```
// Author Glen Popiel, KW5GP
/*

 This program is free software: you can redistribute it and/or modify
 it under the terms of the version 3 GNU General Public License as
 published by the Free Software Foundation.

 This program is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 GNU General Public License for more details.

 You should have received a copy of the GNU General Public License
 along with this program.  If not, see <http://www.gnu.org/licenses/>.

 */

// Uses Morse Library - Author Erik Linder - errors corrected by Glen
Popiel KW5GP

// Beacon mode repeats message at regular intervals
// Foxhunt Mode repeats message for repeat time at interval time
(seconds)

#include <Morse.h>

#define blue 7    // Blue Led Pin
#define green 8   // Green LED Pin
#define red 9     // Red Led Pin

const int mode = 0;   // Mode 0 = Beacon 1 = Foxhunt
const char beacon_call[] = "DE KW5GP Beacon";  // Message to Send
const int key_speed = 20;  // CW send speed
int interval = 10;     // Timer interval in seconds
int repeat = 1; // Repeat message time in seconds - must be a minimum of
1
const int beep_pin = 11; // Pin for CW tone
int key_pin = 12;  // Pin for PTT/Rig Key
long end_time;  // Foxhunt timing
char c;  // Character to send
bool first_pass = true;  // Flag to tell us this is the first time
through
unsigned long offset = 0L;  // Timer value
Morse morse(beep_pin, key_speed, 1); // Set up the Morse Library for use
int msg_length;  // variable to hold the size of the message text array

// Timer function - returns the time in seconds since last Timer Reset
unsigned long Timer()
{
  return (millis()- offset)/1000;  // return time in seconds
}

// Timer Reset function
void TimerReset(unsigned long val = 0L)
```

```
{
  offset = millis() - val;
}

void setup()
{
  msg_length =(sizeof(beacon_call))-1;  // Calculate the size of the
message text array
  pinMode(key_pin, OUTPUT);           // set PTT/Key pin to output
  pinMode(red, OUTPUT);  // Set up the LED pins for output
  pinMode(green, OUTPUT);
  pinMode(blue, OUTPUT);
  digitalWrite(key_pin,LOW); // Turn off the PTT/Key relay
  ledOff();  // Turn off the LED
  delay(5000);  // Wait 5 seconds
  TimerReset();  // Reset the Timer to zero
  ledGreen();  // Turn on the Green LED to indicate Beacon/Keyer ready
}  // End Setup Loop

void loop()  // Main Program Loop
{

  if (Timer() > interval | first_pass)  // Send if the Timer has expired
or if this is the first time through
  {

    first_pass = false;  // Set the flag to off

    if (mode == 0)  // Set the Key mode and LED for Beacon or Foxhunt
    {
      // Set Beacon Mode
      Morse(key_pin, key_speed, 0);    // Set up to key the Relay
      ledBlue();  // Turn on the Blue LED to indicate Beacon Message
Transmitting
    } else {
      // Set Foxhunt Mode
      Morse(beep_pin, key_speed, 1);  // Set up to send modulated CW
      ledRed();  // Turn on the Red LED to indicate Foxhunt Message
Transmitting
      digitalWrite(key_pin, HIGH);  // Key the PTT
    }

    end_time = Timer() + repeat;  // If we're in Foxhunt mode, repeat the
message until the repeat time has expired

    while(Timer() < end_time)  // Check to make sure repeat timer has not
expired (Foxhunt Mode)
    {

      for (int x = 0; x < msg_length; x++)  // Send the message in the
beacon_call character array one character at a time
      {
        c = beacon_call[x];  // Get the next letter to send
        morse.send(c);  // Send it in CW
```

```
      }
      if (mode == 1)  // Send a space if we're in Foxhunt mode to
separate messages
      {
        morse.send(char(32));
      } else {
        end_time = Timer()-1;
      }
    }
    digitalWrite(key_pin, LOW);  // Make sure the PTT is off
    ledGreen();  // Turn the LED Green to indicate Ready condition
    TimerReset();  // Reset the timer to restart the Interval between
messages
  }
}  // End of Loop


// LED Off function
void ledOff()
{
  digitalWrite(red, HIGH);  // Set all RBG LED pins High (Off)
  digitalWrite(green, HIGH);
  digitalWrite(blue, HIGH);
}

// RED LED ON function
void ledRed()
{
  digitalWrite(red, LOW);  // Turn on the RGB Red LED On
  digitalWrite(green, HIGH);
  digitalWrite(blue, HIGH);
}

// Green LED ON function
void ledGreen()
{
  digitalWrite(red, HIGH);
  digitalWrite(green, LOW);  // Turn on the RGB Green LED On
  digitalWrite(blue, HIGH);
}

// Blue LED ON function
void ledBlue()
{
  digitalWrite(red, HIGH);
  digitalWrite(green, HIGH);
  digitalWrite(blue, LOW);  // Turn on the RGB Blue LED On
}
```