Univerzite u Novom Sadu
Fakultet Tehničkih Nauka
Katedra za računarsku tehniku i međuračunarske komunikacije

Algoritmi i arhitekture DSP I

- PROGRAMSKA PODRŠKA DIGITALNIH SIGNAL PROCESORA

Primer 1:
FIR filter – C funkcija

Realizacija preko dve C naredbe:

```
#define N 25
int c[N], x[N];
int filter (void) {
int i, *coef, *in,*out;
for(i=0, coef=c,in =x,out =0;i<N;i++){
    out += (*coeff++)*(*in++)
}
return out;
}
```

- Koriste se dva pokazivaca
- Indirektno adresiranje sa samouvećanjem
- U petlji moženje i akumuliranje (MAC)
- Podeljen adresni prostor (npr: niz c u D0, x u D1) ubrzava petlju

Praktična realizacija FIR filtra

- Povezivanje obrade sa U/I jedinicama putem sprežnih podrutina, dva odvojena procesa:
  - Obrada, filtriranje
  - Učitavanje odbiraka u radnu memoriju DSP pod prekidom (ISR) ili preko DMA kontrolera
- Problem proizvodjač-potrošač:
  - Različite brzine rada/obrade
  - Sprega preko kružnog buffer
  - Semafori (prazan, pun)
- U praksi kružni buffer može biti dovoljno velik da da su dovoljna sva pokazivača (početak, kraj)
- Brzi prekidi kod DSP: rutine za obradu se umeću u protočnu obradu. Broj rutina brzog prekida mora biti manji od stepena protočne obrade!

Primer 2: Izračunavanje matematičkih izraza

- Ulaznim signalom se množi ton nosioca (blok za amplitudnu modulaciju): yk = xk coswtk
- Izračunavanje cos = razvoj u Tejlorov red

- Izračunavanje standardnih matematičkih funkcija zahteva veliki broj mašinskih intrukcija
- Rešenje:
  - Izračunavanje VAN realnog vremena I smeštanje u ROM
  - Ograničenja memorija uslovljava interpolaciju vrednosti

- DSP se često isporučuju sa već upisanim ROM tabelama (sin, cos)

UVOD - SPECIFIČNOSTI

- Projektovanje programske podške obuhvata :
  - Memorijske zahteve,
  - Ograničenja nametnuta korišćenim procesorom na dužinu programa,
  - Vreme izvršenja
- Dijagram toka programa kao razvojno sredstvo
- Upoznavanje
  - Arhitekturom procesora i dostupnim resursima.
  - Razvojnim okruženjem (asembler, simulator, emulator)
  - Algoritmom koji se razvija, memorijskim i vremenskim zahtevima za obradu.

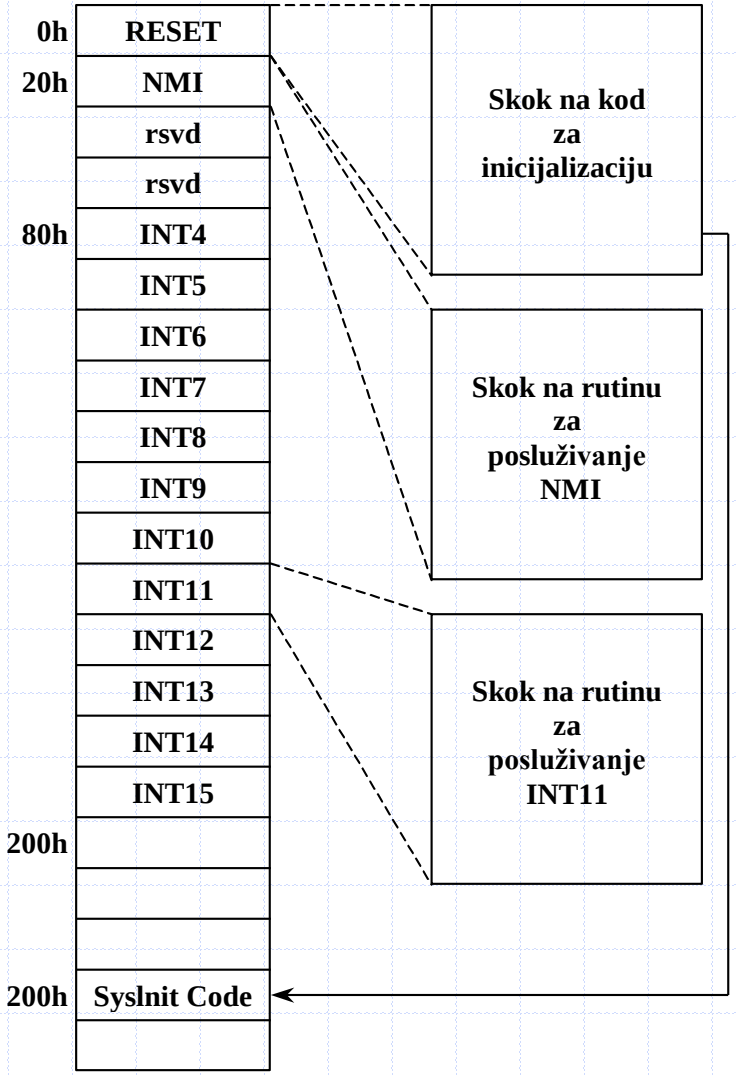
POČETNI KORACI PROGRAMIRANJA

- Početni koraci programiranja uključuju:
  - Deklarisanje naslova,
  - Dodelu brojčanih vrednosti simbolima/labelama i
  - Deklarisanje struktura podataka
- Tipičan program ima:
  - Blok za inicijalizaciju radi izbora režima rada,
  - Inicijalizaciju prekida,
  - Punjenje koeficijenata u memoriju
  - Nakon čega sledi glavni program sa pozivima podprograma i na kraju sami podprogrami i rutine za rukovanje prekidima

VEKTORI PREKIDA 1/2

- Termin “tabela vektora prekida” se koristi za tabelu vrednosti smeštenih u memoriju
- To je skup pokazivača na neke lokacije u programskoj memoriji ili kratke sekvence programskog koda
- Svaka lokacija vektora je pridružena fizičkom ili programskom događaju, tj. prekidu
- Programski tok se preusmerava na izvršenje koda uskladištenog na zadatoj lokaciji, tj. na lokaciju na koju pokazuje pokazivač

VEKTORI PREKIDA 2/2




```
        .sect   vectors.asm
Reset   B       SysInit
        nop
        nop
        nop
        nop
        nop
        nop
        nop
NMI     Instr1
        Instr2
        B       nmi_isr
        Instr4
        Instr5
        Instr6
        Instr7
        Instr8
rsv     .loop  16
         nop
        .endloop
ISR4    B   ...
```

INICIJALIZACIJA PROCESORA PO RESETU

- Prva operacija koju procesor izvršava nakon priključenja na napajanje ili po resetu treba da ga inicijalizuje u stanje koje zahteva glavni program:
  - Režim prekoračenja
  - Pokazivač na stranicu podataka
  - Posluživanje prekida
  - Proširenje znaka
  - Dodatni registri
- Potom sledi inicijalizacija periferija (A/D, D/A, CODEC, SSP, UART itd.)
- Na posletku se startuje glavni algoritam digitalne obrade signala

PROGRAMSKA PODRŠKA DSP UREĐAJA

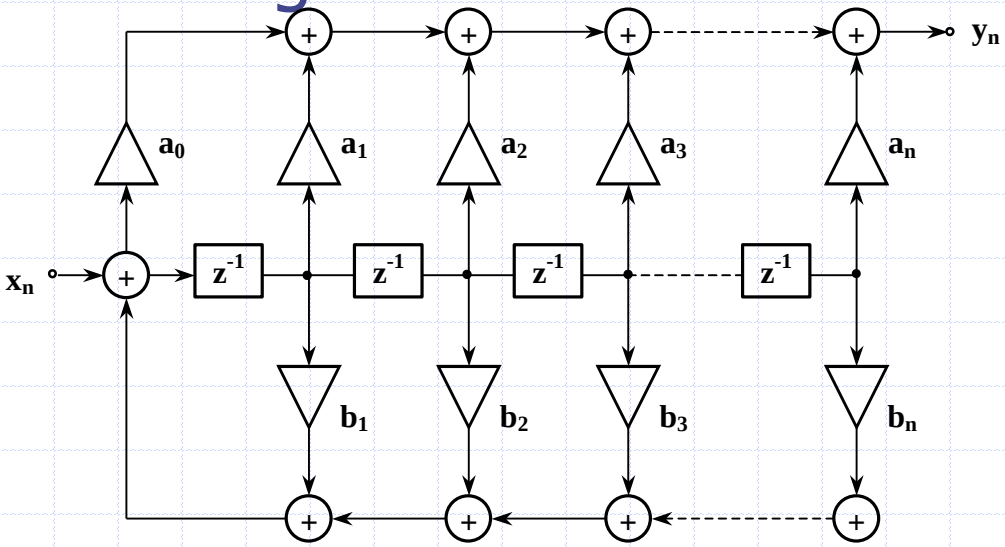
- Zadatak programera je da upravlja prenosom podataka između memorije i unutrašnjih registara na takav način da ALU radi na pravim odbircima podataka u ispravnoj sekvenci radi obavljanja željene obrade
- Efikasna PP = potpuno iskorišćenje mogućnosti paralelne obrade ugrađene u arhitekturu i skup instrukcija konkretnog uređaja koji se koristi
- HLLs – High-Level Languages, kao što su Pascal i C, se u opštem slučaju smatraju neefikasnim za DSP primenu

SKUP INSTRUKCIJA DSP

- **Aritmetičke operacije**
- **Logičke operacije**
- **Premeštanje podataka**
- **Upravljanje tokom programa**
- **Ispitivanje uslova**
- **Upravljanje sistemom**

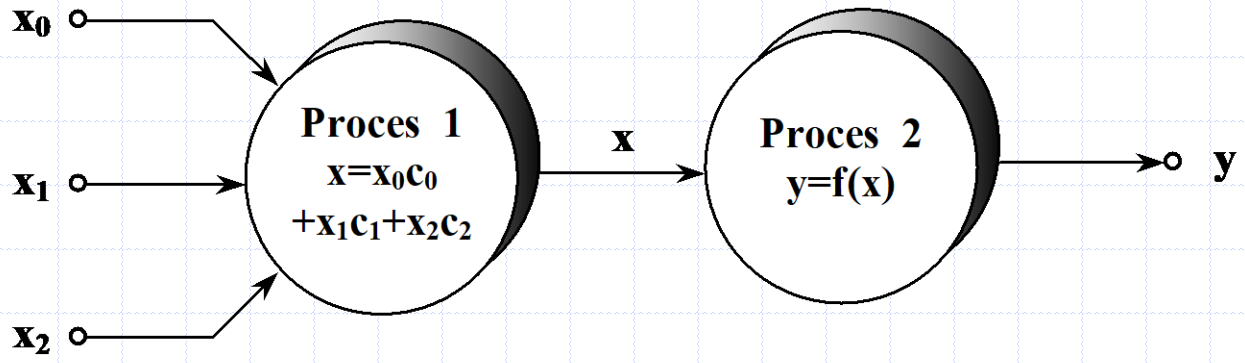
KARAKTERISTIKE PROGRAMSKE PODRŠKE DSP-a

- DSP program ima dve komponente
  - Pribavljanje podataka, prosleđivanje podataka na periferiju
  - Digitalna obrada signala, predstava u obliku grafa toka signala:

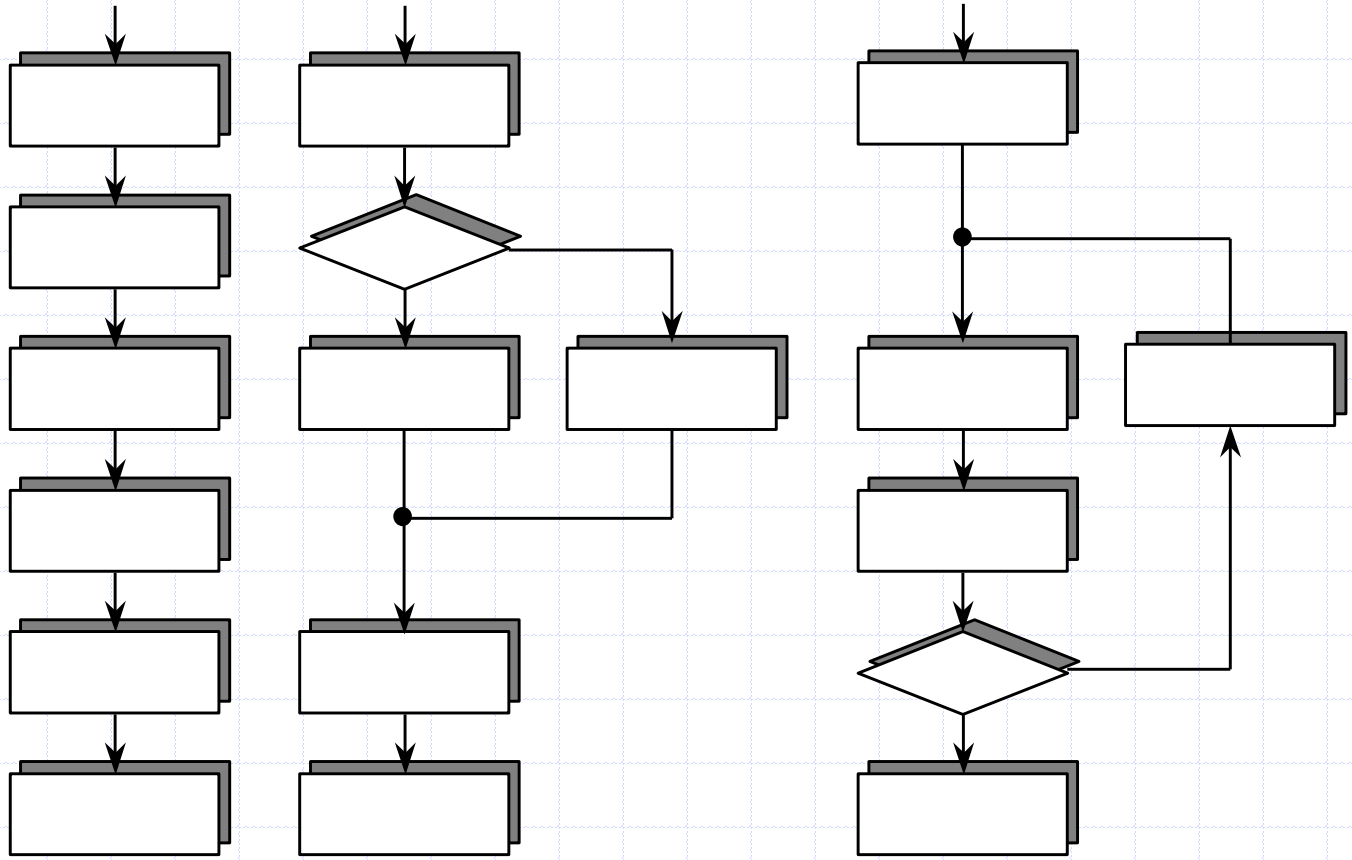


DOKUMENTOVANJE ALGORITMA OBRADE

- Sa stanovišta ulaza izlaza u digitalnu obradu signala, koristi se dijagram toka podataka:



DIJAGRAM TOKA KAO METODA ZA OPIS DSP ALGORITMA



Linearna sekvenca

Ako... Onda... Inače

Radi... Dok...

DOKUMENTOVANJE DSP ALGORITMA STRUKTUIRANIM OPISOM

begin

$y = x_0c_0$

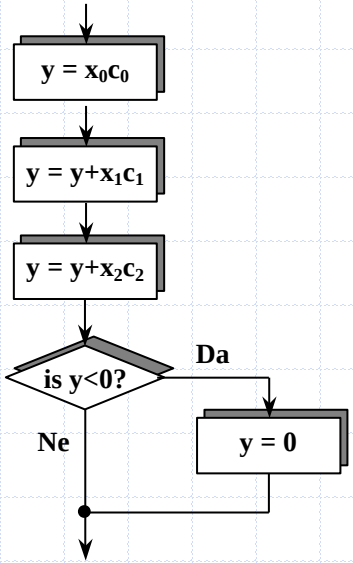
$y = y+x_1c_1$

$y = y+x_2c_2$

if $y < 0$ **then** $y=0$

end

STRUKTUIRAN OPIS



DIJAGRAM TOKA

begin

y = 0
i =0

repeat

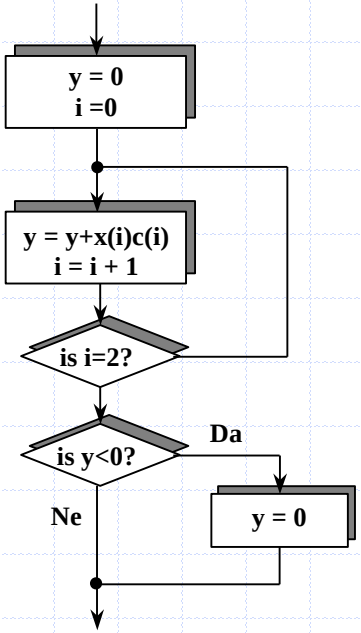
y = y+x(i)c(i)
i = i + 1

until i = 2

if $y < 0$ **then** y = 0

end

STRUKTUIRAN OPIS



DIJAGRAM TOKA

PISANJE KODA U ASEMBLERSKOM JEZIKU

- Asemblerski jezik je udaljen za samo jedan korak od binarnih instrukcionih reči koje instrukcioni dekoder stvarno koristi
- Instrukcije se specificiraju mnemonicima i unapred definisanim simbolima za označavanje unutrašnjih registara
- Blok u okviru blok dijagrama odgovara jednoj asemblerskoj liniji

DSP BIBLIOTEKE

- Za mnoge standardne algoritme postoje biblioteke (engl. software toolboxes), koje se mogu koristiti radi ubrzanja razvojnog procesa za:
  - FIR, IIR, FFT i kompresione algoritme
- PRIMER, TI obezbeđuje sledeće biblioteke:
  - standardna DSP biblioteka - DSPLib
  - biblioteka za obradu slike - IMAGELib
  - biblioteka za podršku integrisanom kolu - CSLib

RADNO OKRUŽENJE (eng. FRAMEWORK)

- Pre pojave standardizovanih programskih radnih okruženja, projektanti programske podrške DSP-a su morali da rade ponovni inžinjering (engl. re-engineering) algoritama prilikom njihove integracije u nov sistem
- Standardno radno okruženje za kodiranje koje rukuje pitanjima upotrebe resursa i definiše kako treba da sarađuju različiti algoritmi
- PRIMER RADNOG OKRUŽENJA - **eXpressDSP™**
  - skup pravila (konvencija) programiranja i sprega aplikativnog programiranja, API
  - pravila programiranja algoritama koja omogućavaju međusobnu saradnju različitih tipova algoritama

RAZVOJ KODA KORIŠĆENJEM VIŠIH PROGAMSKIH JEZIKA

- Najčešće korišćen viši programski jezik (HLL) za razvoj DSP aplikacija je ANSI C
- Efikasnost varira od kompajlera do kompajlera
- Programi pisani u C očekuju
  - definisano radno okruženje
  - Standardno prosleđivanje parametara
  - Metode oživljavanja PP – start up, sekvence posle reset-a
- Kada se piše PP u višem programskom jeziku, potreban je profiling

OPERATIVNO OKRUŽENJE C JEZIKA I NJEGOVA INICIJALIZACIJA

- C izvršno (engl. run-time) okruženje je skup konvencija, i to:
  - organizacija memorije i
  - upotreba registara
  - način korišćenja steka,
  - konvencije poziva funkcija
- PRIMER, TI rts.lib
- Rutina početnog punjača c_int00 obavlja sledeće zadatke inicijalizacije izvršnog okruženja
  - Definiše sistemski stek
  - Inicijalizuje promenljive i dodeljuje oblasti memorije za specifične potrebe
  - Poziva funkciju main čime započinje izvršenje C programa

OPERATIVNO OKRUŽENJE C (RUN-TIME)

- C SISTEMSKI STEK
  - Stek (magazinska memorija) je oblast memorije za čuvanje privremene informacije tokom određenih događaja kao što su pozivi funkcija
- Dodela i inicijalizacija oblasti memorije
  - C okruženje zahteva da se određene oblasti memorije dodele određenim izvršnim zadacima (enlg. run-time tasks)
  - Kompajler stavlja kod i podatke u definisane podblokove pod nazivom SEKCIJE – relokatibilan kod

KORIŠĆENJE SEKCIJA POKAZIVAČA

- Sekcije se smeštaju u memoriju radi prilagođavanja različitim konfiguracijama sistema
- Inicijalizovane sekcije
  - Inicijalizovane sekcije sadrže podatke ili izvršni kod.
- Neinicijalizovane sekcije
  - Neinicijalizovane sekcije služe za zauzimanje prostora u memoriji, obično RAM, koji se u vreme izvršenja koristi za skladištenje privremenih promenljivih
- Komandne datoteke povezivača za smeštanje C sekcija u memoriju
  - Konkretan raspored se definiše komandnom datotekom povezivača

PRIMER KOMANDNE DATOTEKE POVEZIVAČA

Program Z.1. Linker command fole used to set the C environment for the TI C6xxx DSP

```
/*****************************************************************/
/*   IPRAM = Internal Program RAM, Idram = Internal Data RAM     */
/*****************************************************************/
MEMORY
{
        IPRAM     :  origin = 0x0,          len = 0x10000
        IDRAM     :  origin = 0x80000000,   len = 0x10000
}
SECTIONS
{
        .vectors  > IPRAM
        .text     > IPRAM
        .bss      > IDRAM
        .cinit    > IDRAM
        .const    > IDRAM
        .far      > IDRAM
        .stack    > IDRAM
        .cio      > IDRAM
        .sysmem   > IDRAM
}
```

INICIJALIZACIJA SISTEMA I POZIV RUTINE POČETNOG PUNJAČA

- Posle kompajliranja, asembliranja, rutina početnog punjenja mora biti pozvana
- Način na koji se to radi je da se obezbedi vektor prekida koji pokazuje na funkciju početnog punjača c_int00
- Vektor prekida pridružen DSP RESET prekidu treba postaviti da pokazuje na funkciju početnog punjača
  - Postavljanje prve asemblerske instrukcije grananja u datoteci vectors.h, na poziv funkcije c_int00

RUTINE ZA PODRŠKU IZVRŠENJU PROGRAMA

- ANSI C ne obezbeđuje:
  - Dinamičku dodelu memorije
  - Rukovanje U/I
  - Matematičke funkcije
- Sa druge strane DSP uređaji ne podržavaju:
  - Aritmetiku u pokretnom zarezu
  - Deljenje
- Zbog gore navedenih problema, zahteva se dodatnu biblioteku rutina specifičnih za ciljnu platformu - (engl. run-time support library)

PRIMER RUTINE ZA PODRŠKU IZVRŠENJU PROGRAMA

- PRIMER TI C6xxx kompajler biblioteke
  - ANSI C standardnu biblioteku
  - C U-I biblioteku
  - funkcije niskog nivoa za podršku U/I aktivnostima sa operativnim sistemom upravljačkog računara (engl. host)
  - pripadajuće aritmetičke rutine
  - rutina za pokretanje sistema, c_int00
  - funkcije i makroi koji omogućavaju C kodu da pristupi određenim instrukcijama

RUKOVANJE PREKIDIMA U C-u

- Prekidne rutine vremenski kritične sekcije u DSP aplikacijama
- Moguće pisanje prekidnih rutina u C, moraju se poštovati određene konvencije

```
interrupt void example (void)
{
  ...
        C kod ide ovde
  ...
}
```

- Prekidne rutine moguće je pisati i u asembleru - efikasniji – brži kod

POMOĆNA PROGRAMSKA PODRŠKA

- Asembler
- Programski jezici višeg nivoa
- Kompajleri
- Povezivači – linkeri
- Punjači
- Kompaktori

ASEMBLER

- Asembler prevodi datoteku izvornog koda (ASCII tekst) na asemblerskom jeziku specifičnom za procesor u datoteku binarnog objektnog koda za određeni ciljni procesor
- Izuzetno važan alat jer se dosta programske podrške piše u asembleru
- Većina asemblera – makro asembleri
- COFF (engl. Common Object File Format) je standardni format datoteke objektnog koda i podržan je od strane mnogih asemblera

PRIMER MAKROA U ASEMBLERU

```
; Definise se makro "FIR" radi implementacije FIR filtra.
; Parametar "ntaps" definiše broj koeficijenata filtra (tapova).
FIR  macro  ntaps
     clr      a
     rep      #ntaps-1
     mac      x0,y0,a  x:(r0)+,x0  y:(r4)+,y0
     macr     x0,y0,a  (r0)-
     endm
```

```
; Postavi koeficijente i pokazivace na podatke.
     move   #data,r0
     move   #coeffs,r4
; Pozovi makro za FIR sa 256 koeficijenata.
     FIR      256
```

PROGRAMSKI JEZICI VIŠEG NIVOA

- Prednosti razvoja pp u programskim jezicima višeg nivoa:
  - Produktivnost.
  - Mogućnost održavanja
  - Prenosivost
- Nedostaci razvoja u programskim jezicima višeg nivoa
  - Brzina izvršenja
  - Veličina koda

PROGRAMSKI JEZIK C

- C je najpopularniji jezik visokog nivoa za razvoj programske podrške DSP procesora
- Blizak fizičkoj arhitekturi
- Besplatna verzija dostupna – GNU
- Svi trenutno raspoloživi DSP procesori i jezgra koji imaju podršku za jezike visokog nivoa imaju i C kompajler
- Nedostaci C-a:
  - nema fix_point i complex tip podataka
- Proizvođači koriste modifikovan C, kako bi podržali DSP zavisne stvari - ANSI NCEG

PROBLEM NE-EFIKASNOSTI JEZIKA VISOKOG NIVOA

- Glavni razlog za ovu neefikasnost je u tome što su DSP procesori, u opštem slučaju, izuzetno nepodesni za kompajlere.
- Utrošene su godine za razvoj minimalne arhitekture koja zadovoljava funkciju
- Problemi koji vode neefikasnoj pp:
  - Više memorijskih zona
  - Malo registara
  - Skup instrukcija koji nije ortogonalan
  - Bez podrške za stek u fizičkoj arhitekturi
  - Paralelni prenos podataka
  - Hardverska petlja