MERILNIK FREKVNCE STRUNE

Projektna naloga pri predmetu elektronska vezja

Tomaž Kobol

Kazalo

1. Uvod

Merilnik deluje kot števec impulzov, ki jih dobimo s pomočjo peak detektorja in komparatorja, katere mikrokontroler ARM7 prešteje in preračunano vrednost izpiše na zaslon.

1.1. Kitara

Najpogostejša uglasitev je tako imenovana EHGDAE. Toni in frekvence strun so predstavljeni v tabeli, pri čemer je prva struna tista, ki zveni najvišje in šesta tista, ki zveni najnižje.

82.41	E6
110.00	A5
146.83	D4
196.00	G3
246.94	B2
329.63	E1

Tabela 1: Osnovne frekvence in imena tonov praznih strun na kitari

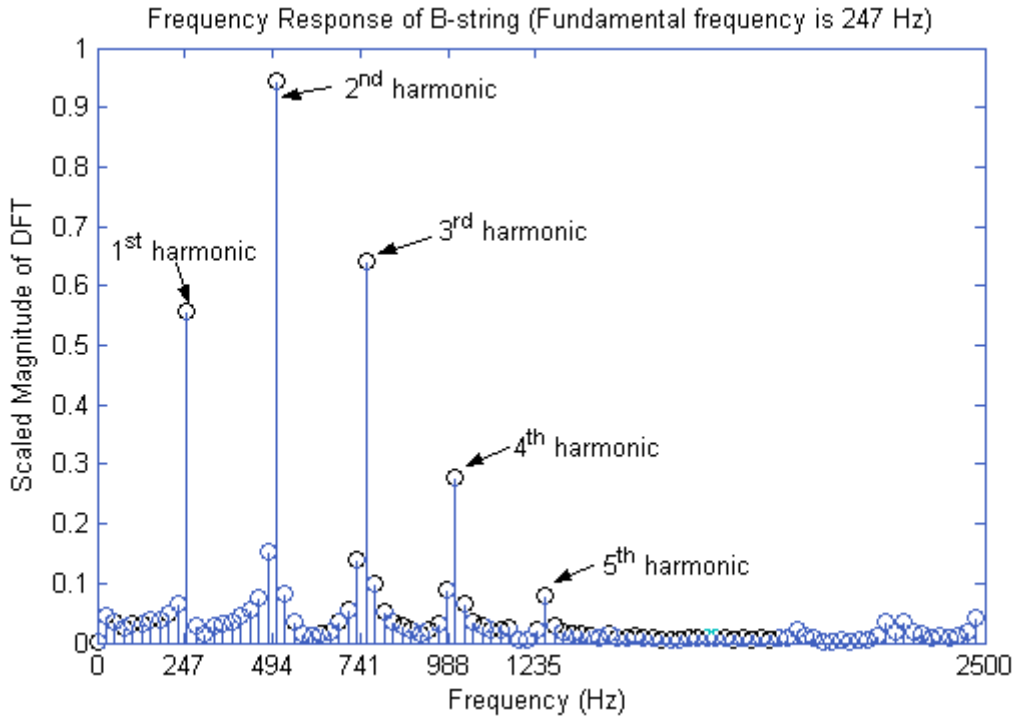
2. Projekt

2.1. Kitarski signal

Preden sem se dodobra lotil projekta sem moral nekoliko podrobneje proučiti obliko analognega signala, ki ga kitara proizvede. Razlikujemo lahko med zvenom in tonom.

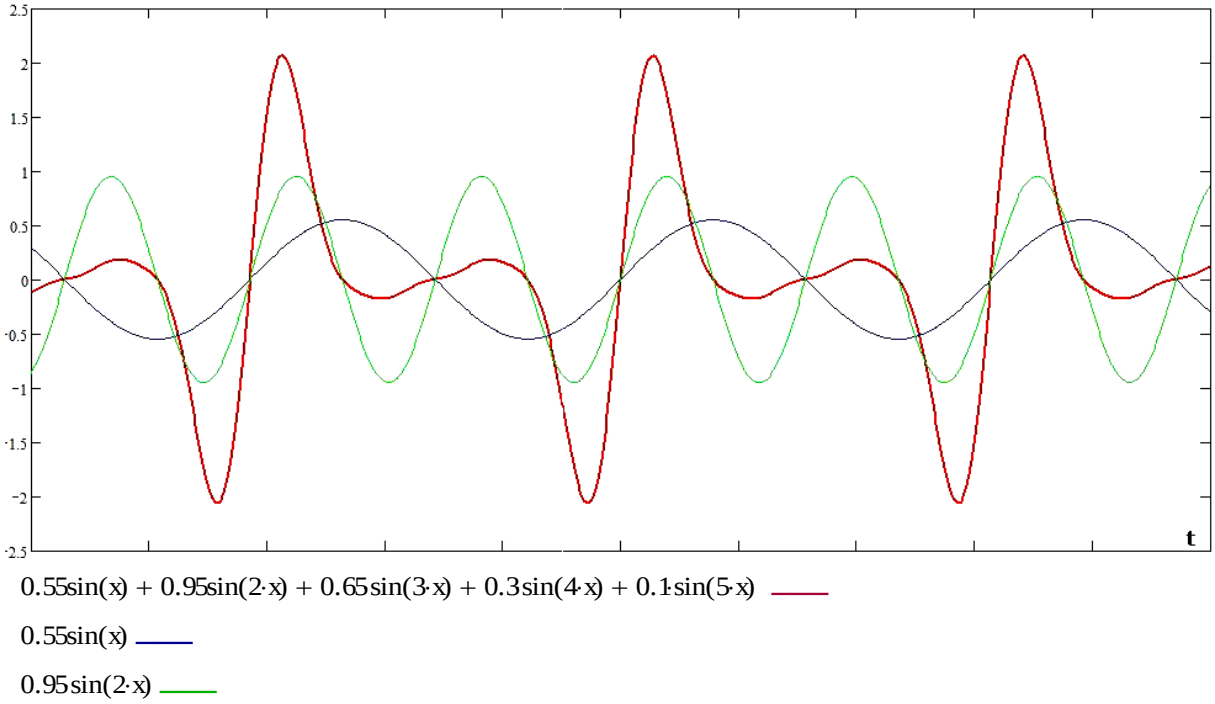
Ton: harmonsko nihanje z eno frekvenco. V življenju ga srečamo redko, generiramo ga lahko z glasbenimi vilicami ali elektronsko – s frekvenčnim generatorjem.

Zven: pravimo mu tudi glasbeni ton, je nihanje, sestavljeno iz osnovne frekvence in njenih višjih harmonskih nihanj. Višje harmonska nihanja določajo barvo zvoka. Glasbeni inštrumenti lahko oddajajo zvok enake frekvence, pa se bodo po zvenu razlikovali - prav zaradi višjih harmonikov.



Slika1: Zven in njegov frekvenčni spekter

Zgornja slika nam kaže frekvenčni odziv druge strune na kitari, to je ton H s frekvenco 247 Hz. Nekoliko nepričakovano je, da je druga harmonska komponenta višja od prve, kar mi je sprva povzročalo kar nekaj težav pri izdelavi merilnika frekvence.



Slika2: Signal kitare v odvisnosti od časa. Vrednosti amplitud in harmonikov so določene iz zgornjega spektra

Analogni signal, ki ga oddaja kitara je, kot smo sami videli, vse prej kot sinusen. Tako dobimo kot rezultat udarca drsalke po struni še kopico alikvontnih tonov. To so tisti toni, ki

kot smo že omenili, določajo barvo zvoka in so višje oziroma "nižje" harmoniki ($f_{osnovna}$, 2 $f_{osnovna}$, $f_{osnovna}/2$, …). Poleg tega je problem tudi, da signal zelo hitro izzveni, zato bi samo s pomočjo komparatoja zelo težko določili mejo od katere bi na izhodu dobili "pravilne" impulze.

Možnosti za rešitev problema:

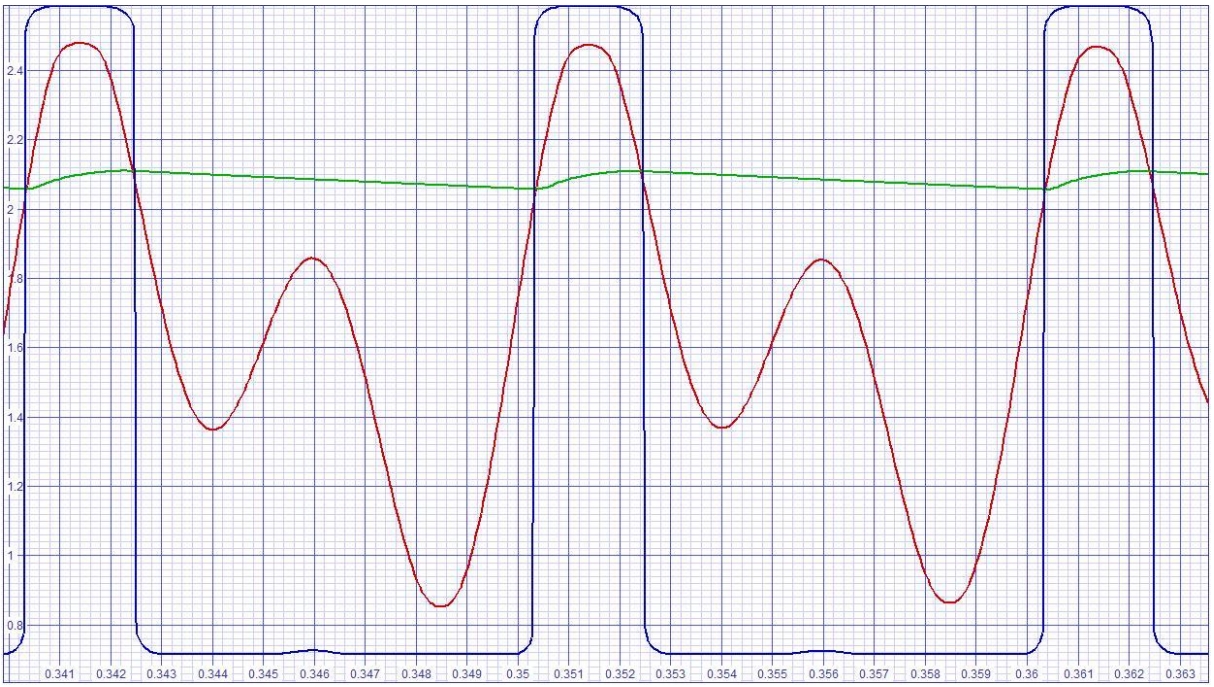
- AD pretvornik, signal bi obdelali digitalno
- filtri: 6 pasovnih filtrov, ki bi izsejali osnovne frekvence posameznih strun
- s pomočjo peak detektorja in komparatorja:

2.2. Vezje in njegovo delovanje:

Prva stopnja:
Izhodnemu signalu odrežemo enosmerno komponento, in mu prištejemo 1,65V enosmerne napetosti, kar je ravno polovica napajanja. Nato signal ojačamo za 10 krat z ojačevalnikom z negativno povratno vezavo.

Druga stopnja:
Nizkoprepustni filter oslabi frekvence nad 1kHz.

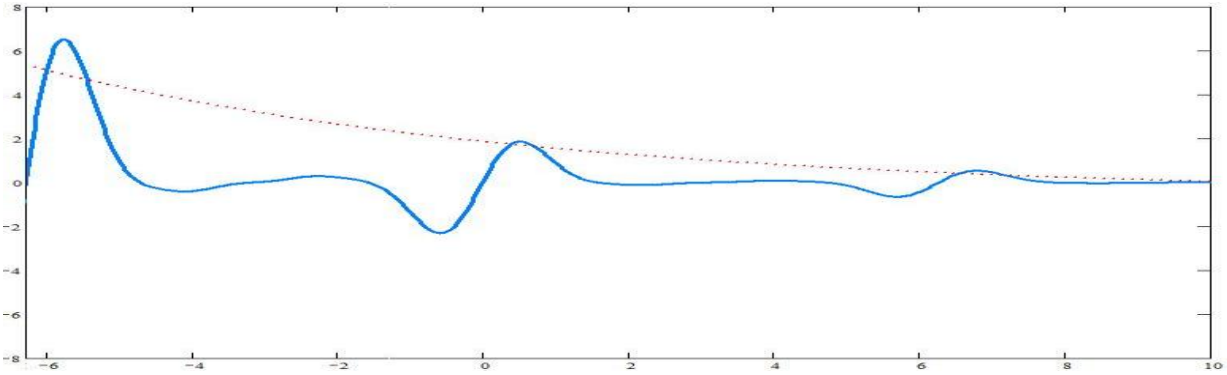
Tretja stopnja:
Detektor vrhnje vrednosti in komparator. Rešitev za prisotnost višjih harmonikov in močnega dušenja signala.



Slika3: Simulacija s programom Spice: *- ojačan signal*
-napetost na kondenzatorju
-impulzi

Signal razdelimo na originalen signal in 90% njegove vrednosti, to naredimo s pomočjo uporovnega delilnika in kondenzatorja, ki zadržuje peak vrednost originalnega signala (zelen signal na zgornji sliki) in sledi vhodnemu signalu (rdeč). Oba signala nato primerja

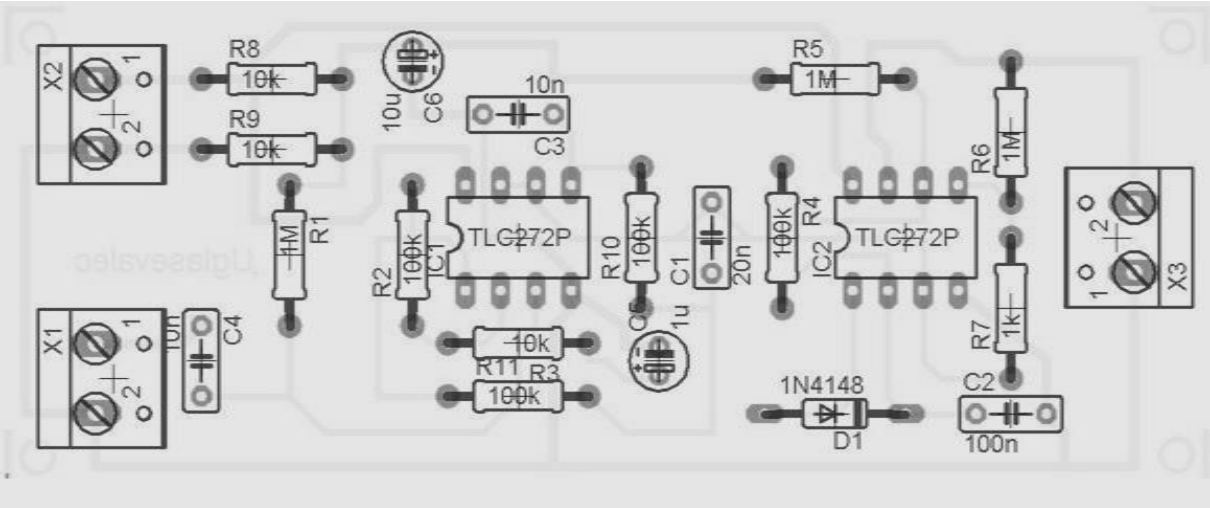
komparator in na izhodu da impulz (modra) le, če se originalni signal nahaja nad 90% peak vrednosti na uporovnem delilniku.

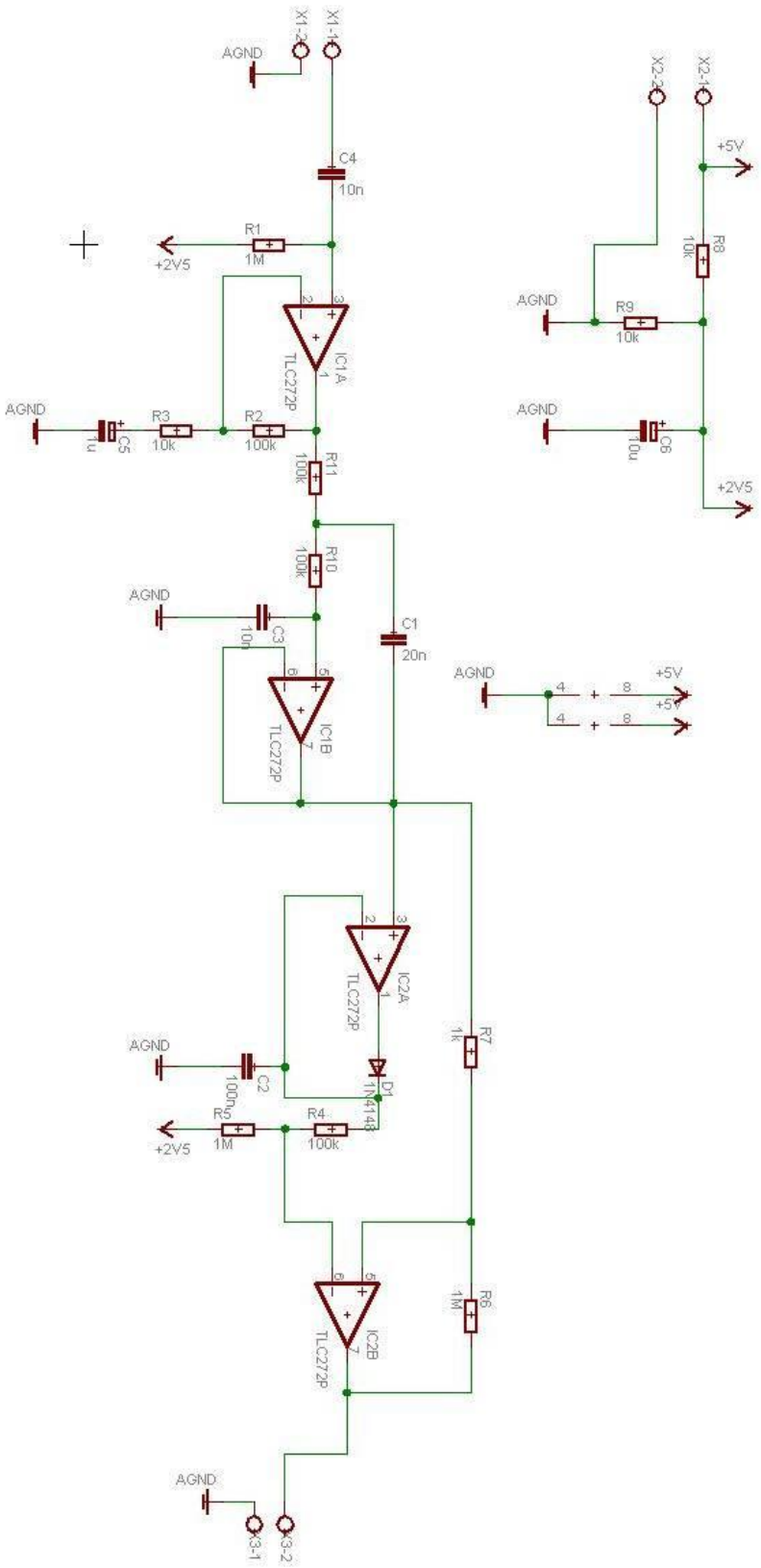


Slika 4: zgolj simbolno prikazan dušen signal in 90% peak vrednosti.

Dogajanje v mikrokontrolerju ARM7:
Procesor šteje impulze ki se pojavijo iz našega vezja in iz impulzov izračuna frekvenco.

2.3. Ploščica in shema vezja




X1-2
X1-1
AGND
C4
10n
R1
1M
+2V5
IC1A
TLC272P
R3
10k
R2
100k
C5
1u
R11
100k
R10
100k
C3
10n
C1
20n
IC1B
TLC272P
IC2A
TLC272P
R7
1k
C2
100n
D1
1N4148
R5
1M
R4
100k
IC2B
TLC272P
R6
1M
X3-1
X3-2
X2-2
X2-1
+5V
R8
10k
R9
10k
C6
10u
+2V5

2.4.Programska koda

```
#include "init.h"
#include "vic.h"
#include "extint.h"
#include "timer.h"

char *lcd_string;
int stevec;
int frekvenca;
char text[]="                              ";
int frekvence[5]={0,0,0,0,0};
int i;
int timer_runing=0;

void impulz()
{
  if(timer_runing==0)
  {
    stevec=0;

  T0TCR=counter_reset;
  T0TCR=counter_enable;
  timer_runing=1;
  }
  else
  {
    stevec++;
  }
  EXTINT=EINT1;
}

void cas()
{
  frekvenca=1*(stevec);
  text[3]=(frekvenca/1000)%10+'0';
  text[4]=(frekvenca/100)%10+'0';
  text[5]=(frekvenca/10)%10+'0';
  text[6]=(frekvenca/1)%10+'0';

  lcd_driver_1();

  T0TCR=counter_reset;
  T0IR=mr0_interrupt;
  VICVectAddr=0;
  timer_runing=0;
}

void start_up()
{
  init(60,cclk,0,0,0);
  stevec=0;

 lcd_string=text;

  text[0]='f';
  text[1]='=';
  text[2]=' ';
  text[3]=' ';
  text[4]=' ';
```

```
  text[5]=' ';
  text[6]=' ';
  text[7]='H';
  text[8]='z';

  lcd_driver_1();

  extint_init(P0_3_EINT1,0,0,EXTMODE1,EXTPOLAR1);

  int match[]={60000000,0,0,0};
  timer0_init(0,match,mr0i,timer);

  voidfuncptr funkcije[16]={impulz,cas,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
  int prekinitve[16]={eint1,timer0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

  vic_init(0,timer0|eint1,funkcije, prekinitve,0);
  i=0;
  timer_runing=0;

  while(1);
  }

void lcd_driver_1()
{
        int i;
        lcd_write_comm(DDRAM | 0x00);
        for(i = 0; i < 16; i++) lcd_write_data(lcd_string[i]);
        lcd_write_comm(DDRAM | 0x40);
        for(; i < 32; i++) lcd_write_data(lcd_string[i]);
}
```

3. Viri

http://fizika.uni-mb.si/urniki_datoteke/seminarji/sag.pdf
http://en.wikipedia.org/wiki/Guitar_harmonics
http://electronicdesign.com/Articles/Index.cfm?AD=1&ArticleID=6264
http://kitare.freehost386.com