DTMF Remote Control Decoder

Version 1.4

This kit provides an interface between any audio path capable of handling Dual Tone Multiple Frequency (DTMF) coding and switched outputs using relays.

Features

A 4 digit PIN code enables access to the unit, this PIN code can be changed remotely or reduced to a single *. The PIN can be reset to the default by manual intervention.

A timeout function on the PIN code which can also be remotely changed to between 5 seconds and 255 seconds (4:15 minutes).

A command sequence end function that overrides the timeout function to prevent “follow on”.

Individual control of up to 8 relays comprising Operate, Release,Timed Toggling and Clock operation.

A full 16 DTMF keypad is required for full access operation although operation with a normal telephone style 12 key unit can be arranged.

DTMF tone reception is acknowledged by a tone. Success tone (after the PIN has been entered) will be at 500Hz, and Fail tone at 250Hz.

Commands

- Using this key at any point aborts the current command and resets the unit requiring the PIN code to be re-entered. Note: the # key cannot be used within a PIN code.

PIN code - After entering the required 4 digit PIN code the following commands are available;

Relay control – A digit between 1 and 8 selects a relay for the following digit entered to control,
the second digit can be;

0 – Release relay
1 – Operate relay
2 – Toggle relay using the relay toggle times.
3 – Switch using Clock times

Any other value will cause the command to abort and the system to reset, requiring the PIN to be re-entered.

Change PIN – The “A” key followed by a further 4 keys (other than #), these 4 digits must then be repeated before the new PIN is saved, once saved the system resets requiring the new PIN to be entered before other commands can be issued. The default PIN as supplied is “CCCC”. If a PIN of four Stars “****” is entered then the PIN function is reduced to a single * , this is primarily for those situations where security is not a concern.
Example: CCCC A 14AD 14AD
If the new PIN is entered incorrectly the second time then the system will reset with no changes.
If the PIN is forgotten, it can be manually reset by, powering down, placing the LINK in position and powering up, wait for the Beep to finish. The LINK can then be removed and the default PIN of “CCCC” will have been restored.

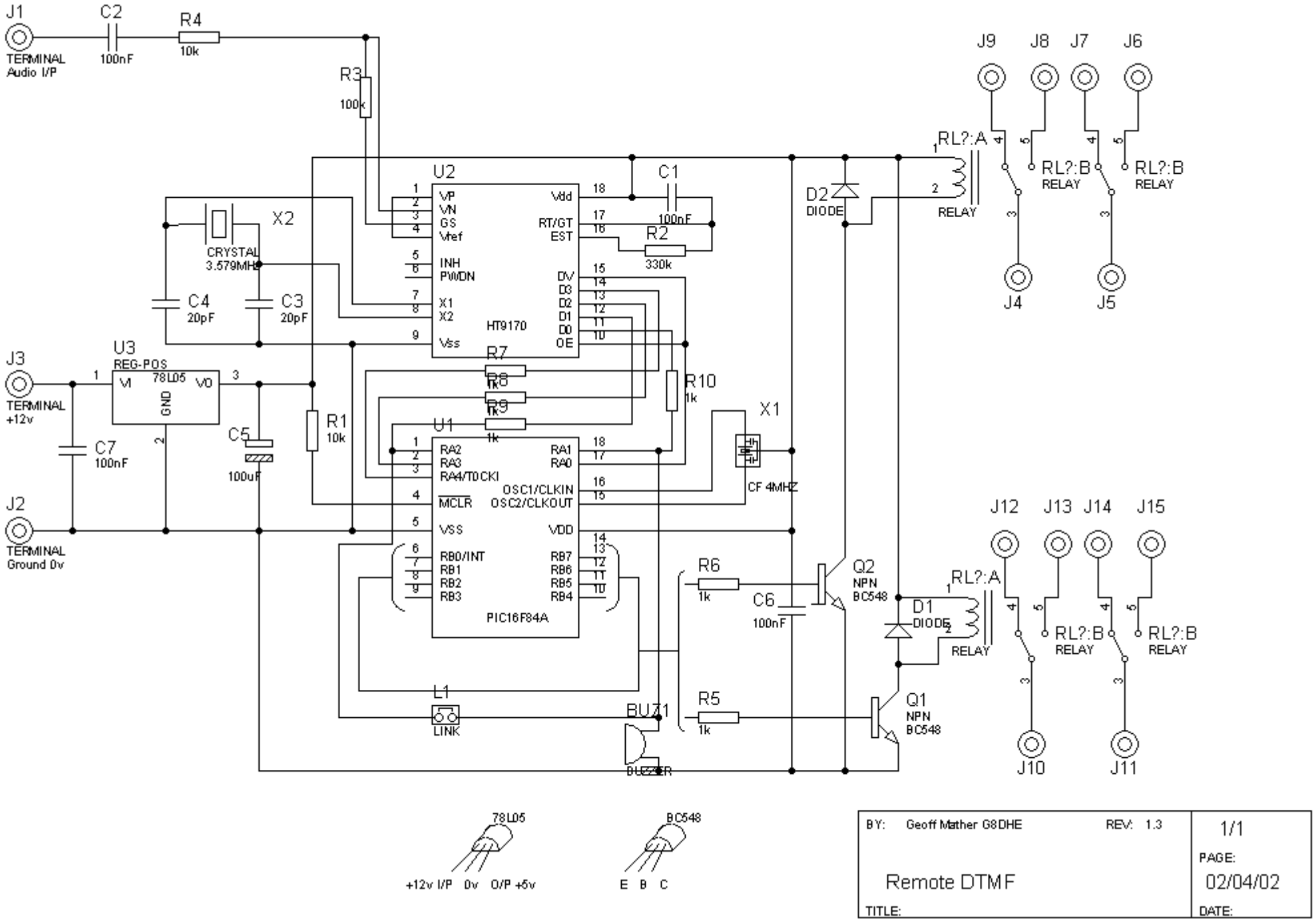
Change Command Timeout – The “B” key followed by the number of seconds required, between 5 and 255, ending with the “*” key. The new timeout period will come into operation after entering the PIN next time.

Change Relay Toggle times – The “C” key followed by the number of seconds the relay(s) are to operate for, followed by “*”, followed by the number of seconds the relay(s) are to be released for followed by “*”.

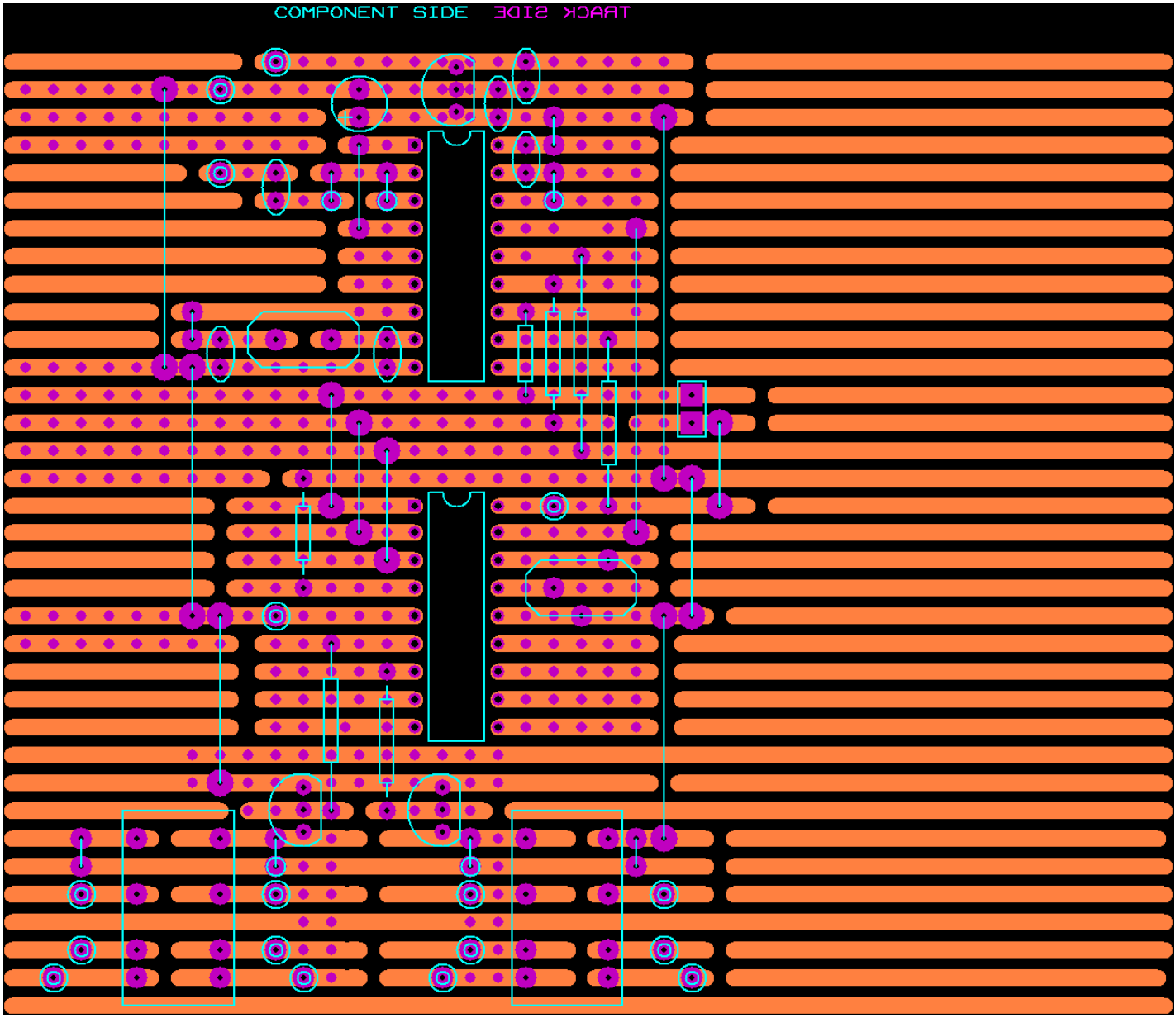
When using Relay Toggle, adding a relay to the group of relays to be switched, will cause a new cycle to begin immediately. A relay can be removed from the group of relays being switched, by either using the Relay Toggle a second time in which case the relay will stay in its current setting, or by using the Release command or Operate command which will force the relay to the requested position.

Change Clock times – The “D” key followed by the current time in 24:00 format, followed by the On time in 24:00 format and finally the Off time in 24:00 format. I.e. D 1339 0500 1900 would set the current time to 1:39 PM the On time at 5:00 AM and the Off time to 7:00 PM. To enable a relay to use the Clock times the particular relays then need to be selected.

When using the Clock times use of the Operate, Release or Toggle functions is permitted at any time without altering the Clock mode. To exit clock mode for a relay then the Relay 3 command is used a second time. Any number of relays can be set to operate at the specified times.



BY: Geoff Mather G8DHE	REV: 1.3	1/1
Remote DTMF		PAGE: 02/04/02
TITLE:		DATE:



D:\Program Files\FED\WIZPIC\Projects\DTMFRemote\DTMFRemote.HEX

```
:020000040000FA
:0600000000008A01162831
:100008008C00030E83018D0004088E0049280B1D07
:100018000F288F140B110E0884000D0E83008C0E10
:100028000C0E090083018B01840181018F012030AE
:100038008B00FF3085000030860083164130810038
:10004800FF3085000030860083121122 4A224A20A0
:100058008B171E220F1C3228C8210F108F1C37281F
:100068005C22D4218F100F1D3B2820210F118F1DDA
:100078003F2861218F110F1E432882210F1214225D
:100088008F1E482812218F125C280B28AA01AB0169
:10009800AD01AE01AF01B001B501B101B201B301CB
:1000A800B401BB01BA01B901B701B801B601080032
:1000B800B41F2D280C302A020319FD28B319BA28B9
:1000C800331AC128331DD728B31AF528331BF5284E
:1000D800B31B8329AF0803199528AA03031D7A289F
:1000E8002F0886042F09B005AF01E728AA03031DCE
:1000F80085282F08B0062F0932140130B100AF014E
:10010800E728AA03031D8C282F08B506AF01E728A6
:1001180007302A02031DFD282F098605B005AF0107
:10012800E7280D302A02031D9B28B315E7280E3057
:100138002A02031DA2283317AE01E7280F302A022E
:10014800031DAA28B3160230AB00E728AA08031D2E
:10015800B328B3170630AB000230AC00E72803140D
:10016800AF0D0318FD28AA0BB428E7283308073970
:10017800C3002A083222B30AE72833080339043EA9
:10018800C3002822AA02031DFD28B30A331DE7284D
:100198000430C300C3034315282243113222C30885
:1001A800031DCE28FD2833080339C30028220F3940
:1001B8002A02031DFD28330AB3000339013C031941
:1001C800F82133190D21B41334102D2805302E02CF
:1001D800031CFD280830C3002E083222AE01331359
:1001E800E7283C210318FB280319E7280028031DEA
:1001F8000029592134142D28B31EEA282B08083861
:10020800C3002E083222AE01AB03031D0C29B31222
:10021800E7280830C3002822AD000800831685149B
:1002280083120130 0B22573006220310050CAA0056
:10023800B417F830B4050800AD0803192629AD0B2A
:1002480026295921B00803190800B103031D080025
:10025800321833293009860509 30C3003729300898
:1002680086040A30C3002822B1000130B206080013
:100278000B302A02031953290A302A02031C472982
:10028800031D5629AA0103102E08AE0DAE0DAE0DA2
:10029800AE07AE072A08AE0703100315080000314BB
:1002A800031108000314031508 00AA01AB01AD01EE
:1002B800AE01AF01B301B4010800BA0A3C303A02FA
:1002C80003197C2138083A02031D712939083B02B9
:1002D800031D71293508860408003 6083A02031DF3
:1002E8000800370 83B02031D0800350986050800 89
:1002F800BA01BB0A18303B020319BB0108000A30D7
:100308002A02031C8A29031D5629AA0103102E0854
:10031800AE0DAE0DAE0DAE07AE072A08AE07AC0B9C
:10032800E728BB302B0784002E0880000230AC0081
:10033800AE01AB0BE7284108173C031CFD28400819
:100348003B3C031CFD283F08173C031CFD283E08C6
:100358003B3C031CFD283D08173C031CFD283C08BA
:100368003B3C031CFD280630AB00BB302B07840048
:1003780000 08AE00063084022E088000AB0BB929B5
:100388004A22B501B313E728331ACC29331D341494
:10039800B415B414C2013411831685 1083120800F1
:1003A800B41C080004303418B406341908000230AC
:1003B8008506C20BE029E829851CE529051DB4112D
:1003C80008000519B4110800F830B405831685141F
:1003D8008312B41D0800C3010430AB000A30322276
:1003E800C30AAB0BF2290800C30128220B3C031DEA
:1003F8000800C30A431DF9293308F8390438B30043
:10040800080000000000900900B082A08009100E6
:10041800F0300722910B0C2A0800FF3093000800E7
:1004280005080139031A2A131C8F161310031D06
:10043800131408000508013903 1D242A93180F1402
:10044800931003 1D9314080083164308831289003 0
:1004580083160814831208088312080083128800 80
:1004680083164308831289008 31603108B1B031419
:100478008B130815553089 00AA308900881488180C
:10048800432A081103188B1783120800831650227 9
:1004980055225922831208 00A101A201A301A40137
:1004A8000800A501A601A7010800A801A9010800E4
:1004B8008316A90AA80FA903D0302802031D6A2AA7
:1004C80003302902031D6A2A0F155922A50F6F2A26
:1004D800A60F6F2AA70AE1302502031D7E2AE43001
:1004E8002602031D7E2A00302702031D7E2A8F154F
:1004F80055225922A10F852AA20F852AA30F852AE2
:10050800A40ADA302102031D972AA4302202031D0F
:10051800972A35302302031D972A00302402031D31
:0A052800972A0F164A2A83120800D2
:00000001FF
```

```
;**** File : D:\PROGRA~1\FED\WIZPIC\P16F84.INC ****
                    00000 #include "D:\PROGRA~1\FED\WIZPIC\P16F84.inc"
                    00001         LIST
                    00002 ; P16F84.INC  Standard Header File, Version 2.00    Microchip Technology, Inc.
                    00137
                    00002
                    00003 #define APROCFREQ D'4011200'
                    00004
                    00005 #define BITSIZE D'14'
                    00006 #define BOOTADDRESS D'0'
                    00007 #define FIRSTRAM H'0C'
                    00008 #define LASTRAM H'4F'
                    00009 #define HASOSCCAL 0
                    00010 #define nPAGESRAM 1
                    00011 #define nPAGESROM 1
                    00012 ;
                    00013 ; Application designer globals
                    00014 ;
                    00015 vINTCON=0               ; Value loaded into INTCON during initialisation
                    00016 vOPTION_REG=0x7f        ; Value loaded into option register during intialisation
                    00017 vPIE1=0                 ; Value loaded into PIE1 during intialisation
                    00018 vPIE2=0                 ; Value loaded into PIE2 during intialisation
                    00019 vTRISA=0xff             ; Value loaded into TRISA during initialisation
                    00020 vTRISB=0xff             ; Value loaded into TRISA during initialisation
                    00021 vTRISC=0xff             ; Value loaded into TRISA during initialisation
                    00022 vTRISD=0xff             ; Value loaded into TRISA during initialisation
                    00023 vTRISE=0X07             ; Value loaded into TRISA during initialisation
                    00024 vPORTA=0xff             ; Value loaded into PORTA during initialisation
                    00025 vPORTB=0xff             ; Value loaded into PORTA during initialisation
                    00026 vPORTC=0xff             ; Value loaded into PORTA during initialisation
                    00027 vPORTD=0xff             ; Value loaded into PORTA during initialisation
                    00028 vPORTE=0xff             ; Value loaded into PORTA during initialisation
                    00029
                    00030 ;
                    00031 ; Macros to save & restore during an interrupt
                    00032 ;
                    00033
                    00034 INTSAVE         macro
                    00035                  movwf W_TEMP            ; Copy W to TEMP register
                    00036                  swapf STATUS,w          ; Swap status to be saved into W
                    00037                  clrf STATUS             ; Bank 0
                    00038                  movwf STATUS_TEMP       ; Save status
                    00039                  movf FSR,w              ; Copy FSR to W
                    00040                  movwf FSR_TEMP          ; Copy FSR from W to FSR_TEMP
                    00041                  #if (1 >1)
                    00042                   movf PCLATH, W          ; Only required if using pages 1, 2 and/or 3
                    00043                   movwf PCLATH_TEMP       ; Save PCLATH into W
                    00044                   clrf PCLATH             ; Page zero, regardless of current page
                    00045                  #endif
                    00046                 endm
                    00047
                    00048 INTRECALL       macro
                    00049                  #if (1 >1)
                    00050                   movf  PCLATH_TEMP,w    ; Restore PCLATH
                    00051                   movwf PCLATH           ; Move W into PCLATH
                    00052                  #endif
                    00053                  movf FSR_TEMP,w       ;Whats wrong with restoring FSR? cqg8dhecq
                    00054                  movwf        FSR             ;Whats wrong with restoring FSR? cqg8dhecq
                    00055                  swapf STATUS_TEMP,w     ; Swap STATUS_TEMP register into W
                    00056                  movwf STATUS            ; Move W into STATUS register
                    00057                  swapf W_TEMP,F          ; Swap W_TEMP
                    00058                  swapf W_TEMP,w          ; Swap W_TEMP into W
                    00059                 endm
                    00060
                    00061 ;
                    00062 ; Long call macro - this sets the paging bits, it assumes PCLATH
                    00063 ; is currently clear, and only sets PCLATH if the label is unknown (i.e. a
                    00064 ; forward reference), or it is known and is not in Page 0. It is therefore more
                    00065 ; efficient than Microchip's lcall macro which always sets the paging bits
                    00066 ;
                    00067 ; It only sets the bits if there is more than 1 page of ROM.
                    00068 ;
                    00069 longcall        macro n
                    00070                  if (1 >1)
                    00071                     if ((!n) || (n>$) || n>=0x800)       ; Must set paging bits
                    00072                      if (n&0x800)
                    00073                       bsf PCLATH,3
                    00074                      else
                    00075                       bcf PCLATH,3
                    00076                      endif
                    00077                      if (1 >2)     ; Test 2nd ROM PAGES
                    00078                       if (n&0x1000)
                    00079                        bsf PCLATH,4
                    00080                       else
                    00081                        bcf PCLATH,4
                    00082                       endif
                    00083                      endif
                    00084                     endif
                    00085                    endif
                    00086                    call n                ; One PAGES of ROM so simple call
                    00087                 endm
                    00088 ;
                    00089 ; This is different, it sets the PCLATH bits on the assumption that they
                    00090 ; are not already known
                    00091 ;
                    00092 ADSetPCLATH    macro n                       ; Set PCLATH bits
                    00093                if (1 >1)
                    00094                 if (n&0x800)
                    00095                   bsf PCLATH,3
                    00096                 else
                    00097                   bcf PCLATH,3
                    00098                 endif
                    00099
                    00100                 if (1 >2)
                    00101                  if (n&0x1000)
                    00102                    bsf PCLATH,4
                    00103                  else
```

```
                  00104                   bcf PCLATH,4
                  00105                  endif
                  00106                 endif
                  00107
                  00108                endif
                  00109               endm
                  00110
                  00111 ;
                  00112 ; Clear PCLATH if it was set in a call, if n is 0 then always clear it
                  00113 ; only clear it in any case if more than 1 page of ROM
                  00114 ;
                  00115 ;ClearPCLATH     macro n
                  00116 ;               if (((n>H'7FF') || (n==0)) && (nPAGESROM>1))
                  00117 ;                #if n&0x800
                  00118 ;                 bcf PCLATH,3
                  00119 ;                #endif
                  00120 ;                #if n&0x800
                  00121 ;                 bcf PCLATH,4
                  00122 ;                #endif
                  00123 ;               endif
                  00124 ;              endm
                  00125 ClearPCLATH     macro n                         ;cqg8dhecq Replacment macro
                  00126                 if (((n>=H'800') || (n==0)) && (1 >1))
                  00127                  #if n==0
                  00128                   clrf PCLATH
                  00129                   else
                  00130                    #if n&0x800
                  00131                     bcf PCLATH,3
                  00132                    #endif
                  00133                    #if n&0x1000
                  00134                     bcf PCLATH,4
                  00135                    #endif
                  00136                  endif
                  00137                 endif
                  00138                endm
                  00139
                  00140 ;
                  00141 ; Variables for interrupts
                  00142 ;
            00158                 cblock H'0C'
000C        00159                  W_TEMP                 ; Save W in interrupt
000C        00160                  STATUS_TEMP            ; Save STATUS in interrupt
000C        00161                  FSR_TEMP               ; Save FSR in interrupt
000C                  00162                endc
                  00168
                  00169 ;
                  00170 ; Macros for page switching variables
                  00171 ; Note we assume that all routines set RP0 and RP1 back to 0 before returning
                  00172 ;
                  00173 ADSetRP               macro n
                  00174                 if n&0x80
                  00175                  bsf STATUS,RP0
                  00176                 endif
                  00177                 if n&0x100
                  00178                  bsf STATUS,RP1
                  00179                 endif
                  00180                endm
                  00181
                  00182 ADClearRP     macro n
                  00183                 if n&0x80
                  00184                  bcf STATUS,RP0
                  00185                 endif
                  00186                 if n&0x100
                  00187                  bcf STATUS,RP1
                  00188                 endif
                  00189                endm
                  00190
                  00191 ;
                  00192 ; Set IRP for indirect access
                  00193 ;
                  00194 ADSetIRP      macro n
                  00195                 if 1 >2
                  00196                  if (n&0x100)
                  00197                   bsf STATUS,IRP
                  00198                  else
                  00199                   bcf STATUS,IRP
                  00200                  endif
                  00201                 endif
                  00202                endm
                  00203
                  00204 ;
                  00205 ; Correct a pointer to variables so that they all fit in the same memory block
                  00206 ;
                  00207 FITINBLOCK    macro Start,Size
                  00208 ;     #if ((Start&0x7f)+Size)>0x80 ;Fails when last assigned address is 7F
                  00209       #if (((Start&0x7f)<0x20) || (((Start&0x7f)+Size-1)>0x7F))
                  00210                                                   ;Works for all values cqg8dhecq
                  00211          Start=((Start+0x80)&0x180)+0x21
                  00212       #endif
                  00213
                  00214                endm
                  00215 ;
                  00216 ; Force a function of size : allow to fit into one page of ROM
                  00217 ;
                  00218 MFORCEPAGE    macro allow
                  00219                local next
                  00220                next=($+allow+32)&0xf800
                  00221                if ($&0xf800)!=next
                  00222                 org next
                  00223                endif
                  00224                endm
                  00225
                  00226 ;
                  00227 ; This macro delays an exact
                  00228 ; number of clock cycles between
                  00229 ; 1 at minimum or 186420 at max
                  00230 ;
                  00263 else          ; Version for one ROM page only
                  00264 ;
                  00265 ; This macro delays an exact
                  00266 ; number of clock cycles between
```

```
00267 ; 8 at minimum or 186420 at max
00268 ;
00269              Delv=0
00270 DELAY macro Cyc
00271        if (Cyc<D'8' || Cyc>D'186420')
00272         SmallCyc=Cyc
00273         if (SmallCyc&1)
00274          nop
00275          SmallCyc-=1
00276         endif
00277         while(SmallCyc)
00278          goto Del#v(Delv)
00279          Del#v(Delv)
00280          Delv=Delv+1
00281          SmallCyc-=2
00282         endw
00283        else
00284         SmallCyc=Cyc
00285         if Cyc>D'775'
00286          BigCyc=(Cyc-D'730')
00287          LoopDelay=BigCyc/D'728'
00288          movlw LoopDelay+1
00289          call BigDel
00290          SmallCyc=Cyc-(D'730'+LoopDelay*D'728'+3)
00291         endif
00292         LoopDelay=(SmallCyc-3)-5      ; Delay<=775 Cyc
00293         movlw LoopDelay/3+1
00294         call Delay0-LoopDelay%3
00295        endif
00296 endm
00297
00298 endif
00299 #define PortOutnCopy D'8'
00300 #define EdgenCopy D'2'
00301 #define PortInnCopy D'5'
00302 ; **********
00303 ;
00304 ; DELAY - macro to delay any exact time
00305 ;
00306 ; **********
00307 #define DELAYUsed 1
00308 ; **********
00309 ;
00310 ; Edge Detector
00311 ;
00312 ; **********
00313 #define EdgeUsed 1
00314 #define Edge1Used 1
00315 #define GotEdge1Used 1
00316 #define GotEdge1Flag Flag1
00317 #define GotEdge1FlagBit 0
00318 #define SelEdge1 D'0'
00319 #define Edge1Port PORTA
00320 #define Edge1Bit 0
00321 ; **********
00322 ;
00323 ; EEPROM
00324 ;
00325 ; **********
00326 #define EEPROMUsed 1
00327 ; **********
00328 ;
00329 ; Timer 0
00330 ;
00331 ; **********
00332 #define Tmr0Used 1
00333 #define Tmr0IntUsed 1
00334 #define OfUsed 1
00335 #define OfFlag Flag1
00336 #define OfFlagBit 1
00337 #define UsePS D'1'
00338 #define PSD D'4'
00339 #define CRE D'1'
00340 #define UseOsc D'1'
00341 ; **********
00342 ;
00343 ; Hours, Minutes and Seconds
00344 ;
00345 ; **********
00346 #define HMSUsed 1
00347 #define SecPassUsed 1
00348 #define SecPassFlag Flag1
00349 #define SecPassFlagBit 2
00350 #define MinPassUsed 1
00351 #define MinPassFlag Flag1
00352 #define MinPassFlagBit 3
00353 #define HourPassUsed 1
00354 #define HourPassFlag Flag1
00355 #define HourPassFlagBit 4
00356 ; **********
00357 ;
00358 ; Port Driver
00359 ;
00360 ; **********
00361 #define PortOutUsed 1
00362 #define PortOut7Used 1
00363 #define InitialValue7 D'0'
00364 #define Out7Port PORTB
00365 #define Out7Bit 7
00366 ; **********
00367 ;
00368 ; Port Driver
00369 ;
00370 ; **********
00371 #define PortOut6Used 1
00372 #define InitialValue6 D'0'
00373 #define Out6Port PORTB
00374 #define Out6Bit 6
00375 ; **********
00376 ;
00377 ; Port Driver
```

```
00378 ;
00379 ; **********
00380 #define PortOut5Used 1
00381 #define InitialValue5 D'0'
00382 #define Out5Port PORTB
00383 #define Out5Bit 5
00384 ; **********
00385 ;
00386 ; Port Driver
00387 ;
00388 ; **********
00389 #define PortOut4Used 1
00390 #define InitialValue4 D'0'
00391 #define Out4Port PORTB
00392 #define Out4Bit 4
00393 ; **********
00394 ;
00395 ; Port Driver
00396 ;
00397 ; **********
00398 #define PortOut3Used 1
00399 #define InitialValue3 D'0'
00400 #define Out3Port PORTB
00401 #define Out3Bit 3
00402 ; **********
00403 ;
00404 ; Port Driver
00405 ;
00406 ; **********
00407 #define PortOut2Used 1
00408 #define InitialValue2 D'0'
00409 #define Out2Port PORTB
00410 #define Out2Bit 2
00411 ; **********
00412 ;
00413 ; Port Driver
00414 ;
00415 ; **********
00416 #define PortOut1Used 1
00417 #define InitialValue1 D'0'
00418 #define Out1Port PORTB
00419 #define Out1Bit 1
00420 ; **********
00421 ;
00422 ; Port Driver
00423 ;
00424 ; **********
00425 #define PortOut0Used 1
00426 #define InitialValue0 D'0'
00427 #define Out0Port PORTB
00428 #define Out0Bit 0
00429 ; **********
00430 ;
00431 ; Edge Detector
00432 ;
00433 ; **********
00434 #define Edge0Used 1
00435 #define GotEdge0Used 1
00436 #define GotEdge0Flag Flag1
00437 #define GotEdge0FlagBit 5
00438 #define SelEdge0 D'1'
00439 #define Edge0Port PORTA
00440 #define Edge0Bit 0
00441 ; **********
00442 ;
00443 ; Port Input
00444 ;
00445 ; **********
00446 #define PortInUsed 1
00447 #define PortIn4Used 1
00448 #define In4Port PORTA
00449 #define In4Bit 0
00450 ; **********
00451 ;
00452 ; Port Input
00453 ;
00454 ; **********
00455 #define PortIn3Used 1
00456 #define In3Port PORTA
00457 #define In3Bit 1
00458 ; **********
00459 ;
00460 ; Port Input
00461 ;
00462 ; **********
00463 #define PortIn2Used 1
00464 #define In2Port PORTA
00465 #define In2Bit 2
00466 ; **********
00467 ;
00468 ; Port Input
00469 ;
00470 ; **********
00471 #define PortIn1Used 1
00472 #define In1Port PORTA
00473 #define In1Bit 3
00474 ; **********
00475 ;
00476 ; Port Input
00477 ;
00478 ; **********
00479 #define PortIn0Used 1
00480 #define In0Port PORTA
00481 #define In0Bit 4
00482 ; **********
00483 ;
00484 ; Application Designer Variables
00485 ;
00486 ; **********
00487
00488 cblock
```

```
000F         00489  Flag1
000F         00490  Temp1
000F         00491  Temp2
000F         00492  Temp3
000F                00493 endc
                    00494
                    00495
                    00496
                    00497 ; Defines specified by 1 or more elements
                    00498
                    00499
                    00500 ; *** Input File - D:\Program Files\FED\WIZPIC\APPWIZ\Misc.Inc
                    00501
                    00502 ;
                    00503 ; This macro allows elements to modify the variables which are written to
                    00504 ; the PORT registers on initialisation. It is called with the
                    00505 ; IOPin port, bit, and wanted initial value
                    00506 ;
                    00507 INITOUTPORT   macro port,bit,value           ; Note default output is 1
                    00508                if value==0
                    00509                 ifdef PORTA
                    00510                  if (port==PORTA)
                    00511                   vPORTA&=~(1<<bit)
                    00512                  endif
                    00513                 endif
                    00514                 ifdef PORTB
                    00515                  if port==PORTB
                    00516                   vPORTB&=~(1<<bit)
                    00517                  endif
                    00518                 endif
                    00519                 ifdef PORTC
                    00520                  if port==PORTC
                    00521                   vPORTC&=~(1<<bit)
                    00522                  endif
                    00523                 endif
                    00524                 ifdef PORTD
                    00525                  if port==PORTD
                    00526                   vPORTD&=~(1<<bit)
                    00527                  endif
                    00528                 endif
                    00529                 ifdef PORTE
                    00530                  if port==PORTE
                    00531                   vPORTE&=~(1<<bit)
                    00532                  endif
                    00533                 endif
                    00534                endif
                    00535               endm
                    00536
                    00537
 TRUE               00538 #ifdef EdgeUsed
                    00539 ; ***************************************
                    00540 ;
                    00541 ; Assembler file for Element : Edge Detector
                    00542 ;   - Operates on any PIC input pin
                    00543 ;
                    00544 ; Written by FED Element Editor - Mon Nov 01 14:01:33 1999
                    00545 ;
                    00546 ; ***************************************
                    00547
                    00548 ; Variables
                    00549 EICnt=(D'2' -1)/8+1
               00550     cblock
0013           00551      EdgeFlags
0013                00552       endc
               00553     cblock EdgeFlags+EICnt
                    00554       endc
                    00555 #endif
                    00556
                    00557
                    00558
                    00559
                    00560
                    00561
                    00562
                    00563
                    00564
                    00565
                    00566
                    00567 ; *** Input File - D:\Program Files\FED\WIZPIC\APPWIZ\ADInc.inc
                    00568
                    00569 ;
                    00570 ; Application Designer, timers code
                    00571 ;
                    00572
                    00573 ;
                    00574 ; Timer 0 extensions
                    00575 ;
                    00576
                    00577
                    00578 ;
                    00579 ; Hour:Minute:Second occurrences
                    00580 ;
                    00581
 TRUE               00582 #ifdef HMSUsed
                    00583       HMSnVar=0               ; Find the number of variables used by HMS
 TRUE               00584       #ifdef HourPassUsed
                    00585        HMSnVar+=4
                    00586       #endif
 TRUE               00587       #ifdef MinPassUsed
                    00588        HMSnVar+=3
                    00589       #endif
 TRUE               00590       #ifdef SecPassUsed
                    00591        HMSnVar+=2
                    00592       #endif
                    00593
               00594     cblock
0014           00595      HMSStart              ; Find the current block start
0014                00596       endc
                    00597
                    00598       HMSCount=HMSStart
                    00599       FITINBLOCK HMSCount,HMSnVar     ; Fit block into 1 RAM page
```

```
                  00600
            00601     cblock HMSCount
                  00602       endc
                  00603
 TRUE             00604       #ifdef HourPassUsed     ; HMS variables
            00605       cblock
00A1        00606        HrCount
00A1        00607        HrCount1
00A1        00608        HrCount2
00A1        00609        HrCount3
00A1              00610        endc
                  00611       #endif
 TRUE             00612       #ifdef MinPassUsed
            00613       cblock
00A5        00614        MinCount
00A5        00615        MinCount1
00A5        00616        MinCount2
00A5              00617        endc
                  00618       #endif
 TRUE             00619       #ifdef SecPassUsed
            00620       cblock
00A8        00621        SecCount
00A8        00622        SecCount1
00A8              00623        endc
                  00624       #endif
                  00625 #endif
                  00626
                  00627
                  00001 ;
                  00002 ; This file is automatically generated. Any changes will be overwritten when
                  00003 ; the application is regenerated
                  00004 ;
0000              00005          org 0
0x0000 0x0000     00006          nop                  ; Leave clear for in circuit debugger
                  00007
0x0001 0x018A     00008          clrf PCLATH            ; Reset program page bits
0x0002 0x2816     00012          goto Start
                  00013
                  00014
                  00015 ;
                  00016 ; Interrupts
                  00017 ;
0004              00018                 org 4
                  00019                 INTSAVE                ; Save variables
0x0004 0x008C     MACRO    movwf W_TEMP            ; Copy W to TEMP register
0x0005 0x0E03     MACRO    swapf STATUS,w          ; Swap status to be saved into W
0x0006 0x0183     MACRO    clrf STATUS             ; Bank 0
0x0007 0x008D     MACRO    movwf STATUS_TEMP       ; Save status
0x0008 0x0804     MACRO    movf FSR,w              ; Copy FSR to W
0x0009 0x008E     MACRO    movwf FSR_TEMP          ; Copy FSR from W to FSR_TEMP
                  00020                 ClearPCLATH 0
                  00021 IntPri:                                ; Priority Interrupts here
                  00022
0x000A 0x2849     00023                 goto UserInterrupt
                  00024 AutoInt:
 TRUE             00025       #ifdef Tmr0IntUsed      ;Timer 0
0x000B 0x1D0B     00026       btfss INTCON,T0IF              ; Test Timer 0 overflow
0x000C 0x280F     00027       goto Tmr0_Int1                 ; No overflow
 TRUE             00028       #ifdef OfUsed
0x000D 0x148F     00029       bsf Flag1 ,1            ; Set bit to show occurrence
                  00030       #endif
0x000E 0x110B     00031       bcf INTCON,T0IF                ; Clear interrupt
                  00032 Tmr0_Int1:                    ; Jump here if no interrupt
                  00033       #endif
                  00034
                  00035                 INTRECALL              ; Restore variables after interrupt
0x000F 0x080E     MACRO    movf       FSR_TEMP,w       ;Whats wrong with restoring FSR? cqg8dhecq
0x0010 0x0084     MACRO    movwf      FSR              ;Whats wrong with restoring FSR? cqg8dhecq
0x0011 0x0E0D     MACRO    swapf STATUS_TEMP,w     ; Swap STATUS_TEMP register into W
0x0012 0x0083     MACRO    movwf STATUS            ; Move W into STATUS register
0x0013 0x0E8C     MACRO    swapf W_TEMP,F          ; Swap W_TEMP
0x0014 0x0E0C     MACRO    swapf W_TEMP,w          ; Swap W_TEMP into W
0x0015 0x0009     00036                 retfie
                  00037
                  00038 ;
                  00039 ; Initialisation - initialise elements in turn
                  00040 ;
                  00041 Start:
0x0016 0x0183     00042                clrf STATUS            ; Reset RAM page bits
0x0017 0x018B     00043                  clrf INTCON            ; Disable interrupts
0x0018 0x0184     00052                  clrf FSR               ; Point to lower RAM page
                  00053
                  00054 ADInit:
                  00055       lPSD=D'4'       ;Timer 0
                  00056       vor=-2
 TRUE             00057       while lPSD
                  00058       vor=vor+1
                  00059       lPSD>>=1
                  00058       vor=vor+1
                  00059       lPSD>>=1
                  00058       vor=vor+1
                  00059       lPSD>>=1
                  00060       endw
 TRUE             00061       #if D'1'
                  00062       vOPTION_REG&=~((1<<PSA)|(1<<T0CS)|(1<<T0SE)|7)
                  00063       vOPTION_REG|=((!D'1' )<<PSA)|((!D'1' )<<T0CS)|((!D'1' )<<T0SE)|vor
0x0019 0x0181     00068       clrf TMR0
 TRUE             00069       #ifdef Tmr0IntUsed
                  00070       vINTCON|=1<<T0IE
                  00071       #endif
                  00072
                  00073       INITOUTPORT PORTB ,7 ,D'0'      ;Port Driver
                  00074
                  00075       vTRISB&=H'7F'
                  00076       INITOUTPORT PORTB ,6 ,D'0'      ;Port Driver
                  00077
                  00078       vTRISB&=H'BF'
                  00079       INITOUTPORT PORTB ,5 ,D'0'      ;Port Driver
                  00080
                  00081       vTRISB&=H'DF'
                  00082       INITOUTPORT PORTB ,4 ,D'0'      ;Port Driver
```

```
                  00083
                  00084       vTRISB&=H'EF'
                  00085       INITOUTPORT PORTB ,3 ,D'0'      ;Port Driver
                  00086
                  00087       vTRISB&=H'F7'
                  00088       INITOUTPORT PORTB ,2 ,D'0'      ;Port Driver
                  00089
                  00090       vTRISB&=H'FB'
                  00091       INITOUTPORT PORTB ,1 ,D'0'      ;Port Driver
                  00092
                  00093       vTRISB&=H'FD'
                  00094       INITOUTPORT PORTB ,0 ,D'0'      ;Port Driver
                  00095
                  00096       vTRISB&=H'FE'
                  00097               ;Port Input
                  00098
                  00099               ;Port Input
                  00100
                  00101               ;Port Input
                  00102
                  00103               ;Port Input
                  00104
                  00105               ;Port Input
                  00106
0x001A 0x018F     00107       clrf Flag1
                  00108
0x001B 0x3020     00109                 movlw vINTCON
0x001C 0x008B     00110                 movwf INTCON
 TRUE             00111                 #ifdef PORTA
0x001D 0x30FF     00112                  movlw vPORTA
0x001E 0x0085     00113                  movwf PORTA
                  00114                 #endif
 TRUE             00115                 #ifdef PORTB
0x001F 0x3000     00116                  movlw vPORTB
0x0020 0x0086     00117                  movwf PORTB
                  00118                 #endif
0x0021 0x1683     00131                 bsf STATUS,RP0
0x0022 0x3041     00132                 movlw vOPTION_REG
0x0023 0x0081     00133                 movwf OPTION_REG
 TRUE             00142                 #ifdef TRISA
0x0024 0x30FF     00143                  movlw vTRISA
0x0025 0x0085     00144                  movwf TRISA
                  00145                 #endif
 TRUE             00146                 #ifdef TRISB
0x0026 0x3000     00147                  movlw vTRISB
0x0027 0x0086     00148                  movwf TRISB
                  00149                 #endif
                  00162 ;*****************************************************************************
                  00163 ;Added by Geoff Mather to handle differences with A/D chips that assign PORTA
                  00164 ;as analoge on power up as opposed to digital I/Ps CQG8DHECQ
                  00170 ;*****************************************************************************
0x0028 0x1283     00171                 bcf STATUS,RP0
                  00172
                  00173 ADInitEnd:
                  00174       longcall InitEdge
0x0029 0x2211     MACRO    call InitEdge               ; One PAGES of ROM so simple call
                  00175       longcall HMSInit
0x002A 0x224A     MACRO    call HMSInit               ; One PAGES of ROM so simple call
0x002B 0x204A     00176                 call UserInitialise
0x002C 0x178B     00177                 bsf INTCON,GIE          ; Finally enable interrupts
                  00178
                  00179 ;
                  00180 ; Main Loop - handle occurrences as they are flagged
                  00181 ;
                  00182 Main:
                  00183       longcall TestEdge1
0x002D 0x221E     MACRO    call TestEdge1              ; One PAGES of ROM so simple call
0x002E 0x1C0F     00184       btfss Flag1 ,0  ;Detected Selected Edge
0x002F 0x2832     00185       goto GotEdge1End
                  00186       longcall CTone
0x0030 0x21C8     MACRO    call CTone               ; One PAGES of ROM so simple call
0x0031 0x100F     00187       bcf Flag1 ,0
                  00188 GotEdge1End:
0x0032 0x1C8F     00189       btfss Flag1 ,1  ;Timer 0 overflow
0x0033 0x2837     00190       goto OfEnd
                  00191       longcall HMSTick
0x0034 0x225C     MACRO    call HMSTick               ; One PAGES of ROM so simple call
                  00192       longcall T0Tone
0x0035 0x21D4     MACRO    call T0Tone               ; One PAGES of ROM so simple call
0x0036 0x108F     00193       bcf Flag1 ,1
                  00194 OfEnd:
0x0037 0x1D0F     00195       btfss Flag1 ,2  ;Occurs once every second
0x0038 0x283B     00196       goto SecPassEnd
                  00197       longcall CountDown
0x0039 0x2120     MACRO    call CountDown               ; One PAGES of ROM so simple call
0x003A 0x110F     00198       bcf Flag1 ,2
                  00199 SecPassEnd:
0x003B 0x1D8F     00200       btfss Flag1 ,3  ;Occurs once every minute
0x003C 0x283F     00201       goto MinPassEnd
                  00202       longcall TimeMin
0x003D 0x2161     MACRO    call TimeMin               ; One PAGES of ROM so simple call
0x003E 0x118F     00203       bcf Flag1 ,3
                  00204 MinPassEnd:
0x003F 0x1E0F     00205       btfss Flag1 ,4  ;Occurs once every hour
0x0040 0x2843     00206       goto HourPassEnd
                  00207       longcall TimeHour
0x0041 0x2182     MACRO    call TimeHour               ; One PAGES of ROM so simple call
0x0042 0x120F     00208       bcf Flag1 ,4
                  00209 HourPassEnd:
                  00210       longcall TestEdge0
0x0043 0x2214     MACRO    call TestEdge0              ; One PAGES of ROM so simple call
0x0044 0x1E8F     00211       btfss Flag1 ,5  ;Detected Selected Edge
0x0045 0x2848     00212       goto GotEdge0End
                  00213       longcall ReadDTMF
0x0046 0x2112     MACRO    call ReadDTMF               ; One PAGES of ROM so simple call
0x0047 0x128F     00214       bcf Flag1 ,5
                  00215 GotEdge0End:
0x0048 0x285C     00219                 goto UserLoop
                  00220
                  00221 ;
                  00222 ; Load oscillator calibration value on reset for 14 bit processors
```

```
                    00223 ;
                    00233
                    00234
                    00235
                    00001 ;G8DHE no 1 clock needs 4016000
                    00002 ;G4PAP no 1 clock needs 4011200
                    00003 ; This file includes all user definable routines. It may be changed at will as
                    00004 ; it will not be regenerated once the application has been generated for the
                    00005 ; first time.
                    00006 ;Version = 1.4
                    00007 ; version 1.0 Basic relay operations
                    00008 ;         1.4 Includes reduction of PIN to a single * code
                    00009 ;DTMF Remote operation
                    00010 ;This program detects a sequence of DTMF signal tones from a HT9170 decoder
                    00011 ;connected to PORT A, DV=b0, D0=b1, D1=b2, D2=b3 & D3=b4
                    00012 ;Looks for a 4 digit PIN sequence held in Flash memory bytes 0-3.
                    00013 ;Once the PIN has been recieved a Timer is started, of a duration held
                    00014 ;in Flash memory at byte 8, during which the remaining command signals must
                    00015 ;be received.
                    00016 ;The command signals consist of a single key followed by a parameter.
                    00017 ;Keys 1-8 identifies the relay to be changed,
                    00018 ;  a parameter of 0= 0x0A =Off, 1= 0x01 = On.
                    00019 ;The PIN number can be changed by sending a DTMF A = 0x0D followed
                    00020 ;  by a new 4 digit parameter (excluding DTMF code #) which is then stored
                    00021 ;  in Flash memory bytes 4-7, this must then be matched by entering the sequence
                    00022 ;  a second time after which it is copied to locations 0-3.
                    00023 ;A command of DTMF B =0x0E allows the timer value to be set,
                    00024 ;  with a parameter in decimal seconds between 1 and 255 ending with DTMF *.
                    00025 ;On receipt of DTMF # = 0x0C then all operations are reset and the timer stopped.
                    00026 ;  A new PIN sequence will then be required to initiate further codes.
                    00027 ;
                    00028 ;On first operation the PIN is DTMF CCCC and the Timer value is 256 seconds
                    00029 ;
                    00030 ;In summary
                    00031 ;  4 Digit PIN  followed by the period set by the Timer to send commands in.
                    00032 ; Commands are;
                    00033 ; 1-8 to select a Relay followed by 0 or 1 to Release / Operate
                    00034 ; "A" followed by new PIN, followed by a repeat of the new PIN it will then be
                    00035 ;   stored and the session ended.
                    00036 ; "B" followed by upto 3 digits less than 255 and ending with *
                    00037 ;   will set the Timer value in seconds from the next access.
                    00038 ; "#" will end the current command session.
                    00039 ; Any invalid command or parameter will cause the session to end, the PIN will
                    00040 ; need to be re-entered to continue.
                    00041 ;                DTMF KEYS to HEX
                    00042 ;
                    00043 ;         1=01    2=02    3=03    A=0D
                    00044 ;         4=04    5=05    6=06    B=0E
                    00045 ;         7=07    8=08    9=09    C=0F
                    00046 ;         *=0B    0=0A    #=0C    D=00
                    00047 ;
                    00048 ;*******************************************************************************
                    00049 ;
                    00050 ; Insert your interrupt handling code if required here. Your code should finish
                    00051 ; with the line "goto AutoInt"
                    00052 ;
                    00053
                    00054 UserInterrupt:
0x0049 0x280B       00055                 goto AutoInt            ; Return to App Designer interrupts
                    00056
                    00057
                    00058 ;*******************************************************************************
                    00059 ;
                    00060 ; Insert your initialisation code if required here. Your code should finish with
                    00061 ; the line "return". Note that when this routine is called Interrupts will not
                    00062 ; be enabled - the Application Designer will enable them before the main loop
                    00063 ;
            00064 cblock
00AA        00065  Current             ;Current DTMF tone being processed
00AA        00066  CopyCount           ;used for copying new PIN bytes
00AA        00067  TCount                      ;Temp counter
00AA        00068  TimeCD                      ;Count Down Timer reset on Zero
00AA        00069  NewTime             ;New Time temporary value
00AA        00070  Relay                       ;Relay selected
00AA        00071  R2T                 ;Relays 2 Toggle
00AA        00072  RelayCD             ;Relay Count Down Timer
00AA        00073  RelayTState         ;Relay Toggle State
00AA        00074                      ;b0=Relay State
00AA        00075 DTMFStatus           ;b0-b1 PIN count
00AA        00076                      ;b2 Valid PIN receieved
00AA        00077                      ;b3 New PIN flag
00AA        00078                      ;b4 Match New PIN flag
00AA        00079                      ;b5 New Toggle times
00AA        00080                      ;b6 New Timer
00AA        00081                      ;b7 New Clock Times
00AA        00082 DTMFTone             ;b0 Zero then OK tone, One then Reset tone
00AA        00083                      ;b1 Tone out
00AA        00084                      ;b2 /2 flag for low tone
00AA        00085                      ;b3 RESET PIN flag
00AA        00086                      ;b4-6 spare
00AA        00087                      ;b7 New DTMF tone
00AA        00088 R2Time                       ;Relays to time
00AA        00089 MOff                 ;Minutes Off
00AA        00090 HOff                 ;Hours Off
00AA        00091 MOn                  ;Minutes On
00AA        00092 HOn                  ;Hours On
00AA        00093 Minutes                      ;Minutes counter
00AA        00094 Hours                        ;Hours counter
00AA        00095 TMOff
00AA        00096 THOff
00AA        00097 TMOn
00AA        00098 THOn
00AA        00099 TM
00AA        00100 TH                   ;Temp versions of above
00AA        00101 ToneCount
00AA               00102 endc
                   00103 UserInitialise:
0x004A 0x01AA      00104       clrf    Current
0x004B 0x01AB      00105       clrf    CopyCount
0x004C 0x01AD      00106       clrf    TimeCD
0x004D 0x01AE      00107       clrf    NewTime
```

```
0x004E 0x01AF       00108       clrf    Relay
0x004F 0x01B0       00109       clrf    R2T
0x0050 0x01B5       00110       clrf    R2Time
0x0051 0x01B1       00111       clrf    RelayCD
0x0052 0x01B2       00112       clrf    RelayTState
0x0053 0x01B3       00113       clrf    DTMFStatus
0x0054 0x01B4       00114       clrf    DTMFTone
0x0055 0x01BB       00115       clrf    Hours
0x0056 0x01BA       00116       clrf    Minutes
0x0057 0x01B9       00117       clrf    HOn
0x0058 0x01B7       00118       clrf    HOff
0x0059 0x01B8       00119       clrf    MOn
0x005A 0x01B6       00120       clrf    MOff
0x005B 0x0008       00121                 return                ; Return to App Designer
                    00122
                    00123 ;*****************************************************************************
                    00124 ;
                    00125 ; Insert your main loop code if required here. Your code should finish with
                    00126 ; the line "goto Main"
                    00127 ;
                    00128 UserLoop:
0x005C 0x1FB4       00129       btfss   DTMFTone,7            ;Check for a new tone receieved
0x005D 0x282D       00130       goto    Main                  ;No so back to main loop
0x005E 0x300C       00131       movlw   0x0C                  ;Check for a reset
0x005F 0x022A       00132       subwf   Current,W             ;Test it
0x0060 0x1903       00133       skpnz
0x0061 0x28FD       00134       goto    PINfail               ;Does a reset on receiving DTMF #
0x0062 0x19B3       00135       btfsc   DTMFStatus,3          ;Are we dealing with a PIN set
0x0063 0x28BA       00136       goto    NewPIN                ;Yes so handle it
0x0064 0x1A33       00137       btfsc   DTMFStatus,4          ;Is it a Match new PIN
0x0065 0x28C1       00138       goto    MatchIt               ;Yes so handle it
0x0066 0x1D33       00139       btfss   DTMFStatus,2          ;Has a PIN been received
0x0067 0x28D7       00140       goto    Pins                  ;No so handle incoming PIN
0x0068 0x1AB3       00141       btfsc   DTMFStatus,5          ;New time settings ?
0x0069 0x28F5       00142       goto    NewTimes              ;Yes so handle them
0x006A 0x1B33       00143       btfsc   DTMFStatus,6          ;Check for NewTimer
0x006B 0x28F5       00144       goto    NewTimes              ;Handles both types now
0x006C 0x1BB3       00145       btfsc   DTMFStatus,7          ;Check for New Clock Times
0x006D 0x2983       00146       goto    NewClock              ;Handle Clock times
0x006E 0x08AF       00147       movf    Relay                 ;Test if we have a Relay number
0x006F 0x1903       00148       skpnz
0x0070 0x2895       00149       goto    Choose
                    00150 OperateRelay:
0x0071 0x03AA       00151       decf    Current               ;Check for a DTMF 1 = Operate relay
0x0072 0x1D03       00152       skpz
0x0073 0x287A       00153       goto    RelayToggle           ;Not a 1 so Current=Current-1
                    00154       movfw   Relay                 ;get the relay to be operated
0x0074 0x082F       MACRO
0x0075 0x0486       00155       iorwf   PORTB                 ;Set the bit to operate the relay
0x0076 0x092F       00156       comf    Relay,W               ;Clear Toggle bit just in case
0x0077 0x05B0       00157       andwf   R2T                   ;it was toggling
                    00158 ;     andwf   R2Time                ;and any Timed relay; NO doesn't make sense
0x0078 0x01AF       00159       clrf    Relay                 ;Clear Relay ready for next time
0x0079 0x28E7       00160       goto    Done                  ;All done
                    00161 RelayToggle:
0x007A 0x03AA       00162       decf    Current               ;Check for a DTMF 2 = Toggle relay
0x007B 0x1D03       00163       skpz
0x007C 0x2885       00164       goto    Clocked               ;Not a 2 so Current=Current-2
                    00165       movfw   Relay                 ;Get relay flag bit
0x007D 0x082F       MACRO
0x007E 0x06B0       00166       xorwf   R2T                   ;Toggle R2T bits as relevant
0x007F 0x092F       00167       comf    Relay,W               ;get the complement
                    00168 ;     andwf   R2Time                ;and clear any timed relay; No as above
0x0080 0x1432       00169       bsf     RelayTState,0         ;Set the Toggle State to operate next
0x0081 0x3001       00170       movlw   0x01                  ;Setup the RelayCD time to 1
0x0082 0x00B1       00171       movwf   RelayCD               ;and cause the timer to runout!
0x0083 0x01AF       00172       clrf    Relay
0x0084 0x28E7       00173       goto    Done
                    00174 Clocked:
0x0085 0x03AA       00175       decf    Current               ;Check for DTMF 3 = Clock Time
0x0086 0x1D03       00176       skpz
0x0087 0x288C       00177       goto    Release               ;Not a 3 so Current=Current-3
                    00178       movfw   Relay                 ;Get the relay flag bit
0x0088 0x082F       MACRO
0x0089 0x06B5       00179       xorwf   R2Time                ;Toggle the bits as relevant
0x008A 0x01AF       00180       clrf    Relay                 ;
0x008B 0x28E7       00181       goto    Done                  ;Thats it
                    00182 Release:
0x008C 0x3007       00183       movlw   0x07                  ;0=0x0A-3=0x07
0x008D 0x022A       00184       subwf   Current,W             ;if it was a Zero=10
0x008E 0x1D03       00185       skpz
0x008F 0x28FD       00186       goto    PINfail               ;wasn't a DTMF 0 so reset
0x0090 0x092F       00187       comf    Relay,W               ;Take the inverse of relay selected
0x0091 0x0586       00188       andwf   PORTB                 ;and clear the bit to release the relay
0x0092 0x05B0       00189       andwf   R2T                   ;Also clear any Toggle bit
                    00190 ;     andwf   R2Time                ;and any timed relay; NO as above
0x0093 0x01AF       00191       clrf    Relay                 ;Clear Relay ready for next time
0x0094 0x28E7       00192       goto    Done                  ;All done
                    00193 Choose:
0x0095 0x300D       00194       movlw   0x0D                  ;Check for New PIN command
0x0096 0x022A       00195       subwf   Current,W             ;Check it
0x0097 0x1D03       00196       skpz
0x0098 0x289B       00197       goto    TestT                 ;No so it could be a SET TIMER command
0x0099 0x15B3       00198       bsf     DTMFStatus,3          ;Set NEW PIN flag
0x009A 0x28E7       00199       goto    Done
0x009B 0x300E       00200 TestT:        movlw   0x0E                  ;Check for New Timer value
0x009C 0x022A       00201       subwf   Current,W             ;Check it
0x009D 0x1D03       00202       skpz
0x009E 0x28A2       00203       goto    TimesT                ;No so what about Changing times
0x009F 0x1733       00204       bsf     DTMFStatus,6          ;Set the Flag
0x00A0 0x01AE       00205       clrf    NewTime
0x00A1 0x28E7       00206       goto    Done
0x00A2 0x300F       00207 TimesT: movlw 0x0F                  ;Check for C key
0x00A3 0x022A       00208       subwf   Current,W             ;Check it
0x00A4 0x1D03       00209       skpz
0x00A5 0x28AA       00210       goto    TestClock             ;No so now for Clock
0x00A6 0x16B3       00211       bsf     DTMFStatus,5          ;Set the New Times flag
0x00A7 0x3002       00212       movlw   0x02                  ;Use as part counter
0x00A8 0x00AB       00213       movwf   CopyCount             ;and this is part two
0x00A9 0x28E7       00214       goto    Done
                    00215 TestClock:
```

```
0x00AA 0x08AA       00216       movf    Current                 ;Checking for D key (0x00)
0x00AB 0x1D03       00217       skpz
0x00AC 0x28B3       00218       goto    TestRelay               ;No so it must be a relay
0x00AD 0x17B3       00219       bsf     DTMFStatus,7            ;Set the flag bit
0x00AE 0x3006       00220       movlw   0x06                    ;Use as part counter
0x00AF 0x00AB       00221       movwf   CopyCount               ;and this is part three
0x00B0 0x3002       00222       movlw   0x02                    ;Two digits each
0x00B1 0x00AC       00223       movwf   TCount                  ;Set temp counter to 2 for digits.
0x00B2 0x28E7       00224       goto    Done
                    00225 TestRelay:
0x00B3 0x1403       00226       setc                            ;Set the C flag and rotate up selection
0x00B4 0x0DAF       00227 Rotate:       rlf     Relay                   ;Move in Bit
0x00B5 0x1803       00228       skpnc                           ;Test to make sure not greater than 8
0x00B6 0x28FD       00229       goto    PINfail                 ;More than 8 so reset system
0x00B7 0x0BAA       00230       decfsz  Current                 ;Count it down
0x00B8 0x28B4       00231       goto    Rotate                  ;Move up again
0x00B9 0x28E7       00232       goto    Done                    ;Finished
                    00233 NewPIN:
                    00234       movfw   DTMFStatus              ;Get DTMFStatus in order to get PIN count
0x00BA 0x0833       MACRO
0x00BB 0x3907       00235       andlw   0x07                    ;Point at bytes 4-7 rather than 0-3
0x00BC 0x00C3       00236       movwf   EEPromAdr               ;Point at place to save
                    00237       movfw   Current                 ;get new DTMF code
0x00BD 0x082A       MACRO
0x00BE 0x2232       00238       call    EEPROMWRITE             ;and save the value
0x00BF 0x0AB3       00239       incf    DTMFStatus              ;Move up the count when 4 Rx then
                    00240                                       ;Match New PIN flag is set
0x00C0 0x28E7       00241       goto    Done                    ;Keep looping
                    00242 MatchIt:
                    00243       movfw   DTMFStatus              ;get status in order to get PIN count
0x00C1 0x0833       MACRO
0x00C2 0x3903       00244       andlw   0x03                    ;Mask out all but PIN count
0x00C3 0x3E04       00245       addlw   0x04                    ;Point at bytes to Match
0x00C4 0x00C3       00246       movwf   EEPromAdr               ;Save as Address to be matched to
0x00C5 0x2228       00247       call    EEPROMREAD              ;Get value to be checked for
0x00C6 0x02AA       00248       subwf   Current                 ;Compare it
0x00C7 0x1D03       00249       skpz
0x00C8 0x28FD       00250       goto    PINfail                 ;Oops not right
0x00C9 0x0AB3       00251       incf    DTMFStatus              ;OK when all 4 match then PIN Valid set
0x00CA 0x1D33       00252       btfss   DTMFStatus,2            ;Check for all matched
0x00CB 0x28E7       00253       goto    Done                    ;Not yet so continue looping
                    00254 CopyPIN:
0x00CC 0x3004       00255       movlw   0x04                    ;byte+1 to copy into
0x00CD 0x00C3       00256       movwf   EEPromAdr               ;save for address
0x00CE 0x03C3       00257 CLoop:        decf    EEPromAdr               ;Move it down
0x00CF 0x1543       00258       bsf     EEPromAdr,2             ;Point at byte to copy
0x00D0 0x2228       00259       call    EEPROMREAD              ;Read Byte
0x00D1 0x1143       00260       bcf     EEPromAdr,2             ;Point at save area
0x00D2 0x2232       00261       call    EEPROMWRITE             ;Write it
0x00D3 0x08C3       00262       movf    EEPromAdr               ;Test it
0x00D4 0x1D03       00263       skpz
0x00D5 0x28CE       00264       goto    CLoop                   ;Not finished yet
0x00D6 0x28FD       00265       goto    PINfail                 ;Only to force a reset!
                    00266 Pins:
                    00267       movfw   DTMFStatus              ;get status in order to get PIN count
0x00D7 0x0833       MACRO
0x00D8 0x3903       00268       andlw   0x03                    ;Mask out all but PIN count
0x00D9 0x00C3       00269       movwf   EEPromAdr               ;Save as Address to be matched to
0x00DA 0x2228       00270       call    EEPROMREAD              ;Get value to be checked for
0x00DB 0x390F       00271       andlw   0x0f                    ;To take care of High nibble
0x00DC 0x022A       00272       subwf   Current,W               ;Compare it
0x00DD 0x1D03       00273       skpz
0x00DE 0x28FD       00274       goto    PINfail                 ;Oops not right
0x00DF 0x0A33       00275       incf    DTMFStatus,W            ;OK when all 4 match then PIN Valid set
0x00E0 0x00B3       00276       movwf   DTMFStatus              ;Save it
0x00E1 0x3903       00277       andlw   0x03                    ;Mask count again
0x00E2 0x3C01       00278       sublw   0x01                    ;If its the first time
0x00E3 0x1903       00279       skpnz   NF                      ;then call check stars else continue
0x00E4 0x21F8       00280       call    CheckStars
0x00E5 0x1933       00281 NF    btfsc   DTMFStatus,2
0x00E6 0x210D       00282       call    LoadTimer               ;Load in the timer count
0x00E7 0x13B4       00283 Done: bcf     DTMFTone,7              ;Clear New tone bit
0x00E8 0x1034       00284       bcf     DTMFTone,0              ;OK Tone on falling edge Port A0
0x00E9 0x282D       00285       goto    Main                    ;Continue
                    00286 NTStore:
0x00EA 0x3005       00287       movlw   0x05                    ;Check for minimum of 5 seconds
0x00EB 0x022E       00288       subwf   NewTime,W               ;Check it
0x00EC 0x1C03       00289       skpc
0x00ED 0x28FD       00290       goto    PINfail                 ;less than 5 so reset
0x00EE 0x3008       00291       movlw   0x08                    ;Point to flash location
0x00EF 0x00C3       00292       movwf   EEPromAdr               ;Save location
                    00293       movfw   NewTime                 ;get new value
0x00F0 0x082E       MACRO
0x00F1 0x2232       00294       call    EEPROMWRITE             ;Save it
0x00F2 0x01AE       00295       clrf    NewTime                 ;reset the value
0x00F3 0x1333       00296       bcf     DTMFStatus,6            ;Clear the New timer flag
0x00F4 0x28E7       00297       goto    Done                    ;OK completed
                    00298 NewTimes:
0x00F5 0x213C       00299       call    NTime                   ;Add the next digit
0x00F6 0x1803       00300       skpnc                           ;Test for C set
0x00F7 0x28FB       00301       goto    NTSoR                   ;Yes its set check next
0x00F8 0x1903       00302       skpnz
0x00F9 0x28E7       00303       goto    Done                    ;Z=1,C=0 so get next key
0x00FA 0x2800       00304       goto    0                       ;Oops Z=0 C=0 not valid!!!! Restart!!
                    00305 NTSoR:
0x00FB 0x1D03       00306       skpz                            ;Test for Z bit
0x00FC 0x2900       00307       goto    NTSave                  ;Save value otherwise fall thru to Reset
                    00308 PINfail:
0x00FD 0x2159       00309       call    ResetValues             ;Restart all values
0x00FE 0x1434       00310       bsf     DTMFTone,0              ;Reset Tone on falling edge Port A 0
0x00FF 0x282D       00311         goto Main                     ;Return to App Designer main loop
                    00312 NTSave:
0x0100 0x1EB3       00313       btfss   DTMFStatus,5            ;Its Relay times not time out
0x0101 0x28EA       00314       goto    NTStore                 ;Store new timer period
                    00315       movfw   CopyCount               ;Get the value pointer
0x0102 0x082B       MACRO
0x0103 0x3808       00316       iorlw   0x08                    ;to generate byte in Flash
0x0104 0x00C3       00317       movwf   EEPromAdr               ;Save pointer
                    00318       movfw   NewTime                 ;Get the value
0x0105 0x082E       MACRO
0x0106 0x2232       00319       call    EEPROMWRITE             ;save the value
```

```
0x0107 0x01AE       00320       clrf    NewTime
0x0108 0x03AB       00321       decf    CopyCount               ;Move down the pointer
0x0109 0x1D03       00322       skpz
0x010A 0x290C       00323       goto    NT2
0x010B 0x12B3       00324       bcf     DTMFStatus,5            ;Clear the Flag as its done
0x010C 0x28E7       00325 NT2:  goto    Done                    ;And now back in both conditions
                    00326 ;*****************************************************************************
                    00327 LoadTimer:
0x010D 0x3008       00328       movlw   0x08                    ;Point at flash address
0x010E 0x00C3       00329       movwf   EEPromAdr
0x010F 0x2228       00330       call    EEPROMREAD              ;Get the count
0x0110 0x00AD       00331       movwf   TimeCD                  ;save it in the counter
0x0111 0x0008       00332       return                          ;finished
                    00333 ;Rising edge routine
                    00334 ReadDTMF:
0x0112 0x1683       00335       bsf     STATUS,RP0
0x0113 0x1485       00336       bsf     TRISA,1                 ;Make sure Bit 1 is an input
0x0114 0x1283       00337       bcf     STATUS,RP0
                    00338       DELAY   .1000                   ;delay 1000us=1ms
0x0115 0x3001       MACRO    movlw LoopDelay+1
0x0116 0x220B       MACRO    call BigDel
0x0117 0x3057       MACRO    movlw LoopDelay/3+1
0x0118 0x2206       MACRO    call Delay0-LoopDelay%3
0x0119 0x1003       00339       clrc                            ;Clear the C flag before
0x011A 0x0C05       00340       rrf     PORTA,W                 ;rotating Port A into W
0x011B 0x00AA       00341       movwf   Current                 ;Save current value
0x011C 0x17B4       00342       bsf     DTMFTone,7              ;Indicate new tone receieved
0x011D 0x30F8       00343       movlw   0xF8                    ;Clear Tone flags
0x011E 0x05B4       00344       andwf   DTMFTone
0x011F 0x0008       00345       return
                    00346 ;*****************************************************************************
                    00347 ;Count Down of Timer
                    00348 CountDown:
0x0120 0x08AD       00349       movf    TimeCD                  ;Test to see if its running
0x0121 0x1903       00350       skpnz
0x0122 0x2926       00351       goto    RCD                     ;Zero not running check for relay times
0x0123 0x0BAD       00352       decfsz  TimeCD                  ;Count Down
0x0124 0x2926       00353       goto    RCD
0x0125 0x2159       00354       call    ResetValues             ;Restart all values
                    00355 RCD:
0x0126 0x08B0       00356       movf    R2T                     ;test to see if any are toggling
0x0127 0x1903       00357       skpnz
0x0128 0x0008       00358       return                          ;Return so that Time stands still
0x0129 0x03B1       00359       decf    RelayCD                 ;Count Down relay times
0x012A 0x1D03       00360       skpz
0x012B 0x0008       00361       return                          ;Not time to toggle yet
0x012C 0x1832       00362       btfsc   RelayTState,0           ;Are we Operating or Releasing ?
0x012D 0x2933       00363       goto    RTOperate               ;Was released so now Operate
                    00364 RTRelease:
0x012E 0x0930       00365       comf    R2T,W                   ;Invert bits into W
0x012F 0x0586       00366       andwf   PORTB                   ;Clear the Relay(s)
0x0130 0x3009       00367       movlw   0x09                    ;Load the pointer to Released time
0x0131 0x00C3       00368       movwf   EEPromAdr               ;Point to it
0x0132 0x2937       00369       goto    RTSetupTime
                    00370 RTOperate:
                    00371       movfw   R2T                     ;get the bits into W
0x0133 0x0830       MACRO
0x0134 0x0486       00372       iorwf   PORTB                   ;Set the Relay(s)
0x0135 0x300A       00373       movlw   0x0A                    ;Load pointer to Operated time
0x0136 0x00C3       00374       movwf   EEPromAdr               ;Point to it
                    00375 RTSetupTime:
0x0137 0x2228       00376       call    EEPROMREAD              ;Get new time
0x0138 0x00B1       00377       movwf   RelayCD                 ;save as new time period
0x0139 0x3001       00378       movlw   0x01                    ;Setup to change Relay State
0x013A 0x06B2       00379       xorwf   RelayTState             ;toggle the state bit
0x013B 0x0008       00380       return                          ;Thats it
                    00381 ;*****************************************************************************
                    00382 NTime:
0x013C 0x300B       00383       movlw   0x0B                    ;Check for end of sequence
0x013D 0x022A       00384       subwf   Current,W
0x013E 0x1903       00385       skpnz
0x013F 0x2953       00386       goto    NTSaveIt
0x0140 0x300A       00387       movlw   0x0A                    ;Check against 10
0x0141 0x022A       00388       subwf   Current,W
0x0142 0x1C03       00389       skpc                            ;Check for >9
0x0143 0x2947       00390       goto    NTAddDigit              ;OK its less than 10
0x0144 0x1D03       00391       skpz
0x0145 0x2956       00392       goto    NTReset                 ;Its greater than 10 so reset
0x0146 0x01AA       00393       clrf    Current                 ;Set Current to Zero
                    00394 NTAddDigit:                           ;First *10 then Add Current
0x0147 0x1003       00395       clrc                            ;Prepare to multiply
                    00396       movfw   NewTime                 ;Save value in W
0x0148 0x082E       MACRO
0x0149 0x0DAE       00397       rlf     NewTime                 ;*2
0x014A 0x0DAE       00398       rlf     NewTime                 ;*4
0x014B 0x0DAE       00399       rlf     NewTime                 ;*8
0x014C 0x07AE       00400       addwf   NewTime                 ;+1
0x014D 0x07AE       00401       addwf   NewTime                 ;+2 now NewTimer=NewTimer*10
                    00402       movfw   Current                 ;Add in new value
0x014E 0x082A       MACRO
0x014F 0x07AE       00403       addwf   NewTime                 ;Complete
0x0150 0x1003       00404       clrc
0x0151 0x1503       00405       setz
0x0152 0x0008       00406       return                          ;Return ready for next digit
                    00407 NTSaveIt:
0x0153 0x1403       00408       setc
0x0154 0x1103       00409       clrz
0x0155 0x0008       00410       return                          ;Return to save value
                    00411 NTReset:
0x0156 0x1403       00412       setc
0x0157 0x1503       00413       setz
0x0158 0x0008       00414       return                          ;Return to Reset
                    00415 ;*****************************************************************************
                    00416 ResetValues:
0x0159 0x01AA       00417       clrf    Current
0x015A 0x01AB       00418       clrf    CopyCount
0x015B 0x01AD       00419       clrf    TimeCD
0x015C 0x01AE       00420       clrf    NewTime
0x015D 0x01AF       00421       clrf    Relay
0x015E 0x01B3       00422       clrf    DTMFStatus
0x015F 0x01B4       00423       clrf    DTMFTone
```

```
0x0160 0x0008       00424                return              ; Return to App Designer
                    00425 ;*****************************************************************************
                    00426 TimeMin:                          ;Called by clock routines
0x0161 0x0ABA       00427       incf    Minutes             ;Increments the Minutes counter
0x0162 0x303C       00428       movlw   .60                 ;Load Top count
0x0163 0x023A       00429       subwf   Minutes,w           ;Compare them
0x0164 0x1903       00430       skpnz
0x0165 0x217C       00431       call    HourInc
                    00432       movfw   MOn                 ;Get the On minutes
0x0166 0x0838       MACRO
0x0167 0x023A       00433       subwf   Minutes,w           ;Compare On minutes
0x0168 0x1D03       00434       skpz
0x0169 0x2971       00435       goto    MinutesOff          ;Nope so check Off time
                    00436       movfw   HOn                 ;Check Hours
0x016A 0x0839       MACRO
0x016B 0x023B       00437       subwf   Hours,w             ;compare hour
0x016C 0x1D03       00438       skpz
0x016D 0x2971       00439       goto    MinutesOff          ;Nope so check Off time
                    00440       movfw   R2Time              ;get the relays
0x016E 0x0835       MACRO
0x016F 0x0486       00441       iorwf   PORTB               ;Turn on the relays that are timed
0x0170 0x0008       00442       return                      ;Time On complete
                    00443 MinutesOff:
                    00444       movfw   MOff                ;Get the Off minutes
0x0171 0x0836       MACRO
0x0172 0x023A       00445       subwf   Minutes,w           ;compare Off minutes
0x0173 0x1D03       00446       skpz
0x0174 0x0008       00447       return                      ;Time doesn't match so finish
                    00448       movfw   HOff                ;Get the Off Hours
0x0175 0x0837       MACRO
0x0176 0x023B       00449       subwf   Hours,w             ;compare the Off hours
0x0177 0x1D03       00450       skpz
0x0178 0x0008       00451       return                      ;No so finish
0x0179 0x0935       00452       comf    R2Time,w            ;get the relays but inverted
0x017A 0x0586       00453       andwf   PORTB               ;Release the relays
0x017B 0x0008       00454       return                      ;finished
                    00455 HourInc:
0x017C 0x01BA       00456       clrf    Minutes
0x017D 0x0ABB       00457       incf    Hours               ;Increment the Hours
0x017E 0x3018       00458       movlw   .24                 ;Load top count
0x017F 0x023B       00459       subwf   Hours,w             ;compare them
0x0180 0x1903       00460       skpnz
0x0181 0x01BB       00461       clrf    Hours               ;reset for new day
0x0182 0x0008       00462 TimeHour:     return              ;Hours complete Call here for Hours Accuracy
                    00463 ;*****************************************************************************
                    00464 NewClock:
0x0183 0x300A       00465       movlw   0x0A                ;Check against 10
0x0184 0x022A       00466       subwf   Current,W
0x0185 0x1C03       00467       skpc                        ;Check for >9
0x0186 0x298A       00468       goto    NCAddDigit          ;OK its less than 10
0x0187 0x1D03       00469       skpz
0x0188 0x2956       00470       goto    NTReset             ;Its greater than 10 so reset
0x0189 0x01AA       00471       clrf    Current             ;Set Current to Zero
                    00472 NCAddDigit:                       ;First *10 then Add Current
0x018A 0x1003       00473       clrc                        ;Prepare to multiply
                    00474       movfw   NewTime             ;Save value in W
0x018B 0x082E       MACRO
0x018C 0x0DAE       00475       rlf     NewTime             ;*2
0x018D 0x0DAE       00476       rlf     NewTime             ;*4
0x018E 0x0DAE       00477       rlf     NewTime             ;*8
0x018F 0x07AE       00478       addwf   NewTime             ;+1
0x0190 0x07AE       00479       addwf   NewTime             ;+2 now NewTimer=NewTimer*10
                    00480       movfw   Current             ;Add in new value
0x0191 0x082A       MACRO
0x0192 0x07AE       00481       addwf   NewTime             ;Complete
0x0193 0x0BAC       00482       decfsz  TCount              ;Decrement digit count
0x0194 0x28E7       00483       goto    Done                ;Loop for second digit
                    00484 CSaveDigits:
0x0195 0x30BB       00485       movlw   TMOff-1             ;Pointer to lowest temp variable
0x0196 0x072B       00486       addwf   CopyCount,W         ;Add in the offset actually less 1
0x0197 0x0084       00487       movwf   FSR                 ;Point into memory
                    00488       movfw   NewTime             ;Get digit pair
0x0198 0x082E       MACRO
0x0199 0x0080       00489       movwf   INDF                ;save the value
0x019A 0x3002       00490       movlw   0x02                ;reset digit count
0x019B 0x00AC       00491       movwf   TCount              ;to 2
0x019C 0x01AE       00492       clrf    NewTime
0x019D 0x0BAB       00493       decfsz  CopyCount           ;Move offset/counter down
0x019E 0x28E7       00494       goto    Done                ;More pairs yet
                    00495 ValidateTimes:
                    00496       movfw   TH                  ;Load in Temp Hours
0x019F 0x0841       MACRO
0x01A0 0x3C17       00497       sublw   .23
0x01A1 0x1C03       00498       skpc
0x01A2 0x28FD       00499       goto    PINfail             ;Hours too large
                    00500       movfw   TM
0x01A3 0x0840       MACRO
0x01A4 0x3C3B       00501       sublw   .59
0x01A5 0x1C03       00502       skpc
0x01A6 0x28FD       00503       goto    PINfail             ;Minutes toolarge
                    00504       movfw   THOn
0x01A7 0x083F       MACRO
0x01A8 0x3C17       00505       sublw   .23
0x01A9 0x1C03       00506       skpc
0x01AA 0x28FD       00507       goto    PINfail
                    00508       movfw   TMOn
0x01AB 0x083E       MACRO
0x01AC 0x3C3B       00509       sublw   .59
0x01AD 0x1C03       00510       skpc
0x01AE 0x28FD       00511       goto    PINfail
                    00512       movfw   THOff
0x01AF 0x083D       MACRO
0x01B0 0x3C17       00513       sublw   .23
0x01B1 0x1C03       00514       skpc
0x01B2 0x28FD       00515       goto    PINfail
                    00516       movfw   TMOff
0x01B3 0x083C       MACRO
0x01B4 0x3C3B       00517       sublw   .59
0x01B5 0x1C03       00518       skpc
0x01B6 0x28FD       00519       goto    PINfail
                    00520 CStore:                                   ;Store three time pairs
```

```
0x01B7 0x3006       00521       movlw   0x06                    ;make offset counter 6 again
0x01B8 0x00AB       00522       movwf   CopyCount
0x01B9 0x30BB       00523 CSLoop:       movlw   TMOff-1                 ;Point to lowest temp variable
0x01BA 0x072B       00524       addwf   CopyCount,w             ;Add offset
0x01BB 0x0084       00525       movwf   FSR                     ;Point at memory
                    00526       movfw   INDF                    ;get the value
0x01BC 0x0800       MACRO
0x01BD 0x00AE       00527       movwf   NewTime                 ;save it temp
0x01BE 0x3006       00528       movlw   0x06                    ;load difference between Temp and final
0x01BF 0x0284       00529       subwf   FSR                     ;Point at final locations
                    00530       movfw   NewTime
0x01C0 0x082E       MACRO
0x01C1 0x0080       00531       movwf   INDF                    ;save value in new final location
0x01C2 0x0BAB       00532       decfsz  CopyCount               ;decrement loop counter
0x01C3 0x29B9       00533       goto    CSLoop                  ;repeat for all 6 values
0x01C4 0x224A       00534       call    HMSInit                 ;Reset Clock variables
0x01C5 0x01B5       00535       clrf    R2Time                  ;Clear current relays using time.
0x01C6 0x13B3       00536       bcf     DTMFStatus,7            ;Clear flag as all done
0x01C7 0x28E7       00537       goto    Done                    ;All done
                    00538 ;*****************************************************************************
                    00539 CTone:                                        ;Comfort Tone on falling edge of DTMF
0x01C8 0x1A33       00540       btfsc   DTMFStatus,4            ;Check for matching PIN
0x01C9 0x29CC       00541       goto    CTOK
0x01CA 0x1D33       00542       btfss   DTMFStatus,2            ;Tone fail unless PIN valid or Matching
0x01CB 0x1434       00543       bsf     DTMFTone,0
0x01CC 0x15B4       00544 CTOK: bsf     DTMFTone,3              ;Reset PIN Flag SET
0x01CD 0x14B4       00545       bsf     DTMFTone,1              ;Set flag for tone out
0x01CE 0x01C2       00546       clrf    ToneCount               ;Use for down counter
0x01CF 0x1134       00547       bcf     DTMFTone,2              ;Clear toggle divider
0x01D0 0x1683       00548       bsf     STATUS,RP0
0x01D1 0x1085       00549       bcf     TRISA,1                 ;Set b1 for Output
0x01D2 0x1283       00550       bcf     STATUS,RP0
0x01D3 0x0008       00551       return                          ;Conditions set to toggle on T0
                    00552 T0Tone:                                       ;Timer 0 Tone
0x01D4 0x1CB4       00553       btfss   DTMFTone,1              ;Test for tone out
0x01D5 0x0008       00554       return                          ;Nope so back again
0x01D6 0x3004       00555       movlw   0x04                    ;Toggle bit if reqd.
0x01D7 0x1834       00556       btfsc   DTMFTone,0
0x01D8 0x06B4       00557       xorwf   DTMFTone                ;Toggle divider
0x01D9 0x1934       00558       btfsc   DTMFTone,2              ;If clear then toggle Output
0x01DA 0x0008       00559       return                          ;Dividing by two so return
0x01DB 0x3002       00560       movlw   0x02                    ;Toggle bit for output
0x01DC 0x0685       00561       xorwf   PORTA                   ;change state of Output bit
0x01DD 0x0BC2       00562       decfsz  ToneCount               ;255 ms Count
0x01DE 0x29E0       00563       goto    TReset
0x01DF 0x29E8       00564       goto    ToneEnd
0x01E0 0x1C85       00565 TReset:       btfss   PORTA,1                 ;Test Whats sent
0x01E1 0x29E5       00566       goto    LowReset
0x01E2 0x1D05       00567       btfss   PORTA,2
0x01E3 0x11B4       00568       bcf     DTMFTone,3              ;No match so clear PIN RESET flag
0x01E4 0x0008       00569       return
                    00570 LowReset:
0x01E5 0x1905       00571       btfsc   PORTA,2
0x01E6 0x11B4       00572       bcf     DTMFTone,3              ;No match so clear PIN RESET flag
0x01E7 0x0008       00573       return
                    00574 ToneEnd:                              ;Finish Tone output
0x01E8 0x30F8       00575       movlw   0xF8
0x01E9 0x05B4       00576       andwf   DTMFTone                ;Clear all Tone flags
0x01EA 0x1683       00577       bsf     STATUS,RP0
0x01EB 0x1485       00578       bsf     TRISA,1                 ;Set PORTA1 back to input
0x01EC 0x1283       00579       bcf     STATUS,RP0
0x01ED 0x1DB4       00580       btfss   DTMFTone,3              ;RESET PIN ?
0x01EE 0x0008       00581       return                          ;All done
                    00582 PINReset:
0x01EF 0x01C3       00583       clrf    EEPromAdr               ;Point at Zero
0x01F0 0x3004       00584       movlw   0x04
0x01F1 0x00AB       00585       movwf   CopyCount
0x01F2 0x30FF       00586 RLoop:        movlw   0xFF                    ;Set the default
0x01F3 0x2232       00587       call    EEPROMWRITE
0x01F4 0x0AC3       00588       incf    EEPromAdr
0x01F5 0x0BAB       00589       decfsz  CopyCount
0x01F6 0x29F2       00590       goto    RLoop
0x01F7 0x0008       00591       return                          ;Done
                    00592 CheckStars:                           ;Check for all PIN digits being *'s
0x01F8 0x01C3       00593       clrf    EEPromAdr               ;Point at first
0x01F9 0x2228       00594 PSTest:       call    EEPROMREAD              ;Check it
0x01FA 0x3C0B       00595       sublw   0x0B                    ;for * =0x0B
0x01FB 0x1D03       00596       skpz
0x01FC 0x0008       00597       return                          ;No match so return doing nothing
0x01FD 0x0AC3       00598       incf    EEPromAdr               ;Move up to next PIN digit
0x01FE 0x1D43       00599       btfss   EEPromAdr,2             ;Check for fourth digit
0x01FF 0x29F9       00600       goto    PSTest                  ;less than 4 checked
                    00601       movfw   DTMFStatus              ;Get the status bits
0x0200 0x0833       MACRO
0x0201 0x39F8       00602       andlw   0xF8                    ;retain all but lower 3 bits
0x0202 0x3804       00603       iorlw   0x04                    ;set just the Valid PIN flag
0x0203 0x00B3       00604       movwf   DTMFStatus              ;Save it again
0x0204 0x0008       00605       return                          ;return telling Valid Pin receieved.
                    00606
                    00001
                    00002 ; *** Input File - D:\Program Files\FED\WIZPIC\APPWIZ\Delays.asm
                    00003
 TRUE               00004 #ifdef DELAYUsed
                    00005 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    00006 ;
                    00007 ; Delay routines
                    00008 ;
                    00009 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    00010 ;
                    00011 ; Insert a delay of up to 772 Cycles
                    00012 ; Loop time = 5 + 3*(W-1), minimum 5
                    00013 ; Call Delay 1 to add 1 cycle
                    00014 ; Call Delay 2 to add 2 Cyc
                    00015 ;
                    00016 ; Remember it takes 2 Cycles to call
                    00017 ; this routine, and 1 cycle to load
                    00018 ; W before calling it
                    00019 ;
                    00020       MFORCEPAGE D'12'
                    00021
0x0205 0x0000       00022 Delay2        nop                     ; 1
```

```
0x0206 0x0000       00023 Delay1         nop                   ; 1
0x0207 0x0090       00024 Delay0         movwf Temp1           ; 1
0x0208 0x0B90       00025 DelayLop decfsz Temp1         ; 1/2
0x0209 0x2A08       00026        goto DelayLop          ; 2
0x020A 0x0008       00027        return                 ; 2
                    00028
                    00029 ;
                    00030 ; Big delays need an outer loop
                    00031 ; This delays 730 + (W-1)*728 Cycles
                    00032 ;
0x020B 0x0091       00033 BigDel        movwf Temp2            ; 1
0x020C 0x30F0       00034 BDLop movlw H'f0'            ; 1
0x020D 0x2207       00035       call Delay0            ; 724
0x020E 0x0B91       00036       decfsz Temp2           ; 1/2
0x020F 0x2A0C       00037       goto BDLop             ; 2
0x0210 0x0008       00038       return                 ; 2
                    00039 #endif
                    00040
                    00041
                    00042
                    00043 ; *** Input File - D:\Program Files\FED\WIZPIC\APPWIZ\Misc.asm
                    00044
                    00045 ;
                    00046 ; Code associated with Edge element which can be duplicated
                    00047 ;
                    00048
 TRUE               00049 #ifdef EdgeUsed              ; Init Code for Edge is only used once
                    00050       MFORCEPAGE 15
                    00051
                    00052 InitEdge:
                    00053       EICnt=(D'2' -1)/8+1     ;Edge Detector
0x0211 0x30FF       00054       movlw H'FF'
 TRUE               00055       while EICnt
0x0212 0x0093       00056       movwf EdgeFlags+EICnt-1
                    00057       EICnt=EICnt-1
                    00058       endw
                    00059       ClearPCLATH InitEdge
0x0213 0x0008       00060       return
                    00061 #endif
                    00062
                    00063
 TRUE               00064 #ifdef EdgeUsed              ; %% - flag to show code repeated for each instance
                    00065               MFORCEPAGE D'15'
                    00066 TestEdge0
                    00067               EIFlag=0/8      ; Flag for this instance
                    00068               EIBit=(0)%8     ; Bit for this instance
                    00069               movfw PORTA
0x0214 0x0805       MACRO
0x0215 0x3901       00070               andlw (1<<0 )
 TRUE               00071               #ifdef GotEdge0Used
 TRUE               00072                #if D'1'               ; For rising edge SelEdge is 1
0x0216 0x1903       00073                  skpnz
0x0217 0x2A1A       00074                  goto TEJump0
0x0218 0x1C13       00075                  btfss EdgeFlags+EIFlag,EIBit
0x0219 0x168F       00076                  bsf Flag1 ,5
                    00083               #endif
0x021A 0x1013       00084 TEJump0       bcf EdgeFlags+EIFlag,EIBit      ; Set the last state flag
0x021B 0x1D03       00085               skpz
0x021C 0x1413       00086               bsf EdgeFlags+EIFlag,EIBit
                    00087               ClearPCLATH TestEdge0           ; Return address
0x021D 0x0008       00088               return
                    00089 #endif
 TRUE               00090 #ifdef EdgeUsed              ; %% - flag to show code repeated for each instance
                    00091               MFORCEPAGE D'15'
                    00092 TestEdge1
                    00093               EIFlag=1/8      ; Flag for this instance
                    00094               EIBit=(1)%8     ; Bit for this instance
                    00095               movfw PORTA
0x021E 0x0805       MACRO
0x021F 0x3901       00096               andlw (1<<0 )
 TRUE               00097               #ifdef GotEdge1Used
                    00103                #else                  ; For falling edge
0x0220 0x1D03       00104                  skpz
0x0221 0x2A24       00105                  goto TEJump1
0x0222 0x1893       00106                  btfsc EdgeFlags+EIFlag,EIBit
0x0223 0x140F       00107                  bsf Flag1 ,0
                    00108                #endif
                    00109               #endif
0x0224 0x1093       00110 TEJump1       bcf EdgeFlags+EIFlag,EIBit      ; Set the last state flag
0x0225 0x1D03       00111               skpz
0x0226 0x1493       00112               bsf EdgeFlags+EIFlag,EIBit
                    00113               ClearPCLATH TestEdge1           ; Return address
0x0227 0x0008       00114               return
                    00115 #endif
                    00116
                    00117
                    00118
                    00119
                    00120
                    00121
                    00122 ; *** Input File - D:\Program Files\FED\WIZPIC\APPWIZ\EEPROM.asm
                    00123
 TRUE               00124 #ifdef EEPROMUsed;
                    00125 ; Assembler file for Element : EEPROM
                    00126 ;   - EEPROM
                    00127 ;
                    00128 ; Written by FED Element Editor - Sat Jan 05 21:33:13 2002
                    00129 ;
                    00130 ; Variables
                    00131
            00132       cblock
00C3        00133        EEPromAdr              ; EEPromAdr
00C3                00134       endc
                    00135
                    00136 ; ***************************************
                    00137 ;
                    00138 ; Parameters used are :
                    00139 ;
                    00140 ; ***************************************
                    00141 ;
                    00142 ; Occurrences are :
```

```
                  00143 ;
                  00144 ; ****************************************
                  00145 ;
                  00146 ; Inputs & Outputs are :
                  00147
                  00148 ; Code
                  00149
                  00150
                  00151 ; **********
                  00152 ; * EEPROMREAD - EEPROMREAD
                  00153 ; **********
                  00154 ;General routine to read PIC EEPROM data calld with address in EEPromAdr
                  00155 ;returns with data in W.  Coded for 16x8x, 16F62x, 16F82 & 16F8xx processors.
                  00156 EEPROMREAD:
                  00157       ADSetRP EEPromAdr
0x0228 0x1683     MACRO    bsf STATUS,RP0
                  00158       movfw   EEPromAdr       ;Get the address into W
0x0229 0x0843     MACRO
                  00168   #else                               ;16x87x & 16F8x processors
0x022A 0x1283     00169       bcf     STATUS,RP0      ;
0x022B 0x0089     00170       movwf   EEADR           ;Save the address
0x022C 0x1683     00171       bsf     STATUS,RP0      ;All processors have EECONx in blocks 1 or 3
                  00172   #endif
0x022D 0x1408     00176       bsf     EECON1,RD       ;Tell it to read data
                  00179   #else
0x022E 0x1283     00180       bcf     STATUS,RP0      ;Block 2 for 16F87X - 0 for 16F84
                  00181       movfw   EEDATA          ;Get the data
0x022F 0x0808     MACRO
                  00182   #endif
                  00183       ADClearRP EECON2
0x0230 0x1283     MACRO    bcf STATUS,RP0
                  00184       ClearPCLATH EEPROMREAD
0x0231 0x0008     00185       return                  ;Thats it.
                  00186
                  00187 ; **********
                  00188 ; * EEPROMWRITE - EEPROMWRITE
                  00189 ; **********
                  00190 ;General routine to write EEPROM data called with address in EEPromAdr
                  00191 ;data to be saved in W.  Global Interrupts are diabled & restored during
                  00192 ;actual write period, returns only after actual write has completed.
                  00193 EEPROMWRITE:
0x0232 0x1283     00197       bcf     STATUS,RP0      ;block 2 for 16F87X - 0 for 16F84
0x0233 0x0088     00198       movwf   EEDATA          ;Save data
                  00199       ADSetRP EEPromAdr
0x0234 0x1683     MACRO    bsf STATUS,RP0
                  00200       movfw   EEPromAdr       ;Get the address into W
0x0235 0x0843     MACRO
                  00210   #else                               ;16x87x & 16F8x processors
0x0236 0x1283     00211       bcf     STATUS,RP0      ;
0x0237 0x0089     00212       movwf   EEADR           ;Save the address
0x0238 0x1683     00213       bsf     STATUS,RP0      ;All processors have EECONx in blocks 1 or 3
                  00214   #endif
0x0239 0x1003     00218       clrc                    ;Use the Carry flag to store GIE state
0x023A 0x1B8B     00219       btfsc   INTCON,7        ;Check GIE bit
0x023B 0x1403     00220       setc                    ;Set C as GIE is set
0x023C 0x138B     00221       bcf     INTCON,GIE      ;Disable interrupts during critical code
0x023D 0x1508     00222       bsf     EECON1,WREN     ;Enable write mode
                  00223 ;The following sequence is mandatory
0x023E 0x3055     00224       movlw   0x55
0x023F 0x0089     00225       movwf   EECON2
0x0240 0x30AA     00226       movlw   0xAA
0x0241 0x0089     00227       movwf   EECON2
0x0242 0x1488     00228       bsf     EECON1,WR       ;Write the data
                  00229 ;End of mandatory sequence
0x0243 0x1888     00230       btfsc   EECON1,WR       ;Check for cleared by hardware
0x0244 0x2A43     00231       goto    $-1             ;loop until written
0x0245 0x1108     00232       bcf     EECON1,WREN     ;Diasable write
0x0246 0x1803     00233       skpnc                   ;Test C for GIE state
0x0247 0x178B     00234       bsf     INTCON,GIE      ;Set it as it was
                  00235       ADClearRP EECON2        ;clear RPbits
0x0248 0x1283     MACRO    bcf STATUS,RP0
                  00236       ClearPCLATH EEPROMWRITE
0x0249 0x0008     00237       return
                  00238 #endif
                  00239
                  00240
                  00241 ; *** Input File - D:\Program Files\FED\WIZPIC\APPWIZ\Timers.asm
                  00242
                  00243
                  00244
                  00245 ; ***
                  00246 ; Original file starts here
                  00247 ; ***
                  00248 ;
                  00249 ; Application Designer, timers code
                  00250 ;
                  00251
                  00252
                  00253 ;
                  00254 ; Hours,Minutes & Seconds
                  00255 ;
                  00256
 TRUE             00257 #ifdef HMSUsed
                  00258               HMS_T0OF=D'256'
 TRUE             00259               #if D'1'
                  00260                HMS_T0OF*=D'4'
                  00261               #endif
                  00262               SECCOUNTS=(D'4011200' /4)/HMS_T0OF
                  00263               MINCOUNTS=(D'4011200' *D'15')/HMS_T0OF
                  00264               HOURCOUNTS=((D'4011200' *D'30')/HMS_T0OF)*D'30'
                  00265
                  00266               MFORCEPAGE D'30'
                  00267 HMSInit:
                  00268               ADSetRP HMSCount
0x024A 0x1683     MACRO    bsf STATUS,RP0
 TRUE             00269               #ifdef HourPassUsed
0x024B 0x2250     00270                call HourInit
                  00271               #endif
 TRUE             00272               #ifdef MinPassUsed
0x024C 0x2255     00273                call MinInit
                  00274               #endif
```

```
 TRUE                00275                #ifdef SecPassUsed
0x024D 0x2259        00276                 call SecInit
                     00277                #endif
                     00278                ADClearRP HMSCount
0x024E 0x1283        MACRO    bcf STATUS,RP0
                     00279                ClearPCLATH HMSInit
0x024F 0x0008        00280                return
                     00281
 TRUE                00282 #ifdef HourPassUsed
0x0250 0x01A1        00283 HourInit:      clrf HrCount
0x0251 0x01A2        00284                clrf HrCount+1
0x0252 0x01A3        00285                clrf HrCount+2
0x0253 0x01A4        00286                clrf HrCount+3
0x0254 0x0008        00287                return
                     00288 #endif
 TRUE                00289 #ifdef MinPassUsed
0x0255 0x01A5        00290 MinInit:       clrf MinCount
0x0256 0x01A6        00291                clrf MinCount+1
0x0257 0x01A7        00292                clrf MinCount+2
0x0258 0x0008        00293                return
                     00294 #endif
 TRUE                00295 #ifdef SecPassUsed
0x0259 0x01A8        00296 SecInit:       clrf SecCount
0x025A 0x01A9        00297                clrf SecCount+1
0x025B 0x0008        00298                return
                     00299 #endif
                     00300                MFORCEPAGE D'75'
                     00301 ;
                     00302 ; Called every timer 0 overflow
                     00303 ;
                     00304 HMSTick:
                     00305                ADSetRP HMSCount
0x025C 0x1683        MACRO    bsf STATUS,RP0
 TRUE                00306  #ifdef SecPassUsed
0x025D 0x0AA9        00307                incf SecCount+1
0x025E 0x0FA8        00308                incfsz SecCount
0x025F 0x03A9        00309                decf SecCount+1
                     00310 HMS_NIS:
0x0260 0x30D3        00311                movlw SECCOUNTS>>0
0x0261 0x0228        00312                subwf SecCount+0,w
0x0262 0x1D03        00313                skpz
0x0263 0x2A6A        00314                goto HMS_NoOccurS
0x0264 0x3003        00315                movlw SECCOUNTS>>8
0x0265 0x0229        00316                subwf SecCount+1,w
0x0266 0x1D03        00317                skpz
0x0267 0x2A6A        00318                goto HMS_NoOccurS
0x0268 0x150F        00319 HMS_SecPassed:        bsf Flag1 ,2
0x0269 0x2259        00320                call SecInit
                     00321 HMS_NoOccurS:
                     00322  #endif
 TRUE                00323  #ifdef MinPassUsed
0x026A 0x0FA5        00324                incfsz MinCount
0x026B 0x2A6F        00325                goto HMS_NIM
0x026C 0x0FA6        00326                incfsz MinCount+1
0x026D 0x2A6F        00327                goto HMS_NIM
0x026E 0x0AA7        00328                incf MinCount+2
                     00329 HMS_NIM:
0x026F 0x3085        00330                movlw MINCOUNTS>>0
0x0270 0x0225        00331                subwf MinCount+0,w
0x0271 0x1D03        00332                skpz
0x0272 0x2A7E        00333                goto HMS_NoOccurM
0x0273 0x30E5        00334                movlw MINCOUNTS>>8
0x0274 0x0226        00335                subwf MinCount+1,w
0x0275 0x1D03        00336                skpz
0x0276 0x2A7E        00337                goto HMS_NoOccurM
0x0277 0x3000        00338                movlw MINCOUNTS>>D'16'
0x0278 0x0227        00339                subwf MinCount+2,w
0x0279 0x1D03        00340                skpz
0x027A 0x2A7E        00341                goto HMS_NoOccurM
0x027B 0x158F        00342                bsf Flag1 ,3
0x027C 0x2255        00343                call MinInit
 TRUE                00344       #ifdef SecPassUsed
0x027D 0x2259        00345                call SecInit
                     00346       #endif
                     00347 HMS_NoOccurM:
                     00348  #endif
 TRUE                00349  #ifdef HourPassUsed
0x027E 0x0FA1        00350                incfsz HrCount
0x027F 0x2A85        00351                goto HMS_NIH
0x0280 0x0FA2        00352                incfsz HrCount+1
0x0281 0x2A85        00353                goto HMS_NIH
0x0282 0x0FA3        00354                incfsz HrCount+2
0x0283 0x2A85        00355                goto HMS_NIH
0x0284 0x0AA4        00356                incf HrCount+3
                     00357 HMS_NIH:
0x0285 0x304A        00358                movlw HOURCOUNTS>>0
0x0286 0x0221        00359                subwf HrCount+0,w
0x0287 0x1D03        00360                skpz
0x0288 0x2A97        00361                goto HMS_NoOccurH
0x0289 0x30CB        00362                movlw HOURCOUNTS>>8
0x028A 0x0222        00363                subwf HrCount+1,w
0x028B 0x1D03        00364                skpz
0x028C 0x2A97        00365                goto HMS_NoOccurH
0x028D 0x3035        00366                movlw HOURCOUNTS>>D'16'
0x028E 0x0223        00367                subwf HrCount+2,w
0x028F 0x1D03        00368                skpz
0x0290 0x2A97        00369                goto HMS_NoOccurH
0x0291 0x3000        00370                movlw HOURCOUNTS>>D'24'
0x0292 0x0224        00371                subwf HrCount+3,w
0x0293 0x1D03        00372                skpz
0x0294 0x2A97        00373                goto HMS_NoOccurH
0x0295 0x160F        00374                bsf Flag1 ,4
0x0296 0x2A4A        00375                goto HMSInit
                     00376 HMS_NoOccurH:
                     00377  #endif
                     00378 HMS_End:
                     00379                ADClearRP HMSCount
0x0297 0x1283        MACRO    bcf STATUS,RP0
                     00380                ClearPCLATH HMSTick
0x0298 0x0008        00381                return
                     00382 #endif
```

```
                  00383
                  00384
                  00385
                  00386
                  00387
                  00388
                  00389
                  00390
                  00391
                  00392
                  00393
                  00394
                  00395
                  00396
                  00397
                  00398
                  00399
                  00400
                  00401


;*
;*** END OF PROGRAM
;*

Labels
======

ADClearRP                       MACRO
ADInit                          00000019H, .25
ADInitEnd                       00000029H, .41
ADSetIRP                        MACRO
ADSetPCLATH                     MACRO
ADSetRP                         MACRO
APROCFREQ                       DEFINE - D'4011200'
AutoInt                         0000000BH, .11
BDLop                           0000020CH, .524
BITSIZE                         DEFINE - D'14'
BOOTADDRESS                     DEFINE - D'0'
BigCyc                          0000010EH, .270
BigDel                          0000020BH, .523
C                               00000000H, .0
CLoop                           000000CEH, .206
CRE                             DEFINE - D'1'
CSLoop                          000001B9H, .441
CSaveDigits                     00000195H, .405
CStore                          000001B7H, .439
CTOK                            000001CCH, .460
CTone                           000001C8H, .456
CheckStars                      000001F8H, .504
Choose                          00000095H, .149
ClearPCLATH                     MACRO
Clocked                         00000085H, .133
CopyCount                       000000ABH, .171
CopyPIN                         000000CCH, .204
CountDown                       00000120H, .288
Current                         000000AAH, .170
DBBITRATE                       00004B00H, .19200
DBBase                          00000700H, .1792
DBPROCFREQ                      003D34C0H, .4011200
DBPages                         00000001H, .1
DBRTB                           00000000H, .0
DBxPort                         00000005H, .5
DC                              00000001H, .1
DELAY                           MACRO
DELAYUsed                       DEFINE - 1
DTMFStatus                      000000B3H, .179
DTMFTone                        000000B4H, .180
DebugVars                       000000E2H, .226
Delay0                          00000207H, .519
Delay1                          00000206H, .518
Delay2                          00000205H, .517
DelayLop                        00000208H, .520
Delv                            00000000H, .0
Done                            000000E7H, .231
EEADR                           00000009H, .9
EECON1                          00000088H, .136
EECON2                          00000089H, .137
EEDATA                          00000008H, .8
EEIE                            00000006H, .6
EEIF                            00000004H, .4
EEPROMREAD                      00000228H, .552
EEPROMUsed                      DEFINE - 1
EEPROMWRITE                     00000232H, .562
EEPromAdr                       000000C3H, .195
EIBit                           00000001H, .1
EICnt                           00000000H, .0
EIFlag                          00000000H, .0
Edge0Bit                        DEFINE - 0
Edge0Port                       DEFINE - PORTA
Edge0Used                       DEFINE - 1
Edge1Bit                        DEFINE - 0
Edge1Port                       DEFINE - PORTA
Edge1Used                       DEFINE - 1
EdgeFlags                       00000013H, .19
EdgeUsed                        DEFINE - 1
EdgenCopy                       DEFINE - D'2'
F                               00000001H, .1
FIRSTRAM                        DEFINE - H'0C'
FITINBLOCK                      MACRO
FSR                             00000004H, .4
FSR_TEMP                        0000000EH, .14
Flag1                           0000000FH, .15
GIE                             00000007H, .7
GotEdge0End                     00000048H, .72
GotEdge0Flag                    DEFINE - Flag1
GotEdge0FlagBit                 DEFINE - 5
GotEdge0Used                    DEFINE - 1
GotEdge1End                     00000032H, .50
GotEdge1Flag                    DEFINE - Flag1
GotEdge1FlagBit                 DEFINE - 0
```

```
GotEdge1Used                    DEFINE - 1
HASOSCCAL                       DEFINE - 0
HMSCount                        000000A1H, .161
HMSInit                         0000024AH, .586
HMSStart                        00000014H, .20
HMSTick                         0000025CH, .604
HMSUsed                         DEFINE - 1
HMS_End                         00000297H, .663
HMS_NIH                         00000285H, .645
HMS_NIM                         0000026FH, .623
HMS_NIS                         00000260H, .608
HMS_NoOccurH                    00000297H, .663
HMS_NoOccurM                    0000027EH, .638
HMS_NoOccurS                    0000026AH, .618
HMS_SecPassed                   00000268H, .616
HMS_T0OF                        00000400H, .1024
HMSnVar                         00000009H, .9
HOURCOUNTS                      0035CB4AH, .3525450
HOff                            000000B7H, .183
HOn                             000000B9H, .185
HourInc                         0000017CH, .380
HourInit                        00000250H, .592
HourPassEnd                     00000043H, .67
HourPassFlag                    DEFINE - Flag1
HourPassFlagBit                 DEFINE - 4
HourPassUsed                    DEFINE - 1
Hours                           000000BBH, .187
HrCount                         000000A1H, .161
HrCount1                        000000A2H, .162
HrCount2                        000000A3H, .163
HrCount3                        000000A4H, .164
INDF                            00000000H, .0
INITOUTPORT                     MACRO
INTCON                          0000000BH, .11
INTE                            00000004H, .4
INTEDG                          00000006H, .6
INTF                            00000001H, .1
INTRECALL                       MACRO
INTSAVE                         MACRO
IRP                             00000007H, .7
In0Bit                          DEFINE - 4
In0Port                         DEFINE - PORTA
In1Bit                          DEFINE - 3
In1Port                         DEFINE - PORTA
In2Bit                          DEFINE - 2
In2Port                         DEFINE - PORTA
In3Bit                          DEFINE - 1
In3Port                         DEFINE - PORTA
In4Bit                          DEFINE - 0
In4Port                         DEFINE - PORTA
InitEdge                        00000211H, .529
InitialValue0                   DEFINE - D'0'
InitialValue1                   DEFINE - D'0'
InitialValue2                   DEFINE - D'0'
InitialValue3                   DEFINE - D'0'
InitialValue4                   DEFINE - D'0'
InitialValue5                   DEFINE - D'0'
InitialValue6                   DEFINE - D'0'
InitialValue7                   DEFINE - D'0'
IntPri                          0000000AH, .10
LASTRAM                         DEFINE - H'4F'
LoadTimer                       0000010DH, .269
LoopDelay                       00000103H, .259
LowReset                        000001E5H, .485
MFORCEPAGE                      MACRO
MINCOUNTS                       0000E585H, .58757
MOff                            000000B6H, .182
MOn                             000000B8H, .184
Main                            0000002DH, .45
MatchIt                         000000C1H, .193
MinCount                        000000A5H, .165
MinCount1                       000000A6H, .166
MinCount2                       000000A7H, .167
MinInit                         00000255H, .597
MinPassEnd                      0000003FH, .63
MinPassFlag                     DEFINE - Flag1
MinPassFlagBit                  DEFINE - 3
MinPassUsed                     DEFINE - 1
Minutes                         000000BAH, .186
MinutesOff                      00000171H, .369
NCAddDigit                      0000018AH, .394
NF                              000000E5H, .229
NOT_PD                          00000003H, .3
NOT_RBPU                        00000007H, .7
NOT_TO                          00000004H, .4
NT2                             0000010CH, .268
NTAddDigit                      00000147H, .327
NTReset                         00000156H, .342
NTSave                          00000100H, .256
NTSaveIt                        00000153H, .339
NTSoR                           000000FBH, .251
NTStore                         000000EAH, .234
NTime                           0000013CH, .316
NewClock                        00000183H, .387
NewPIN                          000000BAH, .186
NewTime                         000000AEH, .174
NewTimes                        000000F5H, .245
OPTION_REG                      00000081H, .129
OfEnd                           00000037H, .55
OfFlag                          DEFINE - Flag1
OfFlagBit                       DEFINE - 1
OfUsed                          DEFINE - 1
OperateRelay                    00000071H, .113
Out0Bit                         DEFINE - 0
Out0Port                        DEFINE - PORTB
Out1Bit                         DEFINE - 1
Out1Port                        DEFINE - PORTB
Out2Bit                         DEFINE - 2
Out2Port                        DEFINE - PORTB
Out3Bit                         DEFINE - 3
Out3Port                        DEFINE - PORTB
```

```
Out4Bit                         DEFINE - 4
Out4Port                        DEFINE - PORTB
Out5Bit                         DEFINE - 5
Out5Port                        DEFINE - PORTB
Out6Bit                         DEFINE - 6
Out6Port                        DEFINE - PORTB
Out7Bit                         DEFINE - 7
Out7Port                        DEFINE - PORTB
PCL                             00000002H, .2
PCLATH                          0000000AH, .10
PINReset                        000001EFH, .495
PINfail                         000000FDH, .253
PORTA                           00000005H, .5
PORTB                           00000006H, .6
PS0                             00000000H, .0
PS1                             00000001H, .1
PS2                             00000002H, .2
PSA                             00000003H, .3
PSD                             DEFINE - D'4'
PSTest                          000001F9H, .505
Pins                            000000D7H, .215
PortIn0Used                     DEFINE - 1
PortIn1Used                     DEFINE - 1
PortIn2Used                     DEFINE - 1
PortIn3Used                     DEFINE - 1
PortIn4Used                     DEFINE - 1
PortInUsed                      DEFINE - 1
PortInnCopy                     DEFINE - D'5'
PortOut0Used                    DEFINE - 1
PortOut1Used                    DEFINE - 1
PortOut2Used                    DEFINE - 1
PortOut3Used                    DEFINE - 1
PortOut4Used                    DEFINE - 1
PortOut5Used                    DEFINE - 1
PortOut6Used                    DEFINE - 1
PortOut7Used                    DEFINE - 1
PortOutUsed                     DEFINE - 1
PortOutnCopy                    DEFINE - D'8'
R2T                             000000B0H, .176
R2Time                          000000B5H, .181
RBIE                            00000003H, .3
RBIF                            00000000H, .0
RCD                             00000126H, .294
RD                              00000000H, .0
RLoop                           000001F2H, .498
RP0                             00000005H, .5
RP1                             00000006H, .6
RTOperate                       00000133H, .307
RTRelease                       0000012EH, .302
RTSetupTime                     00000137H, .311
ReadDTMF                        00000112H, .274
Relay                           000000AFH, .175
RelayCD                         000000B1H, .177
RelayTState                     000000B2H, .178
RelayToggle                     0000007AH, .122
Release                         0000008CH, .140
ResetValues                     00000159H, .345
Rotate                          000000B4H, .180
SECCOUNTS                       000003D3H, .979
STATUS                          00000003H, .3
STATUS_TEMP                     0000000DH, .13
SecCount                        000000A8H, .168
SecCount1                       000000A9H, .169
SecInit                         00000259H, .601
SecPassEnd                      0000003BH, .59
SecPassFlag                     DEFINE - Flag1
SecPassFlagBit                  DEFINE - 2
SecPassUsed                     DEFINE - 1
SelEdge0                        DEFINE - D'1'
SelEdge1                        DEFINE - D'0'
SmallCyc                        0000010BH, .267
Start                           00000016H, .22
T0CS                            00000005H, .5
T0IE                            00000005H, .5
T0IF                            00000002H, .2
T0SE                            00000004H, .4
T0Tone                          000001D4H, .468
TCount                          000000ACH, .172
TEJump0                         0000021AH, .538
TEJump1                         00000224H, .548
TH                              000000C1H, .193
THOff                           000000BDH, .189
THOn                            000000BFH, .191
TM                              000000C0H, .192
TMOff                           000000BCH, .188
TMOn                            000000BEH, .190
TMR0                            00000001H, .1
TRISA                           00000085H, .133
TRISB                           00000086H, .134
TReset                          000001E0H, .480
Temp1                           00000010H, .16
Temp2                           00000011H, .17
Temp3                           00000012H, .18
TestClock                       000000AAH, .170
TestEdge0                       00000214H, .532
TestEdge1                       0000021EH, .542
TestRelay                       000000B3H, .179
TestT                           0000009BH, .155
TimeCD                          000000ADH, .173
TimeHour                        00000182H, .386
TimeMin                         00000161H, .353
TimesT                          000000A2H, .162
Tmr0IntUsed                     DEFINE - 1
Tmr0Used                        DEFINE - 1
Tmr0_Int1                       0000000FH, .15
ToneCount                       000000C2H, .194
ToneEnd                         000001E8H, .488
UseOsc                          DEFINE - D'1'
UsePS                           DEFINE - D'1'
UserInitialise                  0000004AH, .74
UserInterrupt                   00000049H, .73
```

```
UserLoop                                0000005CH, .92
ValidateTimes                           0000019FH, .415
W                                       00000000H, .0
WR                                      00000001H, .1
WREN                                    00000002H, .2
WRERR                                   00000003H, .3
W_TEMP                                  0000000CH, .12
Z                                       00000002H, .2
_CP_OFF                                 00003FFFH, .16383
_CP_ON                                  0000000FH, .15
_HS_OSC                                 00003FFEH, .16382
_LP_OSC                                 00003FFCH, .16380
_PWRTE_OFF                              00003FFFH, .16383
_PWRTE_ON                               00003FF7H, .16375
_RC_OSC                                 00003FFFH, .16383
_WDT_OFF                                00003FFBH, .16379
_WDT_ON                                 00003FFFH, .16383
_XT_OSC                                 00003FFDH, .16381
__16F84                                 00000001H, .1
__PICDE                                 00000001H, .1
lPSD                                    00000000H, .0
longcall                                MACRO
nPAGESRAM                               DEFINE - 1
nPAGESROM                               DEFINE - 1
vINTCON                                 00000020H, .32
vOPTION_REG                             00000041H, .65
vPIE1                                   00000000H, .0
vPIE2                                   00000000H, .0
vPORTA                                  000000FFH, .255
vPORTB                                  00000000H, .0
vPORTC                                  000000FFH, .255
vPORTD                                  000000FFH, .255
vPORTE                                  000000FFH, .255
vTRISA                                  000000FFH, .255
vTRISB                                  00000000H, .0
vTRISC                                  000000FFH, .255
vTRISD                                  000000FFH, .255
vTRISE                                  00000007H, .7
vor                                     00000001H, .1


Program memory used, 'X'=used, '-'=unused

0x0000 : XXX-XXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x0040 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x0080 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x00C0 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x0100 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x0140 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x0180 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x01C0 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x0200 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x0240 : XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX XXXXXXXXXXXXXXX
0x0280 : XXXXXXXXXXXXXXX XXXXXXXXX------- ---------------- ----------------

Other memory blocks are unused
Errors    : 0
Warnings : 0
Messages : 0
```