APPLICATION OPERATION

Here is a circuit idea that can be implemented by using 8-pin PICmicro™ microcontroller.

A Morse Keyer is an aid to wireless operators and amateur radio operators for sending clean Morse Code while reducing operator fatigue, even at a very high speed. The Morse Keyer consists of a dual-contact keyer paddle. One contact produces dots, while the other produces dashes. Dots consist of a short beep, followed by an equally short period of silence. Dashes consist of a long beep that is three times the length of a dot, followed by a silent period which is equal to one short beep or dot.

Pressing the ‘Dot’ paddle produces dots; pressing the ‘Dash’ paddle produces dashes. Pressing both paddles simultaneously produces an alternate pattern of dots and dashes. An alphabet of Morse characters are a combination of dots and dashes. Characters or words, are identified by timing gaps between groups of Morse elements.

A Morse Keyer based on an 8-pin PICmicro™ microcontroller has the facility for sending Morse code with the following additional features:

1. Automatic insertion of letter and word spacing, thereby reducing the chances of error.
2. Facility for monitoring the Morse code with built-in sounder.
3. Facility for changing the speed of sending.
4. Facility for changing the tone frequency output.
5. Iambic operation is possible.

Apart from the keyer paddle, there are two push switches for setting the transmission speed and adjusting the tone output.

To change the speed, press the speed button. Continuous dots (short beeps) will be heard on the buzzer. While pressing the speed button, press the dot paddle to increase the speed; press the dash paddle to decrease the speed.

Microchip Technology Incorporated, has been granted a nonexclusive, worldwide license to reproduce, publish and distribute all submitted materials, in either original or edited form. The author has affirmed that this work is an original, unpublished work and that he/she owns all rights to such work. All property rights, such as patents, copyrights and trademarks remain with author.
SOFTWARE LISTING

The codes are not fully written due to time limit. However, the difficult portion is the subroutine for generating the audio frequency and the timing routine.

Timer and Tones Routine

START    BSF   GPIO,TONE_OUT; Tone output pin is set
          BSF   GPIO,KEYOUT
          MOVF  GPIO,0
          ;INITIALISE TIMER HERE
          MOVWF  STATUS
          DECFSZ TONE_FREQ
          GOTO ON ;Loop here ON until one time period of tone freq is over
          BCF   GPIO,TIME_OUT ;Reset tone output
          BCF   GPIO,KEYOUT
          MOVF TONE_FREQ,0
          MOVWF  TONE_REG
          OFF    MOVF  GPIO,0
          MOVWF  STATUS
          DECFSZ TONE_FREQ
          GOTO   OFF

;TIMER CHECK ROUTINE
          MOVLW  0x27 ;COUNT VALUE FOR 10 MILLISECONDS TIMER
          SUB     TIMER0.0
          BTFSs  STATUS, ZERO
          BTFSS  STATUS,BORROW; Check for 10 milliseconds time
          GOTO   START
          DEC    TONE_TIME
          BTFSS  STATUS,ZERO
          GOTO   START
          SPACE  MOVF  SPACE_TIME
          CLR    TIMER0
          BCF    GPIO,TONE_OUT
          LOOP   MOVLW  0x27
          SUB     TIMER0
          BTFSS  STATUS,ZERO
          BTFSS  STATUS,BORROW
          GOTO   LOOP
          DEC    SPACE_TIME
          BTFSS  STATUS,ZERO
          GOTO LOOP