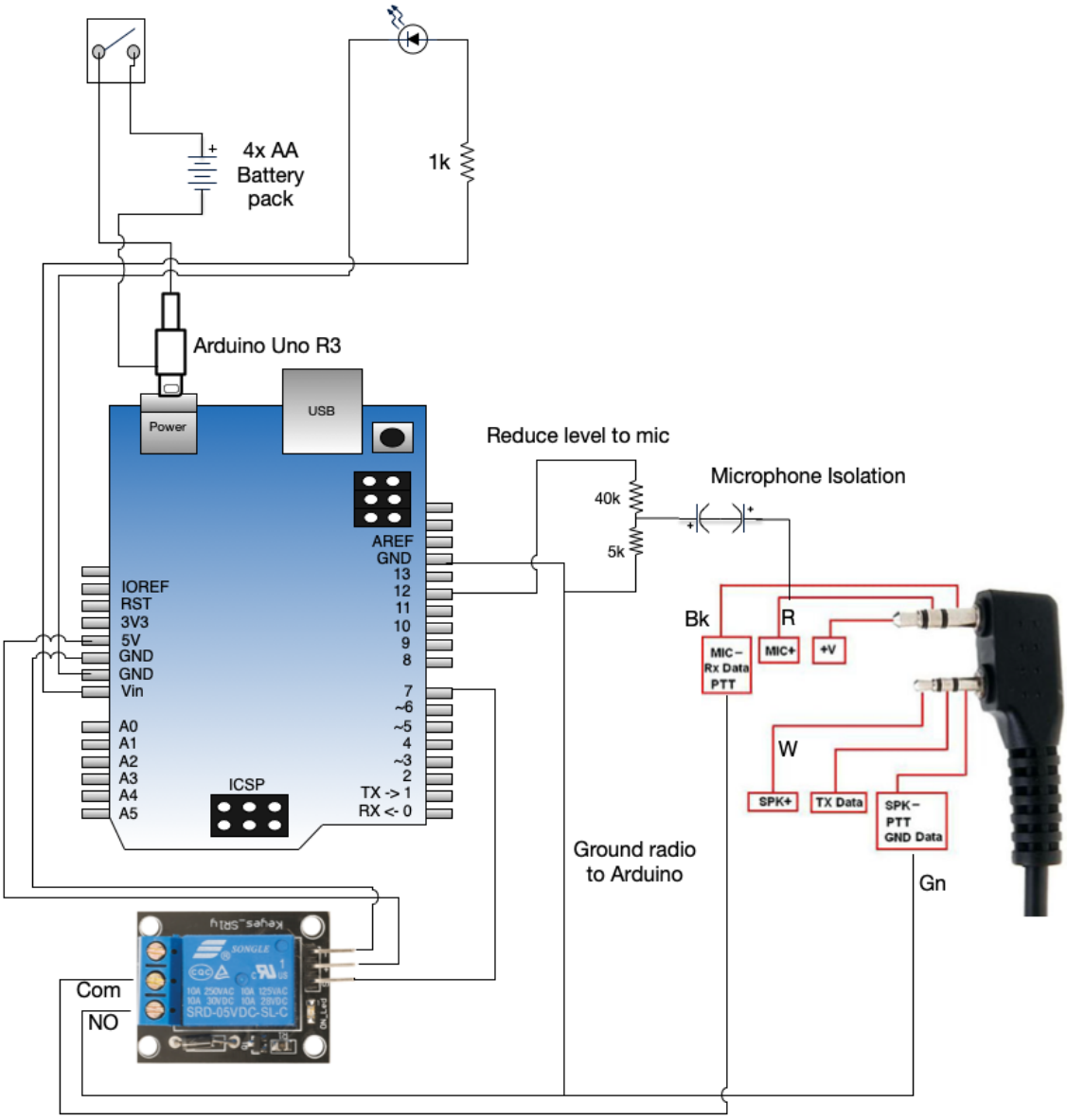

Notes:

- Used 4x AA Batteries instead of a 9V for longevity
- Used a 1k ohm resister for the LED to minimize battery drain
- Used two 10 uF capacitors back to back for a non-polarized isolation of the radio to the Arduino - helped with sound quality and reliability. Also reduced power to the microphone to reduce overload.
- It is important to ground the radio to the Arduino for relay and microphone reliability.
- Used a standard Arduino relay as it has the diode and pull -down resistor built-ion
- Special Thanks to Bob McCracken N4JGO for the circuit upgrades, programming ideas and troubleshooting

Norm
KN4YFI


CORDURA
FABRIC

```
/*By Nelson Farrier
  Key up radio and send tone
  Keys up Baofeng UV-3R radio by turning on and off the relay, then ID's and and
sends a 5 second tone.
*/

// Pin 7 is connected to a relay.
// Pin 12 is connected to a tone circuit.
// On-minute between cycles.

//   modified from: Mike Myers (http://mikemyers.me)  @netnutmike
//   Let's Make It Episode 6 (http://tech-zen.tv/index.
php/shows/let-s-make-it/episodes/59-sensor-fun-with-arduino-1-massive-failure-but-
4-successes-let-s-make-it-episode-6)
//   define the morse code for the alphabet and numbers

char* letters[] = {
  ".-",     // A
  "-...",   // B
  "-.-.",   // C
  "-..",    // D
  ".",      // E
  "..-.",   // F
  "--.",    // G
  "....",   // H
  "..",     // I
  ".---",   // J
  "-.-",    // K
  ".-..",   // L
  "--",     // M
  "-.",     // N
  "---",    // O
  ".--.",   // P
  "--.-",   // Q
  ".-.",    // R
  "...",    // S
  "-",      // T
  "..-",    // U
  "...-",   // V
  ".--",    // W
  "-..-",   // X
  "-.--",   // Y
  "--.."    // Z
};
```

```
char* numbers[] = {
  "-----",   // 0
  ".----",   // 1
  "..---",   // 2
  "...--",   // 3
  "....-",   // 4
  ".....",   // 5
  "-....",   // 6
  "--...",   // 7
  "---..",   // 8
  "----."    // 9   --- end of 1st segment of borrowed code from Mike Myers
};

int relay = 7;
int TonePin = 12;
int frequency = 1000; // frequency of tone

int dotDelay = 70;      // duration of the dot in morse code, this is also the
time between the dots and dashes
int charDelay = 500;    // duration of the wait between letters for Farsnworth
method
int wordDelay = 1100;   // duration of the wait between words for Farsnworth
method
int cycleDelay = 15000;  // Use several time because the largest value is 16383

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(relay, OUTPUT);
  pinMode(TonePin, OUTPUT);
  delay(2000);  // initial delay after powering on
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(relay, HIGH);   // turn the relay on (HIGH is the voltage level)
  delay(1000);                 // wait for a second

  SendText("KN4YFI FOX");

  delay(1000);                 // wait for a second

  tone(TonePin, frequency);    // send 7 sec tone
  delay(7000);
```

```
  noTone(TonePin);

  tone(TonePin, frequency + 100);  // send 7 sec tone
  delay(7000);
  noTone(TonePin);

  tone(TonePin, frequency);    // send 7 sec tone
  delay(7000);
  noTone(TonePin);

  delay(1000);                 // 1 seconds transmit w/o tone

  SendText("KN4YFI FOX");

  delay(1000);                 // wait for a second


  digitalWrite(relay, LOW);    // turn the relay off by making the voltage LOW

  delay(cycleDelay);           // wait for cycle time
  delay(cycleDelay);           // wait for cycle time
  delay(cycleDelay);           // wait for cycle time
  delay(cycleDelay);           // wait for cycle time
}

//=============================================================
//
// modified from: Mike Myers (http://mikemyers.me)  @netnutmike
// Function: morseCodeSequence
//
// Input: Character Array of Dots and Dashes to be sent
//
// Description:
//      This function takes as input an array or "." and "-" and
//      calls dotOrDash for each item in the array.
//
//      At the end of the sequence, there is a delay of 3 times
//      the dot duration.
//=============================================================

void morseCodeSequence(char* sequence)
{
  int i = 0;

  // Loop for each element in the array
```

```
  while (sequence[i] != NULL)
  {
    dotOrDash(sequence[i]);     // Send out the dot or dash
    i++;                        // Increment to the next element in the array
  }
  delay(charDelay);            // gap between letters
}
//=====================================================================
//
// Function: SendText
//
// Input: Character Array of text in English
//
// Description:
//      This function takes text as input and sends Morse code for each letter.
//      There then is a pause after each letter.
//
//=====================================================================

void SendText(char* MorseCodeLetters)
{
  int i = 0;
  char ch;

  // Loop for each element in the array
  while (MorseCodeLetters[i] != NULL)
  {
    ch = MorseCodeLetters[i];
    // Is it lowercase letter?
    if (ch >= 'a' && ch <= 'z')
    {
      morseCodeSequence(letters[ch - 'a']);
    }
    else if (ch >= 'A' && ch <= 'Z')    // Uppercase Letter
    {
      morseCodeSequence(letters[ch - 'A']);
    }
    else if (ch >= '0' && ch <= '9')    // Number
    {
      morseCodeSequence(numbers[ch - '0']);
    }
    else if (ch == ' ')                 // Space (wait for 4 times dotDelay
    {
      delay(wordDelay);      // gap between words
    }
```

```
    else {
    }
    i++;         // Increment to the next element in the array
  }
  delay(charDelay);            // gap between letters
}

//=====================================================================
//
// Function: dorOrDash
//
// modified from: Mike Myers (http://mikemyers.me)  @netnutmike
// Input: Character that should be either a dot or a dash
//
// Description:
//      This function first turns on the output then looks to see
//      if the character is a "." and if so delays the dotDelay.
//
//      If the character is not a "." then the routine assumes it
//      is a "-" and keep the output high for 3 times the length of
//      dotDelay.  This could be improved by making sure the
//      character is a "-" but for most cases it would not matter.
//
//      After the delay time the pin is taken low turning off the
//      tone.
//
//      Then it delays for one dotDelay time so the dots and dashes
//      do not run together.
//=====================================================================

void dotOrDash(char dotOrDash)
{
  tone(TonePin, frequency);
  if (dotOrDash == '.')
  {
    delay(dotDelay);
  }
  else // must be a -
  {
    delay(dotDelay * 3);
  }
  noTone(TonePin);
  delay(dotDelay); // gap between flashes
}
```