```
// Modified by Glen Popiel, KW5GP

// Based on OpenQRP Blog Iambic Morse Code Keyer Sketch by Steven T.
Elliott, K1EL - Used with Permission

//////////////////////////////////////////////////////////////////////
////////////////
//
//  Iambic Morse Code Keyer Sketch
//  Copyright (c) 2009 Steven T. Elliott
//
//  This library is free software; you can redistribute it and/or
//  modify it under the terms of the GNU Lesser General Public
//  License as published by the Free Software Foundation; either
//  version 2.1 of the License, or (at your option) any later version.
//
//  This library 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
//  Lesser General Public License for more details:
//
//  Free Software Foundation, Inc., 59 Temple Place, Suite 330,
//  Boston, MA  02111-1307  USA
//
//////////////////////////////////////////////////////////////////////
////////////////

#include <LCD5110_Basic.h>    // Nokia 5110 LCD

LCD5110 glcd(12,11,10,8,9);
extern uint8_t SmallFont[];

#define ST_Pin 4      // Sidetone Output Pin on Pin 4
#define LP_in 7       // Left Paddle Input on Pin 7
#define RP_in 5       // Right Paddle Input on Pin 5
#define led_Pin 13    // LED on Pin 13
#define Mode_A_Pin  3 // Mode Select Switch Side A
#define Mode_B_Pin  2 // Mode Select Switch Side B
#define key_Pin 6     // Transmitter Relay Key Pin
#define Speed_Pin 0   // Speed Control Pot on A0
#define ST_Freq 600   // Set the Sidetone Frequency to 600 Hz

int key_speed;        // variable for keying speed
int read_speed;       // variable for speed pot
int key_mode;         // variable for keying mode
int last_mode;        // variable to detect keying mode change

unsigned long ditTime;  // Number of milliseconds per dit
char keyerControl;
char keyerState;
String text1, text2, text3, text4, text5, text6 = " ";  // LCD text
variables

static long ktimer;
```

```
int debounce;

////////////////////////////////////////////////////////////////////////
////////////////
//  keyerControl bit definitions
//
#define     DIT_L      0x01     // Dit latch
#define     DAH_L      0x02     // Dah latch
#define     DIT_PROC   0x04     // Dit is being processed
#define     PDLSWAP    0x08     // 0 for normal, 1 for swap
#define     IAMBICB    0x10     // 0 for Iambic A, 1 for Iambic B
#define     ULTIMATIC  0x20     // 1 for ultimatic
#define     STRAIGHT   0x80     // 1 for straight key mode
////////////////////////////////////////////////////////////////////////
///////////////
//
//  State Machine Defines

enum KSTYPE {IDLE, CHK_DIT, CHK_DAH, KEYED_PREP, KEYED, INTER_ELEMENT };

void setup()
{
  // Setup outputs
  pinMode(led_Pin, OUTPUT);         // sets the LED digital pin as output
  pinMode(LP_in, INPUT);            // sets Left Paddle digital pin as
input
  pinMode(RP_in, INPUT);            // sets Right Paddle digital pin as
input
  pinMode(ST_Pin, OUTPUT);          // Sets the Sidetone digital pin as
output
  pinMode(key_Pin, OUTPUT);         // Sets the Keying Relay pin as
output
  pinMode(Mode_A_Pin, INPUT);       // sets Iambic Mode Switch Side A as
input
  pinMode(Mode_B_Pin, INPUT);       // sets Iambic Mode Switch Side B as
input
  digitalWrite(led_Pin, LOW);       // turn the LED off
  digitalWrite(LP_in, HIGH);        // Enable pullup resistor on Left
Paddle Input Pin
  digitalWrite(RP_in, HIGH);        // Enable pullup resistor on Right
Paddle Input Pin
  digitalWrite(Mode_A_Pin, HIGH);   // Enable pullup resistor on Mode
Switch Side A Input Pin
  digitalWrite(Mode_B_Pin, HIGH);   // Enable pullup resistor on Mode
Switch Side B Input Pin

  // Nokia 5110 Display
  glcd.InitLCD(60);
  glcd.setFont(SmallFont);

  cleartext();
  text1 = "KW5GP";
  text2 = "Iambic";
  text3 = "Keyer";
```

```
  text6 = "Initializing";
  updatelcd();
  delay(3000);

  keyerState = IDLE;
  keyerControl = 0;

  key_speed = map(analogRead(Speed_Pin),10,1000,5,35);  // Read the
potentiometer to determine code speed
  loadWPM(key_speed);   // Set the keying speed

  cleartext();
  text1 = "Keyer Ready";
  text3 = "Mode: Iambic-A";
  text5 = String(key_speed) + "wpm";
  updatelcd();

}  // End Setup Loop

void loop()
{
  // Read and set speed
  read_speed = map(analogRead(Speed_Pin),10,1000,5,35);  // Read the
potentiometer to determine code speed
  if (key_speed != read_speed)  // If the Speed Pot has changed, update
the speed and LCD
  {
    key_speed = read_speed;
    loadWPM(key_speed);
    text5 = String(key_speed) + "wpm";
    updatelcd();
  }

  // Read the Mode Switch and set mode
  //
  // Key Mode 0 = Iambic Mode A
  // Key Mode 1 = Iambic Mode B
  // Key Mode 2 = Straight Key
  //
  if (digitalRead(Mode_A_Pin) == LOW)   // Set Iambic Mode A
  {
    key_mode = 0;
    text3 = "Mode: Iambic-A";
  }
  if (digitalRead(Mode_B_Pin) == LOW)  // Set Iambic Mode B
  {
    key_mode = 1;
    text3 = "Mode: Iambic-B";
  }
  if (digitalRead(Mode_A_Pin) == HIGH && digitalRead(Mode_B_Pin) == HIGH)
// Set Straight Key
  {
    key_mode = 2;
    text3 = "Mode: Straight";
```

```
  }
  if (key_mode != last_mode)
  {
    last_mode = key_mode;
    updatelcd();
  }


  if (key_mode == 2) // Straight Key
  {
    // Straight Key Mode
    if ((digitalRead(LP_in) == LOW) || (digitalRead(RP_in) == LOW))
    {
      // Key from either paddle
      digitalWrite(led_Pin, HIGH);
      digitalWrite(key_Pin, HIGH);
      tone(ST_Pin, 600);
    } else {
      digitalWrite(led_Pin, LOW);
      digitalWrite(key_Pin, LOW);
      noTone(ST_Pin);
    }

  } else {

  // Basic Iambic Keyer
  // keyerControl contains processing flags and keyer mode bits
  // Supports Iambic A and B
  // State machine based, uses calls to millis() for timing.
  switch (keyerState)
  {
    case IDLE:      // Wait for direct or latched paddle press
      if ((digitalRead(LP_in) == LOW) || (digitalRead(RP_in) == LOW) ||
(keyerControl & 0x03))
      {
        update_PaddleLatch();
        keyerState = CHK_DIT;
      }
      break;

    case CHK_DIT:      // See if the dit paddle was pressed
      if (keyerControl & DIT_L)
      {
        keyerControl |= DIT_PROC;
        ktimer = ditTime;
        keyerState = KEYED_PREP;
      } else {
        keyerState = CHK_DAH;
      }
      break;

    case CHK_DAH:      // See if dah paddle was pressed
      if (keyerControl & DAH_L)
      {
```

```
        ktimer = ditTime*3;
        keyerState = KEYED_PREP;
      } else {
        keyerState = IDLE;
      }
        break;

    case KEYED_PREP:      // Assert key down, start timing, state shared
for dit or dah
      digitalWrite(led_Pin, HIGH);         // Turn the LED on
      tone(ST_Pin, ST_Freq);               // Turn the Sidetone on
      digitalWrite(key_Pin, HIGH);         // Key the TX Relay
      ktimer += millis();                  // set ktimer to interval end
time
      keyerControl &= ~(DIT_L + DAH_L);    // clear both paddle latch
bits
      keyerState = KEYED;                  // next state
      break;

    case KEYED:      // Wait for timer to expire
      if (millis() > ktimer)  // are we at end of key down ?
      {
        digitalWrite(led_Pin, LOW);      // Turn the LED off
        noTone(ST_Pin);                  // Turn the Sidetone off
        digitalWrite(key_Pin, LOW);      // Turn the TX Relay off
        ktimer = millis() + ditTime;     // inter-element time
        keyerState = INTER_ELEMENT;      // next state
      }
      else if (key_mode == 1) // Check to see if we're in Iambic B Mode
      {
        update_PaddleLatch();           // early paddle latch in Iambic B
mode
      }
      break;

    case INTER_ELEMENT:      // Insert time between dits/dahs
      update_PaddleLatch();               // latch paddle state
      if (millis() > ktimer)  // are we at end of inter-space ?
      {
        if (keyerControl & DIT_PROC)      // was it a dit or dah ?
        {
          keyerControl &= ~(DIT_L + DIT_PROC);   // clear two bits
          keyerState = CHK_DAH;                  // dit done, check for
dah
        } else {
          keyerControl &= ~(DAH_L);              // clear dah latch
          keyerState = IDLE;                     // go idle
        }
      }
      break;
    }
  }
}  // End Main Loop
```

```
//////////////////////////////////////////////////////////////////////
//////
//
//    Latch dit and/or dah press
//
//    Called by keyer routine
//
//////////////////////////////////////////////////////////////////////
//////

void update_PaddleLatch()
{
    if (digitalRead(RP_in) == LOW) {
        keyerControl |= DIT_L;
    }
    if (digitalRead(LP_in) == LOW) {
        keyerControl |= DAH_L;
    }

}

//////////////////////////////////////////////////////////////////////
//////
//
//    Calculate new time constants based on wpm value
//
//////////////////////////////////////////////////////////////////////
//////

void loadWPM (int wpm)
{
    ditTime = 1200/wpm;
}

void updatelcd()    // clears LCD display and writes the LCD data
{
  glcd.clrScr();
  glcd.print(text1,CENTER,0);
  glcd.print(text2,CENTER,8);
  glcd.print(text3,CENTER,16);
  glcd.print(text4,CENTER,24);
  glcd.print(text5,CENTER,32);
  glcd.print(text6,CENTER,40);
}

void cleartext()  // clears the text data
{
  text1 = " ";
  text2 = text1;
  text3 = text1;
  text4 = text1;
  text3 = text1;
  text4 = text1;
  text5 = text1;
```

```
  text6 = text1;
}
```