Univerzitet u Nišu
Elektronski fakultet

Vladimir Ćirić

SINTEZA SAVIJENIH SEMI-SISTOLIČKIH POLJA ZA FIR FILTRIRANJE

Magistarska teza

Niš, 2005

SADRŽAJ

1. UVOD

Digitalna obrada signala (*Digital Signal Processing* – DSP) je kontinualni matematički proces, koji se u realnom vremenu primenjuje na digitalni signal. U DSP procese spadaju algoritmi kao što su diskretna kosinusna transformacija, diskretna *wevelet* transformacija, algoritmi za smanjenje redundantnosti informacija, digitalno filtriranje, konvolucija, korelacija, brze Furier-ove transformacije, itd.

Filtriranje sa konačnim impulsnim odzivom (*Finite Impulse Response Filtering* – FIR Filtering) je jedno od centralnih izračunavanja u digitalnoj obradi signala, sa posebnom primenom u multimedijalnim sistemima. FIR filtri se mogu implementirati softverski na procesorima opšte namene, na DSP procesorima, ili na specijalizovanim (problemu posvećenim) hardverskim akceleratorima. Sa stanovišta brzine sistema najbolji rezultati postižu se hardverskim akceleratorima zasnovanim na problemu posvećenim procesorskim poljima (*Dedicated Arhitectures*).

Postoji mnoštvo, do sada predloženih, arhitektura za FIR filtriranje. Regularna struktura algoritma za FIR filtriranje je najpogodnija za implementaciju na sistoličkim poljima [1]-[5]. Sistolička polja su procesorska polja sa jednostavnim procesnim elementima koja rade sinhrono, imaju regularne veze i eksploatišu protočnost. Semi-sistolička polja su posebna klasa sistoličkih polja, kod kojih postoji izvesna neregularnost u povezanosti procesnih elemenata. Najčešće je reč o postojanju *broadcast* linije [6]. Iskustva u implementaciji procesorskih polja u VLSI (*Very Large Scale Integration*) tehnologiji pokazala su da se veća propusnost sistema postiže kod semi-sistoličkih polja, jer je moguće ostvariti finiju protočnost u odnosu na implementacije sa potpuno-sistoličkim poljima [7].

Tehnologija prenosivih uređaja se razvija velikom brzinom. Razvijen je veliki broj standarda za bežične komunikacije, uključujući varijante IEEE 802.11 specifikacije bežičnih mreža. Tradicionalno, uređaji zahtevaju poseban čip kako bi podržali svaki od standarda. Međutim, koristeći razvoj mobilnih tehnologija provajderi se međusobno utrkuju promovišući nove usluge kao što su razne vrste multimedijalnih usluga. Svaka nova usluga zahteva poseban čip, ili sistem sačinjen od više čipova fizički integrisanih u jednom čipu [8]. Dodatni hardver povećava cenu uređaja, zauzima prostor, povećava i potrošnju i povećava vreme potrebno za projektovanje uređaja. Ovaj problem može biti rešen korišćenjem adaptivnog pristupa. Ovakvim pristupom, softver može da promeni konfiguraciju čipa u toku rada, što omogućava da jedan procesor ima više funkcija [8]-[11]. Svi specijalizovani sistemi imaju usko područje primene i uvek postoji težnja da se za specijalizovani sistem proširi područje mogućih aplikacija [12]-[13]. Povećanje fleksibilnosti

sistema za FIR filtriranje ogleda se u konfigurisanju broja ćelija filtra i dužine koeficijenata. Kod adaptivnog pristupa, parametrima kola u toku rada upravlja softver, dajući upravljačkoj logici kola instrukcije za promenu tipa izračunavanja. Ovakve hardverske module je moguće rekonfigurisati za nekoliko taktnih intervala [8].

Izbor arhitekture za FIR filtriranje uključuje razmatranje kompromisa između složenosti hardvera i propusnosti arhitekture. Poznato je da performanse i cena svakog digitalnog kola zavise od stila projektovanja. Prema tome, kod projektovanja određene arhitekture, uz optimalan kompromis površina-vreme, potreban je pažljiv izbor tehnike projektovanja. Za navedene athitekture, cilj je ostvariti zahteve vezane za propusnost uz minimalni obim hardvera, čime se određuje kompromis površina-vreme [10]. Neophodan je pažljiv izbor tehnike projektovanja integrisanog kola. Tehnika savijanja predstavlja idealnu platformu za ostvarivanja kompromisa površina-vreme [14].

Minimizacija silicijumske površine kod tehnike savijanja ostvaruje se redukovanjem broja funkcionalnih jedinica, kao što su množači i sabirači, odnosno procesni elementi u semi-sistoličkim poljima. Transformacija savijanja se koristi kako bi se na sistematski način projektovala upravljačka logika u DSP arhitekturi kod koje je više operacija vremenski multipleksirano na jednu funkcionalnu jedinicu. Izvršavanjem više operacija DSP algoritma na jednoj funkcionalnoj jedinici, broj funkcionalnih jedinica potrebnih za implementaciju se smanjuje na račun vremena, što za rezultat ima integrisano kolo koje zauzima manje prostora na silicijumu [14].

Cilj ovog rada je sinteza savijenih semi-sistoličkih polja za FIR filtriranje koja imaju mogućnost konfiguracije parametara filtriranja u toku rada. Uvođenjem novog načina primene tehnike savijanja, kojim se omogućava dinamičko preslikavanje operacija, sinteza savijenih semi-sistoličkih polja za FIR filtriranje sa mogućnošću konfiguracije parametara biće predložena na sistematski način, procedurom.

Mogućnost konfiguracije savijenih polja biće istražena i ugrađena u polje kroz primenu tehnike savijanja. Fleksibilnost u izračunavanju biće postignuta dinamičkim preslikavanjem operacija na različite funkcionalne jedinice savijenog sistema. Dinamičkim preslikavanjem operacija, postignutim primenom tehnike savijanja, biće omogućeno prepoznavanje parametara filtriranja kao korisnički definisanih parametra. U ovom radu biće predložena savijena semi-sistolička polja na kojima se implementiraju fleksibilna izračunavanja. Mogućnost povećanja propusnosti uz smanjenje broja operacija koje izvršavaju funkcionalne jedinice savijenog sistema će biti posebno razmatrana. Biće predstavljene dve nove procedure sinteze savijenih polja, zasnovane na tehnici savijanja. Prva procedura omogućava sintezu polja sa izmenjivom dužinom koeficijenata, dok se drugo rešenje odnosi na sintezu polja koje omogućava rad sa izmenjivim brojem ćelija filtra i izmenjivom dužinom koeficijenata.

Rad je podeljen na sedam poglavlja. Nakon uvodnog dela, u drugom poglavlju date su osnove digitalnog filtriranja. Prikazana je operacija konvolucije signala i matematičke osnove digitalnog filtriranja. Posebna pažnja u ovom poglavlju biće posvećena grafičkom predstavljanju strukture digitalnih sistema za filtriranje.

U trećem poglavlju detaljano će biti prikazano izabrano “kandidat-polje” za primenu tehnike savijanja. Arhitektura izabranog “kandidat-polje” biće predstavljena u dva nivoa detaljnosti. Detaljna analiza zavisnosti dimenzija polja od karakterističnih parametara filtra će biti izložena.

Četvrto poglavlje daje teoretsku osnovu za primenu tehnike savijanja pri projektovanju DSP sistema. Tema petog, centralnog poglavlja, sinteza savijenih semi-sistolickih polja za FIR filtriranje. Na početku petog poglavlja biće predstavljena nova transformacija izvorne arhitekture, koja će biti izvedena u cilju pripreme polja za uspešnu primenu tehnike savijanja. Ispravnost rezultata transformisane arhitekture biće dokazana transformacijom prenosne funkcije. Na transformisanu arhitekturu biće primenjena tehnika savijanja na način koji omogućava sintezu savijenog semi-sistolickog polja za FIR filtriranje sa promenljivim faktorom savijanja. Skupovi savijanja će biti dodeljeni arhitekturi u opštem obliku. Očekivani izgled savijene arhitekture će na osnovu dodeljenih skupova savijanja biti ilustrovan blok dijagramom. Nakon dodele skupova savijanja biće izvršena sinteza savijene arhitekture. Savijena arhitektura biće predstavljena grafom toka podataka (*Data Flow Graph* – DFG) u opštem obliku. Princip FIR filtriranja na savijenoj arhitekturi će biti detaljno objašnjen. Tok podataka kroz arhitekturu biće grafički ilustrovan.

Analiza DGF-a savijene arhitekture biće izvršena u cilju prepoznavanja delova arhitekture na koje faktor savijanja ima uticaja. Na osnovu analize DFG-a biće projektovana dodatna upravljačka logika koja omogućava promenu broja operacija savijenih na jednu funkcionalnu jedinicu. Postupak filtriranja sa promenljivim brojem operacija biće detaljno objašnjen u (5.2.3).

U cilju dodatnog povećanja oblasti primene savijenih semi-sistoličkih polja za FIR filtranje, tehnika savijanja će biti primenjena na način koji omogućava prepoznavanje broja i dužine koeficijenata filtra kao korisnički definisanih parametara. Novi način preslikavanja operacija na funkcionalne jedinice savijenog sistema će biti predstavljen. Sinteza savijenog polja sa promenljvim brojem i dužinom koeficijenata biće prikazana detaljno. Tokovi podataka će biti grafički ilustrovani. Novim načinom primene tehnike savijanja biće omogućeno da broj koeficijenata bude različit od projektovanog broja funkcionalih jedinica savijenog sistema. Minimalni broj potrebnih registara biće određen. Savijena arhitektura će biti prikazana DFG-om i funkcionalnim blok dijagrmom. Princip filtriranja na savijenoj arhitekuri biće dat u (5.3.4).

Nakon sinteze savijenog polja, biće prikazan postupak sinteze hardvera za uvođenje bitova koeficijenata u savijeno polje. Hardver za uvođenje koeficijenata biće sintetizovan korišćenjem matematičkih zavisnosti na kojima se zasniva dinamički način preslikavanja operacija na funkcionalne jedinice savijenog polja. Savijenom polju sa promenljivim brojem i dužinom koeficijenata biće dodata mogućnost promene broja operacija savijenih na jednu funkcionalnu jedinicu, što će biti predstavljeno u (5.3.8).

Šesto poglavlje je posvećeno implementaciji savijenih semi-sistoličkih polja za FIR filtriranje u FPGA (*Field Programable Gate Array*) tehnologiji. Savijena semi-sistolička polja za FIR filtriranje su u cilju ilustracije mogućnosti konfiguracije implementirana na *Spartan* II FPGA familiji čipova. Implementacija u FPGA tehnologiji pruža mogućnost nalaženja vremenskih karakteristika predloženih arhitektura, a umesto zauzetosti silicijumske površine dopušta praćenje zauzetosti resursa na čipu. Sve to dopušta proveru kvaliteta kompromisa prostor-vreme. Na početku poglavlja biće prikazana arhitektura i osnovne karakteristike *Spartan* familije FPGA. Rezultati

implementacije savijenih polja biće predstavljeni. Takođe, biće posvećena određena pažnja analizi brzine rada, propusnosti arhitekture i obimu zauzetih resursa za različite konfiguracione parametre filtra. Na kraju šestog poglavlja će biti dat uporedni pregled izvorne arhitekture i sintetizovanih savijenih arhitektura.

U sedmom poglavlju je dat zaključak, dok je u osmom poglavlju navedena lista korišćene literature.

2. DIGITALNO FILTRIRANJE

Pod obradom signala podrazumeva se skup operacija koje se izvršavaju nad signalom. Jedna od najvažnijih operacija u obradi signala je linearno filtriranje signala, kako zbog toga što nalazi veliku primenu u algoritmima za obradu signala, tako i zbog toga što ima relativno jednostavnu matematičku osnovu, pogodnu za implementaciju. Filtriranje se koristi u obradi audio i video signala, za razdvajanje signala sa različitim frekvencama, kombinovanje više signala u jedan signal, za otklanjanje šuma iz signala, izdvajanje signala u određenom frekventnom opsegu, predviđanje promena signala, podešavanje opsega kanala, otklanjanje eha, itd. Svi uređaji za snimanje i reprodukciju zvuka i slike, bez izuzetka, sadrže filtre.

U ovom poglavlju date su osnove diskretnog (digitalnog) filtriranja. Prikazana je operacija konvolucije signala. Matematičke osnove digitalnog filtriranja su date i objašnjene korišćenjem diskretnih diferencnih jednačina. Posebna pažnja posvećena je predstavljanju strukture osnovnih klasa digitalnih filtra.

2.1. KONVOLUCIJA I FILTRIRANJE SIGNALA

Pod obradom signala podrazumeva se preslikavanje jednog ili više ulaznih signala u izlazni signal. Preslikavanje signala može biti opisano operatorom Ψ, koji transformiše signal $x(n)$ u signal $y(n)$ na sledeći način:

$$y(n) = \Psi\{x(n)\}. \tag{2.1}$$

Postoje dva tipa preslikavanja Ψ. Prvi tip je *vremenski nezavisno* preslikavanje, kod koga vrednost signala $y(n)$ zavisi jedino od trenutne vrednosti signala $x(n)$. Drugi tip preslikavanja je *vremenski zavisno* preslikavanje. Vremenski zavisno preslikavanje je tip preslikavanja kod koga trenutna vrednost signala $y(n)$ zavisi, kako od trenutne vrednosti signala $x(n)$, tako i od vrednosti koje je signal $y(n)$ imao u prošlosti. Vremenski zavisno preslikavanje ima veliku primenu u digitalnoj obradi signala [15].

Kod vremenski zavisnih sistema može se javiti nestabilnost. Da bi vremenski zavistan sistem bio stabilan treba da zadovolji dva uslova: uslov linearnosti i uslov neosetljivost na fazu, tj. na pomeraj signala $x(n)$ u vremenu.

Neka su $x_1(n)$ i $x_2(n)$ dva nezavisna ulazna signala i neka važi $y_1(n)=\Psi\{x_1(n)\}$ i $y_2(n)=\Psi\{x_2(n)\}$. Ukoliko se operator Ψ primeni na linearnu kombinaciju signala $c_1x_1(n)+c_2x_2(n)$, gde su c_1 i c_2 nezavisni koeficijenti, i važi princip superpozicije:

$$y(n)=\Psi\{c1x_1(n)+c2x_2(n)\}=c1\Psi\{x_1(n)\}+c2\Psi\{x_2(n)\}=y_1(n)+y_2(n), \quad (2.2)$$

tada je operator Ψ linearan [15].

Neosetljivost operatora Ψ na fazu ulaznog signala se matematički definiše na sledeći način:

$$y(n-n_0)=\Psi\{x(n-n_0)\}. \quad (2.3)$$

Drugim rečima, ukoliko signal $x(n)$ kasni n_0 vremenskih jedinica, tada se signal $y(n)$ razlikuje u odnosu na signal $y(n)$, koji se dobija kada se na ulaz dovede signal bez kašnjenja, jedino u faznom pomeraju.

Konvolucija je linearna operacija, neosetljiva na fazu signala, koja preslikava dva signala, $x(n)$ i $h(n)$, u jedan signal:

$$y(n)=\Psi\{h(n),x(n)\}. \quad (2.4)$$

Konvolucija se definiše na sledeći način:

$$y(n)=\sum_{k=-\infty}^{\infty}h(k)x(n-k). \quad (2.5)$$

Signal $x(n)$ se naziva ulazni signal, a signal $h(n)$ filtrirajući signal. Filtrirajući signal je u većini praktičnih slučajeva periodičan, što u diskternom domenu predstavlja konačan niz vrednosti, koje se nazivaju koeficijenti filtra [16].

Suma u izrazu (2.5) je beskonačna, tako da je praktična implementacija ovog izraza nemoguća. Međutim, u većini praktičnih slučajeva se koristi aproksimacija izraza (2.5), tj. beskonačna suma u izrazu (2.5) se svodi na konačnu.

2.2. DIGITALNI FILTRI

Redukcijom sume u izrazu (2.5) na konačni opseg vrednosti, dolazi se do aproksimacije operacije konvolucije koju je moguće implementirati. Aproksimaciju operacije konvolucije implementiraju digitalni filtri. Digitalni filtri se opisuju diferencnim jednačinama. Ove jednačine predstavljaju aproksimaciju konvolucije i pogodne su za manipulacuiju u cilju nalaženja različitih modela filtra.

2.2.1. Diferencne jednačine

Diferencna jednačina koja daje vezu između izlaznog signala $y(n)$ i ulaznog signala $x(n)$ je:

$$\sum_{k=0}^{N} a_k y(n-k) = \sum_{k=0}^{M} b_k x(n-k), \qquad (2.6)$$

gde su $\{a_k\}$ i $\{b_k\}$ koeficijenti. Moguće je pokazati da jednakost (2.6) zadovoljava uslov linearnosti i uslov neosetljivosti na pomeraj faze [15].

U cilju jasnijeg predstavljanja, jednakost (2.6) može biti napisana u sledećem obliku:

$$y(n) = -\sum_{k=1}^{N} a_k y(n-k) + \sum_{k=0}^{M} b_k x(n-k). \qquad (2.7)$$

Iz izraza (2.7) sledi da je izlazni signal y u trenutku n linearna kombinacija N prethodnih vrednosti izlaznog signala i $M+1$ vrednosti ulaznog signala.

Veoma važan oblik jednačine (2.7) je specijalni slučaj koji se javlja za $N=0$. U ovom slučaju na izlazni signal utiče jedino ulazni signal, a ne i istorija izlaznog signala. Izraz (2.7) za slučaj $N=0$ ima sledeći oblik:

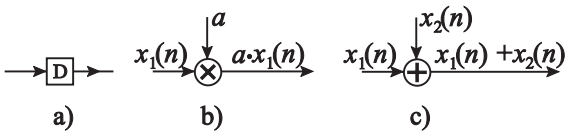
$$y(n) = \sum_{k=0}^{M} b_k x(n-k). \qquad (2.8)$$

Diferencna jednačina (2.8) predstavlja matematički opis digitalnog filtra sa konačnim impulsnim odzivom (*Finite Impulse Response* – FIR). Filtri kod kojih je $N>0$ se nazivaju filtri sa beskonačnim impulsnim odzivom (*Infinite Impulse Response*– IIR).

Diferencne jednačine (2.7) i (2.8) opisuju ponašanja filtra, međutim, ne postoji jednoznačno preslikavanje ovih jednačina u hardver, jer ne sadrže informaciju o redosledu izvršavanja operacija. Kako bi se u predstavljanju filtra što više približili hardverskoj implementaciji, uobičajeno je filtre predstavljati strukturom [15].

2.2.2. Struktura digitalnih sistema za filtriranje

Struktura filtra je grafička reprezentacija koja daje tačan prikaz redosleda izvršavanja operacija. Linearni filtri, neosetljivi na fazu signala, sastoje se od elementarnih operacija *kašnjenja, sabiranja* i *množenja*. Grafičke reprezentacije nabrojanih operacija prikazane su na sl. 2.1.



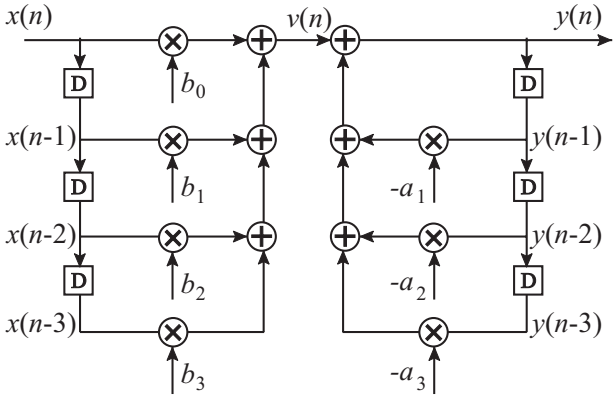
Slika 2.1. Grafičke reprezentacije elementarnih operacija filtra. a) Kašnjenje, b) množenje konstantom, c) sabiranje signala

U cilju ilustracije grafičkog predstavljanja strukture filtra, u daljem tekstu su date osnovne strukture IIR i FIR klase filtra, dobijene direktnim preslikavanjem jednačina u strukturu.

2.2.3. Direktna struktura IIR filtra

Na sl. 2.2 je prikazana direktna struktura IIR filtra, dobijena iz izraza (2.7). Izlazni signal IIR filtra, na osnovu izraza (2.7), zavisi od dve sume. U okviru jedne od ovih suma se nalaze vrednosti ulaznog signala u trenucima n,n-1,...,n-M, dok u drugoj sumi figurišu vrednosti izlaznog signala u trenucima n-1,n-2,...,n-N, što predstavlja povratnu spregu IIR filtra.

IIR filter se može implementirati tako što se na jednoj strani formira težinska suma vrednosti ulaznog signala u trenucima n, n-1, n-2,...,n-M, a na drugoj težinska suma vrednosti izlaznog signala u trenucima n-1, n-2,...,n-N. Na sl. 2.2 je prikazana struktura IIR filtra za N=3 i M=3 [15].



Slika 2.2. Direktna struktura IIR filtra

Signal koji se filtrira se dovodi u lanac kašnjenja, koji ima ulogu da ostatku kola prosledi vrednosti ulaznog signala $x(n-1)$, $x(n-2)$ i $x(n-3)$. Vrednost ulaznog signala x u trenutku n, $x(n)$, *i* vrednosti koje je ulazni signala imao u trenucima n-1, n-2 *i* n-3, a koje su sačuvane u lancu kašnjenja, se množe koeficijentima b_0, b_1, b_2 i b_3. Nakon množenja, dobijeni parcijalni proizvodi se sabiraju, što formira sumu proizvoda ulaznog signala i koeficijenata izraza (2.7).

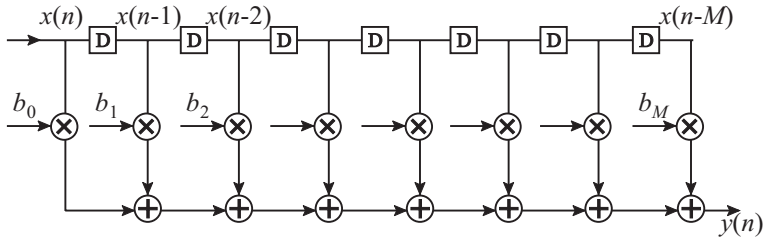
Vrednost izlaznog signala se, nakon određivanja vrednosti u trenutku n, smešta u lanac kašnjenja izlaznog signala na način prikazan na sl. 2.2, kako bi se vrednost sačuvala u narednih N taktnih intervala.

Na isti način na koji se formira suma proizvoda ulaznog signala i koeficijenata, formira se i suma parcijalnih proizvoda izlaznog signala i odgovarajućih koeficijenata. Ove dve sume se sabiraju, čime se dobija vrednost izlaznog signala $y(n)$ (sl. 2.2), u skladu sa izrazom (2.7).

IIR filter, predstavljen strukturom na sl. 2.2, moguće je implementirati, jer je način predstavljanja takav da sadrži sve potrebne informacije neophodne za implementaciju: redosled izvršavanja operacija, kašnjenja između operacija i veze.

2.2.4. Direktna struktura FIR filtra

Na sl. 2.3 je prikazana direktna struktura FIR filtra, gde reč "direktna" opisuje da je struktura formirana direktno na osnovu jednačine (2.8), bez dodatnih transformacija izraza. FIR filter je, na osnovu izraza (2.8), digitalni sistem bez povratne sprege. Strukturu filtra sa sl. 2.3 čini lanac kašnjenja, množači i sabirači [15].



Slika 2.3. Direktna struktura FIR filtra

Signal koji se filtrira, $x(n)$, se dovodi na ulaz lanca kašnjenja. Vrednost izlaznog signala y u trenutku n se dobija množenjem sadržaja lanca kašnjenja, $x(n-k)$, k=0,1,2,...,M, odgovarajućim koeficijentima, u skladu sa izrazom (2.8). Nakon množenja parcijalni rezultati se sabiraju, čime se na izlazim linijama dobija vrednost izlaznog signala $y(n)$.

FIR filter, čija je struktura prikazana na sl. 2.3, je pogodan za implementaciju zbog svoje regularne strukture. Regularna struktura filtra se ogleda u tome da se struktura može implementirati nizom funkcionalnih jedinica (FJ) koje rade sinhrono (svaka FJ se sastoji od jednog množača i jednog sabirača, sl. 2.3), ima regularne veze i eksploatiše protočnost [15].

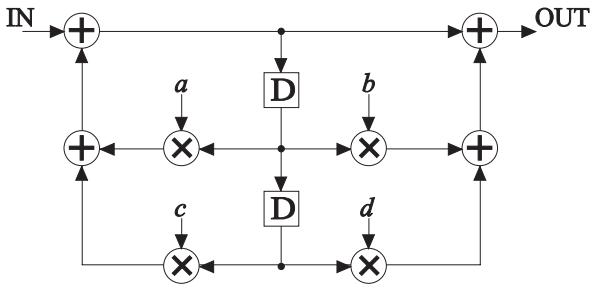
2.2.5. Struktura bikvadratnog filtra

Za razliku od FIR filtra, IIR filtri imaju bolje prenosne karakteristike. Međutim, kako izlazni signal zavisi, ne samo od ulaznog signala, već i od istorije izlaznog signala (povratna sprega), IIR filtri mogu relativno lako postati nestabilni. Kako bi se izbegla mogućnost da IIR filter dođe u nestabilno stanje, projektuju se filtri *malog reda*, odnosno filtri sa malim brojem koeficijenata. IIR filtri *višeg reda* se projektuju kao kaskadna veza nekoliko filtra nižeg reda [15].

Na sl. 2.4 je prikazan IIR filter drugog reda, koji se u literaturi sreće kao *bikvadratni filter*. Izraz "bikvadratni" je uobičajeno ime za digitalne filtre čija prenosna funkcija ima dva pola, tj. dve nule [15]. Prenosna funkcija filtra sa sl. 2.4 je:

$$G(z) = g\,\frac{1 + az^{-1} + cz^{-2}}{1 + bz^{-1} + dz^{-2}}\,.$$

Bikvadratni filter sa sl. 2.4 se zbog svoje jednostavne strukture koristi u literaturi u cilju ilustracije tehnike savijanja. Tehnika savijanja je prikazana u poglavlju 4, a primena tehnike savijanja je ilustrovana korišćenjem bikvadratnog filtra.



Slika 2.4. Struktura bikvadratnog filtra

Izabrana arhitektura za primenu tehnike savijanja je "*bit-plane*" FIR filter. U narednom poglavlju je prikazana struktura "*bit-plane*" polja, kao i princip filtriranja na ovom polju, dok će primena tehnike savijanja na "*bit-plane*" polju biti opisana u poglavlju 5.

3. "BIT-PLANE" FIR FILTER

Regularna struktura algoritma za FIR filtriranje je pogodna za implementaciju na sistoličkim poljima. Sistolička polja su procesorska polja sa jednostavnim funkcionalnim jedinicama koje rade sinhrono, imaju regularne veze i eksploatišu protočnost. Semi-sistolička polja su posebna klasa sistoličkih polja, kod kojih postoji izvesna neregularnost u povezanosti procesnih elemenata. Najčešće je reč o postojanju *broadcast* linije. Iskustva u implementaciji procesorskih polja u VLSI tehnologiji pokazala su da se veća propusnost sistema postiže kod semi-sistoličkih polja, jer je moguće ostvariti finiju protočnost u odnosu na implementacije sa potpuno-sistoličkim poljima. Neke od arhitektura koje se mogu svrstati u ovu grupu su: arhitektura sa paralelnim ulazom i serijskim izlazom, arhitektura bez akumulacije, "*bit-plane*" arhitektura i modifikovana "*bit-plane*" arhitektura. Zajednička karakteristika za ove arhitekture je da se mogu realizovati tako da imaju progrmabilne koeficijente [17].

"*Bit-plane*" (BP) arhitektura je zbog regularnosti arhitekture i dobrih karakteristika u pogledu propusnosti izabrana za polaznu arhitekturu u postupku sinteze savijenih polja za FIR filtriranje.

U ovom poglavlju prikazano je semi-sistoličko polje BP FIR filtra. Na samom početku biće prikazana arhitektura BP FIR filtra u dva nivoa detaljnosti. Prenosna funkcija BP FIR filtra biće data. Potom će biti prikazan funkcionalni opis arhitekture, nakon čega će biti izložena analiza zavisnosti dimenzija BP polja od karakterističnih parametara filtra.

3.1. ARHITEKTURA "BIT-PLANE" FIR FILTRA

Neka je dat niz koeficijenata $c_0,c_1, ...,c_{k-1}$ i neka je dat niz ulaznih reči $[x_i]$ koji se dovodi na ulaze FIR filtra. Tada se, u skladu sa izrazom (2.8), na izlazu FIR filtra dobija niz reči $[y_i]$ određen izrazom:

$$y_i = c_0x_i+c_1x_{i-1}+\ldots+c_{k-1}x_{i-k+1}. \qquad (3.1)$$

Struktura FIR filtra realizovanog BP arhitekturom prikazana je na sl. 3.1 ([6],[12]), gde je:

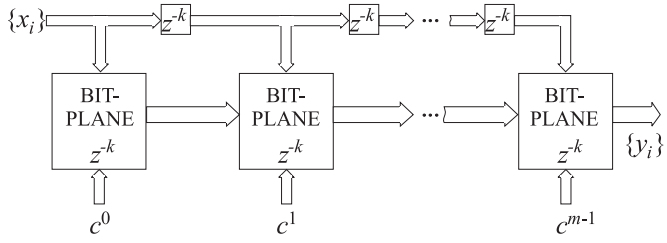
m – dužina koeficijenta (broj bitova u koeficijentu);

k – broj koeficijenata ($c_0,c_1,...,c_{k-1}$);

c^j – vektor od k bitova težine 2^j uzetih iz koeficijenata, odnosno ako važi $c_i \equiv c_i^{m-1}c_i^{m-2}...c_i^0$, gde su $c_i^0,c_i^1,...,c_i^{m-1}$ bitovi koeficijenta c_i težina $2^0,2^1,...,2^{m-1}$ redom, tada je $c^j \equiv c_{k-1}^j,c_{k-2}^j,...,c_0^j$, gde su $c_0^j,c_1^j,...,c_{k-1}^j$ bitovi težine 2^j koeficijenata $c_0,c_1,...,c_{k-1}$, respektivno;

z^{-k} – oznaka da je funkcionalnoj jedinici pridruženo kašnjenje od k taktnih intervala.

BP arhitektura je protočna, visoko-propusna semi-sistolička arhitektura sa regularnim vezama između elemenata polja. Proizvodi c_jx_{i-j} se izračunavaju paralelno i istovremeno se jednom učitana reč x_i višestruko koristi.



Slika 3.1. "*Bit-plane*" arhitektura

Karakteristično za BP arhitekturu sa sl. 3.1 je preuređenje parcijalnih proizvoda, tako da su množači koji realizuju proizvode c_jx_{i-j} zamenjeni BP elementima. U jednom BP elementu se dobija zbir parcijalnih proizvoda ulazne reči i bitova iste težine svih koeficijenata. Rezultat se prosleđuje na naredni BP gde se dodaje zbiru parcijalnih proizvoda naredne težine. Broj BP elemenata je jednak broju bitova u koeficijentu.

Funkcionalni blok dijagram BP FIR filtra sa 3 koeficijenta dužine 4 bita, i ulaznim rečima dužine 5 bitova je prikazana na sl. 3.2. Kao što se sa sl. 3.2 vidi, ova arhitektura je visokog stepena regularnosti i ima jednostavne veze između ćelija. Prenosna funkcija ovog filtra je:

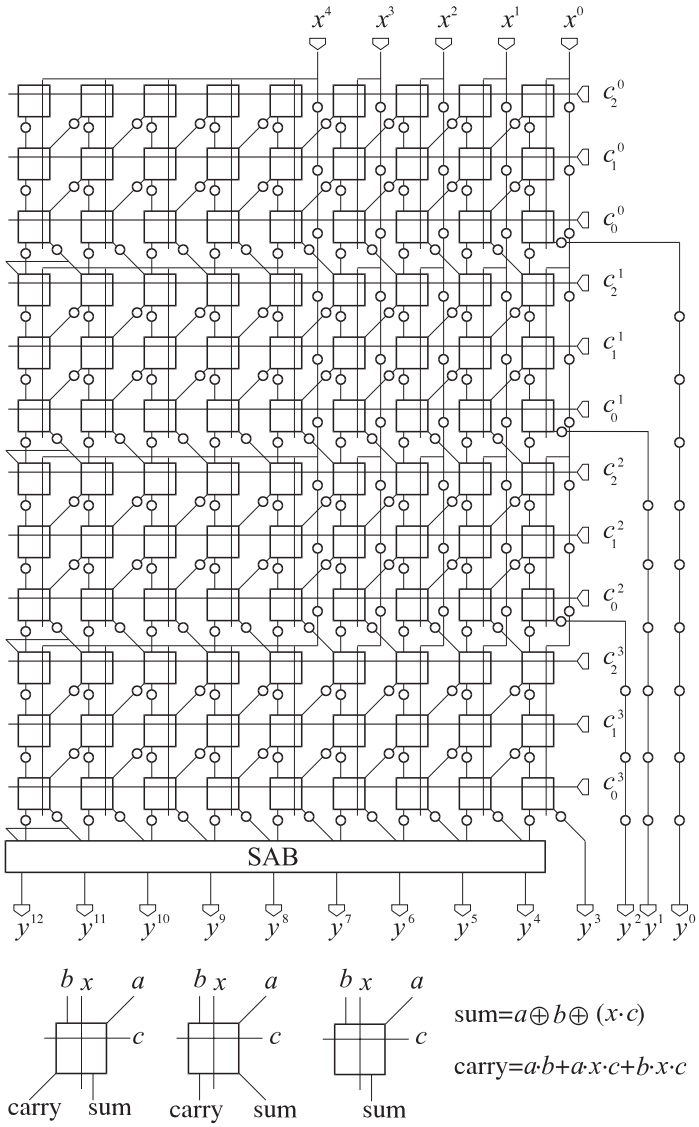
$$\begin{aligned}
G(z) &= z^{-1}(c_0{}^3 2^3 z^{-9}+z^{-1}(c_1{}^3 2^3 z^{-9}+z^{-1}(c_2{}^3 2^3 z^{-9}+\\
&+z^{-1}(c_3{}^3 2^3 z^{-9}+z^{-1}(c_0{}^2 2^2 z^{-6}+z^{-1}(c_1{}^2 2^2 z^{-6}+\\
&+z^{-1}(c_2{}^2 2^2 z^{-6}+z^{-1}(c_3{}^2 2^2 z^{-6}+z^{-1}(c_0{}^1 2^1 z^{-3}+\\
&+z^{-1}(c_1{}^1 2^1 z^{-3}+z^{-1}(c_2{}^1 2^1 z^{-3}+z^{-1}(c_3{}^1 2^1 z^{-3}+\\
&+z^{-1}(c_0{}^0 2^0+z^{-1}(c_1{}^0 2^0+z^{-1}(c_2{}^0 2^0+z^{-1}c_3{}^0 2^0)\ldots).
\end{aligned}$$

U opštem slučaju kada imamo k koeficijenta dužine m bitova i n-tobitne ulazne reči prenosna funkcija BP FIR filtra je:

$$\begin{aligned}
G(z) &= z^{-1}(c_0{}^{m-1} 2^{m-1} z^{-(m-1)k}+z^{-1}(c_1{}^{m-1} 2^{m-1} z^{-(m-1)k}+\ldots+\\
&+z^{-1}(c_{k-1}{}^{m-1} 2^{m-1} z^{-(m-1)k}+z^{-1}(c_0{}^{m-2} 2^{m-2} z^{-(m-2)k}+\ldots+\\
&+z^{-1}(c_{k-1}{}^1 2^1 z^{-k}+z^{-1}(c_0{}^0 2^0+\ldots+z^{-1}c_{k-1}{}^0 2^0)\ldots)=\\
&=z^{-(m-1)k}z^{-1}\Sigma_{i=0,k-1}(\Sigma_{j=0,m-1}c_i{}^j 2^j)z^{-i}=\\
&=z^{-(m^{-1})}k^{-1}\Sigma_{i=0,k-1}c_i z^{-i}. \qquad (3.2)
\end{aligned}$$

Inicijalno kašnjenje arhitekture je $m \cdot k$ taktnih intervala.

Funkcionalni blok dijagram BP FIR filtra sa sl. 3.2 je projektovan za ulazne reči x_i predstavljene u dvojnom komplementu, dok su koeficijenti neoznačeni brojevi. Kao što se sa slike vidi, ulazna reč je proširena bitom najveće težine (bitom znaka).



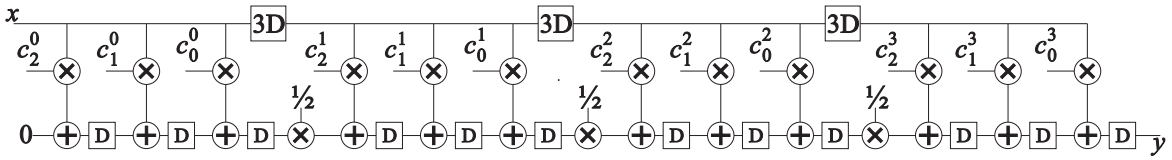
Slika 3.2. BP FIR filter se tri 4-bitna koeficijenta i 5-bitnim označenim ulaznim rečima (*k*=3, *m*=4, *n*=5)

Osnovni gradivni element BP polja je ćelija. Svaka ćelija ima 4 ulaza (a, b, x, y) i 2 izlaza (s i c) širine 1 bit (sl. 3.2). Funkcije koje realizuje ćelija su suma ($s=a\oplus b\oplus(x\cdot y)$) i prenos ($c=a\cdot b+a\cdot x\cdot y+b\cdot x\cdot y$). Horizontalni niz ćelija na sl. 3.2 čini jednu vrstu polja, a k vrsta formiraju jedan BP element. Svaka vrsta polja je zadužena za množenje ulaznog podatka jednim bitom koeficijenta, kao i za akumulaciju parcijalnih proizvoda. Prvi BP element obrađuje bitove najmanje težine, tako da je na njegovom izlazu moguće uzeti bit najmanje težine izlazne reči. Drugi BP element obrađuje bitove težine 2^1, i na njegovom izlazu je određen izlazni bit težine 2^1, itd. BP polje se sastoji od $k\cdot m$ vrsta osnovnih ćelija, odnosno m BP elementa. SAB je brzi sabirač.

U narednom odeljku je prikazan princip filtriranja na "*bit-plane*" semi-sistoličkom polju.

3.2. FIR FILTRIRANJE NA "BIT-PLANE" POLJU

Na sl. 3.3 je prikazan graf toka podataka (*Data Flow Graph* – DFG) BP arhitekture za slučaj k=3 i m=4.



Slika 3.3. BP arhitektura prikazana pomoću DFG-a, za k=3 i m=4

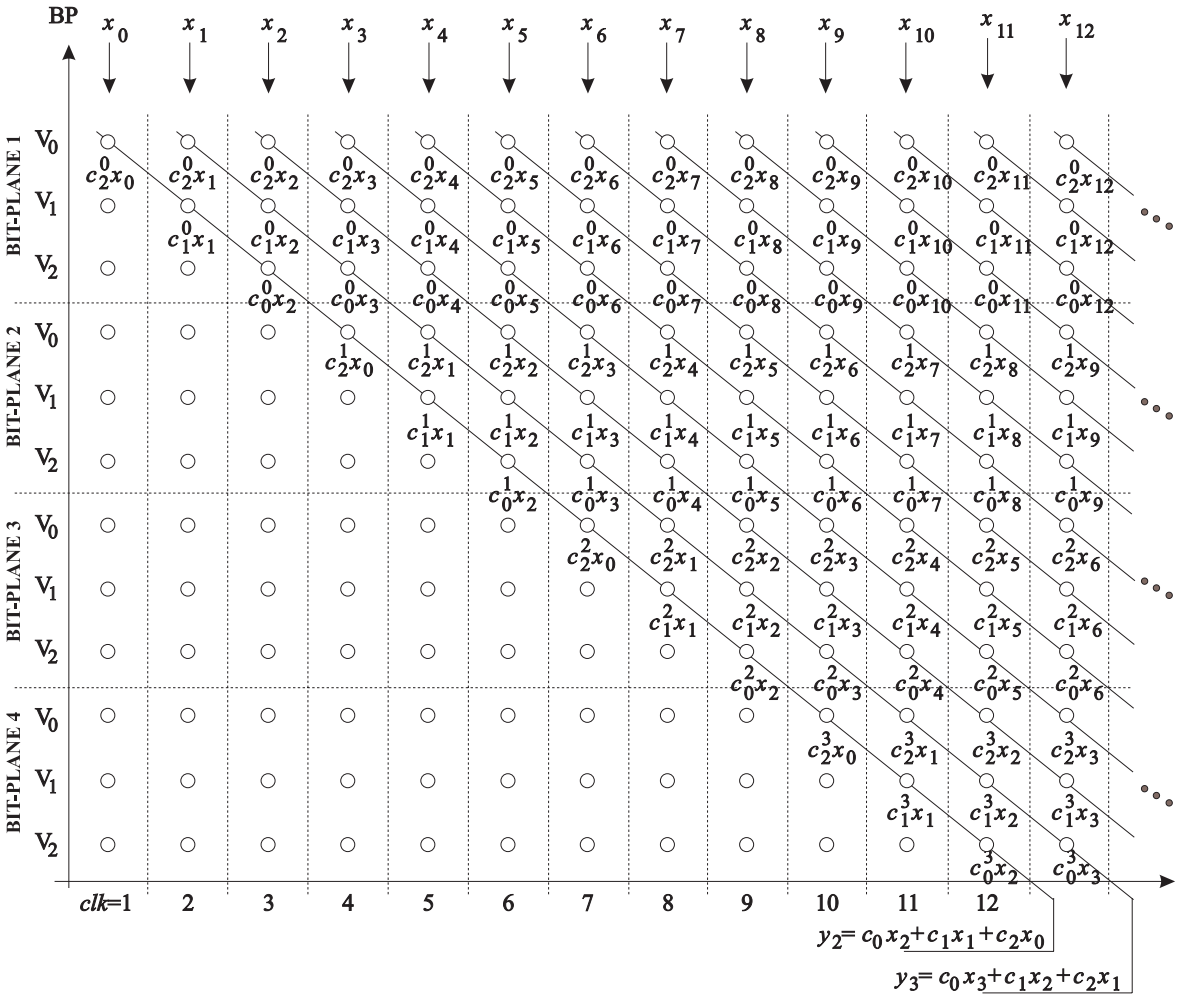
Jedan BP elemenat se sastoji od k=3 množača i isto toliko sabirača i kola za kašnjenje. Svaki od množača predstavlja jednu vrstu osnovnih ćelija arhitekture sa sl. 3.2. BP elementi su razdvojeni kolom za kašnjenje (kD) na putu ulaznih podataka i množačem na putu rezultata, koji množi međurezultat sa ½. Ovaj množač je realizovan pomeranjem među-rezultata za jedno mesto u desno (sl. 3.2).

Na sl. 3.4 prikazan je tok podataka kroz arhitekturu BP FIR filtra. Sa V_0, V_1 i V_2 na sl. 3.4 su označene prva, druga i treća vrsta osnovnih ćelija BP elemenata, respektivno. Na ulaz filtra se u prvom taktnom intervalu dovodi ulazna reč x_0 (sl. 3.4). Prva vrsta ćelija prvog BP elementa množi ovu ulaznu reč bitom težine 2^0 koeficijenta c_2 (sl. 3.4), tako da se u prvom taktnom intervalu na izlaznim linijama prve vrste ćelija dobija parcijalni proizvod $c_2^0x_0$. U narednom taktnom intervalu na ulaz filtra se dovodi ulazna reč x_1. Tada, druga vrsta ćelija prvog BP elementa određuje parcijalni proizvod $c_1^0x_1$ i ovom parcijalnom proizvodu dodaje među-rezultat koji je u prethodnom taktnom intervalu odredila prva vrsta ćelija istog BP elementa. Dalje, u narednom taktnom intervalu se na izlaznim linijama druge vrste ćelija nalazi suma parcijalnih proizvoda $c_2^0x_0+c_1^0x_1$. Treća vrsta prvog BP elementa ovoj sumi dodaje parcijalni proizvod $c_0^0x_2$, koji određuje u trećem taktnom intervalu. Na izlaznim linijama iz treće vrste se u trećem taktnom intervalu nalazi međurezultat $c_2^0x_0+c_1^0x_1+c_0^0x_2$. Time je prvi BP završio izračunavanje među-rezultata bitovima koeficijenata c^0. Među-rezultat $c_2^0x_0+c_1^0x_1+c_0^0x_2$ se tada množi sa ½, odnosno pomera u desno za jedno mesto i u četvrtom taktnom intervalu dovodi na ulaze prve vrste drugog BP elementa koji dodaje parcijalni proizvod $c_2^1x_0+c_1^1x_1+c_0^1x_2$, itd. Kao što je i ranije rečeno, inicijalno kašnjenje BP arhitekture je $m\cdot k$ taktnih intervala. Za slučaj k=3 i m=4 inicijalno kašnjenje kroz BP FIR filter je $m\cdot k$=12 taktnih intervala. Prva izlazna reč filtra, $y_2=c_2x_0+c_1x_1+c_0x_2$, je dostupna na izlaznim linijama arhitekture nakon dvanaestog taktnog intervala (sl. 3.4). BP FIR filter izlaznu reč y_2 određuje na sledeći način:

$$y_2=(c_2^0x_0+c_1^0x_1+c_0^0x_2)+(c_2^1x_0+c_1^1x_1+c_0^1x_2)+$$
$$+(c_2^2x_0+c_1^2x_1+c_0^2x_2)+(c_2^3x_0+c_1^3x_1+c_0^3x_2)$$
$$y_2=c_2x_0+c_1x_1+c_0x_2.$$

Paralelno sa izračunavanjem izlazne reči y_2 (protočno), počev od drugog taktnog intervala, počinje izračunavanje izlazne reči y_3 (sl. 3.4). U drugom taktnom intervalu prva vrsta ćelija prvog BP elementa računa parcijalni proizvod $c_2^0x_1$. U trećem taktnom intervalu druga vrsta prvog BP

elementa na ovaj parcijalni proizvod dodaje parcijalni proizvod $c_1^0 x_2$, itd. Na kraju trinaestog taktnog intervala na izlaznim linijama je dostupan rezultat $y_3=c_2x_1+c_1x_2+c_0x_3$ (sl. 3.4).



Slika 3.4. Tok podataka kroz BP FIR filter

Broj BP elemenata jednak je broju bitova u koeficijentu. Evidentno je da na drugu dimenziju polja, širinu, pored broja bitova u koeficijentu utiče i sam broj koeficijenata. U narednom odeljku je data analiza zavisnosti dimenzija BP semi-sistoličkog polja od parametara filtra.

3.3. DIMENZIJE "BIT-PLANE" POLJA

Neka je broj koeficijenata k, širina koeficijenta m, i neka su ulazne reči širine n bitova. Ukoliko su ulazne reči neoznačeni brojevi, tada važi $0 \le x_i \le 2^n-1$, pa je maksimalna vrednost rezultata koji se može dobiti na izlazu $k(2^m-1)(2^n-1)$. Kako je $k(2^m-1)(2^n-1) < k2^m2^n$ za broj bitova se može uzeti

$$\lceil \log_2 k2^m2^n \rceil = m+n+\lceil \log_2 k \rceil .$$

Ukoliko je x_i označen broj, predstavljen u dvojnom komplementu, i važi $-2^{n-1} \leq x_i \leq 2^{n-1}-1$, tada se izlazna reč nalazi u opsegu

$$k(2^m-1)(-2^{n-1}) \leq y_i \leq k(2^m-1)(2^{n-1}-1),$$

te se za broj bitova može uzeti

$$\lceil \log_2 k(2^m-1)2^{n-1} \rceil.$$

Ovom je potrebno dodati i jedan bit za predstavljanje znaka, pa je broj potrebnih bitova

$$m+n-1+\lceil \log_2 k \rceil+1=m+n+\lceil \log_2 k \rceil.$$

S obzirom da se na izlazu svakog BP elementa dobija po jedan bit, znači ukupno m, to je potrebna širina BP elementa, kada se radi sa neoznačenim brojevima, jednaka:

$$L_N=m+n+\lceil \log_2 k \rceil-m=n+\lceil \log_2 k \rceil.$$

Kada se radi sa označenim ulaznim vrednostima potrebna širina je veća. Naime, sabiranjem dva ili više n-to bitna označena broja suprotnog znaka broj bitova za predstavljanje rezultata može biti manji od n. Međutim, ako se kao rezultat sabiranja dobijaju bitovi sume s i prenosa c tada je broj bitova za predstavljanje prenosa n+1, što je ilustrovano sledećim primerom:

$$\begin{array}{rl} & 0\,0\,1\,1 \equiv 3 \\ & 1\,0\,1\,0 \equiv -10 \\ & \underline{1\,1\,1\,1 \equiv -1} \\ s & 0\,1\,1\,0 \equiv 6 \\ c & \underline{1\,0\,1\,1\,0 \equiv -10} \\ & 1\,1\,1\,0\,0 \equiv -4 \end{array}$$

Zbog toga je broj potrebnih ćelija u svakoj narednoj vrsti veći za jedan. Kako jedan BP element ima k vrsta, to je svaki naredni BP potrebno proširiti za k kolona. Naravno, kako se izlazni element može predstaviti sa

$$\mathrm{brmax}=m+n+\lceil \log_2 k \rceil$$

bitova, nije potrebno izračunavati bitove veće težine jer su jednaki bitu težine $m+n+\lceil \log_2 k \rceil-1$. Na sl. 3.5 je prikazano kako raste broj potrebnih ćelija za k=3, m=4 i n=5.

Broj bitova koji se dobijaju na izlazu prvog BP elementa je $n+k-2$ (jer se na ulazu prve vrste dovode vrednosti 0), a na svakom narednom nivou je za k bitova veći, tako da važi

$$\mathrm{br}(i)=n+ik-2 \qquad (3.3)$$

gde je i redni broj BP elementa (i=1,...,m). Rešavanjem jednačine br(i)=brmax može se dobiti redni broj BP elementa sa maksimalnom širinom (A), na sledeći način:

$$n+ik-2=n+m+\lceil \log_2 k \rceil,$$

$$i=(m+2+\lceil \log_2 k \rceil)/k,$$

odnosno

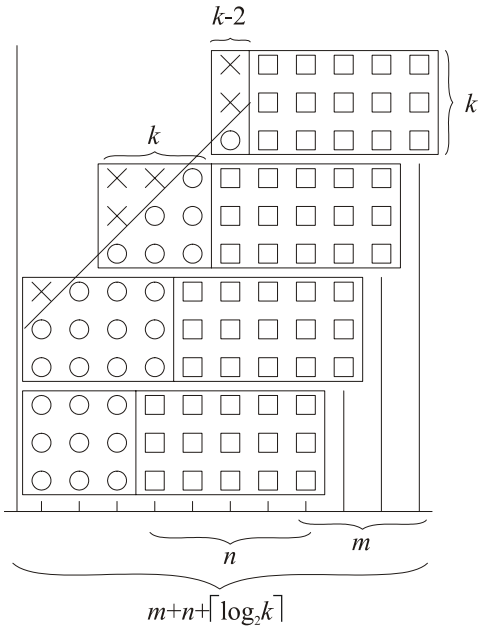
$$\mathrm{A}=\lceil (m+2+\lceil \log_2 k \rceil)/k \rceil,$$

gde je A redni broj BP elementa na čijem se izlazu prema (3.3) dobija više bitova od potrebnog broja. Kako se na izlazu prvog BP elementa dobija bit najmanje težine, na drugom bit naredne težine itd, to je svakom narednom BP elementu potrebna po jedna kolona manje, pa je broj ćelija u jednoj BP vrsti jednak:

$$\mathrm{brc}(i)=\mathrm{br}(i)-(i-1) \quad \text{za } i < \mathrm{A},$$

odnosno

$$\mathrm{brc}(i)=\mathrm{brmax}-i+1, \quad \text{za } i \geq \mathrm{A}.$$



Slika 3.5. Proširivanje polja ćelija

Maksimalna širina je za i=A i iznosi

$$\mathrm{brcmax}=m+n+\lceil \log_2 k \rceil - \mathrm{A}+1.$$

Širina BP elementa kada se radi sa označenim brojevima je jednaka maksimalnoj širini i iznosi

$$\mathrm{L_O}=m+n+\lceil \log_2 k \rceil - \mathrm{A}+1,$$

a posle zamene A i sređivanja

$$\mathrm{L_O}=\lfloor (m+2+\lceil \log_2 k \rceil)(k-1)/k \rfloor + n - 1. \qquad (3.4)$$

Za primer sa sl. 3.2 je k=3, m=4, n=5, pa je A=⌈(4+2+⌈log23⌉)/3⌉=3, odnosno $\mathrm{L_O}$=brcmax=4+5+2-3+1=9. Zbog regularnosti arhitekture svi BP elementi su iste širine. Podjednaka širina svih BP elemenata će omogućiti jednostavniju primenu tehnike savijanja.

Ukupan broj ćelija je $\mathrm{N_C}=\mathrm{L_O}k \cdot m$, a ukupan broj flip-flopova je $\mathrm{N_{FF}}=\mathrm{N_{FF1}}+\mathrm{N_{FF2}}$, gde je $\mathrm{N_{FF1}}$ broj flip-flopova koji služe za postizanje protočnosti, za bitove sume s i prenosa c, a $\mathrm{N_{FF2}}$ broj flip-flopova koji služe za ulazne reči x_i:

$$\mathrm{N_{FF1}}=2\mathrm{N_C}-(k-1)m, \text{ i}$$

$$\mathrm{N_{FF2}}=(m-1)k \cdot n.$$

Ukupan broj flip-flopova je:

$$N_{FF}=2(\lfloor(m+2+\lceil\log_2 k\rceil)(k-1)/k\rfloor+n-1)k\cdot m-(k-1)m+(m-1)k\cdot n.$$

Ovom broju je potrebno dodati flip-flopove za *i*, kojih je $k\cdot m$, kao i flip-flopove koji razdvajaju stepene protočnosti sabirača ukoliko je realizovan protočno.

Pri sintezi DSP arhitekture važno je minimizirati površinu silicijuma integrisanog kola, što se može postići redukovanjem broja funkcionalnih jedinica, registara, multipleksera i veza.

U narednom poglavlju prikazana je tehnika savijanja, čijom primenom je moguće minimizirati površinu silicijuma potrebnu za implementaciju kola na račun brzine rada kola. Opisan je koncept savijanja, dat je matematički model tehnike, prikazane su tehnike za vremensko usklađivanje rada arhitekture i tehnike za minimizaciju broja registara u savijenoj arhitekturi.

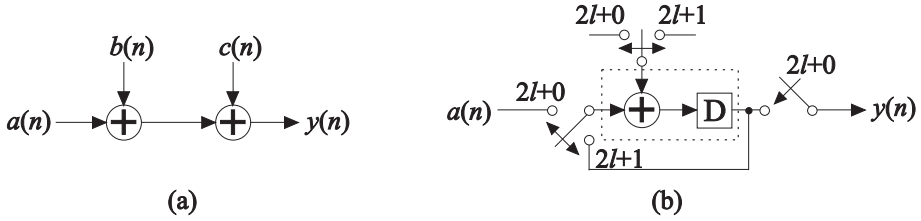
4. TEHNIKA SAVIJANJA

Ovo poglavlje daje teoretsku osnovu za primenu tehnike savijanja u projektovanju specijalizovanih sistema za digitalnu obradu signala. Poglavlje počinje jednostavnim primerom na koji je tehnika savijanja primenjena *ad hoc*. Nakon toga sledi matematički opis savijanja i vremenskog usklađivanja za zadato savijanje. Potom će biti izložena tehnika za projektovanje DSP arhitektura koje koriste najmanji mogući broj registara. Na kraju poglavlja prikazane tehnike biće ilustrovane kroz primer savijanja bikvadratnog filtra i IIR filtra.

4.1. KONCEPT TEHNIKE SAVIJANJA

Transformacija savijanja se koristi za vremensko multipleksiranje više operacija DSP algoritma na jednu funkcionalnu jedinicu. Izvršavanjem više operacija na jednoj funkcionalnoj jedinici broj funkcionalnih jedinica se redukuje, što smanjuje potrebnu površinu silicijuma za implementaciju kola.

Na sl. 4.1 je dat primer kako dve operacije mogu biti vremenski multipleksirane na jednom protočnom sabiraču. DSP algoritam na sl. 4.1(a) izračunava $y(n)=a(n)+b(n)+c(n)$. Na sl. 4.1(b), dve operacije sabiranja prikazane na sl. 4.1(a) su vremenski multipleksirane na jednom protočnom sabiraču. Hardver koji radi u vremenskom multipleksu na sl. 4.1 (b), radi na sledeći način: u ciklusu 0 vrednosti $a(0)$ i $b(0)$ se propuštaju u sabirač, i suma $a(0)+b(0)$ se smešta u memorijski element; u ciklusu 1 suma $a(0)+b(0)$ se vraća na jedan ulaz sabirača, a na drugi ulaz se dovodi $c(0)$. Suma $a(0)+b(0)+c(0)$ se smešta u memorijski element iz koga se u ciklusu 2 prosleđuje na izlaz, a sabirač istovremeno izračunava međurezultat $a(1)+b(1)$. Ovaj proces se nastavlja na način prikazan u tab. 4.1. Po jedan izlazni element se dobija na svaka dva taktna intervala. Takođe, sa ulaza se elementi propuštaju u sabirač na svaka dva taktna intervala ($a(i)$ i $b(i)$ u ciklusu $2i$, a $c(i)$ u ciklusu $2i+1$). Zbog toga vrednosti koje se dovode na ulaz kola moraju biti prisutne dva taktna intervala pre promene. U opštem slučaju, podaci na ulazu arhitekture na koju je primenjena tehnika savijanja se menjaju na svakih N ciklusa, gde je N broj operacija koje se izvršavaju u vremenskom multipleksu na jednoj funkcionalnoj jedinici.



Slika 4.1 (a) Jednostavni DSP algoritam sa dve operacije sabiranja. (b) Savijena arhitektura u kojoj se operacije sabiranja izvršavaju na jednom sabiraču sa jednim stepenom protočnosti

Savijanje omogućava pronalaženje kompromisa između površine i brzine DSP arhitekture. Jedan način za implementaciju DSP algoritma sa sl. 4.1(a) je korišćenjem dva sabirača sa dodatnim memorijskim elementom kojim se postiže protočnost. Ova implementacija zahteva dva sabirača i izračunava po jednu iteraciju algoritma u toku vremenskog intervala potrebnog za obavljanje sabiranja, T_{ADD}. S druge strane implementacija sa savijanjem na sl. 4.1(b) koristi jedan sabirač i izračunava po jednu iteraciju algoritma u vremenskom intervalu $2T_{\mathrm{ADD}}$. U opštem slučaju savijanje se može koristiti za smanjenje broja funkcionalnih jedinica za faktor N, na račun povećanja vremena izračunavanja za isti faktor N. Dva krajnja slučaja su: potpuno paralelna implementacija, kod koje je svakoj operaciji pridružena po jedna funkcionalna jedinica, i korišćenje samo jedne procesorske jedinice, kod koje je ceo algoritam je implementiran na jednoj funkcionalnoj jedinici.

Tabela 4.1. Prikaz operacija u prvih šest ciklusa za savijeni hardver sa sl. 4.1(b)

Ciklus	Levi ulaz u sabirač	Gornji ulaz u sabirač	Izlaz
0	$a(0)$	$b(0)$	-
1	$a(0)+b(0)$	$c(0)$	-
2	$a(1)$	$b(1)$	$a(0)+b(0)+c(0)$
3	$a(1)+b(1)$	$c(1)$	-
4	$a(2)$	$b(2)$	$a(1)+b(1)+c(1)$
5	$a(2)+b(2)$	$c(2)$	-

Arhitektura sa sl. 4.1(b) je dovoljno jednostavna da bi mogla da se projektuje *ad hoc*. Međutim, DSP algoritmi su mnogo složeniji od algoritma sa sl. 4.1(a), pa je tehniku savijanja neophodno primeniti na sistematski način.

4.2. MATEMATIČKI MODEL TEHNIKE SAVIJANJA

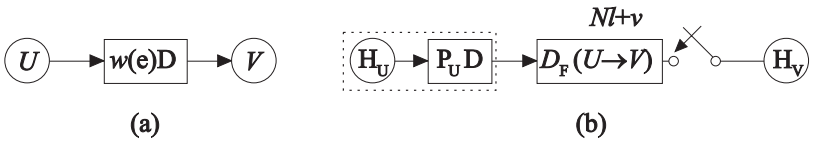
Transformacija savijanja obezbeđuje sistematsku tehniku za projektovanje upravljačke logike za hardver kod koga je nekoliko operacija vremenski multipleksirano na jednu funkcionalnu jedinicu. U ovom odeljku je opisano izvođenje jednačine savijanja, koja predstavlja osnovu ove tehnike, zajedno sa izvođenjem vremenskog usklađivanja (*retiming*).

Posmatrajmo poteg (e) koji spaja čvorove (operacije) U i V, kome je pridruženo kašnjenje w(e), kao što je prikazano na sl. 4.2(a). Neka se l-ta iteracija operacije U i V izvršava u vremenskim

jedinicama $N \cdot l+u$ i $N \cdot l+v$, respektivno, gde u i v predstavljaju red savijanja čvorova U i V, i zadovoljavaju uslov $0 \le u,v \le N-1$. Red savijanja čvora (odnosno operacije) je vremenski interval u kome je podešeno da funkcionalna jedinica izvršava operaciju. FJ koje izvršavaju operacije U i V su označene sa H_U i H_V. N je broj operacija savijenih na jednu FJ i naziva se i faktor savijanja. Ako je FJ H_U protočna, sa P_U stepena protočnosti, tada je rezultat l-te iteracije operacije U dostupan u vremenskom intervalu $N \cdot l+u+P_U$. Kako je potegu $U \rightarrow V$ pridruženo kašnjenje od $w(e)$, to se rezultat l-te iteracije operacije U koristi u $l+w(e)$-toj iteraciji operacije V, i izvršava se tokom intervala $N(l+w(e))+v$. Zbog toga rezultat mora biti sačuvan tokom

$$D_F(U \rightarrow V) = [N(l+w(e))+v] - [N \cdot l + P_U + u] = Nw(e) - P_U + v - u \qquad (4.1)$$

vremenskih jedinica, i ovaj interval ne zavisi od rednog broja iteracije l. Izraz (4.1) se u literaturi sreće kao *jednačina savijanja*. Poteg $U \rightarrow V$ je implementiran u arhitekturi kao putanja od H_U do H_V sa kašnjenjem od $D_F(U \rightarrow V)$ vremenskih jedinica, a podaci na ovoj putanji ulaze u H_V u vremenskoj jedinici $N \cdot l+v$, kao što je ilustrovano na sl. 4.2 (b). Da bi savijeni sistem mogao da se realizuje, mora da važi $D_F(U \rightarrow V) \ge 0$ za sve potege u DFG. Uslov $D_F(U \rightarrow V) \ge 0$ se naziva uslov savijanja.



Slika 4.2. (a) Poteg $U \rightarrow V$ sa kašnjenjem $w(e)$. (b) Odgovarajući savijeni put podataka. Podaci polaze iz funkcionalne jedinice H_U koja ima P_U stepena protočnosti, kasne $D_F(U \rightarrow V)$ vremenskih jedinica, i propuštaju se u funkcionalnu jedinicu H_V u vremenskim jedinicama $N \cdot l+v$.

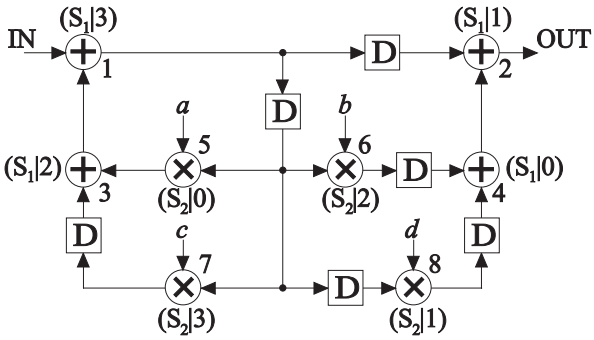
Skup savijanja je uređen skup operacija koje se izvršavaju na istoj funkcionalnoj jedinici. Svaki skup savijanja sadrži po N elemenata, pri čemu neki od tih elemenata mogu biti *null* operacije. Operacija na poziciji j u skupu savijanja (gde j uzima vrednost od 0 do N-1) se izvršava na FJ tokom vremenskog intervala $N \cdot l+j$. Na primer, neka je dat skup savijanja $S_1=[A_1,0,A_2]$ za $N=3$. Operacija pripada skupu savijanja S_1, sa redom savijanja 0 (što se označava sa $(S_1 \mid 0)$), i operacija A_2 pripada skupu savijanja S_1 sa redom savijanja 2 (odnosno $(S_1 \mid 2)$). Kako se *null* operacija nalazi na poziciji 1 u skupu S_1, FJ koja izvršava operacije A_1 i A_2 neće biti korišćena tokom vremenskih jedinica $3l+1$.

Sistematska primena tehnike savijanja je prikazana na primeru savijanja bikvadratnog filtra, prikazanog u poglavlju 2.2.5 (sl. 2.4).

4.2.2. Primer savijanja bikvadratnog filtra

Bikvadratni filter sa sl. 2.4 (poglavlje 2.2.5) je u cilju uspešne primene tehnike savijanja vremenski usklađen i prikazan na sl. 4.3. Tehnikom vremenskog usklađivanja arhitekture se, uz zadržavanje tačnosti izračunavanja, menjaju kašnjenja u granama DFG-a kako bi se zadovoljio uslov savijanja $D_F(U \rightarrow V) \ge 0$. Tehnika vremenskog usklađivanja biće prikazana u narednom poglavlju.

Neka su operacije sabiranja i množenja realizovane protočno, i to sabiranje sa jednim stepenom protočnosti, a množenje sa dva stepena protočnosti (tj. P_S=1 i P_M=2). FJ se mogu taktovati taktnim signalom čija je perioda jednaka jednoj vremenskoj jedinici. Filter je savijen sa faktorom savijanja N=4, i skupovima savijanja prikazanim na sl. 4.3. Faktor savijanja je N=4, što znači da je period iteracije savijenog hardvera 4 vremenske jedinice, tj. svaka operacija bikvadratnog filtra se izvršava tačno jednom na svake 4 vremenske jedinice u savijenoj arhitekturi. Ovo takođe znači da funkcionalna jedinica na savijenom hardveru izvršava 4 operacije DSP algoritma. Skupovi savijanja sa sl. 4.3 mogu biti napisani na sledeći način S_1=[4,2,3,1] i S_2=[5,8,6,7].



Slika 4.3. Vremenski usklađen bikvadratni filter sa dodeljenim skupovima savijanja

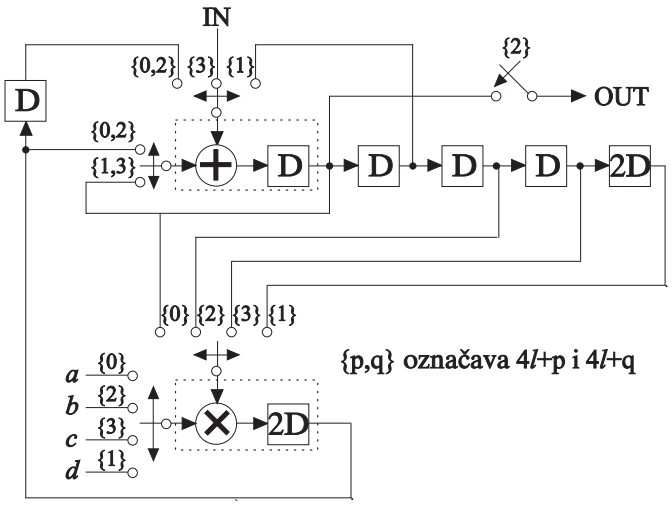
Skup savijanja S_1 sadrži samo operacije sabiranja, koje se realizuju na istom sabiraču. Slično, skup savijanja S_2 sadrži samo operacije množenja, koje se izvršavaju na istom množaču. Na primer, čvor 3 se izvršava u savijenoj arhitekturi u vremenskim intervalima 4l+2 na sabiraču na kome su implementirane operacije iz skupa savijanja S_1.

Jednačine savijanja za DFG sa sl. 4.3 su:

$$
\begin{aligned}
D_F(1\rightarrow 2) &= 4(1)-1+1-3 = 1\\
D_F(1\rightarrow 5) &= 4(1)-1+0-3 = 0\\
D_F(1\rightarrow 6) &= 4(1)-1+2-3 = 2\\
D_F(1\rightarrow 7) &= 4(1)-1+3-3 = 3\\
D_F(1\rightarrow 8) &= 4(2)-1+1-3 = 5\\
D_F(3\rightarrow 1) &= 4(0)-1+3-2 = 0\\
D_F(4\rightarrow 2) &= 4(0)-1+1-0 = 0\\
D_F(5\rightarrow 3) &= 4(0)-2+2-0 = 0\\
D_F(6\rightarrow 4) &= 4(1)-2+0-2 = 0\\
D_F(7\rightarrow 3) &= 4(1)-2+2-3 = 1\\
D_F(8\rightarrow 4) &= 4(1)-2+0-1 = 1
\end{aligned}
\qquad (4.2)
$$

Za sve potege u DFG-u je ispunjen uslov savijanja $D_F(U\rightarrow V)\geq 0$, pa je DFG sl. 4.3 moguće saviti. Savijena arhitektura je prikazana na sl. 4.4, a dobijena je korišćenjem DFG-a sa sl. 4.3 i jednačina savijanja (4.2). Na primer, vrednost kašnjnja potega $1\rightarrow 8$ na osnovu (4.2) je $D_F(1\rightarrow 8)=5$, što znači da postoji poteg od sabirača do množača u savijenom DFG-u sa kašnjenjem 5. Kako se

ovaj poteg završava u čvoru 8, koji ima red savijanja 1 (sl. 4.3), u savijenom DFG-u poteg se dovodi na ulaz tokom intervala $4l+1$. Ovaj savijeni poteg je prikazan na sl. 4.4.



Slika 4.4. Arhitektura savijenog bikvadratnog filtra

Ukoliko nakon formiranja skupova savijanja uslov savijanja $D_F(U\rightarrow V)\geq 0$ nije ispunjen za sve čvorove u grafu, može se sprovesti procedura za vremensko usklađivanje kako bi se zadovoljio ovaj uslov, ili kako bi se ustanovilo da zadati skup savijanja ne može biti realizovan.

4.3. VREMENSKO USKLAĐIVANJE

Vremenskim usklađivanjem (*retiming*) potega $U\rightarrow V$ se kašnjenje ovog potega, w(e), menja na vrednost

$$w_r(e)=w(e)+r(V)-r(U), \qquad (4.3)$$

gde je w_r(e) kašnjenje potega $U\rightarrow V$ u vremenski usklađenom DFG-u, r(X) predstavlja vrednost vremenskog usklađenja za čvor X, odnosno vrednost za koju je potrebno smanjiti kašnjenje izlaznog potega i povećati vrednost kašnjenja ulaznih potega čvora X. Neka D_F'$(U\rightarrow V)$ predstavlja kašnjenje potega dobijenog vremenskim usklađivanjem DFG-a. Kako bi se obezbedilo da odgovarajući poteg u savijenoj arhitekturi ima nenegativno kašnjenje, mora da bude ispunjen uslov D_F'$(U\rightarrow V)\geq 0$, odnosno

$$Nw_r(e)-P_U+v-u\geq 0 \qquad (4.4)$$

Ovaj uslov garantuje da je skup savijanja valjan nakon vremenskog usklađivanja. Kombinovanjem (4.3) i (4.4) dobija se

$$N(w(e)+r(V)-r(U))-P_U+v-u\geq 0.$$

Zamenom Nw(e)-P_U+v-u sa $D_F(U\rightarrow V)$ i rešavanjem po r(U)-r(V) dobija se

$$r(U)-r(V)\leq D_F(U\rightarrow V)/N,$$

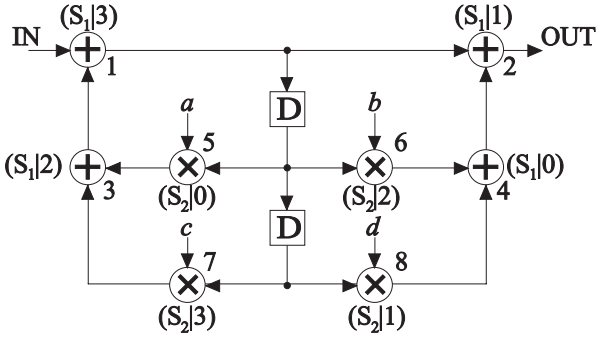
Kako su vrednosti za vremensko usklađivanje celobrojni umnošci vremenskih jedinica, to se prethodna nejednakost može pisati kao

$$r(U)\text{-}r(V) \leq \lfloor D_F(U \rightarrow V)/N \rfloor, \qquad (4.5)$$

gde $\lfloor x \rfloor$ predstavlja najveći ceo broj manji ili jednak broju x.

Na osnovu skupa uslova za DFG, dobijenog na osnovu (4.5) (postoji po jedan uslov za svaki poteg u grafu), može se odrediti da li postoji rešenje za vremensko usklađenje, i ako postoji može se pronaći.

Kao primer uzmimo DFG bikvadratnog filtra prikazanog u poglavlju 2.2.5 na sl. 2.4. DFG bikvadratnog filtra sa dodeljenim skupovima savijanja je prikazan na sl. 4.5. Pretpostavimo da imamo na raspolaganju sabirače sa jednim stepenom protočnosti i množače sa dva stepena protočnosti (tj. P_S=1 i P_M=2). Jednačine savijanja (4.1) i vremenskog usklađivanja (4.5) za DFG sa sl. 4.5 su date u tab. 4.2.



Slika 4.5. Originalni DFG bikvadratnog filtra sa dodeljenim skupovima savijanja.

Sistem nejednačina sa uslovima se može rešiti po r(X) pomoću grafa ograničenja. Ovaj graf se formira tako što se ucrtaju svi čvorovi (ako je broj čvorova k i ako su obeleženi celim brojevima onda su to čvorovi 1 do k). Zatim se za svaku nejednačinu $r(k_1)\text{-}r(k_2) \leq T$ ucrta poteg od k_2 do k_1 i dodeli mu se težina T. Na kraju se ucrta čvor k+1 i spoji se potezima sa svim ostalim čvorovima (1 do k), kojima se dodeli težina 0. Graf ograničenja za nejednačine iz desne kolone tab. 4.2 je prikazan na sl. 4.6. Ako u grafu postoji jedna ili više kontura tako da je zbir težina svih potega u toj konturi negativan, tada ne postoji rešenje za vremensko usklađivanje za zadate uslove savijanja. U protivnom rešenje postoji, i za r(i) se uzima najkraći put između čvorova 9 (odnosno k+1 u opštem slučaju) i i.

Algoritmi, kao što su Bellman-Ford-ov ili Floyd-Warshall-ov se mogu koristiti za utvrđivanje da li postoji kontura sa negativnom težinom, i ako ne postoji, za određivanje najkraćeg puta između čvorova. Graf ograničenja sistema nejednačina datog u tab. 4.2 za vremensko usklađivanje DFG-a sa sl. 4.5, je prikazan na sl. 4.6. Korišćenjem bilo kog od pomenutih algoritama za rešavanje grafa ograničenja, dobijamo da rešenje za sistem nejednačina iz tab. 4.2 postoji, i kao jedno moguće rešenje dobijamo

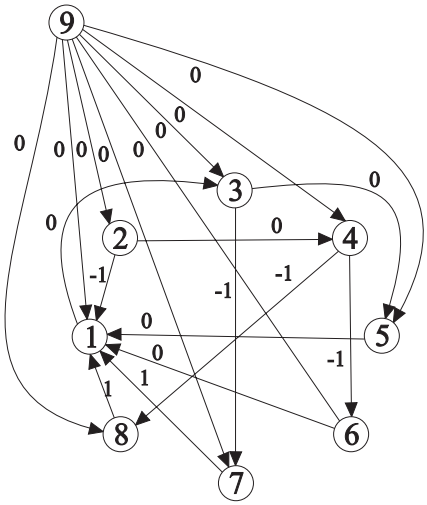
r(1)=-1, r(2)=0, r(3)=-1, r(4)=0, r(5)=-1, r(6)=-1, r(7)=-2 i r(8)=-1.

DFG na kome je primenjeno vremensko usklađivanje je prikazan na sl. 4.3, jednačine savijanja za ovaj graf su date sa (4.2). Savijeni DFG je prikazan na sl. 4.4.

Tabela 4.2. Jednačine savijanja i uslovi za vremensko usklađivanje za DFG sa sl. 4.5.

poteg	jednačina savijanja	vremensko usklađivanje
1→2	$D_F(1\rightarrow2) = -3$	r(1)-r(2) ≤ −1
1→5	$D_F(1\rightarrow5) = 0$	r(1)-r(5) ≤ 0
1→6	$D_F(1\rightarrow6) = 2$	r(1)-r(6) ≤ 0
1→7	$D_F(1\rightarrow7) = 7$	r(1)-r(7) ≤ 1
1→8	$D_F(1\rightarrow8) = 5$	r(1)-r(8) ≤ 1
3→1	$D_F(3\rightarrow1) = 0$	r(3)-r(1) ≤ 0
4→2	$D_F(4\rightarrow2) = 0$	r(4)-r(2) ≤ 0
5→3	$D_F(5\rightarrow3) = 0$	r(5)-r(3) ≤ 0
6→4	$D_F(6\rightarrow4) = -4$	r(6)-r(4) ≤ −1
7→3	$D_F(7\rightarrow3) = -3$	r(7)-r(3) ≤ −1
8→4	$D_F(8\rightarrow4) = -3$	r(8)-r(4) ≤ −1

Iako transformacija savijanja smanjuje broj funkcionalnih jedinica u arhitekturi ona može dovesti do rešenja koje koristi veliki broj registara. Da bi se broj registara u savijenoj arhitekturi smanjio mogu se koristiti tehnike za izračunavanje minimalnog broja registara potrebnog za implementaciju savijene DSP arhitekture, nakon čega je potrebno primeniti i tehnike za razmeštaj podataka u registre. U narednom poglavlju prikazane su tehnike za minimizaciju broja registara u savijenim arhiitekturama.



Slika 4.6. Graf ograničenja za skup nejednačina u desnoj koloni tab. 4.2.

4.4. TEHNIKE ZA MINIMIZACIJU BROJA REGISTARA

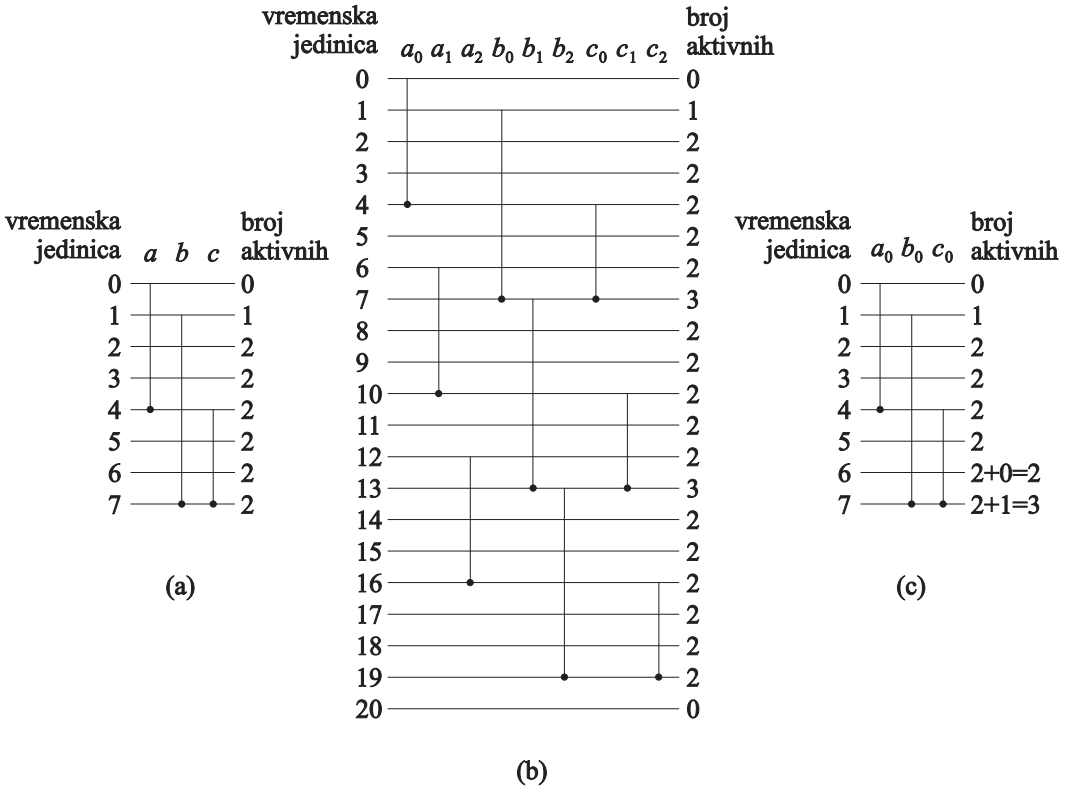
Površina silicijuma se dodatno može smanjiti minimizacijom broja registara koji se koriste u DSP arhitekturi. U ovom odeljku je opisana tehnika za određivanje minimalnog broja registara i tehnika za razmeštaj podataka u registre. Na početku poglavlja je prikazana tehnika za određivanje minimalnog broja registara, nakon čega sledi prikaz tehnike za razmeštaj podataka u registre. Na kraju poglavlja biće ilustrovana primena tehnika za minimizaciju broja registara u projektovanju savijenih arhitektura koje koriste najmanji mogući broj registara.

4.4.1. Analiza aktivnosti promenljivih

Analiza aktivnosti promenljivih je procedura koja se koristi za izračunavanje minimalnog broja potrebnih registara za implementaciju DSP algoritma. Promenljiva je aktivna (*live*) od trenutka kada joj je određena vrednost, pa do trenutka kada je ta vrednost upotrebljena. Nakon upotrebe promenljive, promenljiva postaje neaktivna (*dead*). Promenljiva zauzima jedan registar arhitekture u toku svake vremenske jedinice u kojoj je aktivna. Pri analizi aktivnosti izračunava se broj aktivnih promenljivih tokom svake vremenske jedinice, i određuje se maksimalan broj istovremeno aktivnih promenljivih. To je minimalan broj potrebnih registara za implementaciju DSP algoritma.

Na primer, neka DSP algoritam određuje tri promenljive *a*, *b* i *c*. Neka je promenljiva *a* aktivna u vremenskim intervalima 1, 2, 3 i 4, promenljiva *b* aktivna u vremenskim intervalima 2, 3, 4, 5, 6 i 7, i promenljiva *c* je aktivna u vremenskim intervalima 5, 6 i 7. Pretpostavljajući da se periodi aktivnosti promenljivih u prethodnoj i narednoj iteraciji ne preklapaju, broj aktivnih promenljivih za vreme intervala 1,2,3,4,5,6,7 je 1,2,2,2,2,2,2, respektivno. Minimalan broj registara potreban za implementaciju ovog DSP algoritma je maksimalan broj aktivnih promenljivih u istom trenutku, što je max[1,2,2,2,2,2,2]=2.

Za grafičko predstavljanje aktivnosti promenljivih koriste se linearani i kružni grafikoni aktivnosti. Linearni grafikon aktivnosti promenljive za prethodni primer je prikazan na sl. 4.7(a). Horizontalne linije predstavljaju cikluse grafa (vremenske jedinice), a vertikalne linije predstavljaju aktivnost promenljivih tokom vremena.



Slika 4.7. (a) Linearni grafikon aktivnosti. (b) Linearni grafikon aktivnosti u kome su prikazane tri iteracije uzimajući za period *N*=6. (c) Linearni grafikon aktivnosti koje uzima u obzir periodičnost algoritma, za period *N*=6.

Za prikazivanje aktivnosti na grafikonu koristi se konvencija da promenljiva nije aktivna u taktnom intervalu u kome dobija vrednost, a da je aktivna u ciklusu u kome se upotrebljava ta vrednost. Na primer promenljiva *a* dobija vrednost za vreme ciklusa 0, a upotrebljava se u ciklusu 4, pa je ova promenljiva aktivna u toku ciklusa 1,2,3,4. Na desnoj strani dijagrama aktivnosti je prikazan broj aktivnih promenljivih za vreme svakog ciklusa. Maksimalan broj aktivnih promenljivih u bilo kome koraku je 2, tako da je minimalan broj registara potrebnih za implementaciju DSP algoritama 2.

U opštem slučaju DSP algoritam je periodičan. Aktivnosti promenljivih *a*, *b* i *c* jedne iteracije DSP algoritma mogu se preklapati sa aktivnostima ovih istih promenljivih u drugim iteracijama. Na primer, neka je period iteracije N=6, i neka a_i, b_i i c_i predstavljaju vrednosti promenljivih *a*, *b* i *c* u iteraciji *i*. Dijagram aktivnosti za tri uzastopne iteracije ovog DSP algoritma je prikazan na sl. 4.7(b). Može se videti da se aktivnost vrednosti a_1 poklapa sa aktivnostima vrednosti b_0 i c_0, tako da DSP algoritam zahteva tri registra kada se uzme u obzir periodična priroda algoritma. Zbog toga je pri analizi uvek potrebno uzimati u obzir periodičnost. Na sreću, ova periodičnost može da se uračuna bez eksplicitnog crtanja aktivnosti za nekoliko iteracija. Ovo se postiže tako što se ucrtaju aktivnosti promenljivih u nultoj iteraciji, a za broj aktivnih promenljivih u vremenskom intervalu $n \geq N$ uzima se suma broja aktivnih promenljivih iz nulte iteracije u ciklusima $n-kN$, gde je k=0,1,2... Ovo je prikazano na sl. 4.7(c) gde je broj aktivnih promenljivih u ciklusu 7 jednak sumi broja aktivnih promenljivih nulte iteracije u ciklusima 7 i 1, što iznosi 2+1=3.

U opštem slučaju analiza aktivnosti promenljive počinje konstruisanjem tablice aktivnosti na način prikaazn u tab. 4.3. Ova tabela opisuje operaciju transponovanja matrice 3x3:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}.$$

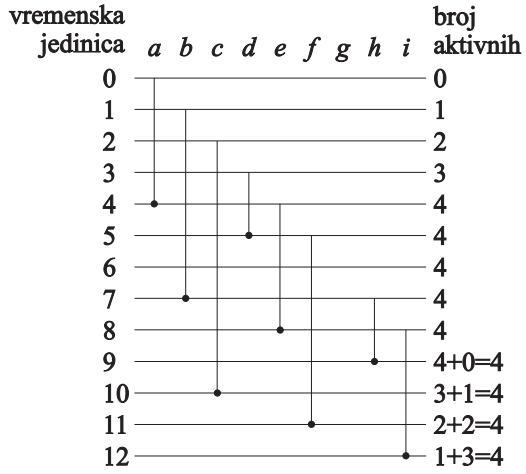
Svaka promenljiva u tabeli 3x3 ima svoje vreme unosa T_{input}, i vreme izlaza sa nultim kašnjenjem (*zero-latency*), T_{zlout}, koje predstavlja izlazno vreme promenljive, ignorišući kašnjenje sistema.

Tabela 4.3. Prikaz aktivnosti kod operacije transponovanja matrice 3x3

promenljiva	T_{input}	T_{zlout}	T_{diff}	T_{output}	period aktivnosti ($T_{input} \rightarrow T_{output}$)
a	0	0	0	4	0→4
b	1	3	2	7	1→7
c	2	6	4	10	2→10
d	3	1	-2	5	3→5
e	4	4	0	8	4→8
f	5	7	2	11	5→11
g	6	2	-4	6	6→6
h	7	5	-2	9	7→9
i	8	8	0	12	8→12

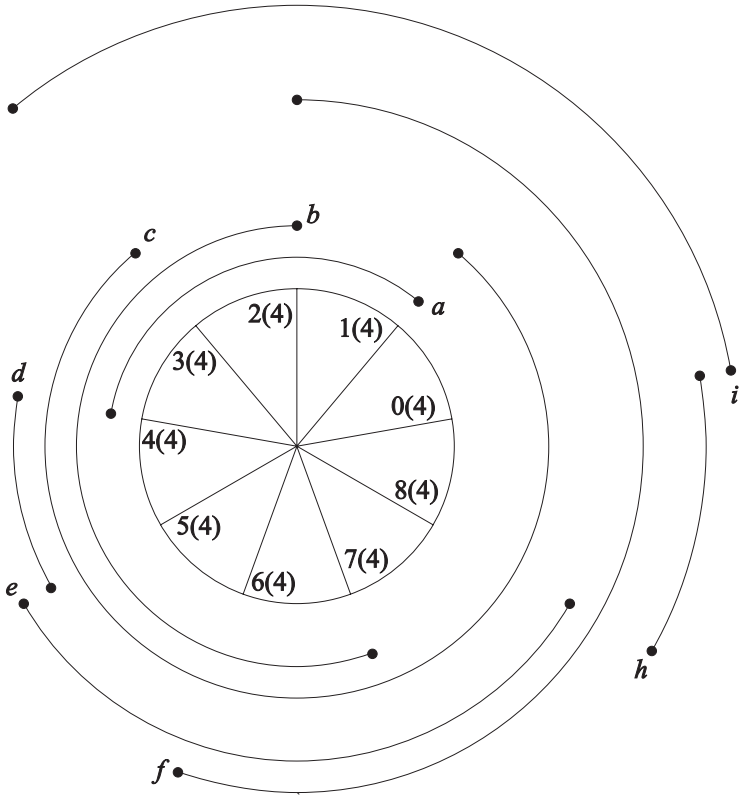
Svaka promenljiva u sistemu sa nultim kašnjenjem je aktivna $T_{diff}=T_{zlout}-T_{input}$ taktnih intervala. Može se desiti da je vreme tokom koga je promenljiva aktivna negativno, odnosno da se izlazna vrednost dobije pre dovođenja odgovarajuće vrednosti na ulazu. Na primer, promenljiva *d* ulazi u sistem u T_{input}=3, a izlazi u T_{zlout}=1, tako da je vreme aktivnosti 1-3=-2 vremenske jedinice, što fizički nema smisla. Da bi se ovo ispravilo sistemu se dodaje kašnjenje. Vreme kašnjenja, T_{lat}, je

jednako amplitudi najnegativnije vrednosti T_{diff}, pa je T_{lat}=-(-4)=4. Pravo izlazno vreme promenljive je $T_{output}=T_{zlout}+T_{lat}$, a vreme aktivnosti promenljivih je ($T_{input} \rightarrow T_{output}$). Pomoću ovih vrednosti može se nacrtati dijagram aktivnosti promenljive. Dijagram aktivnosti za tab. 4.3 je prikazan na sl. 4.8. Perioda ovog DSP algoritma je *N*=9.



Slika 4.8. Linearni grafikon aktivnosti za operaciju transponovanja matrice 3x3, sa periodom *N*=9. Vremena aktivnosti promenljivih su data u tabeli .

Drugi način analize aktivnosti promenljivih je analiza korišćenjem kružnih dijagrama aktivnosti promenljivih. Kružni dijagram aktivnosti koji odgovara linearnom dijagramu sa sl. 4.8 je prikazan na sl. 4.9.



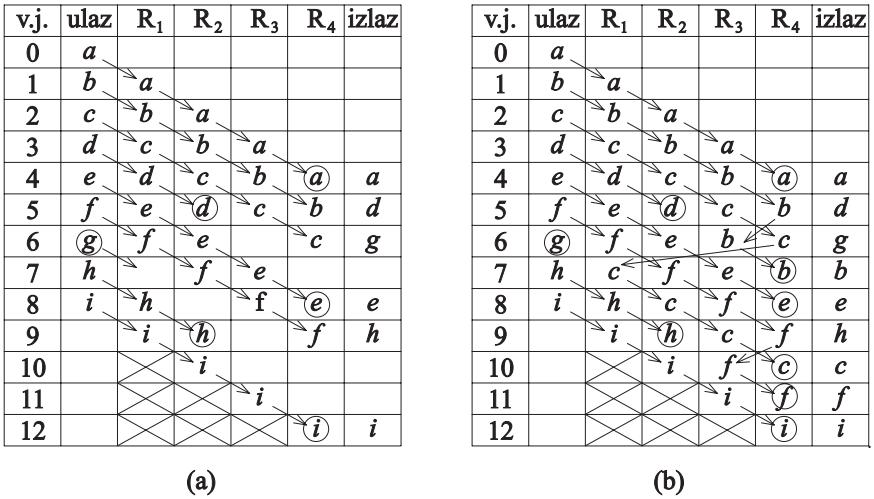
Slika 4.9. Kružni grafikon aktivnosti za transponovanje matrice 3x3.

Kružni dijagram aktivnosti uzima u obzir periodičnu prirodu DSP algoritma U kružnom dijagramu aktivnosti poluprečnik označen sa i (gde je i ceo broj koji zadovoljava uslov $0 \leq i \leq N-1$) predstavlja vremensku jedinicu i, kao i vremenske jedinice $[(N \cdot l+i)]$, gde je l bilo koji nenegativan ceo broj. Na primer, za N=8, vremenska jedinica i=3 ujedno predstavlja i vremenske jedinice i=11,19,27,..., itd. Brojevi u zagradama (sl. 4.8) predstavljaju broj aktivnih promenljivih u datoj vremenskoj jedinici. Na sl. 4.8 promenljiva, koja je dobila vrednost u vremenskoj jedinici j i upotrebljena tokom vremenske jedinice k, je označena kao aktivna od vremenske jedinice j+1 do vremenske jedinice k, jer prema konvenciji vrednost promenljive ne mora da se memoriše u vremenskoj jedinici u kojoj se izračunava.

4.4.2. Alokacija tipa "napred-nazad"

Nakon izračunavanja minimalnog broja registara potrebnih za implementaciju DSP algoritma, potrebno je razmestiti podatke u registre. Alokacija registara tipa "napred-nazad" (*forward-backward*) je šema za razmeštanje podataka u formirano registarsko polje savijene arhitekture.

Razmeštaj podataka u registre može se izvesti pomoću alokacione tabele. Alokaciona šema određuje kako se promenljive upisuju u registre u alokacionoj tabeli. Alokaciona tabela koja koristi "napred-nazad" šemu za razmeštanje podataka za transponovanje matrice 3x3 je prikazan na sl. 4.10(b). Na dalje su opisani koraci potrebni za izvršenje alokacije registara tipa "napred-nazad".



Slika 4.10. (a)Alokaciona tabela za transponovanje matrice 3x3 nakon izvođenja koraka 1 do 4 alokacione šeme. (b) Alokaciona tabela nakon završetka procedure.

Korak 1: Odrediti minimalan broj registara pomoću analize aktivnosti promenljivih.

Korak 2: Uvesti svaku promenljivu u ciklus u kome postaje aktivna. Ako se u jednom ciklusu uvodi više promenljivih, one se dodeljuju većem broju registara tako da se promenljiva koja je najduže aktivna smešta u prvi registar, a preostale promenljive u naredne registre, i to uređene po vremenu trajanja aktivnosti, u opadajućem redosledu.

Korak 3: Vrši se pomeranje unapred (*forward* alokacija) svake promenljive sve dok ne postane neaktivna ili dok ne dospe u poslednji registar. U alokaciji "napred", ako se promenljiva u

trenutnom taktnom intervalu nalazi u registru *i*, onda će se u narednom taktnom intervalu naći u registru *i*+1. Ako registar *i*+1 nije dostupan onda se promenljiva smešta u prvi sledeći slobodan registar.

Korak 4: Kako je zauzimanje lokacija periodično, zauzimanje pozicija u trenutnoj iteraciji se ponavlja i u narednim iteracijama. Zbog toga, ako se u R_j nalazi neka promenljiva u taktnom intervalu *l*, tada će se u R_j nalaziti odgovarajuća promenljiva i u taktnom intervalu *l*+*N*, gde je *N* perioda iteracije. Zbog toga je potrebno "zauzeti" pozicije za R_j u vremenskim jedinicama *l*+*N*, za svako *j* i *l*.

Korak 5: Za promenljive koje su dospele do poslednjeg registra i još uvek su aktivne izračunava se preostalo vreme aktivnosti, i sve promenljive se pomeraju unazad (*backward* alokacija) po FIFO principu (*First In – First Out*). Ako postoji više registara u koje promenljiva može da se vrati, prvo je potrebno potražiti registar ili registre za koji je alokacija "nazad" iz poslednjeg u te registre već bila izvedena. Ako ne postoje takvi registri u razmatranje je potrebno uzeti sve slobodne registre. Ako postoji više od jednog registra koji odgovaraju za vraćanje unazad, između registara koji imaju dovoljan broj sledbenika da se alokacija završi izabrati onaj sa najmanjim brojem sledbenika. Nakon što je promenljiva vraćena unazad, pomerati je unapred sve dok ne postane neaktivna, ili dok ne dospe do poslednjeg registra.

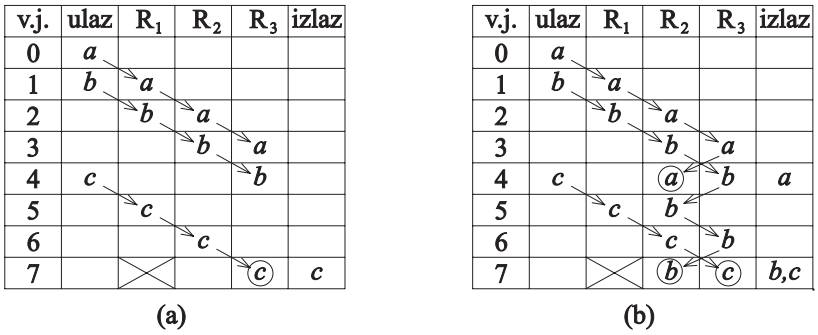
Korak 6: Ponavljati korake 4 i 5 sve dok se ne izvrši kompletna dodela.

Na sl. 4.10 je prikazana "napred-nazad" alokaciona tabela za transponovanje matrice 3x3 čija je tabela aktivnosti data u tab. 4.3. Korak 1 je obavljen na način opisan u prethodnom odeljku, gde je određeno da su za transponovanje matrice 3x3 potrebna 4 registra (sl. 4.8 i sl. 4.9). Koraci 2, 3 i 4 su prikazani na sl. 4.10(a). U koraku 2 svaka promenljiva se unosi u taktnom intervalu u kome počinje njena aktivnost. U ovom primeru, ni u jednom taktnom intervalu se ne unosi više od jedne promenljive. U koraku 3 sve promenljive su pomerene unapred. U koraku 4 su zauzete odgovarajuće pozicije kako ne bi došlo do konflikta pri vraćanju unazad. Na primer, registar R2 je zauzet u taktnom intervalu 11 jer ga koristi promenljiva *a* u taktnom intervalu 11-9=2. Korak 5 je prikazan na sl. 4.10(b). U ovom koraku promenljive su vraćene unazad. Na primer, promenljiva b je vraćena u registar R_3 jer je ovo jedini slobodan registar za alokaciju "nazad". Nakon što je svaka od promenljivih *b*, *c* i *f* vraćena unazad, a potom pomerena unapred dok nije postala neaktivna, alokacija je kompletirana.

Drugi primer alokacije registara tipa "napred-nazad", alokaciona tabela za podatke sa sl. 4.7(c) je data na sl. 4.11. Na sl. 4.11(a) je prikazana alokaciona tabela nakon koraka 1 do 4, a na sl. 4.11 (b) je prikazana alokaciona tabela nakon završetka alokacije.

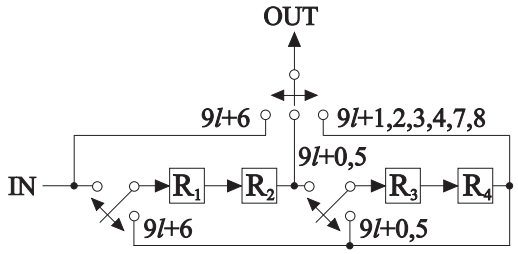
Na osnovu alokacione tabele može se projektovati registarsko polje arhitekture. Arhitektura za transponovanje matrice 3x3 je prikazana na sl. 4.12. Tok podataka kroz arhitekturu je potpuno opisan pomoću alokacione tabele na sl. 4.10(b). Na primer, vraćanje promenljive *b* unazad u taktnom intervalu 5 sa R_4 na R_3 na sl. 4.10(b) je implementirano pomoću povratne veze na sl. 4.12, sa izlaza R_4 na ulaz R_3, gde se propušta u vremenskim jedinicama 9*l*+5. Slično je implementirano vraćanje promenljive *c*. Vraćanje promenljive *c* je implementirano povratnom vezom od izlaza iz R_4 do ulaza u R_3, koja je uspostavljena tokom taktnih intervala 9*l*+6. Promenljiva *f* se vraća nazad u

vremenskim jedinicama $9l+0$ jer se na sl. 4.10(b) vraćanje javlja u taktnom intervalu 9, a sva vremena vraćanja moraju biti oblika $9l+m$, gde je $0 \leq m \leq 8$.

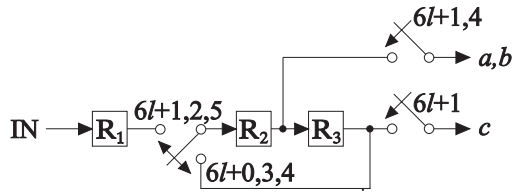


Slika 4.11. (a) Alokaciona tabela za podatke a sl. 4.7(c) nakon što su izvedeni koraci 1 do 4 alokacione procedure. (b) Alokaciona tabela nakon što je alokacija kompletirana.

Arhitektura koja odgovara alokacionoj tabeli sa sl. 4.11(b) je prikazana na sl. 4.13.



Slika 4.12. Arhitektura koja odgovara alokacionoj tabeli sa sl. 4.10(b), za transponovanje matrice 3x3.



Slika 4.13. Arhitektura koja odgovara alokacionoj tabeli sa sl. 4.11(b).

4.4.3. Primeri minimizacije registara u savijenim arhitekturama

Opisane tehnike za minimizaciju registara se mogu koristiti za sintezu upravljačke logike u savijenim arhitekturama koje koriste najmanji mogući broj registara. Osnovni koraci su:

1. Izvršiti vremensko usklađivanje za savijanje
2. Napisati jednačine savijanja
3. Pomoću jednačina savijanja sastaviti tablicu aktivnosti
4. Nacrtati dijagram aktivnosti i odrediti potreban broj registara
5. Izvršiti alokaciju registara tipa "napred-nazad"
6. Nacrtati savijenu arhitekturu koja koristi minimalan broj registara

Postupak minimizacije broja registara je ilustrovan na primeru bikvadratnog filtra i IIR filtra.

4.4.3.1. Primer 1 - Bikvadratni filter

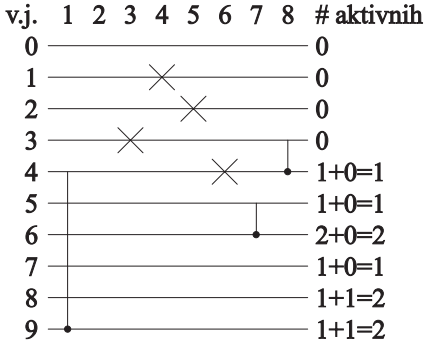
U ovom primeru je projektovana arhitektura sa minimalnim brojem registara za DFG graf na sl. 4.5. DFG nakon vremenskog usklađivanja za savijanje je prikazan na sl. 4.3, a jednačine savijanja su date sa (4.2). Savijena arhitektura bez minimizacije broja registra je prikazana na sl. 4.4. Ova arhitektura koristi 6 registra (3 registra za postizanje protočnosti sabirača i množača nisu uračunata).

Kako je vremensko usklađivanje za savijanje već obavljeno, i kako su jednačine savijanja date sa (4.2), naredni korak je formiranje tabele aktivnosti. Aktivnost promenljivih je prikazana u tab. 4.4. U tabeli aktivnosti je dato vreme aktivnosti ($T_{input} \rightarrow T_{output}$) za svaki čvor DF grafa. Vreme T_{input} za čvor U je $u+P_U$, gde je u red savijanja čvora U, a P_U broj stepena protočnosti funkcionalne jedinice koja izvršava operaciju U. Ova vrednost za T_{input} predstavlja vremensku jedinicu u kojoj hardverska realizacija datog čvora izračunava vrednost za nultu iteraciju DSP algoritma. Na primer, T_{input} za čvor 1 je 3+1=4. Vreme T_{output} za čvor U je $u+P_U+\max[D_F(U \rightarrow V)]$, gde $\max[D_F(U \rightarrow V)]$ predstavlja najveće kašnjenje između kašnjenja svih potega koji polaze iz čvora U. Vrednost T_{output} predstavlja poslednju vremensku jedinicu u kojoj se koristi rezultat operacije U. Na primer, T_{output} za čvor 1 je 3+1+max[1,0,2,3,5]=3+1+5=9. U ovom primeru nije potrebno dodavati vreme kašnjenja izlaznim vrednostima, jer korak u kome se vrši vremensko usklađivanje garantuje da DSP algoritam nema grana sa negativnim kašnjenjima.

Tabela 4.4. Vremena aktivnosti za bikvadratni filter sa sl. 4.3.

operacija	$T_{input} \rightarrow T_{output}$
1	4→9
2	–
3	3→3
4	1→1
5	2→2
6	4→4
7	5→6
8	3→4

Linearni grafikon aktivnosti za tabelu aktivnosti tab. 4.4 je dat na sl. 4.14. Period ponavljanja DSP algoritma je N=4. Na osnovu grafikona aktivnosti sa sl. 4.14 sledi da je minimalan broj registara, potrabnih za implementaciju savijene arhitekture, jednak dva.



Slika 4.14. Dijagram aktivnosti za bikvadratni filter

Naredni korak je alokacija podataka u registare savijene arhitekture alokacijom tipa "napred-nazad". Alokaciona tabela je prikazana na sl. 4.15. Promenljiva n_i u ovoj tabeli predstavlja izlaz iz

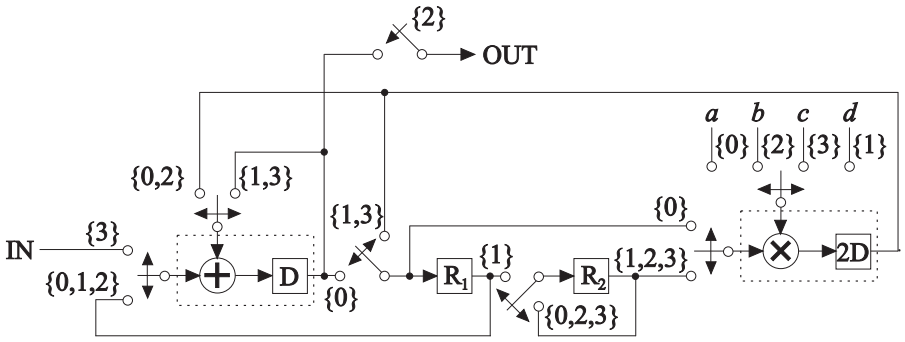
čvora *i*. U ovoj tabeli su prikazane samo tri promenljive, n_1, n_7 i n_8, jer samo ove promenljive imaju vreme aktivnosti koje nije nula. Za vreme izlaza promenljive n_1 uzet je taktni interval 9. Ova promenljiva ima izlaze i u taktnim intervalima 4, 5, 6 i 7. Zbog jasnijeg predstavljanja, u tabeli je prikazano samo poslednje izlazno vreme za svaku promenljivu.

v.j.	ulaz	R_1	R_2	izlaz
0				
1				
2				
3	n_8			
4	n_1	n_8		n_8
5	n_7	n_1		
6		n_7	n_1	n_7
7			n_1	
8			n_1	
9			n_1	n_1

Slika 4.15. Alokaciona tabela za savijeni bikvadratni filter

Na osnovu alokacione tabele sa sl. 4.15 i jednačina savijanja (4.2) dolazi se do arhitekture prikazane na sl. 4.16. Npr, poteg (1→2) u savijenoj arhitekturi ima kašnjenje $D_F(1\rightarrow2)=1$. Ovaj poteg polazi iz čvora 1 pa je za njega vezana promenljiva n_1, a posle jedne vremenske jedinice promenljiva n_1 se nalazi u registru R_1 (sl. 4.15), tako da postoji poteg od R_1 do sabirača, u savijenoj arhitekturi, i ova promenljiva se propušta u sabirač u vremenskim jedinicama $4l+1$, jer čvor 2 ima red savijanja 1. Poteg (1→7) ima kašnjenje $D_F(1\rightarrow7)=3$. Promenljiva n_1 se počev od drugog taktnog intervala nalazi u registru R_2, tako da postoji poteg od R_2 do množača, u koji se propušta ova vrednost u vremenskim jedinicama $4l+3$ jer čvor 7 ima red savijanja 3.

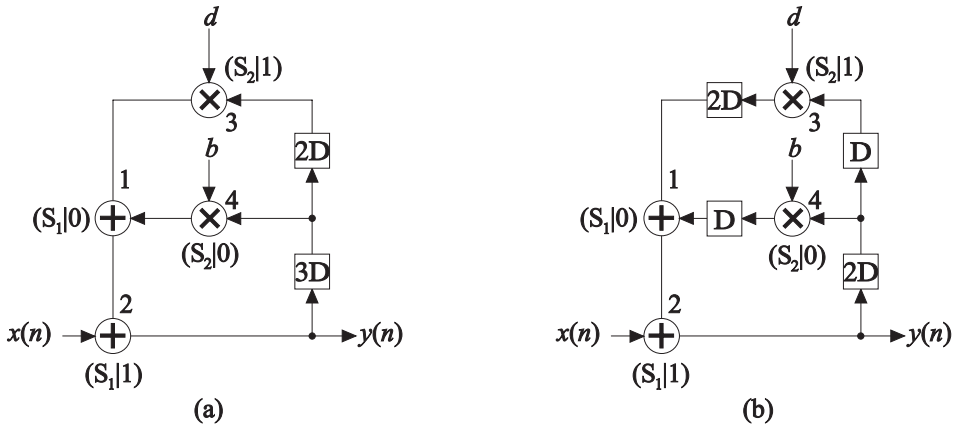
U ovoj arhitekturi je broj registara smanjen sa 6 na 2.



Slika 4.16. Arhitektura savijenog bikvadratnog filtra sa minimalnim brojem registara

4.4.3.2. Primer 2 - IIR filter

U ovom odeljku je prikazan postupak savijanja IIR filtra sa sl. 4.17(a), čija je prenosna funkcija $y(n)=ay(n-3)+by(n-5)+x(n)$. U ovom slučaju sprovedeni su koraci od 1 do 6, navedeni na početku ovog poglavlja, kako bi se dobila savijena arhitektura sa minimalnim brojem registara. U prvom koraku se izvodi vremensko usklađivanje za savijanje. Jednačine savijanja i nejednačine za vremensko usklađivanje su date u tab. 4.5. Pomoću prethodno opisane tehnike za rešavanje sistema nejednačina za vremensko usklađivanje, dobijaju se sledeće vrednosti r(1)=0, r(2)=0, r(3)=-2 i r(4)=-1. DFG na koji je prmenjeno vremensko usklađivanje je prikazan na sl. 4.17(b).



Slika 4.17. IIR filter koji izračunava $y(n)=ay(n-3)+by(n-5)+x(n)$. (b) Vremenski usklađen DFG IIR filtra u kome nema negativnog kašnjenja u potezima.

Jednačine savijanja za DFG graf sa sl. 4.17(b) su:

$$D_F(1\rightarrow 2)=2(0)-1+1-0=0$$
$$D_F(2\rightarrow 3)=2(3)-1+1-1=5$$
$$D_F(2\rightarrow 4)=2(2)-1+0-1=2$$
$$D_F(3\rightarrow 1)=2(2)-2+0-1=1$$
$$D_F(4\rightarrow 1)=2(1)-2+0-0=0 \qquad (4.6)$$

Tabela 4.5. Jednačine savijanja i vremenskog usklađivanja, za DFG sa sl. 4.17(a).

poteg	jednačine savijanja	vremensko usklađivanje za jednačinu savijanja
$1 \rightarrow 2$	$D_F(1\rightarrow 2) = 0$	$r(1)-r(2)\leq 0$
$2 \rightarrow 3$	$D_F(2\rightarrow 3) = 9$	$r(2)-r(3)\leq 4$
$2 \rightarrow 4$	$D_F(2\rightarrow 4) = 4$	$r(2)-r(4)\leq 2$
$3 \rightarrow 1$	$D_F(3\rightarrow 1) = -3$	$r(3)-r(1)\leq -2$
$4 \rightarrow 1$	$D_F(4\rightarrow 1) = -2$	$r(4)-r(1)\leq -1$

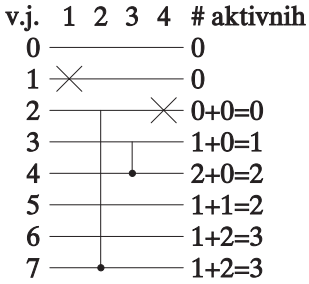
Tabela aktivnosti je data u tab. 4.6, grafikon aktivnosti je dat na sl. 4.18, a alokaciona tabela je data na sl. 4.19(a). Iz alokacione tabele i jednačna savijanja dobija se arhitektura prikazana na sl. 4.19(b). Ova arhitektura koristi 3 registra za čuvanje podataka, ne računajući registre potrebne za postizanje protočnosti, nasuprot 6 registara potrebnih pri direktnoj implementaciji, bez minimizacije.

Tabela 4.6. Tabela aktivnosti za jednačine (4.6).

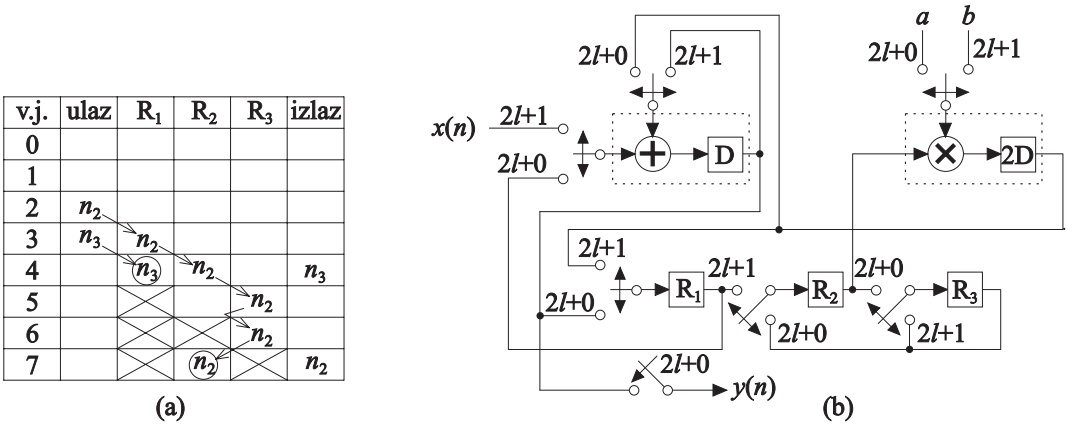
čvor	$T_{input}\rightarrow T_{output}$
1	$1\rightarrow 1$
2	$2\rightarrow 7$
3	$3\rightarrow 4$
4	$4\rightarrow 2$

Korišćenjm transformacije savijanja, na sistematski način se smanjuje se broj potrebnih funkcionalnih jedinica za implementaciju DSP algoritma, dok se korišćenjem tehnike za minimizaciju broja registara smanjuje broj potrebnih registara za pamćenje podataka, odnosno međurezultata, u savijenoj arhitekturi. Primenom ovih tehnika smanjuje se potrebna površina silicijuma na čipu koja je potrebna za implementaciju DSP algoritma.

Naredno poglavlje je posvećeno sintezi savijenih semi-sistoličkih polja za FIR filtriranje zasnovanih na BP semi-sistoličkom FIR filtru.



Slika 4.18. Grafikon aktivnosti za vremena u tablici 3.6.



Slika 4.19. (a) Alokaciona tabela IIR filtra (b) Arhitektura IIR filtra sa minimalnim brojem registara

5. SINTEZA SAVIJENIH SEMI-SISTOLIČKIH POLJA ZA FIR FILTRIRANJE

Poznato je da performanse i cena digitalnih sistema zavise od njegovog dizajna i načina projektovanja. Prilikom sinteze DSP arhitektura neophodan je pažljiv izbor odnosa površina-propusnost. Kod projektovanja DSP arhitektura važnu ulogu igra minimizacija zauzetog prostora na čipu, bilo da se radi o namenskom VLSI dizajnu ili o implementaciji na FPGA platformama. Minimizacija se postiže smanjenjem broja funkcionalnih jedinica i veza između tih funkcionalnih jedinica. Transformacija savijanja se koristi u cilju sistematskog projektovanja upravljačkih kola i upravljačkih signala kod DSP arhitektura kod kojih se više operacija izvršava na jednoj funkcionalnoj jedinici. Broj funkcionalnih jedinica se smanjuje što ima za rezultat smanjenje zauzetosti resursa na čipu.

U ovom poglavlju je prikazana sinteza savijenih semi-sistoličkih polja za FIR filtriranje. Kao polazna arhitektura za sintezu izabrana je arhitektura BP FIR filtra, opisana u poglavlju 3. BP arhitektura je regularna, karakterišu je sitnozrnasta (*fine grain*) protočnost, regularne veze između funkcionalnih jedinica, visoka propusnost, mogućnost odbacivanja bitova najmanje težine bez gubitka pouzdanosti, kao i mogućnost programiranja koeficijenata [12],[13].

Tehnikom savijanja, opisanom u poglavlju 4, redukuje se broj funkcionalnih jedinica na račun vremena. Međutim, savijeni sistem može da se projektuje tako da je na implementiranim funkcionalnim jedinicama savijenog sistema moguće dinamički menjati broj operacija DSP algoritma, imajući u vidu da je za obimniji algoritam potrebno duže vreme izvršenja. Ovim je moguće značajno proširiti oblast primene specijalizovanih DSP sistema. Oblast primene FIR filtra je, takođe, moguće proširiti uvođenjem mogućnosti programiranja broja i dužine koeficijenata. Korišćenjem filtra sa promenljivim brojem i dužinom koeficijenata u sklopu kompleksnog sistema dobija se sistem koji je moguće dinamički reprogramirati kako bi se jednim čipom podržalo više različitih varijanti nekog standarda (npr. varijante standarda bežičnih mreža IEEE 802.11).

Poglavlje počinje pripremom BP polja za primenu tehnike savijanja. BP polje će biti transformisano u cilju uspešne primene tehnike savijanja. Dokaz o očuvanju ispravnosti rezultata transformisanog BP polja će biti dat. Postupak dodele skupova savijanja transformisanom BP polju, kao i sam postupak savajanja biće predstavljen. Proces filtriranja na savijenom polju biće detaljno objašnjen. Arhitektura će biti modifikovana, kako bi se omogućila promena faktora savijanja na polju konstantne veličine. Arhitektura savijenog BP FIR filtra sa promenljivim faktorom savijanja će biti prikazana.

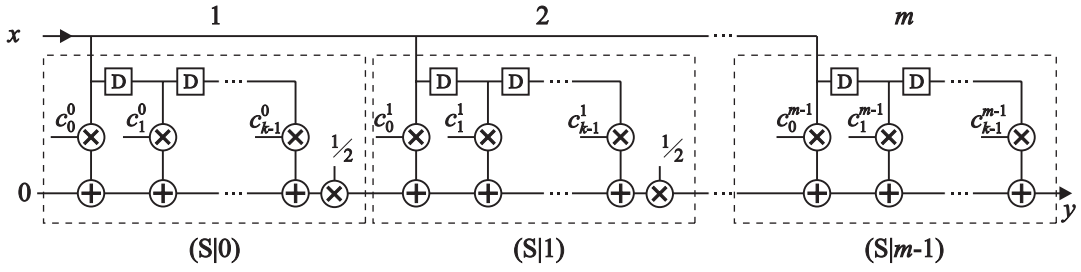
Zavisnosti između načina dodele skupova savijanja i dobijene savijene arhitekture biće uočene. Novi način dodele skupova savijanja koji omogućava sintezu FIR filtra sa promenljivim brojem i dužinom koeficijenata će biti uveden. Postupak sinteze arhitekture sa mogućnošću promene broja i dužine koeficijenata i sinteze arhitekture sa dodatnom mogućnošću promene faktora savijanja će biti prikazan. Algoritam preuređenja bitova koeficijenata u zavisnosti od broja koeficijenata, dužine koeficijenata i faktora savijanja će biti predstavljen. Princip filtriranja na savijenom BP polju sa promenljivim brojem koeficijenata, dužinom koeficijenata i faktorom savijanja biće objašnjen.

5.1. TRANSFORMACIJA "BIT-PLANE" ARHITEKTURE

BP arhitektura se ne može saviti direktnom primenom tehnike savijanja. BP algoritam je baziran na preuređenju parcijalnih proizvoda (sl. 3.2 i sl. 3.3), tako da se množenja koeficijenata i ulaznih reči ne mogu prepoznati kao operacije. Odatle proizilazi neophodnost primene tehnike savijanja na nivou operacija nad bitovima [18].

Svaki množač DFG-a sa sl. 3.3 predstavlja jednu vrstu osnovnih ćelija (potpuni sabirač i I-kolo) funkcionalnog blok dijagrama sa sl. 3.2. Drugim rečima, jedno množenje je distribuirano kroz ceo niz osnovnih ćelija. Čak i ako se označi formiranje svih parcijalnih proizvoda u jednoj vrsti kao "vrsta-operacija", ne može se uspešno primeniti tehnika savijanja jer postoje kašnjenja na putu ulaznih podataka i suma, koje onemogućavaju da se zadovolji uslov savijanja $D_F(U \rightarrow V) \geq 0$ za sve potege u DFG-u.

Rešenje je predloženo u [18]. Ovo rešenje je takvo da se sva izračunavanja u okviru jednog BP-a označe kao "BP operacija" i vremenski multipleksiraju na jednu funkcionalnu jedinicu. Transformisani DFG (TDFG) [18] je prikazan na sl. 5.1. Kašnjenja između BP operacija i kašnjenja na putu sabiranja parcijalnih proizvoda su uklonjena.



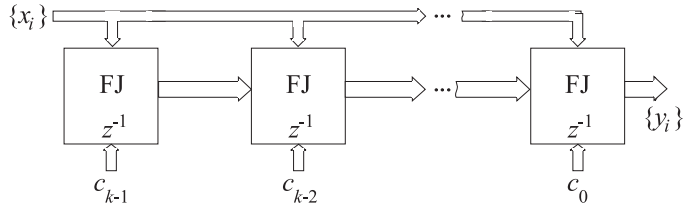
Slika 5.1. Transformisani DFG-TDFG koji sadrži jedan skup savijanja S sa *m* "vrsta-operacija"

TDFG vodi savijenoj BP arhitekturi koja je pogodna za filtriranje sa relativno malim brojem koeficijenata. Predložena transformacija (sl. 5.1) odstranjuje kašnjenja u putevima podataka suma i prenosa. Na ovaj način je uklonjena i protočnost koja je postojala u okviru jednog BP elementa. Dužina kritičnog puta u okviru BP elementa u ovom slučaju direktno zavisi od dužine tog elementa tj. broja koeficijenata. Ovo je ujedno i motivacija za nalaženje drugog rešenja koje će se koristiti u postupku sinteze protočne, savijene BP arhitekture.

5.1.1. Transformacija DFG-a "bit-plane" arhitekture

Drugo rešenje se sastoji u pronalaženju novog skupa transformacija polazne arhitekture koje će omogućiti uspešnu primenu tehnike savijanja. Uspešna primena podrazumeva da se veličina hardvera smanji približno N puta na račun vremena, gde je N faktor savijanja, a da dobijena arhitektura zadrži sve bitne karakteristike polazne arhitekture.

Osobina polja koja može da omogući uspešnu primenu tehnike savijanja je grupisanje bitova koeficijenata na niz sukcesivnih funkcionalnih jedinica, kao što je prikazano na sl. 5.2 [19].

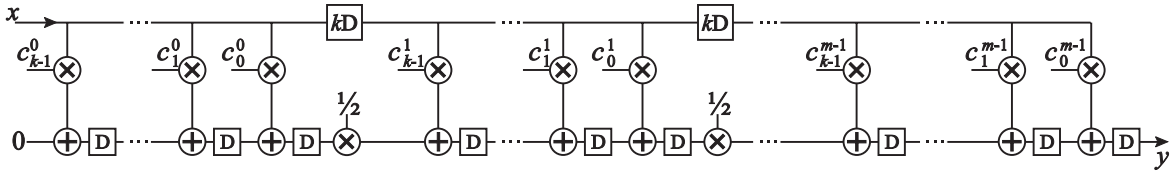


Slika 5.2. Arhitektura transformisanog "*bit-plane*" FIR filtra

Svaka FJ arhitekture prikazane na sl. 5.2 množi ulaznu reč jednim koeficijentom. Ukoliko se na FJ sa sl. 5.2 primeni tehnika savijanja, promena dužine koeficijenata utiče jedino na vreme potrebno za izračunavanje parcijalnih proizvoda u okviru FJ, tj. promena faktora savijanja savijene arhitekture svodi se na promenu vremena potrebnog za izračunavanje parcijalnog proizvoda $c_i x_j$. Grupisanje bitova jednog koeficijenata na jednu FJ, pored uspešne primene tehnike savijanja, rezultovaće relativno jednostavnom upravljačkom logikom za promenu faktora savijanja na polju fiksne veličine.

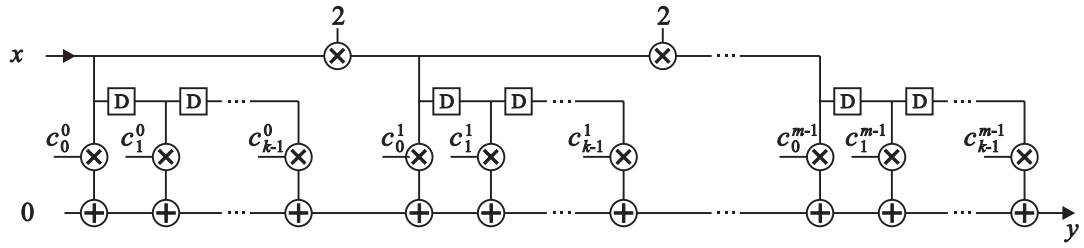
Transformacija BP arhitekture će biti prezentovana na dva načina. Prvo, transformacija će biti predstavljena kao procedura transformacije DFG-a, a nakon toga će transformacija DFG-a biti dokazana transformacijom prenosne funkcije [21].

Korak_1 Početni DFG BP arhitekture za proceduru transformacije je prikazan na sl. 5.3 u opštem obliku, sa k koeficijenata dužine m. Uklanjanjem kašnjenja između BP elemenata, kao i kašnjenja u granama DFG-a za sabiranje parcijalnih proizvoda, dobija se TDFG sa sl. 5.1, koji ima kašnjenja unutar BP elemenata. TDFG omogućava postojanje samo jednog skupa savijanja S=[0,1,…,m-1] sa m BP operacija. Međutim, ovo vodi rešenju publikovanom u [18], koje ima za nedostatak, kako je ranije rečeno, zavisnost kritičnog puta od dužine koeficijenata. Uzmimo TDFG sa sl. 5.1 kao polaznu arhitekturu za korak 2.



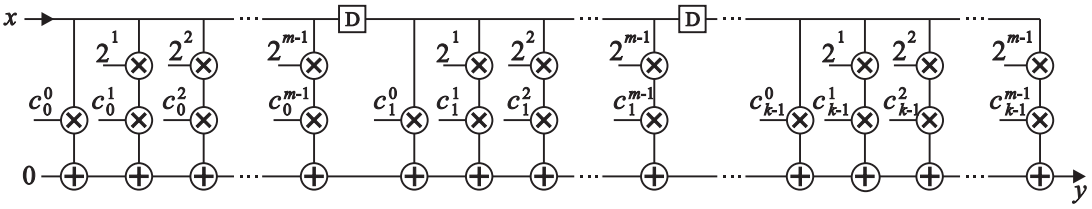
Slika 5.3. DFG BP FIR filtra sa k koeficijenata dužine m bitova

Korak_2 Množenja faktorom ½, koja se nalaze u okviru grana DFG-a za sumiranje parcijalnih proizvoda se uklanjaju iz DFG-a, a na račun toga se dodaju množenja faktorom 2 grani DFG-a za uvođenje ulaznih reči u polje. Ova transformacija je prikazana na sl. 5.4.



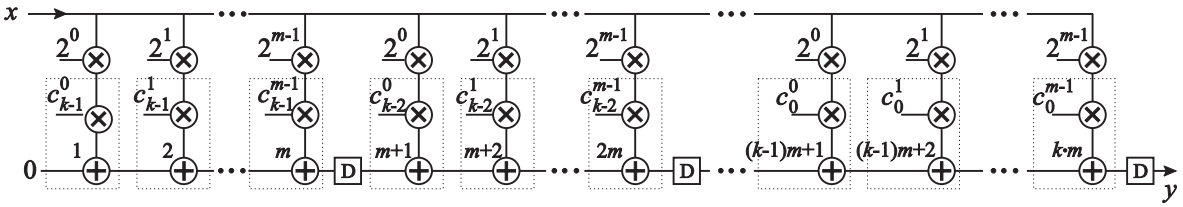
Slika 5.4. Postavljanje množača na putu ulaznih podataka

Korak_3 Preuređenje parcijalnih proizvoda grupisanjem bitova koeficijenta ilustrovano je na sl. 5.5. Preuređenjem parcijalnih proizvoda je promenjena struktura BP elementa u odnosu na strukturu BP elementa osnovne arhitekture. Ova transformacija uvodi kašnjenja u grani DFG-a za uvođenje ulaznih reči u polje. DFG sa sl. 5.5 nije spreman za primenu tehnike savijanja. Razlog je postojanje kašnjenja u okviru grana DFG-a za ulazne podatke.



Slika 5.5. Preuređenje parcijalnih proizvoda grupisanjem bitova koeficijenata

Korak_4 Poslednji korak transformacije je prikazan na sl. 5.6. Kašnjenja koja su bila prisutna u grani za uvođenje ulaznih reči u polje su pomerena tako da sada pripadaju granama DFG-a za sabiranje parcijalnih proizvoda. Posledica ove transformacije je preuređenje koeficijenata u obrnuti redosled. Filtri kod kojih izračunavanje rezultata počinje koeficijentom sa najvećim indeksom se u literaturi nazivaju filtri sa *transponovanovanom* strukturom.



Slika 5.6. DFG transponovanog BP FIR filtra

Transponovani BP (TrBP) FIR filter sa sl. 5.6 je spreman za primenu tehnike savijanja.

U narednom odeljku je dat matematički dokaz o očuvanju ispravnosti rezultata TrBP FIR filtra nakon transformacije.

5.1.2. Dokaz o očuvanju ispravnosti rezultata

Dokaz o očuvanju ispravnosti rezultata nakon transformacije DFG-a BP FIR filtra moguće je izvesti transformacijom prenosne funkcije.

Polazna arhitektura za Korak_1 je prikazana DFG-om sa sl. 5.3 i odgovarajuća prenosna funkcija je data sa (3.2). Prenosna funkcija (3.2), napisana bez zagrada, je:

$$G(z) = z^{-1}c_0^{\,m-1}2^{m-1}z^{-(m-1)k} + z^{-2}c_1^{\,m-1}2^{m-1}z^{-(m-1)k} + \ldots + z^{-k}c_{k-1}^{\,m-1}2^{m-1}z^{-(m-1)k} +$$

$$+ z^{-(k+1)}c_0^{\,m-2}2^{m-2}z^{-(m-2)k} + z^{-(k+2)}c_1^{\,m-2}2^{m-2}z^{-(m-2)k} + \ldots + z^{-2k}c_{k-1}^{\,m-2}2^{m-2}z^{-(m-2)k} +$$

$$\ldots$$

$$+ z^{-[(m-1)k+1]}c_0^{\,0}2^0z^0 + z^{-[(m-1)k+2]}c_1^{\,0}2^0z^0 + \ldots + z^{-mk}c_{k-1}^{\,0}2^0z^0 =$$

$$= \sum_{i=0}^{m-1}\sum_{j=0}^{k-1} z^{-[(m-1-i)k+j+1]}c_j^i 2^i z^{-ik} = \sum_{i=0}^{m-1}\sum_{j=0}^{k-1} c_j^i 2^i z^{-[(m-1)k+j+1]} =$$

$$z^{-[(m-1)k+1]}\sum_{i=0}^{m-1}\sum_{j=0}^{k-1} c_j^i 2^i z^{-j} \qquad (5.1)$$

.

Ukoliko se parcijalni proizvodi preurede tako da budu grupisani prema z^{-j}, prenosna funkcija G(z) ima sledeći oblik:

$$G(z) = z^{-[(m-1)k+1]}\sum_{j=0}^{k-1}\sum_{i=0}^{m-1} c_j^i 2^i z^{-j} = z^{-[(m-1)k+1]}\sum_{j=0}^{k-1} z^{-j}\sum_{i=0}^{m-1} c_j^i 2^i, \qquad (5.2)$$

tj.

$$G(z) = z^0(c_0^{\,m-1}2^{m-1} + c_0^{\,m-2}2^{m-2} + \ldots + c_0^{\,0}2^0 + z^{-1}(c_1^{\,m-1}2^{m-1} + c_1^{\,m-2}2^{m-2} + \ldots + c_1^{\,0}2^0 +$$

$$\ldots$$

$$+ z^{-1}(c_{k-1}^{\,m-1}2^{m-1} + c_{k-1}^{\,m-2}2^{m-2} + \ldots + c_{k-1}^{\,0}2^0)\ldots). \qquad (5.3)$$

DFG koji odgovara prenosnoj funkciji (5.3) je transponovani DFG (TrDFG), prikazan na sl. 5.6. Drugim rečima, transformacije prenosne funkcije od (5.1) do (5.3) prate prethodno opisani scenario transformacije DFG-a BP FIR filtra.

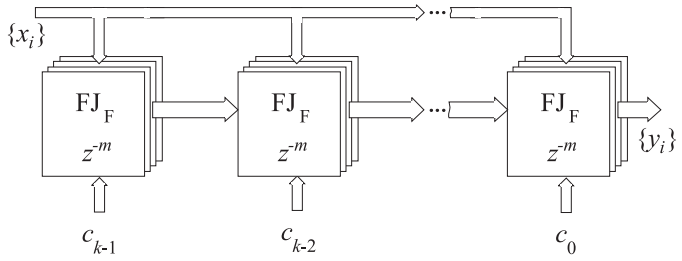
Arhitektura TrBP FIR filtra, predstavljena DFG-om na sl. 5.6, je nepraktična za implementaciju zbog *broadcast* linije za uvođenje ulaznih podataka, ali je dobro pripremljena za primenu tehnike savijanja.

5.2. SAVIJENO SEMI-SISTOLIČKO POLJE ZA FIR FILTRIRANJE SA PROMENLJIVIM FAKTOROM SAVIJANJA

U ovom odeljku je dat postupak sinteze savijenog BP semi-sistoličkog FIR filtra sa promenljivim faktorom savijanja [21]-[25]. Skupovi savijanja biće opisani matematički. Tehnika savijanja će biti primenjena na TrBP arhitekturu u opštem obliku, što će omogućiti analizu savijene arhitekture. Cilj analize je dobijanje zavisnosti promene arhitekture u funkciji parametara savijenog filtra. Na osnovu ovih zavisnosti biće generisan funkcionalni blok dijagram savijene arhitekture, koji ima mogućnost promene faktora savijanja. Savijeno polje biće prikazano grafom toka podataka i funkcionalnim blok dijagramom. Postupak sinteze biće detaljno objašnjen.

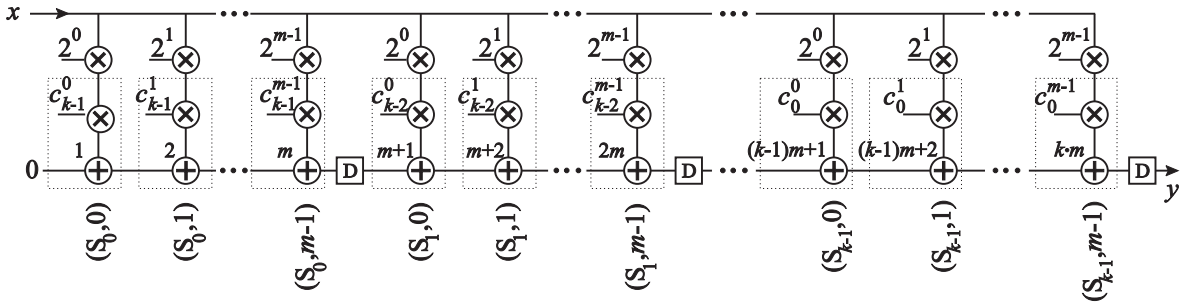
5.2.1. Dodela skupova savijanja

Nakon transformacija kojim je BP FIR filter pripremljen za primenu tehnike savijanja, potrebno je funkcionalnim jedinicama dodeliti skupove savijanja. Skupovi savijanja su dodeljeni tako da operacije nad bitovima istog koeficijenta pripadaju istom skupu savijanja. Ovakav pristup, na osnovu blok dijagrama sa sl. 5.2, rezultuje savijenom arhitekturom koja je prikazana na sl. 5.7. FJ savijene arhitekture sa sl. 5.7 (FJ_F) izvršavaju operacije nad bitovima jednog koeficijenta, što, kao što je ranije rečeno, pored uspešne primene tehnike savijanja, rezultuje i relativno jednostavnom upravljačkom logikom za promenu faktora savijanja na polju fiksne veličine.



Slika 5.7. Blok dijagram arhitekture savijenog BP FIR filtra

Način dodele skupova savijanja je prikazan na sl. 5.8.



Slika 5.8. TrDFG sa pridruženim skupovima savijanja – *k* skupova savijanja gde svaki skup sadrži po *m* operacija

Operacije Tr BP FIR filtra su na DFG-u označene isprekidanim linijama (sl. 5.8). Pod "operacijom" se podrazumeva formiranje parcijalnog proizvoda i akumulacija, odnosno funkcionalnost jedne vrste osnovnih ćelija (poglavlje 3, sl. 3.2). Dodeljeno je k skupova savijanja, $S_0,S_1,S_2,...,S_{k-1}$, gde je k broj koeficijenata. Svaki skup savijanja sadrži m operacija.

Svakoj operaciji TrDFG-a sa sl. 5.8 su pridruženi redni brojevi od 1 do $k \cdot m$. U skladu sa notacijom datom u poglavlju 4, skupovi savijanja su:

$$S_0=[1,2,3,\ldots,m]$$
$$S_1=[m+1,m+2,m+3,\ldots,2m]$$
$$S_2=[2m+1,2m+2,2m+3,\ldots,3m]$$
$$\ldots$$
$$S_i=[i\cdot m+1,i\cdot m+2,\ i\cdot m+3,\ldots,\ (i+1)\cdot m]$$
$$\ldots$$
$$S_{k-1}=[(k-1)\cdot m+1,\ (k-1)\cdot m+2,\ (k-1)\cdot m+3,\ldots,\ k\cdot m]. \qquad (5.4)$$

Operacije algoritma sa sl. 5.8 su označene rednim brojevima od 1 do $k \cdot m$. Ukoliko operacije označimo sa p ($0 \le p \le k \cdot m$), a skupove savijanja sa (S_S | r), gde je r red savijanja operacije u okviru skupa savijanja S_S, tada važi [22]:

$$s=\lfloor (p-1)/m \rfloor$$
$$r=p \bmod m. \qquad (5.5)$$

Faktor savijanja, N, za ovako dodeljene skupove savijanja, je jednak dužini koeficijenata m ($N=m$). Drugim rečima, svaka operacija se izvršava tačno jednom na svake m vremenske jedinice u savijenoj arhitekturi. Ovo takođe znači da jedna FJ savijene arhitekture izvršava m operacija DSP algoritma.

Nakon dodele skupova savijanja može se pristupiti procesu sinteze savijene arhitekture.

5.2.2. Sinteza savijene arhitekture

Skupovi savijanja su arhitekturi sa sl. 5.8 dodeljeni na način opisan sa (5.5). Nakon dodele skupova savijanja je potrebno napisati jednačine savijanja za sve čvorove DFG-a i proveriti uslov savijanja $D_F(U \rightarrow V) \ge 0$. Da bi savijeni sistem mogao da se realizuje, uslov savijanja mora da bude ispunjen za sve potege u DFG.

U jednačini savijanja (4.1) se javljaju: stepen protočnosti operacija (P_U), faktor savijanja (N) i red savijanja čvorova U i V (u i v). Stepen protočnosti operacija TrDFG-a sa sl. 5.8 je $P_U=0$, a U i V su čvorovi TrDFG-a označeni sa $1,2,...,m,m+1,...,k \cdot m$.

Jednačine savijanja (4.1) za TrDFG-a sa sl. 5.8 i dodeljene skupove savijanja (5.4) su:

$$D_F(1 \rightarrow 2)=m\cdot 0-0+1-0=1$$
$$D_F(2 \rightarrow 3)=m\cdot 0-0+2-1=1$$

$$\dots$$

$$D_F(m-1\rightarrow m)=m\cdot 0-0+(m-1)-(m-2)=1$$

$$D_F(m\rightarrow m+1)=m\cdot 1-0+0-(m-1)=1$$

$$D_F(m+1\rightarrow m+2)=m\cdot 0-0+1-0=1$$

$$\dots$$

$$D_F(2m-1\rightarrow 2m)=m\cdot 0-0+(m-1)-(m-2)=1$$

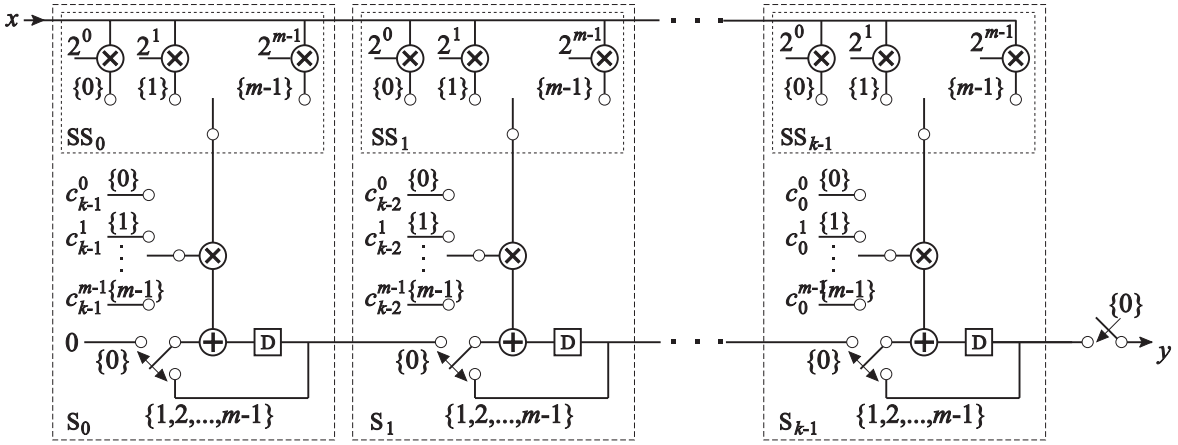
$$D_F(2m\rightarrow 2m+1)=m\cdot 1-0+0-(m-1)=1$$

$$D_F(2m+1\rightarrow 2m+2)=m\cdot 0-0+1-0=1$$

$$\dots$$

$$D_F(k\cdot m-1\rightarrow k\cdot m)=m\cdot 0-0+(m-1)-(m-2)=1. \qquad (5.6)$$

Uslov savijanja, $D_F(U\rightarrow V)\geq 0$, je zadovoljen za svaki par povezanih čvorova (U,V) što dokazuje pripremljenost TrDFG sa sl. 5.8 za primenu tehnike savijanja [21],[22]. Dobijena arhitektura savijenog FIR filtra sa k koeficijenta dužine m je prikazana na sl. 5.9.



Slika 5.9. Savijeni TrBP FIR filter

FJ savijene arhitekture su prikazane isprekidanim linijama i označene sa $S_0,S_1,\dots,S_{k-1}$ (sl. 5.9). Svaka FJ savijene arhitekture izvršava operacije iz jednog skupa savijanja. Kako jednom skupu savijanja pripada m operacija, svaka FJ izvršava m operacija izvorne arhitekture. Savijena arhitektura se sastoji od ukupno k FJ, gde je k broj koeficijenata filtra.

FJ S_0 izvršava operaciju množenja i akumulacije parcijalnih proizvoda nad bitovima koeficijenta c_{k-1} ($c_{k-1}^0, c_{k-1}^1,\dots, c_{k-1}^{m-1}$), S_1 nad bitovima koeficijenta c_{k-2} ($c_{k-2}^0, c_{k-2}^1,\dots, c_{k-2}^{m-1}$), itd. FJ S_{k-1} izvršava operacije nad bitovima koeficijenta c_0 ($c_0^0, c_0^1,\dots, c_0^{m-1}$).

Bitovi koeficijenata dovode se preko prekidača sa m stanja na množače FJ u vremenskim trenucima $N\cdot l+\{0\},N\cdot l+\{1\},\dots,N\cdot l+\{m-1\}$ i to: bit koeficijenta c_i težine 2^0 dovodi se na množač FJ S_i, u trenutku $N\cdot l+\{0\}$, bit težine 2^1 u trenutku $N\cdot l+\{1\}$, itd. Bit težine 2^{m-1} dovodi se na množač u trenutku $N\cdot l+\{m-1\}$. Ovo je na sl. 5.9 označeno sa $\{0\},\{1\},\dots,\{m-1\}$ uz oznaku odgovarajućeg bita.

Drugi operand množača je ulazni niz $[x_i]$. Ulazne reči se u množač uvode, takođe, preko prekidača sa m stanja, prethodno pomnožene faktorom 2^0, 2^1,..., ili 2^{m-1}. Blokovi za uvođenje

ulaznih reči označeni su na sl. 5.9 sa SS_i (i=0,1,...,k-1). Ulazna reč se u trenutku $N \cdot l+\{0\}$ množi faktorom 2^0, u trenutku $N \cdot l+\{1\}$ faktorom 2^1, itd. U trenutku $N \cdot l+\{m\text{-}1\}$ ulazna reč se množi faktorom 2^{m-1}. Ulazne reči [x_i] se preko *broadcast* linije dovode istovremeno svim FJ.

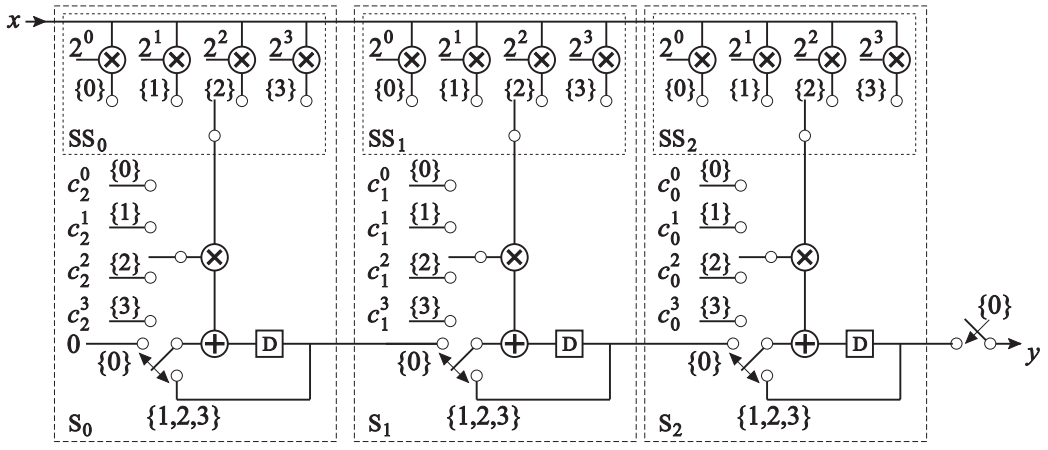
Akumulacija parcijalnih proizvoda realizovana je kolom za sabiranje, kašnjenjem (D) i povratnom vezom, koja u vremenskim trenucima {1,2,...m-1} vraća parcijalni proizvod na sabirač koji ga je proizveo. U vremenskim trenucima {0} na ulaz sabirača iz FJ S_0 dovodi se konstanta 0, dok se svim ostalim sabiračima FJ S_i (i=0,1,..,k-1) dovodi parcijalni proizvod iz prethodne FJ (S_{i-1}) (sl. 5.9). Izlazni niz [y_i] se dobija na svakih m taktnih intervala, tj. u vremenskim trenucima $N \cdot l+\{0\}$.

Princip FIR filtriranja na savijenoj arhitekturi je dat u daljem tekstu.

5.2.3. FIR filtriranje na savijenoj arhitekturi

Kod savijene arhitekture sa sl. 5.9, faktor savijanja N jednak je dužini koeficijenata m (N=m). Drugim rečima, broj funkcionalnih jedinica je redukovan za faktor savijanja N na račun vremena, što znači da arhitektura savijenog TrBP FIR filtra sa sl. 5.9 generiše jednu reč izlaznog niza [y_i] na svaka m taktna intervala. Takođe, ovo znači da se reči ulaznog niza [x_i] ne uvode u arhitekturu u svakom taktnom intervalu, kao što je bio slučaj kod izvorne arhitekture (sl. 3.2 i sl. 3.3), već se reči ulaznog niza [x_i] dovode na ulazne linije savijene arhitekture na m taktnih intervala ($N \cdot l+\{0\}$) i prisutne su na ulaznim linijama m taktnih intervala [21].

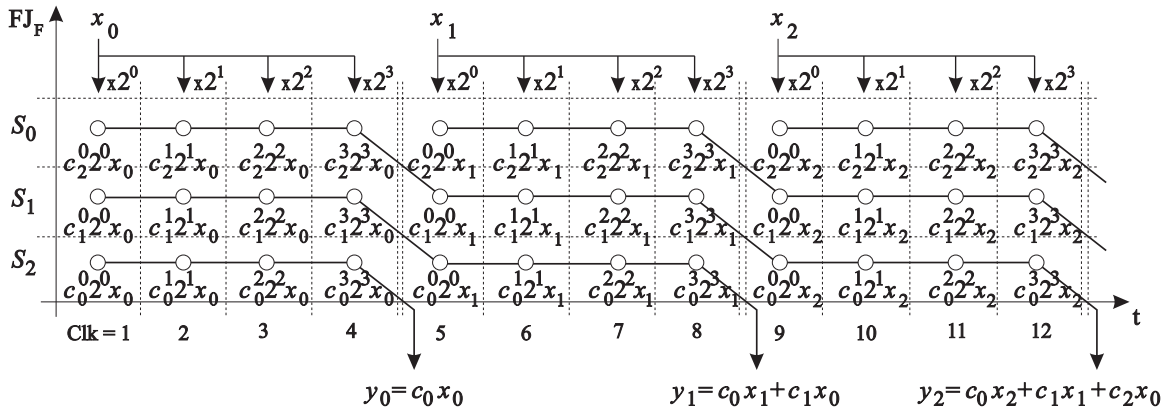
Princip FIR filtriranja na savijenom BP polju biće objašnjena za slučaj k=3 i m=4. DFG savijenog TrBP FIR filtra za k=3 i m=4 prikazan je na sl. 5.10. Na sl. 5.11 prikazan je tok podataka za ovu arhitekturu.



Slika 5.10. DFG savijenog TrBP FIR filtra za k=3 i m=4

U toku prva četiri taktna intervala, na ulaznim linijama arhitekture nalazi se ulazna reč x_0 (sl. 5.11). Ova ulazna reč se množi faktorom 2^0 i preko *broadcast* linije dovodi na ulaze funkcionalnih jedinica na način prikazan na sl. 5.10. FJ S_0 u toku prvog taktnog intervala množi ulaznu reč 2^0x_0

bitom koeficijenta c_0 težine 2^0. Na izlazu FJ S_0 se na kraju prvog taktnog intervala nalazi parcijalni proizvod $c_2^0 2^0 x_0$.



Slika 5.11. Tok podataka kroz arhitekturu savijenog TrBP FIR filtra za *k*=3 i *m*=4

Na isti način FJ S_1 i S_2 u toku prvog taktnog intervala formiraju parcijalne proizvode $c_1^0 2^0 x_0$ i $c_0^0 2^0 x_0$, respektivno (sl. 5.11). U narednom taktnom intervalu FJ S_0, S_1 i S_2 formiraju redom parcijalne proizvode $c_2^1 2^1 x_0$, $c_1^1 2^1 x_0$ i $c_0^1 2^1 x_0$. Ovi parcijalni proizvodi se sabiraju sa parcijalnim proizvodima izračunatim u prethodnom taktnom intervalu, tako da je na izlaznim linijama FJ S_0 dostupan zbir parcijalnih proizvoda $c_2^0 2^0 x_0 + c_2^1 2^1 x_0$, na izlaznim linijama FJ S_1 dostupan zbir parcijalnih proizvoda $c_1^0 2^0 x_0 + c_1^1 2^1 x_0$, a na izlaznim linijama FJ S_2 je dostupan zbir parcijalnih proizvoda $c_0^0 2^0 x_0 + c_0^1 2^1 x_0$ [23],[24].

Na kraju četvrtog (*m*-tog) taktnog intervala na izlaznim linijama FJ S_0 je dostupan međurezultat $c_2 x_0$ ($c_2^0 2^0 x_0 + c_2^1 2^1 x_0 + c_2^2 2^2 x_0 + c_2^3 2^3 x_0$). Na izlaznim linijama FJ S_1 i S_2 nalaze su međurezultati $c_1 x_0$ i $c_0 x_0$, respektivno. Rezultat iz FJ S_2 prosleđuje se na izlaz arhitekture kao prva izlazna reč $y_0 = c_0 x_0$.

U naredna četiri taktna intervala, na ulaznim linijama arhitekture nalazi se ulazna reč x_1 (sl. 5.11). Međurezultat $c_1 x_0$ se iz FJ S_1 prosleđuje u S_2 gde se sabira sa parcijalnim proizvodom koji je ova FJ odredila u petom taktnom intervalu. Na izlaznim linijama FJ S_2 na kraju petog taktnog intervala nalazi se među-rezultat $c_2 x_0 + c_2^0 2^0 x_1$, itd.

Na kraju osmog taktnog intervala na izlaznim linijama savijene arhitekture dostupna je izlazna reč $y_1 = c_0 x_1 + c_1 x_0$ (sl. 5.11).

Rezultat je dostupan na svakih *m* taktnih intervala, i inicijalno kašnjenje savijene TrBP arhitekture je, takođe, *m* taktnih intervala. Arhitektura prikazana na sl. 5.9 realizuje izračunavanje (3.1).

U cilju implementacije promenljivog faktora savijanja, sinteza savijene arhitekture je izvedena u opštem obliku sa *k* ćelija i dužinom koeficijenata *m* (sl. 5.9). U narednom odeljku prikazana je analiza parametara arhitekture i sinteza arhitekture savijenog TrBP FIR filtra sa promenljivim faktorom savijanja.

5.2.4. Arhitektura savijenog transponovanog "bit-plane" FIR filtra sa promenljivim faktorom savijanja

Površina silicijuma koja je potrebna za implementaciju savijenog TrBP FIR filtra je smanjena približno N puta u odnosu na površinu potrebnu za implementaciju BP FIR filtra, dok je vreme potrebno za izračunavanje povećano za isti faktor. Izlazne reči savijenog TrBP FIR filtra su dostupne na izlaznim linijama filtra na svakih $N=m$ taktnih intervala, za razliku od BP ili TrBP FIR filtra, kod koga se po jedna izlazna reč dobija u svakom taktnom intervalu. Vreme potrebno za izračunavanje je direktno proporcionalno faktoru savijanja. Uvođenjem mogućnosti da savijena arhitektura može da smanji faktor savijanja u slučajevima kada se izračunavanja vrše nad koeficijentima manje dužine od nominalne, brzina izračunavanja se može povećati [21], [22].

U cilju sinteze savijene arhitekture sa mogućnošću promene faktora savijanja, potrebno je uočiti i izdvojiti delove savijene arhitekture na koje faktor savijanja (N) ima uticaja, kako bi dodatnom upravljačkom logikom omogućili promenu faktora savijanja, odnosno dužine koeficijenata ($m=N$). Sa sl. 5.9 može se uočiti da dužina koeficijenata m utiče na broj stepena slobode prekidača za uvođenje bitova koeficijenata u množač. Drugo, kod uvođenja niza ulaznih reči u arhitekturu broj stepena slobode prekidača za uvođenje niza jednak je m. Komponente za množenje ulaznih reči x_i faktorom 2^i ($0 \le i \le m-1$) zavise od dužine koeficijenata m (sl. 5.9). Takođe, od dužine koeficijenata zavisi i upravljački signal za upravljanje položajem prekidača na povratnoj vezi iz sabirača. Na sl. 5.12 prikazan je funkcionalni blok dijagram savijenog TrBP FIR fitra sa promenljivim faktorom savijanja.

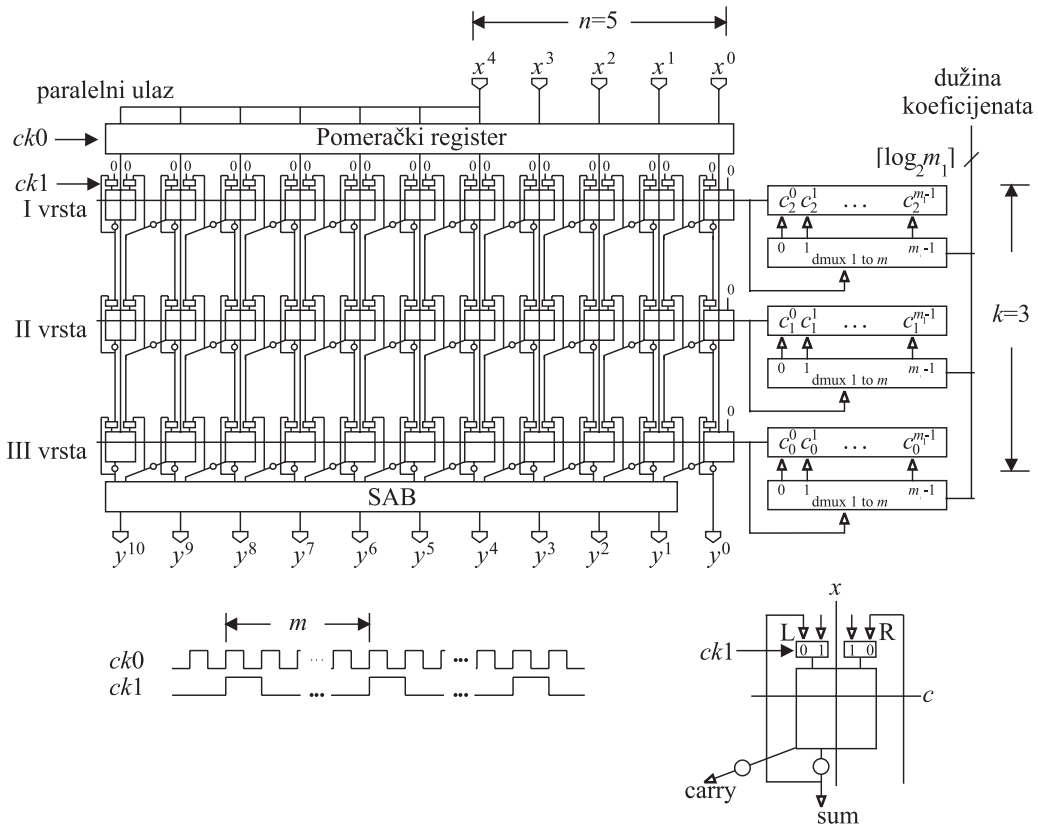
Komponente za množenje ulaznih reči x_i faktorom $2^0, 2^1, \ldots, 2^{m-1}$ ($SS_0, SS_1, \ldots, SS_{k-1}$) su implementirane korišćenjem pomeračkog registra. Trajanje izračunavanja nad jednim koeficijentom zavisi od dužine koeficijenata m, pa na osnovu toga promena dužine koeficijenata ne menja broj skupova savijanja, već utiče na promenu broja operacija u okviru skupa savijanja. Pretpostavimo da dužina koeficijenata može da varira u opsegu $1 \le m \le m_1$, gde je m_1 gornja granica promene dužine koeficijenata. Gornja granica promene dužine koeficijenata, m_1, određena je implementiranom širinom registara za koeficijente. Na sl. 5.12 je prikazan funkcionalni blok dijagram savijene arhitekture sa sl. 5.9, sa izmenljivim faktorom savijanja za $k=3$, $n=5$ i $1 \le m \le m_1$.

Svaki skup savijanja sa sl. 5.9 implementiran je jednom vrstom osnovnih ćelija (sl. 5.12) [23]. Linije x^4 do x^0 uvode u polje bitove ulaznog niza $[x_i]$, prethodno pomnožene faktorom 2^i ($0 \le i \le m-1$). Množenje faktorom 2^i je implementirano pomeračkim registrom na ulazu u arhitekturu. Ulazna reč x_i se upisuje u pomerački registar na svakih m perioda takta. Bitovi ulazne reči se zatim istovremeno dovode svim vrstama osnovnih ćelija.

Linija c uvodi bitove koeficijenata c^j ($j=0, \ldots, m-1$) u odgovarajuće vrste polja. Multiplekseri, dodati svakoj osnovnoj ćeliji, označeni sa L i R na sl. 5.12, upravljaju sumom i prenosom iz FJ tako što ih, ili vraćaju u FJ koja ih je proizvela, ili prenose u narednu FJ. Na ovaj način je implementiran savijeni put svake FJ sa sl. 5.9.

Jedna vrsta osnovnih ćelija predstavlja jednobitno množenje, odnosno formiranje parcijalnog proizvoda i akumulaciju na prethodno izračunati parcijalni proizvod. Jedna vrsta množi ulaznu reč

koeficijentom dužine m za m taktnih intervala i prosleđuje akumulirani rezultat narednoj vrsti. Završno sumiranje obavlja sabirač vezan na izlaze poslednje vrste (sl. 5.12).



Slika 5.12. Funkcionalni blok dijagram savijenog TrBP FIR fitra sa promenljivim faktorom savijanja (*k*=3, *n*=5 i $1 \le m \le m_1$)

Kako je m_1 maksimalna dozvoljena dužina koeficijenata, veličina polja osnovnih ćelija je $k \cdot (m_1 + n + \lceil \log_2 k \rceil)$. Odgovarajući sabirač je dodat k-toj vrsti polja za izračunavanje krajnjih rezultata.

Promenu vrednosti faktora savijanja omogućavaju [21], [22]:

1. promena periode upravljačkog signala *ck*1 (sl. 5.12); potrebno je da perioda ovog upravljačkog signala bude jednaka m, odnosno $m=N$ perioda osnovnog taktnog signala *ck*0;

2. upravljačka logika pridružena *shift/rotate* registrima za uvođenje koeficijenata u arhitekturu, koja skraćuje dužinu registara na m bitova (sl. 5.12).

Broj skupova savijanja je konstantan i jednak broju koeficijenata, dok je broj operacija u skupu savijanja promenljiv i jednak je dužini koeficijenata (m). Dužina koeficijenata može da varira u opsegu $1 \le m \le m_1$, gde m_1 označava maksimalnu dužinu koeficijenata definisanu implementiranom širinom registara. Vreme izračunavanja linearno zavisi od dužine koeficijenata.

Modifikacijom savijene arhitekture uvedena je mogućnost promene faktora savijanja u zavisnosti od dužine koeficijenata. Za slučaj kada arhitektura vrši izračunavanje nad skupom koeficijenta manje dužine od nominalne, umesto proširenja koeficijenata nulama, predložena arhitektura redukuje faktor savijanja prema dužini koeficijenata, čime se povećava propusnost arhitekture m_1/m puta.

Predložena transformacija BP arhitekture omogućila je uspešnu primenu tehnike savijanja na TrBP FIR filtru. Sinteza savijene arhitekture je izvedena u parameatrskom obliku sa k koeficijenta i dužinom koeficijenata m. Površina silicijuma koja je potrebna za implementaciju filtra je smanjena približno N puta na račun vremena. Broj osnovnih ćelija je sveden na broj osnovnih ćelija u jednom BP elementu osnovne arhitekture. Takođe, ukupan broj lečeva odgovara broju lečeva u jednom BP elementu BP FIR filtra. Kritičan put je produžen za kašnjenje kroz multiplekser, tako da je perioda osnovnog takta neznatno povećana. Propusnost je smanjena za N puta u odnosu na izvornu BP arhitekturu.

Uočeni su delovi arhitekture na koje faktor savijanja ima uticaja. Dodatnom upravljačkom logikom je omogućena promena dužine koeficijenata (m), odnosno faktora savijanja (N). U odnosu na izvornu BP arhitekturu, svaka ćeliji savijenog BP FIR filtra ima dva dodatna multipleksera. Predložena arhitektura redukuje faktor savijanja prema dužini koeficijenata, čime je moguće povećati propusnost m_1/m puta.

5.3. SAVIJENO SEMI-SISTOLIČKO POLJE ZA FIR FILTRIRANJE SA IZMENJIVIM BROJEM I DUŽINOM KOEFICIJENATA

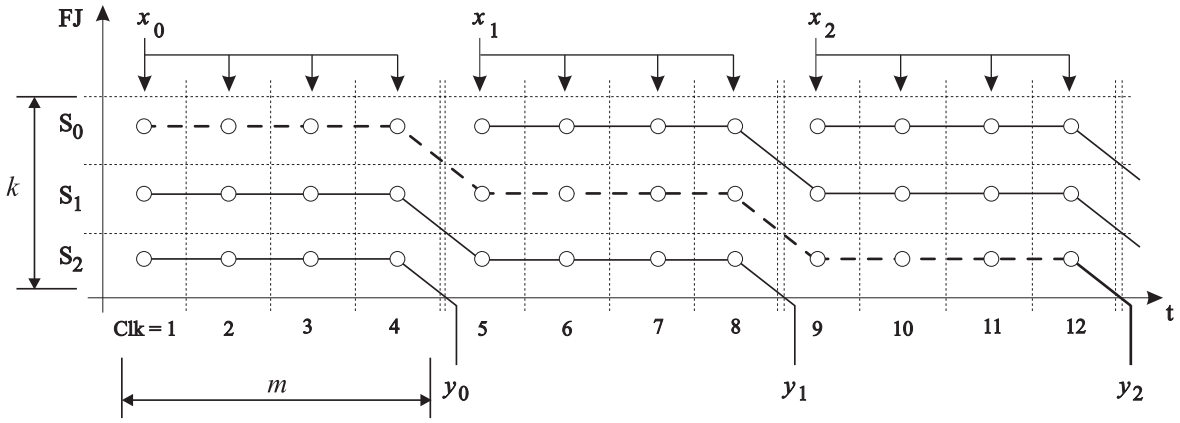
U prethodnom odeljku je prikazan postupak sinteze savijenog semi-sistoličkog polja za FIR filtriranje sa izmenjivim faktorom savijanja. Realizovana savijena arhitektura dozvoljava promenu faktora savijanja na polju konstantne veličine u zavisnosti od dužine koeficijenata. Cilj promene faktora savijanja jeste proširenje oblasti primene semi-sistoličkih polja za FIR filtriranje na taj način što se propusnost savijenog sistema povećava u slučaju vršenja izračunavanjna sa manjom preciznošću. Međutim, dobijena arhitektura ne pruža mogućnost promene broja koeficijenata [26].

U ovom odeljku predstavljeno je savijeno semi-sistoličko polje za FIR filtriranje sa izmenjivim brojem i dužinom koeficijenata. Ovo polje je projektovano u cilju dodatnog proširenja oblasti primene semi-sistoličkih polja na sisteme kod kojih je u toku rada potrebno menjati parametre izračunavanja. Prikazan je postupak sinteze savijenog semi-sistoličkog polja sa mogućnošću promene broja i dužine koeficijenata. Polazna arhitektura u postupku sinteze savijenog polja je arhitektura TrBP FIR filtra (sl. 5.6).

U ovom odeljku će postupak sinteze savijenog semi-sistoličkog polja za FIR filtriranje sa izmenjivim brojem i dužinom koeficijenata biti detaljno objašnjen. Savijeno polje će biti prikazano grafom toka podataka i funkcionalnim blok dijagramom. Princip filtriranja na savijenom polju će biti detaljno objašnjen. Biće prikazan postupak sinteze algoritama za rekonfiguraciju polja, kao i način podešavanja broja i dužine koeficijenata. Savijenom polju sa mogućnošću promene broja i dužine koeficijenata, u cilju ubrzanja izračunavanja kod aplikacija kod kojih je potrebna manja preciznost, biće dodata mogućnost smanjenja faktora savijanja. Sinteza polja sa mogućnošću smanjenja faktora savijanja biće detaljno opisana. Modifikacije algoritama za rekonfiguraciju polja u slučaju polja sa dodatnom mogućnošću smanjenja faktora savijanja biće date.

5.3.1. Dodela skupova savijanja

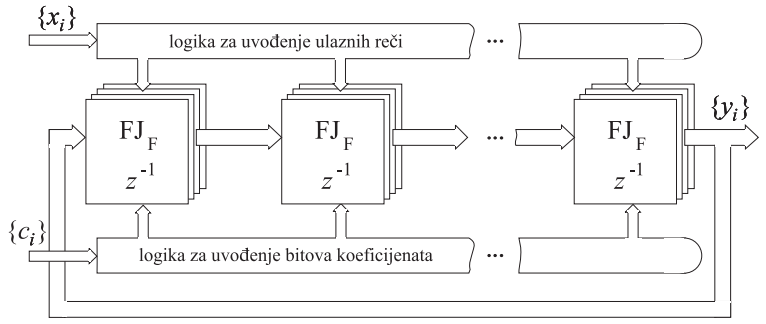
Skupovi savijanja, dodeljeni prilikom sinteze savijenog semi-sistoličkog polja za FIR filtriranje sa promenljivim faktorom savijanja (odeljak 5.2), nisu omogućili sintezu savijenog FIR filtra sa promenljivim brojem koeficijenata. Razlog je moguće uočiti sa skice toka podataka ove arhitekture, prikazane na sl. 5.13 za k=3 i m=4. Isprekidanom linijom na sl. 5.13 je prikazan tok podataka kojim se određuje izlazna reč (y_2).



Slika 5.13. Skica toka podataka kroz savijeno TrBP polje za FIR filtriranje sa promenljivim faktorom savijanja

Kod toka podataka u savijenoj arhitekturi (sl. 5.13) ne postoje povratni tokovi iz poslednje FJ (S_2) u prvu (S_0). Kako je dužina koeficijenata (m) kod ove arhitekture promenljiva, menja se samo broj operacija u okviru skupa savijanja, dok broj koeficijenata filtra ostaje jednak broju skupova savijanja, i nije ga moguće menjati.

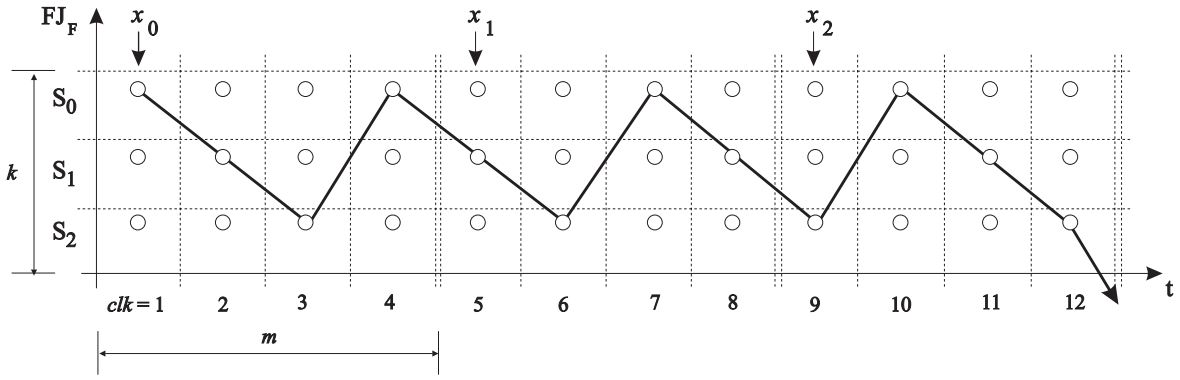
U cilju uspešne sinteze arhitekture sa mogućnošću izmene broja koeficijenata, potrebno je saviti polaznu arhitekturu tako da postoje tokovi podataka od poslednje FJ ka prvoj. Takođe, broj skupova savijanja ne sme da zavisi od broja funkcionalnih jedinica, tj. cilj je da ova dva parametra u opštem slučaju ne budu jednaka. Arhitektura jednog mogućeg rešenja je prikazana na sl. 5.14, a skica odgovarajućeg toka podataka kroz arhitekturu je data na sl. 5.15.



Slika 5.14. Arhitektura savijenog BP FIR filtra sa promenljivim brojem koeficijenata

Kod savijene arhitekture sa sl. 5.14 broj skupova savijanja, u opštem slučaju, nije jednak broju FJ savijenog sistema (FJ_F) [26]. Za razliku od arhitekture sa sl. 5.7, arhitektura prikazana na sl. 5.14 formiranje parcijalnog proizvoda ulazne reči i koeficijenta c_i može da izvrši na bilo kojoj FJ. Izbor

FJ koja će izvršiti operaciju vrši se u zavisnosti od broja i dužine koeficijenata filtra. Zbog toga ovakva arhitektura mora posedovati logiku za uvođenje ulaznih reči, kao i logiku za uvođenje bitova koeficijenata u tačno određenim vremenskim trenucima u određene FJ savijene arhitekture sl. 5.14.



Slika 5.15. Skica novog toka podataka kroz savijenu arhitekturu gde postoje tokovi podataka od poslednje FJ ka prvoj

U postupku sinteze savijenog semi-sistoličkog polja za FIR filtriranje sa promenljivim brojem i dužinom koeficijenata koristićemo sledeće oznake:

k_C – broj koeficijenata,

m_C – dužina koeficijenata,

k – broj skupova savijanja,

c_i^j – bit koeficijenta c_i težine 2^j,

L – ukupan broj operacija DFG-a ,

p – pozicija operacije u okviru DFG-a i

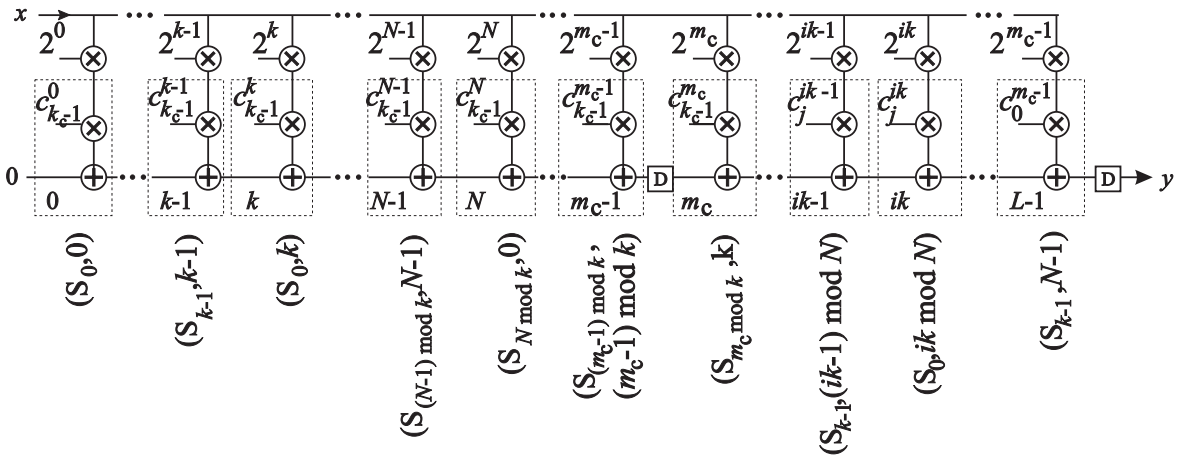
$(S_S \mid r)$ –operacija se izvršava u skupu savijanja S_S u vremenskim trenucima $N \cdot l + \{r\}$.

Način dodele skupova savijanja direktno utiče na izgled toka podataka kroz savijenu arhitekturu [26]. Skupovi savijanja koji su uslovili tok podataka sa sl. 5.13 su dati sa (5.4) i (5.5).

S obzirom na skicu toka podataka sa sl. 5.15, dodelimo TrDFG-u TrBP arhitekture sa sl. 5.6 skupove savijanja $(S_S \mid r)$ na sledeći način:

$$\begin{aligned} s &= (p-1) \bmod k \\ r &= (p-1) \bmod N. \end{aligned} \tag{5.7}$$

Kako je tok podataka kroz arhitekturu takav da operacija na poziciji p ($1 \le p \le k \cdot m$) akumulira parcijalni proizvod na međurezultat dobijen izvršenjem operacije na poziciji p-1, način dodele skupova savijanja (5.7) obezbediće povratni tok podataka iz poslednje funkcionalne jedinice u prvu [26]. Graf toka podataka TrBP arhitekture sa označenim skupovima savijanja (5.7) prikazan je na sl. 5.16.



Slika 5.16. TrDFG sa dodeljenim skupovima savijanja

Primenom matematičkog modela dodele skupova savijanja (5.7), imajući u vidu oznake operacija algoritma sa sl. 5.16, skupovi savijanja su:

$$S_0=[0,k,2k,\ldots,(m-1)k]$$
$$S_1=[1,k+1,2k+1,\ldots,(m-1)k+1]$$
$$S_2=[2,k+2,2k+2,\ldots,(m-1)k+2]$$
$$\ldots$$
$$S_i=[i,k+i,2k+i,\ldots,(m-1)k+i]$$
$$\ldots$$
$$S_{k-1}=[k-1,2k-1,3k-1,\ldots,m\cdot k-1]. \qquad (5.8)$$

Nakon dodele skupova savijanja potrebno je proveriti uslov savijanja za svaki poteg u grafu. U daljem tekstu je prikazan postupak sinteze savijenog semi-sistoličkog polja za skupove savijanja opisane sa (5.7) i (5.8).

5.3.2. Sinteza savijene arhitekture

FIR filter sa sl. 5.16 ima k_C koeficijenta dužine m_C, tako da je ukupan broj operacija u DFG-u jednak $L=k_C\cdot m_C$. Ukupan broj operacija je parametar arhitekture koji je potrebno zadržati konstantnim kroz transformaciju savijanja:

$$L=k_C\cdot m_C=k\cdot N. \qquad (5.9)$$

S obzirom na cilj transformacije savijanja po pogledu izmenjivosti broja koeficijenata i dužine koeficijenata, potrebno je imati na umu to da će k_C i m_C biti promenljivi parametri ciljne, savijene arhitekture, dok će parametri k i N ustvari definisati veličinu polja na kome se obavlja filtriranje [27].

Novi način dodele skupova savijanja (5.7) tj.(5.8), uvodi k skupova savijanja. U opštem slučaju broj skupova savijanja nije jednak broju koeficijenata FIR filtra (k_C). Jednačine savijanja (4.1) za ovako dodeljene skupove savijanja, predstavljene u [26], su:

$$D_F(1\rightarrow 2)=N\cdot 0-0+1-0=1$$

$$D_F(2\rightarrow 3)=N\cdot 0-0+2-1=1$$

…

$$D_F(k\rightarrow k+1)=N\cdot 0-0+k-(k-1)=1$$

…

$$D_F(N\rightarrow N+1)=N\cdot 0-0+0-(N-1)=-(N-1)$$

…

$$D_F(m_C\rightarrow m_C+1)=N\cdot 1-0+(m_C \bmod k)-((m_C-1)\bmod k)=N+1$$

…

$$D_F(n\cdot k\rightarrow n\cdot k+1)=N\cdot 0-0+(n\cdot k \bmod k)-((n\cdot k-1)\bmod k)=1$$

…

$$D_F(L-1\rightarrow L)=N\cdot 0-0+(N-1)-(N-2)=1. \qquad (5.10)$$

Stepen protočnost svih čvorova DFG-a je Pu=0, a U i V su čvorovi transformisanog DFG-a (sl. 5.16) označeni p = 1, 2, 3, …, k, k+1, …, N,N+1,…,m_C,m_C+1,…,$n\cdot k$,$n\cdot k$+1, …, L-1, L ($1\le n\le N$).

Opšti oblik jednačina savijanja (5.10) je:

$$D_F(p\rightarrow p+1)=N\cdot w(e)-1+[(p+1) \bmod N]-[p \bmod N] =$$

$$=\begin{cases} -N+1, p \bmod N = 0 \\ N+1, p \bmod m_C = 0, 1\le p\le L-1. \\ 1, ostalo \end{cases} \qquad (5.11)$$

U DFG-u sa sl. 5.16 postoje kašnjenja između koeficijenata c_i i c_{i+1}, $i=k_C-2, k_C-1,\ldots,0$, a ne između skupova savijanja. Ova kašnjenja dovode do pojave negativnih vrednosti u jednačinama savijanja (5.11) za grane koje izviru iz N-tih čvorova DFG-a:

$$p_U \bmod N = 0. \qquad (5.12)$$

Jednačina savijanja za čvorove U i V za koje važi (5.12) ima za rezultat negativan broj kašnjenja. U ovom slučaju, bez vremenskog usklađivanja rada sistema, sistem nije moguće saviti [28]. Na broj kašnjenja između čvorova možemo uticati i vremenski uskladiti sistem tako da je savijanje moguće. Tehnika koja ovo omogućava je tehnika vremenskog usklađivanja (poglavlje 4, odeljak 4.3).

5.3.3. Vremensko usklađivanje grafa toka podataka TrBP arhitekture

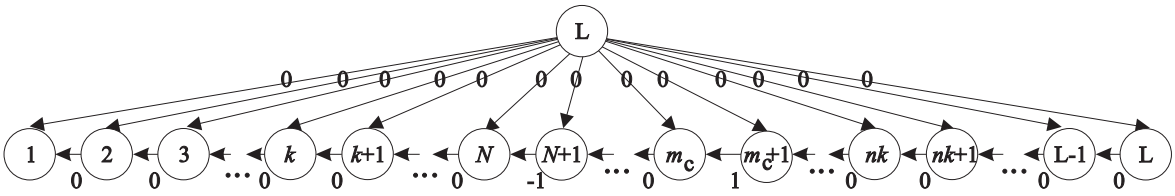
Korišćenjem sistema nejednačina za vremensko usklađivanje (4.5) i sistema jednačina savijanja (5.11) dobija sistem nejednačina predstavljen u [28]:

$$r(p)-r(p+1)\le\begin{cases}-1, & p \bmod N = 0\\ 1, & p \bmod m_C = 0\\ 0, & ostalo\end{cases},\quad 1\le p\le \mathrm{L}-1. \qquad (5.13)$$

Sistem nejednačina (5.13) može se rešiti korišćenjem grafa ograničenja (poglavlje 4, odeljak 4.3).

5.3.3.2. Rešenje za vremensko usklađivanje grafa toka podataka TrBP arhitekture

Graf ograničenja je, kako je ranije rečeno, usmereni graf formiran tako da za svaku nepoznatu r(p) iz sistema nejednačina u grafu postoji po jedan čvor. Svaka nejednačina tipa r(U)-r(V)≤x je prikazana usmerenom granom grafa od čvora V ka čvoru U kojoj je pridružena težina x. Grafu se dodaje još jedan čvor, u ovom slučaju L-ti po redu, koji ima usmerene grane ka svim ostalim čvorovima. Težina svih grana čiji je izvor u čvoru L je jednaka nuli. Graf ograničenja za sistem nejednačina (5.13) je prikazan na sl. 5.17. Rešenje sistema nejednačina za nepoznatu r(p) predstavlja vrednost sume težina grana najkraćeg puta u grafu od čvora L do čvora koji odgovara nepoznatoj r(p).



Slika 5.17. Graf ograničenja sistema nejednačina za vremensko usklađivanje

Za traženje najkraćeg puta kroz graf ograničenja koristimo osobine koje poseduje ovaj graf. Karakteristične osobine grafa ograničenja sa sl. 5.17 su:

- sve grane grafa su usmerene od čvorova sa većim ka čvoru sa manjim indeksom pozicije (p),
- ne računajući L-ti čvor, u grafu postoje grane samo između čvorova čiji se indeksi razlikuju za jedan,
- moguće je matematički izraziti zavisnost težine grane (w_p) koja utiče u čvor p kao:

$$w_p \le\begin{cases}-1, & p \bmod N = 0\\ 1, & p \bmod m_C = 0\\ 0, & ostalo\end{cases},\quad p=1,2,\ \ldots\ ,\ \mathrm{L}-1.$$

Na osnovu prethodno navedenih osobina možemo izračunati broj grana koje imaju težinu w_p=-1 i broj grana koje imaju težinu w_p=1. Broj grana koje imaju težinu w_p=-1, počev od čvora L-1 do čvora p (sl. 5.17), je:

$$\left\lfloor \frac{\mathrm{L}-p}{N} \right\rfloor, \tag{5.14}$$

dok je broj grana koje imaju težinu w_p=1 počev od čvora L-1 do čvora p (sl. 5.17)

$$\left\lfloor \frac{\mathrm{L}-p}{m_c} \right\rfloor. \tag{5.15}$$

Rešenje za vremensko usklađivanje čvora na poziciji p je zbir težina grana od čvora L-1 do čvora p. Grane na tom putu imaju težine 0, 1 ili -1. Kako je broj grana sa težinom 1 određen izrazom (5.15), a broj grana sa težinom –1 određen sa (5.15), opšti oblik rešenja za vremensko usklađivanje je [28]:

$$r(p)=\left\lfloor \frac{\mathrm{L}-p}{m_c} \right\rfloor-\left\lfloor \frac{\mathrm{L}-p}{N} \right\rfloor. \tag{5.16}$$

Uvedimo ograničenje u (5.16), takvo da je uslov r(p)≤0 uvek ispunjen. Tada imamo da je $m_C>N$, odnosno na osnovu (5.9), $k_C<k$.

Za $k_C=k$ javlja se specijalni slučaj. Ukoliko je broj koeficijenta jednak broju skupova savijanja ($k_C=k$) vremensko usklađivanje nije potrebno. Jednačine savijanja za ovaj slučaj su:

$D_F(0\rightarrow 1)=N\cdot 0-0+1-0=1$

$D_F(1\rightarrow 2)=N\cdot 0-0+2-1=1$

…

$D_F(k-1\rightarrow k)=N\cdot 0-0+k-(k-1)=1$

…

$D_F(m_C-1\rightarrow m_C)=N\cdot 1-0+(m_C \bmod k)-((m_C-1) \bmod k)=N+1$

…

$D_F(i\cdot k-1\rightarrow i\cdot k)=N\cdot 0-0+(i\cdot k \bmod k)-((i\cdot k-1) \bmod k)=1$

…

$D_F(\mathrm{L}-2\rightarrow \mathrm{L}-1)=N\cdot 0-0+(N-1)-(N-2)=1.$

Za slučaj $m_C=N$ (tj. $k_C=k$) grane (N-1→N), koje ne ispunjuju uslov savijanja u sistemu jednačina savijanja (5.10) sada sadrže dodatno kašnjenje između svaka dva koeficijenta. U ovom slučaju, jednačina savijanja za granu (N-1→N) ima sledeći oblik:

$$D_F(N-1\rightarrow N)=N\cdot 1-0+0-(N-1)=1.$$

U cilju nalaženja karakterističnih vrednosti za sintezu savijene arhitekture kao što su vremenski trenuci u kojima se ulazne reči uvode u arhitekturu, broj taktnih intervala za koje je ulazna reč aktivna na ulaznim linijama, minimalni broj potrebnih registara, neophodno je izvršiti analizu dobijenog rešenja za vremensko usklađivanje.

5.3.3.3. Analiza rešenja za vremensko usklađivanje

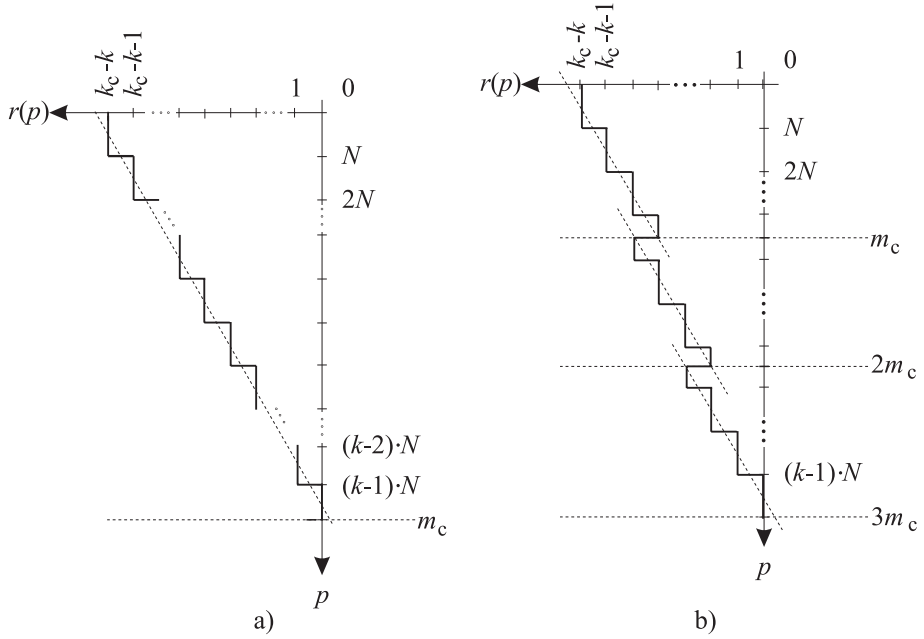
Granične vrednosti rešenja za vremensko usklađivanje (5.16) su:

$$r(1) = \left\lfloor \frac{L-1}{m_c} \right\rfloor - \left\lfloor \frac{L-1}{m_c} \right\rfloor = (k_c - 1) - (k-1) = k_c - k\,, \text{ i}$$

$$r(L) = \left\lfloor \frac{0}{m_c} \right\rfloor - \left\lfloor \frac{0}{N} \right\rfloor = 0\,. \qquad (5.17)$$

Na osnovu (5.17) dobijamo da je opseg vrednosti u kojim se može naći rešenje za vremensko usklađivanje nekog čvora $k_c - k \le r(i) \le 0$. Grafički prikaz rešenja sistema nejednačina za vremenskog usklađivanja (5.16) je dat na sl. 5.18. Na *p*-osi su prikazane oznake indeksa čvorova, a na r-osi vrednosti za vremensko usklađivanje.

Kako važi ograničenje $k_C<k$, vrednosti za vremensko usklađivanje se nalaze na negativnom delu r-ose (sl. 5.18). Iz rešenja skupa nejednačina za vremensko usklađivanje (5.16), odnosno sa sl. 5.18, se vidi da je broj potrebnih, sukcesivnih, ulaznih reči [x_i] koje jednovremeno moraju biti prisutne kao operandi na ulazima FJ, jednak broju celih brojeva u opsegu graničnih vrednosti rešenja za vremensko usklađivanje (5.17) i jednak je $k - k_c + 1$.



Slika 5.18. Grafički prikaz rešenja sistema nejednačina za vremensko usklađivanje: a) za slučaj k_C=1 i m_C=L; b) za slučaj k_C=3 i m_C=L/3

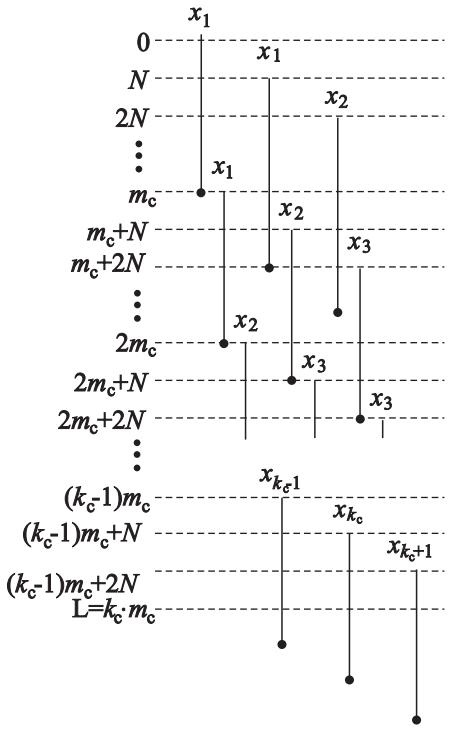
Sa druge srtane, ulazna reč x_i se koristi kao operand u $k - k_c + 1$ uzastopnih izračunavanja. S obzirom da jedno izračunavanje u savijenoj arhitekturi traje *N* taktnih intervala, za izabrani faktor savijanja *N* (*N*>0), savijena arhitektura koristi ulaznu reč $T_{x_i} = (k - k_c + 1) \cdot N$ vremenskih jedinica.

Za određivanje minimalnog broja registara potrebnih za pamćenje ulaznih reči koristimo činjenicu da se ulazna reč x_i koristi kao operand u $k - k_c + 1$ uzastopnih izračunavanja. Minimalni

broj potrebnih registara je jednak maksimalnom broju istovremeno aktivnih ulaznih promenljivih. S obzirom da je cilj sinteze da savijena arhitektura ima promenljiv broj koeficijenata, minimalni broj potrebnih registara se dobija računanjem maksimuma izraza $k - k_c + 1$. Maksimum izraza $k - k_c + 1$ se dobija za minimalni broj koeficijenata k_C (k_C=1). Iz ovoga sledi da je maksimalni broj istovremeno korišćenih ulaznih reči u sistemu $R = k - 1 + 1 = k$, što predstavlja minimalni potreban broj registara za pamćenje ulaznih reči.

Reči ulaznog niza [x_i] se u savijenu arhitekturu uvode svakih N taktnih intervala. Takođe, na osnovu sl. 5.18 se dolazi do zaključka da je potrebno i u svakom $n \cdot m_c$, n=1,2,...,k_C-1 taktnom intervalu uvesti ulaznu reč u sistem.

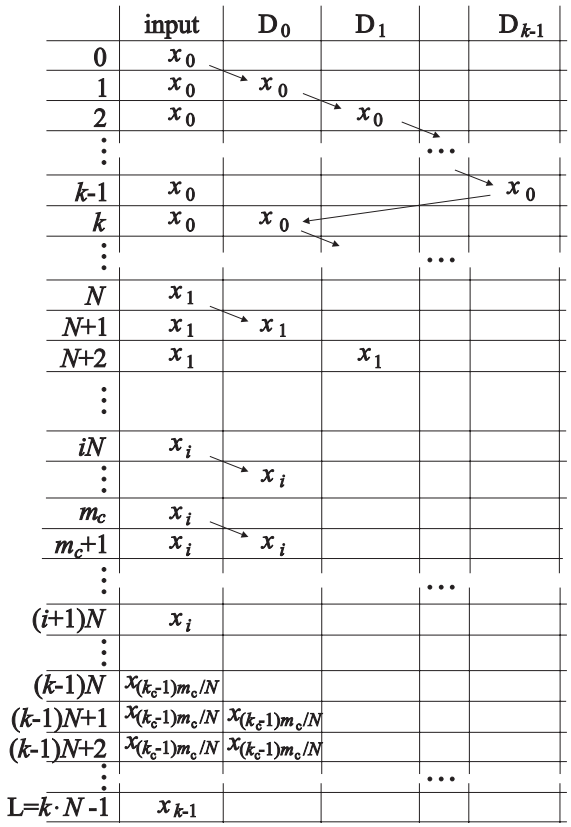
Na osnovu prethodne analize dobijamo opšti oblik linearnog grafikona aktivnosti promenljivih (sl. 5.19).



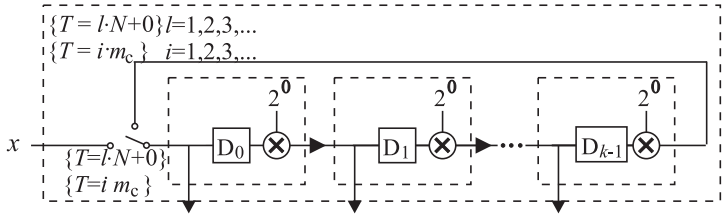
Slika 5.19. Linearni grafikon aktivnosti ulaznih promenljivih za sistema nejednačina za vremensko usklađivanje sa sl. 5.18

Korišćenjem linearnih grafikona aktivnosti sa sl. 5.19, kao i činjenice da je maksimalni broj istovremeno korišćenih ulaznih reči u sistemu $R = k$, formiran je opšti oblik alokacione tabele i prikazan na sl. 5.20 [27]. Razmeštaj podataka u registre može se izvesti pomoću alokacione tabele. Alokaciona šema određuje vremenski raspored operanada u registrima.

Sinteza hardvera za uvođenje reči ulaznog niza [x_i] izvršena je na osnovu alokacione tabele. Hardverski modul za uvođenje ulaznih reči je prikazan na sl. 5.21. Ovaj modul je prikazan kao blok blok-dijagrama sa sl. 5.14. Algoritam sa sl. 5.21 u potpunosti implementira tok podataka iz alokacione tabele sa sl. 5.20.



Slika 5.20. Alokacionia tabela formirana na osnovu linearnog grafikona aktivnosti ulaznih promenljivih za rešenja sistema nejednačina za vremensko usklađivanje



Slika 5.21. Hardverski modul za uvođenje reči ulaznog niza [x_i] u savijenu arhitekturu

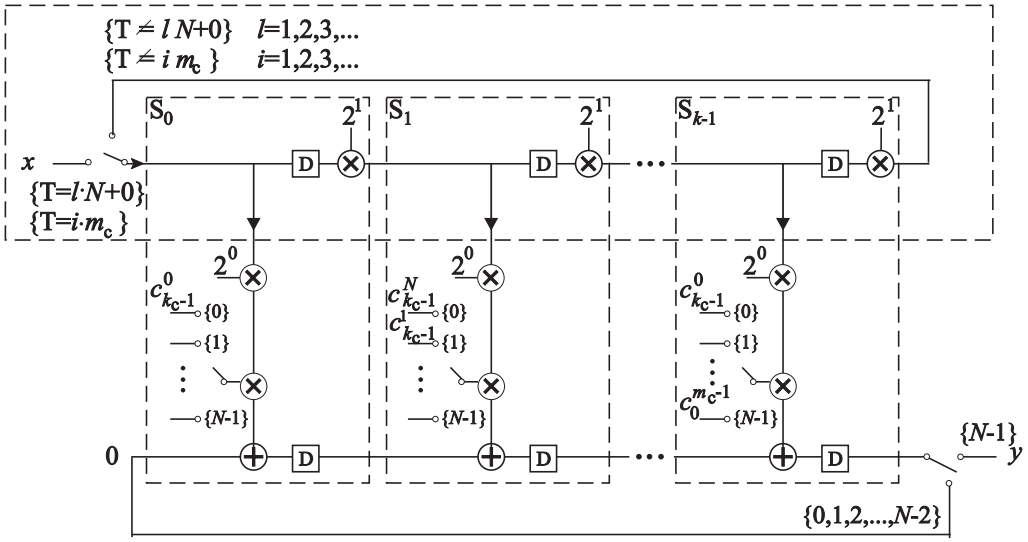
Hardverski modul za uvođenje reči ulaznog niza [x_i] se sastoji od k kola za kašnjenje i jednog prekidača sa dva stanja. Uloga prekidača je da u vremenskim trenucima $N{\cdot}l+\{0\}$ i $i{\cdot}m_C$ (i=1,2,3,...) u arhitekturu uvede ulaznu reč koja se u tom trenutku nalazi na ulaznim linijama. Ulazna reč zatim kruži kroz kola za kašnjenje u skladu sa alokacionom tabelom prikazanom na sl. 5.20.

DFG savijenog TrBP FIR filtra sa promenljivim brojem i dužinom koeficijenata prikazan je na sl. 5.22.

Funkcionalne jedinice savijene arhitekture su izdvojene isprekidanim linijama i označene sa $S_0,S_1,...,S_{k-1}$. Svaka FJ savijene arhitekture izvršava operacije jednog skupa savijanja, odnosno N operacija izvorne arhitekture. Savijena arhitektura se sastoji od ukupno k FJ. Filter vrši izračunavanje sa k_C koeficijenata dužine m_C.

Bitovi koeficijenata se dovode na množače FJ u vremenskim trenucima $N{\cdot}l+\{0\}$, $N{\cdot}l+\{1\}$, …, $N{\cdot}l+\{m\text{-}1\}$ preko prekidača sa N stanja. Ovo je na sl. 5.22 označeno brojevima {0},{1},...,{m-1} uz oznaku odgovarajućeg bita.

Drugi operand množača je ulazni niz [x_i]. Ulazne reči se u možač uvode preko hardvera za uvođenje reči ulaznog niza, koji se nalazi u gornjoj polovini sl. 5.22 i prikazan je DFG-om na sl. 5.21.



Slika 5.22. DFG savijenog TrBP FIR filtra sa izmenjivim brojem i dužinom koeficijenata

Akumulacija parcijalnih proizvoda realizovana je nizom sabirača, kola za kašnjenje i povratnom vezom između poslednje i prve funkcionalne jedinice koja je aktivna u vremenskim trenucima {0,1,2,...,*N*-2}. Izlazni niz [y_i] se dobija na svaka *N* taktna intervala i to u vremenskim trenucima $N \cdot l + \{N-1\}$.

Princip FIR filtriranja na savijenoj arhitekturi je dat u daljem tekstu.

5.3.4. FIR filtriranje na savijenom TrBP semi-sistoličkom polju sa izmenjivim brojem i dužinom koeficijenata

FJ savijene arhitekture, označene sa S_0, S_1, ..., S_{k-1}, izračunavaju po *N* operacija i prikazane su isprekidanim linijama na sl. 5.22. Izračunavanje počinje u FJ S_0, koja u prvom taktnom intervalu daje na svom izlazu parcijalni proizvod $2^0 \cdot c_{k_{C}-1}{}^0 \cdot x_0$. Izračunati parcijalni proizvod se u narednom taktnom intervalu prosleđuje FJ S_1 koja ovom parcijalnom proizvodu dodaje parcijalni proizvod $2^1 \cdot c_{k_{C}-1}{}^1 \cdot x_0$ i narednom skupu prosleđuje međurezultat $(2^0 \cdot c_{k_{C}-1}{}^0 \cdot x_0)+(2^1 \cdot c_{k_{C}-1}{}^1 \cdot x_0)$.

Nakon *k* (*k*<*N*) taktnih intervala, prekidač kola za uvođenje ulaznih reči niza [x_i] se nalazi u gornjem položaju i u FJ S_0 uvodi $2^k \cdot x_0$ iz FJ S_k (sl. 5.22). Na ulaz u sabirač FJ S_0 se u istom taktnom intervalu dovodi među-rezultat $(2^0 \cdot c_{k_{C}-1}{}^0 \cdot x_0)+(2^1 \cdot c_{k_{C}-1}{}^1 \cdot x_0)+...+(2^{k-1} \cdot c_{k_{C}-1}{}^{k-1} \cdot x_0)$, kome ova FJ dodaje parcijalni proizvod $(2^k \cdot c_{k_{C}-1}{}^k \cdot x_0)$. Načinom izbora skupa savijanja (5.7) omogućeno je da "kružno izračunavanje" međurezultata $c_{k_{C}-1} \cdot x_0$ bude završeno za k_C taktnih intervala. Poslednja operacija množenja bitom koeficijenta $c_{k_{C}-1}$ težine 2^{m_C-1} pripada skupu savijanja ($(m_C-1) \bmod k$).

U narednom taktnom intervalu, sledeći po redu skup savijanja započinje množenje ulazne reči $2^0 \cdot x_1$ bitom koeficijenta $c_{k_{C}-2}$ težine 2^0. Ovaj proizvod se akumulira na prethodno akumulirani među-

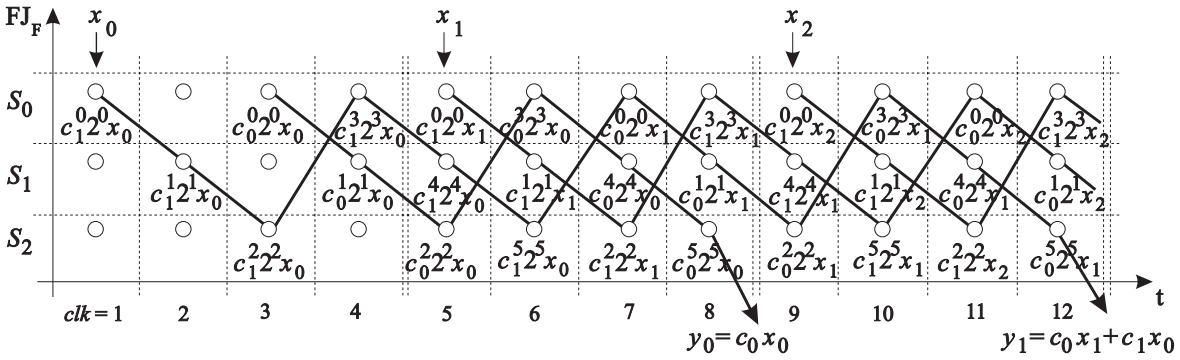
rezultat, tako da u taktnom intervalu t=m_C sabirač funkcionalne jedinice koja izvršava operaciju iz skupa savijanja s=(m_C mod k), računa vrednost sledeće sume:

$$\{(2^0 \cdot c_{k_{C-1}}{}^0 x_0)+(2^1 \cdot c_{k_{C-1}}{}^1 \cdot x_0)+...+(2^{m_{C-1}} \cdot c_{k_{C-1}}{}^{m_{C-1}} \cdot x_0)\}+2^0 \cdot c_{k_{C-2}}{}^0 \cdot x_1 =$$

$$= (c_{k_{C-1}} x_0)+(2^{m_{C-1}} \cdot c_{k_{C-2}}{}^0 \cdot x_1)$$

Tok podataka kroz savijenu arhitekturu sa sl. 5.22 ilustruje opisani algoritam za slučaj k=3, N=4, k_C =2 i m_C=6 (sl. 5.23).

U cilju ilustracije procesa FIR filtriranja na savijenoj arhitekturi, posmatramo izračunavanje prve izlazne reči kod koje su uključeni svi koeficijenti – $y_{k_{C-1}}$. Potrebno je napomenuti da arhitektura simultano vrši izračunavanje izlaznih reči y_0, y_1, ..., $y_{k_{C-1}}$. Prva izlazna reč, y_0, dostupna je na izlaznim linijama arhitekture nakon $2m_C$-N taktnih intervala. Ovo vreme je ujedno i inicijalno kašnjenje arhitekture. Savijeno polje generiše po jednu izlaznu reč na svakih N taktnih intervala.



Slika 5.23. Tok podataka kroz savijenu arhitekturu za k=3, N=4, k_C =2 i m_C=6

Uvođenje bitova koeficijenata u savijenu arhitekturu je problem koji u dosadašnjem postupku sinteze nije razmatran. Bitovi koeficijenata moraju biti uvedeni u arhitekturu u odgovarajućem redosledu, koji zavisi od izabranog broja i dužine koeficijenata. U narednom odeljku prikazana je sinteza hardvera za uvođenje bitova koeficijenata u savijenu arhitekturu.

5.3.5. Sinteza hardvera za uvođenje bitova koeficijenata u savijenu arhitekturu

Blok hardvera za uvođenje bitova koeficijenata u arhitekturu je prikazan na sl. 5.14. Arhitektura sa sl. 5.14 vrši izračunavanje sa promenljivim brojem i dužinom koeficijenata, što je omogućeno načinom dodele skupova savijanja (5.7). Drugim rečima, različite operacije mogu biti izvršavane od strane različitih funkcionalnih jedinica na polju konstantne veličine. Za sintezu hardvera za uvođenje bitova koeficijenata u arhitekturu neophodno je odrediti preslikavanje skupa indeksa bitova koeficijenata na skup FJ. Ovo preslikavanje zavisi, kako od broja, tako i od dužine koeficijenata filtra. Proces sinteze hardvera za uvođenje bitova koeficijenata u savijenu arhitekturu je predstavljen u [29].

5.3.5.1. Preslikavanje skupa indeksa bitova koeficijenata na skup indeksa funkcionalnih jedinica

U cilju nalaženja preslikavanja između skupa indeksa bitova koeficijenata i skupa indeksa FJ savijene arhitekture, koristimo TrDFG sa sl. 5.6 i skupove savijanja date sa (5.7). Pretpostavimo da operacija p, sl. 5.6, kao operand koristi bit težine 2^j koeficijenta c_i. Na osnovu izraza (5.7), operacija p pripada skupu savijanja sa indeksom s=(p–1) mod k, i ima red savijanja r=(p–1) mod N. Drugim rečima, savijena arhitektura koristi c_i^j kao operand u FJ sa indeksom s (0≤s≤k-1) u vremenskim trenucima $N \cdot l$+{r} (0≤r≤N-1 i l=0,1,2,...). Zavisnost indeksa pozicije operacije (p) od ideksa bita koeficijenta (i i j), broja koeficijenata (k_C) i dužine koeficijenata (m_C) na osnovu sl. 5.6 je:

$$p = m_C \left(k_C - \left(i+1 \right) \right) + j + 1 . \qquad (5.18)$$

Zamenom p iz izraza (5.18) u (5.7) dobijamo:

$$\begin{aligned} \mathrm{s} &= \left(m_C \cdot \left(k_C - \left(i+1 \right) \right) + j + 1 \right) \bmod k \\ \mathrm{r} &= \left(m_C \cdot \left(k_C - \left(i+1 \right) \right) + j + 1 \right) \bmod N \end{aligned} . \qquad (5.19)$$

Ukoliko se filtriranje vrši sa k_C koeficijenata dužine m_C na savijenom BP polju dimenzija kxN, pomoću izraza (5.19) je moguće odrediti indeks FJ savijenog sistema (s), koja u trenutku (r) izvršava operaciju nad bitom koeficijenta c_i sa težinom 2^j.

U skladu sa dodeljenim skupovima savijanja (5.7), inverzne zavisnosti i=f(s,r) i j=g(s,r) mogu biti izvedene preslikavanjem pozicije operacije p na elemente matrice A_{kxN}. Kolone matrice A_{kxN} predstavljaju skupove savijanja S_S (0≤s≤k-1), a vrste matrice A_{kxN} predstavljaju redove savijanja operacija r (0≤r≤N-1). Matrica A_{kxN} za slučaj k=3 i m =4 ima sledeći oblik:

$$A_{3\times 4} = \begin{vmatrix} 1 & 5 & 9 \\ 10 & 2 & 6 \\ 7 & 11 & 3 \\ 4 & 8 & 12 \end{vmatrix} . \qquad (5.20)$$

Opšti oblik matrice A_{kxN} je:

$$A_{k\times N} = \begin{vmatrix} 1 & & & N+1 & & \\ & 2 & & & \ddots & \\ & & 3 & & & \\ & & & 4 & & \\ & & & & \ddots & \\ & & & & & k \\ k+1 & & & & & \\ & \ddots & & & \ddots & \\ & & N & & & k \cdot N \end{vmatrix} . \qquad (5.21)$$

Kako bi pojednostavili nalaženje zavisnosti, formiramo matricu A'_{2kxN} izostavljanjem operacija "po modulu" iz matrica (5.20) i (5.21). Za slučaj k=3 i m =4, matrica A'_{2kxN} ima sledeći oblik:

$$A'_{6\times 4}=\left|\begin{array}{ccc|ccc} 1 & 5 & 9 & & & \\ & 2 & 6 & 10 & & \\ & & 3 & 7 & 11 & \\ & & & 4 & 8 & 12 \end{array}\right|. \qquad (5.22)$$

Opšti oblik matrice A'_{2kxN} je:

$$A'_{2k\times N}=\left|\begin{array}{ccccc|ccccc} 1 & N+1 & & & (k-1)\cdot N+1 & & & & & \\ & 2 & N+2 & \ddots & & (k-1)\cdot N+2 & & & & \\ & & 3 & \ddots & 2N+k & & (k-1)\cdot N+3 & & & \\ & & & \ddots & N+k & & & \ddots & & \\ & & & & k & N+k+1 & & & & \\ & & & & & k+1 & N+k+2 & & & \\ & & & & & & k+2 & \ddots & & \\ & & & & & & & \ddots & & \\ & & & & & & & & N & 2N \quad \cdots k\cdot N \end{array}\right|. \qquad (5.23)$$

U matrici A'_{2kxN} se mogu uočiti zavisnost pozicije operacija od dimenzija matrice (k i N) od indeksa funkcionalne jedinice (s) i reda savijanja (r):

$$p=\big((\mathrm{s}+a\cdot k)-\mathrm{r}\big)\cdot N+(\mathrm{r}+1), a=\begin{cases}0, \mathrm{s}\geq \mathrm{r}\\ 1, \mathrm{s}<\mathrm{r}\end{cases}. \qquad (5.24)$$

Izostavljene zavisnosti "po modulu" neophodno je uključiti u dobijeni izraz. Zavisnost indeksa bitova koeficijenata (i i j) od operacije, može biti uočena sa sl. 5.6 i matematički predstavljena sa:

$$\begin{aligned} i &= k_C-1-\left\lfloor\frac{p-1}{m_C}\right\rfloor,\\ j &= (p-1) \mod m_C. \end{aligned} \qquad (5.25)$$

Inverzne zavisnosti $i=f(\mathrm{s},\mathrm{r})$ i $j=g(\mathrm{s},\mathrm{r})$ se dobijaju zamenom pozicije izraza za poziciju p iz izraza (5.24) u izraz (5.25):

$$\begin{aligned} i &= k_C-1-\left(\frac{\big((\mathrm{s}+a\cdot k)-\mathrm{r}\big)\cdot N+\mathrm{r}}{m_C}\right),\ a=\begin{cases}0, \mathrm{s}\geq \mathrm{r}\\ 1, \mathrm{s}<\mathrm{r}\end{cases},\\ j &= \big(\big((\mathrm{s}+a\cdot k)-\mathrm{r}\big)N+\mathrm{r}\big)\bmod m_C, \quad a=\begin{cases}0, \mathrm{s}\geq \mathrm{r}\\ 1, \mathrm{s}<\mathrm{r}\end{cases}. \end{aligned} \qquad (5.26)$$

Na osnovu izraza (5.19) i (5.26) može se zaključiti da je hardver za uvođenje bitova koeficijenata u savijenu arhitekturu potrebno implementirati kao dvodimenzionalno polje dimenzija k i N. Parametri k_C i m_C utiču na raspored bitova koeficijenata u okviru polja.

Izrazi (5.19) i (5.26) opisuju princip rada polja za uvođenje bitova koeficijenata u arhitekturu u procesu inicijalizacije. Na osnovu ovako datog preslikavanja skupa indeksa bitova koeficijenata na skup indeksa FJ je moguće izvršiti sintezu hardvera za uvođenje bitova koeficijenata u savijenu arhitekturu.

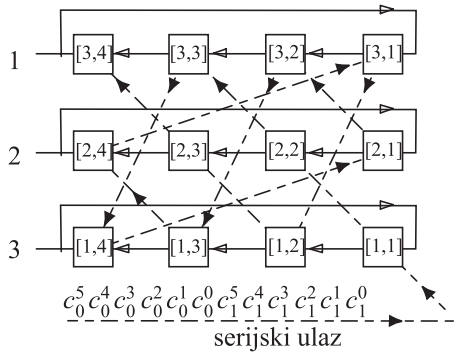
5.3.5.2. Hardverski modul za uvođenje bitova koeficijenata u savijenu arhitekturu

U skladu sa analizom preslikavanja skupa indeksa bitova koeficijenata na skup indeksa FJ, a na osnovu izraza (5.19) i (5.26), dolazi se do zaključka da hardverski modul za uvođenje bitova koeficijenata u savijenu arhitekturu (HUK) ima dva režima rada. Prvi režim rada, je režim rada u kome se polje za uvođenje koeficijenata inicijalizuje, tj. puni bitovima koeficijenata. Drugi režim režim je "uvođenje koeficijenata" u savijenu arhitekturu.

HUK obezbeđuje pravilan redosled uvođenja bitova koeficijenata u savijenu arhitekturu u skladu sa sl. 5.22 i dodeljenim skupovima savijanja (5.7) tj. (5.8). Struktura HUK-a je prikazana na sl. 5.24.

HUK je implementiran kao dvodimenzionalno polje lečeva gde svaki leč pamti po jedan bit koeficijenta (sl. 5.24). Broj vrsta je jednak broju skupova savijanja (k) u savijenoj arhitekturi. Broj kolona je jednak faktoru savijanja N. Izlazne linije HUK (sl. 5.24, veze označene sa 1,2,3) uvode bitove koeficijenata u savijenu arhitekturu. Svaka vrsta polja je zadužena za uvođenje bitova koeficijenata u samo jednu FJ savijene arhitekture. Vrste su implementirane kao pomerački registri, tako da u toku uvođenja bitova koeficijenata u savijenu arhitekturu bitovi koeficijenata rotiraju s desna u levo (sl. 5.24). Ovo je ekvivalentno prekidaču sa m_C stanja sa sl. 5.22, koji nakon poslednjeg položaja, označenog sa $\{N\}$, prelazi ponovo u položaj označen sa $\{0\}$. Ukoliko su bitovi koeficijenata raspoređeni u polje za uvođenje koeficijenata u skladu sa skupovima savijanja (5.8), tada u režimu rada "uvođenje koeficijenata" savijena arhitektura dobija bitove koeficijenata u odgovarajućem redosledu.

Problem inicijalizacije polja može biti rešen korišćenjem izraza (5.19). Staza podataka inicijalizacije HUK, dobijena na osnovu (5.19), prikazana je isprekidanim linijama na sl. 5.24.



Slika 5.24. Hardverski modul za uvođenje bitova koeficijenata u savijenu arhitekturu

Bitovi koeficijenata se uvode u HUK serijski, kroz leč označen sa [1,1] (sl. 5.24), počev od bita najmanje težine koeficijenta c_{k_C-1}. U drugom taktnom intervalu bit $c_{k_C-1}{}^0$ se pomera u leč [2,2] a na njegovo mesto dolazi naredni bit $c_{k_C-1}{}^1$, itd. Kada bit koeficijenata dođe u toku inicijalizacije do poslednje vrste, u narednom taktnom intervalu prelazi u prvu vrstu. Ovo važi i za kolone. Kada bit koeficijenata dođe do poslednje kolone, u narednom taktnom intervalu prelazi u prvu kolonu. Opisani proces implementira zavisnost "po modulu" u izrazu (5.19). Put bita $c_{k_C-1}{}^1$ kroz HUK može biti opisan u skladu sa (5.19):

$$\alpha=((t-1) \bmod k)+1,$$
$$\beta=((t-1) \bmod N)+1. \qquad (5.27)$$

gde su α i β pozicije leča $[\alpha,\beta]$ u kome se u trenutku t (t=1,2,3,...) nalazi bit $c_{k_C-1}^{1}$.

Broj taktnih intervala potrebnih za inicijalizaciju hardvera za uvođenje bitova koeficijenata u savijenu arhitekturu je $k \cdot N$.

Na slici sl. 5.25 prikazan je raspored bitova nakon završetka procesa inicijalizacije HUK za slučaj k=3, N=4, k_C=2 i m_C=6.

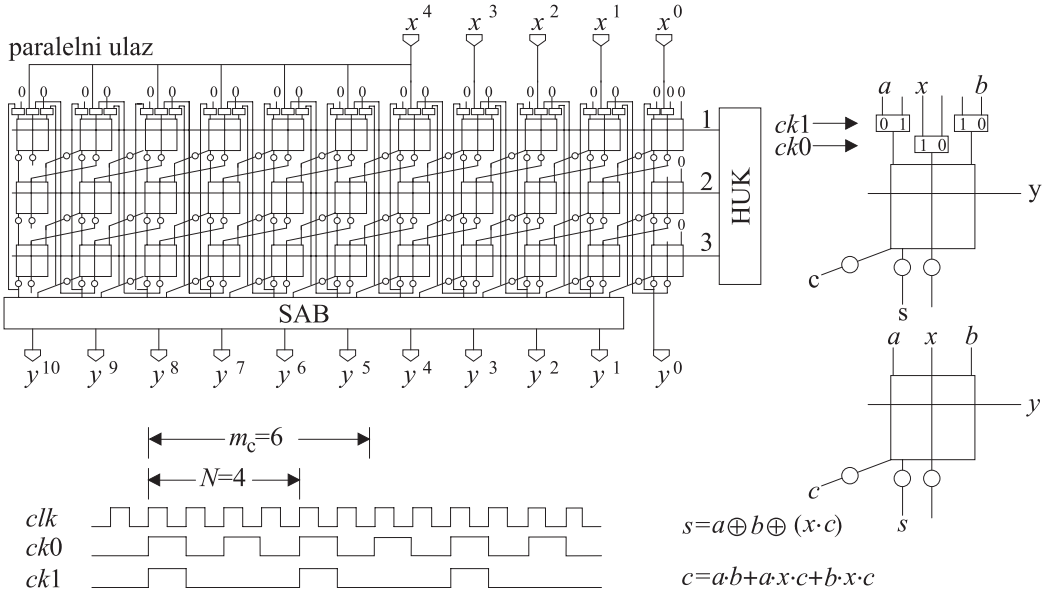
$$\begin{bmatrix} c_1^5 & c_0^2 & c_0^5 & c_1^2 \\ c_1^1 & c_1^4 & c_0^1 & c_0^4 \\ c_0^3 & c_1^0 & c_1^3 & c_0^0 \end{bmatrix}$$

Slika 5.25. Raspored bitova koeficijenata nakon inicijalizacije

Hardverski modul za uvođenje bitova koeficijenata u toku režima inicijalizacije preureduje bitove koeficijenata u skladu sa dodeljenim skupovima savijanja. U toku rada, ovo polje uvodi bitove koeficijenata u savijenu arhitekturu u odgovarajućem redosledu.

5.3.6. Arhitektura savijenog transponovanog "bit-plane" FIR filtra sa promenljivim brojem i dužinom koeficijenata

Na sl. 5.26 je prikazan funkcionalni blok dijagram savijene arhitekture sa promenljivim brojem i dužinom koeficijenata za k=3 i N=4 [30]. Svaka funkcionalna jedinica S_i ($0 \le i \le k$-1) savijene arhitekture sa sl. 5.22 je implementirana jednom vrstom osnovnih ćelija. Između svake dve vrste ćelija sa sl. 5.26 se nalaze lečevi koji razdvajaju FJ na način prikazan na sl. 5.22.



Slika 5.26. Funkcionalni blok dijagram savijene arhitekture sa promenljivim brojem i dužinom koeficijenata (k=3 i N=4)

Množenja faktorom 2^0, od strane hardvera za uvođenje reči ulaznog niza $[x_i]$ u savijenu arhitekturu (sl. 5.21 i sl. 5.22), implemetirana su pomeranjem među-rezultata za jednu poziciju u

levo prilikom prenošenja među-rezultata iz jedne FJ u drugu. Postoje dva tipa osnovnih ćelija. Jedan tip ćelije sadrži multipleksere na ulazu, dok ih drugi tip ćelije ne sadrži (sl. 5.26). Ćelije bez multipleksera se nalaze u svim vrstama izuzev prve. Prvu vrstu polja čine osnovne ćelije sa multiplekserima. Multiplekseri su prisutni u prvoj vrsti polja kako bi vraćanje među-rezultata sa poslednje FJ u prvu bilo moguće (sl. 5.22 i sl. 5.23). Multiplekseri predstavljaju ulaznu logiku koja određuje da li se podatak uvodi sa linija za uvođenje reči ulaznog niza [x_i] ili se u prvu FJ savija među-rezultat iz poslednje. Ovom ulaznom logikom upravlja signal *ck*0 (sl. 5.26). Signal *ck*1 upravlja multiplekserima na taj način da ukoliko je počelo izračunavanje nove reči izlaznog niza [y_i] u arhitekturu uvodi nule, a ako nije vraća među-rezultat iz poslednje FJ.

Hardverski modul za uvođenje bitova koeficijenata u savijenu arhitekturu sa sl. 5.24 se nalazi na desnoj strani polja (sl. 5.26) i označen je sa "HUK". Polje za uvođenje bitova koeficijenata, HUK, taktuje se taktnim signalom *clk* kao i polje osnovnih ćelija savijenog BP filtra.

5.3.7. Upravljanje brojem i dužinom koeficijenata

Kao što je ranije pomenuto, HUK ima dva režima rada. Prvi režim rada, je režim rada polja u kome se HUK inicijalizuje. Drugi režim je režim "uvođenje koeficijenata" u savijenu arhitekturu. HUK ima implementiranu upravljaču liniju za izbor režima rada, koja nije naznačena na sl. 5.24 i sl. 5.26.

Broj bitova koeficijenata koji se uvede u HUK u toku inicijalizacije jednak je $k \cdot N$. Međutim, ovde ne postoji informacija o broju koeficijenata. Odsutna je, takođe, i informacija o dužini unešenih koeficijenata. Ovu informaciju sadrži upravljačka logika, koja na osnovu signala takta *clk* (sl. 5.26) formira signale *ck*0 i *ck*1.

Signal *ck*0, koji upravlja radom multipleksera na ulaznim linijama u polje (sl. 5.26), ima logički nivo "1" na svakih N i m_C intervala taktnog signala *clk* (sl. 5.26). Signal *ck*1, koji takođe upravlja radom multipleksera na ulaznim linijama u polje (sl. 5.26), ima logički nivo "1" na svakih N intervala taktnog signala *clk* (sl. 5.26). Ova dva signala (*ck*0 i *ck*1) daju informaciju o broju koeficijenata kao i o dužini koeficijenata. Na osnovu ovih informacija, signali *ck*0 i *ck*1 upravljaju radom kola za različit broj i dužinu koeficijenata.

Arhitektura sa sl. 5.26 vrši izračunavanje sa promenljivim brojem i dužinom koeficijenata. Proizvod broja i dužine koeficijenata ($k_C \cdot m_C$) fitra sa sl. 5.26 je, na osnovu (5.9), konstantan i jednak je proizvodu broja skupova savijanja (k) i faktora savijanja (N). Smanjenjem dužine koeficijenata (m_C), broj koeficijenata (k_C) mora biti povećan. U većini aplikacija ne postoji potreba za povećanjem broja koeficijenata svaki put kada se dužina koeficijenata smanji, i obrnuto, ne postoji potreba za povećanjem dužine koeficijenata svaki put kada se broj koeficijenata filtra smanji. Ukoliko se broj koeficijenata ne promeni prilikom smanjenja dužine koeficijenata, za izračunavanje je potreban manji broj operacija, pa je, samim tim, i izračunavanje moguće obaviti brže. Arhitektura koja će biti u stanju da izračunavanja sa kraćim koeficijentima od projektovane

dužine koeficijenata obavlja za kraće vreme, mora biti projektovana tako da poseduje mogućnost smanjenja ili broja skupova savijanja (k), ili faktora savijanja (N).

5.3.8. Savijeno polje sa promenljivim brojem i dužinom koeficijenata i mogućnošću smanjenja faktora savijanja

Očigledan način na koji je moguće smanjiti dužinu koeficijenta, i pritom zadržati broj koeficijenata, je zamena bitova najveće težine u koeficijentima filtra nulama. Obrnuto, ukoliko primena zahteva manji broj koeficijenata od broja uslovljenog potrebnom dužina koeficijenata, tada je moguće određene koeficijente zameniti nulama. Inicijalno stanje HUK-a, prikazanog na sl. 5.24, za slučaj k=3, N=4, k_C=2 i m_C=6, kada je dužina koeficijenata smanjena na dužinu m_C^R =3 je:

$$\begin{bmatrix} 0 & c_0^2 & 0 & c_1^2 \\ c_1^1 & 0 & c_0^1 & 0 \\ 0 & c_1^0 & 0 & c_0^0 \end{bmatrix}.$$

U skladu sa redosledom uvođenja bitova koeficijenata u arhitekturu prilikom inicijalizacije HUK (sl. 5.24), za slučaj k=3, N=4, k_C=2, m_C=6 i m_C^R =3, bitovi koeficijenata se uvode u arhitekturu na sledeći način:

$$000c_1^2c_1^1c_1^0\,000c_0^2c_0^1c_0^0\,.$$

Zamenom bitova najveće težine u koeficijentima nulama smanjuje se dužina koeficijenata, međutim, broj operacija koje je potrebno izvršiti na savijenoj arhitekturi ostaje isti, a samim tim i propusnost arhitekture ostaje nepromenjena. U cilju povećanja propusnosti arhitekture na račun broja i/ili dužine koeficijenata, moguće je koristiti rešenje prikazano u poglavlju 5.2.4.

Propusnost arhitekture se povećava smanjenjem broja operacija koje savijeni sistem izvršava (5.9). Broj operacija (L) je jednak proizvodu broja skupova savijanja i faktora savijanja. Smanjivanjem broja skupova savijanja ili faktora savijanja povećava se propusnost arhitekture.

Broj vrsta u savijenom polju sa sl. 5.26 je jednak broju skupova savijanja (k). Broj vrsta je moguće smanjiti, s tim da je za to potrebna relativno složena i prostorno zahtevna upravljačka logika.

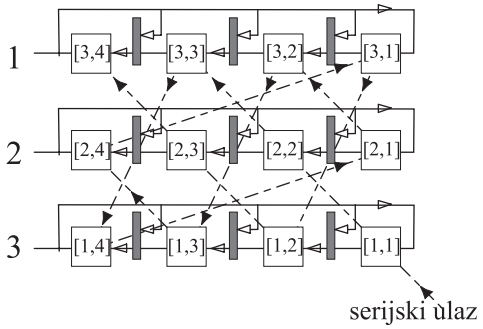
Broj kolona u savijenom polju je jednak $m_C + n + \log_2 k$ i ne zavisi od faktora savijanja. Faktor savijanja (N) utiče na broj kolona u HUK, odnosno, broj kolona u HUK je jednak faktoru savijanja (N). Broj kolona u HUK se može smanjiti korišćenjem pomeračkih registara promenljive dužine, na način prikazan na sl. 5.27.

Korišćenjem HUK sa mogućnošću smanjenja faktora savijanja (sl. 5.27), raspored bitova koeficijenata nakon inicijalizacije, za slučaj k=3, N_{MAX}=4, N=2, k_C=2 i m_C=3, gde je N_{MAX} maksimalni faktor savijanja koji je moguće postići na implementiranom polju, je:

$$\begin{bmatrix} c_1^2 & c_0^2 & 0 & 0 \\ c_0^1 & c_1^1 & 0 & 0 \\ c_1^0 & c_0^0 & 0 & 0 \end{bmatrix}.$$

Za navedeni slučaj, bitovi koeficijenata se za vreme inicijalizacije polja uvode u sledećem redosledu:

$$c_1^2 c_1^1 00 c_0^1 c_0^0 00 c_1^0 c_0^2 00 .$$



Slika 5.27. HUK u savijenu arhitekturu sa mogućnošću smanjenja faktora savijanja, za slučaj k=3 i N=4.

Za algoritam koji uređuje bitove koeficijenata u niz za inicijalizaciju, u slučaju HUK-a sa sl. 5.27, takođe važe jednakosti za određivanje pozicije bita u polju (5.19) i (5.26), s tim da je u polju aktivno samo N kolona od ukupno N_{MAX} kolona. U preostale N_{MAX}-N kolone je potrebno upisati nule. Imajući prethodno rečeno u vidu, formiranje niza bitova koeficijenata na strani aplikacije koja programira filter relativno se jednostavno imlementira. Vreme izračunavanja liearno zavisi od faktora savijanja (N), odnosno dužine koeficijenata (m_C) za konstantan broj koeficijenata (k_C). Propusnost savijenog polja za FIR filtriranje sa promenljivim brojem koeficijenata, dužinom koeficijenata i mogućnošću smanjenja faktora savijanja moguće je povećavti smanjenjem faktora savijanja ($k \cdot N_{MAX}$)/($k_C^R \cdot m_C^R$) puta, gde su k_C^R i m_C^R redukovani broj koeficijenata i dužina koeficijenata, respektivno.

Posledica promeljivosti broja i dužine koeficijenata je da funkcionalne jedinice savijene arhitekture izvršavaju operacije iz različitih skupova savijanja u zavisnosti od broja i dužine koeficijenata sa kojima se vrši filtriranje. Preuređenje bitova koeficijenata se vrši u skladu sa načinom preslikavanja operacija na funkcionlne jedinice u savijenom polju za zadat broj i dužinu koeficiejanta. Funkcionalnost HUK podrazumeva: serijsko uvođenje bitova koeficijenat u arhitekturu (režim inicijalizacije), preuređenje bitova koeficijenata (režim inicijalizacije) i ciklično-paralelno uvođenje bitova koeficijenata u polje (u toku filtriranja). Vreme inicijalizacije je jednako proizvodu broja skupova savijanja i faktora savijanja, i ne zavisi od broja i dužine koeficijenata.

Upravljačka logika koja reguliše promenu broja i dužine koeficijenata je jednostavna i čini je kolo za generisanje periodičnog signala, periode jednake dužini koeficijenata. Ovaj signal se koristi za upravljanje radom hardvera za uvođenje bitova koeficijenata u polje. Dodatna upravljačka logika koja omogućava promenu faktora savijanja ima ulogu da smanji dužinu registara modula za uvođenje bitova koeficijenata u savijeno polje. Ovu logiku čine k·(N-1) multipleksera tipa 2 u 1 i jedan dodatni signal koji upravlja radom multipleksera. Signal koji upravlja radom multipleksera se

dovodi spolja i definiše režima rada polja (inicijalizaciju polja ili radni režim). Kod savijenog BP polja za FIR filtriranje sa promenljivim brojem koeficijenata, dužinom koeficijenata i faktorom savijanja, izračunavanje se može ubrzati na račun smanjenja broja i/ili dužine koeficijenata.

Savijena arhitektura je zadržala sve bitne odlike polazne arhitekture. Površina potrebna za implementaciju je smanjena za nešto manje od N puta u odnosu na izvornu arhitekturu. Smanjenje površine je postignuto na račun vremena. Savijena arhitektura generiše po jednu izlaznu reč na svaka N taktna intervala, bez obzira na dužinu i broj koeficijenata. Parametri arhitekture koji definišu veličinu savijenog polja, odnosno površinu silicijuma koju zauzima filter, su broj skupova savijanja i faktor savijanja. Na savijenom polju je moguće vršiti filtriranje sa promenljivim brojem i dužinom koeficijenata, uz ograničenje da je proizvod broja i dužine koeficijenata, koji određuje broj potrebnih operacija za izračunavanje izlazne reči, jednak proizvodu broja skupova savijanja i faktora savijanja.

U narednom poglavlju su dati rezultati implemntacije savijenih semi-sistoličkih polja za FIR filtriranje. Poglavlje počinje opisom arhitekture FPGA čipova. Nakon opisa argitekture FPGA, dat je prikaz Spartan-II familije FPGA čipova na kojoj je izvršena implementacija savijenih arhitektura. Drugi deo poglavlja je posvećen rezultatima implementacije savojenog polja sa promenljivim faktorom savijanja. U trećem delu su prikazani rezultati implementacije savijenog polja sa promenljivim brojem i dužinom koeficijenata. Na kraju poglavlja je dato poređenje rezultata implementacije izvorne arhitekture i rezultata implementacije savijenih arhitektura..

6. IMPLEMENTACIJA SAVIJENIH ARHITEKTURA

Grupu integrisanih kola specifične namene (ASIC – *Application Specific Integrated Circuits*) čine programabilna logička kola. Za razliku od ostalih tipova ASIC, PLD (*Programmable Logis Devices*) programira sam korisnik. U programabilna logička kola spadaju: programabilne ROM (PROM), programabilne logičke mreže (PLA), programabilna I kola (PAL), registarske PLD i programabilne gejtovske mreže (PGA). Prve tri grupe su kombinacione mreže, dok su zadnje dve sekvencijalne. Programabilne gejtovske mreže imaju veze koje mogu da se programiraju. Za ovu grupu se koristi oznaka FPGA (*Field Programable Gate Array*).

U ovom poglavlju su prikazani rezultati implementacije savijenih semi-sistoličkih polja za FIR filtriranje. Savijena semi-sistolička polja za FIR filtriranje, prikazana u poglavlju 5, implementirana su na *Spartan* II FPGA familiji čipova.

Prvi deo ovog poglavlja je posvećen opisu strukture FPGA čipova. Na početku poglavlja biće prikazana arhitektura i osnovne karakteristike *Spartan* familije FPGA. U drugom delu biće predstavljeni rezultati implementacije savijenog polja za FIR filtriranje sa promenljivim faktorom savijanja. Rezultati implementacije savijenog polja za FIR filtriranje sa izmenjivim brojem i dužinom koeficijenata biće prikazani u trećem delu. Takođe, određena pažnja biće posvećena i analizi brzine rada, propusnosti arhitekture i obimu zauzetih resursa. Na kraju poglavlja biće dato poređenje rezultata implementacije izvorne BP arhitekture i rezultata implementacije projektovanih savijenih polja.

6.1. ARHITEKTURA FPGA

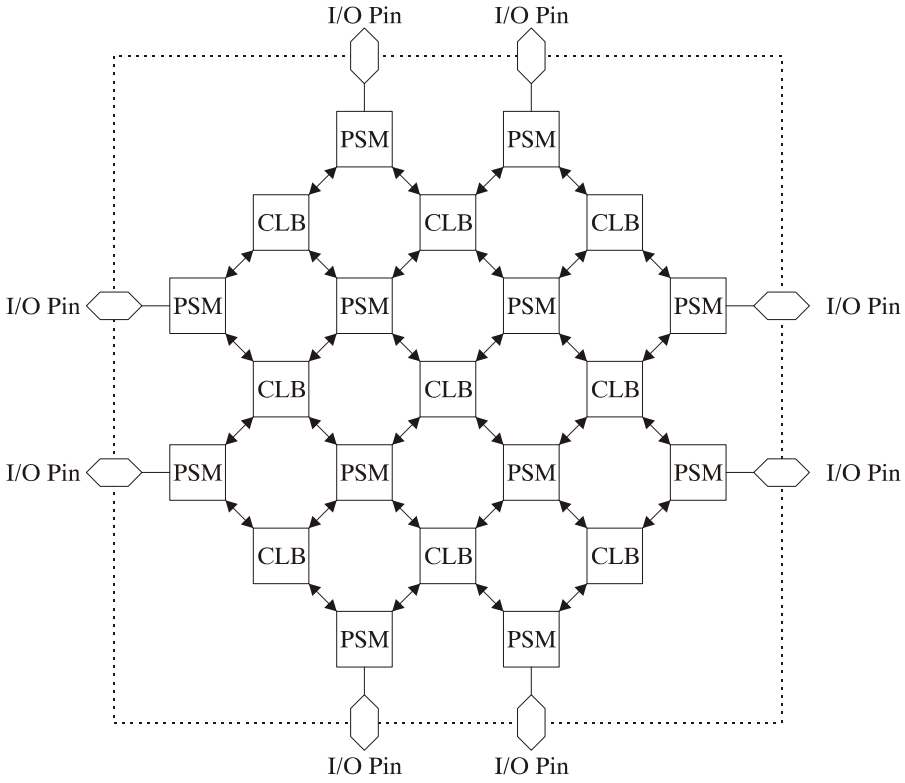
FPGA su čipovi kod kojih je moguće programirati funkcije gradivnih elemenata, kao i veze između njih. Ove čipove karakteriše niska cena projektovanja i znatno brže vreme potrebno za projektovanje. FPGA čipovi su veoma pogodni za digitalnu obradu signala. Rade na većim brzinama od DSP čipova opšte namene, i po kvalitetu odgovaraju DSP čipovima specijalne namene, a uz sve to imaju i nižu cenu. Pored toga, FPGA omogućava da se projekat menja nakon što je proizveden. FPGA je potrebno programirati kako bi se postigla željena funkcionalnost.

Tehnologije koje omogućavaju programiranje FPGA čipova su: *Antifuse* FPGA, *Flash* FPGA i SRAM FPGA. Kod *Antifuze* tehnologije, FPGA čip se programira tako što se određeni skup veza prekida propuštanjem struja koje su veće od nominalne. Ovakav FPGA čip nije moguće reprogramirati. *Flash* FPGA kola je moguće reprogramirati nekoliko hiljada puta. Informacije o vezama unutar kola se čuvaju u *Flash* RAM memoriji. Trenutno dominantna tehnologija je SRAM FPGA. Ova tehnologija dozvoljava neograničen broj reprogramiranja. Vreme potrebno za programiranje SRAM FPGA kola je kraće od vremena programiranja kola koja su implementirana korišćenjem neke od prethodno navedenih tehnologija.

U daljem tekstu je prikazana struktura FPGA čipova.

6.1.1. Struktura FPGA

Opšta struktura FPGA je prikazana na sl. 6.1. Strukturu FPGA čini dvodimenzionalno polje logičkih blokova CLB (*Configurable Logic Block*) povezanih programabilnim međuvezama PSM (*Programmable Switch Matrices*) [31]. Na periferiji čipa se nalaze ulazno/izlazni blokovi IOB (*Input/Output Blocks*). PSM povezuje izlaze i ulaze susednih CLB-ova. FPGA imaju mnogo više CLB-a od I/O pinova, tako da svaki CLB ne može da ima kontakt sa spoljašnjim svetom.



Slika 6.1. Uopštena struktura FPGA čipa

6.1.2. Spartan FPGA

Spartan familija FPGA čipova je serija visokih performansi, visokog kapaciteta, koja obezbeđuje karakteristike CMOS VLSI. Spartan familija FPGA čipova je razvijena 1998. godine. Ova serija je podržana softverom koji pokriva sve faze projektovanja, od unosa šeme kola ili opisa ponašanja sistema, preko planiranja razmeštaja, logičke i funkcionalne simulacije, automatskog razmeštaja blokova i trasiranja veza, do učitavanja konfiguracionog niza bitova na čip, unošenje podataka na čip i čitanje sa njega [31].

Spartan familija FPGA kola ima visoke performanse, veliki broj logičkih kola i veliki skup dodataka. Dodaci su blok RAM (do 56K bita), distribuirani RAM (do 75,264 bita), podrška za 16 U/I standarda i četiri petlje kašnjenja takta (tab. 6.1). Gustina pakovanja se kreće od 15,000 gejtova za najmanji čip, do 200,000 sistemskih gejtova za čip najveće gustine. Gustine pakovanja pojedinih čipova iz Spartan-II familije FPGA su prikazane u tab. 6.1.

Tabela 6.1. Spartan-II familija čipova

Oznaka čipa	Broj logičkih ćelija	Broj gejtova	Dimenzije CLB Matrice	CLB	IOB	Kapacitet distribuiranog RAM-a [b]	Blok RAM [b]
XC2S15	432	15,000	8 x 12	96	86	6,144	16K
XC2S30	972	30,000	12 x 18	216	132	13,824	24K
XC2S50	1,728	50,000	16 x 24	384	176	24,576	32K
XC2S100	2,700	100,000	20 x 30	600	196	38,400	40K
XC2S150	3,888	150,000	24 x 36	864	260	55,296	48K
XC2S200	5,292	200,000	28 x 42	1,176	284	75,264	56K

Postoje tri grupe Spartan čipova: Spartan, Spartan-II i Spartan-XL. Spartan čipovi su zasnovani na Virtex arhitekturi.

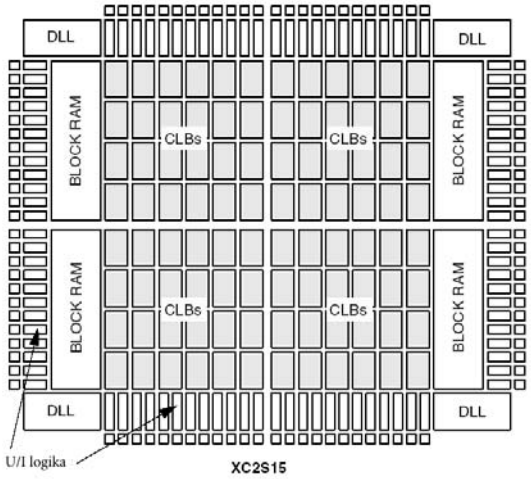
Karakteristike Spartan-II familije FPGA su sledeće [31]:

- Tehnologija druge generacije
  - gustine pakovanja do 5,292 logičkih ćelija sačinjenih od 200,000 sistemskih gejtova;
  - protočne osobine bazirane na Virtex arhitekturi;
  - neograničen broj reprogramiranja;
- Osobine sistemskog nivoa
  - hijerarhijski organizovana memorija;
  - 16-to bitni LUT distribuirani RAM;
  - 4Kb blok RAM;
  - brzi interfejs ka spoljnoj memoriji;
  - arhitektura sa segmentiranim rutiranjem, implementirana u cilju niže potrošnje;
  - logika prenosa za brza izračunavanja;
  - ugrađena podrška za množenje;
  - kaskadni lanci za ulazne podatke;
  - četiri petlje za dodatno upravljanje taktom;
  - IEEE 1149.1 kompatibilna logika sa skeniranje ivica signala.

- Karakteristike U/I interfejsa
  - 16 U/I standarda;
  - PCI kompatibilnost;
  - automatsko preslikavanje i rutiranje.

6.1.2.2. Arhitektura Spartan-II FPGA čipova

Čipovi iz Spartan-II familije imaju regularnu, fleksibilnu, programabilnu strukturu konfigurabilnih logičkih blokova (CLB), okruženih programabilnim ulazno-izlaznim blokovima (IOB). Na svakom uglu čipa (sl. 6.2) se nalazi po jedna petlja kašnjenja takta (*Delay-Locked Loop* - DLL). Dve kolone blok RAM-a se nalaze na različitim stranama čipa, između CLB-ova i IOB kolona. Ovi funkcionalni elementi su povezani hijerarhijski organizovanim kanalima za rutiranje (sl. 6.2) [31].



Slika 6.2. Blok dijagram osnovnih gradivnih blokova Spartan-II čipova

Konfiguracija čipova iz Spartan-II familije se vrši tako što se konfiguracioni podaci učitaju u ćelije interne statičke memorije. Ovakvim pristupom omogućen je neograničen broj reprogramiranja čipova. Konfiguracioni podaci, smešteni u ove ćelije, određuju logičke funkcije i veze koje su implementirane na FPGA. Konfiguracioni podaci mogu biti čitani i direktno sa nekog spoljašnjeg medijuma sa serijskim pristupom (npr. PROM).

6.1.2.3. Funkcionalne jedinice Spartan-II FPGA

Spartan-II familija FPGA čipova je implementirana kao regularna, fleksibilna i programabilna arhitektura sačinjena od polja CLB-ova, okruženih mnoštvom programabilnih U/I blokova. U arhitekturu Spartan II FPGA čipa su uključeni i dodatni resursi kao što su blok RAM i blokovi za upravljanje taktom [31].

Polje Spartan-II čipova (sl. 6.2) se sastoji od pet osnovnih elemenata:

1. CLB (*Configurable Logic Block*) – konfigurabilni logički blok,
2. Blok RAM – interna memorija FPGA čipa,
3. IOB – ulazno/izlazni interfejs između pinova i unutrašnje logike kola,

4. DLL (*Delay Locked Loop*) - petlje kašnjenja za distribucija taktnog signala,

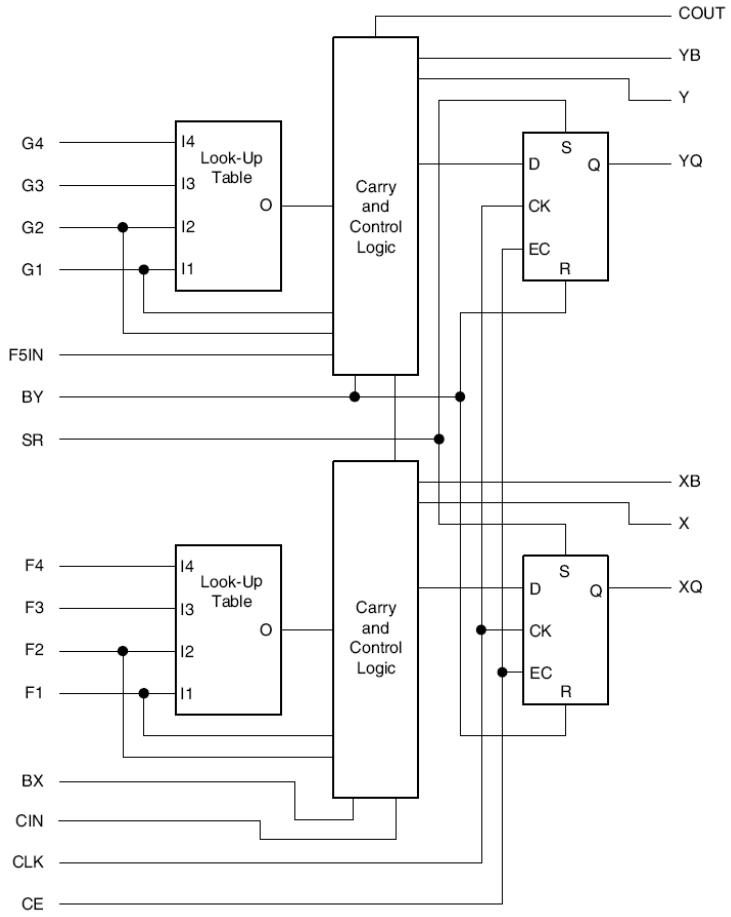
5. Logika za rutranje i veze.

Sa sl. 6.2 se vidi da CLB-ovi imaju pristup svim dodacima i strukturi za rutiranje. U cilju lakšeg i bržeg prenosa signala u čip kao i lakšeg i bržeg prenosa signala iz čipa, IOB su locirani oko CLB-ova i memorijskih elemenata.

Konfiguracioni podaci, koji se nalaze u ćelijama upravljačke memorije, upravljaju radom svih funkcionalnih celina kao i vezama između njih. Ovi podaci se učitavaju u ćelije u trenutku uključivanja napajanja.

Struktura CLB-a Spartan II čipova

Osnovni gradivni blok Spartan-II FPGA kola logička ćelija (LC). LC uključuje četvoro-ulazni generator funkcija, carry logiku i memorijske elemente. Svaki CLB Spartan-II čipa se sastoji od četiri logičkih ćelija, smeštenih u dve identične celine (eng. *Slice* - SL). Na sl. 6.3 je prikazana blok šema SL Spartan-II FPGA čipa. Dodatak logičkim ćelijama je mreža koja kombinuje generatore funkcija čime se dobija generator funkcija sa više ulaza.



Slika 6.3. Blok šema SL Spartan-II čipa

Look-up tabele – Generatori funkcija Spartan-II FPGA čipa (sl. 6.3) su iplementirani kao četvoro-ulazne look-up tabele (LUT). Pored toga što LUT ima ulogu generatora funkcija, svaki

LUT može da bude konfigurisan tako da radi kao 16x1-bitni sinhroni RAM. Dalje, dva LUT-a u okviru jednog SL mogu da grade 16x2-bitni, 32x1-bitni sihroni RAM ili 16x1-bitni sinhroni RAM sa dva porta. Spartan-II LUT, takođe, može da radi kao 16-bitni pomerački registar, koji je idealan za akviziciju podataka pri velikim brzinama.

Memorijski elementi – Memorijski elementi u Spartan-II CLB-u mogu biti konfigurisani tako da menjaju stanje, ili na promenu nivoa signala, ili na sam nivo signala. Ulazi D flip-flopova se pobuđuju od strane generatora funkcija u okviru SL, ili direktno sa ulaza SL, zaobilazeći generatore funkcija (sl. 6.3).

Dodatna logika – F5 multiplekser kombinuje izlaze iz generatora funkcija (sl. 6.3). Na ovaj način je moguće implementirati generator funkcija sa maksimalno pet ualza, 4 u 1 multiplekser ili funkciju koja ima do devet ulaza. Slično, F6 multiplekser kombinuje izlaze sva četiri generatora funkcija u CLB-u, birajući jedan od izlaza F5-multipleksera. Ovo omogućava implemenatciju bilo koje funkcije sa šest ulaza, 8 u 1 multiplekser ili prekidačku funkciju sa maksimalno 19 ulaza. Svaki CLB poseduje četiri direktne linije za uvođenje podataka, tj. jednu po logičkoj ćeliji. Ove linije predstavljaju dodatne ulazne linije u CLB, ili dodatne lokalne puteve za rutiranje signala, pri čemu se ne zauzimaju logički resursi.

Aritmetika – Carry logika (sl. 6.3) omogućava implementaciju brzih aritmetičkih funkcija. CLB Spartan-II čipova podržava dva nezavisna lanca prenosa (*carry chain*). Dužina lanca prenosa je dva bita po CLB-u. Podrška za aritmetičke operacije podrazumeva XOR kolo koje se koristi za implementaciju jednobitnog potpunog sabirača po logičkoj ćeliji. Dodatno AND kolo povećava efikasnost pri implementaciji množača. Lanac prenosa, u obliku kaskadnog lanca generatora funkcija, može biti korišćen za implementaciju logičkih funkcija sa velikim brojem ulaza.

BUFT – Svaki CLB Spartan-II čipa poseduje dva bafera sa tri stanja kao interfejs ka unutrašnjim magistralama.

Blok RAM

Koncept blok RAM memorije dopunjuje relativno mali kapacitet distribuirane memorije, koja je implementirana u LU tabelama CLB elemenata. Blokovi RAM memorije su organizovani kao kolone. Svaki čip iz Spartan-II familije sadrži po dve kolone blok RAM-a (jedna uz svaku vertikalnu ivicu čipa). Kolone se protežu celom dužinom čipa. Visina jednog memorijskog bloka je jednaka zbiru visina četiri CLB-a. Tako, Spartan-II čip sa osam CLB vrsta, ima implementirana dva bloka uz svaku vertikalnu ivicu, tj. ukupno četiri memorijska bloka. U tab. 6.2 prikazana je količina raspoložive memorije za pojedine čipove iz Spartan-II familije.

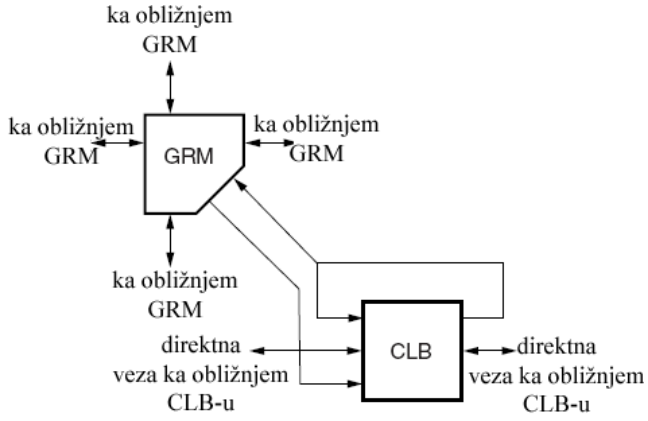
Tabela 6.2. Kapacitet raspoloživog blok RAM-a za Spartan-II

Spartan-II čip	Broj blokova	Blok RAM
XC2S15	4	16K
XC2S30	6	24K
XC2S50	8	32K
XC2S100	10	40K
XC2S150	12	48K
XC2S200	14	56K

Rutiranje signala uokviru čipa

Frekvencu taktnog signala, tj. maksimalnu brzinu rada imlementiranog kola, definiše najduži put između dva memorijska elementa u kolu. Proces za optimizaciju rutiranja, ugrađen u softver za projektovanje kola, ima za cilj da minimizira kašnjenje najdužeg puta signala u kolu. Mreža za rutiranje signala u okviru Spartan II FPGA čipa je implementirana u dva nivoa. Rutiranje signala između fizički bliskih CLB-ova se obavlja preko lokalne mreže za rutiranje, dok se rutiranje između udaljenih CLB-ova obavlja preko globalne mreže za rutiranje.

Lokalna mreža za rutiranje signala – Resursi za lokalno rutiranje, prikazani na sl. 6.4, dozvoljavaju tri tipa veza.



Slika 6.4. Lokalno rutiranje signala

Prvi tip veza su veze između LU tabela, flip-flopova i matrice za rutiranje signala (eng. General Routing Matrix - GRM). Drugi tip veza su povratne, brze veze u okviru CLB-a sa minimalnim kašnjenjem pri rutiranju. Treći tip su direktne veze između horizontalno bliskih CLB-ova. Ove veze imaju ulogu da eliminišu kašnjenja GRM-a.

Globalna mreža za rutiranje signala – Većina signala u okviru Spartan-II čipa se rutira preko globalne mreže za rutiranje signala. Drugim rečima, većina resursa za rutiranje signala pripada ovom nivou hijerarhije. Resursi globalne mreže za rutiranje su implementirani kao horizontalni i vertikalni kanali, vezani za vrste, odnosno kolone CLB-ova. Resursi globalne mreže za rutiranje su: globalna matrica za rutiranje, 24 linija za rutiranje GRM signala do obližnjih GRM-a u sva četiri pravca, 96 linija sa baferima za prenos signala na rastojanje koje nije veće od 6 CLB-ova, 12 dugih linija sa beferima.

Arhitektura FPGA čipova je regularna, fleksibilna i programabilna, što je čini idealnom za implementcaiju sistoličkih i semi-sistoličkih polja. Savijena TrBP semi-sistolička polja su opisana u VHDL-u. Programsko okruženje koje je korišćeno za opis i simulaciju savijenih arhitektura je *Xilinx* ISE *WebPack* 4.2i. Za implemetaciju savijenih polja prikazanih u poglavlju 5 korišćen je Spartan II FPGA čip sa oznakom XC2S200.

6.2. IMPLEMENTACIJA SAVIJENOG SEMI-SISTOLIČKOG POLJA ZA FIR FILTRIRANJE SA PROMENLJIVIM FAKTOROM SAVIJANJA

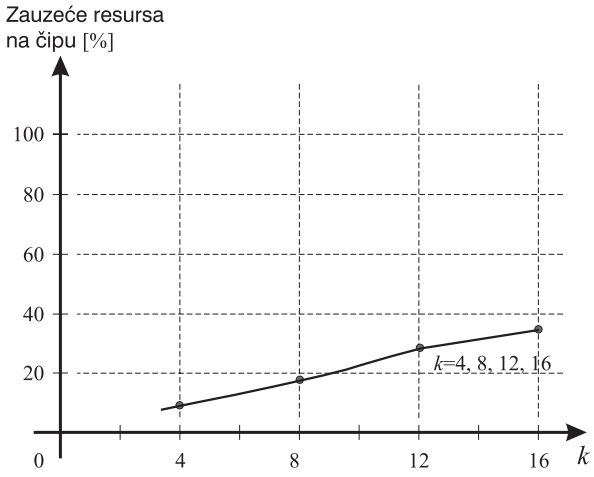
Arhitektura savijenog polja za FIR filtriranje sa promenljivim faktorom savijanja prikazana je na sl. 5.12. U cilju provere funkcionalnosti kola i ilustracije kompromisa u pogledu vremena izračunavanja i zauzeća resursa na čipu, savijena arhitektura sa sl. 5.12 je opisana u VHDL-u i implementirana na Spartan II s200-5pq208 FPGA čipu. Rezultati implementacije prikazani su u tab. 6.3.

Tabela 6.3. Rezultati implementacije savijenog BP FIR filtra sa promenljivim faktorom savijanja

k	*N*	*y*	Zauzeće čipa [%]	MPPD [ns]	MID [ns]
4	8	8	9.4	4.9	4.0
8	8	8	18.0	5.3	4.5
12	8	8	27.7	5.6	5.0
16	8	8	36.6	7.3	6.0

Tabela daje prikaz zauzeća resursa čipa, maksimalne vrednosti za kašnjenja između pinova (*Maximal values for Pin-to-Pin Delay – MPPD*) i maksimalne vrednosti za kašnjenje u okviru kola (*Maximal values ofr Internal Delay – MID*). Arhitektura je implementirana za različite vrednosti broja skupova savijanja (k=4, 8, 12 i 16), tj. implementirani su filtri sa različitim brojem tapova. Kod svih implementiranih arhitektura širina ulaznih reči je n=8, a faktor savijanja N, koji je jednak maksimalnoj dužini koeficijenata (m_1), je N=8.

Grafička reprezentacija zauzeća resursa na čipu u funkciji broja koeficijanata filtra (k), ilustrovana na osnovu podataka iz tab. 6.3, je prikazana na sl. 6.5. Broj koeficijenata filtra (k) je jednak broju skupova savijanja filtra. Kako je broj skupova savijanja jednak broju vrsta savijenog polja (sl. 5.12), projektovane arhitekture sa većim brojem koeficijenata zauzimaju više resursa čipa (sl. 6.5).



Slika 6.5. Zauzeće resursa na čipu u funkciji broja skupova savijanja

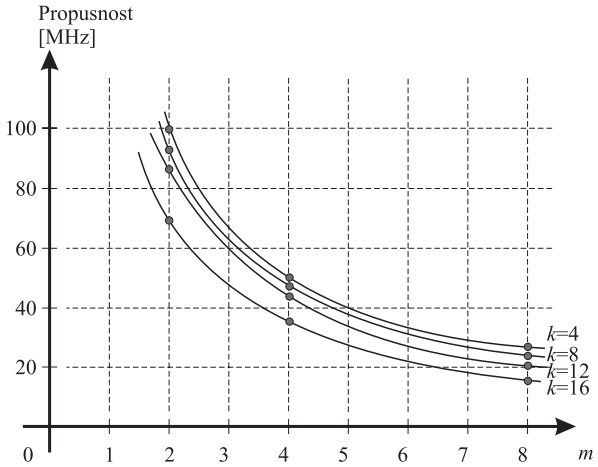
Propusnost implementirane arhitekture je data u tab. 6.4. Sa f_{CLK} je označena taktna učestalost kola, dok f_{TH} predstavlja učestalost dobijanja rezultata na izlaznim linijama filtra. Kako je kod svih

implementiranih arhitektura faktor savijanja N=8, maksimalna dužina koeficijenata je m_1=8. To znači da je na ovim arhitekturama moguće vršiti filtriranje koeficijenatima dužine m=2, 4 ili 8 bitova (tab. 6.4). Smanjenjem dužine koeficijenata (m) povećava se propusnost filtra (tab. 6.4). Naravno, taktna učestalost kola ostaje nepromenjena bez obzira na promenu dužine koefijenata.

Tabela 6.4. Rezultati implementacije savijenog BP FIR filtra sa promenljivim faktorom savijanja

k	*n*	*m*	*y*	f_{CLK} [MHz]	f_{TH} [MHz]
4	8	8	8	204.1	25.5
		4		204.1	50.1
		2		204.1	102.0
8	8	8	8	188.7	23.6
		4		188.7	47.2
		2		188.7	94.4
12	8	8	8	178.6	22.3
		4		178.6	44.6
		2		178.6	89.2
16	8	8	8	137.0	17.1
		4		137.0	34.2
		2		137.0	68.4

Na sl. 6.6 je prikazana promena propusnosti filtra u zavisnosti od dužine koeficijenata za sve implementirane arhitekture.



Slika 6.6. Propusnost savijenog polja u funkciji dužine koeficijenata

6.3. IMPLEMENTACIJA SAVIJENOG SEMI-SISTOLIČKOG POLJA ZA FIR FILTRIRANJE SA IZMENJIVIM BROJEM I DUŽINOM KOEFICIJENATA

Savijeno polje za FIR filtriranje sa izmenjivim brojem koeficijenata, dužinom koeficijenata i promenljivim faktorom savijanja (sl. 5.26 i sl. 5.27) je opisano u VHDL-u. Polje je implementirano na Spartan II s200-5pq208 FPGA čipu, u cilju ilustracije načina filtriranja sa mogućnostima promene parametara u toku rada filtra. Takođe, implementacijom su dobijeni podaci o zauzetosti resursa na čipu za različite vrednosti broja skupova savijanja i faktora savijanja.

Rezultati implementacije za različite vrednosti parametara k i N su prikazani u tab. 6.5. Svaka vrsta u tab. 6.5 sadrži rezultate implementacije jedne arhitekture. Tabela sadrži sledeće parametre implementacije:

n – širina ulazne reči x_i,

k – broj skupova savijanja implemetirane savijene arhitekture,

N_{MAX} – maksimalni faktor savijanja koji se može postići na polju,

y – širina izlazne reči y.

Zauzeće resursa – broj zauzetih SL na čipu, kao i procenat zauzeća čipa,

Širina taktnog intervala – širina taktnog intrvala kola prikazana u ns,

Radna frekvenca kola – frekvenca na kojoj radi kolo, prikazana u MHz.

Zauzeće resursa na čipu u funkciji faktora savijanja za koji je savijeno polje implementirano, na osnovu podataka iz tab. 6.5, grafički je ilustrovano na sl. 6.7 i sl. 6.8. Funkcija promene zauzeća resursa na čipu u zavisnosti od faktora savijanja za koji je savijeno polje projektovano (N_{MAX}), pri konstantnom broju skupova savijanja (k), je prikazana na sl. 6.7. Zavisnost broja zauzetih SL čipa od faktora savijanja (N_{MAX}) je linearna.

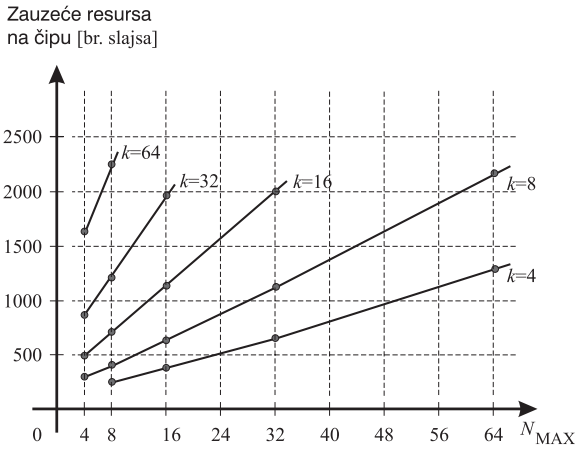
Sa povećanjem faktora savijanja (N_{MAX}), povećava se širina registara polja za uvođenje koeficijenata u arhitekturu, potrebna je veća širina BP polja, a takođe se povećava i širina sabirača (sl. 5.26). Potrebno je napomenuti da arhitektura sa sl. 5.26 ne vrši odsecanje rezultata u cilju smanjenja polja.

Tabela 6.5. Rezultati implementacije savijenih polja sa promenljivim brojem koeficijenata, dužinom koeficijenata i promenljivim faktorom savijanja

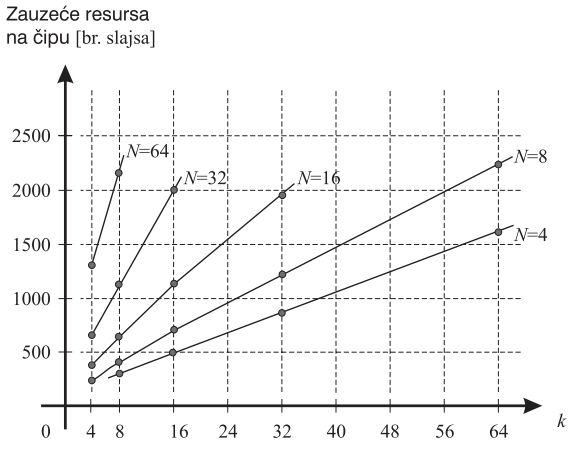
n	k	N_{max}	y	Zauzeće resursa		Takt [ns]	Frekvenca [MHz]
				Br.SL	%		
8	4	8	18	243	10	4.771	209.60
8	4	16	26	377	16	5.429	184.20
8	4	32	42	664	28	5.425	184.33
8	4	64	74	1312	56	6.157	162.42
8	8	4	15	297	13	5.119	195.35
8	8	8	19	412	18	5.147	194.29
8	**8**	**16**	**27**	**642**	**27**	**5.090**	**196.46**
8	8	32	43	1126	48	5.180	193.05
8	8	64	75	2165	92	5.196	192.46
8	16	4	16	498	21	5.463	183.05
8	16	8	20	713	30	5.288	189.11
8	16	16	28	1137	48	5.398	185.25
8	16	32	44	2009	85	4.815	207.68
8	16	64	76	/	/	/	/
8	32	4	17	870	37	4.905	203.87
8	32	8	21	1219	52	4.900	204.08
8	32	16	29	1964	84	5.872	170.30
8	32	32	45	/	/	/	/
8	32	64	77	/	/	/	/
8	64	4	18	1628	69	4.884	204.75
8	64	8	22	2240	95	5.173	193.31
8	64	16	30	/	/	/	/

Funkcija promene zauzeća resursa na čipu u zavisnosti od broja skupova savijanja (k), pri konstantnom faktoru savijanja (N_{MAX}), je prikazana na sl. 6.8. Na sl. 5.26 je prikazano 5 funkcija

promene zauzeća resursa za različite vrednosti faktora savijanja (N_{MAX}). Sa slike se može uočiti da je i ova zavisnost linearna. Promenom broja skupova savijanja (k) povećava se broj vrsta u savijenom BP polju, kao i u polju za uvođenje koeficijenata, što linearno utiče na zauzeće prostora na čipu.



Slika 6.7. Zauzeće resursa na čipu u funkciji faktora savijanja N_{MAX}



Slika 6.8. Zauzeće resursa na čipu u funkciji broja skupova savijanja k

Prethodna diskusija se odnosi na zavisnost zauzetosti resursa na čipu od parametara vezanih za implementaciju filtra. Međutim, jedna od najvažnijih osobina savijenog polja je mogućnost filtriranja sa različitim brojem koeficijenata, dužinom koeficijenata i faktorom savijanja na polju konstantne veličine, promenom navedenih parametara filtriranja u toku rada filtra. Propusnost jedne od implementiranih arhitektura (n=8, k=8, N_{MAX}=16 i y=27 – osenčana kolona u tab. 6.5) za različite vrednosti broja koeficijenata (k_C) i dužine koeficijenata (m_C) je prikazana u tab. 6.6.

Tabela 6.6. Prikaz povećanja propusnosti arhitekture u funkciji faktora savijanja

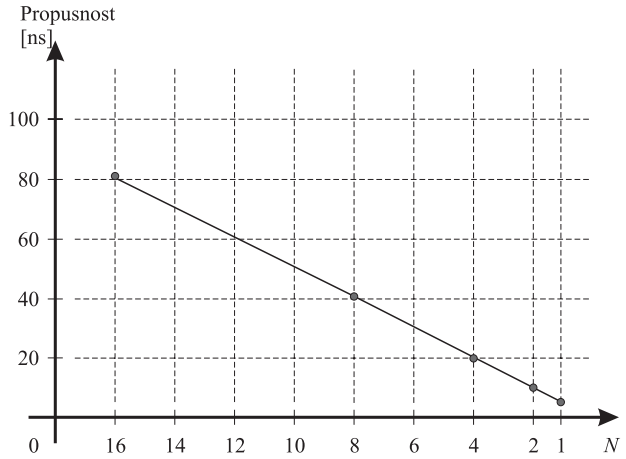
k	N	k_C	m_C	Takt		Propusnost			Inicijalno kašnjenje - T_{INC}	
				[ns]	[MHz]	[clk]	[ns]	[MHz]	[clk]	[ns]
8	16	1	128	5.09	86.69	16	81.44	12.28	267	1359.03
		2	64	5.09	86.69	16	81.44	12.28	139	707.51
		4	32	5.09	86.69	16	81.44	12.28	75	381.75
		8	16	5.09	86.69	16	81.44	12.28	43	218.87
8	8	1	64	5.09	86.69	8	40.72	24.56	147	748.23
		2	32	5.09	86.69	8	40.72	24.56	83	422.47
		4	16	5.09	86.69	8	40.72	24.56	51	259.59
8	4	1	32	5.09	86.69	4	20.36	49.12	87	442.83
		2	16	5.09	86.69	4	20.36	49.12	55	279.95
		4	8	5.09	86.69	4	20.36	49.12	39	198.51
8	2	1	16	5.09	86.69	2	10.18	98.23	57	290.13
		2	8	5.09	86.69	2	10.18	98.23	41	208.69
8	1	1	8	5.09	86.69	1	5.09	196.46	42	213.78
		2	4	5.09	86.69	1	5.09	196.46	34	173.06

Širina taktnog intervala, taktna učestalost, propusnost i inicijalno kašnjenje su dati za različite vrednosti broja i dužine koeficijenata, vodeći računa o smanjenju faktora savijanja (N) na minimalnu moguću vrednost (5.9).

Za inicijalno kašnjenje T_{INC} prikazano u tab. 6.6 važi jednakost $T_{INC}=2m_C-N+T_{SAB}$, gde je T_{SAB} inicijalno kašnjenje sabirača sa sl. 5.26. Inicijalno kašnjenje sabirača u implementiranoj arhitekturi

je jednako širini sabirača (T_{SAB}=y=27). Korišćenjem specifičnih osobina *Spartan* II FPGA arhitekture, kod konkretne implementacije je postignuto da sabirač, za širine izlaznih reči datih u tab. 6.5, vrlo malo utiče na zauzeće resursa na čipu u odnosu na polje.

Na osnovu podataka iz tab. 6.6, na sl. 6.9 je grafički ilustrovana zavisnost propusnosti polja u funkciji izabranog faktora savijanja. Smanjenjem faktora savijanja smanjuje se vreme koje protekne od trenutka kada je izlazna reč y_i dostupna na izlaznim linijama do trenutka kada je izlaznim linijama dostupna izlazna reč y_{i+1}, tj. povećava se propusnost polja (sl. 6.9).



Slika 6.9. Propusnost polja u funkciji izabranog faktora savijanja

6.4. POREĐENJE REZULTATA IMPLEMENTACIJE IZVORNE "BIT-PLANE" ARHITEKTURE I DOBIJENIH SAVIJENIH ARHITEKTURA

U cilju poređenja karakteristika izvorne BP arhitekture i savijenih arhitektura, izvorna BP arhitektura je implementirana na *Spartan* II s200-5pq208 FPGA čipu, odnosno na istom čipu na kome su implementirane i savijene arhitekture. Rezultati implementacije su prikazani u tab. 6.7. U tabeli su uporedo prikazani rezultati implementacije izvorne arhitekture i savijenih arhitektura u pogledu resursa koji su zauzeti na čipu, maksimalnog kašnjenja između spoljnih priključaka, maksimalnog kašnjenja unutar kola, radne učestalosti i propusnosti kola. Uz svaki rezultat dobijen implementacijom izvorne arhitekture nalaze se odgovarajući rezultati dobijeni implementacijom savijenih arhitektura. Uporedno su prikazani rezultati implementacije izvorne BP arhitekture sa k=4,8,12 i 16 koeficijenata dužine m=8 i širinom ulazne reči n=8 sa rezultatima implementacije savijenih arhitektura sa k=4,8,12 i 16 skupova savijanja, faktorom savijanja N=8 i širinom ulaznih reči n=8. Oznakom A1 u tabeli su predstavljeni rezultati implementacije savijenog semi-sistoličkog polja za FIR filtriranje sa promenljivim faktorom savijanja, dok su oznakom A2 predstavljeni rezultati implementacije savijenog semi-sistoličkog polja za FIR filtriranje sa promenljivim brojem i dužinom koeficijenata.

Što se zauzeća resursa na čipu tiče, dobijeni rezultati su u granicama očekivanih. Kao što je ranije rečeno, površina koju zauzimaju savijena polja je smanjena za nešto manje od N puta u odnosu na izvornu arhitekturu. Odstupanja nastaju jer određenu površinu zauzima i upravljačka logika. Iz rezultata prikazanih u tab. 6.7 se vidi da u većini slučajeva arhitektura označena sa A2 zauzima manje resursa od arhitekture označene sa A1. Razlog za ovo leži u mnogo manjem broju multipleksera koji su pridruženi osnovnim ćelijama polja A2 u odnosu na ćelije polja A1. Arhitektura A2 ima multipleksere samo u prvoj vrsti polja (poglavlje 5.3.6).

Tabela 6.7. Poređenje rezultata implementacije izvorne BP arhitekture i savijenih arhitektura za različite vrednosti broja skupova savijanja (k), pri čemu je faktor savijanja N=8, a širina ulaznih reči n=8.

k		**4**	**8**	**12**	**16**
Zauzeće resursa na čipu [%]	BP	28.6	59.5	98.9	99.9
	A1	9.4	18.0	27.7	36.6
	A2	10.0	18.0	24.7	30
Max. kašnjenje između spolj. priključaka [ns]	BP	6.5	4.9	6.0	6.5
	A1	4.9	5.3	5.6	7.3
	A2	4.7	5.1	5.0	5.2
Max. kašnjenje unutar kola [ns]	BP	3.1	3.6	4.6	5.8
	A1	4.0	4.5	5.0	6.0
	A2	3.9	4.1	5.0	5.4
Radna učestalost [MHz]	BP	153.8	204.1	166.7	153.8
	A1	204.1	188.7	178.6	137.0
	A2	212.8	196.1	200.0	192.3
Propusnost kola [MHz]	BP	153.8	204.1	166.7	153.8
	A1	25.5	23.6	22.3	17.1
	A2	26.6	24.5	25.0	24.0

U tab. 6.7 su, pored zauzeća rasursa čipa, prikazana i kašnjenja kroz arhitekture, kao i radne učestalosti kola. Razliku u radnim učestalostima kola, na osnovu funkcionalnih blok-dijagrama arhitektura sa sl. 3.2, sl. 5.12 i sl. 5.26, definišu razlike u postojanju multipleksera koji su pridruženi osnovnim ćelijama polja. Međutim, na ove rezultate nezanemarljiv uticaj ima i optimizacija koju uključuje softverski paket korišćen za implementaciju.

Iz rezultata implementacije prikazanih u tab. 6.7 se vidi da je postignuto očekivano smanjenje površine potrebne za implementaciju savijenih arhitektura u odnosu na izvornu BP arhitekturu. Radne učestalosti savijenih arhitektura su nešto veće od radnih učestalosti postignutih implementacijom izvorne arhitekture, ali je pritom propusnost savijenih arhitektura N puta manja.

7. ZAKLJUČAK

Ovaj rad je posvećen sintezi savijenih semi-sistoličkih polja za FIR filtriranje. Rezultati prikazani u radu vode povećanju oblasti primene savijenih semi-sistoličkih polja. Mogućnosti konfiguracije savijenih polja su istražene i ugrađenje u polje kroz primenu tehnike savijanja. Omogućeno je dinamičko konfigurisanje parametara filtriranja na predloženim savijenim poljima. Arhitekture implementiraju fleksibilna izračunavanja i nude mogućnost povećanja propusnosti uz smanjenje broja operacija koje izvršavaju funkcionalne jedinice savijenog sistema. Fleksibilnost u izračunavanju je postignuta dinamičkim preslikavanjem operacija na različite funkcionalne jedinice savijenog sistema. Dinamičko preslikavanje, ostvareno primenom tehnike savijanja, omogućava prepoznavanje parametara filtriranja kao korisnički definisanih parametra.

Nakon izbora pogodnog kandidat-polja iz domena semi-sistoličkih polja, pre primene tehnike savijanja, uvedene su transformacije polja. Transformacije izvorne arhitekture omogućavaju uspešnu primenu tehnike savijanja, a zasnovane su na radu sa grafovima toka podataka. Novo-uvedene transformacije su dokazane matematičkim putem. Predstavljene su dve nove procedure sinteze savijenih polja. Prva procedura omogućava rad sa izmenjivom dužinom koeficijenata, dok se drugo rešenje odnosi na sintezu polja koje omogućava rad sa izmenjivim brojem ćelija filtra i izmenjivom dužinom koeficijenata. Prikazani su rezultati implementacije izabranog kandidat-polja i predloženih arhitektura u FPGA tehnologiji.

U daljem tekstu zaključka sledi pregled ostvarenih rezultata.

Iz domena semi-sistoličkih polja izabrano je "*bit-plane*" polje kao kandidat-polje za primenu tehnike savijanja. Arhitektura BP FIR filtra je prikazana u dva nivoa detaljnosti. Date su prenosne funkcije i funkcionalni opis BP arhitekture. Izložena je detaljna analiza zavisnosti dimenzija BP polja od karakterističnih parametara filtra.

Data je teoretska osnova za primenu tehnike savijanja pri projektovanju DSP sistema. Prikazan je matematički model savijanja i vremenskog usklađivanja arhitekture za zadato savijanje. Izložena je tehnika za projektovanje DSP arhitektura koje koriste minimalni broj registara. U cilju ilustracije tehnika, dati su primeri savijanja bikvadratnog filtra i IIR filtra.

Predstavljena je transformacija izvornog BP filtra u transponovani BP FIR filter, koja je izvedena u cilju pripreme polja za uspešnu primenu tehnike savijanja. Ispravnost rezultata transponovanog BP FIR filtra je dokazana transformacijom prenosne funkcije filtra.

Na arhitekturu TrBP FIR filtra je primenjena tehnika savijanja u cilju sinteze savijenog semi-sistolickog polja za FIR filtriranje sa promenljivim faktorom savijanja. Skupovi savijanja su arhitekturi dodeljeni u opštem obliku. Očekivani izgled savijene arhitekture je na osnovu dodeljenih skupova savijanja ilustrovan blok dijagramom. Skupovi savijanja su dati u matematičkom obliku i naznačeni na DFG-u TrBP arhitekture. Nakon dodele skupova savijanja, izvršena je sinteza savijene arhitekture. Date su jednačine savijanja i provereni uslovi savijanja. Savijena arhitektura je predstavljena DFG-om. DFG savijene arhitekture je prikazan u opštem obliku sa k skupova savijanja i faktorom savijanja N. Objašnjen je princip FIR filtriranja na savijenoj arhitekturi. Tok podataka kroz arhitekturu je grafički ilustrovan.

Savijena arhitektura je zadržala sve bitne odlike polazne arhitekture kao što su sitno-zrnasta protočnost i regularnost. U odnosu na izvornu BP arhitekturu površina potrebna za implementaciju je smanjena za nešto manje od N puta na račun vremena. Za razliku od polazne arhitekture, kod koje se po jedna izlazna reč dobija u svakom taktnom intervalu, savijena arhitektura generiše po jednu izlaznu reč na svakih N taktnih intervala. Izvršena je analiza DGF-a savijene arhitekture i uočeni su delovi na koje faktor savijanja ima uticaja. Projektovana je dodatna upravljačka logika koja omogućava promenu faktora savijanja arhitekture. Dodatna upravljačka logika podrazumeva k *shift*/*rotate* registra promenljive dužine, na mesto registara za pamćenje koeficijenata, kao i upravljačke signale za upravljanje dužinom registara. Prikazan je funkcionalni blok dijagram savijenog TrBP FIR filtra sa promenljivim faktorom savijanja. Ukoliko izračunavanje ne zahteva projektovanu dužinu koeficijenata, na račun smanjenja dužine koeficijenata, moguće je smanjiti faktor savijanja savijene arhitekture i time povećati propusnost. Propusnost savijene arhitekture linearno zavisi od dužine koeficijenata. Postupak filtriranja sa promenljivom dužinom koeficijenata je detaljno objašnjen. Broj koeficijenata kod ove arhitekture je nepromenjiv i jednak je projektovanom broju skupova savijanja.

U cilju dodatnog povećanja oblasti primene BP FIR filtra uvođenjem mogućnosti promene broja koeficijenata TrBP FIR filtru su dodeljeni novi skupovi savijanja. Novi način dodele skupova savijanja je omogućio sintezu savijenog semi-sistoličkog polja za FIR filtriranje sa promenljivim brojem i dužinom koeficijenata. Kod ove arhitekture je moguće na račun dužine koeficijenata povećati broj koeficijenata i na račun broja koeficijenata povećati dužinu koeficijenata.

Sinteza savijenog polja sa promenljvim brojem i dužinom koeficijenata je prikazana detaljno. Izvršena je analiza uticaja dodele skupova savijanja na tok podataka kroz savijenu arhitekturu. Tokovi podataka su grafički ilustrovani. Na osnovu analize tokova podataka uveden je novi način dodele skupova savijanja koji omogućava da broj koeficijenata bude različit od projektovanog broja skupova savijanja. Skupovi savijanja su dodeljeni arhitekturi u opštem obliku. Arhitektura vrši izračunavanje sa k_C koeficijenata dižine m_C. Date su jednačine savijanja za dodeljene skupove savijanja. Na sistem je primenjeno vremensko usklađivanje kako bi se omogućilo savijanje. Sistem nejednačina za vremensko usklađivanje je rešen korišćenjem grafa ograničenja. Rešenje sistema nejednačina je ilustrovano grafički. U cilju nalaženja karakterističnih vrednosti za sintezu savijene arhitekture kao što su npr. vremenski trenuci u kojima se ulazne reči uvode u arhitekturu, broj taktnih intervala za koje je ulazna reč aktivna na ulaznim linijama, minimalni broj potrebnih registara, izvršena je analiza rešenja za vremensko usklađivanje. Minimalni broj potrebnih registara

je određen korišćenjem linearnih grafikona aktivnost. Alokacija sadržaja registara je izvršena uz pomoć alokcione tabele. Na osnovu alokacione tabele je izvršena sinteza hardvera za uvođenje reči ulaznog niza u savijenu arhitekturu. Savijena arhitektura je prikazana DFG-om i funkcionalnim blok dijagrmom. Princip filtriranja na savijenoj arhitekuri je detaljno objašnjen.

Savijena arhitektura je zadržala sve bitne odlike polazne arhitekture. Površina potrebna za implementaciju je smanjena za nešto manje od N puta u odnosu na izvornu arhitekturu. Smanjenje površine je postignuto na račun vremena. Savijena arhitektura generiše po jednu izlaznu reč na svakih N taktnih intervala, bez obzira na dužinu i broj koeficijenata. Parametri arhitekture koji definišu veličinu savijenog polja, odnosno površinu silicijuma koju zauzima filter, su broj skupova savijanja i faktor savijanja. Na savijenom polju je moguće vršiti filtriranje sa promenljivim brojem i dužinom koeficijenata, uz ograničenje da je proizvod broja i dužine koeficijenata, koji određuje broj potrebnih operacija za izračunavanje izlazne reči, jednak proizvodu broja skupova savijanja i faktora savijanja.

Direktna posledica promeljivosti broja i dužine koeficijenata je da funkcionalne jedinice savijene arhitekture izvršavaju operacije iz različitih skupova savijanja u zavisnosti od broja i dužine koeficijenata sa kojima se vrši filtriranje. Nakon sinteze savijenog polja, data je sinteza hardvera za uvođenje bitova koeficijenata u polje. Hardver za uvođenje bitova koeficijenata u polje preuređuje bitove koeficijenata u toku inicijalizacije polja. HUK je sintetizovan korišćenjem matematičkih zavisnosti na kojima se zasniva način dodele skupova savijanja. Preuređenje bitova koeficijenata se vrši u skladu sa načinom preslikavanja operacija na funkcionlne jedinice u savijenom polju za zadat broj i dužinu koeficiejanta. Funkcionalnosti HUK su: serijsko uvođenje bitova koeficijenata u arhitekturu (režim inicijalizacije), preuređenje bitova koeficijenata (režim inicijalizacije) i ciklično-paralelno uvođenje bitova koeficijenata u polje (u toku filtriranja). Vreme inicijalizacije je jednako proizvodu broja skupova savijanja i faktora savijanja, i ne zavisi od broja i dužine koeficijenata. Jednostavnost HUK polja i regularnost veza vodi efikasnoj implementaciji po pogledu resursa na čipu. Upravljačka logika koja reguliše promenu broja i dužine koeficijenata je jednostavna i čini je kolo za generisanje periodičnog signala, periode jednake dužini koeficijenata. Ovaj signal se koristi za upravljanje radom hardvera za uvođenje bitova koeficijenata u polje.

Savijenom polju sa promenljivim brojem i dužinom koeficijenata dodata je mogućnost promene faktora savijanja. Dodatna upravljačka logika koja omogućava promenu faktora savijanja ima ulogu da smanji dužinu registara modula za uvođenje bitova koeficijenata u savijeno polje. Ovu upravljačku logiku čine multiplekseri dodati ćelijama HUK-a, koji omogućavaju implementaciju pomeračkih registara promenljive dužine. Kod savijenog BP polja za FIR filtriranje sa promenljivim brojem koeficijenata, dužinom koeficijenata i faktorom savijanja, izračunavanje se može ubrzati na račun smanjenja broja i/ili dužine koeficijenata.

Savijena semi-sistolička polja za FIR filtriranje su u cilju ilustracije mogućnosti konfiguracije implementirana na *Spartan* II FPGA familiji čipova. Prikazana je arhitektura i osnovne karakteristike *Spartan* familije FPGA. Dati su rezultati implementacije savijenih polja. Takođe, posvećena je pažnja analizi brzine rada, propusnosti arhitekture i obimu zauzetih resursa za različite konfiguracione parametre filtra.

Uvođenjem novog načina primene tehnike savijanja, kojim se omogućava dinamičko preslikavanje operacija, na sistematski način je izvršena sinteza savijenih semi-sistoličkih polja za FIR filtriranje sa mogućnošću dinamičke konfiguracije parametara. Na ovaj način je povećana oblast primene semi-sistoličkih polja za FIR filtriranje na aplikacije koje u toku rada zahtevaju promenu parametara filtra.

8. LITERATURA

[1] Y-C. Lin, F-C. Lin, "Classes of Systolic Arrays for Digital Filtering", *Int. J. Electronics*, Vol. 70, No. 4, 1991, pp. 729-737.

[2] I. Milentijevic, M. Stojcev, D. Maksimovic, "Configurable Digit - Serial Convolver of Type F", *Microelectronics Journal*, Vol. 27. No. 6, Sep. 1996, pp. 559-566.

[3] P. Corsonello, S. Perri, and G. Cocorullo, "Area-Time-Power Tradeoff in Cellular Arrays VLSI Implementations", *IEEE Transaction on Very Large Scale Integration (VLSI) Systems*, Vol. 8, No. 5, Oct. 2000, pp. 614-624.

[4] R. Lin, "Reconfigurable Parallel Inner Product Processor Architectures", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol.9, No.2, Apr. 2001, pp. 261-272

[5] R. Hawley, B. Wong, T. Lin, J. Laskowski, H. Samueli, "Design Techniques for Silicon Compiler Implementations of High-Speed FIR Digital Filters", *IEEE Journal of Solid-State Circuits,* Vol 31, No. 5, May 1996.

[6] I. Milentijević, "Projektovanje visoko pouzdanog semi-sistoličkog polja sa cifarsko-serijskim prenosom podataka za realizaciju konvolucije", Magistarski rad, Niš 1993.

[7] K.K. Parhi, "VLSI Digital Signal Processing Systems (Design and Implementation) ", *John Wiley & Sons*, In., New York, 2000.

[8] L. Paulson, L. Garber, "Reconfiguring Wireless Phones with Adaptive Chips", *IEEE Computer,* Vol. 36, Number 9, September 2003, pp. 9-11.

[9] D. Reuver, H. Klar, "A configurable Convolution Clup with Programmable Coefficients", *IEEE Journal of Solid State Circuits,* Vol. 27, No. **7,** July 1992, pp. 1121 - 1123.

[10] T. C. Denk, K. K. Parhi, "Synthesis of Folded Pipelined Architectures for Multirate DSP Algorithms", *IEEE Transaction on Very Large Scale Integration (VLSI) Systems,* Vol. 6*,* No. 4, Dec. 1998, pp. 595-607.

[11] K. K. Parhi, "VLSI Digital Signal Processing Systems (Design and Implementation)", *John Wiley & Sons*, In., New York, 2000.

[12] T. Noll, "Semi-systolic Maximum Rate Transversal Filters with Programmable coefficients", *Workshop of Systolic Architectures*, Oxford, 1986, pp. 103-112.

[13] D. Reuver, H. Klar, "A Configurable Convolution Chip with Programmable Coefficients", *IEEE Journal of Solid State Circuits*, Vol. 27, No. 7, July 1992, pp. 1121 -1123.

[14] T. C. Denk, K. K. Parhi, ”Synthesis of Folded Pipelined Architectures for Multirate DSP Algorithms”, *IEEE Transaction on Very Large Scale Integration (VLSI) Systems*, Vol.6, No. 4, Dec. 1998, pp. 595-607.

[15] J. Smith, “Introduction to Digital Filters with Audio Applications”, *Center for Computer Research in Music and Acoustics* (CCRMA), *Stanford University, California, USA*, May 2004, http://ccrma.stanford.edu/~jos/filters/

[16] D. Gibbon, “Convolution: filtering”, *Acoustic Phonetics and Speech Technology*, *University of Bielefeld, Germany*, May, 1996, http://coral.lili.uni-bielefeld.de/Classes/Summer96/Acoustic/acoustic/node16.html

[17] B. Parhami, D. Kwai, “Parallel Architectures and Adaptation Alghoritms for Programmable FIR Digital Filters with Fully Pipelined Data and Control Flows”, *Journal of Information Science Engineering* 19, (2003).

[18] I. Milentijevic, I. Nikolic, O. Vojinovic, V. Ciric and T. Tokic, "Folded Bit-Plane Architectures", *Proceedings of the Second International Conference on Informatics and Information Technology,* 20-23. December 2001, pp. 282-292.

[19] I. Milentijevic, T. Tokic, I. Nikolic, O. Vojinovic and V. Ciric, “Synthesis of Folded FIR Filter Architecture with Reordered Partial Products”, *Proceedings of a Workshop on Computational Intelligence and Informational Technologies*, Niš, 20-21. June 2001, pp. 155-160.

[20] I. Milentijevic, I. Nikolic**,** V. Ciric, O. Vojinovic, and T. Tokic, “Synthesis of Folded Fully Pipelined Bit-Plane Architecture”, *Proc. 23rd International Conference on Microelectronics (MIEL 2002) Vol 2*, Niš, Yugoslavia, May 2002, pp. 683-686.

[21] I. Milentijevic, V. Ciric, O. Vojinovic, T. Tokic, “Folded Semi-Systolic FIR Filter Architecture with Changeable Folding Factor”, *Neural, Parallel & Scientific Computations, Dynamic Publishers,* Atlanta, Vol. 10, No 2, 2002, pp. 235-247.

[22] I. Milentijević, V. Ćirić, O. Vojinović, T. Tokić, "Synthesis Procedure for Folded FIR Filter Architecture with Changeable Folding Factor", 3*rd Int. Conf. CiiT,* Molika, Macedonia, December 2002*,* pp. 12-20.

[23] T.Tokić, V. Ćirić, I.Milentijević, O.Vojinović, "FPGA Implementation of Folded Semi-Systolic FIR Filter with Changeable Folding Factor", 3*rd Int. Conf. CiiT,* Molika, Macedonia, Deecember 2002, pp. 21-30.

[24] I. Milentijevic, V. Ciric, T. Tokic and O. Vojinovic, "FPGA Implementation of Folded FIR Filter Architecture with Changeable Folding Factor". *Facta Universitatis*, Ser. Electronics and Energetics, Vol. 15, No 3, December, p451-464. University of Niš, Yugoslavia.

[25] I. Milentijevic, V. Ciric, T. Tokic and O. Vojinovic, “Folded Bit-Plane FIR Filter Architecture with Changeable Folding Factor”, *DSD 2002, EUROMICRO - Digital System Design,* Dortmund, Germany, September 2002. pp. 45-52.

[26] I. Milentijevic, V. Ciric, “Assignment of Folding Sets for Adaptive FIR Filtering on Folded Array”, *Proceedings of the WPS-DSD 2003, 29th EUROMICRO Conference,* Belek, Turkey, September 2003, pp. 21-22.

[27] V. Ciric, I. Milentijevic, “Configurable Folded Bit-Plane Architecture for FIR Filtering”, *Proceedings of the WPS-DSD 2003, 29th EUROMICRO Conference,* Belek, Turkey, September 2003, pp. 23-24.

[28] V. Ciric, I. Milentijevic, O. Vojinovic, “Retiming and Register Number Minimization for Adaptive FIR Filter Arcitecture”, *Proceedings of a Workshop on Computational Intelligence and Information Technologies,* Nis, Serbia and Montenegro, October 2003, pp. 93-96.

[29] I. Milentijević, V. Ćirić, "Synthesis of Coefficient Bit Reordering Module for Folded Bit-Plane Arrays", 4rd *Int. Conf. CiiT,* Molika, Macedonia, December 2003*,* (prihvaćeno za objavljivanje u zborniku).

[30] I. Milentijevic, V. Ciric, O. Vojinovic, “Flexible Folded FIR Filter Architecture”, *Proc. 24th International Conference on Microelectronics (MIEL 2004) Vol 2*, Niš, Yugoslavia, May 2004, pp. 723-726.

[31] Xilinx, Inc, “Spartan-II FPGAs: Spartan-II architecture”, *Xilinx Products and Services, Silicon Solutions*, http://www.xilinx.com/xlnx/xil_prodcat_product.jsp?iso...7862, USA, 2001.

SPISAK SLIKA

SPISAK TABELA

SPISAK SKRAĆENICA

ASIC - Application Specific Integrated Circuits

BP - *bit-plane*

CLB - Configurable Logic Block

DFG - Data Flow Graph

DLL - Delay Locked Loop

DSP - Digital Signal Processing

IIR - beskonačni impulsni odziv (*Infinite Impulse Response*)

IOB - Input-Output Block

FIFO - First In First Out

FIR - konačni impulsni odziv (*Finite Impulse Response*)

FJ - funkcionalna jedinica

FPGA - Filed Programmable Gate Array

HUK - Hardver za uvođenje bitova koeficijenata

LC - Logic cell

LUT - Look up table

MAC - *multiply/accumulate* jedinica.

PAL - Programabilna I kola

PGA - Programabilne gejtovske mreže

PLD - Programmable Logic Devices

PLA - Programmable Logic Arrays

PSM - Programmable Switch Matrices

SL - Slice

TDFG - Transformisani DFG

TrBP - Transponovani BP FIR filter

TrDFG - Transponovani DFG

VLSI - Very Large Scale Integration