SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 1042

# DIREKTNA DIGITALNA SINTEZA SINUSNOG SIGNALA NA DSP PROCESORU

Tomislav Rac

Zagreb, siječanj 2010.

**Sadržaj**

# Uvod

Za digitalni prijenos signala često je bitno da odašiljać i prijemnik rade na istoj frekvencije, pa se zbog toga implementiraju lokalni generatori signala. U tu svrhu često se koriste sintetizatori frekvencije. To su generatori koji na svom izlazu generiraju najčešće sinusni signal stabilne frekvencije s velikom mogučnošću njenog podešavanja. Koriste se direktne, indirektne i hibridne metode za realizaciju sintetizatora frekvencije. U ovom radu opisana je i implementirana direktna digitalna sinteza (DDS, engl. *Direct Digital Synthesis*). To je metoda generiranja analognog signala pomoću digitalnih blokova podataka te referentnog takta frekvencije. Implementacija direktne digitalne sinteze izvršit će se na procesoru za digitalnu obradu signala (DSP, engl. *Digital signal processor)* ADSP-BF537 iz porodice Blackfin tvrtke Analog Devices.

# 1. Direktna digitalna sinteza

Postupci direktne digitalne sinteze poznati su još iz 70-tih godina, a nastali su iz potrebe za generiranjem signala vremenski nepromjenjive kvalitete u odnosu na analogne oscilatore koji su starenjem značajno mijenjali svojstva. Sklopovi za tu metodu najprije su se gradili s diskretnim komponentama, a zatim i integrirano, da bi se u današnje doba, pojavom procesora za digitalnu obradu signala, stvorili uvjeti za programsku implementaciju.

Osim direktne metode sinteze signala kojom se mogu generirati signali do 1 GHz s finim podešavanjem, postoji i indirektna metoda koja koristi PLL (engl. *Phase-Locked Loop)* sintezu za frekvencije do 10 GHz, te hibridna metoda koja generira signale frekvencije do 40 GHz koristeći dobre osobine PLL i DDS sinteze. Tehnika direktne digitalne sinteze danas je postala alternativa PLL tehnici u realizaciji sintetizatora. DDS je determinističke prirode budući da sintezirani signal nastaje iz zadane digitalne definicije.

Uporabom programskih posupaka, metodu izravne digitalne sinteze moguće je modificirati i prilagoditi specifičnim zahtjevima korisnika.

## 1.1. Princip rada direktne digitalne sinteze

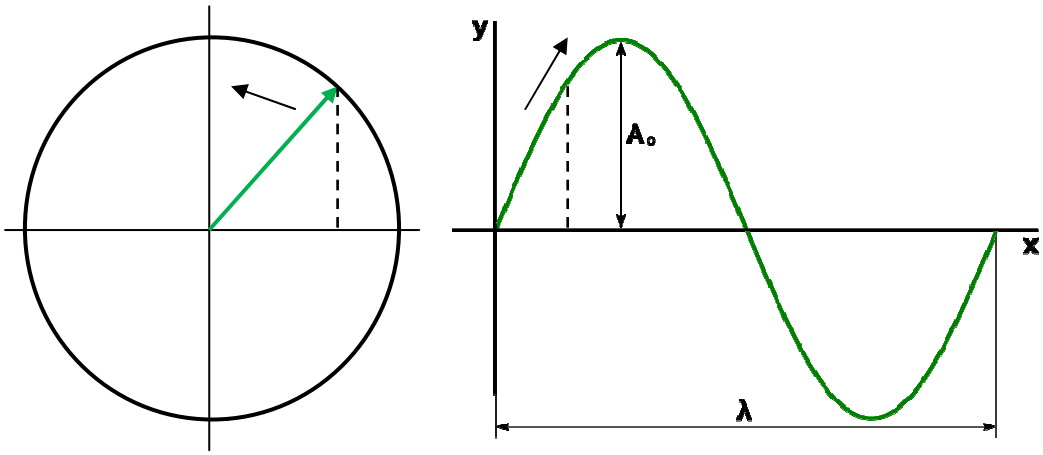
Direktna digitalna sinteza je tehnika korištenja digitalnih procesnih blokova podataka u svrhu generiranja frekvencijski i fazno promjenjivog izlaznog signala pomoću jednog referentnog takta frekvencije. Referentnom frekvencijom determinirana je i trenutna faza signala te brzina promjene faze (korak faze).

Osnovu direktne digitalne sinteze čini *fazni akumulator*. To je aritmetički registar koji obavlja funkciju diskretne matematičke integracije:

$$\phi(i) = \phi(i-1) + \Delta p \qquad (1)$$

Željeni izlazni signal je najčešće periodična sinusna funkcija sa periodom $2\pi$. Nakon faze od $2\pi$ funkcija se ponavlja pa se funkcija promatra samo u rasponu od $0$ do $2\pi$. To nam omogučuje da se fazni akumulator digitalno implementira kao *fazna kružnica* (engl. *Phase wheel)*.

Sinusna oscilacija se može zamisliti kao vektor koji se okreće oko sredine koordinatnog sustava te iscrtava faznu kružnicu. Generiranje sinusoide prikazano je na Slici 1. Svaka točka na kružnici odgovara ekvivalentnoj točki sinusne funkcije. Rotacijom vektora generira se odgovarajuća sinusoida, pri čemu jedan obilazak kružnice konstantnom brzinom rezultira jednu periodu sinusoide.



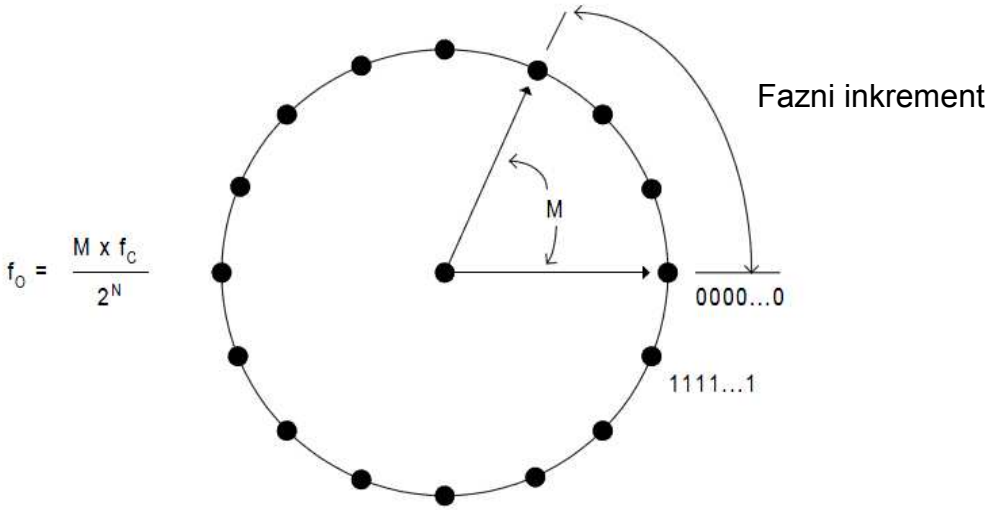

**Slika 1. Generiranje sinusoide faznom kružnicom**

Fazni akumulator sadrži informaciju ekvivalentnu rotaciji vektora oko fazne kružnice. Broj diskretnih vrijednosti faze na faznoj kružnici određen je brojem bita faznog akumulatora N, a vrijednost može biti od 0 do N-1. Odnos rezlucije akumulatora i broja točaka fazne kružnice prikazan je na Slici 2.

| N | Broj točaka |
|---|---|
| 8 | 256 |
| 12 | 4096 |
| 16 | 65535 |
| 20 | 1048576 |
| 24 | 16777216 |
| 28 | 268435456 |
| 32 | 4294967296 |
| 48 | 281474976710656 |

**Slika 2. Odnos rezolucije akumulatora i broja točaka fazne kružnice**

Za konstantnu frekvenciju, fazni inkrement tj. frekvencijska kontrolna riječ M će biti konstanta. Promjena faze signala određene frekvencije je aritmetička progresija koju odlično generiraju digitalni akumulatori.Različite izlazne frekvencije postižu se promjenom vrijednosti faznog inkrementa M (Slika 3). Za 28-bitni fazni akumulator i vrijednost inkrementa M 0000....0001 potrebno je $2^{28}$ taktova (inkrementa) da se dogodi preljev akumulatora. Ako se vrijednost M promjeni na 0111....1111 potrebno je 2 takta (minimalno potrebno zbog Nyquistovog teorema).

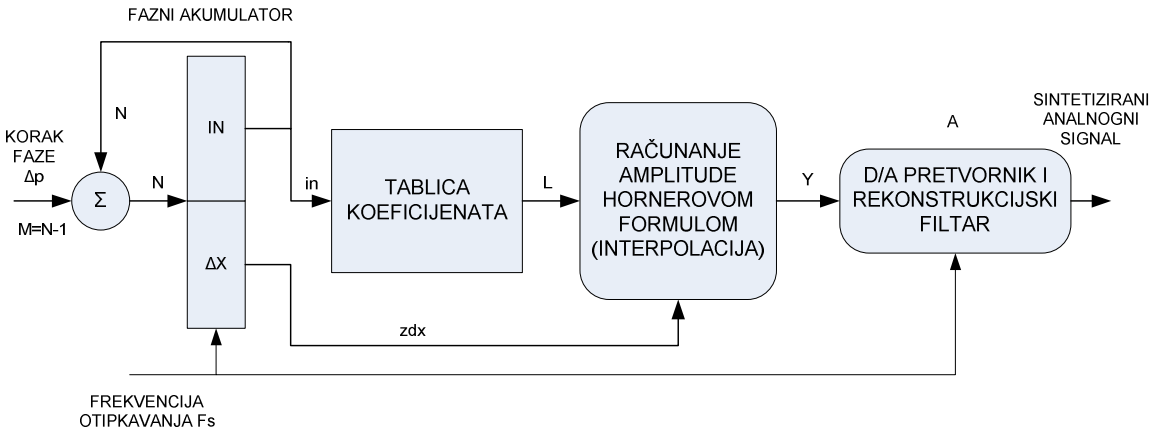



**Slika 3. Fazna kružnica**

Izlaz faznog akumulatora je linearan pa ne može biti direktno iskorišten za generiranje sinusnog signala. Zato se koristi *tablica uzoraka* zadane periodičke funkcije. Na osnovu stanja faznog akumulatora indirektno se adresiraju uzorci u tablici i njihova se vrijednost digitalno-analognom pretvorbom te rekonstrukcijskim filtrom vraća u domenu analognog signala. Većina DDS arhitektura koristi simetriju sinusnih signala pa se u tablicu uzoraka zapisuje samo prvi kvadrant signala. Ostala tri kvadranta jednostavnim transformacijama rekonstruiraju se iz prvog. Drugi kvadrant rekonstruira se obrnutim slijedom čitanja tablice, treći dodavanjem negativnog predznaka, a četvrti dodavanjem negativnog predznaka i obrnutim redosljedom čitanja uzoraka iz tablice. Na taj se način broj uzoraka u tablici prividno povećava četiri puta, ali bez potrebe za dodatnom memorijom.Koriste se i druge metode za kompresiju podataka u memoriji koje se zasnivaju na Taylorovoj aproksimaciji i na glatkoći i monotonosti sinusne funkcije. U implementaciji DDS-a

koja je obrađena u ovom radu u tablicu uzoraka pohranjivat će se koeficijenti koji će preko Hornerove formule dati određenu amplitudu. Također, zbog praktičnih razloga, za adresiranje uzoraka iz tablice koristi se samo dio najvažnijih bitova akumulatora. Ostatak bitova akumulatora koristi se za kasniju interpolaciju signala.

Shema postupka direktne digitalne sinteze prikazan je na slici 4.




**Slika 4. Blok shema postupka direktne digitalne sinteze signala**

Broj M predstavlja širinu riječi kojom se zadaje željena frekvencija signala. In je informacija širine IN koja predstavlja adresu koeficijenata u tablici. L je rezolucija koeficijenata. To je variabilan broj jer je rezolucija svakog koeficijenta različita i određena je kvantizacijom. U Hornerovu forumulu za računanje amplitude ulazi i frakcijski dio, ostatak faznog akumulatora. Izračunata amplituda širine je Y i to je format izlaza. Broj A predstavlja razlučivost D/A pretvornika. Najvišu frekvenciju izlaznog sinteziranog signala određuje frekvencija otipkavanja $F_S$ koja utječe i na razlučivost namještanja frekvencije u D/A pretvorniku.

Vrijednost u frekvencijskom registru širine M je broj $\Delta_P$ koji se dodaje vrijednosti faznog akumulatora širine N iz predhodnog koraka. Fazni akumulator opisuje trenutnu fazu signala $\Phi_i$ i dijelom adresira tablicu uzoraka periodične funkcije.

Frekvencijskim registrom namještena željena frekvencija $f_{IZL}$ sinteziranog signala determinirana je korakom faze $\Delta_P$ i frekvencijom otipkavanja $F_S$.

$$f_{IZL} = \Delta_P \cdot \frac{F_S}{2^N} \quad (2)$$

Minimalna frekvencija $f_{MIN}$ izlaznog sinteziranog signala dobiva se uz $\Delta_P$=1. To znači da će fazni akumulator proći kroz sve točke fazne kružnice.

$$f_{MIN} = \frac{F_S}{2^N} \quad (3)$$

Sukladno Nyquistovom kriteriju o otipkavanju, teorijski najviša frekvencija signala kojeg je ovim postupkom moguće generirati je

$$f_{MAX} = \frac{F_S}{2} \quad (4)$$

## 1.2. Karakteristike direktne digitalne sinteze signala

Direktna digitalna sinteza signala omogućuje visoku točnost frekvencije, temperaturnu i vremensku stabilnost, širokopojasno podešavanje i vrlo brzo fazno kontinuirano podešavanje frekvencije. Karakteristike digitalne sinteze ograničene su karakteristikama digitalno-analognog pretvarača, koji je sastavni dio DDS-a. Digitalna informacija koja se dovodi na ulaz D/A pretvornika mora predstavljati trenutnu vrijednost amplitude izlaznog signala. Rezolucija D/A pretvornika određena je brojem ulaznih bitova. Njegova konačna rezolucija izaziva grešku kvantiziranja. No, pogreške u dobivanju trenutne amplitude nastaju i prije u samom algoritmu direktne digitalne sinteze. Te se pogreške prvenstveno odnose na proces interpolacije te kvantizaciju amplitude, a u nekim izvedbama i zbog fazne kvantizacije.

**Fazna kvantizacija** odvija se na faznoj informaciji i reducira broj bitova za indeksiranje sa N na P, a ostatak fazne riječi se odbacuje. Razlog za tu kvantizaciju leži u nastojanju da se zazume što manje memorije prilikom

pohranjivanja uzoraka signala. Za implementaciju tablice uzoraka bez kvantizacije faze potrebno bi bilo $2^N \cdot M$ bita što u praktičnom smislu nije prihvatljivo. Koristeći P bitova za adresiranje uzorka signala potrebno je $2^P \cdot M$ bitova memorije što je puno manje, te je ostvarivo. Nažalost, kao posljedica odsjecanja bitova faznog akumulatora javlja se šum koji nije uniformnog karaktera i ne rasprostire se unutar Nyquistovog područja kao temeljni šum. U metodi direktne digitalne sinteze, taj je šum periodičan, a perioda ponavljanja ovisi o omjeru namještene frekvencije sinteziranog signala i frekvencije otipkavanja. Kod nekih je omjera periodičnost kvantizacijskog šuma naglašenija pa se u spektru izlaznog signala pojavljuju dodatne linije (engl. *Spurs*). U implementaciji DDS-a u ovom radu, fazne kvantizacije u pravom smislu nema. Dio informacije faznog akumulatora koji se ne koristi za adresiranje koeficijenata iz tablice se ne odbacuje već se koristi u interpolaciji. Pomoću adresiranih koeficijenata i frakcijskog dijela akumulatora računa se amplituda koristeći Hornerovu formulu.

Pograška se javlja prilikom **kvantizacije amplitude**. Vrijednosti sinusoidne funkcije u osnovi su iracionalni brojevi koji se prikazuju kodnim riječima određene širine, zavisno o memorijskim mogućnostima. Šum koji nastaje kvantizacijom po amplitudi ni u ovom slučaju ne možemo smatrati uniformnim zbog periodičke prirode sinusoidne funkcije. U implementaciji DDS-a koja je obrađena u ovom radu kvantizirat će se koeficijenti (a, b, c, d) .

# 2. Programska implementacija DDS-a u Matlabu

Implementacija direktne digitalne sinteze sinusnog signala u programskom paketu Matlab poslužila je kao primjer algoritma koji je potrebno ostvariti na DSP procesoru.

Fazni akumulator koji se simulira u programu veličine je 28 bita, no samo se p=10 bita koristi za adresiranje tablice koeficijenata. Ostatak faznog akumulatora služi za akumulaciju faze. Program koristi tablicu koeficijenata samo za prvi kvadrant, dakle od 0 do $\pi/2$. Za kuteve od $\pi$ do $2\pi$ (treći i četvrti kvadrant) radi se samo negacija rezultata ćime se efektivno povečava rezolucija tablice za 1 bit. Također se koristi simetrija ( ili antisimetrija) koeficijenata unutar jedne poluperiode. Koeficijenti D i B su antisimetrični oko sredine, a koeficijenti C i A su simetrični. Na osnovu ovog pojednostavljenja, dovoljno je u ROM tablicu pohraniti samo prvu poluperiodu svakog koeficijenta, a onda ovisno o predzadnjem najvišem bitu samo korigirati predznak koeficijenta uz neparne potencije funkcije. Detalji oko ove simetrije/antisimetrije, te načina izračuna zrcalnih indeksa biti će kasnije objašnjeni. Dakle korištenjem obje modifikacije, ostvaruje se ušteda u pohrani tablice, pa se 10-bitna tablica može realizirati korištenjem 8-bitne. Koeficijenti će se kvantizirati različitim stupnjevima kvantizacije. Kvantizirati će se i pojedini produkti u Hornerovoj formuli. Koristeći Hornerovu formulu (5), kvantizirane koeficijente $\hat{d}[n]$, $\hat{c}[n]$, $\hat{b}[n]$ i $\hat{a}[n]$, te fazu $\Delta x$ izračunat će se amplituda izlaznog signala za taj korak.

$$w_n(\Delta x) = \left(\left(\hat{d}[n] \cdot \Delta x + \hat{c}[n]\right) \cdot \Delta x + \hat{b}[n]\right) \cdot \Delta x + \hat{a}[n] \qquad (5)$$

Sinusna amplituda $y_s(\phi) = \sin(\phi)$ se za danu fazu $\phi$ može aproksimirati sa po djelovima kubičnom funkcijom $y_r(\phi)$ sa $N$ jednakih segmenata unutar sva četiri kvadranta. Kubični polinomni segment između faznih uzoraka $n \cdot 2\pi/N$ i $(n+1) \cdot 2\pi/N$ može se ostvariti koristeći Hornerovu formulu (5). Normalizirani kubični argument $\Delta x$ koji se kreće u granicama od 0 do 1, jednak je nuli na lijevom faznom uzorku $n \cdot 2\pi/N$ , a jedinici na $(n+1) \cdot 2\pi/N$ uzorku. Kontinuirana fazna

reprezentacija aproksimirane sinusoide dobivena je ulančavanjem individualnih kubičnih segmenata:

$$y_r(\phi) = \left\{ w_n(\Delta x) | \Delta x = N\frac{\phi}{2\pi} - n\,, \phi\epsilon \left[ n\frac{2\pi}{N}, (n+1)\frac{2\pi}{N} \right] , \forall n \in [0, N-1\,] \right\} \qquad (6)$$

Sam program podjeljen je u više cjelina. Ovdje će biti opisane samo one koje će biti iskorištene u implementaciji na DSP procesoru.

Izračun koeficijenata koji će se pohraniti u memoriju može se izvesti na više načina:

1) **LMS koeficijenti –** računaju se idealni koeficijenti RMS modela sa integralnom mjerom odstupanja. Koeficijenti su reprezentirani u spektralnoj domeni, a njihovi moduli i faze su izraženi analitički.
2) **Cheby koeficijenti –** računaju se idealni koeficijenti MAE modela sa maximalnom apsolutnom vrijednosti kao mjerom odstupanja ali samo aproksimativno jer se ustvari koristi običan razvoj u Cheby polinome. Koeficijenti su reprezentirani u spektralnoj domeni preko svojih modula i faza.
3) **SFDR koeficijenti –** koeficijenti se generiraju kao idealne sinusne sekvence zadanih modula i početnih faza koje se računaju analitičkim izrazima.
4) **Spline koeficijenti –** računaju se idealni B-spline koeficijenti, moduli i faze su izdraženi analitički.

Koeficijenti su u programu izračunati B-spline metodom te je svaki koeficijent kvantiziran s različitim brojem bitova. Koeficijent $\hat{d}[n]$ kvantiziran je najmanjim brojem bitova, a koeficijent $\hat{a}[n]$ najvećim. Takvi kvantizirani koeficijenti pohranjeni su u ROM i to samo za prvi kvadrant (256 unosa).

```
phf=rem([0:Nt-1]*bin+poc,Nt)/Nt;
ph=floor(phf*2^b_ph_acc)*(2^-b_ph_acc);
in=floor(ph*N)+1;
inMSB=floor((in-1)/(N/2));
Zdx=ph*N-in+1;
```

```
Zdx=2*Zdx-1;
```

Ovim odsječkom kôda simulira se fazni akumulator. Varijabla *Nt* sadrži broj točaka za evaluaciju korištenjem DFT-a i iznosi $2^{20}$. Stoga je najmanja frekvencija (inkrmenet) što se može generirati $\frac{1}{2^{20}}$. *Phf* sadži vektor svih faza akumulatora ali tek *ph* kvantizira fazu na rezoluciju faznog akumulatora. Index se izračuna tako da se nomirana faza pomnoži sa potrebnim brojem segmenata preko cijelog kruga što je 1024 iliti $2^{10}$ i od toga uzme cijeli broj od o do N-1. U varijablu *inMSB* pohranjen je bit akumulatora najviše važnosti i on će određivati prvu ili drugu poluperiodu. Broj bita frakcije *Zdx* jednak je razlici rezolucije faznog akumulatora i broja bita za adresiranje tablice i iznosi 18 bita. Vrijednost frakcionalnog argumenta kretao se od 0 do 1 ali je zadnjom naredbom prebačen na raspon od -1 do 1.

```
in=rem((in-1),N/2)+1;
inNSB=floor((in-1)/(N/4));
ind=in;
ind(find(inNSB))=((N/2-1)-(in(find(inNSB))-1))+1;
sign_coef=[sign_coef_fix(1)*((-1).^inNSB)
           sign_coef_fix(2)*ones(1,Nt)
           sign_coef_fix(3)*((-1).^inNSB)
           sign_coef_fix(4)*ones(1,Nt)];
```

Izolira se sada stvarni indeks tako da se odbaci bit inMSB koji određuje predznak(poluperiodu). U inNSB se upiše predzadnji najviši bit koji je važan kod iskorištavanja simetrije/antisimetrije koeficijenata.

Simetrija je sljedeća: Nulti segment je uparen sa $N/2 - 1$ zrcaljenjem oko $\pi/2$, pa zatim prvi segment i $N/2 - 2$ i tako sve do $N/4 - 1$ i $N/2 - (N/4 - 1 + 1) = N/4$. Takvi zrcalni indeksi su vezani običnim prvim komplementom.

Sve što je potrebno napraviti je sljedeće:

1) Izdvoji MSB i NSB te ostatak NSB-1 do 0
2) Ako je NSB jednak jedinici (drugi ili četvrti kvadrant) tada komplementiraj ostatak
3) Sa takvim ostatkom čitaj koeficijente iz tablice dimenzije $N/4$ koja sadrži samo prvi kvadrant

4) Zatim, ako je NSB postavljen u jedinicu tj. ako smo u drugom ili četvrtom kvadrantu, potrebno je samo komplementirati predznak od koeficijenta d i b
5) Nakon toga, ako je i najviši bit (MSB) postavljen u jedinicu, potrebno je komplementirati sva četiri predznaka

U varijablu *ind* pohranjuju se prvi komplementi indeksa tamo gdje je inNSB jednak jedinici. U polje vektora *sign_coef* smješteni su svi predznaci koeficijenata ne zaboravljajući i fiksne predznake za d i c koeficijente koji su uvijek negativni.

```
yishq = sign_coef(1,:).*cubBqn(1,ind);
fprintf('\n Potrebne operacije mnozenja u Horner evaluaciji');
for i=2:nS+1,
rez=rez_coef(i)-sr(i-1);
b_dx=rez+b_dxq_g;
dxq=floor(Zdx*2^(b_dx))*(2^-(b_dx));
dxq=yishq.*dxq;
dxq=round(dxq*2^rez)*2^-rez;
dxq=dxq*2^-(sr(i-1));
yishq=dxq+sign_coef(i,:).*cubBqn(i,ind);
fprintf('\n coe (1.%d) x dxq (1.%d) -> pr (1.%d) -> Q na (1.%d)',...
            rez_coef(i-1),b_dx,rez_coef(i-1)+b_dx,rez);
   fprintf('\n nakon sr(%d)=%d i zbrajanja .. format izlaza je
(1.%d)',...
            (i-1),sr(i-1),rez_coef(i));
end;
fprintf('\n\n');
yishq=((-1).^inMSB).*yishq;
```

Predhodnim odsječkom kôda ostvarena je cijelobrojna realizacija algoritma izračuna amplitude. U varijabli yishq nalazit će se izračunata amplituda izlaznog signala. Najprije se u nju pohrani kvantizirani i skalirani d koeficijent.

Dovoljno je da izlazna rezolucija produkta bude za jedan bit točnija od sljedečeg koeficijenta kojem se pribraja i to nakon pomaka u desno za određen broj koraka. Tako se najviši bit iza decimalne točke može iskoristiti kao prijenos (engl. *Carry)* za operaciju zaokuženja umnoška. Potrebna rezolucija za kvantizaciju umnoška nalazi se u varijabli *rez.* Ulazni argument množenja Zdx ne mora biti puno točniji od izlazne rezolucije produkta, pa se i on može skratiti na odgovarajuću širinu. Njegova širina krati se na rezoluciju *b_dx* koja je za dva bita veća od izlazne rezolucije produkta *rez.* Sada se taj kvantizirani Zdx množi sa *yishq*. Sljedeći korak je da se taj umnožak zaokruži i poravna na rezoluciju sljedečeg koeficijenta. Tada se izvrši pomak u desno za potreban broj koraka se

se na to zbroji sljedeći koeficijent. Postupak se ponavlja u *for* petlji dok se realizira cijela Hornerova formula. Na kraju je potrebno još podesiti predznak amplitude u ovisnosti o najvišem bitu *inMSB.*

# 3. Blackfin procesor ADSP-BF537

## 3.1. Procesori za digitalnu obradu podataka

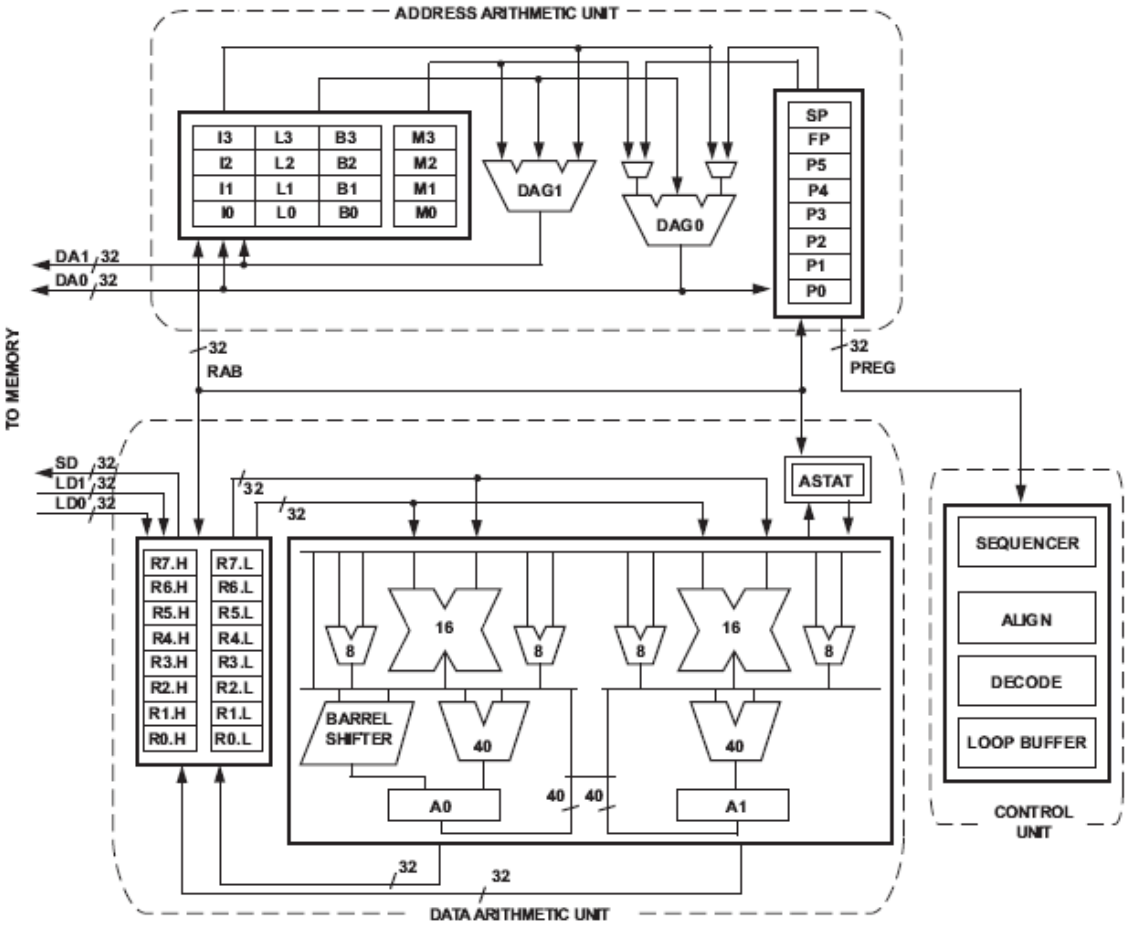
Digitalna obrada signala jedna je od najupotrebljavanijih tehnologija današnjice. Primjenjuje se u stereo sustavima, auto industriji, osobnim računalima, celularnoj telefoniji, obradi slike itd. "Uređaji" koji obavljaju digitalnu obradu zovu se DSP procesori (engl. *Digital signal processors*). U početku su bili namjenjeni obradbi visokofrekventnih signala, radio signala, zvučnih signala i mikrovalova, ali se primjena proširila i na ostala područja gdje se pojavljuje zahtjev za velikim brojem proračuna kao npr. u visokokvalitetnom procesiranju grafike i u inžinjerskim simulacijama. DSP rade sa signalima iz stvarnog svijeta odnosno u realnom vremenu pa je za njihov rad karakteristično da moraju biti u stanju brzo obrađivati primljene signale. Realni signali su kontinuirani, analogni signali koji se mijenjaju u vremenu, a koji nisu pogodni za brzu obradbu pa se zato moraju pretvoriti u pogodnije, diskretne signale. Transformacija analognih u digitalne signale obavlja se pomoću kvalitetnih analogno-digitalnih pretvarača (ADC - Analog-to-Digital Converter) koji osiguravaju dovoljno uzimanje uzoraka kako bi digitalni signal mogao zadržati bitna svojstva kao i stvarni, analogni signal.

Bitna razlika između DSP-a i mikroprocesora je što su mikroprocesori građeni za obavljanje općenitih funkcija i što se koriste velikim blokovima programa poput operacijskih sustava. Iz tih razloga mikroprocesori ne rade u realnom vremenu jer jednostavno ne stignu zadovoljiti sve potrebe ugrađenih funkcija. Za razliku od toga DSP procesori imaju striktno namjensku ulogu, koristeći se malim blokovima funkcija s kojima rade vrlo brzo. Zato DSP procesori mogu raditi kao dodatni procesori glavnom procesoru, mikroprocesoru.

## 3.2. Karakteristike procesora ADSP-BF537

Blackfin je porodica procesora tvrtke Analog Devices bazirana na novoj arhitekturi MSA (engl. *Micro Signal Arhitecture*) sa jedinstvenom jezgrom namjenjena digitalnoj obradi signala. Blackfin procesori imaju RISC protočnu strukturu sa DSP dodatkom koji se temelji na dvije paralelne jedinice za množenje

(MAC- engl *Multiply-Accumulate*). Takva kombinacija je posebno prilagođena za upotrebu u audio i video obradama signala te u komunikacijskim sustavima. Kao što je prikazano na Slici 5, jezgra sadrži još i dva 40-bitna akumulatora, dvije 40-bitne aritmetičko logičke jedinice (ALU), četiri 8-bitne video ALU, te 40-bitni posmačni registar (engl. *Shifter*). Računalne jedinice obrađuju 8-bitne, 16-bitne ili 32-bitne podatke iz registarskog stoga. Registarski stog čini osam 32-bitnih registara koji su izvor i cilj svih ALU i MAC operacija. Time je ostvarena ortogonalnost instrukcijskog skupa i olakšano je programiranje i prevođenje u višim programskim jezicima. U radu sa 16-bitnim podacima, registarski stog se dijeli na šesnaest ravnopravnih 16-bitnih registara kako je prikazano slikom 5.




**Slika 5. Blackfin procesorska jezgra**

Svaki sklop za množenje podržava množenje dva 16-bitna podatka u jednom ciklusu s mogućnošću akumulacije rezultata u 40-bitni akumulator. Podržani su cijelobrojni i frakcionalni podaci sa i bez predznaka. Aritmetičko

logičke jedinice obavljaju tradicionalni set aritmetičkih i logickih operacija sa 16 ili 32-bitnim podacima. Dodano je mnogo posebnih instrukcija radi ubrzavanja rada kod obrade signala kao što su modulo $2^{32}$ množenje, zaokruživanje, detekcija predznaka i eksponenta i sl. Pri operacijama sa 16-bitnim podacima omogućeno je obavljanje paralelnih operacija nad parom registara, npr. zbrajanje gornje polovice registra R1 s gornjom polovicom R0 te istodobno oduzimanje donje polovice R0 od donje polovice R1. Postojanje dviju ALU omogućava paralelno izvršavanje bilo kojih dviju operacija, pa je tako moguće obaviti i do četiri instrukcije nad parom registara. 40-bitni posmačni registar može primiti podatak te obaviti nad njim operaciju pomaka, rotiranja, normaliziranja ili odsjecanja.

Arhitektura procesora omogućuje tri načina rada: korisnički (engl. *User*), nadglednički (engl. *Supervisor*) i emulacijski (engl. *Emulation*). Korisnički način rada ograničava pristup sistemskim resursima te daje zaštičenu softversku okolinu. Nadglednički i emulacijski način rada daju potpuni pristup resursima jezgre. Instrukcijski set Blackfin procesora optimiziran je tako da se najčešće korištene instukcije kodiraju 16-bitnim kodom. Složene DSP instrukcije su kodirane 32-bitnim kodom kao multifunkcionalne instrukcije. 32-bitne i 16-bitne instrukcije mogu se paralelno dobavljati što omogučuje programeru manje korštenje resursa jezgre u jednom instrukcijskom ciklusu. Procesor podržava mješanje takvih naredbi u paketima od 64 bita radi boljeg iskorištenja programske memorije. Blackfin procesori koriste algebarski asemblerski jezik ali je arhitektura prilagođena i za korištenje C prevodioca.

Usklađivać programa (engl. *Program Sequencer)* kontrolira tok izvršavanja instrukcija, poravnavanje instrukcija te njihovo dekodiranje. U svrhu kontrole toka programa, usklađivać podupire PC relativne i indirektne uvjetne skokove (sa statičkim predviđanjem grananja) te pozive rutina. Hardver omogučuje *zero-overhead looping* što znači da se hardver bavi sa kontrolom petlje a ne instrukcije grananja. Ostvarenje petlje ne troši cikluse pa se na taj način ubrzava izvođenje programa. Arhitektura je potpuno isprepletena što znači da nema vidljivog efekta cijevovoda kod izvođenja instrukcija sa podatkovnom ovisnošću.

Adresna aritmetička jedinica daje dvije adrese za dva simultana dohvata iz memorije. Sadrži 4 skupine 32-bitnih registara za ostvarivanje kružnih

međuspremnika a to su: registri za baznu adresu (engl. *Base registers*), za duljinu buffera (engl.*Length registers*), uvećanje adrese (engl. *Modify registers*), te registri koji sadrže adresu podatkovne strukture (engl. *Index registers*). Osim toga postoji i 8 dodatnih 32-bitnih registara koji predstavljaju pokazivač na memorisku lokaciju. Blackfin procesori imaju Harvardsku arhitekturu u kombinaciji sa hijerarhijskom strukturom memorije.

**Arhitektura memorije**

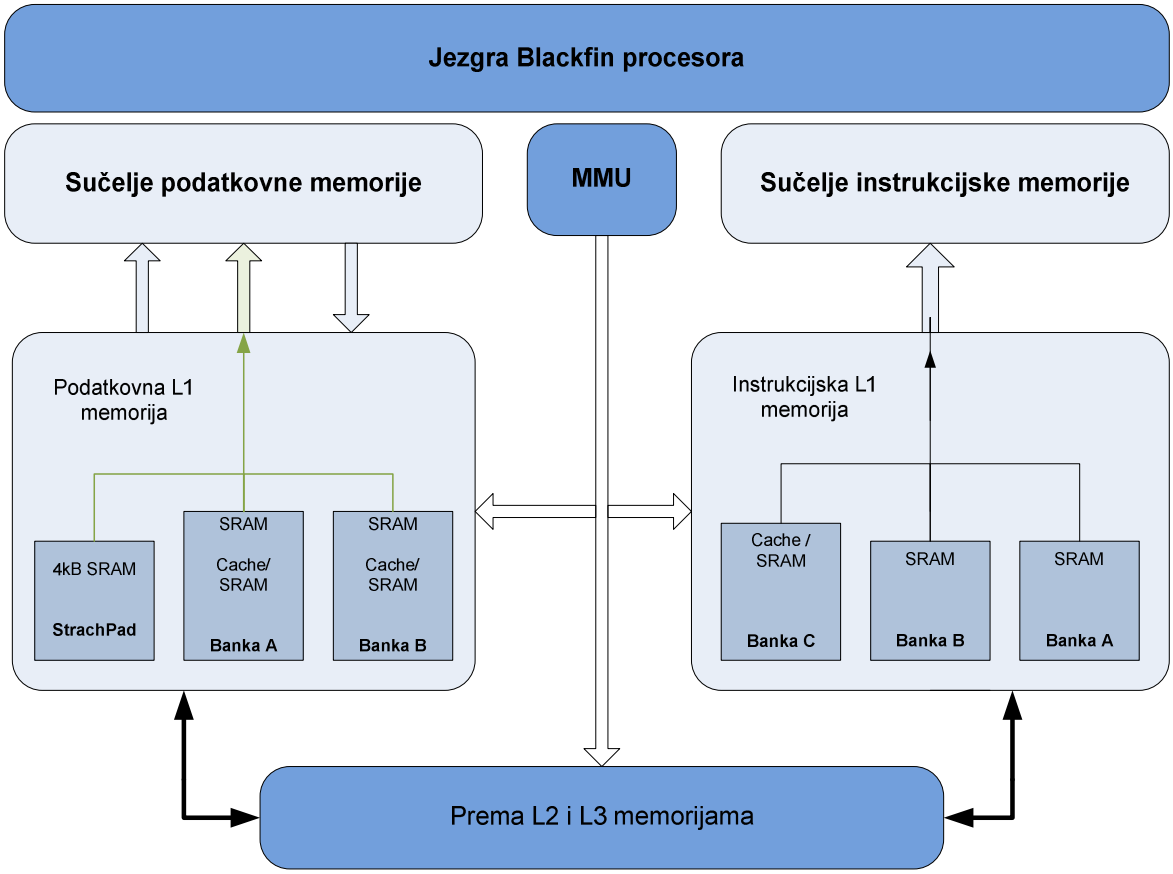
Arhitektura Blackfin procesora predstavlja memoriju kao opći adresni prostor od 4Gb koristeći 32-bitne adrese. Svi resursi, uključujući unutrašnju memoriju, vanjsku memoriju, ulazno/izlazne kontrolne registre, zuzimaju zasebne djelove zajedničkog adresnog prostora. Memorijski djelovi adresnog prostora su organizirani u hijerarhijsku strukturu da omogučuju dobar balans cijene i performansi između vrlo brze memorije kao što je priručna memorija (engl. *Cache)* ili SRAM i velikih, jeftinih *off-chip* memorijskih sustava nižih performansi.

Level 1 (L1) memorija je neposredno vezana za jezgru i radi na punom taktu procesora, stoga ostvaruje najviše brzine rada za kritične programske odsječke. Organizacija L1 memorije prilagođena je za brzo izvođenje obrade signala ali uz zadržavanje maksimalne jednostavnosti programskog modela kao kod običnih kontrolera. L1 memorija može biti ostvarena kao pričuvna memorija, kao SRAM memorija direktno mapirana u memorijski prostor jezgre ili kao kombinacija pričuvne i SRAM memorije. Proizvoljna podjela L1 memorije omogućava smještanje kritičnih djelova koda u SRAM memoriju osiguravajući veliku propusnost uz minimalnu latenciju. Pričuvna memorija koristi se za pristup u više nivoe hijerarhijskog modela ubrzavajući tako izvođenje zadataka s manje strogim vremenskim uvjetima.

U procesoru ADSP-BF537 L1 memorija je podjeljena u tri bloka:

- *L1 instrukcijska memorija* je implementirana kao SRAM memorija i kao pričuvna memorija sa četverostrukim pristupom. Radi na punoj brzini procesora.
- *L1 podatkovna memorija* se ostvaruje kao SRAM memorija i/ili kao pričuvna memorija sa dvostrukim pristupom. Također radi na punoj brzini procesora.

- *L1 blok RAM memorija* (engl. *Scratchpad)* radi na istim brzinama kao ostale memorije ali je ostvarena SRAM memorijom i ne može se konfigurirati kao pričuvna memorija.




**Slika 6. Konfiguriranje L1 memorije**

*Off chip* memorijski sustav, do kojeg se pristupa preko sučelja vanjske sabirnice EBIU (engl. *External Bus Interface Unit*), proširuje memoriju SDRAM-om, FLASH memorijom i SRAM-om do 132Mb fizičke memorije. Svi Blackfin procesori imaju i višestruke nezavisne DMA kontrolere koji podržavaju automatizirani prijenos podataka uz minimalno opterećenje procesorske jezgre.

## 3.3. Implementacija DDS-a na Blackfin procesoru

Programska podrška za razvoj programa za blackfin procesore je VisualDSP++. Unos asemblerskog ili C koda je slobodnog formata, naredbe se mogu unositi u bilo koji dio reda, mogu se protezati kroz više redova, može se više naredaba nalaziti u jednom redu, ne razlikuju se velika i mala slova,a naredbe moraju završiti znakom točka-zarez.

Ovaj program pisan je programskim jezikom C, te preveden VisualDSP C prevodiocem. U potpunosti se slijedi algoritam realizacije direktne digitalne sinteze u Matlabu. Programom nije obuhvaćen dio koji opisuje postupak izračuna koeficijenata te njihova kvantizacija već je implementiran sam postupak direktne digitalne sinteze. Generirani koeficijenti su preuzeti iz Matlaba te pohranjeni u memoriju kao jednodimenzionalna polja.

```
#include<stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ccblkfn.h>
#define N 1024

double round(double a)  {
return (int)(a + 0.5);
}
```

Prije glavnog programa deklarirana je funkcija koja zaokružuje *double* tip podatka na najbliži cijeli broj.

```
FILE *fp;
 int sign_coef_fix[]={-1,-1,1,1};
 int sign_coef[4];
 int red_spline=3;
 int rez_coef[]={4,14,23,31};
 int sr[]={10,9,8};
 int rez=0;
int b_dxq_g=2;
int b_dx;
int i;
double izlaz[1048576];
double inkrement;
double ph=0;
double phf;
```

```
double Zdx=0;
double yishq;
double dxq;
int in=0;
int ind,j;
int inMSB=0;
int inNSB=0;
inkrement= 1./1048576;
```

Deklaracija varijabli. U *sign_coef_fix* varijabli čuvaju se fiksni predznaci. U prvom kvadrantu koeficijenti *d* i *c* su uvijek negativni. U varijabli *rez_coef* navedene su pojedine rezolucije koeficijenata.Vidimo da je rezolucija *d* koeficijenta samo 4 bita dok je rezolucija *a* koeficijenta 31 bit. *Inkrement* faznog akumulatora bit će $\frac{1}{1048576}$ što direktno određuje frekvenciju izlaznog signala.

```
for(phf=0, j=0; phf<=1; phf=phf+inkrement, j++)
```

U ovoj *for* petlji ostvaruje se simulacija faznog akumultora. U svakom koraku vrijednost inkrementa dodaje se na predhodnu vrijednost akumulatora. Sljedeći odsječak koda potpuno je ekvivalentan onom iz Matlaba. Kvantizira se *phf* na rezoluciju faznog akumulatora, računa se index te frakcionalni dio.

```
ph=floor(phf*pow(2,28))*pow(2,-28);
in=floor(ph*N);
inMSB=floor((in-1)/(N/2));
Zdx=ph*N-in;
Zdx=2*Zdx-1;
```

Sljedeća linija koda izbacuje najvažniji bit iz indeksa koji određuje redznak te tako dobivamo stvarni indeks. To se postiže ostatkom djeljenja N/2 sa trenutnim indeksom.

```
in= ((in-1)-((int)((in-1)/(N/2))*(N/2)))+1;
```

Sljedeće se odrđuje najviši predzadnji bit u svrhu iskorištavanja simetrije i asimetrije koeficijenata. U ovisnosti o tom bitu radi se dvojni komplement adrese koja se sprema u varijablu *ind.*

```
ind=in;
inNSB=floor((in-1)/(N/4));
if(inNSB) {
        ind=(((N/2)-1)-in)+1;
}
```

Ovim kodom prikazano je vračanje orginalnih negativnih predznaka koeficijenata *d* i *c* . Također, u ovisnosti o bitu inNSB mjenjaju se predznaci koeficijenata *d* i *b*

```
sign_coef[0]=sign_coef_fix[0]*pow(-1,inNSB);
sign_coef[1]=sign_coef_fix[1];
sign_coef[2]=sign_coef_fix[2]*pow(-1,inNSB);
sign_coef[3]=sign_coef_fix[3];
```

Sljedeći dio ponovno je ekvivalentan kodu u Matlabu osim što se za koeficijente nije deklarirala matrica $4 \times 2^{20}$ već je svaki koeficijent odvojen u posebno jednodimenzionalno polje. To rezultira neke preinake u kodu iz Matlaba, no algoritam je i dalje isti. Uzima se kvantizirani d koeficijent, množi se sa kvantiziranim frakcijskih dijelom *dxq* te se i taj umnožak kvantizira na ogovarajuću širinu.

```
yishq = sign_coef[0]*d[ind];
for(i=1;i<=red_spline+1;i++){
rez=rez_coef[i]-sr[i-1];
b_dx=rez+b_dxq_g;
dxq=floor(Zdx*pow(2,b_dx))*pow(2,-b_dx);
dxq=yishq*dxq;
dxq=round(dxq*pow(2,rez))*pow(2,-rez);
dxq=dxq*pow(2,-sr[i-1]);
```

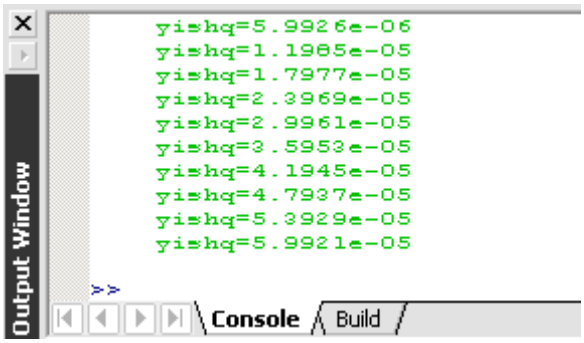
Sljedećom switch naredbom odlučuje se koji se koeficijent zbraja na dosadašnju vrijednost *dxq-a* u ovisnosti dubini pozicije u Hornerovoj formuli.

```
switch( i )
{
case 1:
        yishq=dxq+sign_coef[i]*c[ind];
break;
 case 2 :
        yishq=dxq+sign_coef[i]*b[ind];
break;
```

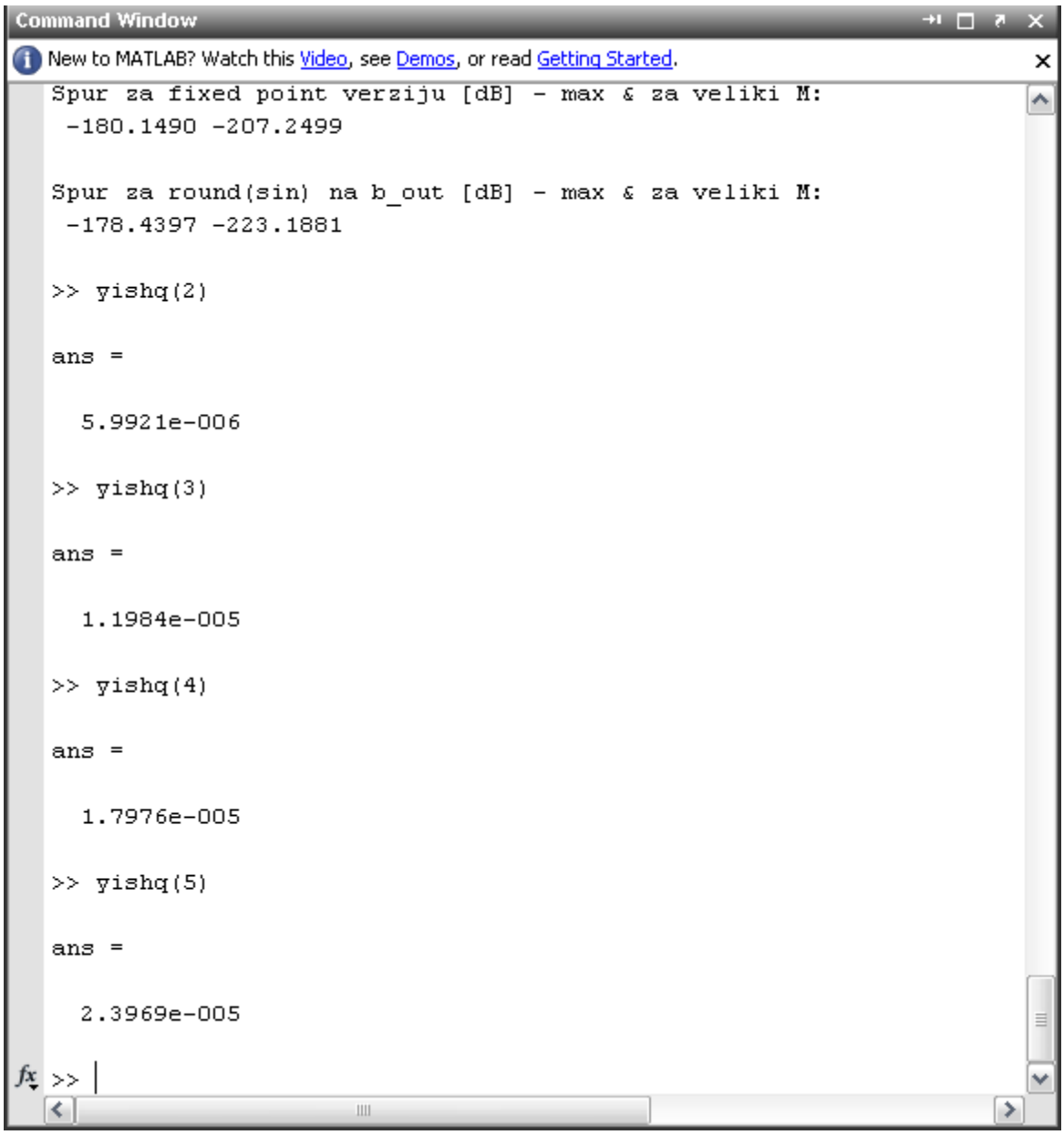
```
case 3 :
        yishq=dxq+sign_coef[i]*a[ind];
break;
}
```

Na kraju se krajnji *yishq* pohranjuje u polje rezultata i ispisuje u izlaznu datoteku te se prelazi na sljedeći inkrement akumulatora.

Dobivene amplitude izlaznog signala (slika 7.) identične su onim dobivene u Matlabu (slika 8) samo što je brzina izvođena programa u simulatoru VisualDSP-a drastično sporija. Na slikama je dan ispis prvih nekoliko amplituda.




**Slika 7. Izvođenje programa u VisualDSP-u**




**Slika 8. Izvođenje programa u Matlab-u**

U prilogu na kraju rada cijeli kod je uvršten kao dodatak. Izostavljena je samo deklaracija koeficijenata.

# Zaključak

Iako je tehnika direktne digitalne sinteze prvi put opisana još 1971. godine, tek je zadnjih desetak godina izazvala zanimanje zbog naglog razvoja digitalne tehnike i računala. DDS ima i prednosti i nedostatke u odnosu na druge tehnike frekvencijekse sinteze. DDS sustav učinkovito omogučuje generiranje sinusne reference koja je sinkronizirana s ulaznim taktom frekvencije. Arhitekture DDS-a predstavljaju okruženja koja omogućavaju daljinsku kontrolu i upravljanje u realnom vremenu pomoću mikroprocesora. Direktna digitalna sinteza je zbog svojih dobrih osobina našla primjenu u onim realizacijama koje su ranije bile ograničene samo na primjenu PLL.

VLASTORUČNI POTPIS

# Literatura

[1] Giannini, R. Sinteza signala za podražaje u audiološkim pretragama. Magistarski rad. Fakultet elektrotehnike i računarstva, 1999.

[2] Vještica, R. Izravna digitalna sinteza (DDS) sinusnog signala. Diplomski rad. Elektrotehnički fakultet Osijek, 2003.

[3] Petrinović, Davor, Arhiva dokumenata za Blackfin porodicu, 22.04.2009, http://www.fer.hr/predmet/ppzdos/vjezbe_i_samostalni_rad , 12.12.2009.

[4] Dragan Sekulić, Saša Filipović, Radojka Praštalo, Osnove DDS-a, *Osnove direktne digitalne sinteze,*

www.etfbl.net/dokument.php/5482/1/Osnove_DDSa.doc , 10.1.2010.

[5] Analog Devices, C/C++ Compiler and Library Manual for Blackfin Processors, 10.2009,

http://www.analog.com/static/imported-files/software_manuals/50_ccblk_man_5.2.pdf , 12.12.2009

[6] Analog Devices, A Technical Tutorial on Digital Signal Synthesis, 1999,

http://www.analog.com/static/imported-files/tutorials/450968421DDS_Tutorial_rev12-2-99.pdf , 4.1.2010.

# DIREKTNA DIGITALNA SINTEZA SINUSNOG SIGNALA NA DSP PROCESORU

## Sažetak

Mnogo raznih komunikacijskih sustava, kao što je GPS, digitalni modem i razni radarski sustavi zahtjevaju lokalni generator sinusnog signala promjenjive frekvencije. U tu svrhu koriste se sintetizatori frekvencije koji rade na principu direktne digitalne sinteze signala. To je postupak dobivanja željenog analognog, uglavnom sinusnog signala iz digitalnih uzoraka signala pohranjenih u memoriji. Postupak sinteze temelji se na konvencionalnom faznom akumulatoru, a pretvorba faze u vrijednost sinusnog uzorka provodi se postupkom kubične B-spline interpolacije. Implementacija direktne digitalne sinteze ostvarena je na Blackfin procesoru za digitalnu obradu signala ADSP-BF537 tvrtke Analog Devices prema referentnom kodu u Matlabu.

## Ključne riječi

Direktna digitalna sinteza, DDS, fazni akumulator, B-spline interpolacija, DSP procesor, digitalna obrada signala

# DIRECT DIGITAL SYNTHESIS OF SINEWAVE SIGNAL ON A DSP PROCESSOR

## Summary

Many communications systems, such as global positioning system (GPS), digital modems, and RADAR systems, require a locally generated tunable reference sinusoid. Direct digital synthesizers are used for this purpose. They are based on direct digital sinthesis. Direct digital sinthesis is a tehnique for using digital data processing blocks witch are stored in memory as a means to generate a analog sinewave output signal. Algorithm of direct digital sinthesis is based on convencional phase acummulator and conversion of phase to amplitude is implement by cubic B-spline interpolation. Implementaton of DDS is realize on Analog Devices Blackfin DSP processor ADSP-BF537.

## Keywords

Direct digital sinthesis, DDS, phase acummulator, B-spline interpolation, DSP processor, digital signal processing

# Dodatak

```
#include<stdio.h>
#include <math.h>
#include <stdlib.h>
#define N 1024

double round(double a) {
return (int)(a + 0.5);
}
int main() {
FILE *fp;
    int sign_coef_fix[]={-1,-1,1,1};
    int sign_coef[4];
    int red_spline=3;
    int rez_coef[]={4,14,23,31};
    int sr[]={10,9,8};
    int rez=0;
    int b_dxq_g=2;
    int b_dx;
    int i;
    double izlaz[1048576];
    double inkrement;
    double ph=0;
    double phf;
    double Zdx=0;
    double yishq;
    double dxq;
    int in=0;
    int ind,j;
    int inMSB=0;
    int inNSB=0;
    inkrement= 1./1048576;
    fp=fopen("dat.txt" ,"w");
    if(fp=NULL){
```

```
            printf("Greška kod otvaranja datoteke");
            exit(1);
            }
   for(phf=0,j=0;phf<=1;phf=phf+inkrement,j++) {
   ph=floor(phf*pow(2,28))*pow(2,-28);
   in=floor(ph*N);
   inMSB=floor((in-1)/(N/2));
   Zdx=ph*N-in;
   Zdx=2*Zdx-1;
   in= ((in-1)-((int)((in-1)/(N/2))*(N/2)))+1;
   ind=in;
   inNSB=floor((in-1)/(N/4));
   if(inNSB){
         ind=(((N/2)-1)-in)+1;
}

sign_coef[0]=sign_coef_fix[0]*pow(-1,inNSB);
sign_coef[1]=sign_coef_fix[1];
sign_coef[2]=sign_coef_fix[2]*pow(-1,inNSB);
sign_coef[3]=sign_coef_fix[3];

yishq = sign_coef[0]*d[ind];
for(i=1;i<=red_spline+1;i++){

      rez=rez_coef[i]-sr[i-1];
      b_dx=rez+b_dxq_g;
      dxq=floor(Zdx*pow(2,b_dx))*pow(2,-b_dx);
      dxq=yishq*dxq;
      dxq=round(dxq*pow(2,rez))*pow(2,-rez);
      dxq=dxq*pow(2,-sr[i-1]);
      switch( i )
{
   case 1:
       yishq=dxq+sign_coef[i]*c[ind];
       break;
```

```
    case 2 :
       yishq=dxq+sign_coef[i]*b[ind];
       break;
    case 3 :
       yishq=dxq+sign_coef[i]*a[ind];
       break;
}


}
    yishq=(pow((-1),inMSB))*yishq;
    izlaz[j]=yishq;
    fprintf (fp, "yishq[%d]= %f \n",j,izlaz);

    //printf("yishq=%f\n",yishq);


}

    fclose(fp);

    getchar();
    return 0;

}
```