PROJEKTNI ZADATAK

Simulacija audio podsistema kod Nintendo NES konzole

Nintendo NES (*Nintendo Entertainment System)* je 8-bitna video konzola, proizvedena 1985. godine u Japanu. Svojevremeno, kao najprodavanija i najpopularnija konzola u svetu, donela je određene revolucionarne inovacije u industriju video igara.

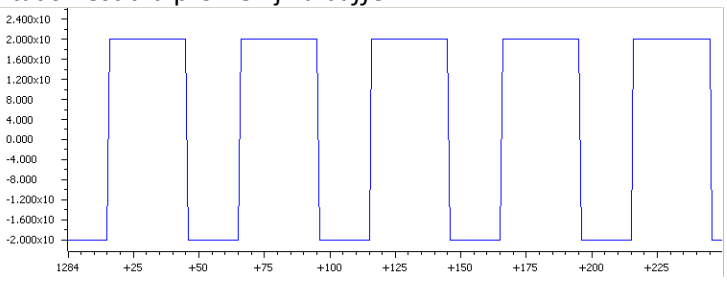
Kroz ovaj zadatak realizovaćete blok za generisanje audio signala, koji simulira rad audio podsistema kod NES konzole. Audio podsistem kod NES se sastoji iz četiri generatora prostih signala. Kombinovanjem zvuka nastalog koristeći ova četiri generatora dobijaju se sve melodije i zvučni efekti korišćeni u video igrama na ovoj konzoli.

1 Zadatak 1 – 1 bod

U okviru datoteke gen_signal.c, realizovati funkciju:

- `void gen_square(int n, float a, float f, float D, int ph, float buffer[])`

koja generiše povorku pravougaonih impulsa u trajanju od *n* odbiraka. Maksimalna amplituda signala zadata je sa *a,* normalizovana frekvencija signala sa *f*, a faktor ispunjenosti *D* izražen je u procentima. Parametar *ph* predstavlja fazni pomeraj (NAPOMENA: Fazni pomeraj je predstavljen sa brojem odbiraka, ne u radijanima). Rezultat smestiti u promenljivu *buffer.*



Slika 1 - Povorka pravougaonih impulsa u vremenskom domenu

Pomoću realizovane funkcije generisati:

- povorku pravougaonih impulsa amplitude 1, frekvencije 0,02, ispunjenosti 50.00%, faznog pomeraja 25 i trajanja 1000 odbiraka.

Signal prikazati u vremenskom i frekventnom domenu uz pomoć alata unutar *Code Composer Studio* razvojnog okruženja (*Tools->Graph*).

Očekivani izlaz iz zadatka:

- datoteke sa prikazom dva signala u vremenskom i frekventnom domenu imenovane kao: *Zadatak1_time.bmp, Zadatak1_freq.bmp*

2 Zadatak 2 – Generisanje belog šuma (2 boda)

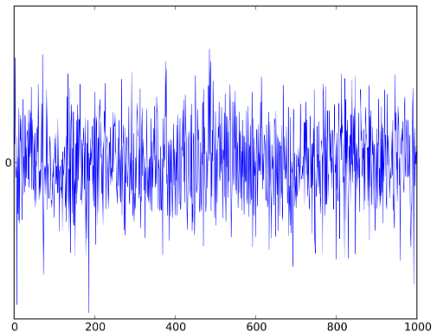
U okviru datoteke gen_signal.c, realizovati funkciju:

- `void gen_white_noise(int n, float a, float buffer[])`

koja generiše beli šum (signal sa uniformnom raspodelom energije u spektru). Maksimalna amplituda signala zadata je sa *a,* trajanje signala sa *n*. Rezultat smestiti u promenljivu *buffer.*
Beli šum, drugim rečima predstavlja povorku slučajnih brojeva, sa uniformnom raspodelom. Za generisanje slučajnih brojeva koristiti stadnardnu funkciju:

- `int rand (void);`

Funkcija *rand* vraća slučajan ceo broj u opsegu [0, RAND_MAX]. Voditi računa da izlazni signal treba da uključi i negativne brojeve, odnosno da opseg vrednosti unutar signala bude [-a, a].



Slika 2 - Beli šum u vremenskom domenu

Pomoću realizovane funkcije generisati:

- signal amplitude 0.6, trajanja 1000 odbiraka.

Signal prikazati u vremenskom i frekventnom domenu uz pomoć alata unutar *Code Composer Studio* razvojnog okruženja (*Tools->Graph*).

Očekivani izlaz iz zadatka:

- datoteke sa prikazom dva signala u vremenskom i frekventnom domenu imenovane kao: *Zadatak2_time.bmp, Zadatak2_freq.bmp*

3 Zadatak 3 – 2 boda

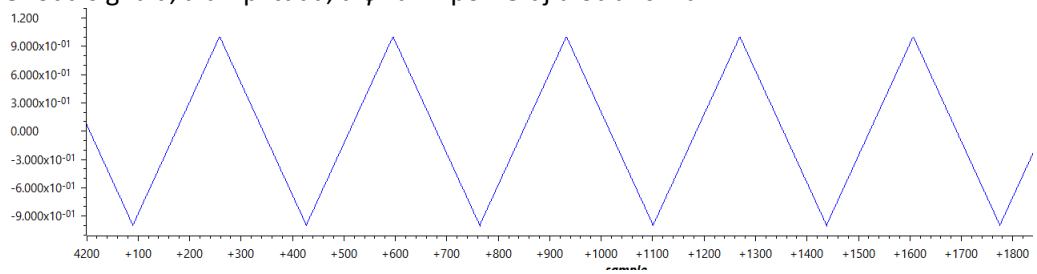
U okviru datoteke gen_signal.c, realizovati funkciju:

- `void gen_triangle(int n, float a, float f, int ph, float buffer[])`

koja generiše povorku trouglastih impulsa u trajanju od *n* odbiraka. Maksimalna amplituda signala zadata je sa *a,* normalizovana frekvencija signala sa *f*, a fazni pomeraj izražen u broju odbiraka sa *ph*. Rezultat smestiti u promenljivu *buffer.*
Generisanje trouglastog signala se može izvršiti koristeći sledeću jednačinu:

$$y(n) = \frac{4 \cdot a}{T} \cdot \left(\left| (n + \varphi) \,\%\, T - \frac{T}{2} \right| - \frac{T}{4} \right)$$

gde je *T* perioda signala, *a* amplituda, a *φ* fazni pomeraj u odbircima.



Slika 3 - Prikaz trouglastog signala u vremenskom domenu

Pomoću realizovane funkcije generisati:

- povorku trouglova amplitude 1, frekvencije 0,02, faznog pomeraja 25 i trajanja 1000 odbiraka.

Signal prikazati u vremenskom i frekventnom domenu uz pomoć alata unutar *Code Composer Studio* razvojnog okruženja (*Tools->Graph*).

Očekivani izlaz iz zadatka:

- datoteke sa prikazom dva signala u vremenskom i frekventnom domenu imenovane kao: *Zadatak3_time.bmp, Zadatak3_freq.bmp*

4 Zadatak 4 – Simulacija audio podsistema kod NES (5 bodova)

Realizovati blok za simulaciju jedinice za obradu zvuka kod NES konzola. Blok se sastoji iz **4 generatora signala:**

- Generator **povorke trouglastih impulsa**
- Generator **belog šuma**
- Generator **povorke pravougaonih impulsa, sa faktorom ispunjenosti 30%**
- Generator **povorke pravougaonih impulsa, sa faktorom ispunjenosti 50%**

Izlazni signal se formira sabiranjem tonova dobijenih od svakog od generatora.

Blok za sintezu zvučnog signala je delimično implementiran.

U datoteci *notes.h* data je enumeracija kojom su predstavljeni tonovi, kao i funkcije za preslikavanje tona u frekvenciju i tona u string sa nazivom.

- `float note_to_freq(notes_t note)`
- `const char* note_to_string(notes_t note)`

U datoteci *super_mario_theme.h* data je kompozicija „*Overworld theme*“, tema iz čuvene igrice Super Mario iz 1985. godine, predstavljena notnim zapisom datim u *notes.h*. Svaki ton je opisan sa 3 polja:

- *time* – trenutak kada je potrebno početi sviranje tona
- *duration* – trajanje tona
- *note* – koji ton je potrebno odsvirati

Trajanje i početak tona su predstavljeni rednim brojem bloka. Tempo sviranja određen je veličinom bloka BLOCK_SIZE.

U datoteci *adsr.h* data je funkcija koja predstavlja ADSR blok (*attack, decay, sustain, release*), koji služi za uobličavanje tonova tako što vrši modifikaciju amplitude signala.

- `void ADSR(float buffer[], Int16 n, Int16 current_offset, Int16 tone_duration)`
  - *buffer – ulazno/izlazni niz*
  - *n – broj odbiraka*
  - *a - amplituda*
  - *current_offset– fazni pomeraj (broj odbiraka od početka sviranja tona*
  - *duration – trajanje tona u okviru buffer-a*

Realizacija glavne petlje za generisanje tonova data je u main.c.

Neophodno je realizovati unutrašnjost petlje prateći uputstva u komentarima. Potrebno je za svaki generator:

- Proveriti da li je potrebno odsvirati ton u datom trenutku
- Ukoliko je potrebno, generisati odgovarajući signal zadate frekvencije, izračunatog faznog pomeraja i amplitude 1.0
- Generisani signal propustiti kroz ADSR jedinicu
- Izlazne signale potom kvantizovati, i to:
  - Trouglaste impulse kvantizovati sa 15 bita, i potom klipovati sa 14 bita
  - Beli šum kvantizovati sa 12 bita
  - Pravougaone signale kvantizovati sa 14 bita
- Dobijeni signal dodati na trenutni sadržaj izlaznog bafera (*outputBuffer*)

Za jedan odabran ton prikazati signal u vremenskom i frekventnom domenu odmah nakon generisanja, nakon primene ADSR i nakon kvantizacije/klipovanja.

Izračunati SNR nakon kvantizacije i klipovanja signala za odabrani ton.

Očekivani izlaz:

- datoteka „super_mario.wav“ koja sadrži sintetizovanu pesmu.
- datoteke sa prikazom signala u vremenskom i frekventnom domenu imenovane kao: *Zadatak4gen_time.bmp, Zadatak4gen_freq.bmp, Zadatak4adsr_time.bmp, Zadatak4adsr_freq.bmp*, *Zadatak4clip_time.bmp, Zadatak4clip_freq.bmp*
- datoteka tone.txt koja sadrži sledeće informacije: ton koji je prikazan na slikama, njegova izvorna frekvencija, vrednost SNR koja je izračunata: