```
        #picaxe 08m


'       VER. 2.0        07/30/2007  12:15AM     Changed chirp tone to triple 125
'       VER. 2.1        08/01/2007   6:45PM     Changed chirp tone to double 125 &
added Soundbox



'                       Maine Bugs Picaxe (08M) Project
'            Logical Pin = (#)   _____   (#)
'                         Vcc     [ |_| ]     Ground
'      serial in from PC / (5) [     ] (0) Led 1  /  Attention line from PC
'            Piezo Speaker (4) [ 08M ] (1) Led 2
'                          (3) [     ] (2) Temperature sensor
'                               -----

'       A SHORT course in Picaxe Input / Output designators.

' The commands in the Picaxe microprocessor refer to the Input/Output pins with
' their LOGICAL Input/Output addresses. The PHYSICAL address of the I/O pin is
' different. Here is a simple table of the pins:
'       Logical pin #    Physical pin #
'               0                       7
'               1                       6
'               2                       5
'               3                       4
'               4                       3
'               5                       2

' The LOGICAL pin #0 is shared by both the PC download cable AND one of the LEDs.
' For the Cricket to function properly, disconnect the downloading cable after the
' download procedure....now Logical pin #0 is not shared with anything...


'               When fooling around with BYTES, we usually like to start with bit 0
'               What the heck is bit 0?  ...you say!

'
*******************************************************************************
*
'
'       A short course in Bits and Bytes!
'
'
===============================================================================
=
'
'       BINARY is a number system where there are only 2 (hence BI) digits 0 and 1
'               Computers run on BINARY because there are 2 states for the logic
contained
'               inside them...OFF (0)  and ON (1).
'               Groups of digits that are commonly used are:
'                               Bit     =  1 digit      or                    b
'                               Slice =  2 Bits or                   bb
'                               Nibble=  4 Bits or          bbbb
'                               Byte    =  8 Bits       or              bbbbbbbb
'                       and a   Word    = 16 Bits       or  bbbbbbbbbbbbbbbb
'
'       In little single chip microprocessors, the registers for handling data and
math are
'               typically 8 bits wide so BYTES are the typical form of data they
deal with.
'
'                       A BYTE is 8 bits usually written in a left to right in
asending sequence
'                               of bits where each bit represents an increasing
power of 2
```

```
'               Bit #        7      6     5     4     3     2     1     0

'       Place Value             2^7th  2^6th 2^5th 2^4th 2^3rd 2^2nd 2^1st 2^0th
'               or              128     64    32    16    8     4     2     1

'               So           1      1     1     0     0     0     0     1   =
11100001 (binary)
'           Represents    128  +  64 +  32 +  0  +  0  +  0  +  0  +  1   =   225
 (decimal)


' You already know all this; but, in a different number BASE.....decimal
'                               Decimal, the numbers that we use, are digits (0-9)
written from
'                               right to left where each digit represents an
increasing power of 10

'               Digit #        7       6      5      4      3      2      1      0

'       Place Value             10^7th  10^6th 10^5th 10^4th 10^3rd 10^2nd 10^1st
10^0th
'                             10,000,000                       1,000
10
'                                        1,000,000         10,000
100                 1
'                                                 100,000

'               So             0       7      0      4      0      0      0      0
'           Represents       0+ 7,000,000 + 0 + 40,000  + 0    + 0    + 0    + 0 =
7,040,000 (decimal)


'Now back to the Morse Code
'
'       Here is a nifty way to encode the Morse Code alphabet into BYTE sized
bites...
'               First, start by changing Dits and Dahs into 0's and 1's
respectively.
'                       So
'                               a V would be 0001  (dit dit dit dah)
'
'               Secondly, arrange the zeros and ones from right to left in the
order of sending
'                       So
'                               0001 is now 1000

'                       add any zeros to make the code always 5 bits wide
'                       So
'                               1000 now becomes 01000
'
'               Next, take the number of dits and dahs in the Morse character
and...
'                       write that in binary
'                       So
'                               4 dits & dahs becomes 100

'                       and stick it in front of the 5 bit code from the step above
'                       So
'                               100 + 01000 = 10001000
'                       which happens to make the encoded characters always 1 byte
long
'                       So
'                               the V character = 10001000 (binary) = 128 + 16 =
144 (decimal)
'
'               Next, when you want to decode the data, you mask off the top 3
bits.
'                       To mask off bits, logical 'AND' the data with a mask byte
which
```

```
'                       contains 1's only in the bits you want to have in the
result.
'                       So
'                               10001000 & 11100000 yields 10000000 = 128 (decimal)
'                       And then divide the result by 32 which happens to be the
value of the
'                       place for bit5 which is the lowest 1 bit in the mask
'                       Now
'                               10000000 /32 (or shifted 5 places) = 00000100 or 4
(decimal)
'                       and there are 4 dits and dahs in V.
'
'
'               Lastly, when you want to check the data for dits and dahs...
'                       Starting at the very right, or bit0, construct a mask
containg ONLY
'                       that bit set and logical 'AND' the mask with the data byte.
'                       If the result is zero, then the bit you are checking is a 0
or DIT
'                       If the result is greater than zero, then it is a 1 or DAH.
'                       Continue checking the data bits in the BYTE until you have
looked at
'                       the proper number of bits as coded in the upper 3 bit
value.
'
'                       So from the top 3 bits
'                               10000000 & 11100000 = 10000000 = 128 (decimal) /32
= 4 elements
'
'                               10001000 & 00000001 = 00000000 = dit = element 1
'                               10001000 & 00000010 = 00000000 = dit = element 2
'                               10001000 & 00000100 = 00000000 = dit = element 3
'                               10001000 & 00001000 = 00001000 = dah = element 4
'
'       and that is how the Morse characters are coded into data BYTES
'
'                       Morse Code characters encoded into BYTES
'                                                       ELEMENTS
'                  Char.  Morse Code          #  CODE           BYTE    DECIMAL
'                     A       dit dah.............2   01..........01000010    66
'                     B       dah dit dit dit.....4   1000.........10000001    129
'                     C       dah dit dah dit.....4   1010.........10000101    133
'                     D       dah dit dit.........3   100..........01100001    97
'                     E       dit.................1   0............00100000    32
'                     F       dit dit dah dit.....4   0010.........10000100    132
'                     G       dah dah dit.........3   110..........01100011    99
'                     H       dit dit dit dit.....4   0000.........10000000    128
'                     I       dit dit.............2   00...........01000000    64
'                     J       dit dah dah dah.....4   0111.........10001110    142
'                     K       dah dit dah.........3   101..........01100101    101
'                     L       dit dah dit dit.....4   0100.........10000010    130
'                     M       dah dah.............2   11...........01000011    67
'                     N       dah dit.............2   10...........01000001    65
'                     O       dah dah dah.........3   111..........01100111    103
'                     P       dit dah dah dit.....4   0110.........10000110    134
'                     Q       dah dah dit dah.....4   1101.........10001011    139
'                     R       dit dah dit.........3   010..........01100010    98
'                     S       dit dit dit.........3   000..........01100000    96
'                     T       dah.................1   1............00100001    33
'                     U       dit dit dah.........3   001..........01100100    100
'                     V       dit dit dit dah.....4   0001.........10001000    136
'                     W       dit dah dah.........3   011..........01100110    102
'                     X       dah dit dit dah.....4   1001.........10001001    137
'                     Y       dah dit dah dah.....4   1011.........10001101    141
'                     Z       dah dah dit dit.....4   1100.........10000011    131
'                     1       dit dah dah dah dah.5   01111........10111110    190

'                     2       dit dit dah dah dah.5   00111........10111100    188
'                     3       dit dit dit dah dah.5   00011........10111000    184
```

```
'                       4       dit dit dit dit dah.5   00001.......10110000    176
'                       5       dit dit dit dit dit.5   00000.......10100000    160
'                       6       dah dit dit dit dit.5   10000.......10100001    161
'                       7       dah dah dit dit dit.5   11000.......10100011    163
'                       8       dah dah dah dit dit.5   11100.......10100111    167
'                       9       dah dah dah dah dit.5   11110.......10101111    175
'                       0       dah dah dah dah dah.5   11111.......10111111    191
'
'END OF LESSON!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!


'       PRETENDING TO BE A CRICKET:

'       The formula for determining the temperature is to count the number of
cricket chirps in 15 seconds
'       and adding 39 to calculate the temperature in degrees F. It is said that
this method is accurate
'       to 1 degree F.
'
'       So if you want to emulate a cricket.....
'
'               First determine the temperature, so we read the temperature from a
sensor.
'               Next, subtract 39 to find out how many chirps to chirp.
'               Then, assuming a 50% duty cycle for chirp to silence, divide 15000
(millisecs) by 2 then by
'                               the number of chirps to determine the length of
both the chirp and the silent
'                               interval between chirps.
'               Now you know many chirps, chirp time and time between chirps.
'               Chirp away!




'Define the Input & Output lines

        Output 0        'Led #1 output active low 0=Led on 1=Led off
        Output 1        'Led #2 output active low 0=Led on 1=Led off
        Input   2       'Temperature Sensor 1-Wire Input
        Output 4        'Piezo sounder Output for Chirp


'Define come constants used in generating Morse Code

        Symbol Ditlen= 150      'time length of basic DIT = 150 millisec.
        Symbol Ntrdit=225               'time between element flashes = 225
millisec.
        Symbol Charlen=500      'time between character groups = 500 millisec.
        Symbol Time1=3000               'time between chirping and sending call
sign = 3 sec. or 3000 msec.
        Symbol Time2=5000               'time between sending call sign and
chirping = 5 sec. or 5000 msec.
        Symbol Led1pin=0
        Symbol Led2pin=1
        Symbol Rtmrpin=2

'Define some variables to hold variable data

        Symbol I=b1     'Loop variable for call sign letter position
        Symbol J=b2     'Loop variable for element position in Morse character
        Symbol K=b3     'Variable holding successive Morse bit data from character
        Symbol L=b9     'playing with sound variable Length

        Symbol Mchar=b4 'Variable holding the current Morse character for
transmission
        Symbol Numdit=b5        'Variable holding the # of elements data of the
```

```
Morse character
        Symbol Ditdat=b6        'Variable holding the element data of the Morse
character
        Symbol Mask=b7  'Variable holding changing mask data to obtain
'                                       successive bits in the Morse character
        Symbol Send=w4  'Word length variable holding 'rather' large numbers
'                                       for the time constants Ditlen*3, 500, 1000
        Symbol Time=w5  'Word length variable holding collective time of chirps


Init:   High Led1pin: High Led2pin              'Turn both LEDs off!
        Pause 1000                                      'Wait a sec!

'************ comment out the following line with a ' to play with sounds in the
Soundbox
        Goto Criket

'*********  The SoundBox  *****************

MosQto:
        L=2
        B2=101
        B3=102
        B4=103
        B5=104
        B6=105
        B7=106
        B8=107

        Sound 4,(B2,L,B3,L,B4,L,B5,L,B6,L,B7,L,B8,L)

        PAUSE 1000

        Sound 4,(125,2,0,2,125,2)               'pretty good cricket @ only
(2+2+2)*10ms long (.06sec) better for many chirps

        PAUSE 1000

        Sound 4,(125,2,0,2,125,2,0,2,125,2)     'Triple-tone chirp (source: N1RX
Bruce Beford) @ (2+2+2+2+2)*10ms long (.1sec)

        PAUSE 1000

        GOTO MOSQTO
'****************************************



Criket:Readtemp 2,I                             'Read temperature from sensor on
Pin2 (sensor reads Celsius)

        J=I*9/5+32-39                           'Calculate to degrees F and then
find number of chirps to chirp

'*********  BRUCE N1RX  ******************
'       Time=J*100                                      'Each Triple-Chirp= .1sec
'****************************************

'*********  REX W1REX  ******************
        Time=J*60                                       'double chirp for HOT
weather = .06sec
'****************************************



        Send=15000-Time/J/10                    'Calculate length of inter-chirp
delay..

Chirps:K=1                                              'Start chirping loop for
```

```
number of chirps

Chirp:

        Low LED1pin:Low LED2pin                 'Turn on LEDs for 'chirp'
First:

'***********  BRUCE N1RX  *****************
'       Sound 4,(125,2,0,2,125,2,0,2,125,2)     'Triple-tone chirp (source: N1RX
Bruce Beford)
'****************************************

'***********  REX W1REX  ******************
        Sound 4,(125,2,0,2,125,2)               'Doublle-tone chirp for hot weather
'****************************************

        High Led1pin:High Led2pin               'Turn the LEDs back off

        K=K+1                                           'Rack up a chirp
        If K>J then Chupd                               'All chirped up?
        Sound 4,(0,Send)
        Goto Chirp


Chupd:Pause Time1                                       'Wait Time1 milliseconds
before moving on


        GOTO MORSE


Morse:

'*******************************************************************************
****

        For J=1 to 5                            'My call sign is 5 characters long
'       Put the length of YOUR callsign in place of the 5 in the line above

'*******************************************************************************
****

        K=J-1                                           'adjust for using in the
Lookup command


'*******************************************************************************
****

        Lookup K,(128,32,130,130,103),Mchar     'Get each character in my call sign
in order
'       Put YOUR callsign in the above line in place of the 128,32...103

'*******************************************************************************
****



        Gosub Didah                                     'Flash the character

        Next J                                  'Next character

        Pause Time2                                     'Pause Time2 milliseconds
before continuing
        Goto Criket                                     'Start all over again

Didah:Numdit= Mchar & %11100000                 'Break off the top 3 bits of the
Morse character
        Numdit=Numdit /32 -1                    'Fix them as 0 to 7 for number of
```

```
elements -1 (adjusted for Lookup command)
        Ditdat=Mchar & %00011111                'Break off the lower 5 bits which
are element data

        For I=0 to Numdit                               'Loop through flash routine
for # of elements times

        Lookup I,(1,2,4,8,16),Mask              'Make masks to mask off the lower 5
bits 1 bit at a time

        Send=Ditlen                                     'Element time starts out as
a Dit
        K=Ditdat & Mask                         'Mask off the proper bit in the
character sequence
        If K =0 Then Flash                      'Is the bit a zero? Then the
Element time is OK...
Got1:   Send=3*Ditlen                           'Bit is a 1 so element is a dash,
element time = 3 * Dit

Flash:Low Led1pin:Low Led2pin                   'Turn ON the LEDs
        Pause Send                                      'Pause for the proper
element time
        High Led1pin:High Led2pin               'Turn OFF the LEDs
        Pause Ntrdit                            'Wait for the proper inter-element
time

        Next I                                  'Next element

        Pause Charlen                           'Pause for the inert-character time
        Return                                  'Done sending character
```