```
1           #picaxe 08m2
2
3
4   '       VER. 1.0        02/26/2017
5
6
7
8   '                       PocketPAL Picaxe (08M2) Project
9   '            Logical Pin = (#)  _____  (#)
10  '                           Vcc     [ |_| ]   Ground
11  '      serial in from PC / (5) [     ] (0)   Attention line from PC
12  ' Piezo/Speaker under TEST (4) [ 08M ] (1)
13  '       Input from counter (3) [     ] (2)   INTERNAL PIEZO for OP comms via
    MORSE
14
15
16  '       A SHORT course in Picaxe Input / Output designators.
17
18  ' The commands in the Picaxe microprocessor refer to the Input/Output pins with
19  ' their LOGICAL Input/Output addresses. The PHYSICAL address of the I/O pin is
20  ' different. Here is a simple table of the pins:
21  '       Logical pin #    Physical pin #
22  '               0                       7
23  '               1                       6
24  '               2                       5
25  '               3                       4
26  '               4                       3
27  '               5                       2
28
29  ' The LOGICAL pin #0 is shared by both the PC download cable AND one of the LEDs.
30  ' For the Cricket to function properly, disconnect the downloading cable after the
31  ' download procedure....now Logical pin #0 is not shared with anything...
32
33
34  '               When fooling around with BYTES, we usually like to start with bit
    0
35  '               What the heck is bit 0?  ...you say!
36
37  '
    ********************************************************************************
    *
38  '
39  '       A short course in Bits and Bytes!
40  '
41  '
    ================================================================================
    =
42  '
43  '       BINARY is a number system where there are only 2 (hence BI) digits 0 and 1
44  '               Computers run on BINARY because there are 2 states for the logic
    contained
45  '               inside them...OFF (0)  and ON (1).
46  '               Groups of digits that are commonly used are:
47  '                               Bit     =  1 digit      or  b
48  '                               Slice   =  2 Bits       or  bb
49  '                               Nibble  =  4 Bits       or  bbbb
50  '                               Byte    =  8 Bits       or  bbbbbbbb
51  '                       and a   Word    = 16 Bits       or  bbbbbbbbbbbbbbbb
52  '
53  '       In little single chip microprocessors, the registers for handling data
    and math are
54  '               typically 8 bits wide so BYTES are the typical form of data they
    deal with.
55  '
56  '                       A BYTE is 8 bits usually written in a left to right in
    desending sequence
57  '                               of bits where each bit represents an increasing
    power of 2
58
59  '               Bit #        7      6     5     4     3     2     1     0
60
```

```
61  '       Place Value             2^7th  2^6th 2^5th 2^4th 2^3rd 2^2nd 2^1st 2^0th
62  '               or              128     64    32    16    8     4     2     1
63
64  '               So           1      1     1     0     0     0     0     1       =
    11100001 (binary)
65  '           Represents    128  +  64 +  32 +  0  +  0  +  0  +  0  +  1         =
       225    (decimal)
66
67
68  ' You already know all this; but, in a different number BASE.....decimal
69  '                               Decimal, the numbers that we use, are digits
    (0-9) written from
70  '                               right to left where each digit represents an
    increasing power of 10
71
72  '               Digit #        7       6      5      4      3      2      1      0
73
74  '       Place Value             10^7th  10^6th 10^5th 10^4th 10^3rd 10^2nd 10^1st
    10^0th
75  '                             10,000,000                       1,000
                      10
76  '                                       1,000,000         10,000
    100                 1
77  '                                                 100,000
78
79  '               So             0       7      0      4      0      0      0      0
80  '           Represents       0+ 7,000,000 + 0 + 40,000  + 0    + 0    + 0    + 0
    = 7,040,000 (decimal)
81
82
83  'Now back to the Morse Code
84  '
85  '       Here is a nifty way to encode the Morse Code alphabet into BYTE sized
    bites...
86  '               First, start by changing Dits and Dahs into 0's and 1's
    respectively.
87  '                       So
88  '                               a V would be 0001  (dit dit dit dah)
89  '
90  '               Secondly, arrange the zeros and ones from right to left in the
    order of sending
91  '                       So
92  '                               0001 is now 1000
93
94  '                       add any zeros to make the code always 5 bits wide
95  '                       So
96  '                               1000 now becomes 01000
97  '
98  '               Next, take the number of dits and dahs in the Morse character
    and...
99  '                       write that in binary
100 '                       So
101 '                               4 dits & dahs becomes 100
102
103 '                       and stick it in front of the 5 bit code from the step
    above
104 '                       So
105 '                               100 + 01000 = 10001000
106 '                       which happens to make the encoded characters always 1
    byte long
107 '                       So
108 '                               the V character = 10001000 (binary) = 128 + 16 =
    144 (decimal)
109 '
110 '               Next, when you want to decode the data, you mask off the top 3
    bits.
111 '                       To mask off bits, logical 'AND' the data with a mask byte
    which
112 '                       contains 1's only in the bits you want to have in the
    result.
113 '                       So
```

```
114  '                                10001000 & 11100000 yields 10000000 = 128
     (decimal)
115  '                        And then divide the result by 32 which happens to be the
     value of the
116  '                        place for bit5 which is the lowest 1 bit in the mask
117  '                        Now
118  '                                10000000 /32 (or shifted 5 places) = 00000100 or
     4 (decimal)
119  '                        and there are 4 dits and dahs in V.
120  '
121  '
122  '                Lastly, when you want to check the data for dits and dahs...
123  '                        Starting at the very right, or bit0, construct a mask
     containg ONLY
124  '                        that bit set and logical 'AND' the mask with the data
     byte.
125  '                        If the result is zero, then the bit you are checking is a
     0 or DIT
126  '                        If the result is greater than zero, then it is a 1 or DAH.
127  '                        Continue checking the data bits in the BYTE until you
     have looked at
128  '                        the proper number of bits as coded in the upper 3 bit
     value.
129  '
130  '                        So from the top 3 bits
131  '                                10000000 & 11100000 = 10000000 = 128 (decimal)
     /32 = 4 elements
132  '
133  '                                10001000 & 00000001 = 00000000 = dit = element 1
134  '                                10001000 & 00000010 = 00000000 = dit = element 2
135  '                                10001000 & 00000100 = 00000000 = dit = element 3
136  '                                10001000 & 00001000 = 00001000 = dah = element 4
137  '
138  '       and that is how the Morse characters are coded into data BYTES
139  '
140  '                        Morse Code characters encoded into BYTES
141  '                                                          ELEMENTS
142  '                    Char.  Morse Code          #  CODE           BYTE    DECIMAL
143  '                        A       dit dah.............2   01..........01000010    66
144  '                        B       dah dit dit dit.....4   1000........10000001
     129
145  '                        C       dah dit dah dit.....4   1010........10000101
     133
146  '                        D       dah dit dit.........3   100.........01100001    97
147  '                        E       dit.................1   0...........00100000    32
148  '                        F       dit dit dah dit.....4   0010........10000100
     132
149  '                        G       dah dah dit.........3   110.........01100011    99
150  '                        H       dit dit dit dit.....4   0000........10000000
     128
151  '                        I       dit dit.............2   00..........01000000    64
152  '                        J       dit dah dah dah.....4   0111........10001110
     142
153  '                        K       dah dit dah.........3   101.........01100101
     101
154  '                        L       dit dah dit dit.....4   0100........10000010
     130
155  '                        M       dah dah.............2   11..........01000011    67
156  '                        N       dah dit.............2   10..........01000001    65
157  '                        O       dah dah dah.........3   111.........01100111
     103
158  '                        P       dit dah dah dit.....4   0110........10000110
     134
159  '                        Q       dah dah dit dah.....4   1101........10001011
     139
160  '                        R       dit dah dit.........3   010.........01100010    98
161  '                        S       dit dit dit.........3   000.........01100000    96
162  '                        T       dah.................1   1...........00100001    33
163  '                        U       dit dit dah.........3   001.........01100100
     100
164  '                        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!
'!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!




'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=12        'time length of basic DIT = 12 millisec.
        Symbol Ntrdit=12        'time between element flashes = 12 millisec.
        Symbol Charlen=500      'time between characters = 500 millisec.
        Symbol HZ=125           'frequency of Morse tones

'Define some variables to hold variable data

        Symbol I=b0     'Loop/General variable
        Symbol J=b1     'Loop/General variable
        Symbol K=b2     'Loop/General variable
        Symbol L=b3     'another Loop/General variable

        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 D1=b8    'MSB (Most Signifigant Digit) of Measured/Calculated Frequency
        Symbol D2=b9    '2nd Digit of Measured/Calculated Frequency
        Symbol D3=b10   '3rd Digit of Measured/Calculated Frequency
        Symbol D4=b11   '4th Digit of Measured/Calculated Frequency
        Symbol D5=b12   'MLSB (Least Signifigant Digit) of Measured/Calculated
```

```
    Frequency
217
218         Symbol SEND=w12 'Word length variable holding the count representing
    FREQ/1024
219         Symbol MSUB=w13 'Word length variable holding large number math values
    (>256)
220
221
222 Init:   COUNT 3,1000,SEND       'count the FREQUENCY output from counter chip for
    1 second
223                                 'results = BIG number like 6865
224         MSUB=SEND/250*6         'caclulate .004% * 6 = .024% = 162
225         SEND=SEND+MSUB          'add .024% due to /1024 counter chip 6865+162 =
    7027!
226 '       DEBUG SEND
227
228         IF SEND>0 THEN XTAL     'GOT a frequency reading so crystal plugged
    in...go measure it...!
229 '       NO crystal so issue a "72 HI PAL" message in Morse
230
231 TONES:  FOR D1 = 0 TO 8                           '9 chars in intro message
232         Lookup D1,(163,188,0,128,64,0,134,66,130),Mchar 'get D1(th char in intro
233         IF Mchar>0 THEN SENDIT                  '=0 then just a pause
234         PAUSE CHARLEN                           '               HERE
235         GOTO MOTONE                             'go to the next char
236 SENDIT: gosub didah                             'go announce the char in Morse
237         pause 50                                'just a little time between
    chars...
238
239 MOTONE: NEXT D1                                 'next char in intro
240         PAUSE 500                               'a .5sec pause between intro &
    tones
241         J=3                                     'incremental step in note pitch
    for slide 'whistle'
242
243 '       now generate an asending/desending test tone pattern on the INTERNAL Piezo
244 SLIDE:
245         FOR I=51 TO 126 STEP J                  'asending note pitches
246         SOUND 2,(I,K)                           'sound note on INTERNAL Piezo
247 '       SOUND 4,(I,K)                           'sound note on Piezo/Speaker
    under test
248         NEXT I                                  'next note in sequence
249         FOR I=126 TO 51 STEP -J                 'asending note pitches
250         SOUND 2,(I,K)                           'sound note on INTERNAL Piezo
251 '       SOUND 4,(I,K)                           'sound note on Piezo/Speaker
    under test
252         NEXT I                                  'next note in sequence
253
254 '       now generate an asending/desending test tone pattern on the EXTERNAL
    Piezo under test
255
256         FOR I=51 TO 126 STEP J                  'asending note pitches
257 '       SOUND 2,(I,K)                           'sound note on INTERNAL Piezo
258         SOUND 4,(I,K)                           'sound note on Piezo/Speaker
    under test
259         NEXT I                                  'next note in sequence
260         FOR I=126 TO 51 STEP -J                 'asending note pitches
261 '       SOUND 2,(I,K)                           'sound note on INTERNAL Piezo
262         SOUND 4,(I,K)                           'sound note on Piezo/Speaker
    under test
263         NEXT I                                  'next note in sequence
264
265         K=K+1                                   'lengthen the notes by one unit
266 '       go do it again...and again...and again...until the lower right power
    button is released
267         GOTO SLIDE
268
269 XTAL:   MSUB=SEND/10000          'calculate MSB of crystal frequency
270         D1=MSUB                 'DIGIT 1 = Ten Thousands KHZ
271         MSUB=D1*10000           're-CALC *actual* Ten Thousands KHZ
272         SEND=SEND-MSUB          'remove that value from freq. in order to calc
```

```
    next digit
273         D2=SEND/1000            'calculate DIGIT 2 = Thousands KHZ
274         MSUB=D2*1000            're-CALC *actual* Thousands KHZ
275         SEND=SEND-MSUB          'remove that value from freq. in order to calc
    next digit
276         D3=SEND/100             'calculate DIGIT 3 = Hundreds KHZ
277         MSUB=D3*100             're-CALC *actual* Hundreds KHZ
278         SEND=SEND-MSUB          'remove that value from freq. in order to calc
    next digit
279         D4=SEND/10              'calculate DIGIT 4 = Tens KHZ
280         MSUB=D4*10              're-CALC *actual* Tens KHZ
281         D5=SEND-MSUB            'and DIGIT 5 = Ones...the remainder...
282
283 '       DEBUG SEND              'DEBUG all the variables WHILE developing the
    program
284
285         IF D1=0 THEN NOzero             'SURPRESS THE LEADING ZERO if there is
    one...
286
287         Lookup D1,(191,190,188,184,176,160,161,163,167,175),Mchar 'Morse code
    values for chars 0-9
288         gosub didah
289         pause 50
290 NOzero:
291
292         Lookup D2,(191,190,188,184,176,160,161,163,167,175),Mchar 'Morse code
    values for chars 0-9
293         gosub didah
294         pause 50
295         Lookup D3,(191,190,188,184,176,160,161,163,167,175),Mchar 'Morse code
    values for chars 0-9
296         gosub didah
297         pause 50
298         Lookup D4,(191,190,188,184,176,160,161,163,167,175),Mchar 'Morse code
    values for chars 0-9
299         gosub didah
300         pause 50
301         Lookup D5,(191,190,188,184,176,160,161,163,167,175),Mchar 'Morse code
    values for chars 0-9
302         gosub didah
303
304         pause 2000                              'pause 2 sec before doing it ALL
    again
305         goto init
306
307 Didah:  Numdit= Mchar & %11100000              'Break off the top 3 bits of the
    Morse character
308         Numdit=Numdit /32 -1                    'Fix them as 0 to 7 for number of
    elements -1 (adjusted for Lookup command)
309         Ditdat=Mchar & %00011111                'Break off the lower 5 bits which
    are element data
310
311         For I=0 to Numdit                       'Loop through flash routine for #
    of elements times
312
313         Lookup I,(1,2,4,8,16),Mask              'Make masks to mask off the lower
    5 bits 1 bit at a time
314
315         SEND=Ditlen                             'Element time starts out as a Dit
316         K=Ditdat & Mask                         'Mask off the proper bit in the
    character sequence
317         If K=0 Then FIST                        'Is the bit a zero? Then the
    Element time is OK...
318 Got1:   SEND=3*Ditlen                           'Bit is a 1 so element is a dash,
    element time = 3 * Dit
319
320 FIST:   Sound 2,(HZ,SEND)                       'Sound Piezo for the proper
    element time            'Turn OFF the LEDs
321         Pause Ntrdit                            'Wait for the proper
    inter-element time
322         Next I                                  'Next element
```

```
323
324             Pause Charlen                           'Pause for the inert-character
    time
325             Return                                  'Done sending character
326
327
328
329
330
331
332
333
```