Коми Республикаса велӧдан, наука да том йӧз политика министерство
Министерство образования, науки и молодежной политики Республики Коми
Государственное профессиональное образовательное учреждение
«Сыктывкарский целлюлозно – бумажный техникум»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
РАБОТА С КНОПКАМИ ДЛЯ МИКРОКОНТРОЛЛЕРОВ PIC16FXX НА ЯЗЫКЕ АССЕМБЛЕР

МДК 02.01 Микропроцессорные системы

Специальность: 09.02.01 Компьютерные системы и комплексы

Сыктывкар 2021 г.

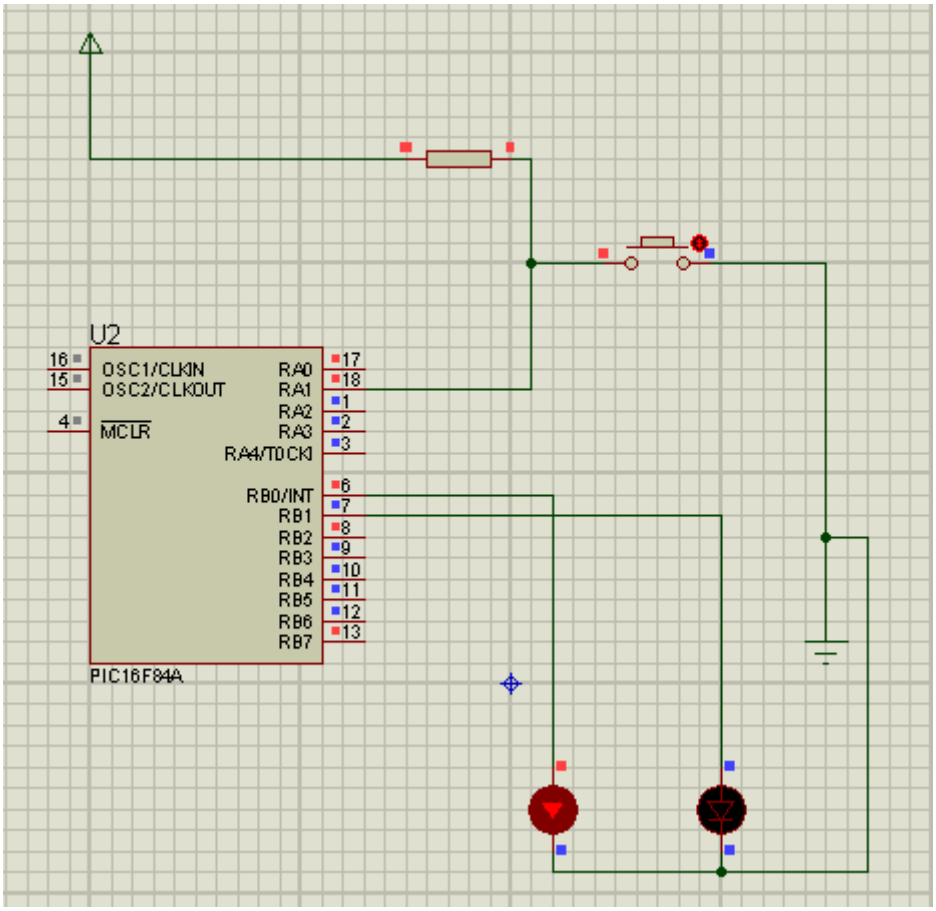
Учебное пособие предназначено для студентов III курса по направлению подготовки 09.02.01 Компьютерные системы и комплексы. Пособие служит практическим руководством для программирования микроконтроллеров серии PIC16FXX. Рассмотрены примеры программ с различными реализациями программного кода. Рекомендуется для использования на лабораторных работах по МДК 02.01 Микропроцессорные системы, выполнения курсовых и дипломных проектов.

Преподаватель _____ Расов Д.Д.

Оглавление

1. ПРОГРАММА ОПРЕДЕЛЕНИЯ НАЖАТИЯ КНОПКИ 4
2. ОПРЕДЕЛЕНИЕ СОСТОЯНИЯ КНОПКИ 7
3. ПЕРЕКЛЮЧЕНИЕ ВЫБРАННЫХ ЛИНИЙ ПОРТА НА ВЫВОД ... 10
4. ПРОГРАММА ОПРЕДЕЛЕНИЯ НАЖАТИЯ ТРЕХ КНОПОК 14
5. СООТВЕТСТВИЕ КНОПОК СВОИМ НОМЕРАМ 18
6. ВКЛЮЧЕНИЕ СВЕТОДИОДОВ ПО НОМЕРУ В ДВОИЧНОЙ СИСТЕМЕ 22
БИБЛИОГРАФИЧЕСКИЙ СПИСОК 27

1. ПРОГРАММА ОПРЕДЕЛЕНИЯ НАЖАТИЯ КНОПКИ



Программный код:

; Программа определения нажатия кнопки. Если кнопка не нажата, то светится LED_1, а если кнопка нажата - LED_2
LIST P=PIC16F84A

#INCLUDE “P16F84A.INC”
; Конфигурация микроконтроллера: стандартный кварцевый генератор & сторожевой таймер выключен & будет производиться выдержка при включении питания & защита памяти программ выключена
; __CONFIG 0x3FF1
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _CP_OFF

ERRORLEVL -207 ; Эта строчка говорит о том, что не надо выводить сообщение о предупреждении № 207
; Само предупреждение гласит, что метка, например, "MAIN" находится не на первой колонке строчки.

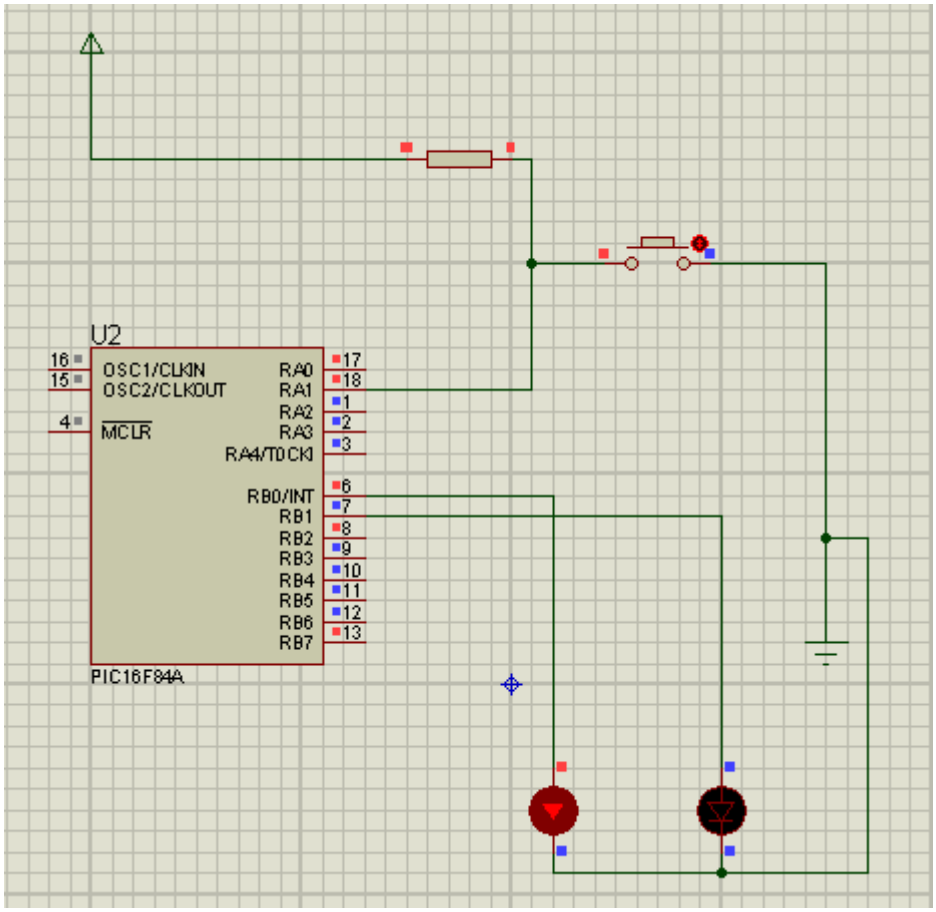
ERRORLEVEL -302 ; О предупреждении 302, но если вместо "MOVWF TRISA" записать "MOVWF TRISA^80h", (добавляя к названию регистра

"^80", мы определяем место нахождения данного регистра) то данное предупреждение исчезнет.

```
#define     LED_1     PORTB,0    ; определяем расположение LED_1
#define     LED_2     PORTB,1    ; определяем расположение LED_2

CBLOCK      20H                  ; блок РОН начинается с адреса 20h
            Reg_1                ; объявляем регистр Reg_1 для задержки
            Reg_2                ; объявляем регистр Reg_2 для задержки
ENDC                             ; конец блока

BANK0       MACRO                ; макрос перехода в BANK0
        BCF STATUS,RP0           ; сброс бита RP0
        BCF STATUS,RP1           ; сброс бита RP1
            ENDM                 ; конец макроса

BANK1       MACRO                ; макрос перехода в BANK1
        BSF STATUS,RP0           ; установка бита RP0
        BCF STATUS,RP1           ; сброс бита RP1
            ENDM                 ; конец макроса

        ORG 0    ; установка текущего адреса на 0
        GOTO     BEGIN           ; пропускаем фрагмент задержки

; Подпрограмму задержек можно автоматически сформировать с помощью программы задержек

DELAY                            ; начало сгенерированной задержки
        movlw     .221
        movwf     Reg_1
        movlw     .13
        movwf     Reg_2
        decfsz    Reg_1,F
        goto      $-1
        decfsz    Reg_2,F
        goto      $-3
        nop
        nop
        return    ; конец сгенерированной задержки

BEGIN                            ; начало настройки портов
        BANK1                    ; переход в BANK1
        CLRF      TRISB          ; PORTB на вывод
        MOVLW B'10'
        MOVWF     TRISA          ; PORTA на ввод
```

```
        BANK0                   ; переход в BANK0

MAIN    BTFSC PORTA,1 ; Проверяем состояние кнопки
        GOTO  DALEE   ; Кнопка не нажата - на выводе PORTA,1 - "1"
Программа выполняется далее
        BTFSS PORTA,1
        GOTO $-1
        GOTO PROG_2   ; на выводе PORTA,1 - "0" т.е. кнопка еще
нажата. Переходим на выполнение программы PROG_2

DALEE   BSF LED_1     ; Включаем светодиод led_1
        BCF LED_2     ; Выключаем светодиод led_2
        GOTO MAIN     ; идем на анализ состояния кнопки

PROG_2  BSF LED_2     ; Включаем светодиод led_2
        BCF LED_1     ; Выключаем светодиод led_1
        GOTO $      ; идем на анализ состояния кнопки

END
```

2. ОПРЕДЕЛЕНИЕ СОСТОЯНИЯ КНОПКИ



; Подпрограмма 2 определения состояния кнопки. В этой подпрограмме не используется программная задержка, но реакция программы на изменение состояния кнопки происходит после ее отпускания.
; Работа в PROTEUS: Включаем - светится LED_1. Нажимаем на кнопку - светится LED_1.
; Отпускаем кнопку - светится LED_2, LED_1 гаснет. Все программа на этом зацикливается.

```
LIST  P=PIC16F84A

#INCLUDE “P16F84A.INC”
; Конфигурация микроконтроллера: стандартный кварцевый генератор & сторожевой таймер выключен & будет производиться выдержка при включении питания & защита памяти программ выключена
; __CONFIG 0x3FF1
 __CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _CP_OFF

ERRORLEVL  -207  ; Эта строчка говорит о том, что не надо выводить сообщение о предупреждении № 207
```

; Само предупреждение гласит, что метка, например, "MAIN" находится не на первой колонке строчки.

ERRORLEVEL -302 ; О предупреждении 302, но если вместо "MOVWF TRISA" записать "MOVWF TRISA^80h", (добавляя к названию регистра "^80", мы определяем место нахождения данного регистра) то данное предупреждение исчезнет.

; Определяем к каким разрядам портов подключены кнопки

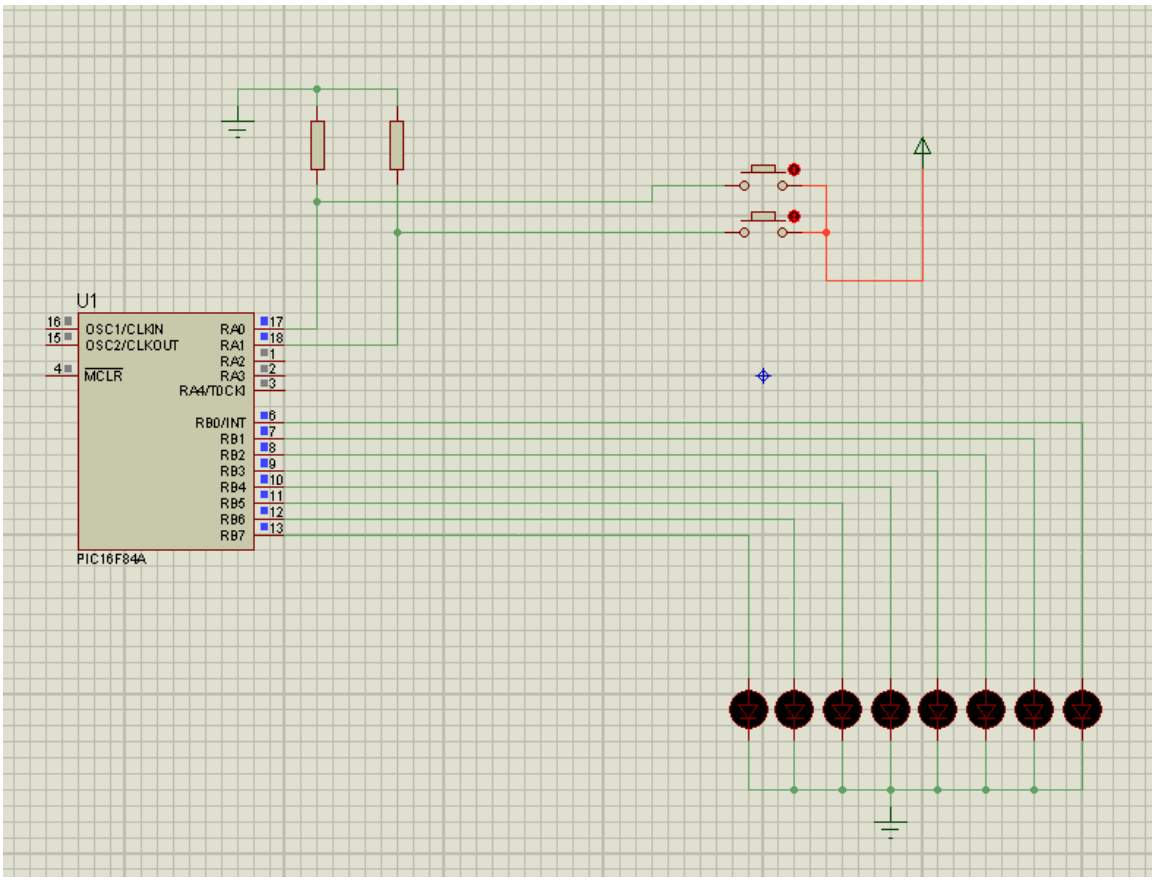
```
#define     LED_1     PORTB,0    ; определяем расположение LED_1
#define     LED_2     PORTB,1    ; определяем расположение LED_2
#define     KEY_1     PORTA,1    ; Кнопка включения светодиода LED_1

CBLOCK      20H                  ; блок РОН начинается с адреса 20h
            Reg_1                ; объявляем регистр Reg_1 для задержки
            Reg_2                ; объявляем регистр Reg_2 для задержки
ENDC                             ; конец блока

BANK0       MACRO                ; макрос перехода в BANK0
        BCF STATUS,RP0           ; сброс бита RP0
        BCF STATUS,RP1           ; сброс бита RP1
            ENDM                 ; конец макроса

BANK1       MACRO                ; макрос перехода в BANK1
        BSF STATUS,RP0           ; установка бита RP0
        BCF STATUS,RP1           ; сброс бита RP1
            ENDM                 ; конец макроса

        ORG 0                    ; установка текущего адреса на 0
        GOTO    BEGIN            ; пропускаем фрагмент задержки
```

; Подпрограмму задержек можно автоматически сформировать с помощью программы задержек

```
DELAY                            ; начало сгенерированной задержки
        movlw     .221
        movwf     Reg_1
        movlw     .13
        movwf     Reg_2
        decfsz    Reg_1,F
        goto      $-1
        decfsz    Reg_2,F
        goto      $-3
        nop
        nop
```

```
        return                  ; конец сгенерированной задержки

BEGIN                           ; начало настройки портов
        BANK1                   ; переход в BANK1
        CLRF      TRISB         ; PORTB на вывод
        MOVLW B'10'
        MOVWF     TRISA         ; PORTA на ввод
        BANK0                   ; переход в BANK0

MAIN        BTFSC KEY_1         ; Проверяем состояние кнопки
            GOTO  DALEE         ; Кнопка не нажата - на выводе PORTA,1 - "1"
                                ; Программа выполняется далее
            BTFSS KEY_1         ; Повторно проверяем потенциал на KEY_1
            GOTO $-1            ; на выводе PORTA,1 - "0" т.е. кнопка
                                ; еще нажата
            ; Возвращаемся на один шаг назад и ждем пока не отпустят кнопку
            GOTO PROG_2         ; Кнопку отпустили, идем на исполнение
                                ; программы PROG_2
DALEE       BSF LED_1           ; Включаем светодиод LED_1
            BCF LED_2           ; Выключаем светодиод LED_2
            GOTO MAIN           ; идем на анализ состояния кнопки

PROG_2      BSF LED_2           ; Включаем светодиод LED_2
            BCF LED_1           ; Выключаем светодиод LED_1
            GOTO $              ; Программа закольцовывается на этой команде
                                ; до выключения

END
```

3. ПЕРЕКЛЮЧЕНИЕ ВЫБРАННЫХ ЛИНИЙ ПОРТА НА ВЫВОД



Код программы:

; Программа определения состояния кнопки. В этой подпрограмме не используется программная задержка, но реакция программы на изменение состояния кнопки происходит после ее отпускания.
; Работа в PROTEUS: Включаем – индикаторы не светятся. Нажимаем на кнопку KEY1 - светятся светодиоды, подключенные к порту PORTB0-3, нажимаем еще раз - гаснут. При нажатии на KEY2 – загораются светодиоды, подключенные к порту PORTB4-7, нажимаем еще раз – гаснут.

```
LIST  P=PIC16F84A

#INCLUDE “P16F84A.INC”
; Конфигурация микроконтроллера: стандартный кварцевый генератор & сторожевой таймер выключен & будет производиться выдержка при включении питания & защита памяти программ выключена
; __CONFIG 0x3FF1
 __CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _CP_OFF

ERRORLEVL  -207  ; Эта строчка говорит о том, что не надо выводить сообщение о предупреждении № 207
```

; Само предупреждение гласит, что метка, например, "MAIN" находится не на первой колонке строчки.

ERRORLEVEL -302 ; О предупреждении 302, но если вместо "MOVWF TRISA" записать "MOVWF TRISA^80h", (добавляя к названию регистра "^80", мы определяем место нахождения данного регистра) то данное предупреждение исчезнет.

; Определяем к каким разрядам портов подключены кнопки

```
CBLOCK   20H          ; блок РОН начинается с адреса 20h
         Reg_1              ; объявляем регистр Reg_1 для задержки
         Reg_2              ; объявляем регистр Reg_2 для задержки
ENDC                  ; конец блока

BANK0    MACRO        ; макрос перехода в BANK0
       BCF STATUS,RP0 ; сброс бита RP0
       BCF STATUS,RP1 ; сброс бита RP1
         ENDM         ; конец макроса

BANK1    MACRO        ; макрос перехода в BANK1
       BSF STATUS,RP0 ; установка бита RP0
       BCF STATUS,RP1 ; сброс бита RP1
         ENDM         ; конец макроса

       ORG 0          ; установка текущего адреса на 0
       GOTO   BEGIN   ; пропускаем фрагмент задержки
```

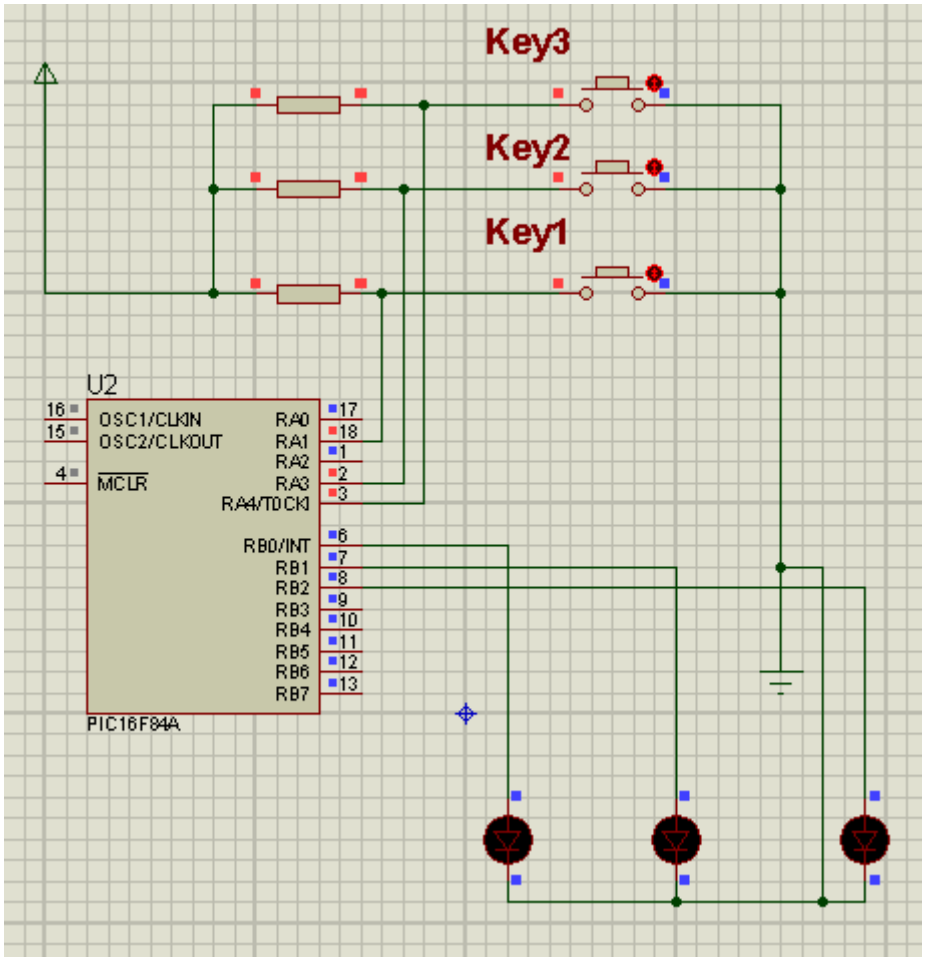
; Подпрограмму задержек можно автоматически сформировать с помощью программы задержек

```
DELAY                 ; начало сгенерированной задержки
       movlw    .221
       movwf    Reg_1
       movlw    .13
       movwf    Reg_2
       decfsz   Reg_1,F
       goto     $-1
       decfsz   Reg_2,F
       goto     $-3
       nop
       nop
       return         ; конец сгенерированной задержки

BEGIN                 ; начало настройки портов
```

```
        BANK1                   ; переход в BANK1
        CLRF      TRISB         ; PORTB на вывод
        MOVLW B'11'
        MOVWF     TRISA         ; PORTA на ввод
        BANK0                   ; переход в BANK0

MAIN        BTFSC KEY_1
            GOTO  DALEE

            BTFSS KEY_1         ; Повторно проверяем потенциал на KEY_1
            GOTO $-1            ; на выводе PORTA,1 - "0" т.е. кнопка
                                ; еще нажата
            ; Возвращаемся на один шаг назад и ждем пока не отпустят кнопку
            GOTO PROG_2         ; Кнопку отпустили, идем на исполнение
                                ; программы PROG_2
DALEE       BSF LED_1           ; Включаем светодиод LED_1
            BCF LED_2           ; Выключаем светодиод LED_2
            GOTO MAIN           ; идем на анализ состояния кнопки

PROG_2      BSF LED_2           ; Включаем светодиод LED_2
            BCF LED_1           ; Выключаем светодиод LED_1
            GOTO $              ; Программа закольцовывается на этой команде
                                ; до выключения

LIGHT0
        BTFSS   PORTA,0         ; Проверяем состояние на PORTA в 0 бите
        GOTO    LIGHT1          ; Кнопка не нажата - на выводе PORTA,0 – “0”
                                ; Переходим на проверку состояния PORTA
                                ; в 1 бите
M1
        BTFSC   PORTA,0         ; Кнопка нажата – на выводе PORTA,0 – “1”
                                ; Проверяем отжатие кнопки (кнопка нажата –
        GOTO    M1              ; зацикливаем на M1)

        BTFSC   PORTB,0         ; светодиод на PORTB,2 – горит?
        GOTO    M2              ; ДА – переходим на LIGHT1 погасить на линиях
                                ; 0 - 3
        MOVLW B'00001111'       ; НЕТ – в рабочий регистр маску для установки
                                ; 1 на линиях 0 - 3
        IORWF   PORTB           ; Применяем логическое “ИЛИ” содержимого
                                ; рабочего регистра с PORTB
        CALL    DELAY           ; вызываем задержку
        GOTO    MAIN            ; переходим на начало проверки
M2
        MOVLW B'11110000'       ; В рабочий регистр размещаем маску для
```

```
                                ; выключения светодиодов на линиях 0–3
                                ; PORTB
        ANDWF  PORTB            ; Применяем логическое “И” содержимого
                                ; рабочего регистра с PORTB
        CALL   DELAY            ; вызываем задержку
        GOTO   MAIN             ; переходим на начало проверки

LIGHT1
        BTFSS  PORTA,1          ; Проверяем состояние на PORTA в 1 бите
        GOTO   MAIN             ; Кнопка не нажата - на выводе PORTA,1 – “0”
                                ; Переходим на MAIN
M3
        BTFSC  PORTA,0          ; Кнопка нажата – на выводе PORTA,1 – “1”
                                ; Проверяем отжатие кнопки (кнопка нажата –
        GOTO   M3               ; зацикливаем на M3)

        BTFSC  PORTB,4          ; светодиод на PORTB,4 – горит?
        GOTO   M4               ; ДА – переходим на M4 погасить на линиях
                                ; 4 - 7
        MOVLW  B'11110000'      ; НЕТ – в рабочий регистр маску для установки
                                ; 1 на линиях 4 - 7
        IORWF  PORTB            ; Применяем логическое “ИЛИ” содержимого
                                ; рабочего регистра с PORTB
        CALL   DELAY            ; вызываем задержку
        GOTO   MAIN             ; переходим на начало проверки
M4
        MOVLW  B'00001111'      ; В рабочий регистр размещаем маску для
                                ; выключения светодиодов на линиях 4–7
                                ; PORTB
        ANDWF  PORTB            ; Применяем логическое “И” содержимого
                                ; рабочего регистра с PORTB
        CALL   DELAY            ; вызываем задержку
        GOTO   MAIN             ; переходим на начало проверки

END
```

4. ПРОГРАММА ОПРЕДЕЛЕНИЯ НАЖАТИЯ ТРЕХ КНОПОК



Программный код:

; Программа определения нажатия 3х кнопок.
; Работа в PROTEUS: Включаем - светодиоды не светятся.
; Нажимаем на кнопку KEY1 - светится LED_1, будет светиться пока не нажмем на KEY2
; Нажимаем на KEY2 - светится LED_2, будет светиться пока не нажмем на KEY1
; Нажимаем на KEY3 - светится LED_3, будет светиться пока не нажмем на любую другую

```
LIST  P=PIC16F84A

#INCLUDE “P16F84A.INC”
; Конфигурация микроконтроллера: стандартный кварцевый генератор & сторожевой таймер выключен & будет производиться выдержка при включении питания & защита памяти программ выключена
; __CONFIG 0x3FF1
```

```
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _CP_OFF
```

ERRORLEVEL -207 ; Эта строчка говорит о том, что не надо выводить сообщение о предупреждении № 207
; Само предупреждение гласит, что метка, например, "MAIN" находится не на первой колонке строчки.

ERRORLEVEL -302 ; О предупреждении 302, но если вместо "MOVWF TRISA" записать "MOVWF TRISA^80h", (добавляя к названию регистра "^80", мы определяем место нахождения данного регистра) то данное предупреждение исчезнет.

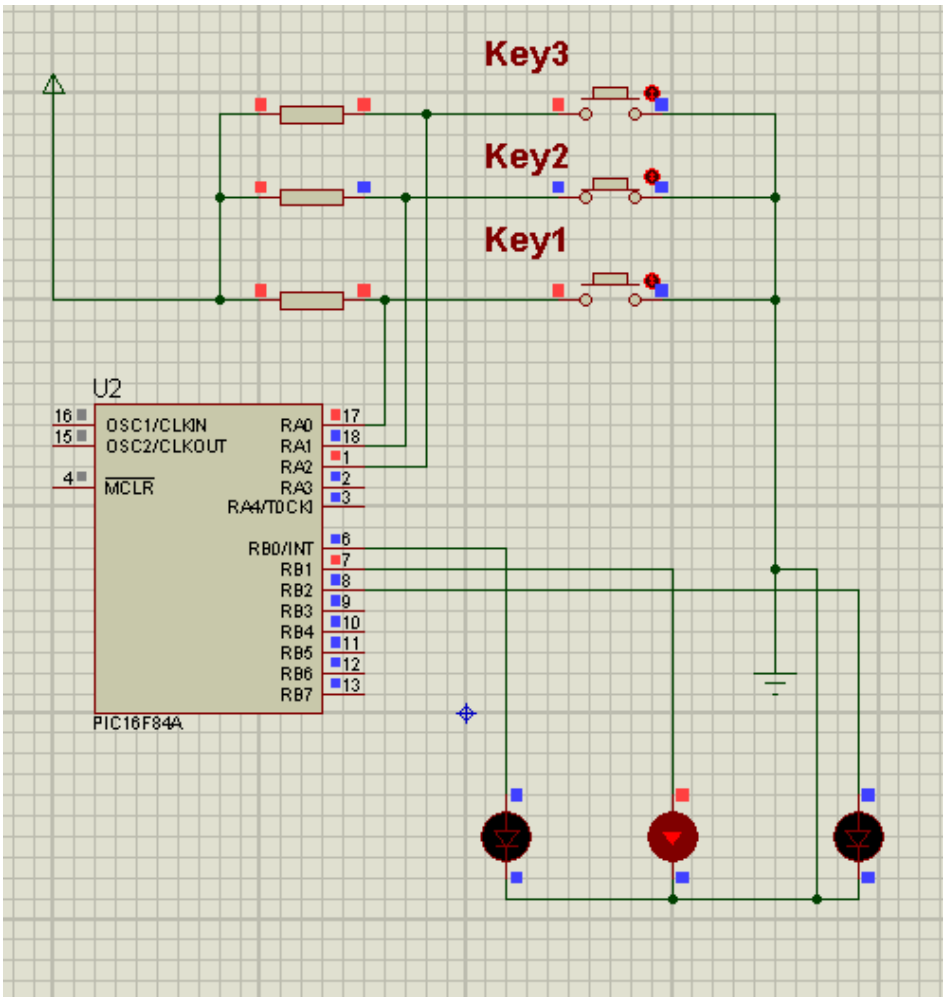
```
#define     LED_1     PORTB,0     ; определяем расположение LED_1
#define     LED_2     PORTB,1     ; определяем расположение LED_2
#define     LED_3     PORTB,2     ; определяем расположение LED_3
#define     KEY_1     PORTA,1     ; Кнопка включения светодиода LED_1
#define     KEY_2     PORTA,3     ; Кнопка включения светодиода LED_2
#define     KEY_3     PORTA,4     ; Кнопка включения светодиода LED_3
#define     KNOPKI    b'00011010' ; Маска для выделения информации на тех
разрядах PORTA, к которым подключены кнопки
#define     KEY1      b'00011000' ; Константа нажатия первой кнопки – 1
разряд PORTA равен нулю - кнопка нажата
#define     KEY2      b'00010010' ; Константа нажатия второй кнопки – 3
разряд PORTA равен нулю - кнопка нажата
#define     KEY3      b'00001010' ; Константа нажатия третьей кнопки – 4
разряд PORTA равен нулю - кнопка нажата
#define     Z_        STATUS,Z    ; Флаг нулевого результата
CBLOCK      20H       ; блок РОН начинается с адреса 20h
            Reg_1           ; объявляем регистр Reg_1 для задержки
            Reg_2           ; объявляем регистр Reg_2 для задержки
            Reg_KNOPKI      ; Регистр для работы с информацией от кнопок
ENDC                  ; конец блока

BANK0       MACRO           ; макрос перехода в BANK0
        BCF STATUS,RP0      ; сброс бита RP0
        BCF STATUS,RP1      ; сброс бита RP1
            ENDM            ; конец макроса

BANK1       MACRO           ; макрос перехода в BANK1
        BSF STATUS,RP0      ; установка бита RP0
        BCF STATUS,RP1      ; сброс бита RP1
            ENDM            ; конец макроса

        ORG 0     ; установка текущего адреса на 0
        GOTO      BEGIN     ; пропускаем фрагмент задержки
```

; Подпрограмму задержек можно автоматически сформировать с помощью программы задержек

```
DELAY                       ; начало сгенерированной задержки
        movlw     .221
        movwf     Reg_1
        movlw     .13
        movwf     Reg_2
        decfsz    Reg_1,F
        goto      $-1
        decfsz    Reg_2,F
        goto      $-3
        nop
        nop
        return    ; конец сгенерированной задержки

BEGIN                       ; начало настройки портов
        BANK1               ; переход в BANK1
        CLRF      TRISB     ; PORTB на вывод
        MOVLW     B'00111010'
        MOVWF     TRISA     ; PORTA на ввод
        BANK0               ; переход в BANK0
        CLRF      PORTA     ; Инициализация портов
        CLRF      PORTB
        CLRF      Reg_KNOPKI

MAIN
        MOVFW PORTA         ; Читаем информацию на все разрядах PORTA
        ANDLW KNOPKI        ; Выделяем информацию от кнопок путем поразрядного логического "и" с константой "KNOPKI" - 00011010 -, где нули информация затирается и остаются нужные нам разряды, результат оставляем в аккумуляторе
        MOVWF Reg_KNOPKI ; Запоминаем в вспомогательном регистре
        SUBLW KEY1          ; Сравниваем с константой нажатой первой кнопки,
        BTFSC Z_            ; если оны равны - флаг нулевого результата поднят -
        GOTO  PROG_1        ; идем на выполнение первой программы
        MOVFW Reg_KNOPKI ; Читаем данные PORTA сохраненные в регистре
        SUBLW KEY2          ; сравниваем с константой нажатой второй кнопки,
        BTFSC Z_            ; если они равны,
        GOTO  PROG_2        ; то идем на выполнение второй программы
```

```
        MOVFW Reg_KNOPKI ; Снова читаем сохраненные данные и
        SUBLW  KEY3      ; сравниваем с константой третьей кнопки
        BTFSC  Z_        ; если они равны, то
        GOTO   PROG_3    ; выполняем программу №3
        GOTO   MAIN      ; ни одна из констант не совпала, поэтому
                         ; возвращаемся на повторный анализ положения кнопок.

; Подпрограмма обработки нажатой кнопки KEY_1

PROG_1
            BSF  LED_1     ; Включаем светодиод LED_1
            BCF  LED_2     ; Выключаем светодиод LED_2
            BCF  LED_3     ; Выключаем светодиод LED_3
            GOTO MAIN

PROG_2      BSF  LED_2     ; Включаем светодиод LED_2
            BCF  LED_1     ; Выключаем светодиод LED_1
            BCF  LED_3     ; Выключаем светодиод LED_3
            GOTO MAIN      ; идем на начало

PROG_2      BSF  LED_3     ; Включаем светодиод LED_3
            BCF  LED_1     ; Выключаем светодиод LED_1
            BCF  LED_2     ; Выключаем светодиод LED_2
            GOTO MAIN      ; идем на начало

END
```

5. СООТВЕТСТВИЕ КНОПОК СВОИМ НОМЕРАМ



Программный код:

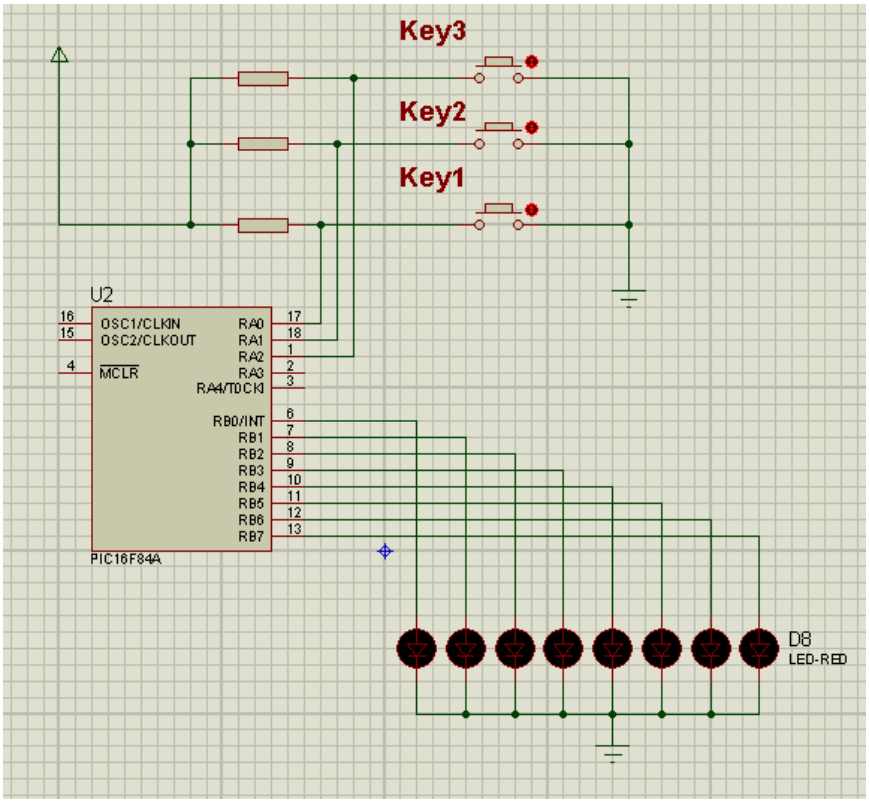
; Если кнопки подсоединить по порядку, т.е. KEY_1 к PORTA,0 KEY_2 к PORTA,2 а KEY_3 к PORTA,2
; тогда состояние этих разрядов PORTA после его конвертации будет выглядеть след. образом.
; Состояние порта при нажатой кнопке KEY_1 - 00000110, после конвертации - 11111001
; После выделения нужных нам разрядов получим 00000001. т.е. нажатой кнопке 1 соответствует число 1
; Работа в PROTEUS: Включаем - светодиоды не светятся.
; Нажимаем на кнопку KEY1 - светится LED_1, будет светиться пока не нажмем на любую другую.
; Нажимаем на KEY2 - светится LED_2, будет светиться пока не нажмем на любую другую.
; Нажимаем на KEY3 - светится LED_3, будет светиться пока не нажмем на любую другую.

```
LIST  P=PIC16F84A

#include "p16f84a.inc"

; Конфигурация микроконтроллера: стандартный кварцевый генератор & сторожевой таймер выключен & будет производиться выдержка при включении питания & защита памяти программ выключена
; __CONFIG 0x3FF1

 __CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _CP_OFF

ERRORLEVEL  -207  ; Эта строчка говорит о том, что не надо выводить сообщение о предупреждении № 207
; Само предупреждение гласит, что метка, например, "MAIN" находится не на первой колонке строчки.

ERRORLEVEL  -302  ; О предупреждении 302, но если вместо "MOVWF TRISA" записать "MOVWF  TRISA^80h", (добавляя к названию регистра "^80", мы определяем место нахождения данного регистра) то данное предупреждение исчезнет.

#define      LED_1      PORTB,0     ; определяем расположение LED_1
#define      LED_2      PORTB,1     ; определяем расположение LED_2
#define      LED_3      PORTB,2     ; определяем расположение LED_3
#define      KEY_1      PORTA,0     ; Кнопка включения светодиода LED_1
#define      KEY_2      PORTA,1     ; Кнопка включения светодиода LED_2
#define      KEY_3      PORTA,2     ; Кнопка включения светодиода LED_3
#define      KNOPKI     b'00000111' ; Маска для выделения информации на тех разрядах PORTA, к которым подключены кнопки
#define      KEY1       b'00000001' ; Константа нажатия первой кнопки – 1 разряд PORTA равен нулю - кнопка нажата
#define      KEY2       b'00000010' ; Константа нажатия второй кнопки – 3 разряд PORTA равен нулю - кнопка нажата
#define      KEY3       b'00000100' ; Константа нажатия третьей кнопки – 4 разряд PORTA равен нулю - кнопка нажата
#define      Z_         STATUS,Z    ; Флаг нулевого результата

CBLOCK   20H            ; блок РОН начинается с адреса 20h
         Reg_1                ; объявляем регистр Reg_1 для задержки
         Reg_2                ; объявляем регистр Reg_2 для задержки
         Reg_KNOPKI           ; Регистр для работы с информацией от кнопок
ENDC                    ; конец блока

BANK0    MACRO                ; макрос перехода в BANK0
         BCF STATUS,RP0       ; сброс бита RP0
```

```
        BCF STATUS,RP1      ; сброс бита RP1
ENDM                ; конец макроса

BANK1     MACRO             ; макрос перехода в BANK1
        BSF STATUS,RP0      ; установка бита RP0
        BCF STATUS,RP1      ; сброс бита RP1
ENDM                ; конец макроса

        ORG 0     ; установка текущего адреса на 0
        GOTO    BEGIN       ; пропускаем фрагмент задержки

; Подпрограмму задержек можно автоматически сформировать с помощью программы задержек

DELAY                       ; начало сгенерированной задержки
        movlw     .221
        movwf     Reg_1
        movlw     .13
        movwf     Reg_2
        decfsz    Reg_1,F
        goto      $-1
        decfsz    Reg_2,F
        goto      $-3
        nop
        nop
        return    ; конец сгенерированной задержки

BEGIN                       ; начало настройки портов
        BANK1               ; переход в BANK1
        CLRF      TRISB     ; PORTB на вывод
        MOVLW     B'00000111'
        MOVWF     TRISA     ; PORTA на ввод
        BANK0               ; переход в BANK0
        CLRF      PORTA     ; Инициализация портов
        CLRF      PORTB
        CLRF      Reg_KNOPKI

MAIN
        MOVFW PORTA         ; Читаем информацию на все разрядах PORTA
        MOVWF Reg_KNOPKI ; Запоминаем в вспомогательном регистре
        COMF    Reg_KNOPKI,W ; Конвертируем выражение
        ANDLW KNOPKI            ; Выделяем информацию от кнопок путем поразрядного логического «И» с константой KNOPKI – 00000111 – (где нули – информация затирается и остаются нужные нам разряды), результат оставляем в аккумуляторе
```

```
        ADDWF  PCL,F
        GOTO   EXIT_OPROS     ; 0 – сюда попадаем, если ничего не нажато
        GOTO   PROG_1         ; 1 – сюда попадаем, если нажата 1 кнопка
        GOTO   PROG_2         ; 2 – сюда попадаем, если нажата 2 кнопка
        GOTO   EXIT_OPROS     ; 3 – сюда попадаем, если нажата 1 и 2
кнопка
        GOTO   PROG_3         ; 4 – сюда попадаем, если нажата 3 кнопка
        GOTO   EXIT_OPROS     ; 5 – сюда попадаем, если нажата 1  и 3
кнопка
        GOTO   EXIT_OPROS     ; 6 – сюда попадаем, если нажата 2 и 3
кнопка
        GOTO   EXIT_OPROS     ; 7 – сюда попадаем, если нажаты все 3
кнопки

EXIT_OPROS
            GOTO MAIN         ; конец программы опроса кнопок

PROG_1
            BSF   LED_1       ; Включаем светодиод LED_1
            BCF   LED_2       ; Выключаем светодиод LED_2
            BCF   LED_3       ; Выключаем светодиод LED_3
            GOTO      MAIN

PROG_2      BSF LED_2         ; Включаем светодиод LED_2
            BCF LED_1         ; Выключаем светодиод LED_1
            BCF   LED_3       ; Выключаем светодиод LED_3
            GOTO MAIN         ; идем на начало

PROG_3      BSF LED_3         ; Включаем светодиод LED_3
            BCF LED_1         ; Выключаем светодиод LED_1
            BCF   LED_2       ; Выключаем светодиод LED_2
            GOTO MAIN         ; идем на начало

END
```

6. ВКЛЮЧЕНИЕ СВЕТОДИОДОВ ПО НОМЕРУ В ДВОИЧНОЙ СИСТЕМЕ



Программный код:

; В данной программе используются все комбинации нажатия кнопок.
; Если кнопки подсоединить по порядку, т.е. KEY_1 к PORTA,0 KEY_2 к PORTA,2 а KEY_3 к PORTA,2
; тогда состояние этих разрядов PORTA после его конвертации будет выглядеть след. образом.
; Состояние порта при нажатой кнопке KEY_1 - 00000110, после конвертации - 11111001
; После выделения нужных нам разрядов получим 00000001. т.е. нажатой кнопке 1 соответствует число 1
; Работа в PROTEUS: Включаем - светодиоды не светятся. Светодиоды светятся пока нажаты кнопки
; Нажимаем на кнопку KEY1 - светится LED_1,
; Нажимаем на KEY2 - светится LED_2,
; Нажимаем на KEY2 и кнопку KEY1 - светится LED_3
; Нажимаем на KEY3 - светится LED_4, и т.д.

```
LIST  P=PIC16F84A

#INCLUDE “P16F84A.INC”
```

; Конфигурация микроконтроллера: стандартный кварцевый генератор & сторожевой таймер выключен & будет производиться выдержка при включении питания & защита памяти программ выключена
; __CONFIG 0x3FF1
__CONFIG _FOSC_XT & _WDTE_OFF & _PWRTE_ON & _CP_OFF

ERRORLEVEL -207 ; Эта строчка говорит о том, что не надо выводить сообщение о предупреждении № 207
; Само предупреждение гласит, что метка, например, "MAIN" находится не на первой колонке строчки.

ERRORLEVEL -302 ; О предупреждении 302, но если вместо "MOVWF TRISA" записать "MOVWF TRISA^80h", (добавляя к названию регистра "^80", мы определяем место нахождения данного регистра) то данное предупреждение исчезнет.

```
#define     LED_1     PORTB,0     ; определяем расположение LED_1
#define     LED_2     PORTB,1     ; определяем расположение LED_2
#define     LED_3     PORTB,2     ; определяем расположение LED_3
#define     LED_4     PORTB,3     ; определяем расположение LED_4
#define     LED_5     PORTB,4     ; определяем расположение LED_5
#define     LED_6     PORTB,5     ; определяем расположение LED_6
#define     LED_7     PORTB,6     ; определяем расположение LED_7

#define     KEY_1     PORTA,0     ; Кнопка включения светодиода LED_1
#define     KEY_2     PORTA,1     ; Кнопка включения светодиода LED_2
#define     KEY_3     PORTA,2     ; Кнопка включения светодиода LED_3
#define     KNOPKI    b'00000111' ; Маска для выделения информации на тех
разрядах PORTA, к которым подключены кнопки
#define     Z_        STATUS,Z    ; Флаг нулевого результата

CBLOCK      20H       ; блок РОН начинается с адреса 20h
            Reg_1           ; объявляем регистр Reg_1 для задержки
            Reg_2           ; объявляем регистр Reg_2 для задержки
            Reg_KNOPKI      ; Регистр для работы с информацией от кнопок
ENDC                  ; конец блока

BANK0       MACRO           ; макрос перехода в BANK0
        BCF STATUS,RP0      ; сброс бита RP0
        BCF STATUS,RP1      ; сброс бита RP1
ENDM                        ; конец макроса

BANK1       MACRO           ; макрос перехода в BANK1
        BSF STATUS,RP0      ; установка бита RP0
        BCF STATUS,RP1      ; сброс бита RP1
```

```
ENDM                        ; конец макроса

        ORG 0               ; установка текущего адреса на 0
        GOTO    BEGIN       ; пропускаем фрагмент задержки
```

; Подпрограмму задержек можно автоматически сформировать с помощью программы задержек

```
DELAY                       ; начало сгенерированной задержки
        movlw     .221
        movwf     Reg_1
        movlw     .13
        movwf     Reg_2
        decfsz    Reg_1,F
        goto      $-1
        decfsz    Reg_2,F
        goto      $-3
        nop
        nop
        return    ; конец сгенерированной задержки

BEGIN                       ; начало настройки портов
        BANK1               ; переход в BANK1
        CLRF      TRISB     ; PORTB на вывод
        MOVLW     B'00000111'
        MOVWF     TRISA     ; PORTA на ввод
        BANK0               ; переход в BANK0
        CLRF      PORTA     ; Инициализация портов
        CLRF      PORTB
        CLRF      Reg_KNOPKI

MAIN
            MOVFW     PORTA     ; Читаем информацию на все разрядах
PORTA
            MOVWF     Reg_KNOPKI
            COMF      Reg_KNOPKI,W
            ANDLW     KNOPKI    ; Выделяем информацию от кнопок путем
поразрядного логического "И" с константой "KNOPKI" - 00011010 -, где нули
информация затирается и остаются нужные нам разряды, результат оставляем
в аккумуляторе
            CALL      DELAY
            MOVFW     PORTA
            MOVWF     Reg_KNOPKI
            COMF      Reg_KNOPKI,W
            ANDLW     KNOPKI
```

```
            ADDWF   PCL,F
            GOTO    EXIT_OPROS
            GOTO    PROG_1
            GOTO    PROG_2
            GOTO    PROG_3
            GOTO    PROG_4
            GOTO    PROG_5
            GOTO    PROG_6
            GOTO    PROG_7

EXIT_OPROS
            GOTO    MAIN

; Подпрограмма обработки нажатой кнопки KEY_1

PROG_1
            BSF  LED_1      ; Включаем светодиод LED_1
            BCF  LED_2      ; Выключаем светодиод LED_2
            BCF  LED_3      ; Выключаем светодиод LED_3
            BCF  LED_4      ; Выключаем светодиод LED_4
            BCF  LED_5      ; Выключаем светодиод LED_5
            BCF  LED_6      ; Выключаем светодиод LED_6
            BCF  LED_7      ; Выключаем светодиод LED_7
            GOTO MAIN       ; идем на начало

PROG_2      BSF LED_2       ; Включаем светодиод LED_2
            BCF LED_1       ; Выключаем светодиод LED_1
            BCF  LED_3      ; Выключаем светодиод LED_3
            BCF  LED_4      ; Выключаем светодиод LED_4
            BCF  LED_5      ; Выключаем светодиод LED_5
            BCF  LED_6      ; Выключаем светодиод LED_6
            BCF  LED_7      ; Выключаем светодиод LED_7
            GOTO MAIN       ; идем на начало

PROG_3      BSF LED_3       ; Включаем светодиод LED_3
            BCF LED_1       ; Выключаем светодиод LED_1
            BCF  LED_2      ; Выключаем светодиод LED_2
            BCF  LED_4      ; Выключаем светодиод LED_4
            BCF  LED_5      ; Выключаем светодиод LED_5
            BCF  LED_6      ; Выключаем светодиод LED_6
            BCF  LED_7      ; Выключаем светодиод LED_7
            GOTO MAIN       ; идем на начало

PROG_4
            BCF  LED_1      ; Выключаем светодиод LED_1
```

```
        BCF   LED_2      ; Выключаем светодиод LED_2
        BCF   LED_3      ; Выключаем светодиод LED_3
        BSF   LED_4      ; Включаем светодиод LED_4
        BCF   LED_5      ; Выключаем светодиод LED_5
        BCF   LED_6      ; Выключаем светодиод LED_6
        BCF   LED_7      ; Выключаем светодиод LED_7
        GOTO MAIN        ; идем на начало

PROG_5
        BCF   LED_1      ; Выключаем светодиод LED_1
        BCF   LED_2      ; Выключаем светодиод LED_2
        BCF   LED_3      ; Выключаем светодиод LED_3
        BCF   LED_4      ; Выключаем светодиод LED_4
        BSF   LED_5      ; Включаем светодиод LED_5
        BCF   LED_6      ; Выключаем светодиод LED_6
        BCF   LED_7      ; Выключаем светодиод LED_7
        GOTO MAIN        ; идем на начало

PROG_6
        BCF   LED_1      ; Выключаем светодиод LED_1
        BCF   LED_2      ; Выключаем светодиод LED_2
        BCF   LED_3      ; Выключаем светодиод LED_3
        BCF   LED_4      ; Выключаем светодиод LED_4
        BCF   LED_5      ; Выключаем светодиод LED_5
        BSF   LED_6      ; Включаем светодиод LED_6
        BCF   LED_7      ; Выключаем светодиод LED_7
        GOTO MAIN        ; идем на начало

PROG_7
        BCF   LED_1      ; Выключаем светодиод LED_1
        BCF   LED_2      ; Выключаем светодиод LED_2
        BCF   LED_3      ; Выключаем светодиод LED_3
        BCF   LED_4      ; Выключаем светодиод LED_4
        BCF   LED_5      ; Выключаем светодиод LED_5
        BCF   LED_6      ; Выключаем светодиод LED_6
        BSF   LED_7      ; Включаем светодиод LED_7
        GOTO MAIN        ; идем на начало

END
```

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. IC-Prog [Электронный ресурс]. – Электрон. дан. и прогр. – 2015. – Режим доступа: http://www.ic-prog.com. – Загл. с экрана.

2. MicroCHIP. Словарь терминов и сокращений [Электронный ресурс]. – Электрон. дан. – 2015. – Режим доступа: http://www.microchip.ru/?mid=3x2. – Загл. с экрана.

3. MicroCHIP. Статьи на русском языке [Электронный ресурс]. – Электрон. дан. – 2015. – Режим доступа: http://www.microchip.ru/lit/?mid=1x0. – Загл. с экрана.

4. MPASM. Руководство пользователя [Электронный ресурс]. – Электрон. дан. – 2001. – Режим доступа: http://www.microchip.ru/files/d-sheets-rus/mpasm.pdf. – Загл. с экрана.

5. MPLab [Электронный ресурс]. – Электрон. дан. – 2008. – Режим доступа: http://www.mplab.ru/. – Загл. с экрана.

6. MPLAB IDE. Интегрированная среда разработки для микроконтроллеров PICmicro компании Microchip Technology Incorporated [Электронный ресурс]. – Электрон. дан. – 2001. – Режим доступа: http://www.microchip.ru/files/d-sheets-rus/mplab_ide.pdf. – Загл. с экрана.

7. PIC16F62X. Однокристальные 8-разрядные FLASH CMOS микроконтроллеры компании Microchip Technology Incorporated [Электронный ресурс]. – Электрон. дан. – 2001. – Режим доступа: http://www.microchip.ru/files/d-sheets-rus/pic16f62x.pdf. – Загл. с экрана.

8. Предко, М. PIC-микроконтроллеры: архитектура и программирование: пер. с англ. / М. Предко. – М. : ДМК Пресс, 2010.

9. Тавернье К., PIC-микроконтроллеры. Практика применения: Пер. с фр. – М.: ДМК Пресс, 2004. – 272 с.: ил. (Серия «Справочник»).

Ссылка на список лекций: http://rabbit11web.ru/index.php/materialy-dlya-studentov/materialy-dlya-to-31-proektirovanie-tsifrovykh-sistem-vi-semestr/lektsii