```
'nRF24L01 LIB
'19.09.2017

'Memory Map - register address defines
#define CONFIG      0x00
#define EN_AA       0x01
#define EN_RXADDR   0x02
#define SETUP_AW    0x03
#define SETUP_RETR  0x04
#define RF_CH       0x05
#define RF_SETUP    0x06
#define STATUS_REG  0x07  'STATUS is a PIC register -> STATUS_REG for nRF24L01
#define OBSERVE_TX  0x08
#define RX_ADDR_P0  0x0A
#define RX_ADDR_P1  0x0B
#define RX_ADDR_P2  0x0C
#define RX_ADDR_P3  0x0D
#define RX_ADDR_P4  0x0E
#define RX_ADDR_P5  0x0F
#define TX_ADDR     0x10
#define RX_PW_P0    0x11
#define RX_PW_P1    0x12
#define RX_PW_P2    0x13
#define RX_PW_P3    0x14
#define RX_PW_P4    0x15
#define RX_PW_P5    0x16
#define FIFO_STATUS 0x17

'Bit Mnemonics
#define MASK_RX_DR  6
#define MASK_TX_DS  5
#define MASK_MAX_RT 4
#define EN_CRC      3
#define CRCO        2
#define PWR_UP      1
#define PRIM_RX     0
#define ENAA_P5     5
#define ENAA_P4     4
#define ENAA_P3     3
#define ENAA_P2     2
#define ENAA_P1     1
#define ENAA_P0     0
#define ERX_P5      5
#define ERX_P4      4
#define ERX_P3      3
#define ERX_P2      2
#define ERX_P1      1
#define ERX_P0      0
#define AW          0
#define ARD         4
#define ARC         0
#define PLL_LOCK    4
#define RF_DR_HIGH  3
#define RF_DR_LOW   5
#define RF_PWR      1
#define LNA_HCURR   0
#define RX_DR       6
#define TX_DS       5
#define MAX_RT      4
#define RX_P_NO     1
#define TX_FULL     0
#define PLOS_CNT    4
#define ARC_CNT     0
#define TX_REUSE    6
#define FIFO_FULL   5
#define TX_EMPTY    4
#define RX_FULL     1
#define RX_EMPTY    0
```

```
'Command Name Mnemonics (Instructions)
#define R_REGISTER    0x00
#define W_REGISTER    0x20 '001A AAAA
#define REGISTER_MASK 0x1F
#define R_RX_PAYLOAD  0x61
#define W_TX_PAYLOAD  0xA0
#define FLUSH_TX      0xE1
#define FLUSH_RX      0xE2
#define REUSE_TX_PL   0xE3
#define WL_NOP        0xFF
```

```
' nRF24L01 TX Test 1
' 19.09.2017
'                                ------------
'  V3.3P                        -| 1       20 |- GND
'  PORTA.5                      -| 2       19 |- PORTA.0 SER_OUT
'  PORTA.4                      -| 3       18 |- PORTA.1
'  PORTA.3                      -| 4       17 |- PORTA.2
'  PORTC.5                      -| 5       16 |- PORTC.0
'  PORTC.4                      -| 6       15 |- PORTC.1
'  PORTC.3                   __  -| 7       14 |- PORTC.2
'  PORTC.6                   SS  -| 8       13 |- PORTB.4 SDI
'  PORTC.7                   SDO -| 9       12 |- PORTB.5
'  PORTB.7                   CE  -| 10      11 |- PORTB.6 SCK
'                                ------------
'
'          nRF24L01 Pins top view, SMD view
' ---------------------------------------
' | GND        1 *    * 2 V3.3P  __      |
' | CE         3 *    * 4 CSN   (SS)     |
' | SCK        5 *    * 6 MOSI  (SDO)    |
' | MISO (SDI) 7 *    * 8 IRQ            |
' |                                       |
' |                          ------       |
' |                         |Quarz |      |
' |                          ------       |
' ---------------------------------------
#chip 16F690,8
#config OSC=INTRC_OSC_NOCLKOUT, PWRTE = ON

#include "C:\gcbasic_src\software_uart_v11.h"
#include "nRF24L01.h"

'nRF24L01 Settings and defined commands
#define FlushTX Write_Cmd(FLUSH_TX) 'used in RX mode, discard packet

#define STANDBY_I 0b01110010 'Power up, Interrupt's disabled
#define TX_LENGTH 2 ' Number of bytes to send

'Serial settings
#define BAUD_RATE 2400
#define SER_OUT PORTA.0
dir SER_OUT out

'SPI pin settings

#define SDO PORTC.7
dir SDO out

#define SDI PORTB.4
dir SDI in

#define SS PORTC.6
dir SS out

#define SCK PORTB.6
dir SCK out

#define CE PORTB.7
dir CE out

'global variables
dim Dummy,counter as byte
Dummy = 0
counter = 0

dim RXTX_ADDR(3) 'RX TX 3 byte address data pipe 0
dim tx_data(TX_LENGTH) 'data bytes to send

'Start up
wait 1 s 'wait for LCD ready
LCD_Clear
SerPrint("TX Test1")
```

```
wait 2 s
LCD_Clear

'nRF_Setup
'init SPI & CE pin

SPIMode MasterSlow,0 'SPI Mode Setup

set SDO off
set SCK off
set SS on
set CE off

' nRF Setup
RXTX_ADDR(1) = 0xB1
RXTX_ADDR(2) = 0xB2
RXTX_ADDR(3) = 0xB3

Write_Reg(CONFIG, STANDBY_I)    ' STANDBY_I Mode
Write_Reg(EN_AA, 0x00)          ' Disable auto ack
Write_Reg(SETUP_AW, 0x01)       ' 3 byte address
Write_Reg(SETUP_RETR, 0x00)     ' Retransmit disabled
Write_Reg(RF_CH, 0x3F)          ' RF channel 63
Write_Reg(RF_SETUP, 0x26)       ' 250kbps, 0dBm
Write_Addr_Pipe(TX_ADDR)        ' 3 byte TX addres data pipe 0
TX_Mode
FlushTX
' end nRF_Setup

'----------------------------------- main -------------------------------------
Do
 tx_data(1) = counter
 tx_data(2) = 255 - counter
 counter++
 Write_Payload
 'TX_Mode
 wait 1 s
 FlushTX
Loop

'----------------------------------- subs -------------------------------------
sub Read_Reg(in Register as byte) 'Read content of a 1 byte register
    Dummy = 0
    Register = R_REGISTER + Register

    set SS off
    SPITransfer Register,Dummy
    SPITransfer R_REGISTER,Dummy
    set SS on
end sub 'Read_Reg
'------------------------------------------------------------------------------
sub Write_Reg(in Register as byte, in Data as byte) 'Write 1 byte to register
    Dummy = 0
    Register = W_REGISTER + Register

    set SS off
    SPITransfer Register, Dummy
    SPITransfer Data, Dummy
    set SS on
end sub 'Write_Reg
'------------------------------------------------------------------------------
```

```
sub Write_Addr_Pipe(in Register as byte)
    Register = W_REGISTER + Register

    set SS off
    SPITransfer Register, Dummy

    for byte_num = 1 to 3
       SPITransfer RXTX_ADDR(byte_num), Dummy
    next

    set SS on
end sub 'Write_Addr_Pipe
'-------------------------------------------------------------------------------
sub Write_Cmd(in command as byte)
    set SS off
    SPITransfer command, Dummy
    set SS on
end sub 'Write_Cmd
'-------------------------------------------------------------------------------
sub TX_Mode
    set CE off
    WRITE_REG(CONFIG, 0b1111010) '1 byte CRC, POWER UP, PTX, INT's disabled
end sub 'TX_Mode
'-------------------------------------------------------------------------------
sub Write_Payload
  set SS off
  SPITransfer W_TX_PAYLOAD, Dummy
  for num = 1 to TX_LENGTH
     SPITransfer tx_data(num), Dummy
  next
  set SS on

  set CE on
  wait 150 us
  set CE off
end sub 'Write_Payload
```

```
' nRF24L01 RX Test 1
' 18.09.2017
'                                 ------------
'  V3.3P                        -| 1       20 |- GND
'  PORTA.5                      -| 2       19 |- PORTA.0 SER_OUT
'  PORTA.4                      -| 3       18 |- PORTA.1
'  PORTA.3         PUSH_BUTTON  -| 4       17 |- PORTA.2
'  PORTC.5                      -| 5       16 |- PORTC.0
'  PORTC.4                      -| 6       15 |- PORTC.1
'  PORTC.3                  __  -| 7       14 |- PORTC.2 ALARM
'  PORTC.6                  SS  -| 8       13 |- PORTB.4 SDI
'  PORTC.7                  SDO -| 9       12 |- PORTB.5
'  PORTB.7                  CE  -| 10      11 |- PORTB.6 SCK
'                                 ------------
'
'          nRF24L01 Pins top view, SMD view
' ---------------------------------------
' | GND        1 *    * 2 V3.3P  __       |
' | CE         3 *    * 4 CSN   (SS)      |
' | SCK        5 *    * 6 MOSI  (SDO)     |
' | MISO (SDI) 7 *    * 8 IRQ             |
' |                                       |
' |                          ------       |
' |                         |Quarz |      |
' |                          ------       |
' ---------------------------------------
#chip 16F690,8
#config OSC=INTRC_OSC_NOCLKOUT, PWRTE = ON

#include "C:\gcbasic_src\software_uart_v2.h"
#include "nRF24L01.h"

'nRF24L01 Settings and defined commands
#define FlushRX Write_Cmd(FLUSH_RX) 'used in RX mode, discard packet

#define STANDBY_I 0b01110010 'Power up, Interrupt's disabled
#define RX_LENGTH 2 ' Number of bytes to receive

'Serial settings
#define BAUD_RATE 9600
#define SER_OUT PORTA.0
dir SER_OUT out

'bsf IOC,IOCA3 'enable Interrupt PORTABChange on PORTA.3
'On Interrupt PORTABChange Call ISR1 'Interrupt Service Routine 1

#define ALARM PORTC.2 'Buzzer
dir ALARM out

'SPI pin settings

#define SDO PORTC.7
dir SDO out

#define SDI PORTB.4
dir SDI in

#define SS PORTC.6
dir SS out

#define SCK PORTB.6
dir SCK out

#define CE PORTB.7
dir CE out

'global variables
dim Dummy as byte
Dummy = 0

dim RXTX_ADDR(3) 'RX TX 3 byte address data pipe 0
```

```
dim rx_data(RX_LENGTH) 'data bytes to receive

rx_data(1) = 0
rx_data(2) = 0

'dim PUSH_BUTTON as bit 'later Menu setup
'PUSH_BUTTON = 0

'Start up
wait 1 s 'wait for LCD ready
LCD_Clear
SerPrint("RX Test1")
wait 2 s
LCD_Clear

'nRF_Setup
'init SPI & CE pin

SPIMode MasterSlow,0 'SPI Mode Setup

set SDO off
set SCK off
set SS on
set CE off

' nRF Setup
RXTX_ADDR(1) = 0xB1
RXTX_ADDR(2) = 0xB2
RXTX_ADDR(3) = 0xB3

Write_Reg(CONFIG, STANDBY_I)    ' STANDBY_I Mode
Write_Reg(EN_AA, 0x00)          ' Disable auto ack
Write_Reg(EN_RXADDR, 0x01)      ' Enable data pipe 0
Write_Reg(SETUP_AW, 0x01)       ' 3 byte address
Write_Reg(SETUP_RETR, 0x00)     ' Retransmit disabled
Write_Reg(RF_CH, 0x3F)          ' RF channel 63
Write_Reg(RF_SETUP, 0x26)       ' 250kbps, 0dBm
Write_Reg(RX_PW_P0, RX_LENGTH)  ' RX payload, num of bytes, max 32
Write_Addr_Pipe(RX_ADDR_P0)     ' 3 byte RX addres data pipe 0
RX_Mode
FlushRX
' end nRF_Setup

'----------------------------------- main -------------------------------------
Do
 Read_Payload
 LCD_Clear
 print_byte(rx_data(1))
 LCD_Line2
 print_byte(rx_data(2))
 wait 1 s
Loop

'----------------------------------- subs -------------------------------------
sub Read_Reg(in Register as byte) 'Read content of a 1 byte register
    Dummy = 0
    Register = R_REGISTER + Register

    set SS off
    SPITransfer Register,Dummy
    SPITransfer R_REGISTER,Dummy
    set SS on
end sub 'Read_Reg
'--------------------------------------------------------------------------------
```

```
sub Write_Reg(in Register as byte, in Data as byte) 'Write 1 byte to register
    Dummy = 0
    Register = W_REGISTER + Register

    set SS off
    SPITransfer Register, Dummy
    SPITransfer Data, Dummy
    set SS on
end sub 'Write_Reg
'-------------------------------------------------------------------------------
sub Write_Addr_Pipe(in Register as byte)
    Register = W_REGISTER + Register

    set SS off
    SPITransfer Register, Dummy

    for byte_num = 1 to 3
       SPITransfer RXTX_ADDR(byte_num), Dummy
    next

    set SS on
end sub 'Write_Addr_Pipe
'-------------------------------------------------------------------------------
sub Write_Cmd(in command as byte)
    set SS off
    SPITransfer command, Dummy
    set SS on
end sub 'Write_Cmd
'-------------------------------------------------------------------------------
sub RX_Mode
    WRITE_REG(CONFIG, 0b1111011) '1 byte CRC, POWER UP, PRX, INT's disabled
    set CE on
    wait 130 us 'wait for RX ready
end sub 'RX_Mode
'-------------------------------------------------------------------------------
sub Read_Payload
  set SS off
  SPITransfer R_RX_PAYLOAD, Dummy

  for num = 1 to RX_LENGTH
     SPITransfer WL_NOP, rx_data(num)
  next

  set SS on
end sub 'Read_Payload
'-------------------------------------------------------------------------------
sub ISR1
    PUSH_BUTTON = 1
    wait 100 ms
end sub 'ISR1
```