1 IMPLEMENTÁCIA FIR FILTRA POMOCOU PROCESORA ADSP218X

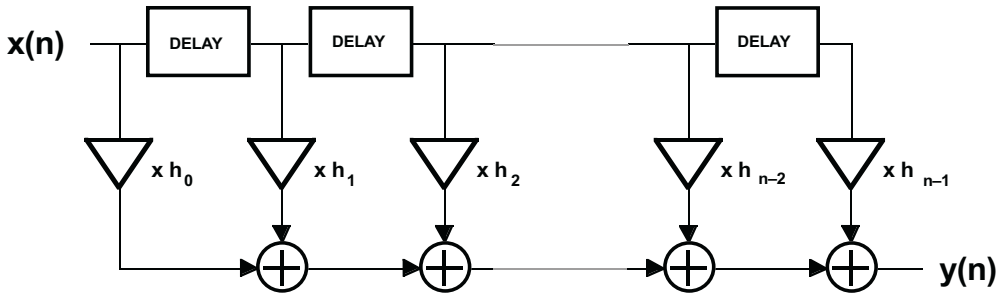
1.1 ÚVOD

FIR filter patrí z pohľadu signálových procesorov medzi najjednoduchší algoritmus ČSS. Prakticky všetky dostupné signálové procesory sú optimalizované práve pre tento typ algoritmu a procesory Analog Devices ADSP218x nie sú výnimkou. Programový kód pre FIR filter umožňuje demonštrovať základné vlastnosti harvardskej architektúry, modulo adresovanie a využitie MAC jednotky. Na analyzovanom zdrojovom kóde FIR filtra budeme tiež demonštrovať základné vlastnosti prostredia Visual DSP.

Opísaný zdrojový kód zatiaľ neumožní prácu pomocou reálnych technických prostriedkov (vývojovou doskou EZ-KIT2181 Lite) a jeho funkčnosť bude overovaná pomocou simulátora.

1.2 FIR FILTER

Preberaný podprogram FIR filtra patrí do knižnice voľne dostupného programového vybavenia dodávaného firmou Analog Devices a je detailne opísaný v knihe [1] na str. 67-69, ktorá je dostupná aj v elektronickej forme. Opisované algoritmy typicky obsahujú jadro algoritmu (v našom prípade **fir.asm**) a opis tohto algoritmu. Štruktúra implementovaného FIR filtra je zobrazená na obrázku 1.



Obr.1 Štruktúra transverzálneho FIR filtra

Program, ktorý inicializuje všetky potrebné registre a zabezpečí načítanie vstupných vzoriek je uložený v súbore **fir_test.asm**. Tieto súbory boli upravené tak, aby ich bolo možné spracovať v prostredí VisualDSP. Súbor **fir_test.zip [2]** obsahuje

všetky súbory projektu v prostredí VisualDSP ako aj ďalšie testovacie súbory potrebné na overenie správnej činnosti FIR filtra.

Súbor fir.asm upravený pre prostredie VisualDSP vyzerá takto:

```
/*****************************************************************************************************************
Nazov suboru:          FIR.asm
Datum modifikacie:     07-03-2002 MD

Opis:                  Podprogram realizuje FIR filter urceny koeficientami h(k) a
                       vstupnymi vzorkami x(n-k).

                       Rovnica:      y(n) = Suman pre k=0 do N-1 sucinov h(k)*x(n-k)

Parametre:             I0 --> ukazuje na najstarsiu vzorku v oneskorovacej linke
                       L0 = dlzka filtra (N)
                       I4 --> zaciatok tabulky koeficientov filtra (v PM)
                              POZOR, koeficienty musia byt v poradi:
                                   h(N-1), h(N-2), ..., h(1), h(0)
                       L4 = dlzka filtra (N)
                       M1,M5 = 1
                       CNTR = dlzka filtra-1 (N-1)

Predpoklady:           Oneskorovacia linka musi mat dlzku N
                       Oneskorovacia linka je umiestnena v bloku DM.
                       Instrukcie a koeficienty su ulozene v bloku PM.

Navratove hodnoty:     MR1 = sucet sucinov (zaokruhleny a saturovany)
                       I0 --> najstarsia vstupna vzorka v oneskorovacej linke
                       I4 --> zaciatok tabulky koeficientov filtra (v PM)

Zmenene registre:      MX0,MY0,MR

Pocet cyklov:          N - 1 + 5 + 2 cycles

Vyuzitie pamate:       Pocet instrukcnych slov (24-bitovych):
                       11 + N - 1 instrukcnych slov

                       Pocet datovych slov (16-bitovych):
                              N - pocet koeficientov (24-bitovych)
                              N - pocet slov v oneskorovacej linke (16-bitovych)

Poznamky:              Vsetky koeficienty a vzorky su v zlomkovom formate 1.15

*****************************************************************************************************************/

.GLOBAL fir;

/* kod v programovej pamati */
.section/pm program;

fir:   MR=0, MX0=DM(I0+=M1), MY0=PM(I4+=M5); /* nulovanie MR, predvyber MX0, MY0 */
       DO sop UNTIL CE;                      /* vypocet FIR filtra */
sop:          MR=MR+MX0*MY0(SS), MX0=DM(I0+=M1), MY0=PM(I4+=M5);
       MR=MR+MX0*MY0(RND);                   /* zaokruhlenie vysledku */
       IF MV SAT MR;                         /* pripadna saturacia vysledku */
       RTS;                                  /* navrat z podprogramu */
```

Pre opis fir.asm je dôležité predovšetkým nastavenie modulo adresovania pre registre **i0** a **i4**, ktoré sa uplatňuje v predchádzajúcom kóde v častiach, ktoré sú označené hrubo zvýrazneným písmom. Inicializáciu modulo adresovania zabezpečuje hlavný program fir_test.asm. V programovej pamäti (PM) sú uložené koeficienty FIR filtra a v dátovej pamäti (DM) sú ukladané aktuálne spracovávané vzorky $x(n), x(n-1), \ldots x(n-N+1)$, pričom N je počet koeficientov implementovaného FIR filtra. Koeficienty FIR filtra sú uložené v poradí $h(N-1), h(N-2), \ldots, h(1), h(0)$.

Pre nastavenie modulo adresovania pre niektorý z registrov **i***=**i0**, **i1**, **i2, i3**, **i4**, **i5**, **i6**, **i7** adresových aritmetických jednotiek DAG1 a DAG2 je potrebné splniť dve podmienky (sú nastavené v kóde fir_test.asm pre i0 a i4):

1) nastaviť príslušný **l*** register na **hodnotu zhodnú** s veľkosťou vytvoreného úseku cirkulačnej pamäte (v našom príklade na hodnotu **l0**=**l4**=N, čo zabezpečí modulo adresovanie pomocou registrov **i0** a **i4**), t.j. platí

$$l^* = N \tag{1.1}$$

2) umiestniť začiatok úseku cirkulačnej pamäte na adresu, ktorá je určená vzťahom[1]

$$start\_addr = M * 2^{\lceil \log_2(N) \rceil} \tag{1.2}$$

pričom M je ľubovoľné celé číslo[2]. Napríklad pre $N = 5$ musí začiatok pamäte začínať na adresách, ktoré sú násobkom hodnoty 8, t.j. na adresách 0, 8, 16, ...

Splnenie týchto podmienok (t.j. inicializáciu registrov **i*** a **l***) zabezpečuje hlavný program fir_test.asm a podporgram fir predpokladá ich korektné nastavenie.

1.2.1 PODPROGRAM FIR

Podprogram fir ktorý realizuje FIR filter s ľubovoľným počtom koeficientov (vhodne inicializovaných v programe, ktorý podprogram FIR filtra využíva) nám umožňuje opísať niektoré základné **inštrukcie procesora** Analog Devices ADSP218x a tiež niektoré **direktívy asemblera**[3] easm218x.exe.
Znaky /* */ označujú komentár a text medzi týmto znakmi nie je asemblerom ďalej spracovávaný. Direktíva **.GLOBAL fir,** zabezpečí, že návestie **fir** bude viditeľné aj v iných programoch a teda v týchto programoch je možné podprogram fir volať.

Funkcia fir využíva len 6 inštrukcií, pričom prvá zabezpečí vynulovanie akumulátora MR v MAC jednotke a predvýber vstupných operandov MX0 a MY0 MAC jednotky. Na začiatku je tak v MX0 najstaršia vzorka v oneskorovacej linke a v MY0 je koeficient $h(N-1)$. Nasledujúca inštrukcia DO zabezpečí realizovanie nasledujúcej slučky $(N-1)$-krát (hodnota $N-1$ je zapísaná do registra CNTR v hlavnom programe fir_test.asm a zabezpečí automatické hardvérovo podporované vykonanie slučky $N-1$ bez nutnosti testovania konca slučky). V slučke sa realizuje výpočet

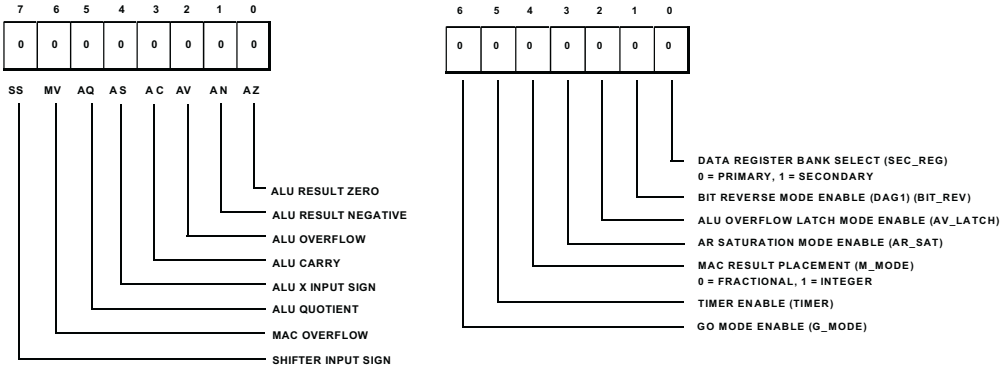
$$MR = \sum_{k=N-1}^{1} x(n-k) * h(k) \tag{1.3}$$

[1] Funkcia $f(x) = \lfloor x \rfloor$ je v anglickej terminológii nazývaná „floor function". Jej hodnotou je najväčšie celé číslo, ktoré je menšie alebo rovné x. Podobne funkcia $g(x) = \lceil x \rceil$ nazývaná „ceiling function", vracia najmenšie celé číslo väčšie alebo rovné x.

[2] Podmienka zaručuje, že spodných $k = \lceil \log_2(N) \rceil$ bitov adresy bude nulových. Táto podmienka vychádza z technickej realizácie adresových aritmetických jednotiek a je typicky vyžadovaná aj u signálových procesorov od iných výrobcov (napr. Motorola DSP560xx). Niektoré novšie procesory (napr. ADSP219x majú vylepšené technické prostriedky, a začiatok cirkulačných pamätí môže byť umiestnená na ľubovoľnej adrese) .

[3] Inštrukcie sú príkazy pre procesor (samozrejme vyjadrené symbolickým zápisom), ktoré vie cieľový procesor interpretovať a vykonať. Direktívy sú príkazy pre program realizujúci preklad (a teda cieľový procesor ich nevie interpretovať) zdrojového kódu (v našom cvičení easm218x.exe) a definuje napr. umiestnenie premenných v pamäti procesora, definovanie makier a pod.

pričom v akumulátore MR sa výsledky akumulujú s presnosťou 40 bitov. Predpokladá sa, že všetky údaje (vstupné vzorky a koeficienty) sú vo formáte 1.15 a súčin 16-bitových vzoriek v znamienkovom formáte (označenie (ss) v príslušnej inštrukcii) je vo formáte 1.31 (t.j. procesor je prepnutý do módu zlomkovej aritmetiky, ktorý je určený príslušným bitom v registri MSTAT zobrazenom na obrázku 2).



Obr.2 MSTAT (Mode Status) register určuje operačný mód procesora a ASTAT (Arithmetic Status) register zobrazuje stavové informácie generované výpočtovými jednotkami procesora. Namiesto priameho zápisu resp. čítania týchto registrov je možné využiť špeciálne inštrukcie – napr. inštrukcia DIS M_MODE prepne MAC jednotku do módu so zlomkovou aritmetikou. Prepnutie do celočíselnej aritmetiky je možné realizovať inštrukciou ENA M_MODE.

Nasledujúca inštrukcia zabezpečí dopočítanie výsledku v tvare

$$MR = \sum_{k=N-1}^{0} x(n-k) * h(k) \tag{1.4}$$

a zaokrúhlenie strednej časti akumulátora MR – t.j. registra MR1. Register MR1 obsahuje výsledok filtrácie FIR filtrom. V prípade že došlo k nastaveniu príznaku MV (ktorý sa nachádza v ASTAT zobrazenom na obrázku 2) (t.j. výsledok v 40 bitovom akumulátore MR je väčší ako cieľový 16 bitový MR1), realizuje sa obmedzenie na maximálnu 16-bitovú kladnú hodnotu ($0x7FFF \doteq 0.999969$) resp. minimálnu zápornú hodnotu ($0x8000 = -1.0$). Na zistenie príznaku MV nie je potrebné čítať register ASTAT, je použitá špeciálna inštrukcia podmieneného výkonu inštrukcie. Posledná inštrukcia RTS zabezpečí návrat z podprogramu FIR filtra.

Uvedený podprogram demonštruje efektívne využitie **MAC operácie** (*multiply and accumulate*), t.j. operáciu

$$MX = MX + MX0 \times MY0 \tag{1.5}$$

a dodatočné paralelné presuny z DM a PM. Činnosť celého podprogramu FIR bude podrobne vysvetlená v rámci cvičenia, doteraz prebrané informácie o procesore ADSP218x (predchádzajúce prednášky a cvičenia) sú však dostatočné na jeho kompletnú analýzu.

1.2.2 PROGRAM FIR_TEST.ASM

Program **fir_test.asm** využíva predchádzajúci podprogram fir na realizáciu celého FIR filtra s konkrétnymi koeficientmi a umožňuje overiť činnosť FIR filtra pre konkrétne vstupné hodnoty. Program fir_test.asm vyzerá takto:

```
/*************************************************************************************************************
Nazov suboru:            Fir_test.asm
Datum modifikacie:     07-03-2002 MD
Opis:      Demonstruje inicializaciu a cinnost FIR filtra implementovaneho v procesore ADSP2181.
           Priklad vyuziva priamu formu FIR filtra s jednoduchou presnostou. Vtupne data a koeficienty
           su v zlomkovom formate 1.15. Vstupne vzorky su nacitavane z prijimacieho registra RX1
           serioveho rozhrania SPORT1 a filtrovane udaje su zapisovane do vysielacieho registra TX1
           portu SPORT1.
*************************************************************************************************************/

#define    Ncoef        5                    /* pocet koeficientov FIR filtra */
#define    Fvz        256                    /* deliaci pomer 256 */

.EXTERN fir;

/* datova pamat - DM data */
.section/data data1;
.VAR/circ    Delay_Line[Ncoef];              /* oneskorovacia linka, klucove slovo circ zabezpeci
                                                umiestnenie modulo bufra na spravnu pociatocnu adresu */

/* programova pamat - PM data */
.section/pm data2;
.VAR/circ   COEFF[Ncoef] = "coef.dat";       /* koeficienty filtra v tvare h(N-1), h(N-2, ... h(1), h(0) */

.section/pm  interrupts;
                                             /* --tabulka interuptovych vektorov-- */
__reset: JUMP start; NOP; NOP; NOP;          /* resetovaci vektor */
      RTI; NOP; NOP; NOP;                    /* IRQ2 */
      RTI; NOP; NOP; NOP;                    /* IRQL1 */
      RTI; NOP; NOP; NOP;                    /* IRQL2 */
      RTI; NOP; NOP; NOP;                    /* SPORT0 vysielanie*/
      RTI; NOP; NOP; NOP;                    /* SPORT0 prijem */
      RTI; NOP; NOP; NOP;                    /* IRQE */
      RTI; NOP; NOP; NOP;                    /* BDMA */
      RTI; NOP; NOP; NOP;                    /* SPORT1 vysielanie */
      JUMP sample; NOP; NOP; NOP;            /* SPORT1 prijem */
      RTI; NOP; NOP; NOP;                    /* casovac */
      RTI; NOP; NOP; NOP;                    /* znizeny prikon (Power down) */

.section/pm  program;
                              /* -----inicializacia----- */
/***Inicializacia serioveho portu a zakladnych riadiacich registrov***/
/* aj ked nsaledujuci kod este nie je urceny pre konkretne technicke
   prostriedky, jeho filozofia sa priblizuje praktickemu vyuzitiu.
   Vstupne vzorky sa nacitavaju z prijimacieho registra SPORT1 (RX1)
   a po filtracii, ktora je realizovana v prijimacom interrupte SPORT1,
   su filtrovane vzorky zapisovane do vysielacieho registra SPORT1 (TX1) */


start:
      AX0=0x0000;
      DM(0x3FFE)=AX0;                        /* vsetky DM pouzivaju 0 cakacich stavov */
      DM(0x3FFD)=AX0;                        /* casovac je nepouzity */
      DM(0x3FFC)=AX0;                        /* nulovanie registrov */
      DM(0x3FFB)=AX0;
      DM(0x3FFA)=AX0;                        /* viackanalove prijimanie */
      DM(0x3FF9)=AX0;                        /* zakazane */
      DM(0x3FF8)=AX0;                        /* viackanalove prijimanie */
      DM(0x3FF7)=AX0;                        /* zakazane */
      DM(0x3FF6)=AX0;                        /* riadenie SPORT0 nepouzite */
      DM(0x3FF5)=AX0;                        /* casovanie SPORT0 nepouzite */
      DM(0x3FF4)=AX0;                        /* casovanie SPORT0 nepouzite */
      DM(0x3FF3)=AX0;                        /* autobufer SPORT0 nepouzity */
```

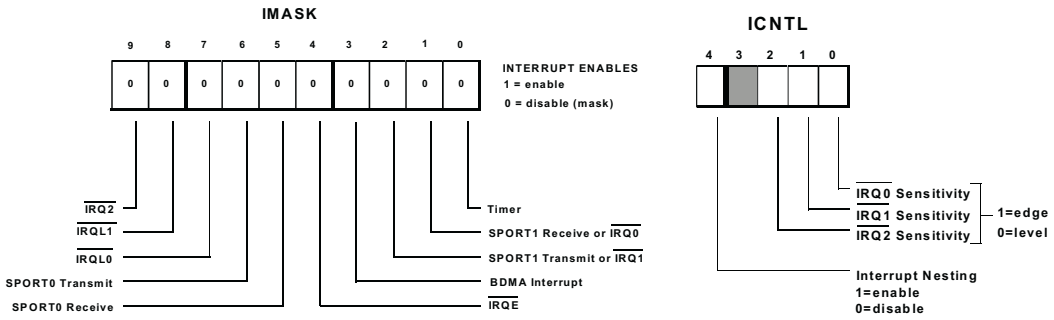
```
/* konfiguracia SPORT1 */
      AX0=0x6B1F;                   /* interne seriove hodiny */
      DM(0x3FF2)=AX0;               /* RFS reqd, normalny ramec,*/
                                    /* TFS reqd, normalny ramec,*/
                                    /* interne RFS, TFS, */
                                    /* bez kompresie, 16-bitove slova */
      AX0=0x0002;                   /* generuje 2.048 MHz SCLK1*/
      DM(0x3FF1)=AX0;               /* z 12.288 MHz CLKIN */
      AX0=Fvz-1;                    /* deli SCLK1 256 pre 8 kHz */
      DM(0x3FF0)=AX0;               /* vzorkovaciu frekvenciu */
      AX0=0x0000;
      DM(0x3FEF)=AX0;               /* autobufer SPORT1 nepouzity */

      I0=Delay_Line;                /* incializacia smernika na oneskorovaciu linku */
      M1=1;
      L0=LENGTH (Delay_Line);       /* inicializacia modulo adresovania */
      I4=COEFF;                     /* incializacia smernika na koeficienty */
      L4=length(COEFF);             /* inicializacia modulo adresovania */
      M5=1;

      AX0=0;
      CNTR=Ncoef;                   /* 'Ncoef' pozicii v oneskorovacej linke */
      DO zero UNTIL CE;
zero:   dm(I0,M1)=AX0;              /* nulovanie oneskorovacej linky */

      ICNTL=0x07;                   /* povolenie (edge sensitive) IRQs*/
      IMASK=0x02;                   /* povolenie prijimacieho int. od SPORT1 */

      AX0=0x0C00;                   /* SPORT1 povoleny, cakacie stavy */
      DM(0x3FFF)=AX0;               /* PM=0, boot cakacie stavy=0, */
                                    /* zavadzacia stranka (boot page) 0 */
      DIS M_MODE;                   /* zlomkovy mod MAC jednotky */

/*------ Cakanaie na vzorku ------------*/
receive:IDLE;                       /* cakanie na prerusenie od prijimaca */
      Jump receive;

/*------ Spracovanie vzorky ------------*/
sample: MX0=RX1;                    /* zapis prijatej vzorky do MX0 */
      DM(I0,M1)=MX0;                /* zapis na najstarsiu poziciu v oneskorovacej linke */

      CNTR=Ncoef-1;                 /* inicializacia pocitadla pre FIR filter */
      call fir;                     /* volanie podprogramu FIR filtra */

      TX1=MR1;                      /* vyselanie filtrovanej vzorky (v MR1) */
      RTI;                          /* navrat z prerusenia */
```

Aj keď uvedený zdrojový kód ešte nie je určený pre konkrétne technické prostriedky, je napísaný tak, aby sa čo najviac priblížil praktickej realizácii. Program predpokladá, že vstupné vzorky prichádzajú z prijímacieho registra sériového portu SPORT1[4]. Na začiatku programu je realizovaná inicializácia základných registrov procesora, konfigurácia SPORT1, inicializácia (vynulovanie) oneskorovacej linky, povolenie prerušení a skok do „nekonečnej slučky“ s inštrukciou IDLE (táto inštrukcia prepne procesor do nízkopríkonového režimu v ktorom procesor čaká až do výskytu prerušenia). Samotná filtrácia je realizovaná po príjme novej vzorky zo SPORT1, ktorú procesor spracuje pri obsluhe prerušenia od prijímača SPORT1, pričom procesor prečíta prijatú vzorku z registra RX1. **Najnovšia** vzorka sa zapíše na miesto **najstaršej** vzorky v oneskorovacej linke a inicializuje sa register CNTR. Po tejto príprave procesor

[4] Jeden z dvoch sériových kanálov, ktoré sú v procesoroch ADSP218x. Tieto periférie sú široko konfigurovateľné, pričom môžu napr. generovať všetky hodinové a rámcové signály interne prípadne použiť externe generované signály, môžu komprimovať resp. dekomprimovať spracovávané údaje (podľa A resp. μ zákona), prijímať 3 až 16 bitové slová a pracovať aj vo viackanálovom režime (čo sa typicky využíva napr. pri spracovaní časovo multiplexovaných signálov). V našej aplikácii predpokladáme, že k SPORT1 je pripojený hypotetický 16-bitový AD-DA kodek, ktorý využíva procesorom interne generované hodinové a rámcové signály.

realizuje skok do podprogramu[5], kde sa realizuje filtrácia FIR filtrom. Po návrate je v MR1 výstupná filtrovaná vzorka a tá sa zapíše[6] do vysielacieho registra TX1 periférie SPORT1.

Konfigurácia procesora a jeho periférií je realizovaná zápisom do registrov mapovaných do DM na adresách 0x3FF1-0x3FFF resp. špecializovaných registrov (IMASK, ICNTL), ktorých štruktúra a význam jednotlivých bitov je uvedený na obrázku 3. Pre pochopenie funkcie FIR filtra však ich detailná znalosť nie je potrebná.



Obr.3 Štruktúra registrov IMASK a ICNTL

V zdrojovom kóde sú dve direktívy

```
.section/data data1;
.VAR/circ    Delay_Line[Ncoef];
```

a

```
.section/pm data2;
.VAR/circ   COEFF[Ncoef] = "coef.dat";
```

Prvá definuje oneskorovaciu linku v Datovej pamäti s veľkosťou Ncoef, pričom začiatok alokovanej pamäte bude splňovať[7] podmienku (1.2). Konkrétne umiestnenie v DM procesora zabezpečí **linker** v procese linkovania. Podobne druhá direktíva zabezpečí umiestnenie koeficientov filtra, ktoré sú v súbore coef.dat[8], do Programovej pamäte, pričom je opäť zabezpečená podmienka (1.2).

Činnosť programu firt.asm bude podrobne analyzovaná počas cvičenia.

1.3 PREKLAD FIR FILTRA V PROSTREDÍ VISUALDSP

V súbore **fir_test.zip [2]** sú všetky potrebné súbory:

[5] Ak by mal byť program optimalizovaný na rýchlosť bolo by výhodnejšie využiť volanie makra FIR filtra, ktoré by zabezpečilo vsunutie inštrukcií FIR filtra priamo do tela obsluhy prerušenia. Tým by sa ušetrili 2 inštrukcie (CALL a RTS). V prípade, že by v systéme bolo niekoľko FIR filtrov môže byť naopak výhodnejšie použiť volanie podprogramu tak ako to je realizované v tomto príklade a zmenšiť tak dĺžku kódu v programovej pamäti.

[6] Zápis do vysielacieho registra je súčasťou obsluhy prerušenia od prijímača a teda stačí, ak je povolené len prerušenie od prijímača SPORT1.

[7] Toto definuje direktíva **circ.**

[8] Koeficienty sú v súbore zapísané v poradí $h(N-1),\ldots,h(1),h(0)$.

FIR.ASM	Zdrojovy kod podprogramu FIR filtra pre ADSP-2181
FIR_TEST.ASM	Testovaci program pre FIR filter (vyuziva seriovy port SPORT1)
ADSP-2181.LDF	Linker description file (opis systemu)
FIR_TEST.DPJ	VDSP projekt FIR prikladu. Projekt generuje subor fir_test.dxe.
COEF.DAT	Koeficienty FIR filtra (v celociselnom formate) v tvare h(N-1), h(N-2), ..., h(1), h(0)
X.DAT	Vstupne vzorky FIR filtra (v zlomkovom formate 1.15).
Y.DAT	Vystupne vzorky FIR filtra (v zlomkovom formate 1.15)
TST.M	Testovaci subor pre program MATLAB (vykresli chybu vypoctu v ADSP)

Preloženie celého projektu je možné realizovať priamo v integrovanom prostredí VisualDSP – **Project\Open** a **Project\Rebuild All**. Vlastnosti projektu je možné definovať v položke **Project\Project Options**, pričom je potrebné predovšetkým nastaviť správny cieľový procesor (ADSP2181). Ďalej je vhodné nastaviť aj generovanie ladiacich informácií (**Generate debug information**) v položke **Project\Project Options\Assemble**. Toto nastavenie umožní ladenie priamo v zdrojovom kóde. Pokiaľ chceme mať prehľad o umiestnení premenných v PM a DM je potrebné zabezpečiť generovanie **map súboru** (Generate symbol map) v položke **Project\Project Options\Link**. Linkovanie preložených častí projektu je realizované na základe opisu systému v súbore **ADSP-2181.LDF**, v ktorom sú definované vlastnosti hypotetického systému na báze ADSP2181. Tento súbor umožňuje programátorovi prispôsobiť projekt vlastnostiam cieľového hardvéru a jeho formát je podrobnejšie opísaný v príslušnej dokumentácii [3]. Pre pochopenie činnosti FIR filtra jeho detailná znalosť nie je potrebná.

1.4 LADENIE FIR FILTRA V PROSTREDÍ VISUALDSP

Ladenie v prostredí VisualDSP je pomerne jednoduché a s využitím integrovaného prostredia aj veľmi intuitívne. V procese ladenia je cieľom skontrolovať činnosť programu po jednotlivých inštrukciách ako aj pri spracovaní konkrétnych vstupných vzoriek. Je výhodné ak výstup simulácie je možné porovnať napr. s **vysoko-úrovňovou simuláciou** napr. v Matlabe[9]. Pre tento účel je výhodné ak je možné pripojiť k simulátoru **vstupné a výstupné súbory**, ktoré simulátorom čítané resp. do ktorých simulátor bude zapisovať výstupné hodnoty. Je výhodné ak formát týchto súborov je **prenositeľný** aj do iných systémov. Priradenie vstupných a výstupných súborov je možne realizovať príkazmi **Settings\Streams** pričom v položke **Debug target** je možné definovať konkrétne zariadenie **Device (Source** resp. **Destination)** (v našom prípade SPORT1) a v položkách **File** je možné definovať zodpovedajúce súbory v PC ako aj ich formát[10]. Počas simulácie je možné program krokovať príkazom **Debug\Step Into (F11)** prípadne spustiť príkazom **Debug\Run (F5)**. V prípade zložitejších simulácií je vhodné definovať **body zastavenia** (tzv. *breakpoints*), čo je možné vykonať príkazom **break**.

Počas cvičenia bude vykonaná kompletná simulácia programu fir_test.asm so vstupnými vzorkami zo súboru **x.dat**, ktoré sú súčasťou projektu.

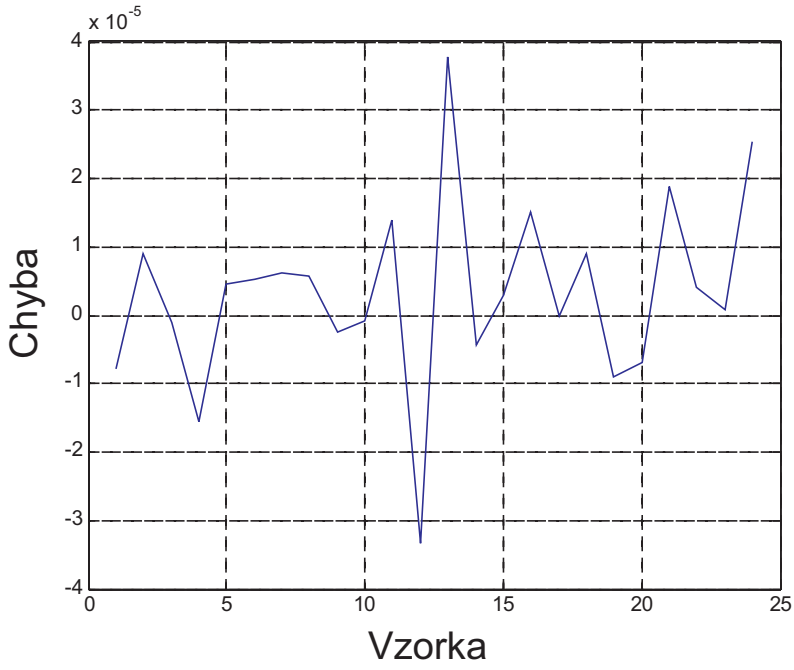
Nasledujúci jednoduchý program v Matlabe umožňuje porovnať výsledky zo simulácie ADSP2181 s referenčným výpočtom FIR filtra v Matlabe.

[9] Súčasťou projektu je aj testovací súbor test.m, pomocou ktorého je možné porovnať presnosť výpočtu FIR filtra v 16-bitovom procesore ADSP2181 a výpočtom FIR filtra v prostredí MATLAB s presnosťou 64 bitov.

[10] Sú podporované formáty: Hexadecimal, Unsigned Integer, Signed Integer, Octal, Fractional, Character, Binary.

```
load coef.dat;                  % koeficienty FIR filtra v celociselnom formate (16.0)
                                % v tvare h(N), h(N-1), ..., h(1) t.j. otocene!!!
h = coef/2^15;                  % koeficienty filtra v zlomkovom formate (1.15)
N = max(size(h));
for k=1:N
   hh(k) = h(N+1-k);            % transformacia do tvaru h(1), h(2), ... h(N)
end
load x.dat;                     % vstupne vzorky (format 1.15)
load y.dat;                     % vysledky vypocitane vo VisualDSP (format 1.15)
yy = filter(hh,1,x);            % referencny vypocet FIR filtra v Matlabe (pouzite
                                % otocene koeficienty!)

m1 = max(size(yy));             % velkost yy
m2 = max(size(y));              % velkost y
m = min(m1,m2);

e = yy(1:m) - y(1:m);           % rozdiel medzi Matlabom a VisualDSP

plot(e);                        % max chyba by mala byt 2^(-15)
title('Chyba vypoctu v ADSP218x');
xlabel('Vzorka');
ylabel('Chyba');
grid;
```

pričom na obrázku 4 je zobrazená chyba výpočtu FIR filtra v procesore ADSP218x. Z obrázku je zrejmé, že maximálna chyba je približne na úrovni 1 LSB $\doteq 2^{-15} = 3.1*10^{-5}$.



Obr.4 Chyba výpočtu FIR filtra v procesore ADSP218x

1.5 ZÁVER

Implementovaný FIR filter vyžaduje na realizáciu FIR filtra s N koeficientmi $N+10$ inštrukčných cyklov. Signálové procesory rodiny ADSP218x realizujú od 33 do 80 MIPS a tieto hodnoty určujú maximálne frekvencie vzorkovania F_{vz} resp. pri danej frekvencii vzorkovania maximálny rád FIR filtra N, ktorý je možné implementovať.

Napr. pre FIR filter s $N = 1000$ koeficientmi (čo už je pomerne zložitý FIR filter) je možné s procesorom, ktorý realizuje 50 MIPS pracovať až do

$$F_{vz} \approx \frac{50 \times 10^6}{N + 10} = \frac{50 \times 10^6}{1000 + 10} = 49{,}5\left[kHz\right] \tag{1.6}$$

čo dokumentuje efektívnosť architektúry ADSP pri realizácii tohto základného algoritmu ČSS.

LITERATÚRA

[1] Mar, A.: *Digital Signal Processing Applications using the ADSP-2100 Family, Volume 1*. Prentice Hall, Englewood Cliffs, 1992. (dostupné aj v elektronickej forme **\SPvT\Knihy \DSP_Books\Using_ADSP-2100\…**)

[2] (dostupné v elektronickej forme **\SPvT\Cvicenia\Fir_test.zip**)

[3] *VisualDSP++2.0 Linker and Utilities Manual for ADSP-21xx DSPs*. Analog Devices, Inc., April 2001, (dostupné aj v elektronickej forme **\VisualDSP\Docs\21xx_lkm*.pdf**).