PIC Microcontroller Report

PIC 16F684

This report briefly describes methods used in order to fulfil the requirements of the Assigment 2b.

2013

Alexander Drabek

1219460

2013-05-02

# Introduction

The given task was to develop an assembly program for PIC16F684 microcontroller.

Developed program should light each diode for half a second and then half a second delay starting from last diode D7 going up to first D0 and looping repeating forever. I have attempted to developed complex solution as well as expected simpler way of it. Code which fulfils the requirements is included in Appendix B also called a simple solution. Both programs are working and are well commented .

**Developed Solutions**

# **A complex solution**

The way I have tackled the problem was slightly different. Rather than developing programs which will execute command after a command in incremental order I have developed a complex solution based on loops and lookup tables for both TRISA and PORTA configuration.

|  |
| --- |
| C:\Users\photoNSA\Desktop\TRISA-lookup.png |
| Picture1 - One of two of lookup tables - TRISA |

Firstly, the program loads the initial configuration and it is defining and setting the values for counters. Secondly, I have created a couple of labels (like start\_loop) and using ‘GOTO’ command I have made the program looping around itself in a way that after loading each TRISA configuration it will loop 2 times blinking corresponding LEDs diodes. This pattern should be repeated 4 times in order to blink all LED diodes, because of stack overflow error I have written a simple temporary code for last two diodes. The last step is the process of looping forever which require resetting all counters to initial configuration and repeat whole program.

In appendix A I attached the assembly code.   
I realized that complex solution blinks diodes in a wrong order. From D0 to D7. In order to fulfil the requirements I have created another program called simple solution.

## Simple Solution

This program also loads initial configuration and reserving space in RAM for counters.  
In this program I have used definitions in order to make the code simpler to read. Program executes commands from the beginning to the end with an exception when it is calling a delay subroutine. This solution also follows the pattern of changing the configuration of TRISA for each of two LED diodes. The last step is the process of looping forever which repetition of the whole program.

I have commented the program in a way that is easy to notice the pattern that for each TRISA configuration we are blinking two LEDs diodes.  
Assembly code for the simple solution program is provided in Appendix B.

**Conclusion**

After developing both solutions I have understood that the **proposed** solution should be adjusted to the desired goal and should be developed in accordance with the device which would be programmed. Complex solution has an advantage of saving space on ROM memory.  
  
In comparison, the simple solution is taking slightly less space in program memory (4 lines) and consume more ROM memory.

Because of lookup tables and facing many problems I have gained better understanding of PIC microcontrollers as well knowledge about hardware.

# Appendix A

|  |
| --- |
| Assembly code of complex Solution |
| ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; Filename: Pic\_demonstration-Complex.asm  ; Author: Alexander Drabek  ; Date: 01.05.13  ; Description: This Assembler program for PIC16F684 turns ON and OFF LEDs from D7 to D0  ; with a delay of 0.5 second  ;Check if trisa reaching the 4th and PORTA eight leds.  ;To report:  ;1.separate PCL in Bank1 and Bank0  ;2.Lookup table and undaestanding of options of the code.  ;3.we are assuming to work in Bank0  ;4.overflow.  list p=16F684 ; List directive to define processor  #include <p16f684.inc> ; Processor specific variable definitions  \_\_CONFIG \_CP\_OFF & \_WDT\_OFF & \_PWRTE\_ON & \_INTRC\_OSC\_NOCLKOUT & \_MCLRE\_OFF & \_CPD\_OFF  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Assignment Statements \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ;Delay counters  COUNT1 EQU 20h ; Store COUNT1 at address 20h  COUNT2 EQU 21h ; Store COUNT2 at address 21h  normal EQU 22h ; Store normal at address 22h - responsible for general loop(4 times change trisa)  COUNT3 EQU 26h ;  ;Loops counters  PORTA\_Internal EQU 23h ;Store PORTA\_Internal at address 21h  COUNT\_LED\_NUMBER EQU 24h ; Store COUNT\_LED\_NUMBER at address 21h  COUNT\_TRISA\_NUMBER EQU 25h ; Store COUNT\_TRISA\_NUMBER at address 21h  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* "Start" of Program \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; Initial configuration  ;errorlevel -302 ; Disable bank switch warning  org 0x000 ; Processor reset vector  bcf STATUS,RP0 ; Bank 0 selected  movlw 07h ; Set RA<2:0> to digital and  movwf CMCON0 ; Comparators turned OFF  bsf STATUS,RP0 ; Bank 1 selected  clrf ANSEL ; Digital I/O selected  movlw B'00111111' ; Move in W - 0x3F - Set all I/O pins as digital inputs  movwf TRISA ; Configure I/O ports  clrf INTCON ; Disable all interrupts, clear all flags  bcf STATUS,RP0 ; Bank 0 selected  clrf PORTA ; Clear all outputs  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; Load values for counters.  AGAIN;reset the value  movlw B'11101010' ; First load a value of say, F0h in the W register  movwf COUNT2 ; Now move it to COUNT2 register  movlw 0x03; ; First load a value of say, F0h in the W register  movwf COUNT3 ; Now move it to COUNT2 register  movlw B'00000100' ; Load a value 5, in the W register  movwf normal ; Move it to normal  movlw B'00000010' ; First load a value of say, 1 in the W register  movwf PORTA\_Internal ; Now move it to PORTA\_Internal  movlw B'00000000' ; First load a value of say, 1 in the W register  movwf COUNT\_LED\_NUMBER ; Now move it to COUNT2 register  movlw B'00000000' ; First load a value of say, 1 in the W register  movwf COUNT\_TRISA\_NUMBER ; Now move it to COUNT2 register  goto start\_loop  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Lookup\_TRISA  movlw low TableTRISA  ADDWF COUNT\_TRISA\_NUMBER,0  movwf PCL ; Jump to Specified entry  TableTRISA RETLW B'00001111'; 15-DECIMAL ; Element 0 of lookup table  RETLW B'00101011'; 43-DECIMAL ; Element 1 of lookup table  RETLW B'00011011'; 27-DECIMAL ; Element 2 of lookup table  ; RETLW B'00111001'; 57 - overflow!  Lookup\_PORTA  movlw low TablePORTA  ADDWF COUNT\_LED\_NUMBER,0  movwf PCL ; Jump to specified entry  TablePORTA RETLW B'00010000' ; Sending HIGH to D0 -OK  RETLW B'00100000' ; Sending HIGH to D1 -ok  RETLW B'00010000' ; Sending HIGH to D2 -ok  RETLW B'00000100' ; Sending HIGH to D3 -ok  RETLW B'00100000' ; Sending HIGH to D4 -ok  RETLW B'00000100' ; Sending HIGH to D5 -ok  RETLW B'00000100' ; Sending HIGH to D6 -ok  RETLW B'00000010' ; Sending HIGH to D7 -ok  ;it is always loading first one of eac lookup tables!  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ;2 diods -could be done by flip flop values  start\_loop  ITERNAL\_TRISA\_LOOP ;Change the trisa after 2 diods ,do it till end of lookup table    CALL Lookup\_TRISA ; Call the table. -it should obtain the address for next memory cell with trisa config  bsf STATUS,RP0 ;Bank 1  movwf TRISA ;Move config to TRISA  bcf STATUS, RP0 ;Reset bank to 0  decfsz normal,1 ;Decrease the number of posible configuration in this iteration value of 5 -Not sure --it will omit las one (option 4)  goto INTERNAL\_LOOP\_2DIODS ; Blink 2 diods including delays  ;Temporary solution for overflow  bsf STATUS,RP0 ; Bank 1 selected  movlw B'00111001' ; TRISA for D6 and D7  movwf TRISA  bcf STATUS, RP0 ; Bank 0 selected  call DELAY  CALL Lookup\_PORTA ; Call the table. -it should obtain the address for next memory cell with PORTA config  movwf PORTA ;light this diod!  call DELAY ; for a half second  clrf PORTA ; Clear PORTA  incfsz COUNT\_LED\_NUMBER,1 ;increase LED NUmber! save into variable!  call DELAY  CALL Lookup\_PORTA ; Call the table. -it should obtain the address for next memory cell with PORTA config  movwf PORTA ;light this diod!  call DELAY ; for a half second  clrf PORTA ; Clear PORTA  ;end of temporary solution  goto AGAIN ;  INTERNAL\_LOOP\_2DIODS ;develop the delay ! +TRISA CHANGE  bcf STATUS, RP0 ;  call DELAY  CALL Lookup\_PORTA ; Call the table. -it should obtain the address for next memory cell with PORTA config  movwf PORTA ;light this diod!  call DELAY ; for a half second  clrf PORTA ; Clear PORTA  incfsz COUNT\_LED\_NUMBER,1 ;increase LED NUmber! save into variable!  decfsz PORTA\_Internal,1 ; it should have value of 2 XD  goto INTERNAL\_LOOP\_2DIODS ; only 1 repeat!  incfsz COUNT\_TRISA\_NUMBER,1 ;increase the TRISA config number  movlw B'00000010' ; First load a value of say, 1 in the W register  movwf PORTA\_Internal ; Now move it to COUNT2 register  goto start\_loop ;- Loop forever -  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; DELAY Subroutine  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; Generate a delay period  DELAY  LOOP1 decfsz COUNT1,1 ; Decrement COUNT1 and skip next instruction if zero  goto LOOP1 ; else loop back to LOOP1  decfsz COUNT2,1 ; Decrement COUNT2 and skip next instruction if zero  goto LOOP1 ; else loop back to LOOP1  decfsz COUNT3,1 ; Decrement COUNT1 and skip next instruction if zero  goto LOOP1 ; else loop back to LOOP1  movlw 0x03 ; Reload the second counter for the next iteration  movwf COUNT3 ; Decrement COUNT3 and skip next instruction if zero  movlw 0xF0 ; Reload the second counter for the next iteration  movwf COUNT2  return ; else loop back to LOOP1    ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  end  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |

## Appendix B

|  |
| --- |
| Assembly code of simple solution |
| ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; Filename: FlashD7-D0\_LEDS.asm  ; Author: Alexander Drabek  ; Date: 26.02.13  ; Description: PIC16F684 assembly program which blinks LEDs from D7 to D0 in infinite loop,  ; There is 0.5 second delay for ON and OFF period.  ;  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  list p=16F684 ; list directive to define processor  #include <p16f684.inc> ; processor specific variable definitions  \_\_CONFIG \_CP\_OFF & \_WDT\_OFF & \_PWRTE\_ON & \_INTRC\_OSC\_NOCLKOUT & \_MCLRE\_OFF & \_CPD\_OFF  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Reserving space for counters in memory \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  COUNT1 EQU 20h ; Available GPR at address 20h used for storing COUNT1  COUNT2 EQU 21h ; Available GPR at address 21h used for storing COUNT2  COUNT3 EQU 22h ; Available GPR at address 22h used for storing COUNT3  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* using #define clause to make program simply to understand \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; Define the Inputs/Outputs-TRISA configuration  #define TRISA\_CONFIG\_D6\_D7 B'00111001' ; D7 and D6 connected between RA2 pin and RA1 pin  #define TRISA\_CONFIG\_D4\_D5 B'00011011' ; D5 and D4 connected between RA2 and RA5  #define TRISA\_CONFIG\_D2\_D3 B'00101011' ; D3 and D2 connected between RA4 and RA2  #define TRISA\_CONFIG\_D0\_D1 B'00001111' ; D1 and D0 connected between RA4 and RA5  ; Define the 'data output' PORTA setting for switching ON the LEDs D2 to D3  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Start of Program \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; Initial configuration - Common for all programs for CIS018-1  org 0x000 ; Processor reset vector  bcf STATUS,RP0 ; Bank 0 selected  movlw 07h ; Set RA<2:0> to digital and  movwf CMCON0 ; Comparators turned OFF  bsf STATUS,RP0 ; Bank 1 selected  clrf ANSEL ; Digital I/O selected  movlw B'00111111' ; Move in W - 0x3F - Set all I/O pins as digital inputs  movwf TRISA ; Configure I/O ports  clrf INTCON ; Disable all interrupts, clear all flags  bcf STATUS,RP0 ; Bank 0 selected  clrf PORTA ; Clear all outputs  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; To preload a counters values;  movlw B'11101010' ; Load the value of F0h in the W register  movwf COUNT2 ; Now move it to COUNT2  movlw 0x02 ; Load the value of 0x02 into W register  movwf COUNT3 ; Now move it to COUNT3  ; COUNT1 will have default value of FFh  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ;Starting point of the main Program  START\_LOOP  ;Configuration for TRISA  bsf STATUS,RP0 ; Bank 1 selected  movlw TRISA\_CONFIG\_D6\_D7 ; Configure LEDs D6 and D7 in TRISA  movwf TRISA ;Load the configuration to TRISA  bcf STATUS, RP0 ; Bank 0 selected - Clearing RP0 bit  ;Lighting next diods  movlw B'00000010' ;Write Porta configuration for D5 into W Register  movwf PORTA ;Send content of W register into PORTA to light up LED D7  call DELAY ;Wait 0.5 second  clrf PORTA ;Clear PORTA=Turn OFF the LED  call DELAY ;Wait 0.5 second  movlw B'00000100' ;Write Porta configuration for D5 into W Register  movwf PORTA ;Send content of W register into PORTA to light up LED D6  call DELAY ;Wait 0.5 second  clrf PORTA ;Clear PORTA=Turn OFF the LED  call DELAY ;Wait 0.5 second  ; Setting the next I/O -TRISA configuration  bsf STATUS,RP0 ; Bank 1 selected  movlw TRISA\_CONFIG\_D4\_D5 ; Configure LEDs D4 and D5 in TRISA  movwf TRISA ;Load the configuration to TRISA  bcf STATUS, RP0 ; Bank 0 selected  ;Lighting next diods  movlw B'00000100' ;Write Porta configuration for D5 into W Register  movwf PORTA ;Send content of W register into PORTA to light up LED D5  call DELAY ;Wait 0.5 second  clrf PORTA ;Clear PORTA =Turn OFF the LED  call DELAY ;Wait 0.5 second  movlw B'00100000' ;Write Porta configuration for D4 into W Register  movwf PORTA ;Send content of W register into PORTA to light up LED D4  call DELAY ;Wait 0.5 second  clrf PORTA ;Clear PORTA =Turn OFF the LED  call DELAY ;Wait 0.5 second  ; Setting the I/O -TRISA configuration  bsf STATUS,RP0 ;Bank 1 selected  movlw TRISA\_CONFIG\_D2\_D3 ;Configure LEDs D2 and D3 in TRISA  movwf TRISA ;Move the loaded configuration from W Register to TRISA  bcf STATUS, RP0 ; Bank 0 selected  ;Lighting next diods  movlw B'00000100' ; Write Porta configuration for D3 into W Register  movwf PORTA ; Send content of W register into PORTA to light up LED D3  call DELAY ; Wait 0.5 second  clrf PORTA ; Clear PORTA =Turn OFF the LED  call DELAY ;Wait 0.5 second  movlw B'00010000' ; Write Porta configuration for D2 into W Register  movwf PORTA ; Send content of W register into PORTA to light up LED D2  call DELAY ;Wait 0.5 second  clrf PORTA ; Clear PORTA =Turn OFF the LED  call DELAY ;Wait 0.5 second  ;Setting the I/O -TRISA configuration  bsf STATUS,RP0 ; Bank 1 selected  movlw TRISA\_CONFIG\_D0\_D1 ; TRISA Configuration for LEDs D0 and D1  movwf TRISA ;Move the loaded configuration from W Register to TRISA  bcf STATUS, RP0 ; Bank 0 selected  ;Lighting next diods  movlw B'00100000' ; Write Porta configuration for D1 into W Register  movwf PORTA ; Send content of W register into PORTA to light up LED D1  call DELAY ;Wait 0.5 second  clrf PORTA ; Clear PORTA =Turn OFF the LED  call DELAY ;Call subroutine which cause delay of 0.5 second  movlw B'00010000' ; Write Porta configuration for D0 into W Register  movwf PORTA ; Send content of W register into PORTA to light up LED D0  call DELAY ;Wait 0.5 second  ; Sending data through PORTA to switch OFF LED D0  clrf PORTA ; Clear PORTA - led goes OFF  call DELAY ;Wait 0.5 second  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  goto START\_LOOP ; Infinity Loop  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; DELAY Subroutine  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ; Generate a delay period  DELAY  LOOP1 decfsz COUNT1,1 ; Decrement COUNT1 and skip next instruction if zero  goto LOOP1 ; else loop back to LOOP1  decfsz COUNT2,1 ; Decrement COUNT2 and skip next instruction if zero  goto LOOP1 ; else loop back to LOOP1  decfsz COUNT3,1 ; Decrement COUNT3 and skip next instruction if zero  goto LOOP1 ; else loop back to LOOP1  movlw B'11101010' ; First load a value of say, F0h in the W register  movwf COUNT2 ; Now move it to COUNT2 register  movlw 0x03 ; Reload the third counter for the next iteration  movwf COUNT3 ; Move the value from W register to COUNT3  return  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  end  ;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |