

FT-817 Roger Beep & Beacon

by Iain Crawford, VK5ZD

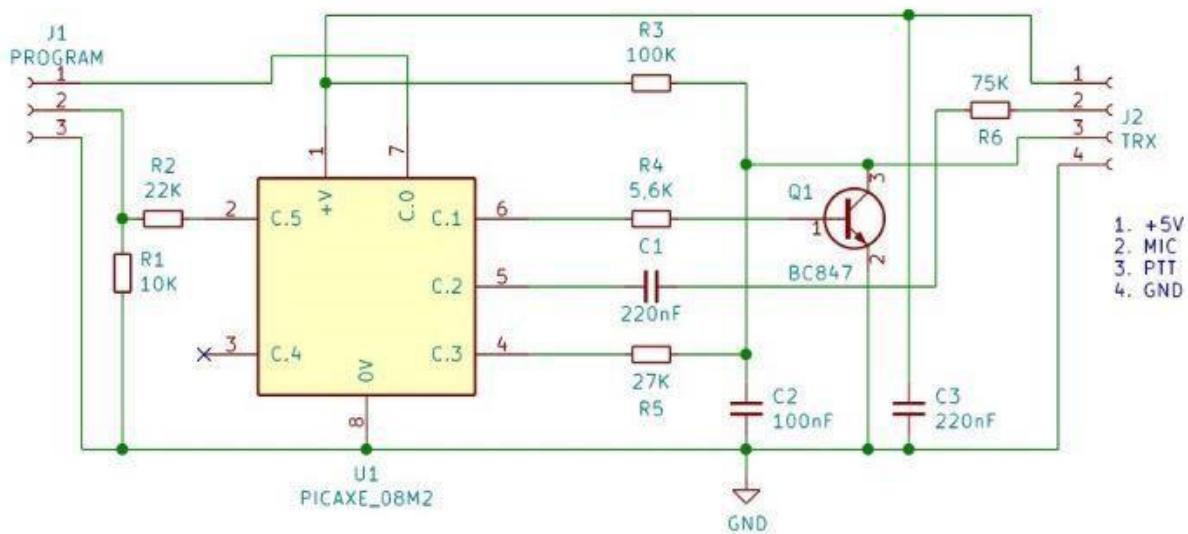
Iain a ajouté un 'roger beep' en utilisant un Pixaxe-08M2 dans le micro MH-31 du FT-817s.

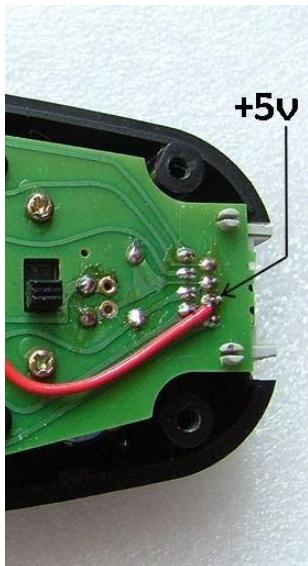
Très utile en Hyper, on demande souvent de transmettre une balise pour affiner le pointage.

Pour cela Iain était obligé de passer en FM et maintenir le PTT ou passer en CW pour envoyer une balise.

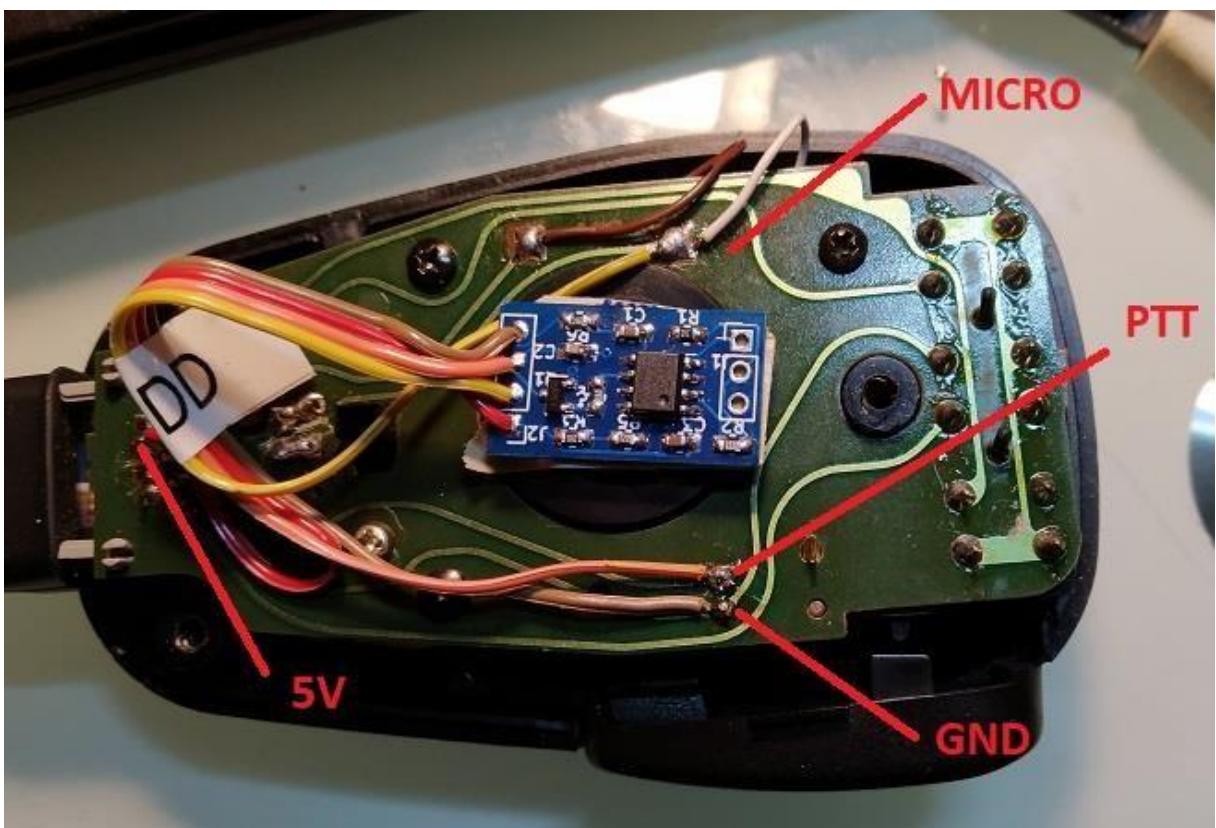
Il a modifié le code du roger beep pour rajouter une option balise. La balise s'active en appuyant 3 fois sur le PTT. Une fois activée, une balise avec indicatif est envoyée. Elle peut être arrêtée en appuyant sur le PTT. Il y'a aussi un «timeout» qui désactive la balise apres 5 minutes.

Le schéma de la carte intégré dans le MH-31:





J'ai utilisé l'interrupteur pour activer ou désactiver mon rodger beep.

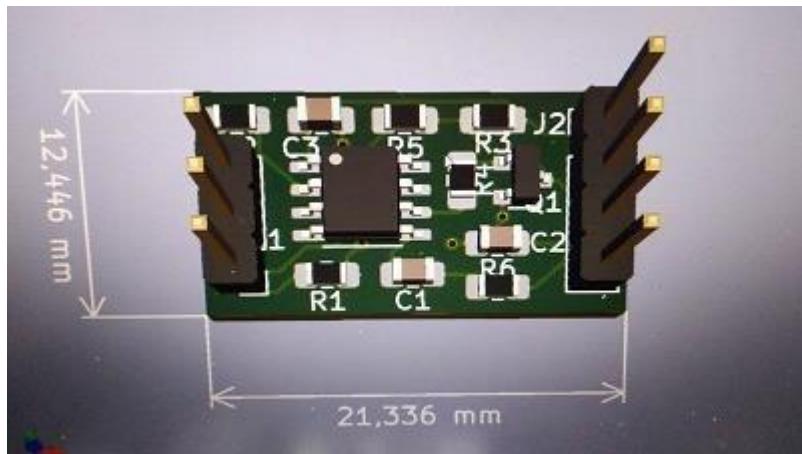


Maciek F4VSQ a développé une carte miniature qui s'insère parfaitement dans le microphone.

Morse code vers code Arduino EEPROM

Convertir

Jean Louis F4CLB a fait un petit soft pour le codage du message. [MorseEepromPicaxe.zip](#) Le programme a modifier est [ici](#)



Le logiciel de programmation <https://picaxe.com/software/picaxe/picaxe-editor-6/>

PICAXE CODE

```
#picaxe 08m2

' pins used
symbol PTT_OUT      = 1    symbol
AUDIO_OUT        = 2    symbol
PTT_PIN          = pin3  symbol
PTT_IN           = 3    symbol

' constants
symbol AUDIO_NOTE      = 118      ' set roger beep audio frequency (1-127)  symbol
AUDIO_DURATION     = 6       ' ~10ms units symbol PRE_BEEP_DELAY   = 100      ' ms
before beep symbol POST_BEEP_DELAY = 50      ' ms after beep symbol PULSES_FOR_IDENT
= 3                 ' pulses to activate ident symbol COUNT_PERIOD   = 800      ' ms to
count pulses symbol IDENT_TIMEOUT   = 300      ' seconds before beacon timeout symbol
ID_INTERVAL        = 30      ' seconds between callsign transmission symbol MORSE_NOTE
= 108               ' sets the beacon audio frequency (1 - 127) symbol DIT_TIME     = 40
' ms symbol DAH_TIME        = DIT_TIME * 3 symbol WORD_SPACE     = DIT_TIME * 5

' variables
symbol PulseCount      = w3      ' count of PTT presses;      <shared variable> symbol
MorseCode           = b0      ' encoded morse character; must be b0
```

```

symbol LastBit          = bit0           ' must be bit0 symbol MorseTime      = b2           '
tone duration;           <shared variable> symbol TimeoutCheck    = b2           ' check
if time to send ID;    <shared variable> symbol ReadPointer
= b3
symbol TimeoutCounter   = w3           ' <shared variable>

eeprom (0, 9, 2, 0, 24, 13, 32, 19, 9, 0, 255)      ' spc DE spc VK5ZD spc end
start:
  ' configure I/O pins
  output PTT_OUT,
  AUDIO_OUT input PTT_IN
  ' go to receive mode low
  PTT_OUT
main:
do
  ' wait until PTT is
  pressed do until PTT_PIN
  = 0 loop

  ' count pulses on PTT
  count PTT_IN, COUNT_PERIOD, PulseCount
  ' (a) has the PTT been released and
  ' (b) did we count enough pulses
  if PTT_PIN <> 0 AND PulseCount >= PULSES_FOR_IDENT then
    ' zero the timeout counter
    TimeoutCounter = 0 '
    go to transmit mode
    high PTT_OUT
    ' send the CW
    identification gosub
    SendString ' go to receive
    mode low PTT_OUT
    ' send the tone until either
    ' (a) the PTT is pressed again or
    ' (b) the timeout expires
    do until PTT_PIN = 0 OR TimeoutCounter = IDENT_TIMEOUT
      ' go to transmit mode
      high PTT_OUT
      ' send ~1 second of tone
      sound AUDIO_OUT, (MORSE_NOTE, 100)      ' 100 ~ 1 second
      ' increment the timeout counter
      inc TimeoutCounter
      ' check if we're due to send the CW identification again
      TimeoutCheck = TimeoutCounter // ID_INTERVAL
      if TimeoutCheck = 0 then gosub SendString
      ' go to receive mode
  low PTT_OUT loop
  ' the loop has finished so send the final ID ' go to transmit mode high
  PTT_OUT
  ' send the CW
  identification gosub
  SendString ' go to receive
  mode low PTT_OUT
end if

' wait until PTT is released do while PTT_PIN = 0
Loop

  ' this bit creates the 'roger beep'
  ' go to transmit
  mode high PTT_OUT '
  pause pause
  PRE_BEEP_DELAY '
  create the beep
  sound AUDIO_OUT, (AUDIO_NOTE, AUDIO_DURATION) ' go to receive mode low
  PTT_OUT
  ' pause to debounce PTT switch pause
  POST_BEEP_DELAY
loop ' forever end

SendString:

```

```

' send the morse identification string. characters are read '
from eeprom until 255 is found.

' set the pointer to the first character
ReadPointer = 0
do
    ' get the next character
    read ReadPointer, MorseCode
    ' send the morse character
    gosub SendCharacter
    ' set the pointer to the next character
    inc ReadPointer
loop until MorseCode = 255
return

SendCharacter:
if MorseCode = 255 then
    ' do nothing elseif
MorseCode = 0 then pause
WORD_SPACE
else do if LastBit = 0
    then
        MorseTime = DIT_TIME / 10
    else
        MorseTime = DAH_TIME / 10
    end if
    sound AUDIO_OUT, (MORSE_NOTE, MorseTime)
    pause DIT_TIME           ' delay between bits MorseCode =
    MorseCode / 2            ' Shift right
loop until MorseCode = 1
pause DAH_TIME           ' delay between letters
end if return

' Morse Character Code Table

' A= 6   B=17   C=21   D= 9   E= 2   F=20   G=11   H=16   I= 4   J=30   ' K=13   L=18   M=
7   N= 5   O=15   P=22   Q=27   R=10   S= 8   T= 3   ' U=12   V=24   W=14   X=25   Y=29
Z=19
' 1=62   2=60   3=56   4=48   5=32   6=33   7=35   8=39   9=47   0=63

' Space          = 0
' Full Stop     . =106      Colon       : = 71 ' Comma          , =115      Semicolon
; = 85 ' Question Mark ? = 76      Equal Sign   = = 49 ' Apostrophe   ' = 94      Plus
+ = 42

' Exclamation Mark ! =117      Hyphen/Minus - = 97 ' Foward Slach   / = 41      Underscore
=108 ' Opening Bracket ( = 45      Quotation Mark " = 82 ' Closing Bracket ) =109      Dollar
Sign   $ =200 ' Ampersand   & = 34      AT Sign     @ = 86

```

Iain
VK5ZD