatible mode
- 12-channel, 10-bit Analog-to-Digital Converter (ADC)
- Operational Amplifier
- Analog Comparator
- 4-channel Direct Memory Access (DMA) controller supports internal or external DMA requests
- Two full-duplex 9-bit Universal Asynchronous Receiver/Transmitter (UARTs) with support for Local Interconnect Network (LIN) and Infrared Data Association (IrDA)
- Internal Precision Oscillator (IPO)
- Inter-Integrated Circuit (I2C) master/slave controller
- Enhanced Serial Peripheral Interface (ESPI)
- 12-bit Pulse Width Modulation (PWM) module with three complementary pairs or six independent PWM outputs with deadband generation and fault trip input
- Three standard 16-bit timers with Capture, Compare, and PWM capability
- Watchdog Timer (WDT) with internal RC oscillator
- 76 General-Purpose Input/Output (GPIO) pins
- 24 interrupts with programmable priority
- On-Chip Debugger (OCD)
- Voltage Brownout (VBO) protection and Power-on Reset (POR)
- 2.7 V to 3.6 V operating voltage with 5 V-tolerant inputs
- 0°C to +70°C standard temperature and -40°C to +105°C extended temperature operating ranges

Block Diagram

Figure 1 shows the architecture of the ZNEO Z16F Series of MCUs.

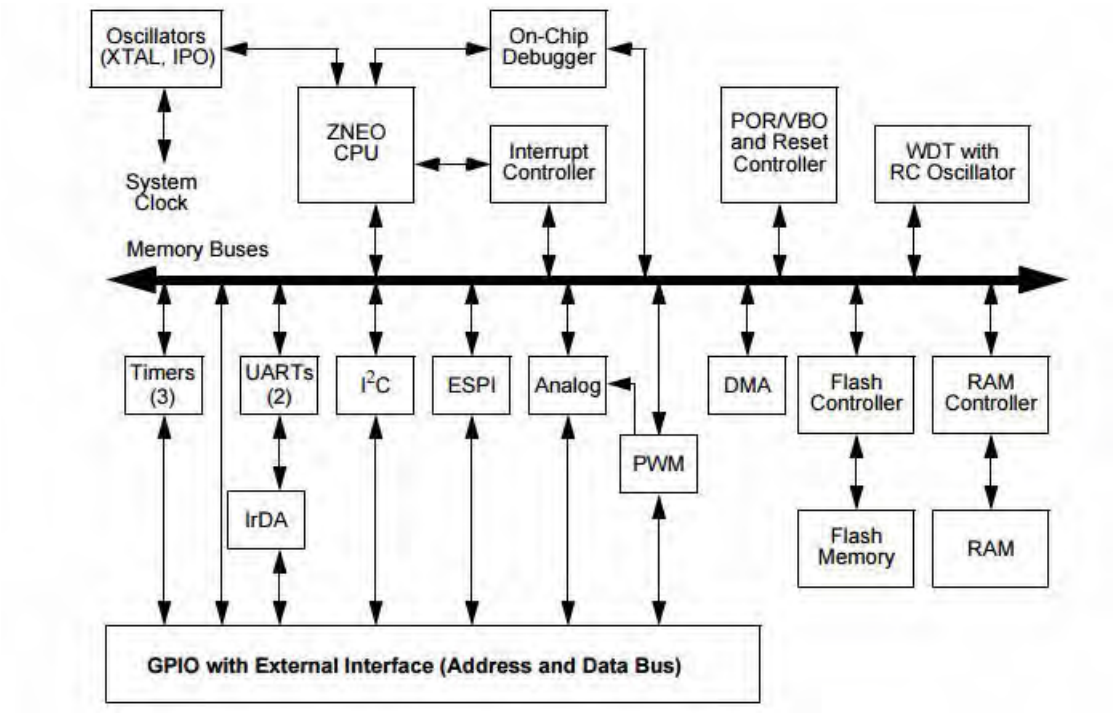

Figure 1. ZNEO Z16F Series Block Diagram

Discussion

A DAC generates an analog output that is proportional to the digital input it receives. An n-bit DAC features a reference voltage commonly referred to as V_{ref}. The DAC output varies from 0 to V_{ref}, corresponding to the input, which varies from 0 to the maximum digital value (2N – 1) allowed at the inputs for a unipolar DAC. A bipolar DAC considers the digital input to be a signed number and varies its output from – to + as per the digital coding and/or interpretation of the input.

This application note describes:

- Basic concepts of PWMs
- Accuracy of PWMs when used to generate analog voltages
- Basic filters for PWM outputs
- PWM code examples

Pulse Width Modulation Basics

A pulse width modulated signal is a digital signal with a constant frequency, where the duty cycle, i.e. the fraction of time the signal is high or low, represents the modulating variable. It is a digital signal that can be used to represent an analog value, i.e. where the analog value is the duty cycle.

Zilog MCU timer peripherals are able to directly generate a PWM output with only an initial setup, and require no continuing intervention from the CPU. The frequency of the PWM and the duty cycle is an integer number of system clock pulses, as shown in the following equations:

$$P_{PWM} = n \times P_{sysclk}$$

$$D_{PWM} = i / n$$

In the above equations, *P* represents the period, *n* is the number of system clock pulses in each PWM period, *i* is the number of system clock pulses in the duty cycle on-time high side, and *D* is the duty cycle. *D* will be a number between 0 and 1.

The average voltage of the output is calculated using the following equation:

$$V_{avg} = V_{on} \times D + V_{off} \times (1 - D)$$

Zilog microcontrollers have CMOS outputs that swing from rail to rail when driving a capacitive load; therefore,

$$V_{on} = V_{cc}$$

$$V_{off} = 0$$

Therefore, to create a simple analog to digital converter, it is necessary to only drive a low pass filter with a PWM, using the filter to demodulate the PWM. Without considering sources of error, the output voltage in this case will be:

$$V_{out} = D \times V_{cc}$$

In the above equation, *D*, the duty cycle, is taken as the percentage of time the PWM signal is high.

Accuracy of PWM Generated Voltages

The error of a PWM-generated reference voltage is the sum of several sources of inaccuracy, including the following:

PWM resolution. The integer number of system clock pulses in the PWM period determines the resolution. For instance, if the PWM period is 100 clocks, and the duty cycle can only vary in integers, the minimum resolution of 1 clock would give an error of 1/100, or 1%. Usually, PWM periods are given in simple binary values like `0xFF` (255 decimal) or `0x03FF` (1023 decimal). A resolution of 256 counts or 8 bits gives a minimum error of about ½%.

Supply voltage. The supply voltage is the main determinant of the V_{on} voltage given above. Modern system supplies can have 1% error at best. If the precision of the PWM DAC is of utmost importance, pay attention to the power regulation scheme.

Demodulation ripple voltage. Filtering the output does not produce a perfectly constant voltage. Some error remains due to the high frequency components of the PWM that are not eliminated, depending on the filter characteristics (see Basic Filters for PWM Signal). There is little reason to minimize ripple below the errors caused by PWM resolution and supply voltage. A well designed system and filter will minimize the contribution of ripple to error, but will not eliminate it.

Achieving the highest accuracy comes with trade-offs. Increasing the resolution, i.e. the number of clocks in the PWM period, lowers the carrier frequency and increases the ripple voltage given the same filter parameters. Reducing the ripple by decreasing the cutoff frequency of the filter reduces the bandwidth of the digital-to-analog conversion.

For most practical systems, it is most difficult to eliminate a supply voltage error. Therefore, it is not sensible to strive for resolution or ripple errors much lower than supply errors. If supply error is around 1%, then a resolution of 8 bits and a ripple error < 1% is a practical compromise.

Basic Filters for PWM Signal

Multiple basic design texts and references address the performance of low pass filters. The simplest implementation of a low pass filter is a 1^{st} order, passive RC circuit. Simple reasoning states that in order to smooth out the PWM signal, the RC constant should be much longer than the PWM period. Since a 1^{st} order filter's stop band roll off rate is –20 dB per decade, to get a 1% ripple error, the typical RC constant would need to be approximately 1/100 of the PWM period or 100 times the PWM frequency, achieving –40 dB ripple, or about 1%. An example follows.

With 8-bit resolution (256 total counts), and setting the ZNEO internal oscillator frequency of 5.5296 MHz = PWM clock frequency:

$$PWM = (256) \times (1/f) = 256 \times 1/5.5296\ Mhz = 46.3\mu s$$

$$Fc = -40\ dB\ cutoff\ frequency = 1/(4.63\mu s \times 100) = 216\ Hz$$

Using Fc = 216Hz and solving for RC:

$$Fc = 1/(2 \times \pi \times RC)$$

$$RC = 736\ \mu s \qquad \text{(Equation 1)}$$

The resulting time constant is 736 µs. In terms of the *R* value, it depends on the load resistance (*RL*) that the RC filter would drive. A standard design rule is to make R = RL/10. By incorporating this guideline, the user does not lose too much power in the filter resistor. Not following the formula could also distort the calculation, causing the load resistance to be too low in comparison to the filter resistor. The load resistor also influences the filter frequency. A factor of 1/10 makes the error negligible. For example, if the load resistance is 736 Ohms (RL=736 Ohms) then the filter resistor R=73.6 ohms and the filter capacitor C=10 uFd.

The RC calculation shows why having reasonable precision on a reference voltage leads to the trade-off mentioned above – reduced DAC bandwidth. If better bandwidth is desired without sacrificing precision, a second or higher order filter is required. A second order filter, with a roll off rate of –40 dB per decade, would effectively increase the bandwidth by 10.

One other consideration is output impedance. Loading a CMOS PWM and RC circuit will change its characteristics, and perhaps also change the V_{on} and V_{off} parameters in our initial PWM consideration. A common way to address this issue is to use an operational amplifier (op amp) configured as a unity gain driver. Because the ZNEO and several other Zilog microcontrollers contain an op amp, this option is easily and cheaply available to the system designer.

Other Filters for PWM Signal

A brief description of some types of filters that are not passive RC type filters is provided in this section. These filters are considered active filters.

Butterworth Filter

The Butterworth filter is also referred to as a maximally flat or flat flat filter. This class of filters approximates the ideal filter well in the pass band. Frequency response curves of different types of filters are shown in Figure 2. The Butterworth filter has an essentially flat amplitude-frequency response up to the cutoff frequency. The sharpness of the cut-off can be seen in the image. The Butterworth, Chebyshev, and Bessel filters reach a roll-off slope of –40 db/decade at frequencies much larger than the cut-off frequency. Although the Butterworth filter achieves the sharpest attenuation, its phase-shift as a function of frequency is non-linear. It has a monotonic drop in gain, with frequency in the cut-off region and a maximally flat response below cut-off frequency, as illustrated in Figure 2. The But-

terworth filter has characteristics somewhere between those of Chebyshev and Bessel filters. This filter has a moderate roll-off of the skirt and slightly non-linear phase responses.

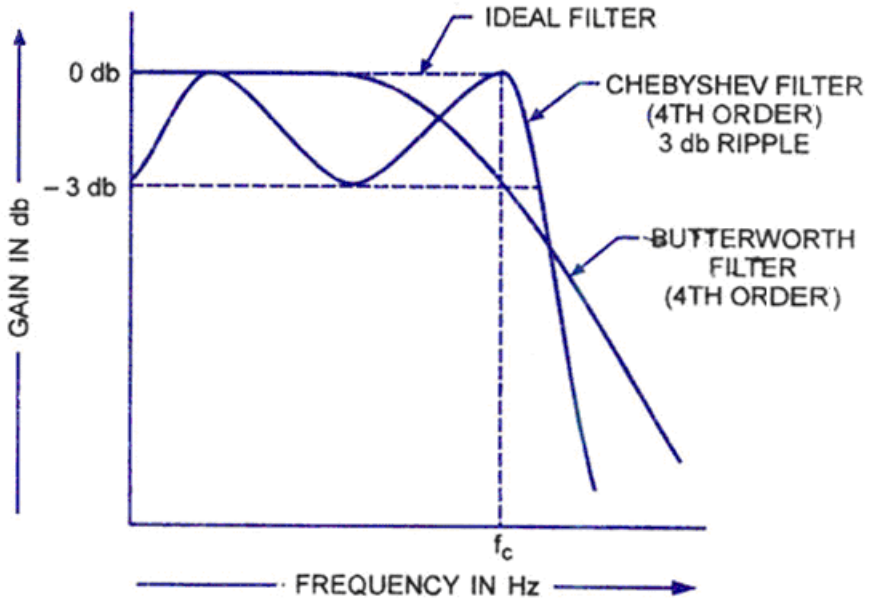

Figure 2. Chebyshev and Butterworth Filters

Chebyshev Filter

The Chebyshev filter, also shown in Figure 2, gives a sharper cut-off than the Butterworth filter in the passband. This filter is also called an equal ripple filter. Butterworth and Chebyshev filters exhibit large phase shifts near the cut-off frequency. A drawback of the Chebyshev filter is the appearance of gain maxima and minima below the cut-off frequency. This gain ripple, expressed in db, is an adjustable parameter in filter design.

The faster the roll-off, the greater the peak-to-peak ripples in the passband. The phase response is highly non-linear in the skirt region. Such unequal delays of data frequency in the passband cause severe pulse distortion and therefore, increased errors at modern demodulators. This issue can be partially overcome by increasing the BW of the filter so that the phase region is extended. A Chebyshev filter is used where very sharp roll-off is required. However, this is achieved at the expense of a gain ripple in the lower frequency passband.

Bessel Filter

The Bessel filter provides ideal phase characteristics with an approximately linear phase response nearly up to the cut-off frequency. It has a very linear phase response but a fairly gentle slope, as shown in Figure 3. The Bessel filter is used for applications where the phase characteristic is important. It is a minimal phase shift filter even though its cut-off characteristics are not very sharp. It is well suited for pulse/PWM applications.

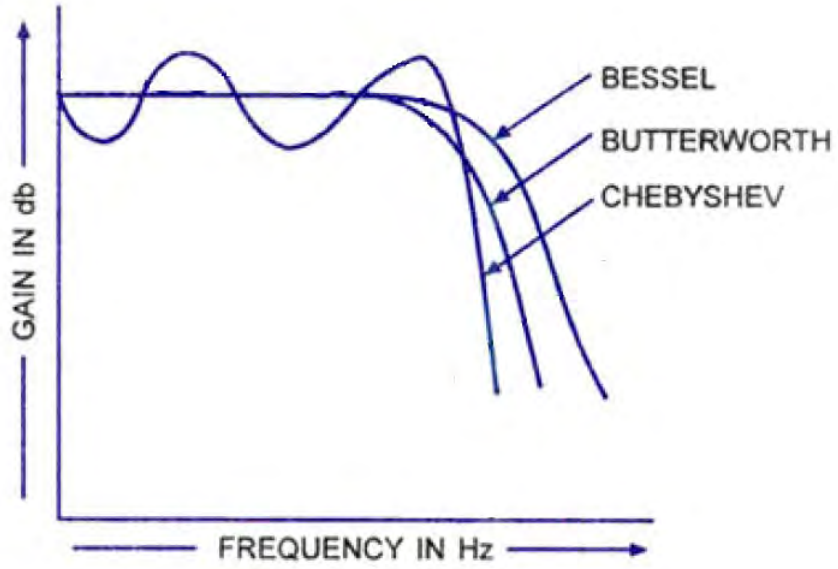

Figure 3. Bessel Filter Frequency Response

Elliptic Filter

The Elliptic filter has the sharpest roll-off of all filters in the transition region but has ripples in both the pass band and stop band regions, as illustrated in Figure 4. The Elliptic filter can be designed to have very high attenuation for certain frequencies in the stop band, which reduces the attenuation for other frequencies in the stop band.

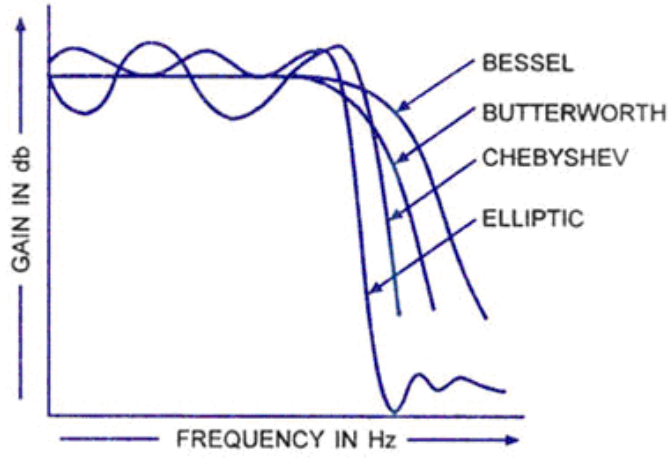

Figure 4. Elliptic Filter Frequency Response

Basic Filter PWM Code Example

The PWM signal's duty cycle, in conjunction with the basic RC Filter load, is used to create the desired DAC voltage. If V_{REF} is 2V, the created voltage is a percentage of 2V and controlled by the percentage of the ON time of the PWM duty cycle through a voltage translator. A flow chart is shown in Appendix A. Flowchart on page 15.

The code listings using Standard Timer PWM and Multi-Channel Timer PWM are available in the source code file, AN0357-SC01.

Basic Filter Software Implementation

The PWM code follows the approach described earlier in the application note under Accuracy of PWM Generated Voltages on page 4. It describes using a resolution of 256 counts or 8-bit. This translates to using the lower 8-bits of the16-bit PWM. For a range of 0V–2.0V, 2.0V would represent a duty cycle of 100% which would be characterized by a full 8-bit count or a count of 256 (`0xFF`), while 1.5V, 1.0V, 0.5V, and 0V would represent 75%, 50%, 25%, and 0% duty cycles respectively, or counts of `0xC0`, `0x7F`, `0x40`, and `0x00` respectively.

The application software takes a reading from an analog channel using the 10-bit ADC. The applied voltage on the analog channel is limited to a range of 0–2.0V since the V_{ref} voltage is 2.0V. The 10-bit ADC result is converted and processed into a special 8-bit value. This 8-bit value now represents the duty cycle percentage and is loaded into the counter that controls the duty cycle at the end of the PWM cycle. The PWM is updated and the PWM output signal generates the new duty cycle. At the end of the PWM cycle, a vectored interrupt occurs and a new ADC reading is taken to start the processing of the next duty cycle value while the PWM is in its current cycle.

Note: The counter for the PWM runs continuously so the PWM signal never stops.

Developing the Application with ZNEO

Two software files are provided with this ZNEO application note. One uses a PWM feature of a standard timer while the other uses the Multi-Channel PWM Timer. Both software files use the same approach. The only difference is the peripheral that is used to generate the PWM signal. See the illustration in Appendix B. Functional Diagram of System on page 16 for the functional architecture of the system. Because the PWM output signal continuously varies from 0V to 3.3V and back, a voltage divider is used to translate the voltage to the RC Filter so that only 0–2.0V is applied to the load. See the schematic diagram in Appendix C. Schematic Diagram on page 17.

Testing the Application

This section describes the procedure to test or demonstrate this application and provides information about equipment used, system configuration, and the expected results.

Equipment Used

- 3.3V Power Supply to power the application board.
- An application board where the ZNEO MCU is mounted and connected according to the schematic in Appendix C. Schematic Diagram on page 17.
- A Variable Power Supply (VPS) is used to vary the voltage that is applied to PB0/ ANA0 input pin
- An oscilloscope to look at the resulting voltage signals
- USB Smart Cable
- ZDSII ZNEO 5.2.0
- A Personal Computer (PC)

System Configuration

The 3.3V power supply powers the ZNEO MCU application board. The 0–2V VPS is connected to the input of the ADC pin PB0/ANA0. The MCT PWM output pin PC6/PWMH0 and the Timer1 output pin PC1/T1OUT are connected to header JMP1 pins 1& 3, respectively. The header JMP1 pin2 is connected to a voltage divider R3 and R5 to generate a maximum voltage of 2 V when the PWM duty cycle is at 100% since the PWM output signal from T1OUT and PWMH0 varies 0–3.3Vp–p.

If using the MCT PWM software, then shunt JMP1 pin 1 to pin 2 so the load sees the signal from pin PC6/PWMH0. If using the Timer1 PWM software, then shunt JMP1 pin 3 to pin 2 so the load sees the signal from pin PC1/T1OUT. See the schematic diagram in Appendix C. Schematic Diagram on page 17.

The RC filter is comprised of R4 and C8. The RC filter components are selected from the earlier equation 1 where RC = 736 us (~ 73.2 Ω x 10 uFd). The 10uFd was chosen for its large energy storage capabilities and availability. The 73.2 Ω was chosen for its low resistance capacitive charging capabilities.

Setup

The VPS applies a voltage on PB0/ANA0 pin. The ADC then measures the voltage and converts it to a digital ADC value. The ADC value is converted to a PWM duty cycle value which is then loaded into the PWM duty cycle register during the PWM interrupt service routine that occurs when the PWM completes its duty cycle. The new TP2 voltage reflects the change in the new PWM signal from the duty cycle update.

Procedure

1. Power up the application board.
2. Using the ZDSII ZNEO and USB Smart Cable, compile and Flash program the code to any Z16F28, Z16F64, or Z16F32 MCU device that is on the application board.
3. After downloading the code to the ZNEO MCU, click the **Stop Debugging** icon to stop the debugger to disconnect the application board from the ZDSII software.
4. Power off the MCU and remove the USB Smart Cable.
5. Connect the variable power supply to PB0/ANA0 and set the applied voltage to 0V.
6. Connect the oscilloscope to TP2 to measure the voltage at C8.
7. Put the oscilloscope probe on R3 at TP1 to look at the PWM waveform. It should show a 0% duty cycle waveform as shown in Figure 5.

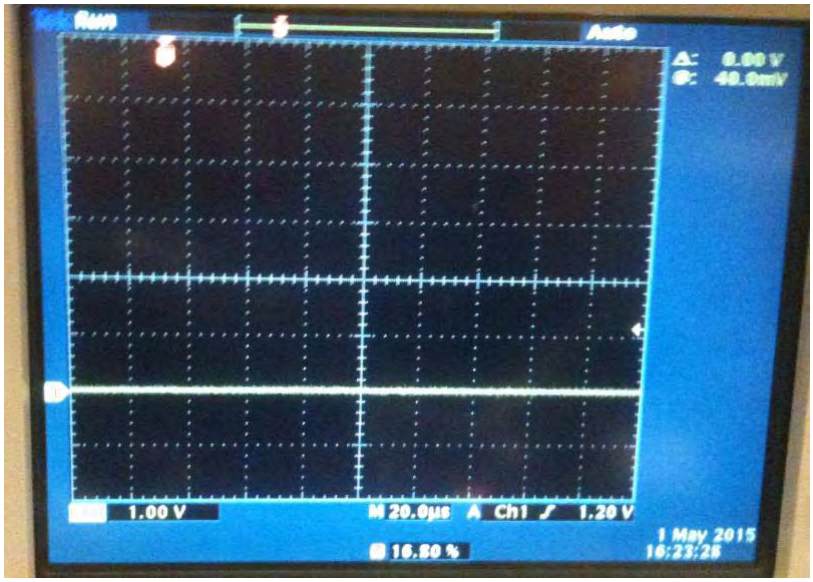

Figure 5. 0% Duty Cycle Waveform

8. Power up the application board and adjust the VPS to 0.5V. The oscilloscope should read 0.5V.
9. Put the oscilloscope probe on R3 at TP1 to look at the PWM waveform. It should show a 25% duty cycle waveform as shown in Figure 6.

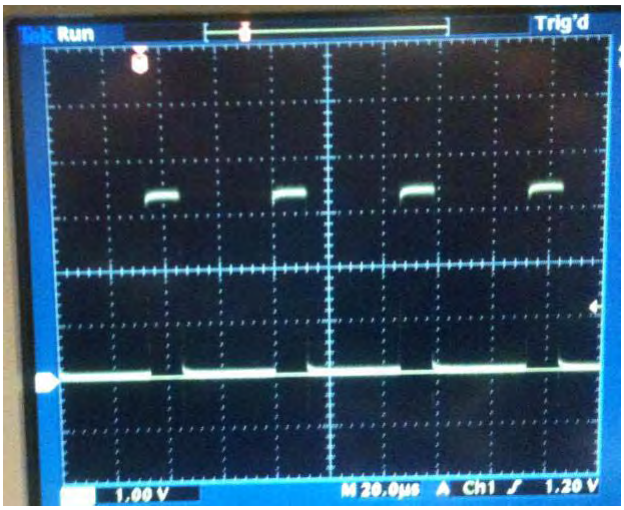

Figure 6. 25% Duty Cycle Waveform

10. Adjust the VPS to 1V. The oscilloscope at R3 should show a 50% duty cycle like the waveform shown in Figure 7 and the voltage at C8 is 1V.

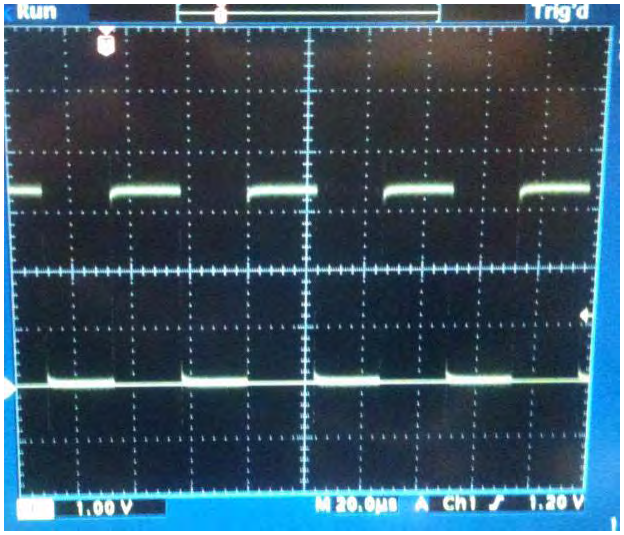

Figure 7. 50% Duty Cycle Waveform

11. Adjust the VPS to 1.5V. The oscilloscope should show a 75% duty cycle like the waveform shown in Figure 8 and the voltage at C8 is 1. 5V.

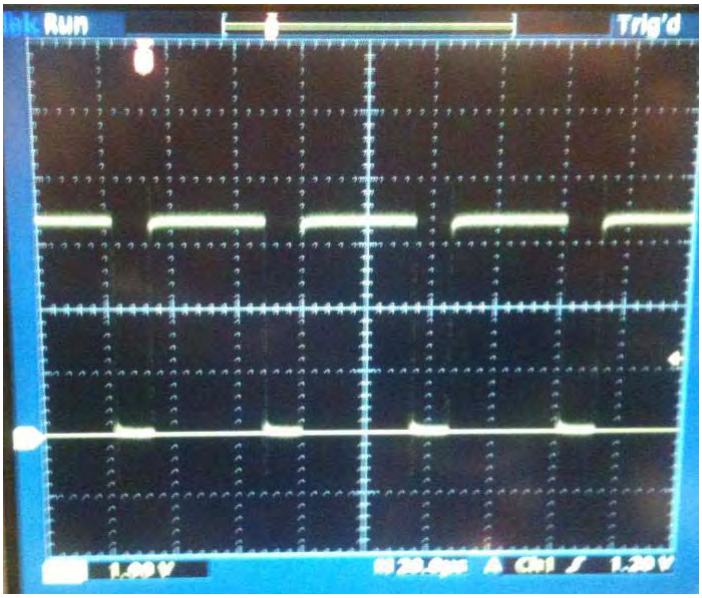

Figure 8. 75% Duty Cycle Waveform

12. Adjust the VPS to 2V. The oscilloscope should show a 100% duty cycle like the waveform shown in Figure 9 and the voltage at C8 is 2V.

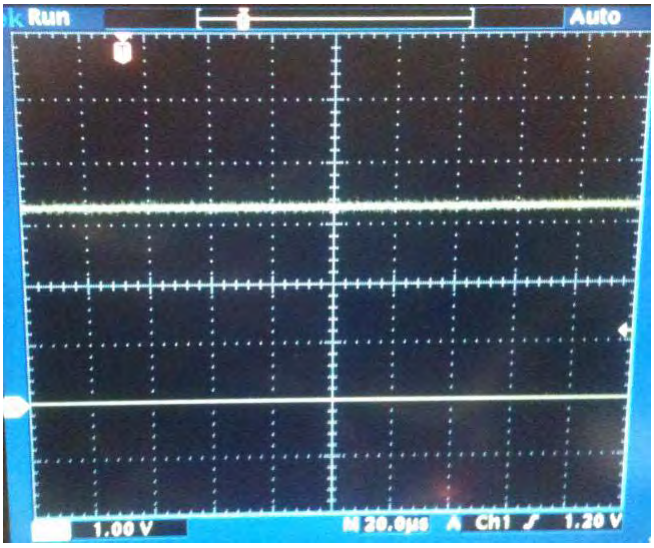

Figure 9. 100% Duty Cycle Waveform

13. Power off the VPS, and then switch off the power to the application board. This completes the testing/demonstration of the PWM software.

Results

The results of the testing showed that the ZNEO MCT PWM and its standard Timer1 PWM can both perform like a DAC for voltages 0–2V by applying a PWM signal to a basic RC filter.

Summary

The ZNEO MCU can be used in applications that requires a single channel digital-to analog conversion. The ability to generate an arbitrary voltage or an analog signal is useful in circuit applications.

References

The documents and sources referenced in this application note are listed in Table 1. Each of these documents can be obtained from the Zilog website or other source by clicking its associated link.

Table 1. Related Documentation

Document ID	Description
PS0220	ZNEO Z16F Series Product Specification
UM0188	ZNEO CPU core User Manual
http://www.circuitstoday.com/active-filter-types	Images and information about Butterworth filter, Chebyshev filter, Bessel filter, and Elliptic filter

Appendix A. Flowchart

Figure 10 shows a flow of the microcontroller operation.

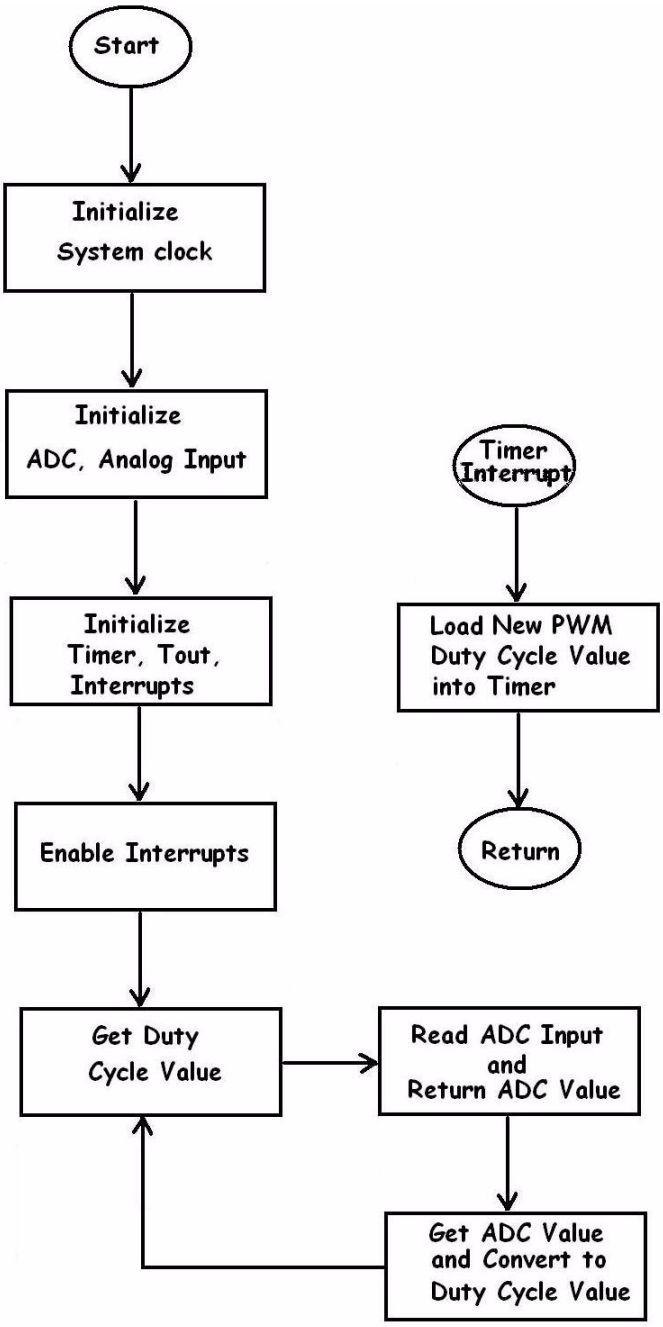

Figure 10. Microcontroller Operation Flow Chart

Appendix B. Functional Diagram of System

Figure 11 shows a functional system diagram.

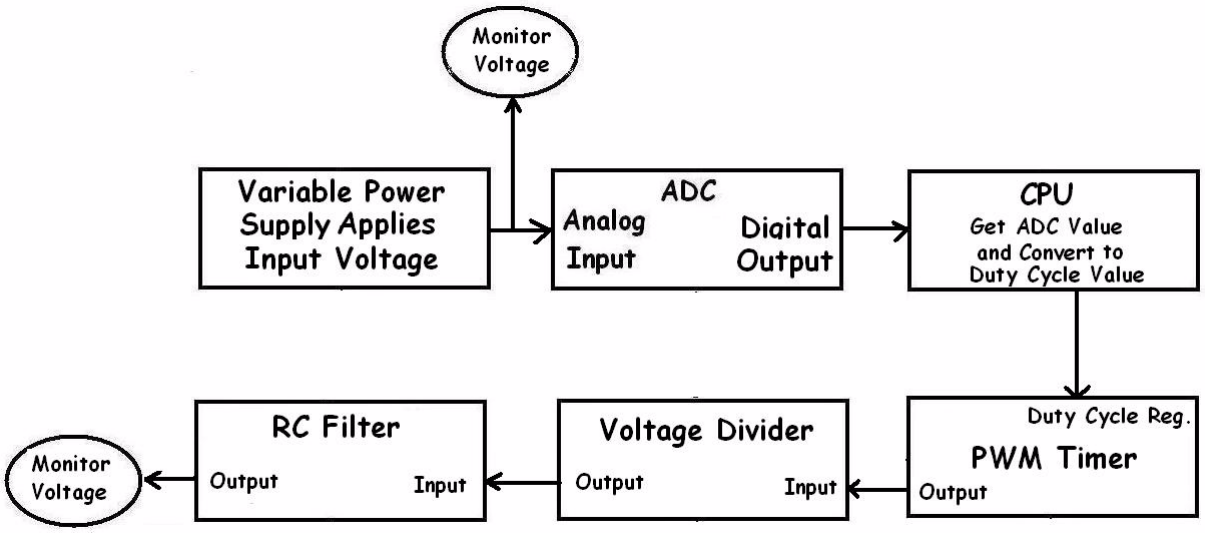

Figure 11. Functional System Diagram

Appendix C. Schematic Diagram

Figure 12 shows a schematic diagram of the PWM as DAC.

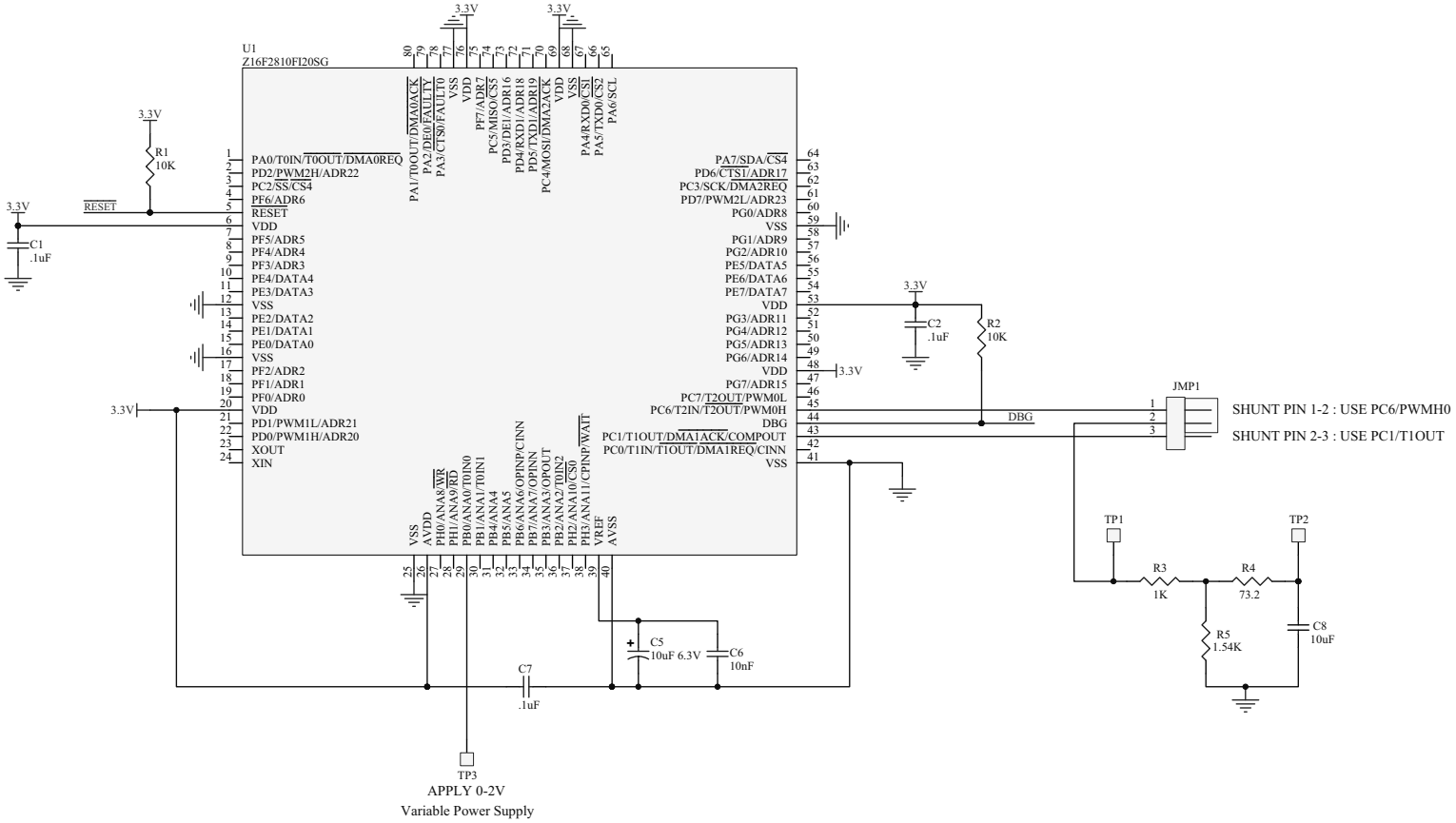

Figure 12. Schematic Diagram

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.

ZNEO and Z16FMC is a trademark or registered trademark of Zilog, Inc. All other product or service names are the property of their respective owners.