Buzzerd Code:

```
/*  RickRollBoxCode

    AUTHOR: Samantha Lagestee
    Copyright 2017 samilagestee at gmail dot com

    This program is free software: you can redistribute it and/or
    modify it under the terms of the GNU General Public License as
    published by the Free Software Foundation, either version 3 of
    the License, or (at your option) any later version.

    DISCLAIMER: The song "Never Gonna Give You Up" by Rick Astley
    is not the creative property of the author. This code simply
    plays a Piezo buzzer rendition of the song.
*/

#define  a3f    208     // 208 Hz
#define  b3f    233     // 233 Hz
#define  b3     247     // 247 Hz
#define  c4     261     // 261 Hz MIDDLE C
#define  c4s    277     // 277 Hz
#define  e4f    311     // 311 Hz
#define  f4     349     // 349 Hz
#define  a4f    415     // 415 Hz
#define  b4f    466     // 466 Hz
#define  b4     493     // 493 Hz
#define  c5     523     // 523 Hz
#define  c5s    554     // 554 Hz
#define  e5f    622     // 622 Hz
#define  f5     698     // 698 Hz
#define  f5s    740     // 740 Hz
#define  a5f    831     // 831 Hz

#define rest    -1

int piezo = 7;
int led = 9;
int button = 2;
int sensor = A0;

volatile int beatlength = 100; // determines tempo
float beatseparationconstant = 0.3;

int threshold;

int a; // part index
int b; // song index
int c; // lyric index
```

```
boolean flag;

// Parts 1 and 2 (Intro)

int song1_intro_melody[] =
{c5s, e5f, e5f, f5, a5f, f5s, f5, e5f, c5s, e5f, rest, a4f, a4f};

int song1_intro_rhythmn[] =
{6, 10, 6, 6, 1, 1, 1, 1, 6, 10, 4, 2, 10};

// Parts 3 or 5 (Verse 1)

int song1_verse1_melody[] =
{ rest, c4s, c4s, c4s, c4s, e4f, rest, c4, b3f, a3f,
  rest, b3f, b3f, c4, c4s, a3f, a4f, a4f, e4f,
  rest, b3f, b3f, c4, c4s, b3f, c4s, e4f, rest, c4, b3f, b3f, a3f,
  rest, b3f, b3f, c4, c4s, a3f, a3f, e4f, e4f, e4f, f4, e4f,
  c4s, e4f, f4, c4s, e4f, e4f, e4f, f4, e4f, a3f,
  rest, b3f, c4, c4s, a3f, rest, e4f, f4, e4f
};

int song1_verse1_rhythmn[] =
{ 2, 1, 1, 1, 1, 2, 1, 1, 1, 5,
  1, 1, 1, 1, 3, 1, 2, 1, 5,
  1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 3,
  1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 4,
  5, 1, 1, 1, 1, 1, 1, 1, 2, 2,
  2, 1, 1, 1, 3, 1, 1, 1, 3
};

char* lyrics_verse1[] =
{ "We're ", "no ", "strangers ", "", "to ", "love ", "", "\r\n",
  "You ", "know ", "the ", "rules ", "and ", "so ", "do ", "I\r\n",
  "A ", "full ", "commitment's ", "", "", "what ", "I'm ", "thinking ", "", "of",
"\r\n",
  "You ", "wouldn't ", "", "get ", "this ", "from ", "any ", "", "other ", "",
"guy\r\n",
  "I ", "just ", "wanna ", "", "tell ", "you ", "how ", "I'm ", "feeling", "\r\n",
  "Gotta ", "", "make ", "you ", "understand", "", "\r\n"
};

// Parts 4 or 6 (Chorus)

int song1_chorus_melody[] =
{ b4f, b4f, a4f, a4f,
  f5, f5, e5f, b4f, b4f, a4f, a4f, e5f, e5f, c5s, c5, b4f,
  c5s, c5s, c5s, c5s,
  c5s, e5f, c5, b4f, a4f, a4f, a4f, e5f, c5s,
  b4f, b4f, a4f, a4f,
  f5, f5, e5f, b4f, b4f, a4f, a4f, a5f, c5, c5s, c5, b4f,
```

```
  c5s, c5s, c5s, c5s,
  c5s, e5f, c5, b4f, a4f, rest, a4f, e5f, c5s, rest
};

int song1_chorus_rhythmn[] =
{ 1, 1, 1, 1,
  3, 3, 6, 1, 1, 1, 1, 3, 3, 3, 1, 2,
  1, 1, 1, 1,
  3, 3, 3, 1, 2, 2, 2, 4, 8,
  1, 1, 1, 1,
  3, 3, 6, 1, 1, 1, 1, 3, 3, 3, 1, 2,
  1, 1, 1, 1,
  3, 3, 3, 1, 2, 2, 2, 4, 8, 4
};

char* lyrics_chorus[] =
{ "Never ", "", "gonna ", "", "give ", "you ", "up\r\n",
  "Never ", "", "gonna ", "", "let ", "you ", "down", "", "\r\n",
  "Never ", "", "gonna ", "", "run ", "around ", "", "", "", "and ", "desert ", "",
"you\r\n",
  "Never ", "", "gonna ", "", "make ", "you ", "cry\r\n",
  "Never ", "", "gonna ", "", "say ", "goodbye ", "", "", "\r\n",
  "Never ", "", "gonna ", "", "tell ", "a ", "lie ", "", "", "and ", "hurt ",
"you\r\n"
};

void setup()
{
  pinMode(piezo, OUTPUT);
  pinMode(led, OUTPUT);
  pinMode(button, INPUT_PULLUP);
  pinMode(sensor, INPUT);
  attachInterrupt(digitalPinToInterrupt(button), getFaster, FALLING);
  digitalWrite(led, LOW);
  Serial.begin(9600);
  flag = false;
  a = 4;
  b = 0;
  c = 0;
  threshold = analogRead(sensor) + 250;
}

void loop()
{
  int sensorreading = analogRead(sensor);
  if (sensorreading < threshold) { // if bright, play
    flag = true;
  }
  else if (sensorreading > threshold) { // if dark, pause
    flag = false;
```

```
  }

  // play next step in song
  if (flag == true) {
    play();
  }
}

void play() {
  int notelength;
  if (a == 1 || a == 2) {
    // intro
    notelength = beatlength * song1_intro_rhythmn[b];
    if (song1_intro_melody[b] > 0) {
      digitalWrite(led, HIGH);
      tone(piezo, song1_intro_melody[b], notelength);
    }
    b++;
    if (b >= sizeof(song1_intro_melody) / sizeof(int)) {
      a++;
      b = 0;
      c = 0;
    }
  }
  else if (a == 3 || a == 5) {
    // verse
    notelength = beatlength * 2 * song1_verse1_rhythmn[b];
    if (song1_verse1_melody[b] > 0) {
      digitalWrite(led, HIGH);
      Serial.print(lyrics_verse1[c]);
      tone(piezo, song1_verse1_melody[b], notelength);
      c++;
    }
    b++;
    if (b >= sizeof(song1_verse1_melody) / sizeof(int)) {
      a++;
      b = 0;
      c = 0;
    }
  }
  else if (a == 4 || a == 6) {
    // chorus
    notelength = beatlength * song1_chorus_rhythmn[b];
    if (song1_chorus_melody[b] > 0) {
      digitalWrite(led, HIGH);
      Serial.print(lyrics_chorus[c]);
      tone(piezo, song1_chorus_melody[b], notelength);
      c++;
    }
    b++;
```

```
    if (b >= sizeof(song1_chorus_melody) / sizeof(int)) {
      Serial.println("");
      a++;
      b = 0;
      c = 0;
    }
  }
  delay(notelength);
  noTone(piezo);
  digitalWrite(led, LOW);
  delay(notelength * beatseparationconstant);
  if (a == 7) { // loop back around to beginning of song
    a = 1;
  }
}

void getFaster() { // decrease beat length in order to increase tempo
  beatlength = beatlength / 2;
  if (beatlength < 20) { // loop back to original tempo
    beatlength = 100;
  }
}
```