code MCU interface

```
'*****************************************************************
'*  Name    : nRF24L01 addressing test                           *
'*  Author  : AVC                                                *
'*  Notice  : Copyright (c) 2008 AVC                             *
'*          : All Rights Reserved                                *
'*  Date    : 27/10/2008                                         *
'*  Version : 1.0                                                *
'*  Notes   : 5 bytes pload, 3x re-transmit, 2Mbps, 0dbm output  *
'*          : pipe0 on channel 40                                *
'*****************************************************************
'-------------------------------------------------------------------------------
' PIC16F818 SAMPLE TEST ROUTINE
'-------------------------------------------------------------------------------
' PIN  NAME         USE/CONNECTION
'  1   RA2 (AN2)    N/C
'  2   RA3 (AN3)    N/C
'  3   RA4 (AN4)    N/C
'  4   RA5 (/MCLR)  N/C
'  5   VSS          GND
'  6   RB0 (INT)    led
'  7   RB1          SDI
'  8   RB2          SDO
'  9   RB3          CE
' 10   RB4          SCK
' 11   RB5          CSN
' 12   RB6          Irq
' 13   RB7          ser_lcd
' 14   VDD          5V
' 15   RA6 (ST)     N/C
' 16   RA7 (ST)     N/C
' 17   RA0 (AN0)    N/C
' 18   RA1 (AN1)    N/C
'===============================================================================

'-----------config (PM-assembler)-----------------------------------------------
@ DEVICE PIC16F818,MCLR_OFF,INTRC_OSC_NOCLKOUT,WDT_ON,BOD_ON
@ DEVICE PROTECT_OFF,CPD_OFF

'-----------init-----------------------------------------------------------------
TRISA = %00000000   ' Set PORTA to all output
TRISB = %01000010   ' Set PORTB SDI en IRQ input, SDO en SCK output
ADCON1 = 7          ' Set PORTA and PORTB to digital
OSCCON= $70         ' Internal 8MHz osc
define  OSC 8       ' osc op 8 MHz
'-----------variables & settings-------------------------------------------------
SSPEN      VAR      SSPCON.5     'SSP Enable bit
CKP        VAR      SSPCON.4     'Clock Polarity Select
SMP        VAR      SSPSTAT.7    'Data input sample phase
CKE        VAR      SSPSTAT.6    'Clock Edge Select bit
SSPIF      VAR      PIR1.3       'SPI interrupt flag
ser_lcd    VAR     PORTB.7     'LCD output
Ce         var     PORTB.3     'CE pin nRF24L01
CSN        VAR     PORTB.5     'CSN pin nRF24L01
Irq        VAR     PORTB.6     'IRQ pin nRF24L01
led        var     PORTB.0     'debug led
RXD        var     PORTA.6     'data to PC
TXD        var     PORTA.7     'data from PC
i          VAR     BYTE         'loop counter
num_byte   var     byte        'temp value
```

```
val         var     byte        'temp value
Pack_count  VAR     word        'number of packets asked
counter     var     word        'number of packets received
dat         var     word        'unscaled ADC data
timer_0     var     word        'timer_0
timer_1     var     word        'timer_1
data_out    VAR     BYTE[6]     'data sent
data_in     var     Byte[6]     'data received
sample_per  var     word        'sample period
num_sample  var     word        'number of samples
P9600       CON     84          '9600 baud true
P19200      con     32          '19200 baud true
'----------nRF24L01 interrupt flags-----------------------------------------
Idle_int    con      $00        'Idle no interrupt pending
Max_rt      con      $10        'Max # of Tx retrans interrupt
Tx_inter    con      $30        'Tx interrupted
Rx_ds       con      $40        'Rx data received
'----------SPI(nRF24L01) commands-------------------------------------------
Read_reg    con      $00        'def read command to register
Write_reg   CON      $20        'def write command to register
Rd_rx_pload con      $61        'def Rx payload register address
Wr_tx_pload con      $A0        'def Tx payload register address
Flush_tx    con      $E1        'def flush Tx register command
Flush_rx    con      $E2        'def flush Rx register command
Reuse_tx_pl con      $E3        'def reuse Tx payload register command
Nop_comm    con      $FF        'def No operation
'----------SPI(nRF24L01) registers addresses---------------------------------
Config_nrf  con      $00        'Config register address
En_aa       con      $01        'enable auto acknowledgment register address
En_rxaddr   con      $02        'enable RX addresses register address
Setup_aw    con      $03        'setup address width register address
Setup_retr  con      $04        'setup auto retrans register address
Rf_ch       con      $05        'RF channel register address
Rf_setup    con      $06        'RF setup register address
Stat_us     con      $07        'Status register address
Observe_tx  con      $08        'Observe TX register address
Cd          con      $09        'Carrier detect register address
Rx_addr_p0  con      $0A        'RX address pipe0 register address
Rx_addr_p1  con      $0B        'RX address pipe1 register address
Rx_addr_p2  con      $0C        'RX address pipe2 register address
Rx_addr_p3  con      $0D        'RX address pipe3 register address
Rx_addr_p4  con      $0E        'RX address pipe4 register address
Rx_addr_p5  con      $0F        'RX address pipe5 register address
Tx_addr     con      $10        'TX address register address
Rx_pw_p0    con      $11        'RX payload width pipe0 register address
Rx_pw_p1    con      $12        'RX payload width pipe1 register address
Rx_pw_p2    con      $13        'RX payload width pipe2 register address
Rx_pw_p3    con      $14        'RX payload width pipe3 register address
Rx_pw_p4    con      $15        'RX payload width pipe4 register address
Rx_pw_p5    con      $16        'RX payload width pipe5 register address
Fifo_status con      $17        'FIFO status register register address
'----------SPI settings-----------------------------------------------------
SSPEN = 1                   'enable SPI pins
SSPCON.0=1                  'SPI rate=OSC/16
CKP = 0                     'clock idle low
CKE = 1                     'transmit on active to idle
SSPIF = 0                   'clear buffer full status
SMP = 0                     'sample in middle of data
'--------------SPI init---------------------------------------------------
Ce=1                            'init spi pins
pause 10                        'wait 10 ms hardware is stable
Ce=0                            'set CE pin low disable Rx
CSN=0                           'set CSN pin low
```

```
Pack_count=0                    'number of packets sent
sample_per=500
num_sample=20
'--------------Common RX-TX settings-------------------------------------------
    data_out[0]=Write_reg+Rx_addr_p0 'Rx address for pipe0
    data_out[1]="N"                 '5 byte address
    data_out[2]="E"
    data_out[3]="T"
    data_out[4]="1"
    data_out[5]="4"
    num_byte=5
    gosub spi_write
    data_out[0]=Write_reg+En_aa      'enable auto ACK pipe0
    data_out[1]=$01                 '1 enable, 0 disable
    num_byte=1
    gosub spi_write
    data_out[0]=Write_reg+En_rxaddr  'enable Rx address pipe0
    data_out[1]=$01
    num_byte=1
    gosub spi_write
    data_out[0]=Write_reg+Rf_ch       'Set RF channel
    data_out[1]=40                    'number of channel used
    num_byte=1
    gosub spi_write
    data_out[0]=Write_reg+Rx_pw_p0    'Set Rx pload width pipe0
    data_out[1]=5                     'number of bytes used in data sent
    num_byte=1
    gosub spi_write
    data_out[0]=Write_reg+Rf_setup    'Set RF: Odbm, 2Mbps
    data_out[1]=$0F
    num_byte=1
    gosub spi_write

'    data_out[0]=Config_nrf            'debug
'    num_byte=1
'    gosub spi_read
'    gosub disp_lcd_hex
'    pause 1000

'============Main Tx=============================================================
Main_tx:
    gosub pc_out                      'input from pc
    data_out[0]=Flush_tx              'flush TX_fifo buffer
    num_byte=0
    gosub spi_write
    data_out[0]=Write_reg+Stat_us     'reset IRQ bits
    data_out[1]=%00110000
    num_byte=1
    gosub spi_write
    gosub setup_tx                    'setup Tx
    data_out[0]=Wr_tx_pload           'put 5 bytes data in Tx pload buffer
    data_out[1]="R"                   'hex:$52
    data_out[2]=sample_per.HIGHBYTE   'sample period
    data_out[3]=sample_per.LOWBYTE
    data_out[4]=num_sample.HIGHBYTE   'number of samples
    data_out[5]=num_sample.LOWBYTE
    num_byte=5
    gosub spi_write
    pauseus 500                       'pause 500 us
    Ce=1                              'CE=1 (toggle) transmit (TX) FIFO buffer
    pauseus 500                       'pause 500 us
    Ce=0
    pause 1                           'pause 1ms
```

```
    trans_irq:
        If Irq !=0 then trans_irq     'wait until IRQ, active low
    data_out[0]=Stat_us               'read status register
    num_byte=1
    gosub spi_read
'    gosub disp_lcd_hex
    val=data_in[0]&%01110000          'mask the IRQ bits STATUS byte
    if (val = Max_rt) then max_retry  'maximum TX retries
    if (val = Tx_inter) then tx_int   'Tx interrupted
    high led
    data_out[0]=Write_reg+Stat_us
    data_out[1]=%00100000             'clear TX_DS IRQ bit
    num_byte=1
    gosub spi_write

'============Main Rx=============================================================
Main_rx:
    Ce=0
    gosub  setup_rx                       'setup Rx
    pause 2                               'delay for Rx starting
    Ce=1                                  'set nRF24L01 in Rx mode
    for Pack_count=1 to num_sample
    timer_0=0
    timer_1=0
    receive:
        timer_0=timer_0+1                 'wait until IRQ & soft timeout
        if timer_0=0 then timer_1=timer_1+1
        If (Irq !=0) and (timer_1=2) then rx_int
        IF (Irq !=0) then receive
    '    high led                          'debug led
        Ce=0
        loop:
            data_out[0]=rd_rx_pload       'Read 5 bytes Rx pload
            num_byte=5
            gosub spi_read
            dat.HIGHBYTE=data_in[2]
            dat.LOWBYTE=data_in[3]
            counter.HIGHBYTE=data_in[4]
            counter.LOWBYTE=data_in[5]
'            gosub disp_lcd
            gosub pc_in
            data_out[0]=Fifo_status       'Read FIFO status
            num_byte=1
            gosub spi_read
            val= data_in[1]               'FIFO status register
            if val.0=0 then loop          'test RX_EMPTY=1, RX_FIFO empty
        data_out[0]=Write_reg+Stat_us     'reset RX_DR status bit
        data_out[1]=%01000000             'write 1 tp RX_DR to reset IRQ
        num_byte=1
        gosub spi_write
        Ce=1                              're-enable receiver
    next Pack_count
    Ce=0
    goto Main_tx
end

'============Subroutines=========================================================
disp_lcd_hex:
    SEROUT2 ser_lcd,P9600,[22,12]
    PAUSE 5
    SEROUT2 ser_lcd,P9600,[hex2 data_in[0]," ", hex2 data_in[1]," ", hex2
data_in[2],13]
```

```
    SEROUT2 ser_lcd,P9600,[hex2 data_in[3]," ", hex2 data_in[4]," ", hex2
data_in[5],13]
    PAUSE 5
    RETURN

disp_lcd:
    SEROUT2 ser_lcd,P9600,[22,12]
    PAUSE 5
    SEROUT2 ser_lcd,P9600,[data_in[1],13]
    SEROUT2 ser_lcd,P9600,[DEC4 dat," ",dec4 counter,13]
    PAUSE 5
    RETURN

pc_out:
    serin2 TXD,P19200,[DEC5 sample_per,dec5 num_sample]
    return

pc_in:
'    SEROUT2 RXD,P9600,[data_in[1]," ",DEC4 dat," ",dec5 counter,13]
    serout2 RXD,P19200,[data_in[1],hex4 dat,hex4 counter,13]
    return

spi_write:
    CSN=0
    For i = 0 to num_byte          'loop for # byte
              SSPBUF=data_out[i]   'send array variable
              GoSub buffer_ok      'wait until buffer ready
       Next i                      'next location
       CSN=1
    return

spi_read:
    CSN=0
       For i = 0 to num_byte           'loop for # byte
             SSPBUF = data_out[0]      'write to SSPBUF to start clock
             GoSub buffer_ok           'wait for receipt
             data_in[i] = SSPBUF       'store received character in array
       Next I                          'get next byte
    CSN=1
       Return

buffer_ok:
       IF SSPIF = 0 Then buffer_ok            'wait for SPI interupt flag
       SSPIF = 0                              'reset flag
       Return

max_retry:
'    gosub disp_lcd_hex
    data_out[0]=Flush_tx             'flush TX buffer
    num_byte=0
    gosub spi_write
    data_out[0]=Write_reg+Stat_us
    data_out[1]=%00010000            'clear MAX_RT IRQ bit
    num_byte=1
    gosub spi_write
    data_in[1]="E"
    gosub pc_in
    goto Main_tx

tx_int:
'    gosub disp_lcd_hex
    data_out[0]=Flush_tx             'flush TX buffer
    num_byte=0
```

```
    gosub spi_write
    data_out[0]=Write_reg+Stat_us
    data_out[1]=%00110000             'clear TX_DS & MAX_RT IRQ bit
    num_byte=1
    gosub spi_write
    data_in[1]="T"
    gosub pc_in
    goto Main_tx

rx_int:
'    gosub disp_lcd_hex
    Ce=0                              'set CE pin low disable Rx
    data_out[0]=Flush_rx              'flush RX buffer
    num_byte=0
    gosub spi_write
    data_out[0]=Write_reg+Stat_us
    data_out[1]=%01110000             'clear RX_DR, TX_DS & MAX_RT IRQ bit
    num_byte=1
    gosub spi_write
    data_in[1]="R"
    gosub pc_in
    goto Main_tx

setup_rx:
    data_out[0]=Write_reg+Config_nrf  'Config:PRX=1,PWR_UP=1, CRC=2, enabled
    data_out[1]=$0F
    num_byte=1
    gosub spi_write
    return

setup_tx:
    data_out[0]=Write_reg+Tx_addr     'Tx address
    data_out[1]="N"                   '5 byte address
    data_out[2]="E"
    data_out[3]="T"
    data_out[4]="1"
    data_out[5]="4"
    num_byte=5
    gosub spi_write
    data_out[0]= Write_reg+Setup_retr 'Set retransmit @ ACK
    data_out[1]= 3                    '3x re-transmit (default)
    num_byte=1
    Gosub spi_write
    data_out[0]=Write_reg+Config_nrf  'Config:PRX=0,PWR_UP=1, CRC=2, enabled
    data_out[1]=$0E
    num_byte=1
    gosub spi_write
    return
```