APPLICATION OPERATION

This design entry uses the PICmicro’s ability to be programmed in-circuit to create a large family of devices from one simple design. This will allow the cost savings of mass production while giving the sales leverage of customized components without long turn-around times or increased inventory requirements. Additionally, since the target market for this product is consumer and industrial equipment manufacturers, the agency approvals (UL, CSA, European, etc.) should be easier to get and maintain since there is only a few hardware prototypes for the many incarnations.

The devices that are being replaced either provide a time delay between the control signal coming on and the relay activating, or provide a delay between the control signal going off and the relay deactivating. The time delay is either a set value at manufacture or adjustable within a set range. This functionality is duplicated in this design using the PIC12C671, a standard 5 VDC relay, and (in the case of an adjustable delay) a potentiometer. In fact this design makes it possible to have separate delays for turn on and turn off.

As mentioned previously, there are many implementations of the software for one common hardware design. The source code included in this application note is based on the dual adjustable model since the other implementations are a subset of this one. The range for the adjustment is 0 – 31 seconds in this example.
APPENDIX A: SOURCE CODE

;***************************************************************************
;                          TM_DLY01.ASM
;              Variable time delay, on & off, relay program
;***************************************************************************
; AUTHOR:            Paul McCoy
; DATE:              31 May 1997
; ORGANIZATION:      Zykora Inc.
; REVISION:          1.00.00
;
;***************************************************************************
;
LIST    p=12C671; PIC12C671 is the target processor
#include <P12C671.INC>

CBLOCK 0x20                    ; autosets register variable addresses
flag                    ; flag register for program flags
second_cnt              ; counter for one second timer
temp_stat               ; holds status register for interrupts
temp_w                  ; holds W register for interrupts
time_cnt
ENDC

;***************************************************************************
;                         FLAG BITS
;***************************************************************************
COMMAND       equ      00        ; Set – input line is set
ACTUAL        equ      01        ; Set – output line is set
TIMING        equ      02        ; Set – Timer is running

;***************************************************************************
;          OTHER DEFINES
;***************************************************************************
LAST_LOCATE   equ      0x3ff

;***************************************************************************
;          Program
;***************************************************************************
org           0x00                           ;reset vector
goto          start
org 0x04     ;interrupt vector
movwf temp_w ;note: if Bank1 is used this space must be reserved
swapf STATUS,W
bcf STATUS,RP0
movwf temp_stat
btfsc INTCON,2
call Timer
btfsc INTCON,1
bcf INTCON,1
btfsc INTCON,0
call Input_change
swapf temp_stat,W
movwf STATUS
swapf temp_w,F
swapf temp_w,W
retfie

;***************************************************************************
;                 Initialization block
**electromechanical timer replacement**

```
start:
    clrf GPIO
    clrf flag
    bsf STATUS,RP0  ;switch to Bank 1
    movlw B'11000101'
    movwf OPTION_REG  ;tmr0 scaled 64:1
    movlw B'00000100'  ;GPIO 0 & 1 A/D; 3, 4, & 5 digital
    movwf ADCON1
    movlw B'11101111'  ;GPIO 5 output; all others input
    movwf TRISIO
    clrf PIE1
    call LAST_LOCATE
    movwf OSCCAL
    bcf STATUS,RP0  ;switch back to Bank 0
    movlw B'01000001'  ;select channel 0 and turn on A/D
    movwf ADCON0
    btfsc GPIO,3
    call Initial_set
    movlw -D'125'  ;count up 125 to rollover
    movwf TMR0
    movlw B'10101000'  ;enable timer interrupt
    movwf INTCON

;***********************************************************************************
;***************************************************************************
;                          Main program loop
;***************************************************************************

top:
    goto top

;***********************************************************************************
;                         Normally energized initialization routine
;***************************************************************************

Initial_set:
    bsf GPIO,3
    bsf flag,ACTUAL
    bsf flag,COMMAND
    bsf ADCON0,3
    return

;***********************************************************************************
;                          A/D Conversion Subroutine
;***************************************************************************

Adloop:
    bsf ADCON0,2
adwait:
    btfss PIR1,6
    goto adwait
    bcf PIR1,6
    return

;***********************************************************************************
;                          Timer interrupt routine
;***************************************************************************
```
Electromechanical Timer Replacement

Timer:

```assembly
bcf   INTCON,2
movlw  -D'125'   ; count up 125 to rollover
movwf  TMR0
decfsz second_cnt,F
return
movlw  D'125'   ; 125*125*64 = 1 million clocks
                    ; equals one second
movwf  second_cnt
decfsz time_cnt,F
return
btfss  flag,TIMING
return
bcf   flag,TIMING
btfsc  flag,COMMAND
goto  set_out
bcf   GPIO,5
bcf   flag,ACTUAL
return

set_out:
bsf   GPIO,5
bsf   flag,ACTUAL
return
```

;*******************************************************************************
;                         Input interrupt routine
;*******************************************************************************

Input_change:

```assembly
bcf   INTCON,0
btfss  GPIO,3
goto  input_off
btfsr  flag,COMMAND
return
bsf   flag,COMMAND
btfsr  flag,ACTUAL
goto  error_on
call  Adloop
bsf   ADCON0,3
rrf   ADRES,F
rrf   ADRES,F
rrf   ADRES,F
movf  ADRES,W
btfsr  STATUS,Z
goto  timed_on
movwf  time_cnt
movlw  -D'125'
movwf  TMR0
bcf   INTCON,2
movlw  D'125'
movwf  second_cnt
bsf   flag,TIMING
return

error_on:
bcf   flag,TIMING
return

timed_on:
bsf   GPIO,5
bsf   flag,ACTUAL
return

input_off:
btfss  flag,COMMAND
return
btfss  flag,ACTUAL
```
call        Adloop
bcf         ADCON0,3
rrf         ADRES,F
rrf         ADRES,F
rrf         ADRES,F
movf        ADRES,W
btfsc       STATUS,2
goto        timed_off
movwf        time_cnt
movlw       -D'125'
movwf        TMR0
bcf         INTCON,2
movlw       D'125'
movwf        second_cnt
bsf         flag,TIMING
return
error_off:
bcf         flag,TIMING
return
timed_off:
bcf        GPIO,5
bcf        flag,ACTUAL
;
Information contained in this publication regarding device applications and the like is intended for suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip’s products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. in the U.S.A. and other countries. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.