Itasdi

Innovative Teaching Approaches in development of Software Designed Instrumentation and its application in real-time systems

Praktikum iz merno-akvizicionih sistema

Co-funded by the Erasmus+ Programme of the European Union

Itasdi

Innovative Teaching Approaches in development of Software Designed Instrumentation and its application in real-time systems

Faculty of Technical Sciences

Ss. Cyril and Methodius University
Faculty of Electrical Engineering and Information Technologies

Zagreb University of Applied Sciences

School of Electrical Engineering
University of Belgrade

Faculty of Physics
Warsaw University of Technology

Co-funded by the Erasmus+ Programme of the European Union

Lekcija 5

Generisanje zvučnog signala. Trigerovana kontinualna akvizicija. Primer feedback aplikacije

III deo

1. Cilj vežbe

Cilj vežbe je da studente:

- upozna sa *Labview* funkcijama za generisanje zvučnih signala
- upozna sa konceptom trigerovane akvizicije u *NI Labiew* paketu pomoću DAQmx VIs
- osposobi ih za samostalno kreiranje *feedback Labview* aplikacije koja omogućava generisanje digitalnog upravljanja kada signal prikupljen na analognom ulazu zadovoljava zadati uslov.

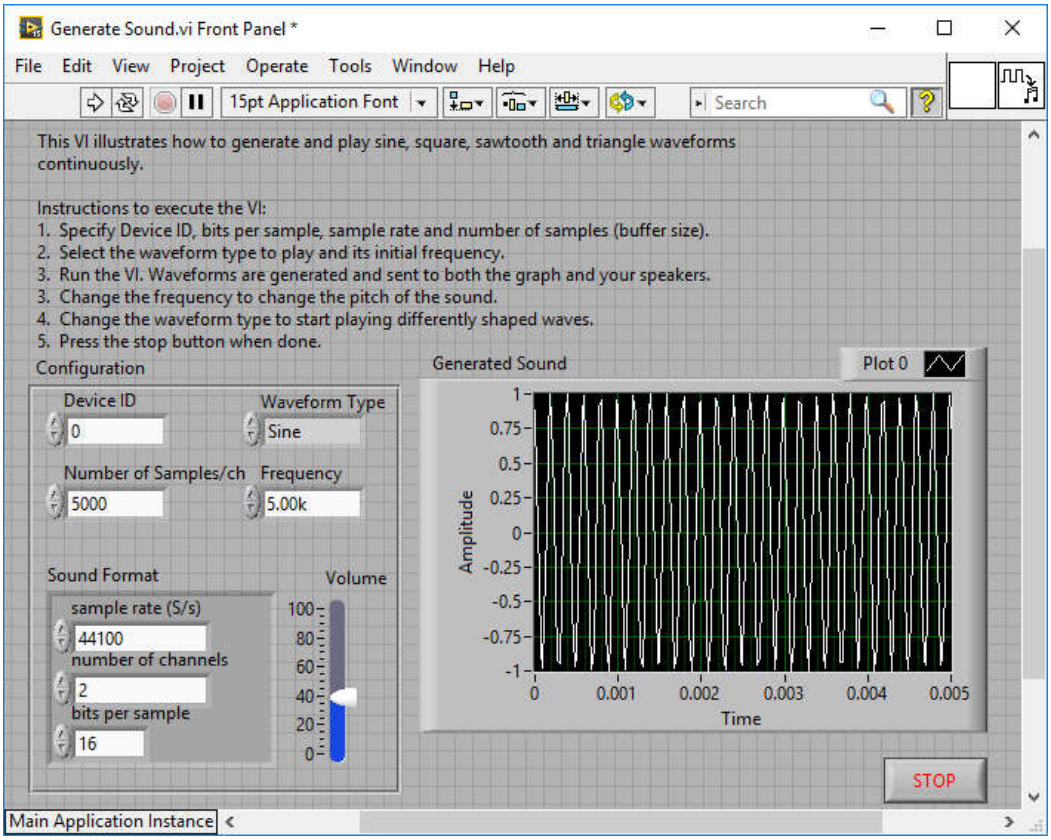
2. Oprema

- Računar sa instaliranim ***NI Labview*** softverskim paketom i ***NI DAQ-mx*** drajverima.
- *NI DAQ* uređaj
- Otpornici, dioda, mikrofon, slušalice

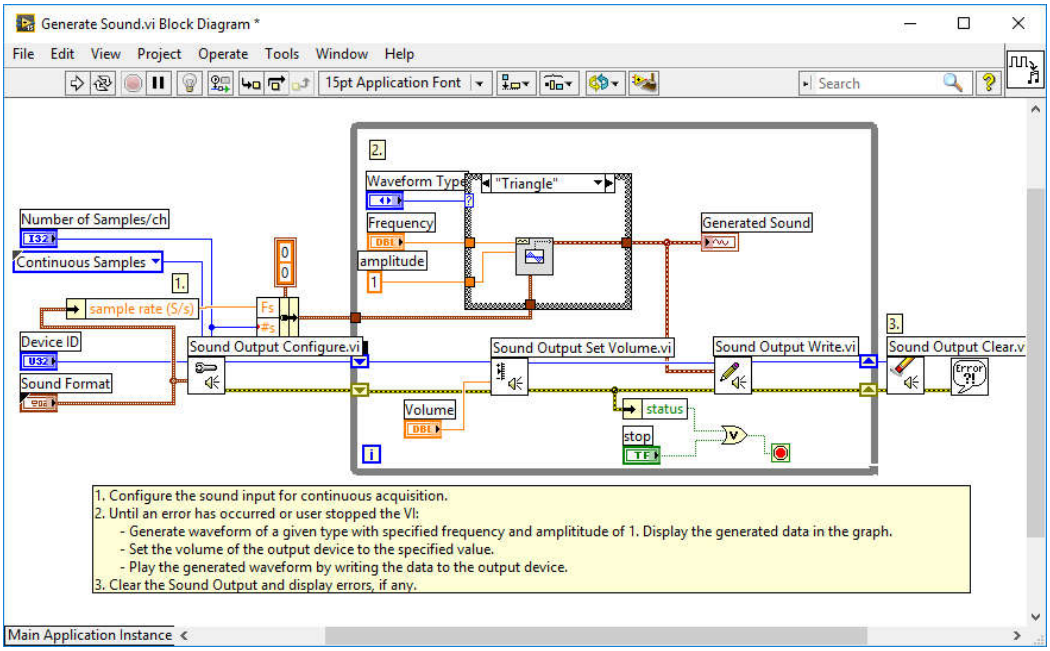
Napomena: Nakon instalacije *NI Labview Development* ili *Professional* okruženja, treba dodati i *NI DAQ-mx* drajvere. Odgovarajuću verziju drajvera je moguće download-ovati sa:
http://www.ni.com/en-rs/support/downloads/drivers/download.ni-daqmx.html#291872

5.1. Generisanje zvučnog signala

1. Otvoriti primer *Generate Sound.vi* čiji su *Front Panel* i *Block Diagram* prikazani na Sl. 1 i 2, respektivno. Proučiti objašnjenje rada programa u Block *Diagram*-u. Otvoriti i program *Cont Acq&Graph Voltage-Int Clk_DBL.vi* i uočiti sličnosti i razlike.
   Uočiti da je redni broj uredjaja, tj. zvučne kartice 0 po *default*-u (kontrola *Device ID* na *Front Panel*-u).
2. Pokrenuti program i „slušati“ različite frekvencije i oblike signala (menjati vrednosti kontrola *Frequency* i *Waveform Type*).



Sl. 1: *Front Panel* programa *Generate Sound.vi*

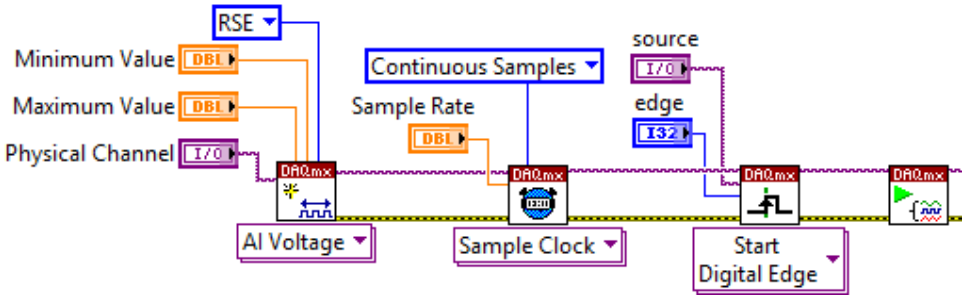


Sl. 2: *Block Diagram* programa *Generate Sound.vi*

5.2 Trigerovana kontinualna akvizicija

Trigerovana akvizicija je tip akvizicije u kome prikupljanje odbiraka u bafer započinje kada je ispunjen odredjen uslov (tzv. „triger“): npr. kada nivo ili nagib analognog ulaznog signala ima zadatu vrednost, ili kada je detektovana uzlazna ili silazna ivica digitalnog ulaznog signala i sl. Pri ovom tipu akvizicije, moguće je zadati broj odbiraka koji će se prikupiti pre ili posle „triger“ signala, a moguće je i zadati kontinualnu akviziciju počev od pojave „triger“ signala. U ovom vežbanju će biti prikazana trigerovana kontinualna akvizicija u kojoj je „triger“ pojava silazne ivice digitalnog kanala PFI0.

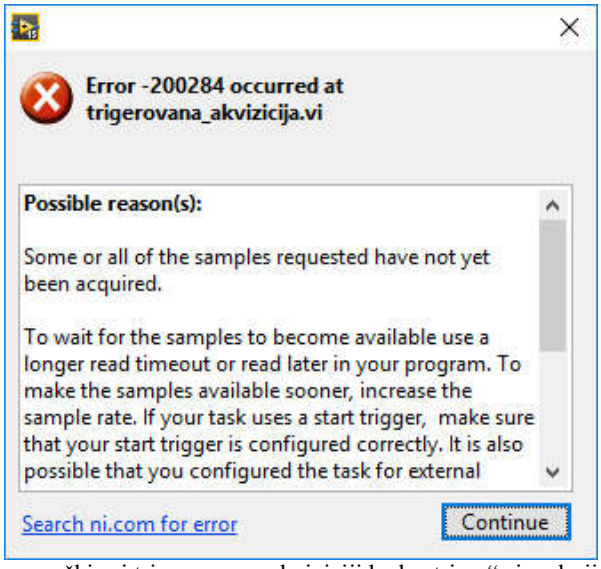
1. Priključiti *NI DAQ* uređaj preko USB-a na računar.
2. Kontrolu *Physical Channel* na *Front Panel*-u primera ***Cont Acq&Graph Voltage-Int Clk_DBL.vi*** podesiti da snima analogni ulaz *ai0* (selektovati vrednost *DevN/ai0*, gde je *device number NI DAQ* uređaja).
3. Pokrenuti primer ***Cont Acq&Graph Voltage-Int Clk_DBL.vi.*** Na grafiku će se pojaviti signal koji postoji u okolini pošto kanal *ai0* nije nigde povezan. Zaustaviti izvršavanje programa.
4. Uočiti u *Block Diagram*-u primera ***Cont Acq&Graph Voltage-Int Clk_DBL.vi*** da je polimorfna funkcija *DAQmx Read* podešena na *Analog 1D DBL 1Chan Nsamp,* tj. podešena je da pri svakoj iteraciji *While* petlje čita iz bafera *Nsamp* odbiraka sa jednog analognog kanala (u ovom konkretnom slučaju je *Samples to Read*=1000).
5. Žicom povezati PFI0 pin sa +5 V pinom.
6. Primer ***Cont Acq&Graph Voltage-Int Clk_DBL.vi*** dopuniti funkcijom *Measurement I/O>>NI DAQmx>>DAQmx Start Trigger* kao na Sl. 3. U padajućem meniju ove polimorfne funkcije izabrati *Start>>Digital Edge*. Na odgovarajućim ulazima ove funkcije napraviti dve kontrole *source* i *edge*.



Sl. 3: Dopuna programa ***Cont Acq&Graph Voltage-Int Clk_DBL.vi*** za konfigurisanje trigerovane akvizicije

7. Podesiti vrednost kontrole *source* na *Front Panel*-u na /DevN/PFI0 (gde je N redni broj NI DAQ uređaja). Podesiti vrednost kontrole *edge* na *Front Panel*-u na „Falling“.
8. Pokrenuti izvršavanje modifikovanog programa. Uočiti da se na grafiku ne pojavljuju odbirci. Nakon 10 s će se pojaviti poruka o greški kao na Sl. 4. Program će završiti izvršavanje nakon pritiska na dugme *Continue*.

NAPOMENA: vreme čekanja „triger“ signala je podešeno na 10 s pomoću konstante *timeout* u *While* petlji. Menjati vreme čekanja „trigera“ i uočiti razliku.

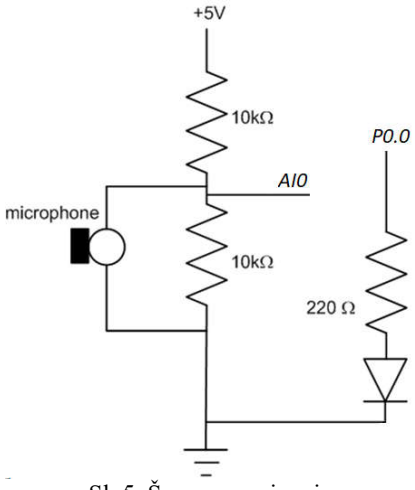


Sl. 4: Poruka o greški pri trigerovanog akviziciji kada „triger“ signal nije detektovan

9. Ponovo pokrenuti izvršavanje programa, ali sada žicu (koja je priključena s jedne strane na pin PFI0, a s druge strane na +5 V) prebaciti sa +5 V na GND (ovo se mora uraditi u okviru *timeout* vremena). Uočiti da će nakon ovog prelaska sa visokog na niski nivo napona, tj. nakon detekcije padajuće ivice digitalnog signala, početi kontinualna analogna akvizicija.
10. Sačuvati modifikovani primer kao *05_01_trigerovana_akvizicija.vi*.

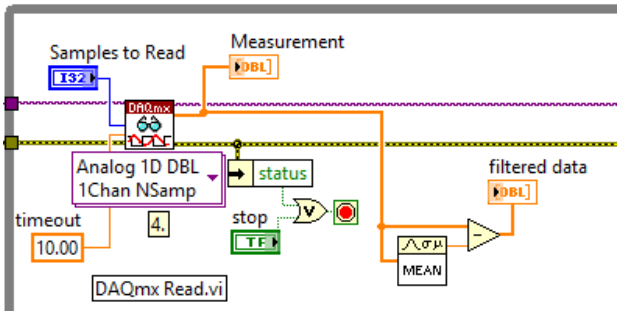
5.3 Primer *feedback* aplikacije

1. **Isključiti *NI DAQ* uređaj iz računara.** Povezati kolo kao na Sl. 5 na *NI DAQ* uređaj. Mikrofon povezati direktno u konektore *AI0* i *GND*, a ne preko protoborda (zbog širma koji ne može da se priključi na protobord). Za ostale komponente koristiti i protobord.



Sl. 5: Šema povezivanja

2. Žicom povezati PFI0 pin sa +5 V pinom. **Uključiti *NI DAQ* uređaj u računar.**
3. Pokrenuti aplikaciju *05_01_trigerovana_akvizicija.vi* i posmatrati izgled analognog signala *ai0* na grafiku kada se govori u mikrofon. Uočiti promene u amplitudi i frekvenciji signala pri promeni jačine i frekvencije glasa. Uočiti da postoji jednosmerna komponenta signala.
4. Odkloniti jednosmernu komponentu analognog signala tako što će se od njega oduzeti njegova srednja vrednost. Srednju vrednost niza odbiraka pročitanih u jednoj iteraciji *While* petlje izračunati pomoću funkcije *Mathematics>>Probability & Statistics>>Mean*, Sl. 6.



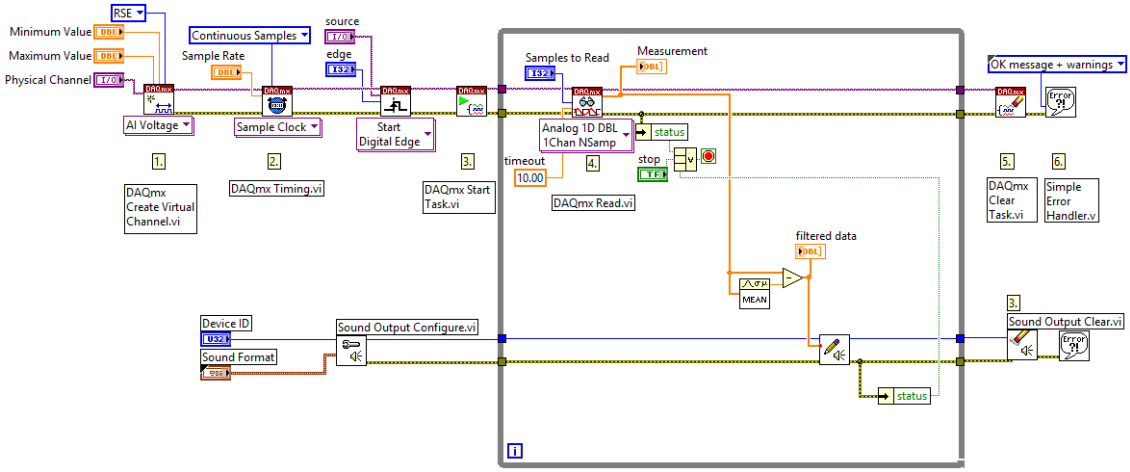
Sl. 6: Otklanjanje jednosmerne komponente

5.3.1 Zadatak

Polazeći od programa *05_01_trigerovana_akvizicija.vi*, dizajnirati aplikaciju koja omogućava:

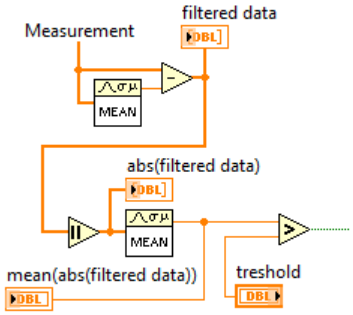
a. čitanje analognog signala sa mikrofona *ai0* (mikrofon je priključen u kolo sa Sl. 5)
b. prikupljeni analogni signal *ai0* sa mikrofona nakon elementarne obrade (otklanjanje jednosmerne komponente) emituje na slušalicama
c. da se upali LED (sa Sl. 5) ako anvelopa analognog signala sa mikrofona *ai0* pređe zadatu vrednost praga.

1. Kombinovanjem programa *05_01_trigerovana_akvizicija.vi* i programa *Generate Sound.vi* napraviti aplikaciju koja omogućava da se pročitani analogni signal sa kanala *ai0* emituje na zvučniku, Sl. 7.



Sl. 7: Aplikacija koja omogućava trigerovanu kontinualnu akviziciju i emitovanje analognog signala na zvučnik

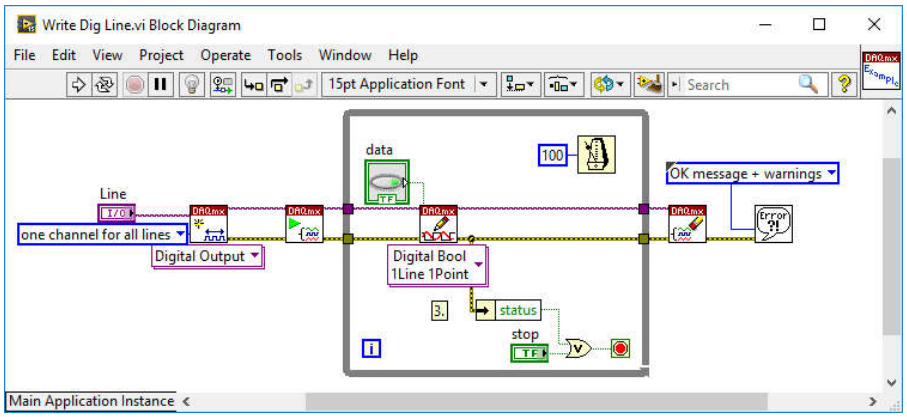
11. U *Sound Format* kontroli podesiti da frekvencija generisanja zvuka bude ista kao za akviziciju: 10 000 Hz. Pokrenuti izvršavanje aplikacije. Pričati u mikrofon, a slušalice će emitovati glas.
    NAPOMENA: Pokušati da se na slušalice direktno, bez korekcije jednosmerne komponente, prosledi signal sa *Measurement* indikatora. Šta će se sada čuti na slušalicama i zašto?
12. Proceniti anvelopu analognog signala sa mikrofona *ai0* na sledeći način, Sl. 8:
    a. korigovati jednosmernu komponentu
    b. ispraviti signal korišćenjem funkcije za određivanje absolutne vrednosti *Programming>>Numeric>>Absolute Value*
    c. odrediti srednju vrednost ispravljenog signala pomoću *Mathematics>>Probability & Statistics>>Mean*.
13. Uporediti procenjenu anvelopu signala za pragom koji će se zadavati pomoću kontrole *treshold*, Sl. 8, na *Front Panel*-u. Rezultat poređenja vrednosti anvelope sa pragom je logička promenljiva koja se može iskoristiti kao upravljački signal za paljenje/gašenje LED-a.



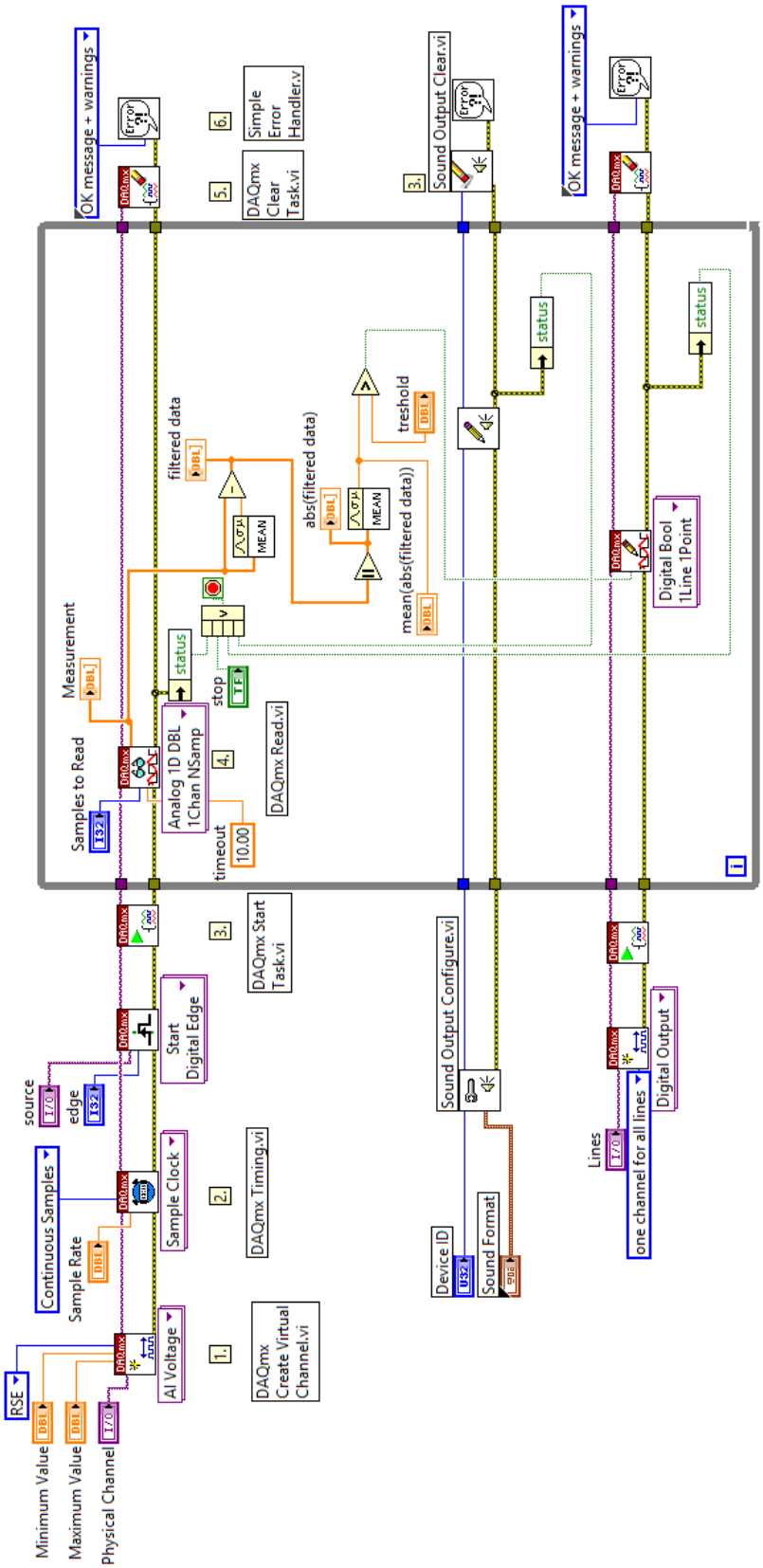
Sl. 8: Procena anvelope analognog signala sa mikrofona *ai0*

Pitanja: Da li bi mogao analogni signal sa mikrofona *ai0* odmah nakon korekcije jednosmerne komponente da se upotrebi kao upravljački signal za paljenje/gašenje LED-a? (Pomoć: LED ne može da se beskonačno brzo pali/gasi) Zašto je neophodno ispraviti signal? (Pomoć: kolika je srednja vrednost neispravljenog signala kome je prethodno otklonjena jednosmerna komponenta?)

14. Otvoriti program *Write Dig Line.vi*, Sl. 9. Kombinovanjem programa sa Sl. 7, 8 i 9, napraviti aplikaciju koja pali/gasi LED u zavisnosti od nivoa signala na mikrofonu, tzv. *feedback* aplikaciju, Sl. 10.



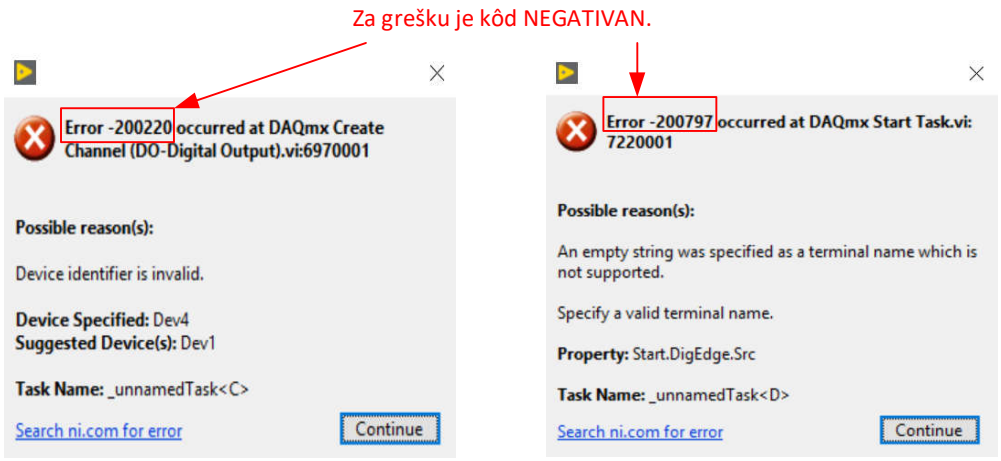
Sl. 9: *Block Diagram* programa *Write Dig Line.vi*



Sl. 10: Primer realizacije *feedback* aplikacije

15. Sačuvati primer kao *05_02_primer_feedbacka.vi*.
16. Isključiti *NI DAQ* uređaj iz računara. Pokrenuti program. Koliko puta će se prozor sa greškama pojaviti i zašto? Koje opcije će biti ponuđene na prozoru: nastavak izvršavanja (*Continue*) ili/momentalno zaustavljanje (*Stop*)?

    NAPOMENA: Na Sl. 11. su prikazani prozori sa greškama koji se pojavljuju u tački 16. Obratiti pažnju da su kôdovi greški (***Errors***) **negativni** brojevi. U slučaju samo upozorenja (***Warnings***) kôdovi su **pozitivni** brojevi. Logička vrednost **status** promenljive u *error* klasteru je za greške **FALSE**, a za upozorenja **TRUE**.



Sl. 11: Kodovi greški (***Errors***) su **negativni** brojevi

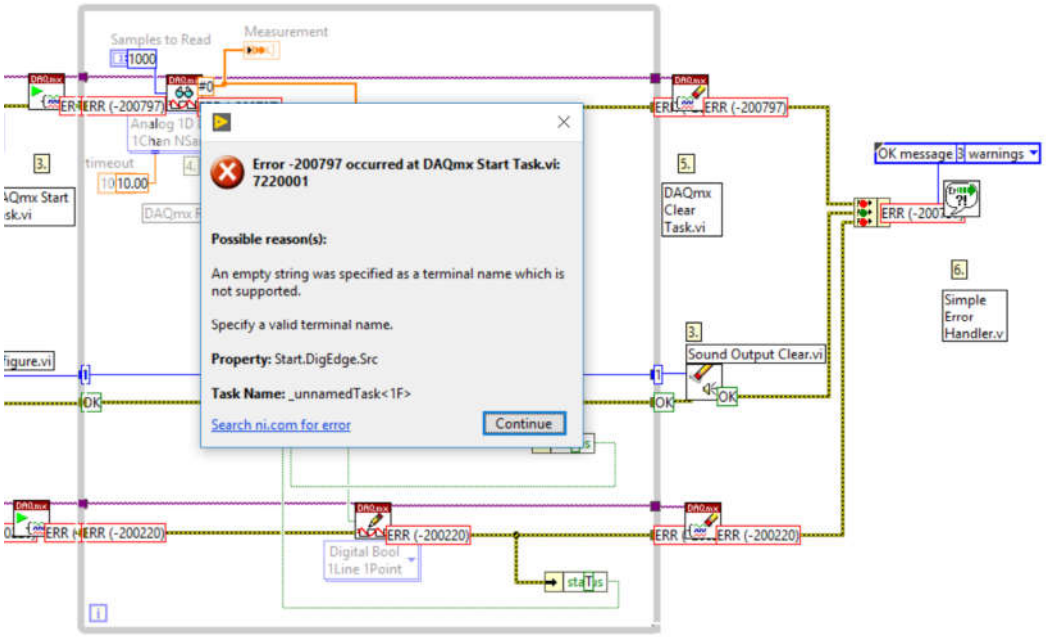
17. Za funkciju *Simple Error Handler*  na ulazu „*type of dialog*“ se može povezati jedna od sledećih vrednosti *enum* konstante (ili kontrole):
    a. *no dialog* – u slučaju greške ili upozorenja se neće prikazati prozor
    b. *OK message* – u slučaju greške će se prikazati prozor sa *Continue* dugmetom
    c. *continue or stop message* – u slučaju greške će se prikazati prozor sa *Continue* i *Stop* dugmetom
    d. *OK message + warnings* – u slučaju greške ili upozorenja će se prikazati prozor sa *Continue* dugmetom
    e. *continue/stop + warnings* – u slučaju greške ili upozorenja će se prikazati prozor sa *Continue* i *Stop* dugmetom

Sl. 12. Opcije ulaza „*type of dialog*“ funkcije *Simple Error Handler*

Isprobati sve navedene opcije *Simple Error Handler* funkcije kada je *NI DAQ* uređaj isključen iz računara. Pritiskom na dugme *Continue* u prozoru greške, izvršavanje programa se nastavlja, a pritiskom na dugme *Stop* u prozoru greške, izvršavanje programa se odmah zaustavlja.

18. Funkcija *Merge Errors*  iz palete ***Functions»Programming»Dialog & User Interfaces*** omogućava da se objedine rezultati svih *error* klastera u programu u jedan izlazni klaster koji se potom može dovesti na odgovarajući ulaz *Simple Error Handler* funkcije. Modifikovati primer *05_02_primer_feedbacka.vi*. tako da se svi error klasteri objedine pomoću funkcije *Merge Errors*. Pomoću opcije *Highlight Execution*  posmatrati izvršavanje kôda kada je *NI DAQ* uređaj isključen iz računara, Sl. 13. Uočiti da ne postoji greška ili upozorenje zbog generisanja zvuka (zeleni ulaz u *Merge Errors* funkciju u môdu za debagovanje) već da postoje samo greške usled neprikljućenosti *NI DAQ* uređaja (dva crvena ulaza u *Merge Errors* funkciju u môdu za debagovanje). Koji je kôd greške na izlazu *Merge Errors* funkcije i zašto?

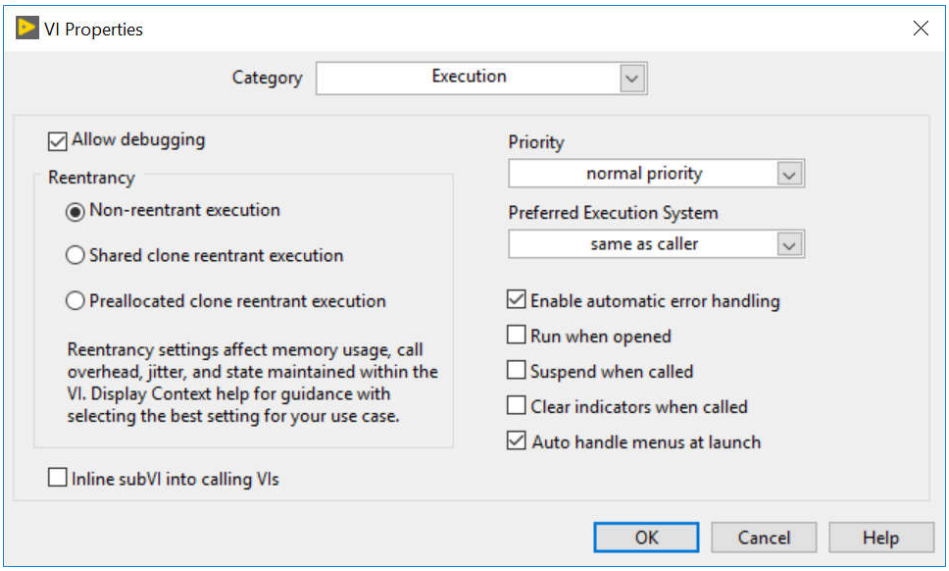
NAPOMENA: Ako više *error* klastera „predaje grešku" *Merge Errors* funkciji, ona će na svom izlazu dati kôd prve greške (prema redosledu povezivanja ulaznih priključaka), tj. NEĆE SPAJATI greške. U slučaju da nema „predaje grešaka", *Merge Errors* funkcija će na svom izlazu imati kôd prvog upozorenja (prema redosledu povezivanja ulaznih priključaka).



Sl. 13. Debagovanje kôda kada je *NI DAQ* uređaj isključen iz računara, a primenjena funkcija za objedinjavanje grešaka i/ili upozorenja

19. Sačuvati primer kao *05_03_primer_feedbacka_merge_errors.vi*.
20. U tačkama 16-19 je demonstriran manuelni način za prikaz greški/upozorenja (*Manual Error Handling*). Izbrisati sada funkciju *Simple Error Handler* iz programskog kôda primera *05_03_primer_feedbacka_merge_errors.vi* i u opciji ***File»VI Properties»Execution*** čekirati opciju *Enable automatic error handling*, Sl. 14. Ovim je podešeno automatsko prikazivanje greški/upozorenja (*Automatic Error Handling*) u tekućem programu. Takođe, proveriti i da li je u opciji ***Tools»Options»Block Diagram»Error Handling*** selektovana opcija koja omogućava automatsku podršku.

21. Pokrenuti izvršavanje programa. Uočiti da iako ne postoji u kôdu funkcija *Simple Error Handler*, program će prikazati prozor za grešku i blinkanjem će ukazati na čvor u program u kome je greška (u ovom primeru će to biti čvor *Merge Errors*).



Sl. 14. Podešavanje za automatsko prikazivanje grešaka/upozorenja

NAPOMENA: Ako program ne koristi neku od funkcija za prikaz greški/upozorenja (*Error Handler* funkcije) moguće je podesiti da okruženje automatski prikazuje prozor sa greškom/upozorenjem.