LunaMod s ATtiny45

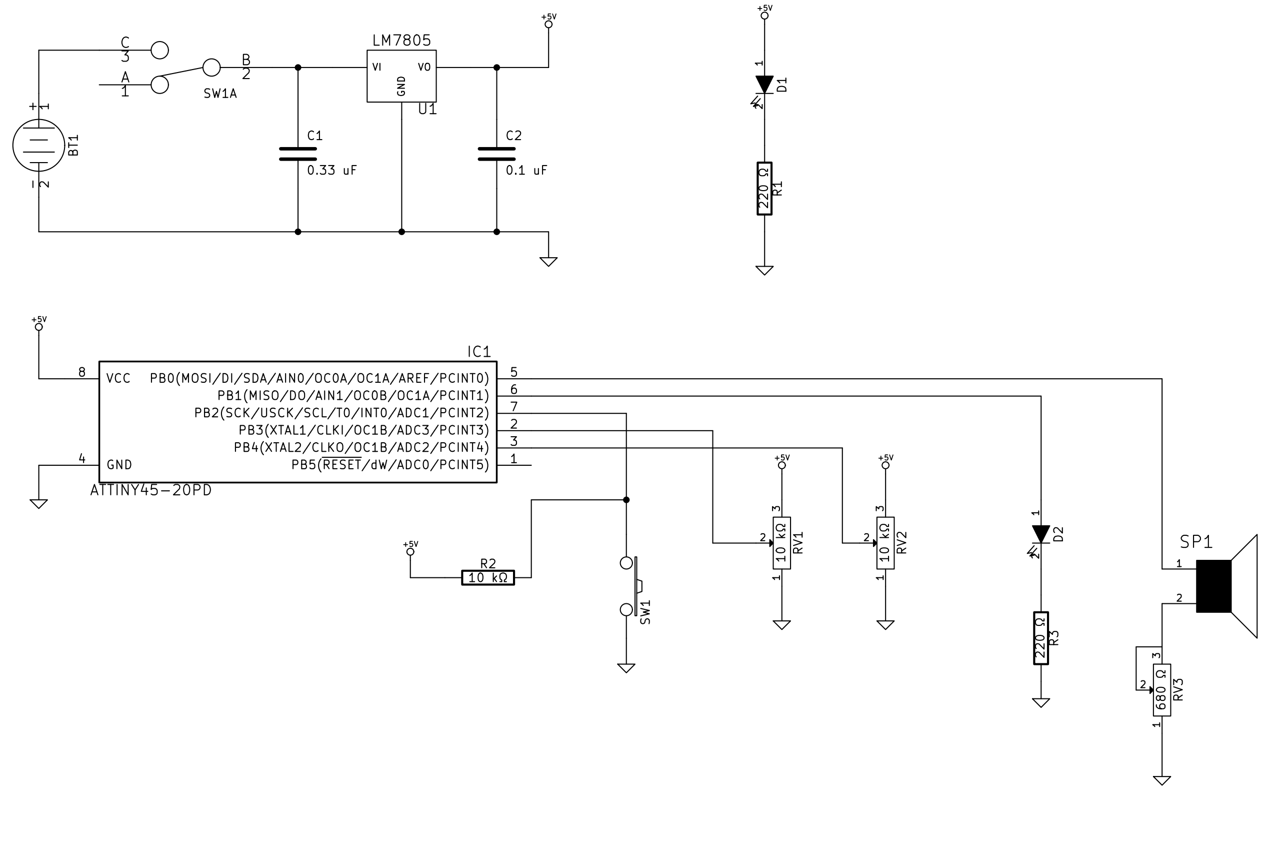
Juraj Smieško

11. júna 2012

1 Úvod

Už dávnejšie ma zaujal takzvaný „noise generator“ LunaMod od Briana McNamara [1]. Bol však postavený na architekture PicAxe. Rob Miles prepísal kód pre Arduino [2] a pomocou návodu od [3] sa dá použiť aj ATtiny44/45/84/85.

2 Schéma


+5V
C
3
A
1
B
2
SW1A
LM7805
VI
VO
GND
U1
BT1
+
1
-
2
C1
0.33 uF
C2
0.1 uF
D1
220 Ω
R1
IC1
8
VCC
PB0(MOSI/DI/SDA/AIN0/OC0A/OC1A/AREF/PCINT0)
5
PB1(MISO/DO/AIN1/OC0B/OC1A/PCINT1)
6
PB2(SCK/USCK/SCL/T0/INT0/ADC1/PCINT2)
7
PB3(XTAL1/CLKI/OC1B/ADC3/PCINT3)
2
PB4(XTAL2/CLKO/OC1B/ADC2/PCINT4)
3
4
GND
PB5(RESET/dW/ADC0/PCINT5)
1
ATTINY45-20PD
R2
10 kΩ
SW1
10 kΩ
RV1
10 kΩ
RV2
D2
220 Ω
R3
SP1
680 Ω
RV3

Obr. 1: Schéma zapojenia.

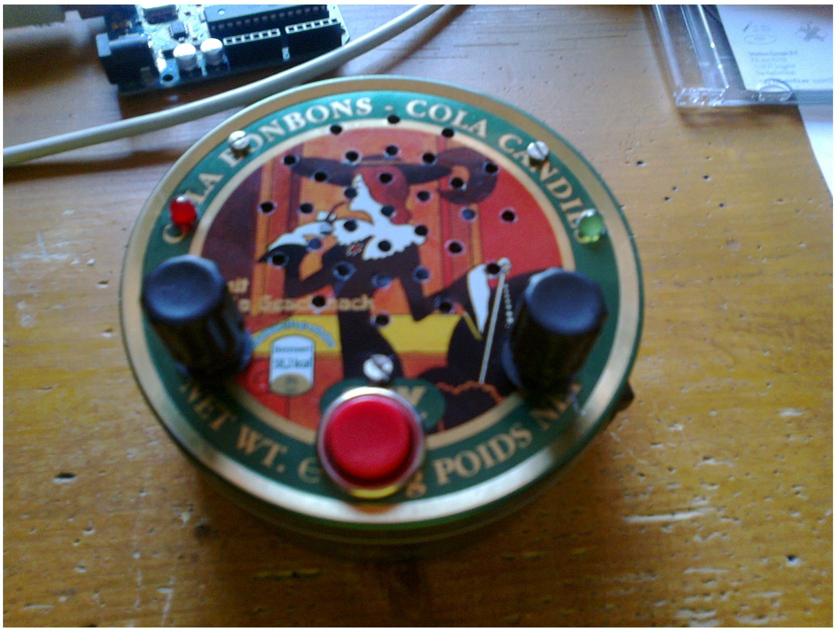
3 Program

Program nachádzajúci sa v ATtiny45 je založený na [2]:

```
/*
 *
 * LunaMod for Arduino & Attiny45
 * Remix by Rob Miles
 * Tacoma, WA August 8th 2011
 *
 * I saw the original project in Make vol. 26
 * by Brian McNamara
 * Brian's was running on a PicAxe and I only have attiny45s so.......
 *
 * The freqout section where the real magic happens is from Paul Badger's
 * synth code on the main Arduino site
 *
 * I kept this pretty straight forward but with an Arduino this could get
 * a lot more complicated if you like. Even on an attiny45 if you use you
 * add a button to the led pin you coud sneak in more effects
 *
 */

#define frequencyPot   3                          // frequency pot tied to pin 15 which is A1
#define tempoPot       2                          // tempo pot tied to pin 16 which is A2
#define buttonPin      2                          // programming button tied to pin 17 which is A3
#define ledPin         1                          // status led tied to pin 18 which is A4
#define speakerPin     0                          // speaker or output pin 19 whish is A5
                                                  // if you use a speaker it should be at least
                                                  // a 16 ohm speaker an should have a resistor,
                                                  // maybe 200ohm to 1K ohm, between the negative
                                                  // lead and ground.
                                                  // a potentiometer would be even better.

int currentStep = 0;                              // this is just to track which tone in memory
                                                  // we are currently playing

int steps[] = {500,500,100,100,100,100,100,100,   // this is our tone storage areae
               100,100,100,100,100,100,100,100,   // I used 64 tones or 8 tones per beat
               500,500,100,100,100,100,100,100,   // you can change these manually and experiment
               100,100,100,100,100,100,100,100,   // if you like
               500,500,100,100,100,100,100,100,
               100,100,100,100,100,100,100,100,
               500,500,100,100,100,100,100,100,
               100,100,100,100,100,100,100,100};

int tempo = 0;                                    // tempo or speed between tones
int duration = 0;                                 // how long each of the 64 tones plays
int frequency = 0;                                // current tone
int pitchval = 1;

void setup()                                      // set up your pins....
{
  pinMode (frequencyPot, INPUT);
  pinMode (tempoPot, INPUT);
  pinMode (buttonPin, INPUT);
  digitalWrite(buttonPin, HIGH);

  pinMode (ledPin, OUTPUT);
  pinMode (speakerPin, OUTPUT);

}

void loop()
{
  for (int i=0; i<63; i++)                            // 64 individual notes played
  {

  currentStep = i;                                    // save our current position in the loop
                                                      // for later

  if (i == 0 || i == 16 || i == 32 || i == 48){       // keep track of the beat on the led
  digitalWrite(ledPin, HIGH);}

  if (i == 7 || i == 23 || i == 39 || i == 55){       // keep track of the beat on the led
  digitalWrite(ledPin, LOW);}





  if (digitalRead(buttonPin) == LOW)                  // is the program button being pressed
    {                                                 // if so lets write a new tone the this location

    steps[currentStep] = (analogRead(frequencyPot));  // read the frequency pot and set the new tone
    freqout (steps[currentStep], duration);           // set the parameters for frequout below and play it
    freqout (steps[currentStep]+64, duration);        // play another tone a little bit different than
                                                      // the original to give
                                                      // it a little more depth. this can be changed to
                                                      // your liking
    freqout (steps[currentStep]+128, duration);       // play another tone a little bit different than
                                                      // the original to give
                                                      // it a little more depth. this can be changed to
                                                      // your liking

    }

    else {                                            // else play the tone

    freqout (steps[currentStep], duration);           // set the parameters for frequout below and play it
    freqout (steps[currentStep]+64, duration);        // play another tone a little bit different than
                                                      // the original to give
                                                      // it a little more depth. this can be changed to
                                                      // your liking
    freqout (steps[currentStep]+128, duration);       // play another tone a little bit different than
                                                      // the original to give
                                                      // it a little more depth. this can be changed to
                                                      // your liking


  }
    tempo = (analogRead(tempoPot)/4);                 // read the tempo pot
    duration = tempo/8;                               // set the individual tone durations
    delay(tempo);                                     // wait a bit
}
}


void freqout(int freq, int t)
{
  int hperiod;
  long cycles, i;

  hperiod = (500000 / ((freq - 7) * pitchval));

  cycles = ((long)freq * (long)t) / 1000;

  for (i=0; i<= cycles; i++)
  {
    digitalWrite(speakerPin, HIGH);
    delayMicroseconds(hperiod);
    digitalWrite(speakerPin, LOW);
    delayMicroseconds(hperiod - 1);
  }
}
```

Keďže sa ATtiny nachádza v pätici a je možné ho preprogramovať, upravujem program podľa aktuálnej nálady.

4 Výsledok



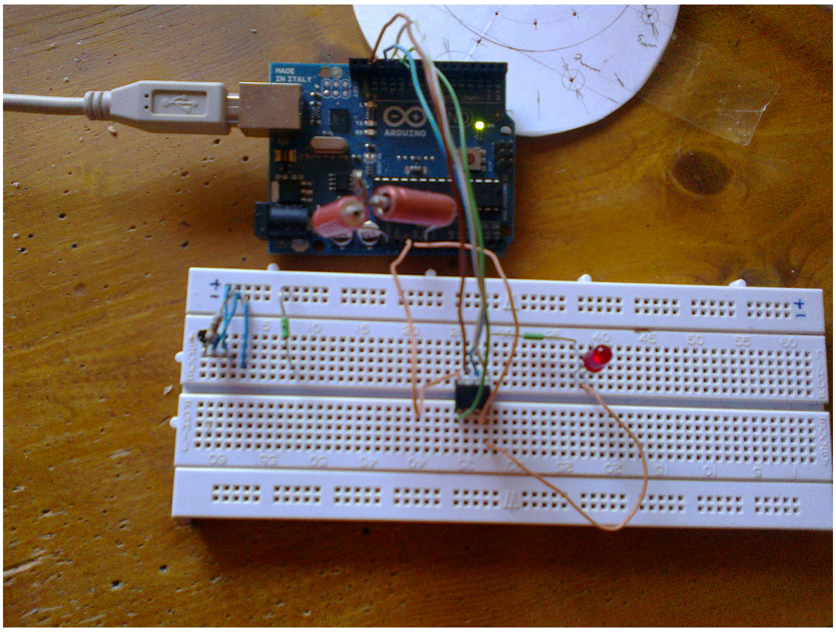
Obr. 2: Hotový LunaMod, ľavý potenciometer ovláda tempo, pravý potenciometer ovláda frekvenciu, červená led vľavo indikuje tempo, zelená led vpravo zapnutie/vypnutie.

Literatúra

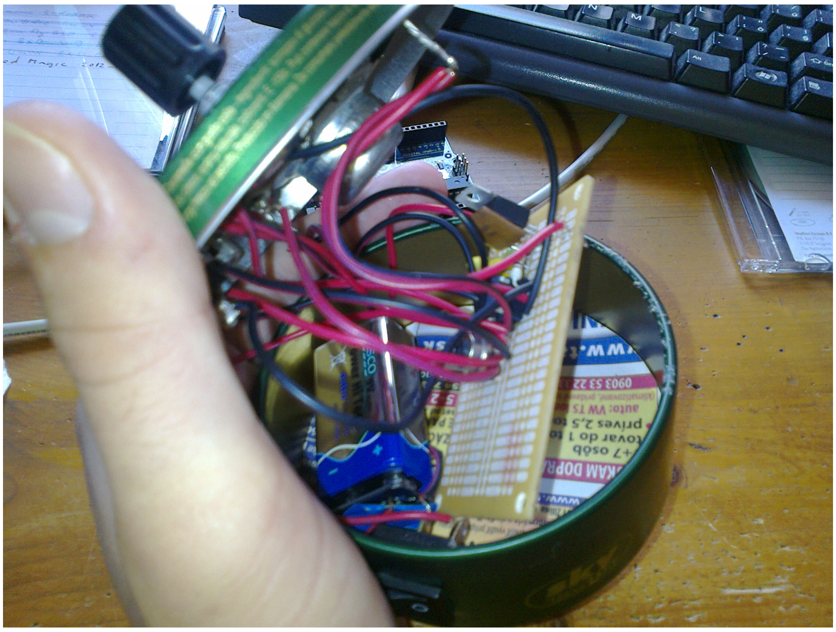
[1] http://makezine.com/26/lunamod/ [cit. 2012-06-06]

[2] http://hackaday.com/2011/08/08/noise-generator-ported-to-run-on-small-avr-also-arduino-compatible/ [cit. 2012-06-06]

[3] http://hlt.media.mit.edu/?p=1695 [cit. 2012-06-06]



Obr. 3: Programovanie ATtiny45.



Obr. 4: Vnútornosti LunaMod-u.