Consumer Appliance, Widget, Gadget

Random Phase Power Controller

Author: *Borislav Goudjounov*
Sofia, Bulgaria
email: astro@mail.techo-link.com

INTRODUCTION:

This application note describes a method for implementing a random phase power controller using the PIC12C5XX series of 8-pin, 8-bit microcontrollers. The devise requires two-digit BCD encoder, a shift register (parallel in, serial out), two optoisolators and a TRIAC.

APPLICATION OPERATION:

The device operates in two modes – as independent power controller, or slave device on I^2C bus. The mode of operation is defined automatically - just connect SDA and SCL physically with bus wires and devise is under control to I^2C bus. When the devise is an independent power controller, the value of regulation (in percents) is read from BCD encoder. In I^2C bus connecting, value from the encoder is used as a device address. Conversion to the value from encoders, to random phase angle (a timer constant) is realized with a table. Because of this, the I^2C bus protocol is emulated by software in the device is used 'Low speed mode'. In this mode, transfer rate is about 2 KHz. The transfer protocol is different as follows:

- A START condition S
- S start byte 00000001
- A 'dummy' acknowledge clock pulse
- A repeated START condition Sr.

After this procedure follows: slave address, RW bit, and max 5 bytes of data. The schematic is shown in Figure 1.

The resistor R1 connected to AC line, the Bridge and the Opto MOC8021 generate a syncro pulse in every zero crossing of the AC frequency. The time between two pulses is 10mS at 50 Hz and 8.3ms at 60 Hz.

The Opto MOC3021 is especially designed to control triacs. If the output 4 to PIC12C5XX is in 'H' state, the MOC3021 opens the TRIAC.

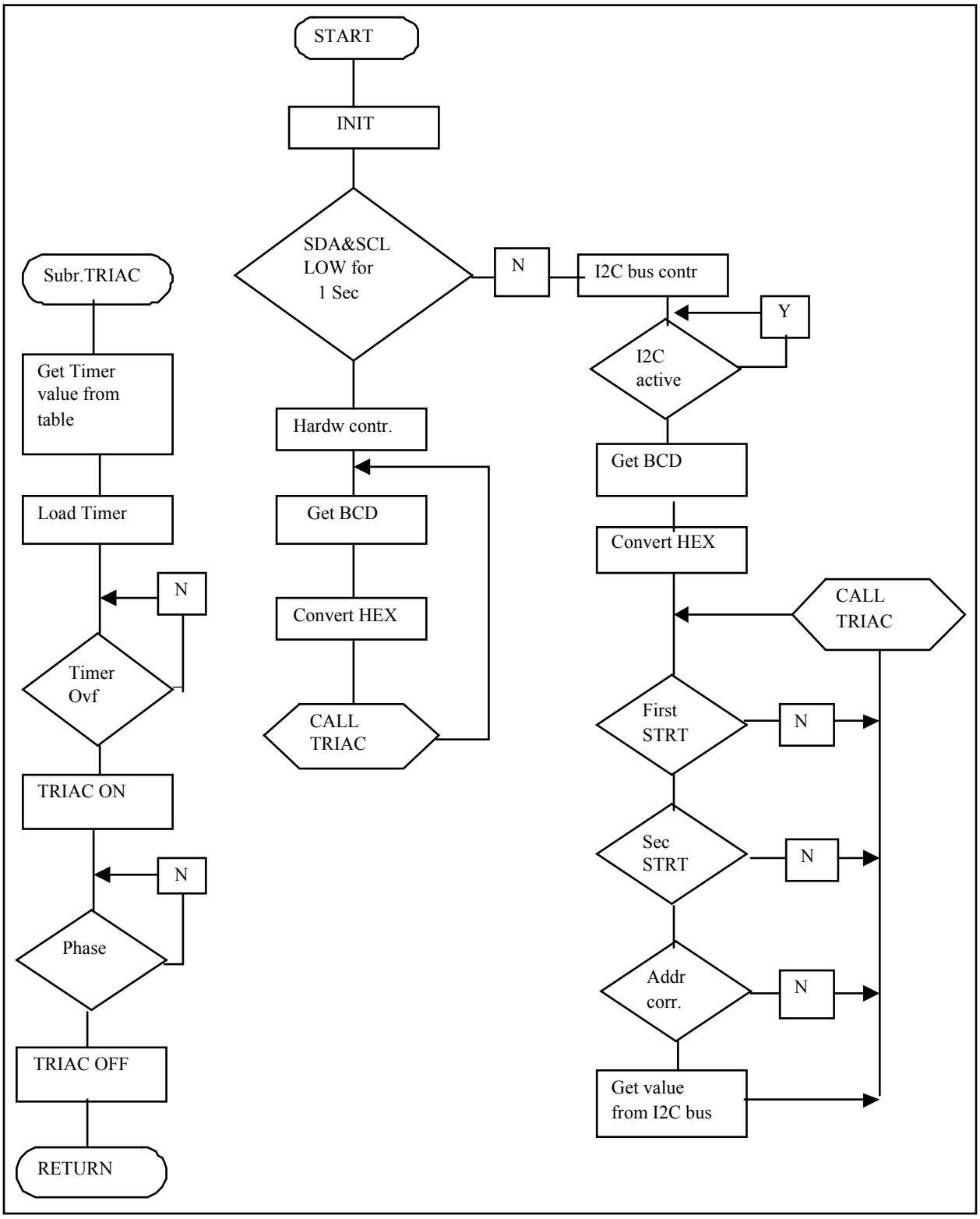
Shift register loads serially the two BCD digits which are set in encoders.

When the inputs SCL and SDA are not connected, its logical condition is 'L', because they are pulldown to GND via resistors R5 and R6. This fact is the criteria to connection with the bus. In the I^2C, bus specification SCL and SDA can't be together low, for a long time.

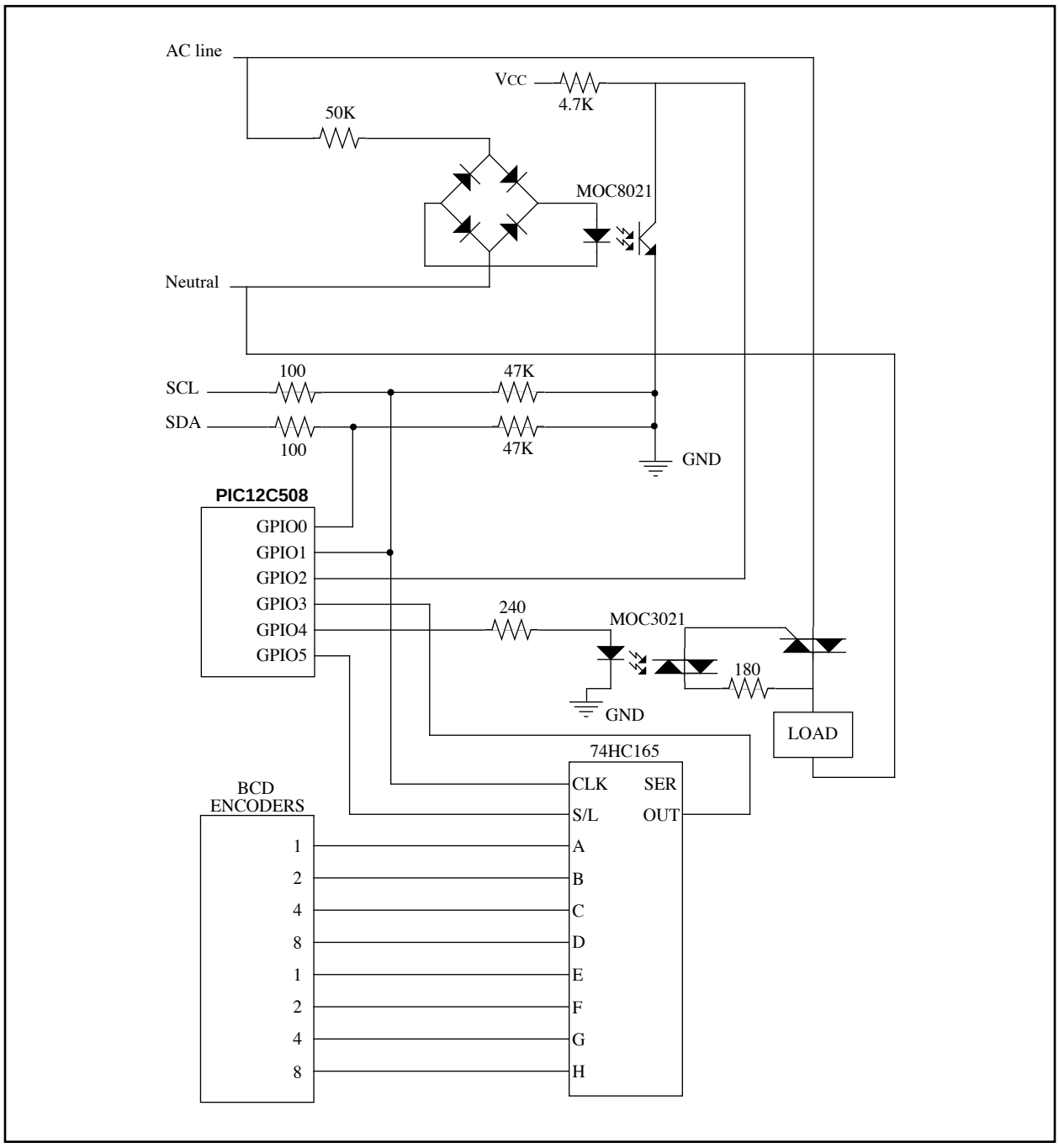
The resistors R3 and R4 prevent I/O pins of unlimited current.

Power supply is not shown.

Flow Chart: (Note: The structure of program code is a little different from flow chart, due to it being necessary to organize over page CALL's).



Graphical hardware representation:



APPENDIX A: SOURCE CODE

```
                    TITLE include file p12c509a.inc
;
            list     p=12c509, r=hex
;==========================================================================
;
;           Verify Processor
;
;==========================================================================

            IFNDEF __12C509
              MESSG "Processor-header file mismatch.  Verify selected processor."
            ENDIF

;==========================================================================
;
;           Register Definitions
;
;==========================================================================
;
indf        equ      0          ;indirect addresing
tmr0        equ      1          ;8-bit timer/counter
pcl         equ      2          ;plogram counter low
;==========================================================================
;
;                                status register
;
;==========================================================================
status      equ      3          ;status register
;==========================================================================
;                                status bits
;==========================================================================
gpwuf       equ      7          ;bit7 1 reset from wake-up from sleep, 0 other
pa0         equ      5          ;bit5 0=page0, 1=page1
to          equ      4          ;bit4 timeout 1=after power up, 0=WDT timeout
pd          equ      3          ;bit3 power down 1=after pow. up, 0=exec. SLEEP
z           equ      2          ;bit2 zero flag
dc          equ      1          ;bit1 half carry flag
c           equ      0          ;bit0 carry flag
;==========================================================================
fsr         equ      4          ;indirect address pointer
osccal      equ      5          ;oscilator calibrate register bits 7..4
;==========================================================================
;                                general purpose input/output tegister
;==========================================================================
gpio        equ      6          ;general purpose register pa3 - input only
;==========================================================================
;                                gpio bits
;==========================================================================
sda         equ      0          ;pa0 - serial data
scl         equ      1          ;pa1 - serial clock
fin         equ      2          ;pa2 - phase pulse
shin        equ      3          ;pa3 - shift data from register
fout        equ      4          ;pa4 - triac controll
shlo        equ      5          ;pa5 - shift/load to register
;==========================================================================
;                                option register bits
;==========================================================================
gpwu        equ      7          ;bit7 ena=0, dis=1 wake-up on pin change pa0, pa1, pa3
gppu        equ      6          ;bit6 ena=0, dis=1 weak pull-ups
tocs        equ      5          ;bit5 timer clock 1=t0ckl, 0=fosc/4
tose        equ      4          ;bit4 edge of t0ckl 0=L to H, 1=H to L
psa         equ      3          ;bit3 prescaler 0=timer0, 1=WDT
ps2         equ      2          ;bits 2-0 prescaler rate
ps1         equ      1          ;for timer0 000..fff -> 1/2..1/256
```

```
ps0         equ       0          ;for WDT    000..fff -> 1/1..1/128
;==========================================================================
f           equ       1
w           equ       0
;==========================================================================
;
;           RAM Definition
;
;==========================================================================

            __maxram 0x3f

;==========================================================================
;
;           Configuration Bits
;
;==========================================================================

_mclre_on   equ       0x0fff
_mclre_off  equ       0x0fef
_cp_on      equ       0x0ff7
_cp_off     equ       0x0fff
_wdt_on     equ       0x0fff
_wdt_off    equ       0x0ffb
_lp_osc     equ       0x0ffc
_xt_osc     equ       0x0ffd
_intrc_osc  equ       0x0ffe
_extrc_osc  equ       0x0fff
;
               Main program


            TITLE 'RANDOM PHASE POWER CONTROLLER'
;-----------------------------------------------
;           program:  PICPOW.ASM
;           author:   Borislav Goudjounov
;           company:  AstroFORCE
;           date:     09-20-1997
;-----------------------------------------------
#include <p12c509a.inc>
;
line        set       50         ;set line 50 Hz, else 60 Hz
;                    *** user ram
del1sc      equ       0x7
flag        equ       0x8
fase        equ       0          ;bit2=1 if triac is 'fire'
temp        equ       0x9
temp1       equ       0xa
bcount      equ       0xb
vreg        equ       0xc
bitcou      equ       0xd
bitbuf      equ       0xe
timout      equ       0xf
;
jsr         macro     address
            bsf       status,pa0
            call      address
            bcf       status,pa0
            endm
;
            org       0
;
init
            movwf     osccal
            movlw     0x0f       ;pa0=i, pa1=i, pa2=i, pa3=i, pa4=o,pa5=o
            tris      6          ;init gpio
```

```
           movwf    gpio       ;set sda=1, scl=1, phase=0, shift/load=0
           movlw    b'11000101' ;set prescaler rate 1/64
           option
           clrf     flag       ;clear all flags
           clrf      vreg      ;clear value register
;
;-----------------------------------------------------------------------
;          Test for the connection with i2c-bus. The criterion is:
;          (scl=L)&(sda=L) for about one second.
;-----------------------------------------------------------------------
main
           movlw    0x40
           movwf    del1sc     ;set timer counter=64
m01        movlw    1
           movwf    tmr0       ;load timer=1
m02        movf     gpio,w     ;test scl=0, sda=0
           andlw    3          ;mask bits 2..5
           btfss    status,z   ;result=zero?
           goto     i2c        ;no, goto i2c routine
           movf     tmr0,f     ;test for timer oveflow
           btfss    status,z   ;timer=zero?
           goto     m02        ;no, repeat
           decfsz   del1sc,f   ;decrement timer counter
           goto     m01        ;test again
           movlw    0x0d       ;set scl output
           option
           bcf      gpio,scl   ;set scl = 'L'
;
;-------------------------------------------------------
;          'Hardware' control routine
;          triac is controled by encoder's value
;-------------------------------------------------------
;
           bcf      gpio,fout  ;turn triac off
loopag     call     shifd      ;get bcd value from encoders
           call     xcdb       ;convert bcd to hex
           jsr      triac      ;call user subroutine
           goto     loopag     ;again
;-------------------------------------------------------
;          Subroutine to get value from encoders
;-------------------------------------------------------
shifd      movlw    8          ;set bit counter value
           movwf    bcount     ;load counter
           bsf      gpio,shlo  ;set 'shift/load' to shift
shloop     movf     gpio,w     ;load w from gpio
           movwf    temp1      ;story w in temporary
           rlf      temp1,f    ;shift left
           rlf      temp1,f    ;shift left
           rlf      temp1,f    ;shift left
           rlf      temp1,f    ;shift left, 'shin'=carry
           rrf      temp,f     ;bit n -> result reg.
           bcf      gpio,scl   ;clear clock
           bsf      gpio,scl   ;set clock
           bcf      gpio,scl   ;clear clock
           decfsz   bcount,f   ;decrement bit counter
           goto     shloop     ;continue loop
           bcf      gpio,shlo  ;set 'shift/load' to load
           retlw    0
;-------------------------------------------------------
;          Subroutine to convert bcd to hex
;-------------------------------------------------------
xcdb
           movlw    0xa
           movwf    bcount     ;set counter
           swapf    temp,w     ;get upper nibble
           andlw    0xf        ;clear lower
```

```
           clrf     vreg        ;clear register
bcdl       addwf    vreg,f      ;add with reg. 10 times
           decfsz   bcount,f    ;decrement counter
           goto     bcdl
           movf     temp,w      ;get lower nibble
           andlw    0xf         ;clear upper
           addwf    vreg,f      ;add with register (10*upp.+low.)
           retlw    0           ;return
;
;
;-------------------------------------------------------
;          Subroutine to control triac via i2c bus
;-------------------------------------------------------
i2c
           movlw    0xff-15     ;set timer to count 16 times
           movf     tmr0,f
i2cl1      btfss    gpio,scl    ;scl is still 'H'?
           goto     i2c         ;i2c is active
           movf     tmr0,f      ;test for time = 2 scl's
           btfss    status,z    ;if time is less, wait
           goto     i2cl1
;
           movlw    0x0d        ;set scl output
           tris     6           ;init gpio
           bcf      gpio,scl    ;clear scl
           movlw    0x0f        ;pa0=i, pa1=i, pa2=i, pa3=i, pa4=o,pa5=o
           tris     6           ;init gpio
           call     xcdb        ;convert addr. to hex
           movf     vreg,w      ;get value
           movwf    temp        ;move to temp
           clrf     vreg        ;clear vreg
;
idstart
           jsr      triac       ;call user subroutine
           btfss    gpio,sda    ;wait sda become 'H'
           goto     idstart     ;no, repeat
id1
           jsr      triac       ;call user subroutine
           btfsc    gpio,sda    ;wait sda become 'L'
           goto     id1         ;no, repeat
           btfss    gpio,scl    ;scl is 'H' when sda trans from 'H' to 'L
           goto     idstart     ;no, contonue
;
bytein     movlw    8
           movwf    bitcou      ;set bit counter
clrise
           jsr      triac       ;call user subroutine
           btfss    gpio,scl    ;scl = 1?
           goto     clrise      ;no, repeat
           rrf      gpio,w      ;rotate gpio, bit -> carry
           rlf      bitbuf,f    ;rotate bitbuf, carry -> bit
clfall
           jsr      triac       ;call user subroutine
           btfsc    gpio,scl    ;scl = 0?
           goto     clfall      ;no, rpeat
           decfsz   bitcou,f    ;8 bit receive?
           goto     clrise      ;no, repeat
           movlw    0x01        ;get 0x01
           subwf    bitbuf,w    ;compare with start byte
           btfss    status,z    ;start byte = 0x01 ?
           goto     idstart     ;no, ignore
;
sstart
           goto     sstart      ;no, repeat
           btfss    gpio,scl    ;second start occur
           goto     idstart     ;no low speed transmition
```

```
;
bytein1    movlw    8
           movwf    bitcou     ;set bit counter
clrise1
           jsr      triac      ;call user subroutine
           btfsc    gpio,sda   ;wait for sda become 'L'
           btfss    gpio,scl   ;scl = 1?
           goto     clrise1    ;no, repeat
           rrf      gpio,w     ;rotate gpio, bit -> carry
           rlf      bitbuf,f   ;rotate bitbuf, carry -> bit
clfall1
           jsr      triac      ;call user subroutine
           btfsc    gpio,sda   ;wait for sda become 'L'
           btfsc    gpio,scl   ;scl = 0?
           goto     clfall1    ;no, repeat
           decfsz   bitcou,f   ;8 bit receive?
           goto     clrise1    ;no, repeat
;
           bcf      status,c   ;clear carry
           rrf      bitbuf,f   ;rw bit -> carry
           btfsc    status,c   ;read is not aveliable
           goto     idstart    ;ignore transmition
           movf     bitbuf,w   ;get address
           subwf    vreg,w     ;compare address
           btfss    status,z   ;if not the same, ignore
           goto     idstart    ;no, ignore
;
sack
           movlw    0x0e       ;set sda output
           tris     6          ;init gpio
           bcf      gpio,sda   ;send ack
sack1
           jsr      triac      ;call user subroutine
           btfsc    gpio,sda   ;wait for sda become 'L'
           btfsc    gpio,sda   ;wait for sda become 'L'
           btfss    gpio,scl   ;scl 'H'
           goto     sack1      ;no, wait
sack2
           jsr      triac      ;call user subroutine
           btfsc    gpio,scl   ;scl 'L'
           goto     sack2      ;no, wait
           nop
           nop
           nop
           nop
           nop                 ;adjust thd:dat = 5uS
           bsf      gpio,sda   ;resume sda
           movlw    0x0f       ;pa0=i, pa1=i, pa2=i, pa3=i, pa4=o,pa5=o
           tris     6          ;init gpio
;
bytein2    movlw    8
           movwf    bitcou     ;set bit counter
clrise2
           jsr      triac      ;call user subroutine
           btfss    gpio,scl   ;scl = 1?
           goto     clrise2    ;no, repeat
           rrf      gpio,w     ;rotate gpio, bit -> carry
           rlf      bitbuf,f   ;rotate bitbuf, carry -> bit
clfall2
           jsr      triac      ;call user subroutine
           btfsc    gpio,scl   ;scl = 0?
           goto     clfall2    ;no, repeat
           decfsz   bitcou,f   ;8 bit receive?
           goto     clrise2    ;no, repeat
;
           movlw    0x0e       ;set sda output
```

```
            tris     6           ;init gpio
            bcf      gpio,sda    ;send ack
sack3
            jsr      triac       ;call user subroutine
            btfss    gpio,scl    ;scl 'H'
            goto     sack3       ;no, wait
sack4
            jsr      triac       ;call user subroutine
            btfsc    gpio,scl    ;scl 'L'
            goto     sack4       ;no, wait
            nop
            nop
            nop
            nop
            nop                  ;adjust thd:dat = 5uS
            bsf      gpio,sda    ;resume sda
            movlw    0x0f        ;pa0=i, pa1=i, pa2=i, pa3=i, pa4=o,pa5=o
            tris     6           ;init gpio
;
            movf     bitbuf,w    ;get value
            movwf    vreg        ;put in vreg
;
endcon
            movlw    0x28        ;set counter with 40
            movwf    timout      ;load counter
ec1
            jsr      triac       ;call user subroutine
            btfsc    gpio,sda    ;sda = 'L'
            goto     ec1         ;no, repeat
ec2
            jsr      triac       ;call user subroutine
            btfss    gpio,sda    ;sda = 'H'
            goto     ec2         ;no, repeat
            btfsc    gpio,scl    ;scl = 'H' when sda = 'L' to 'H'
            goto     idstart     ;stop condition occur
ec3
            jsr      triac       ;call user subroutine
            btfss    gpio,scl    ;wait scl become 'H'
            goto     ec3
ec4
            jsr      triac       ;call user subroutine
            btfsc    gpio,scl    ;wait scl become 'L'
            goto     ec4
            decfsz   timout,f    ;40 scl pass? (> 5 bytes)
            goto     ec1         ;no, repeat
            goto     idstart     ;timeout to stop cond. ignore
;
triac1      bsf      status,pa0  ;set page 1
            goto     triac       ;go to triac
triac2      retlw    0           ;return
;
            org      0x200
;
;-------------------------------------------------------------
;           Subroutine to drive triac
;-------------------------------------------------------------
triac
            btfss    flag,fase   ;test 'fire' flag
            goto     fin1        ;skip start phase routine
            btfsc    gpio,fin    ;test for phase = 0
            retlw    0           ;no, return
            bcf      gpio,fout   ;turn off triac
            movf     vreg,w      ;get hex value
            call     gotable     ;get timer value from table
            movwf    tmr0        ;load timer
            bcf      flag,fase   ;clear 'fire' flag
```

```
            retlw    0          ;return
;
fin1        btfss    gpio,fin   ;test for phase = 1
            retlw    0          ;no return
            movf     tmr0,f     ;test timer
            btfss    status,z   ;timer overflow
            retlw    0          ;no, return
            bsf      gpio,fout  ;'fire' triac
            bsf      flag,fase  ;set 'fire' flag
            bcf      status,pa0 ;restore page 0
            retlw    0          ;return
;
;-----------------------------------------------------
;           Table to convert prcents to timer value
;-----------------------------------------------------
gotable     addwf    pcl,f
            radix    dec
            if       line == 50
;-----------------------------------------------
;           Table for 50 Hz AC line
;-----------------------------------------------
table       dt       0,2,3,5,6,8,9,10,12,14
            dt       16,17,19,20,22,23,25,27,28,30
            dt       31,33,34,36,37,39,41,42,43,45
            dt       47,48,50,51,53,55,56,58,59,60
            dt       62,64,66,67,69,70,72,73,75,76
            dt       78,79,81,83,84,86,87,89,90,92
            dt       94, 95, 97, 98,100,101,103,105,106,108
            dt       109,111,112,114,115,117,119,120,122,123
            dt       125,126,128,129,131,133,134,136,137,139
            dt       140,142,144,145,147,148,150,151,153,155
            else
;-----------------------------------------------------
;           Table for 60 Hz AC line
;-----------------------------------------------------
table       dt       0,1,3,4,5,7,8,9,10,12
            dt       13,14,16,17,18,19,21,22,23,25
            dt       26,27,29,30,31,32,34,35,36,38
            dt       39,40,42,43,44,45,47,48,49,50
            dt       52,53,55,56,57,58,60,61,62,64
            dt       65,66,68,69,70,71,73,74,75,77
            dt       78,79,80,82,83,84,86,87,88,90
            dt       91,92,94,95,96,97,99,100,101,103
            dt       104,105,107,108,109,110,112,113,114,116
            dt       117,118,120,121,122,123,125,126,127,129
            endif
;
            end
```