// A027-Final Code- Pats Infrared Mini Juke Box - 22/4/2019, 7 extra Tunes added 4/6/2023, checked 22/11/2023 and working fine.

//Pat's son Christian was the only person who was able to sort out the errors

//and get it working.

#include <IRremote.h>

#include <anyrtttl.h>

#include <avr/pgmspace.h>

// Pins

const int led1Pin = 2;

const int led2Pin = 3;

const int led3Pin = 4;

const int led4Pin = 5;

const int led5Pin = 6;

const int led6Pin = 7;

const int led7Pin = 8;

const int led8Pin = 9;

const int led9Pin = 10;

const int led10Pin = 11;

const int buzzerPin = 12;  // hooks up the speaker to pin 12

const int irReceivePin = A4;

const int baudRate = 9600;

const int delayTime = 200;

IRrecv irReceive(irReceivePin);

void setup()

{

  // set all the LED pins to outputs

  pinMode(led1Pin, OUTPUT);

  pinMode(led2Pin, OUTPUT);

  pinMode(led3Pin, OUTPUT);

  pinMode(led4Pin, OUTPUT);

  pinMode(led5Pin, OUTPUT);

  pinMode(led6Pin, OUTPUT);

  pinMode(led7Pin, OUTPUT);

  pinMode(led8Pin, OUTPUT);

  pinMode(led9Pin, OUTPUT);

  pinMode(led10Pin, OUTPUT);

  pinMode(buzzerPin, OUTPUT);

  pinMode(irReceivePin, INPUT);

  irReceive.enableIRIn();

  Serial.begin(baudRate);

}

void loop()

{

  decode\_results irResults;

  if (irReceive.decode(&irResults))

  {

    Serial.print(F("IR: 0x"));

    Serial.println(irResults.value, HEX);

    // Stop the IR Receiver because the interrupt cannot be disabled for extended periods

    auto tStartMillis = millis();

    IrReceiver.stop();

    dispatch(irResults.value);

    Serial.println(F("DEBUG: dispatch() completed")); // Debugging - Remove me when bug is fixed

    // Restore IR timer. millis() - tStartMillis to compensate for stop of receiver. This enables a correct gap measurement.

    IrReceiver.startWithTicksToAdd((millis() - tStartMillis) \* (MICROS\_IN\_ONE\_MILLI / MICROS\_PER\_TICK));

    irReceive.resume(); // Receive the next value

    Serial.println(F("DEBUG: irReceive.resume() completed")); // Debugging - Remove me when bug is fixed

    delay(100);

  }

}

void dispatch(unsigned long irCode)

{

  switch(irCode)

  {

    // LED Buttons

    case 0x010:

      // 1 Button

      digitalWrite(led1Pin, HIGH);

      break;

    case 0x810:

      // 2 Button

      digitalWrite(led2Pin, HIGH);

      break;

    case 0x410:

      // 3 Button

      digitalWrite(led3Pin, HIGH);

      break;

    case 0xC10:

      // 4 Button

      digitalWrite(led4Pin, HIGH);

      break;

   case 0x210:

      // 5 Button

      digitalWrite(led5Pin, HIGH);

      break;

   case 0xA10:

      // 6 Button

      digitalWrite(led6Pin, HIGH);

      break;

   case 0x610:

      // 7 Button

      digitalWrite(led7Pin, HIGH);

      break;

   case 0xE10:

      // 8 Button

      digitalWrite(led8Pin, HIGH);

      break;

   case 0x110:

      // 9 Button

      digitalWrite(led9Pin, HIGH);

      break;

   case 0x910:

      // 0 Button

      digitalWrite(led10Pin, HIGH);

      break;

   case 0xA70:

      // Centre Button

      digitalWrite(led1Pin, LOW);

      digitalWrite(led2Pin, LOW);

      digitalWrite(led3Pin, LOW);

      digitalWrite(led4Pin, LOW);

      digitalWrite(led5Pin, LOW);

      digitalWrite(led6Pin, LOW);

      digitalWrite(led7Pin, LOW);

      digitalWrite(led8Pin, LOW);

      digitalWrite(led9Pin, LOW);

      digitalWrite(led10Pin, LOW);

      break;

   case 0x2F0:

      // Up Button

      digitalWrite(led1Pin, HIGH);

      delay(delayTime);

      digitalWrite(led1Pin, LOW);

      delay(delayTime);

      digitalWrite(led2Pin, HIGH);

      delay(delayTime);

      digitalWrite(led2Pin, LOW);

      delay(delayTime);

      digitalWrite(led3Pin, HIGH);

      delay(delayTime);

      digitalWrite(led3Pin, LOW);

      delay(delayTime);

      digitalWrite(led4Pin, HIGH);

      delay(delayTime);

      digitalWrite(led4Pin, LOW);

      delay(delayTime);

      digitalWrite(led5Pin, HIGH);

      delay(delayTime);

      digitalWrite(led5Pin, LOW);

      delay(delayTime);

      digitalWrite(led6Pin, HIGH);

      delay(delayTime);

      digitalWrite(led6Pin, LOW);

      delay(delayTime);

      digitalWrite(led7Pin, HIGH);

      delay(delayTime);

      digitalWrite(led7Pin, LOW);

      delay(delayTime);

      digitalWrite(led8Pin, HIGH);

      delay(delayTime);

      digitalWrite(led8Pin, LOW);

      delay(delayTime);

      digitalWrite(led9Pin, HIGH);

      delay(delayTime);

      digitalWrite(led9Pin, LOW);

      delay(delayTime);

      digitalWrite(led10Pin, HIGH);

      delay(delayTime);

      digitalWrite(led10Pin, LOW);

      delay(delayTime);

      break;

    case 0xAF0:

      // Down Button

      digitalWrite(led10Pin, HIGH);

      delay(delayTime);

      digitalWrite(led10Pin, LOW);

      delay(delayTime);

      digitalWrite(led9Pin, HIGH);

      delay(delayTime);

      digitalWrite(led9Pin, LOW);

      delay(delayTime);

      digitalWrite(led8Pin, HIGH);

      delay(delayTime);

      digitalWrite(led8Pin, LOW);

      delay(delayTime);

      digitalWrite(led7Pin, HIGH);

      delay(delayTime);

      digitalWrite(led7Pin, LOW);

      delay(delayTime);

      digitalWrite(led6Pin, HIGH);

      delay(delayTime);

      digitalWrite(led6Pin, LOW );

      delay(delayTime);

      digitalWrite(led5Pin, HIGH);

      delay(delayTime);

      digitalWrite(led5Pin, LOW);

      delay(delayTime);

      digitalWrite(led4Pin, HIGH);

      delay(delayTime);

      digitalWrite(led4Pin, LOW);

      delay(delayTime);

      digitalWrite(led3Pin, HIGH);

      delay(delayTime);

      digitalWrite(led3Pin, LOW);

      delay(delayTime);

      digitalWrite(led2Pin, HIGH);

      delay(delayTime);

      digitalWrite(led2Pin, LOW);

      delay(delayTime);

      digitalWrite(led1Pin, HIGH);

      delay(delayTime);

      digitalWrite(led1Pin, LOW);

      delay(delayTime);

      break;

    case 0x2D0:

      // Left Button

      digitalWrite(led10Pin, HIGH);

      digitalWrite(led1Pin, HIGH);

      delay(delayTime);

      digitalWrite(led10Pin, LOW);

      digitalWrite(led1Pin, LOW);

      delay(delayTime);

      digitalWrite(led9Pin, HIGH);

      digitalWrite(led2Pin, HIGH);

      delay(delayTime);

      digitalWrite(led9Pin, LOW);

      digitalWrite(led2Pin, LOW);

      delay(delayTime);

      digitalWrite(led8Pin, HIGH);

      digitalWrite(led3Pin, HIGH);

      delay(delayTime);

      digitalWrite(led8Pin, LOW);

      digitalWrite(led3Pin, LOW);

      delay(delayTime);

      digitalWrite(led7Pin, HIGH);

      digitalWrite(led4Pin, HIGH);

      delay(delayTime);

      digitalWrite(led7Pin, LOW);

      digitalWrite(led4Pin, LOW);

      delay(delayTime);

      digitalWrite(led6Pin, HIGH);

      digitalWrite(led5Pin, HIGH);

      delay(delayTime);

      digitalWrite(led6Pin, LOW);

      digitalWrite(led5Pin, LOW);

      delay(delayTime);

      break;

    case 0xCD0:

      // Right Button

      digitalWrite(led6Pin, HIGH);

      digitalWrite(led5Pin, HIGH);

      delay(delayTime);

      digitalWrite(led6Pin, LOW);

      digitalWrite(led5Pin, LOW);

      delay(delayTime);

      digitalWrite(led7Pin, HIGH);

      digitalWrite(led4Pin, HIGH);

      delay(delayTime);

      digitalWrite(led7Pin, LOW);

      digitalWrite(led4Pin, LOW);

      delay(delayTime);

      digitalWrite(led8Pin, HIGH);

      digitalWrite(led3Pin, HIGH);

      delay(delayTime);

      digitalWrite(led8Pin, LOW);

      digitalWrite(led3Pin, LOW);

      delay(delayTime);

      digitalWrite(led9Pin, HIGH);

      digitalWrite(led2Pin, HIGH);

      delay(delayTime);

      digitalWrite(led9Pin, LOW);

      digitalWrite(led2Pin, LOW);

      delay(delayTime);

      digitalWrite(led10Pin, HIGH);

      digitalWrite(led1Pin, HIGH);

      delay(delayTime);

      digitalWrite(led10Pin, LOW);

      digitalWrite(led1Pin, LOW);

      delay(delayTime);

      break;

    // Tunes Buttons

    case 0x338:

      // Picture Button

      Serial.println(F("The Simpsons"));

      PROGMEM static const char theSimpsons[] = "The Simpsons:d=4,o=5,b=160:c.6,e6,f#6,8a6,g.6,e6,c6,8a,8f#,8f#,8f#,2g,8p,8p,8f#,8f#,8f#,8g,a#.,8c6,8c6,8c6,c6";

      anyrtttl::blocking::playProgMem(buzzerPin, theSimpsons);

      break;

    case 0xB38:

      // Audio Button

      Serial.println(F("Star Wars"));

      PROGMEM static const char starWars[] = "StarWars:d=4,o=5,b=45:32p,32f#,32f#,32f#,8b.,8f#.6,32e6,32d#6,32c#6,8b.6,16f#.6,32e6,32d#6,32c#6,8b.6,16f#.6,32e6,32d#6,32e6,8c#.6,32f#,32f#,32f#,8b.,8f#.6,32e6,32d#6,32c#6,8b.6,16f#.6,32e6,32d#6,32c#6,8b.6,16f#.6,32e6,32d#6,32e6,8c#6";

      anyrtttl::blocking::playProgMem(buzzerPin, starWars);

      break;

    case 0x738:

      // Time Button

      Serial.println(F("Mission Impossible"));

      PROGMEM static const char missionImpossible[] = "MissionImp:d=16,o=6,b=95:32d,32d#,32d,32d#,32d,32d#,32d,32d#,32d,32d,32d#,32e,32f,32f#,32g,g,8p,g,8p,a#,p,c7,p,g,8p,g,8p,f,p,f#,p,g,8p,g,8p,a#,p,c7,p,g,8p,g,8p,f,p,f#,p,a#,g,2d,32p,a#,g,2c#,32p,a#,g,2c,a#5,8c,2p,32p,a#5,g5,2f#,32p,a#5,g5,2f,32p,a#5,g5,2e,d#,8d";

      anyrtttl::blocking::playProgMem(buzzerPin, missionImpossible);

      break;

    case 0xF38:

      // System Button

      Serial.println(F("007"));

      PROGMEM static const char jamesBond[] = "Bond:d=4,o=5,b=80:32p,16c#6,32d#6,32d#6,16d#6,8d#6,16c#6,16c#6,16c#6,16c#6,32e6,32e6,16e6,8e6,16d#6,16d#6,16d#6,16c#6,32d#6,32d#6,16d#6,8d#6,16c#6,16c#6,16c#6,16c#6,32e6,32e6,16e6,8e6,16d#6,16d6,16c#6,16c#7,c.7,16g#6,16f#6,g#.6";

      anyrtttl::blocking::playProgMem(buzzerPin, jamesBond);

      break;

    case 0x6D0:

      // Dot Button

      Serial.println(F("Indiana Jones"));

      PROGMEM static const char indianaJones[] = "Indiana:d=4,o=5,b=250:e,8p,8f,8g,8p,1c6,8p.,d,8p,8e,1f,p.,g,8p,8a,8b,8p,1f6,p,a,8p,8b,2c6,2d6,2e6,e,8p,8f,8g,8p,1c6,p,d6,8p,8e6,1f.6,g,8p,8g,e.6,8p,d6,8p,8g,e.6,8p,d6,8p,8g,f.6,8p,e6,8p,8d6,2c6";

      anyrtttl::blocking::playProgMem(buzzerPin, indianaJones);

      break;

    case 0x1EE9:

      // I<< Button

      Serial.println(F("Addams Family"));

      PROGMEM static const char addamsFamily[] = "AddamsFamily:d=4,o=5,b=160:8c,f,8a,f,8c,b4,2g,8f,e,8g,e,8e4,a4,2f,8c,f,8a,f,8c,b4,2g,8f,e,8c,d,8e,1f,8c,8d,8e,8f,1p,8d,8e,8f#,8g,1p,8d,8e,8f#,8g,p,8d,8e,8f#,8g,p,8c,8d,8e,8f";

      anyrtttl::blocking::playProgMem(buzzerPin, addamsFamily);

      break;

    case 0xCE9:

      // Square Button

      Serial.println(F("ABBA"));

      PROGMEM static const char abba[] = "ABBA:d=4,o=6,b=45:16d#6,16d#6,32d#6,32f6,32g6,32g#6,32g#6,32g6,32f6,16f.6,32d6,32d6,32d6,32d6,32d#6,32f6,32g6,32g6,32f6,32d#6,16d#.6,32d#6,32d#6,32d#6,32d#6,32f6,32g6,32g#6,32g#6,32g6,32f6,16f6";

      anyrtttl::blocking::playProgMem(buzzerPin, abba);

      break;

    case 0x5EE9:

      // >>I Button

      Serial.println(F("Beatles"));

      PROGMEM static const char beatles[] = "Beatles Let It Be:d=4,o=5,b=100:16e6,8d6,c6,16e6,8g6,8a6,8g.6,16g6,8g6,8e6,16d6,8c6,16a,8g,e.6,p,8e6,16e6,8f.6,8e6,8e6,8d6,16p,16e6,16d6,8d6,2c.6";

      anyrtttl::blocking::playProgMem(buzzerPin, beatles);

      break;

    case 0x6CE9:

      // << Button

      Serial.println(F("Barbie Girl"));

      PROGMEM static const char barbieGirl[] = "Barbiegirl:d=4,o=5,b=125:8g#,8e,8g#,8c#6,a,p,8f#,8d#,8f#,8b,g#,8f#,8e,p,8e,8c#,f#,c#,p,8f#,8e,g#,f#";

      anyrtttl::blocking::playProgMem(buzzerPin, barbieGirl);

      break;

    case 0x2CE9:

      // > Button

      Serial.println(F("Flintstones"));

      PROGMEM static const char flintstones[] = "Flntstn:d=4,o=5,b=200:g#,c#,8p,c#6,8a#,g#,c#,8p,g#,8f#,8f,8f,8f#,8g#,c#,d#,2f,2p,g#,c#,8p,c#6,8a#,g#,c#,8p,g#,8f#,8f,8f,8f#,8g#,c#,d#,2c#";

      anyrtttl::blocking::playProgMem(buzzerPin, flintstones);

      break;

    case 0x4CE9:

      // II Button

      Serial.println(F("Funky Town"));

      PROGMEM static const char funkyTown[] = "Funkytown:d=4,o=6,b=125:8c#6,8c#6,8b5,4c#6,4g#5,8g#5,8c#6,8f#6,8f6,1c#6,8c#6,8c#6,8b5,4c#6,4g#5,8g#5,8c#6,8f#6,8f6,8c#6,";

      anyrtttl::blocking::playProgMem(buzzerPin, funkyTown);

      break;

    case 0x1CE9:

      // >> Button

      Serial.println(F("Sesamestreet"));

      PROGMEM static const char sesameStreet[] = "sesamestreet:d=4,o=5,b=160:2c6,a,2f,8f,8g,a,p,2c6,a,1f,p,2c6,a,2f,8f,g,a,2b,c6,2d6,p,8c6,8d6,d#6,d6,c6,a,g,8g,8a,a#,a,8g,8c,8c,1c";

      anyrtttl::blocking::playProgMem(buzzerPin, sesameStreet);

      break;

  }

}