STEP-DOWN VOLTAGE CONVERTER

INTRODUCTION

In many cases, electronic circuits require an accurate and stable voltage supply. The voltage provided by transformers, batteries, generators, and so on are unstable and not very accurate. Therefore, a voltage regulator must be placed between the source and the load.

In the past, linear voltage regulators were used for this task. They were inexpensive, stable, exact, and very easy to use. However, they burned great quantities of fuel, though, which meant they wasted a lot of power.

The alternative solution was switching regulators. Though switching regulators were expensive, complicated, and generated ripple, they were very economical. In fact, designs of switching regulators have greatly improved in recent years. The disadvantages have declined, and switching regulators are widely used today.

ADVANTAGES

The step down voltage converter is probably the most commonly used switching regulator. Using a ZiLOG microcontroller (MCU), this application note describes a basic step down voltage converter. The step down voltage converter can drive all kinds of loads (ohmic, inductive, capacitive), including:

- Halogen light bulbs
- Motors
- Relays

The main advantage of this step down voltage converter, compared to others is that the ZiLOG converter is able to perform other tasks as well. Using this converter, an engineer can implement many designs including an IR-receiver, revolution-regulation, current surveillance, or power supply failure handler.

There is abundant literature available explaining the function of a step down converter. This application note is intended to explain one form of implementation.

REQUIREMENTS

The first requirement for creating a step down voltage converter is a Pulse Width Modulator (PWM). The MCU essentially generates the PWM. There are two MCU choices:

- The Z86E02
- The Z86E31

The Z86E02 is the most cost effective and generates PWMs up to almost 8kHz. If a 16kHz PWM is required, use the Z86E31. This 7-bit PWM has fair resolution. If the PWM frequency is raised, then the resolution must be lowered.

REQUIREMENTS (Continued)

The second requirement is a switch. A switch is made from the transistor network (BC548B + TIP116), which can source up to 4A. The design engineer must consider a heat sink for the TIP116 for this specific application. The following calculation illustrates this approximation:

$$P_{heat} = Uce * I * fpwm * (t_{on}+t_{off})$$

P_{heat} = power with which the transistor silicon is heated [W]
U_{CE} = voltage across collector and emitter [V]
$\hat{I}$ = maximum current through the transistor /load [A]
$fpwm$ = PWM frequency [Hz]
t_{on}, t_{off} = transistor switching times [s]

The heat transmission resistance of the heat sink can be calculated as follows:

$$R_{th} - \frac{(\hat{T}_{silicon} - \hat{T}_{ambient})}{P_{heat}}$$

R_{th} = thermal resistance from silicon to ambient $\left[\frac{K}{W}\right]$
$\hat{T}_{silicon}$ = maximum silicon temperature [K]
$\hat{T}_{ambient}$ = maximum ambient temperature [K]

The necessary thermal resistance of the heat-sink is represented as:

$$R_{th\ heat\text{-}sink} = R_{th} - R_{th\ transistor} + R_{th\ transistor > heat\text{-}sink}$$

$R_{th\ heat\text{-}sink}$ = *thermal resistance from heat sink to ambient* $\left[\frac{K}{W}\right]$

$R_{th\ transistor}$ = *thermal resistance from silicon to package* $\left[\frac{K}{W}\right]$

$R_{th\ transistor > heat\text{-}sink}$ = *thermal resistance* $\left[\frac{K}{W}\right]$

The thermal resistance from the transistor package to the heat sink depends on whether a thermal coat, glimmer isolation (ca. 2K/W), or crema (0.5K/W) is used.

The third part necessary for a step down converter is the inductance. The minimal inductance (L1) value is calculated with the following formula:

$$L_1 = \frac{U_0}{4 * f_{pwm} * \Delta I}$$

L_1 = inductance [*H*]
U_0 = supply voltage [*V*]
f_{pwm} = PWM frequency [*Hz*]
ΔI = maximum current ripple [*A*]

To keep the current ripple low, high inductance values are preferable. However, to keep the regulation loop stable, low inductance values are better. Keep the inductance value as small as possible to maintain regulation loop stability.

VOLTAGE CONTROL

To make a step down voltage converter, the voltage must be controlled. This control is accomplished by feeding the voltage across the load back into the MCU. The load voltage is available at port 31, and an adjustable reference voltage is available at port 33. The MCU compares the two voltages and adjusts the duty cycle of the PWM accordingly, provided that the two voltages are not the same. When the voltage across the load is too low compared with the reference voltage, then the MCU increases the duty cycle of the PWM. When the voltage across the load is too high compared with the reference voltage, then the MCU decreases the duty cycle of the PWM. P1 adjusts the reference voltage and depending on this reference the MCU shortens or lengthens the duty cycle of the PWM. This condition causes the load voltage to lower or rise.

REGULATION LOOP

The step down voltage converter is a regulation loop. The program provides a simple integrating regulator with a fixed phase shift of 90˚. The transistor network is an amplifier. The inductor is a PT1, with a maximum 90˚ phase shift. The voltage measuring is done by R5-R6, which is an amplifier with an amplification of 0.325. Consequently, there is a maximum phase shift of 180˚ with an ohmic or inductive load. Ohmic loads are neutral, whereas inductive loads decrease the stability. With a capacitive load, however, the phase shift can reach a maximum of 270˚. Therefore, with an ohmic or inductive load, the step down converter can oscillate; with a capacitive load, the converter can be unstable under worst-case conditions.

To avoid this problem, keep the value of the inductance (L1) as low as possible. If swinging tendencies of the step down converter are still too high, reduce the value of *integration_time* in the program. As a final step, reduce the input voltage.

Follow these steps when the swinging tendency is too high. Under normal conditions, these steps are not necessary. Figure 1 illustrates the regulation loop sequence.

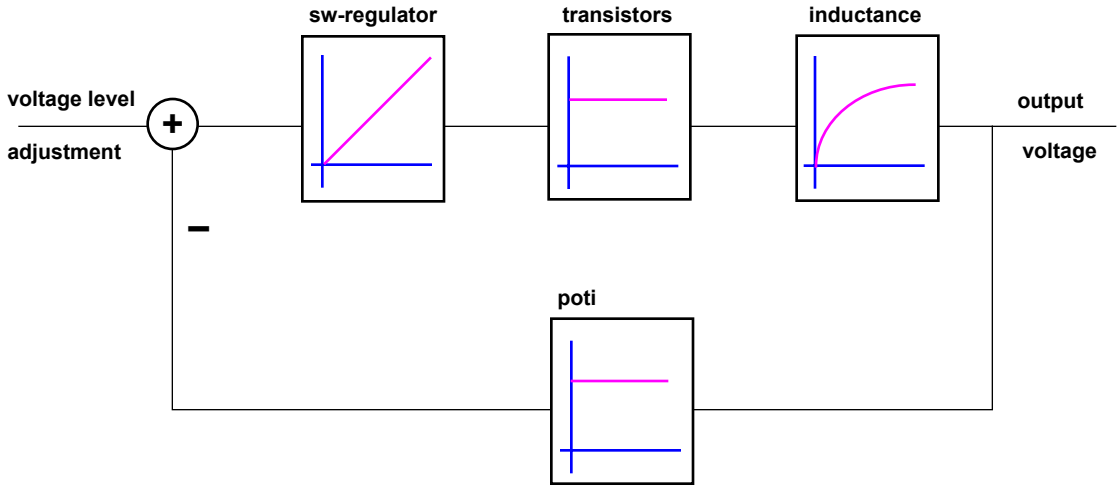

Figure 1. Regulation Loop Diagram for Ohmic or Inductive Loads

THE MAIN PROGRAM

The definition of variables, constants, and macros are defined at the beginning of the software code. The constants may be altered to customize the voltage-regulator. Next, set the *Softstart* feature to be active or inactive. From that point, select either the Z86E02 or the Z86E31 as the target controller.

The Z8 initializes and enters the main program. The main program consists of NOPs which can be replaced by custom-made subroutines. The regulator and the PWM generator are interrupt-driven.

First, load the timer (T1) with the next half-period of the PWM cycle. Next, enter the regulator routine. From that point, the PWM is adjusted to achieve the pre-adjusted value of the voltage. Finally, exit the interrupt; the Z8 returns to the main

SCHEMATICS (Continued)

program. The Z8 is a very robust and fast MCU, and there is enough core performance left to execute various other tasks in the main program. Figure 2 illustrates the main program sequence. Figure 3 illustrates the interrupt routine.

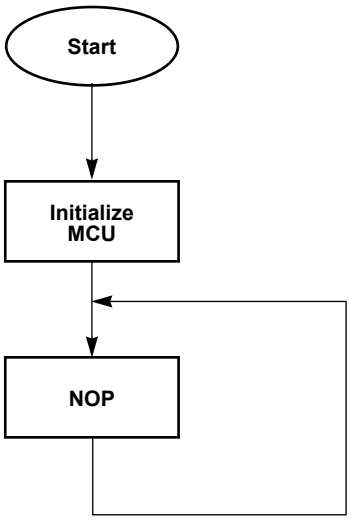

Figure 2. Program Sequence

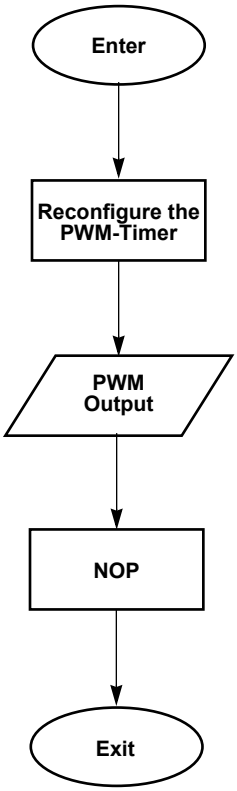

Figure 3. Interrupt Routine

SCHEMATICS

Figures 4 and 5 illustrate the schematic diagrams of the two chips.

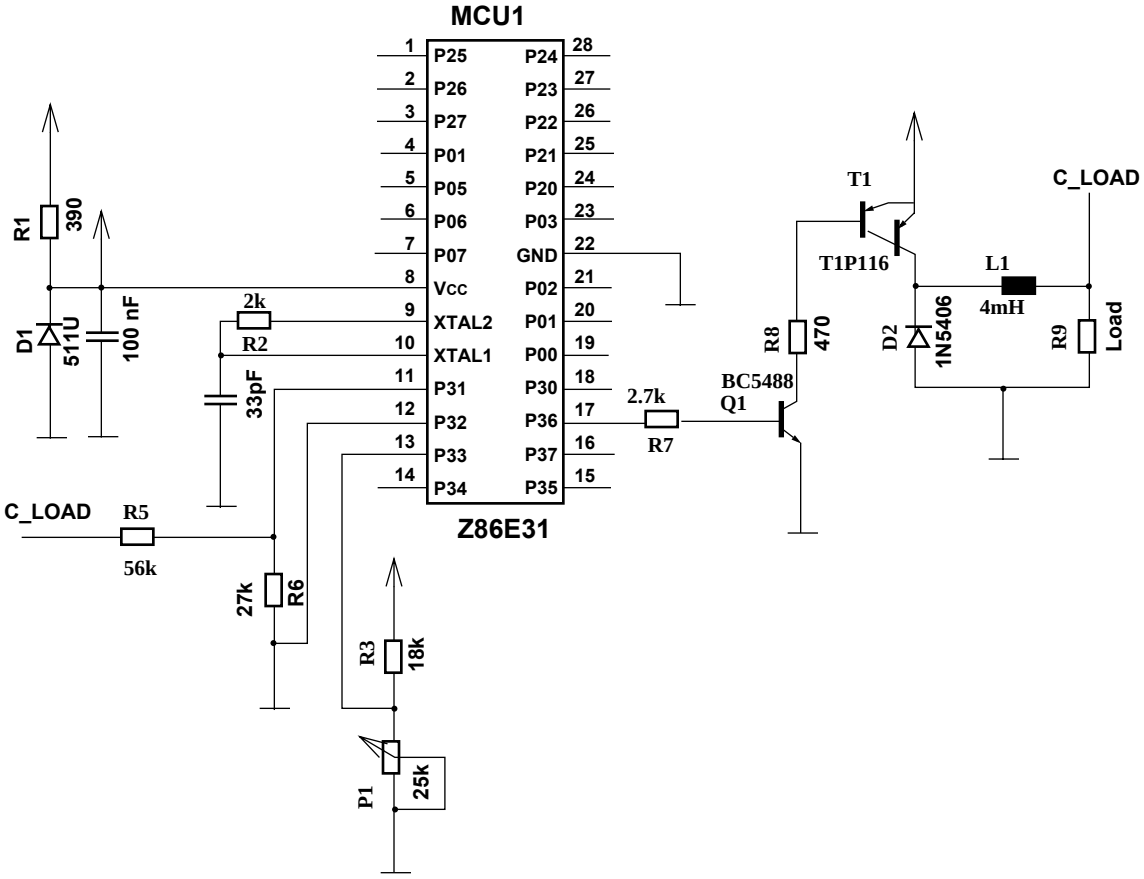

Figure 4. Voltage Regulator Schematic Using the Z86E31

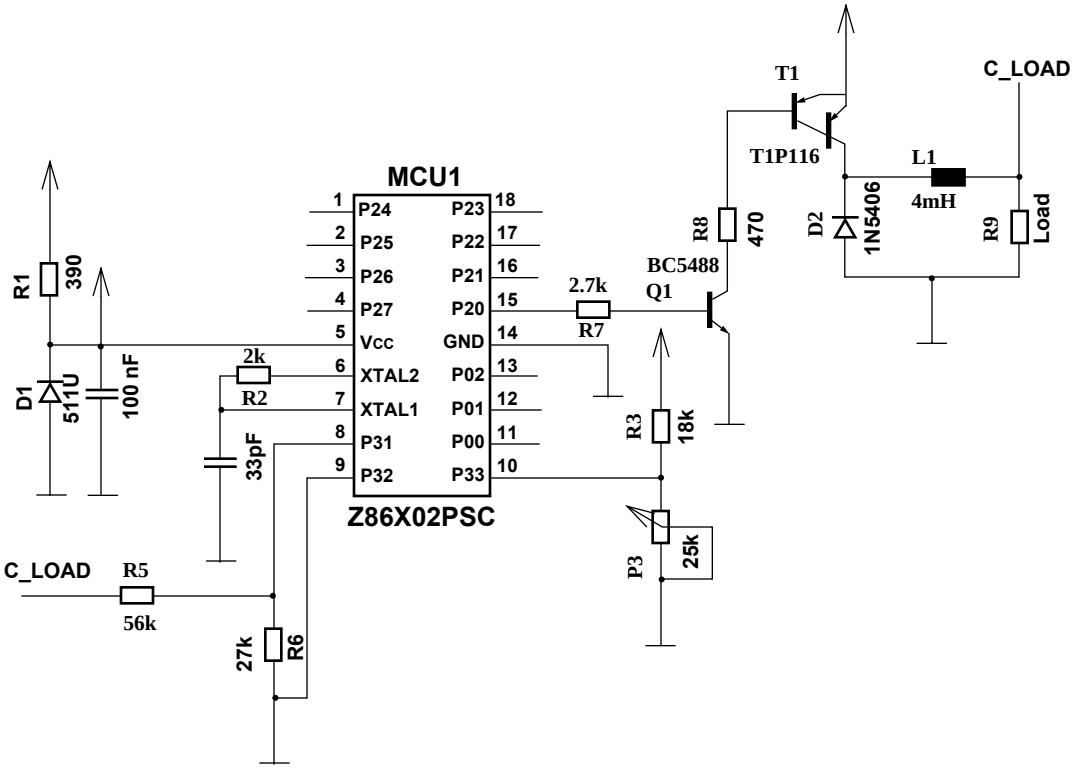

Figure 5. Voltage Regulator Schematic Using the Z86X02PSC

SAMPLE CODE (Continued)

SAMPLE CODE

Pages 7–13 illustrate the step-by-step process of creating a step down voltage regulator.

CONCLUSION

By following the basic guidelines contained in this application note, any design engineer can ensure a quick and easy implementation of a step down voltage converter for a variety of design applications.

```
;*****************************************************************
;       This application note is for 8MHz.
;       “Using A Z86E02 MCU For “Step Down Voltage Regulation” via pwm.
;
;       FILE:           vol_reg1.S
;       DATE:           23.12.97
;       MCU:            Z86E02
;       PROJECT:  Voltage Regulator
;       AUTHOR:         Klaus Buchenberg
;       SOFTWARE: REVISION 3.0
;
;       This program is assembled by ZiLOG ZMASM assembler
;*****************************************************************

      GLOBALS     ON

;*****************************************************************
;       When the softstart option is wished, then set
;       SOFTSTART_WISHED to 1 = Yes
;*****************************************************************
SOFTSTART_WISHED.equ    0 ; 1 = Yes     0 = No

;*****************************************************************
;       Bitnumber definitions
;*****************************************************************
bitno0 .equ%01
bitno1 .equ%02
bitno2 .equ%04
bitno3 .equ%08
bitno4 .equ%10
bitno5 .equ%20
bitno6 .equ%40
bitno7 .equ%80

;*****************************************************************
;               PORTS DEFINITION
;*****************************************************************

;       Port 0 pin
;       P00: power. UP key
;upkey  .equ    bitno0              ; not used
;       P01: power. DOWN key
;downkey.equ    bitno1; not used
;       P02: idle input


;       Port 2 pin
;       P20: output for pwm
pwm_output.equ    bitno0
;       P21-P27: output idle N.C.


;       Port 3 pin
;       P31: output voltage sensing
Usense            .equ  bitno1
;       P32: idle input, pull to ground when not used
;       P33: voltage reference
Uref        .equ  bitno3

*****************************************************************
;               REGISTERS DEFINITION
;*****************************************************************
integration_timer .equ  r4    ; counts the number of periods
; softstart_counter     .set  r5    ; counter for softstart routine
duration_of_low_cycle   .equ  r6    ; length of low time of period
duration_of_high_cycle  .equ  r7    ; length of high time of period
; delay_counter         .set  %FE   ; counter for delay routine
```

```
;*****************************************************************
;               BITS DEFINITION
;*****************************************************************



;*****************************************************************
;               CONSTANTS DEFINITION
;*****************************************************************
pre1_min          .equ    01111011b ; PRE1=30, continuous mode, int. clock
                                    ; pwm freq. =
                                    ; osc. freq./(PRE1 * 8*no_voltage_levels)
                                    ; in this case pwm freq. = 521Hz
                                    ; pwm freq. = 8E6Hz/(30*8*64)
no_voltage_levels .equ    64              ; number of voltage levels
start_up_voltage  .equ    2                     ; voltage start level <=
no_voltage_levels
max_high_cycle    .equ    63                    ; max duration of high cycle
inv_softstart_time      .equ    255                   ; extends start up time by 3
                                    ; extension factor = 765/inv_softstart_time

;*****************************************************************
;               MACROS
;       Refer to Z8 technical manual for macro definition
;*****************************************************************
bset        MACRO register,bitnumber            ; set the appropriate bit in
            or          \register,#\bitnumber         ; the specified register
            MACEND

bclr        MACRO register,bitnumber            ; clear the appropriate bit in
            and         \register,# ~(\bitnumber)     ; the specified register
            MACEND

brset       MACRO register,bitnumber,label            ; IF the appropriate bit in
            tcm         \register,#\bitnumber         ; the specified register is set
            jr          z,\label                      ; THEN jump to label
            MACEND                              ; ELSE go on

brclr       MACRO register,bitnumber,label            ; IF the appropriate bit in the
            tm          \register,#\bitnumber         ; specified register is reset
            jr          z,\label                      ; THEN jump to label
            MACEND                              ; ELSE go on

pwm_high_cycle MACRO
            bset    r2,pwm_output
            MACEND

pwm_low_cycle     MACRO
            bclr    r2,pwm_output
            MACEND

;*****************************************************************
;               INTERRUPTS VECTOR
;*****************************************************************

            .MLIST
            .LIST

;       Interrupt vector address %00 to %0C

            .ORG   %0000
            .word   irq0
            .word   irq1
            .word   irq2
            .word   irq3
            .word   irq4
            .word   timer1

;*********************************************
;               PROGRAM STARTS HERE           *
```

```
;*********************************************

BEGINNING:
            .ORG %0C
irq0:
irq1:
irq2:
irq3:
irq4:
            di
            ld      P01M,#00000101b ;P0, P1 input, internal stack
            ld      P2M,#00000000b ;P20-P27 output
            ld      P3M,#00000011b ;P3 analog + P2 push pull
            and   P2,#11111110b    ;Switch off transistor

            clr     SPH
            ld      SPL,#%40           ; INIT STACK POINTER
            ld      IPR,#00001010b  ; IRQ5 has highest priority
            ld      IMR,#00100000b  ; enable T1 interrupt

; INITIALIZE RAM TO “0”
;           srp     #%30
;           ld      R14,#%3d
;zram:            clr     @R14
;           djnz    R14,zram



;       Initialize all registers

            srp   #%00                          ; set working register to %00
            clr   integration_timer
            ld    pre1,#pre1_min                ; preset T1

            ld    duration_of_high_cycle,#start_up_voltage
                                                ; preset voltage level
;           ld    T1,duration_of_high_cycle
;           ld    duration_of_low_cycle,#no_voltage_levels
;           sub   duration_of_low_cycle,duration_of_high_cycle

            ei
            ld    TMR,#00011100b
            IF    SOFTSTART_WISHED
            call  softstart                         ; increase voltage slowly
            ENDIF

;*****************************************************************
;/////////////////////////////////////////////////////////////////
;       MAIN USER PROGRAM
;       The step down voltage regulator runs as a batch task via T1 interrupt.
;       This is the user program that runs in front.
;/////////////////////////////////////////////////////////////////
Main:
            NOP                     ; insert your instructions
                              ; here
            jp Main

;*****************************************************************
;-----------------------------------------------
;               SUBROUTINES
;-----------------------------------------------

            IF    SOFTSTART_WISHED
;////////////////////////////////////////////////
;       1.5uS x (30x222) = 10 mS
;////////////////////////////////////////////////

delay_counter     .set  %FE
```

```
delay10msec:
            ld    delay_counter,#222      ; 6 cycles
loop1:
            nop                                         ; 6 cycles
            nop                                         ; 6 cycles
            dec   delay_counter           ; 6 cycles
            jr    nz,loop1          ; 12 cycles
            ret

delay100msec:
            call  delay10msec
            call  delay10msec
            call  delay10msec
            call  delay10msec
            call  delay10msec
            call  delay10msec
            call  delay10msec
            call  delay10msec
            call  delay10msec
            call  delay10msec
            ret

            ENDIF

;////////////////////////////////////////////////////////////////////////////////
//////////////////
;               Increase voltage slowly after the start.
;               (extends light bulb life or speeds up
;          the engine slow = less start momentum)
;////////////////////////////////////////////////////////////////////////////////
//////////////////
            IF    SOFTSTART_WISHED
softstart_counter .set  r5

softstart:
            brset r3,Usense,end_softstart                   ; Is Uref reached yet?
            ld    softstart_counter,#inv_softstart_time
softstart_delay:
            call  delay100msec
            djnz  softstart_counter,softstart_delay
slow_down:
            dec   duration_of_high_cycle              ; No, then slow down
            ;ld   duration_of_low_cycle,#no_voltage_levels
            ;sub  duration_of_low_cycle,duration_of_high_cycle
            jr    softstart                     ; voltage rising.

end_softstart:
            ret
            ENDIF



;////////////////////////////////////////////////////////////////
;       Timer1 interrupt occurs on each edge of the period.
;       The timing depends on the power level = high/low ratio.
;////////////////////////////////////////////////////////////////
timer1:
            push    rp                          ; working register
            srp     #%00                                    ; group 0 reserved
                                                ; for timer1
                                                ; interrupt

            brclr   r2,pwm_output,turn_on                   ; If last pwm-cycle was
                                                ; low then next pwm-
                                                ; cycle is high
turn_off:
            pwm_low_cycle
            ld      T1,duration_of_high_cycle               ; On next T1 end_of_count,
```

```
                                                    ; low cycle is finished, and
                                                  ; duration_of_high_cycle
                                                  ; is loaded from T1 into
                                                  ; 8bit-down-counter
            jr      end_of_interrupt
turn_on:
            pwm_high_cycle
            ld      T1,duration_of_low_cycle              ; On next T1 end_of_count,
                                                ; high cycle is finished,
                                                  ; and duration_of_low_cycle
                                                  ; is loaded from T1 into 8bit-      ;
down-counter

;////////////////////////////////////////////////////////////////////////////////////
//////////////////
;       This is an integrating regulator routine for the voltage.
;       If Usense is below Uref then the voltage is increased
;       by increasing the duration of the pwm-high-cycle.
;       If Usense is above Uref then the voltage is decreased
;       by decreasing the duration of the pwm-high-cycle
;////////////////////////////////////////////////////////////////////////////////////
            djnz  integration_timer,end_of_interrupt
            brclr r3,Usense,increase_voltage
decrease_voltage:
            djnz  duration_of_high_cycle,cycle_adjust
increase_voltage:
            inc   duration_of_high_cycle
            cp    duration_of_high_cycle,#max_high_cycle
            jr    GT,decrease_voltage
cycle_adjust:
            ld    duration_of_low_cycle,#no_voltage_levels
            sub   duration_of_low_cycle,duration_of_high_cycle

end_of_interrupt:
            pop   rp
            iret

            END
```

Information Integrity:

The information contained within this document has been verified according to the general principles of electrical and mechanical engineering. Any applicable source code illustrated in the document was either written by an authorized ZiLOG employee or licensed consultant. Permission to use these codes in any form besides the intended application, must be approved through a license agreement between both parties. ZiLOG will not be responsible for any code(s) used beyond the intended application. Contact your local ZiLOG Sales Office to obtain necessary license agreements.

ZiLOG, Inc.
910 East Hamilton Avenue, Suite 110
Campbell, CA 95008
Telephone (408) 558-8500
FAX 408 558-8300
Internet: http://www.zilog.com