

### 3 PROCESSADOR DIGITAL DE SINAIS (DSP)

Basicamente, o DSP é um processador com uma arquitetura interna customizada para a execução das operações usuais de processamento digital de sinais (PDS), tais como: soma, atrasos, multiplicação com acumulação, etc. Normalmente, a arquitetura do DSP é composta por: unidades computacionais (ALU, SHIFTER, MAC), de geração de endereços, de controle de programa, memória de dados, de programas, diversos barramentos, amplo conjunto de periféricos, etc. Essa variedade de recursos diferencia o DSP dos processadores de uso geral bem como possibilita ao DSP atender aos requisitos computacionais de desempenho dos algoritmos de PDS. A Fig. 102 apresenta os principais blocos que compõem o DSP.

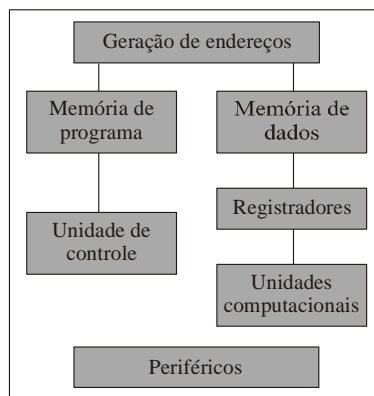


Fig. 102 – Principais blocos que compõem o DSP.

A ANALOG DEVICES (AD) [8] é um dos principais fabricantes de processadores DSP e possui diferentes opções de dispositivos de ponto fixo e ponto flutuante. Aqui, é considerado o dispositivo DSP BF-533 de ponto fixo (16bits) da família Blackfin da AD [9].

Para facilitar a implementação e testes de novas aplicações, a AD disponibiliza um ambiente de desenvolvimentos integrado (IDE) e kits de desenvolvimento. O IDE da AD é denominado Visualdsp [10] e os kits de desenvolvimento de EZ-KIT Lite [11]. Deve-se ressaltar que o VisualDSP permite a implementação de algoritmos utilizando as linguagens de programação “C” [12] e “Assembly” [13]. Para exemplificar a implementação de algoritmos usando tais linguagens, é apresentado, na Tabela 3, um trecho de código cuja função executada é equivalente.

Tabela 3 – Trecho de código equivalente em “C” e “Asm”

Código em “C”	Código em “Asm”
<pre> for (j=1;j&lt;10;j++) { Acc=L_mac(Acc,*pa++,*pi--); } </pre>	<pre> a0=0  r1.l=w[i1++]  r2.l=w[i0--]; Loop loop_3 lc1 = P1; Loop_begin loop_3; a0+=r1.l*r2.l  r1.l= w[i1++]  r2.l=w[i0--]; Loop_end loop_3; </pre>

Com relação ao EZ-KIT Lite, suas principais características são [11]:

- Processador DSP BF 533 (756 MHz),
- Memória SDRAM (64 MB),
- Memória flash (2 MB),
- 4 entradas e 6 saídas (96KHz/48KHz),
- Leds,
- Push buttons.

A Fig. 103 mostra o diagrama de blocos do EZ-KIT Lite com o conversor analógico digital (CAD), o conversor digital analógico (CDA) e o DSP BF-533.

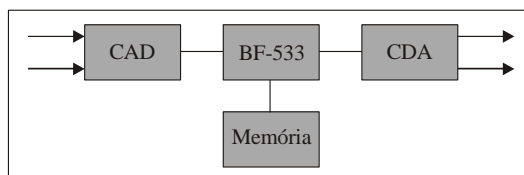


Fig. 103 - Diagrama de blocos do EZ-KIT Lite.

Na sequência, são apresentados alguns exemplos de algoritmos que serão posteriormente implementados e executados em tempo real no EZ-KIT Lite.

## 4 EFEITOS DIGITAIS – DIAGRAMA DE BLOCOS

### 4.1 Introdução

O *delay* (atraso) é o bloco básico usado na obtenção de diversos efeitos como, por exemplo, vibrato, *flanger*, *chorus*, *reverb*, eco, etc.

O *delay* atrasa um sinal digital de um número de amostras desejadas. Dependendo de qual sinal é atrasado, obtém-se um *Comb filter* FIR (sinal de entrada) ou *Comb filter* IIR (sinal de saída). A equação de um *Comb filter* FIR [14] é dada por:

$$y(n) = x(n) + ax(n-D) \quad (1)$$

onde  $y(n)$  é a amostra de saída,  $x(n)$  é a amostra de entrada,  $a$  é o fator de ponderação,  $D$  é o valor do atraso em amostras e  $x(n-D)$  é a amostra de entrada atrasada de  $D$  amostras.

O diagrama de blocos do *Comb filter* FIR é mostrado na Fig. 104 e sua correspondente resposta em frequência é apresentada na Fig. 105.

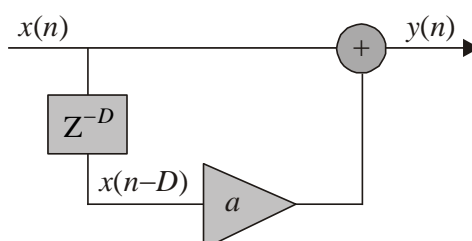


Fig. 66 .

Fig. 104 - Diagrama de blocos do Comb filter FIR.

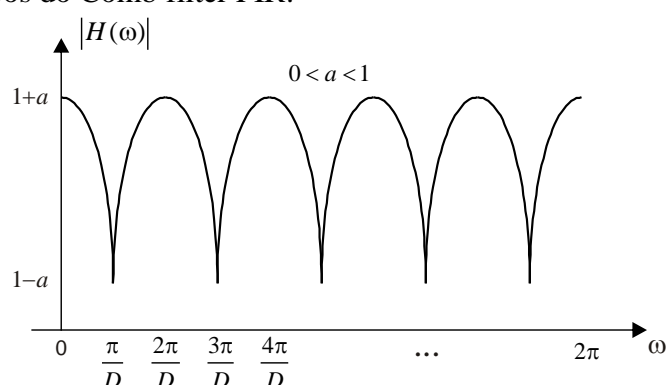


Fig. 105 Resposta em frequência do Comb filter FIR [14].

A equação recursiva de um *Comb filter* IIR é dada por [14]:

$$y(n) = x(n) + ay(n-D) \quad (2)$$

onde  $y(n-D)$  é a amostra de saída atrasada de  $D$  amostras.

O diagrama de blocos do *Comb filter* IIR é apresentado na Fig. 106 e sua correspondente resposta em frequência na Fig. 107.

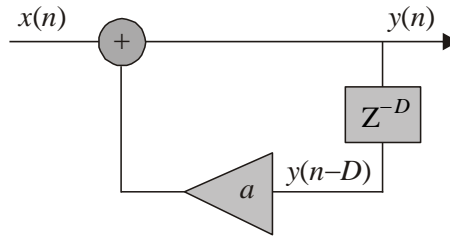


Fig. 106 - Diagrama de blocos do Comb filter IIR.

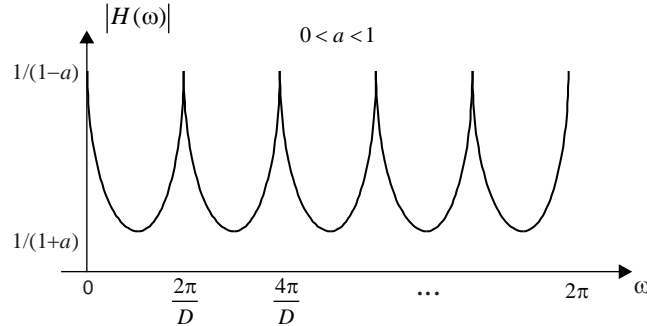


Fig. 107 - Resposta em frequência do Comb filter IIR [14].

## 4.2 Vibrato

O valor escolhido para o atraso  $D$  em amostras (de forma equivalente, em segundos) altera o som percebido bem como o tipo de efeito de áudio. Por exemplo, o efeito vibrato é implementado na forma digital variando o atraso  $D$  continuamente no tempo [14]-[16]. O diagrama de blocos que representa tal implementação é mostrado na Fig. 108.

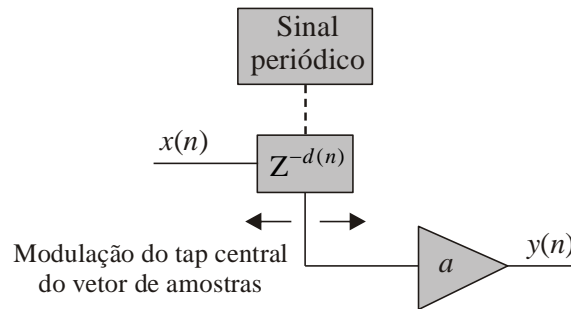


Fig. 108 Diagrama de blocos do vibrato.

A equação correspondente do vibrato é dada por [14]:

$$y(n) = ax(n - d(n)) \quad (3)$$

onde  $y(n)$  é a amostra de saída,  $d(n)$  é o atraso variante no tempo,  $a$  é o fator de ponderação e  $x(n - d(n))$  é a amostra de entrada atrasada de  $d(n)$  amostras.

O atraso variante no tempo ( $d(n)$ ) é obtido a partir de um oscilador de baixa frequência (LFO) que gera um sinal periódico, dado por [14]:

$$d(n) = \frac{D}{2} [1 - \cos(2\pi n f_{cy})] \quad (4)$$

onde  $D$  é o valor máximo do atraso em amostras e  $f_{cy}$  é a frequência do LFO.

Na prática, o valor máximo utilizado para o atraso está entre 0 ms e 3 ms. Considerando, por exemplo, que a frequência de amostragem é 8kHz, obtém-se 24 amostras para o máximo atraso  $D$ . Efeitos interessantes são obtidos quando a frequência do LFO é menor que 5Hz.

Para a implementação em DSP do algoritmo do vibrato, deve-se alocar um vetor de tamanho  $D$  para armazenar as amostras de entrada bem como calcular o atraso  $d(n)$ . Normalmente, existem duas opções para se obter esse atraso. A primeira considera uma tabela com os valores anteriormente armazenados. Na segunda opção, é utilizada uma equação recursiva para calcular o sinal periódico. Normalmente, devido à baixa frequência do LFO (menor que 5Hz), há uma relação de compromisso entre complexidade computacional e

utilização de recursos de memória, Na primeira, tem-se uma menor complexidade computacional com maior recurso de memória, enquanto na segunda opção tem-se uma maior complexidade computacional com menor recurso de memória [17].

Outra consideração relevante é sobre o valor obtido para o atraso  $d(n)$ , o qual pode ser fracionário. Pode-se perceber que tal situação resulta em um problema, dado que  $d(n)$  corresponde a um índice do vetor de amostras. Normalmente, a solução usada em tal situação é considerar a interpolação linear entre duas amostras nos instantes de tempo  $M + 1$  e  $M$ , conforme apresentado por [18]:

$$y(n) = x(n - [M + 1]) \text{frac} + x(n - M)(1 - \text{frac}) \quad (5)$$

onde  $\text{frac}$  corresponde ao valor fracionário do atraso.

### 4.3 Flanger/Chorus

Os efeitos *flanger/chorus* podem ser implementados de forma equivalente ao *Comb filter* FIR, ou seja, adicionando o sinal de entrada com uma réplica atrasada no tempo, conforme diagrama de blocos apresentado na Fig. 109. Equivalente ao vibrato, a réplica é obtida de um atraso variante no tempo ( $d(n)$ ), que é controlado por um LFO. Normalmente, no *flanger*, o atraso é menor que 10ms e a frequência do LFO é menor ou igual a 2Hz. Já, para o *chorus*, o atraso deve ficar entre 10ms e 30ms e a frequência do LFO menor ou igual a 1Hz [18-19].

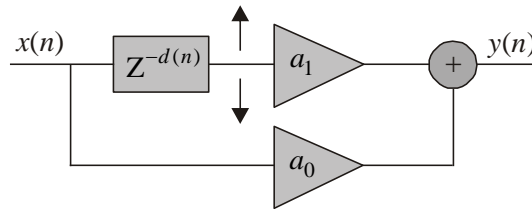


Fig. 109 Diagrama de blocos flanger/chorus.

A equação geral para os efeitos *flanger/chorus* é dada por [14]:

$$y(n) = a_0 x(n) + a_1 x(n - d(n)) \quad (6)$$

onde  $y(n)$  é a amostra de saída,  $d(n)$  é o atraso variante no tempo,  $a_0$  e  $a_1$  são os fatores de ponderação e  $x(n - d(n))$  é a amostra de entrada atrasada de  $d(n)$  amostras.

A correspondente resposta em frequência para  $a_0 = 1$  e  $0 < a_1 < 1$  é mostrada na Fig. 110. Pode-se observar em tal Fig. que a variação de  $d(n)$  causa a atenuação de diferentes componentes de frequência.

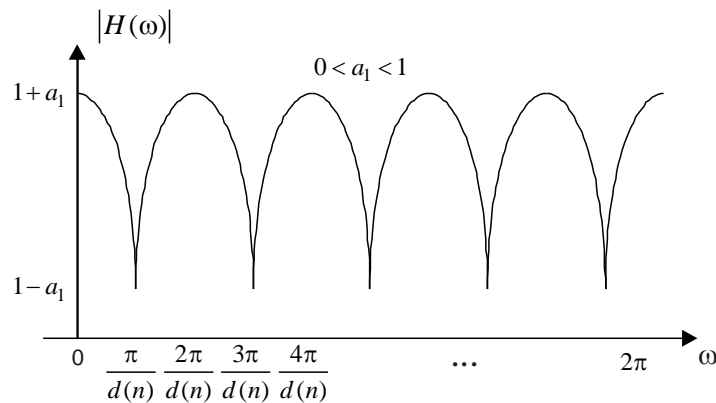


Fig. 110 Resposta em frequência do flanger/chorus.

Outra estratégia de implementação do *flanger/chorus* foi proposta por Dattorro [19]. A Fig. 111 mostra o diagrama de blocos de tal estratégia.

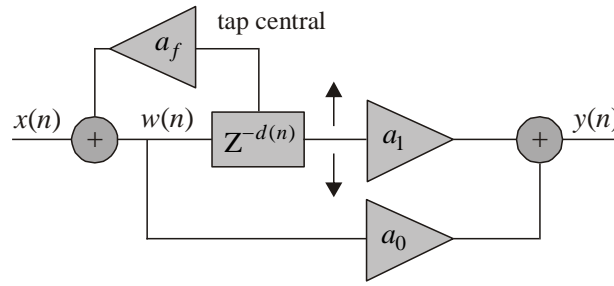


Fig. 111 Diagrama de blocos do *flanger/chorus* proposto em [19].

As equações que correspondem a implementação da Fig. 111 são dadas por:

$$w(n) = x(n) - a_f w(n - \frac{D}{2}) \quad (7)$$

$$y(n) = a_0 w(n) + a_1 w(n - d(n)). \quad (8)$$

Deve-se observar que o diagrama apresentado na Fig. 111 permite representar os efeitos vibrato, *flanger* e *chorus*. Para cada efeito, deve-se escolher os valores adequados para os coeficientes, tamanho do atraso, frequência e tipo do sinal usado no LFO, conforme apresentado na Tabela 4 [19].

Tabela 4 Representação dos efeitos vibrato, *flanger* e *chorus* usando a estrutura proposta por Dattorro

Efeito	Coeficientes	Atraso D (ms)	LFO (Hz)	Tipo de sinal usado no LFO
Vibrato	$a_f = 0$ $a_0 = 0$ $a_1 = 1$	0 – 3	0.1 até 5Hz	Senoidal
<i>Flanger</i>	$a_f = 0.7$ $a_0 = 0.7$ $a_1 = 0.7$	$D < 10$	$< 2\text{Hz}$	Senoidal
<i>Chorus</i>	$a_f = -0.7$ $a_0 = 0.7$ $a_1 = 1$	$10 < D < 30$	$< 1\text{Hz}$	Senoidal Ruído filtrado banda estreita

#### 4.4 Tremolo

O Tremolo é implementado digitalmente multiplicando o sinal de entrada pela saída do LFO. Tal operação equivale a modular em amplitude a entrada, conforme diagrama em blocos apresentado na Fig. 112.

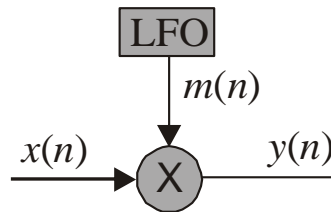


Fig. 112 Diagrama em blocos do Tremolo.

Na prática, a frequência de saída do LFO é ajustada entre 1Hz e 10 Hz. A equação para o cálculo de tal frequência é dada por [17]:

$$m(n) = \sin(2\pi.n.f_{cy}) \quad (9)$$

onde  $m(n)$  é o sinal de modulação e  $f_{cy}$  é a frequência do sinal de modulação.

O sinal na saída do Tremolo é dado por:

$$y(n) = m(n)x(n) \quad (10)$$

onde  $x(n)$  é o sinal de entrada e  $y(n)$  é o sinal de saída.

## 4.5 Eco

O eco pode ser implementado digitalmente a partir do *Comb filter* FIR com a especificação do valor desejado do atraso  $D$ . Por exemplo, considerando que a frequência de amostragem seja 8kHz e que se deseja um atraso máximo de 0,1 segundos, deve-se adotar o valor 800 para o atraso  $D$  e alocar um tamanho correspondente para o vetor de atrasos  $x(n-D)$ .

A equação para a implementação do eco é dada por [14]:

$$y(n) = x(n) + ax(n-D) \quad (11)$$

onde  $y(n)$  é a amostra de saída,  $x(n)$  é a amostra de entrada,  $a$  é o fator de ponderação,  $D$  é o valor do atraso em amostras e  $x(n-D)$  é a amostra de entrada atrasada de  $D$  amostras.

O diagrama de blocos para a implementação do eco é mostrado na Fig. 113.

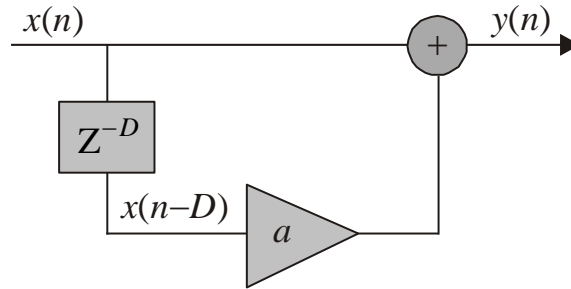


Fig. 113 Diagrama de blocos do eco.

## 4.6 Reverberação

O efeito de reverberação ou *reverb* simula a reflexão do som em um ambiente fechado, produzindo repetições do som original. O som produzido na reverberação é classificado em três componentes: som direto, reflexões iniciais (*early*) e reflexões finais (*late*). As reflexões iniciais ocorrem de 10ms a 100ms após o som direto. Já as reflexões finais são maiores que 100ms [14]. A Fig. 114 mostra os componentes da reverberação.

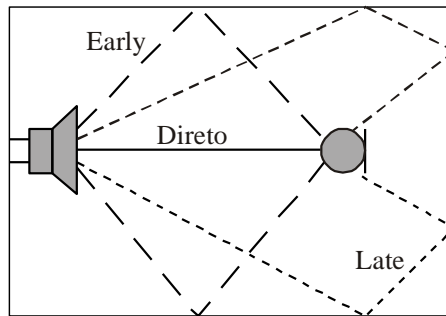


Fig. 114 Componentes da reverberação.

A implementação digital da reverberação deve, necessariamente, considerar os três componentes citados, de forma a produzir um efeito mais próximo do “real”. Os algoritmos mais utilizados para a implementação da reverberação foram propostos por Schoreder [20] e Moorer [21], consistindo em uma sequência de *Comb filter* IIR (C1 a C4) e filtros passa-tudo (A1 e A2), conforme apresentado no diagrama de blocos da Fig. 115.

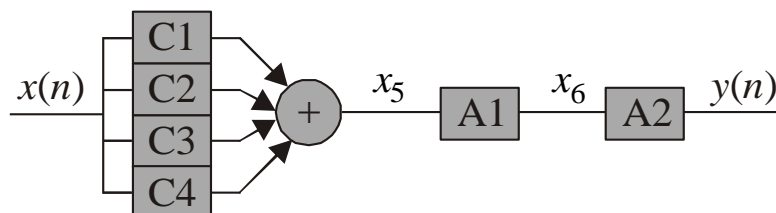


Fig. 115 Diagrama de blocos da reverberação.

As equações dos *Comb filter* IIR são dadas por:

$$y_{C1}(n) = x(n) + a_1 y(n - D_1) \quad (12)$$

$$y_{C2}(n) = x(n) + a_2 y(n - D_2) \quad (13)$$

$$y_{C3}(n) = x(n) + a_3 y(n - D_3) \quad (14)$$

$$y_{C4}(n) = x(n) + a_4 y(n - D_4). \quad (15)$$

Já as equações correspondentes aos filtros passa-tudo são dadas por:

$$y_{A1}(n) = a_5 y(n - D_5) - a_5 x_5(n) + x_5(n - D_5) \quad (16)$$

$$y_{A2}(n) = a_6 y(n - D_6) - a_6 x_6(n) + x_6(n - D_6). \quad (17)$$

#### 4.7 Wah-Wah

O efeito wah-wah é implementado na forma digital adicionando a saída de um filtro passa faixa de banda estreita com o sinal de entrada, conforme apresentado na Fig. 116. Normalmente, o filtro passa-faixa utilizado é projetado com variáveis de estado e sua frequência central é variada continuamente [18] de 300Hz a 3000Hz.

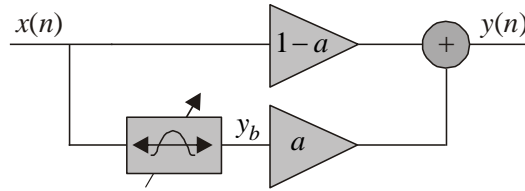


Fig. 116 Diagrama de blocos do wah-wah.

Na Fig. 117, é apresentado o diagrama de blocos do filtro com variáveis de estado.

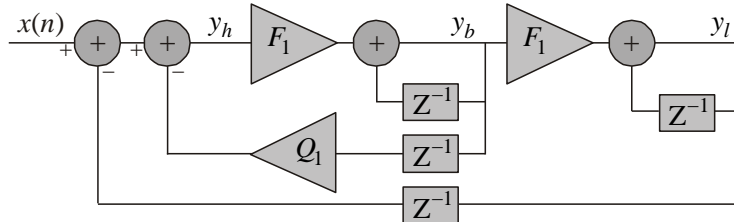


Fig. 117 Diagrama de blocos do filtro com variáveis de estado.

As equações usadas para o filtro com variáveis de estado para o PB, PF e PA são respectivamente:

$$y_l(n) = F_1 y_b(n) + y_l(n-1) \quad (18)$$

$$y_b(n) = F_1 y_h(n) + y_b(n-1) \quad (19)$$

$$y_h(n) = x(n) - y_l(n-1) - Q_1 y_b(n-1). \quad (20)$$

A equação final para a implementação do Wah-wah é:

$$y(n) = (1-a)x(n) + ay_b(n). \quad (21)$$

#### 4.8 Phaser

Diferentemente do uso de filtros passa-tudo utilizados nos circuitos analógicos, na forma digital, o efeito *phaser* é implementado adicionando a saída de um filtro *notch* com o sinal de entrada, conforme apresentado na Fig. 118.

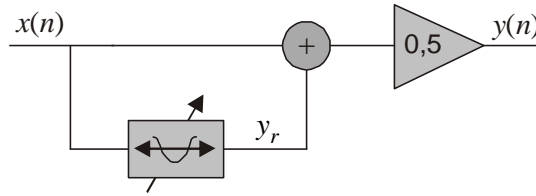


Fig. 118 Diagrama de blocos do phaser.

A função de transferência do *phaser* é dada por [18]:

$$H(z) = \frac{1}{2}[1 + A_2(z)]. \quad (22)$$

São utilizadas estruturas de segunda ordem para o filtro *notch*, cuja função de transferência é dada por [18]:

$$A_2(z) = \frac{-a + (d - da)z^{-1} + z^{-2}}{1 + (d - da)z^{-1} - az^{-2}}. \quad (23)$$

Os parâmetros “ $a$ ” e “ $d$ ” são obtidos a partir das especificações de frequência de corte ( $f_c$ ), frequência de amostragem ( $f_s$ ) e largura de banda ( $f_b$ ) do filtro por [18]:

$$a = \frac{\tan((\pi f_b / f_s) - 1)}{\tan((2\pi f_b / f_s) + 1)} \quad (24)$$

$$d = -\cos(2\pi f_c / f_s). \quad (25)$$

A equação discreta para a implementação do filtro *notch* é:

$$y_r(n) = -ax(n) + d(1-a)x(n-1) + x(n-2) - d(1-a)y_1(n-1) + ay(n-2). \quad (26)$$

$$\text{A equação final para a implementação do phaser é: } y(n) = 0.5[x(n) + y_r(n)]. \quad (27)$$

## 5 EQUALIZADOR

Diferentemente dos equalizadores analógicos, que usam  $N$  seções *Bump*, o equalizador digital é normalmente implementado por uma sequência de seções *Shelving* e *Bump*, conforme apresentado na Fig. 119. Essa estratégia de implementação evita o surgimento das oscilações indesejáveis mostradas na Fig. 88 (Parte 3 da apostila). As seções *Shelving* são utilizadas no primeiro e no último estágio do equalizador com a função de controle de graves e de agudos, respectivamente. Já as seções *bump* têm a função de controle das médias frequências.

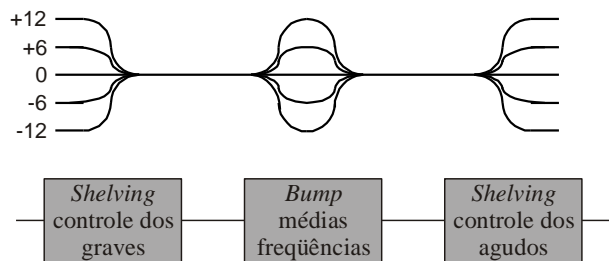


Fig. 119 Conexão entre seções shelving e bump [22].

### 5.1 Seção Shelving

As seções *Shelving* reforçam (*boost*) ou atenuam (*cut*) os componentes de baixa e alta frequência pela variação nos parâmetros de frequência de corte ( $f_c$ ) e ganho ( $H_o$ ) do filtro. As equações do *shelving* para controle dos graves e agudos de primeira ordem [22], respectivamente, são:



$$H(z) = 1 + \frac{H_o}{2} [1 + A(z)] \quad (28)$$

$$H(z) = 1 + \frac{H_o}{2} [1 - A(z)] \quad (29)$$

onde  $H_o$  é o ganho e  $A(z)$  é a função de transferência de um filtro passa-tudo de primeira ordem dada por:

$$A(z) = \frac{z^{-1} + a}{1 + az^{-1}}. \quad (30)$$

O parâmetro “ $a$ ” é obtido a partir da especificação da frequência de corte ( $f_c$ ) e da frequência de amostragem ( $f_s$ ). As equações do parâmetro “ $a$ ” para o controle das baixas frequências são dadas por [15]:

$$a = \frac{\tan((\pi f_c / f_s) - 1)}{\tan((\pi f_c / f_s) + 1)} [Boost] \quad (31)$$

$$a = \frac{\tan((\pi f_c / f_s) - V_o)}{\tan((\pi f_c / f_s) + V_o)} [Cut]. \quad (32)$$

As equações de “ $a$ ” para o controle das altas frequências são [22]:

$$a = \frac{\tan(\pi f_c / f_s) - 1}{\tan(\pi f_c / f_s) + 1} [Boost] \quad (33)$$

$$a = \frac{V_o \tan(\pi f_c / f_s) - 1}{V_o \tan(\pi f_c / f_s) + 1} [Cut] \quad (34)$$

onde  $V_o$  é obtido a partir da especificação do ganho em dB por:

$$V_o = 10^{G/20} \quad (35)$$

$$H_o = V_o - 1. \quad (36)$$

O diagrama de blocos para a implementação do *shelving* é mostrado na Fig. 118.

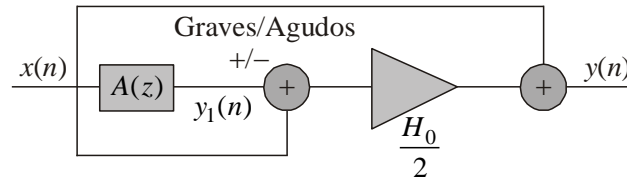


Fig. 118 Diagrama para a implementação do shelving [18].

A equação discreta para a implementação do filtro passa-tudo é dada por:

$$y_1(n) = ax(n) + x(n-1) - ay_1(n-1). \quad (37)$$

Já a equação de saída do *shelving* (controle de graves/controle de agudos) é dada por:

$$y(n) = \frac{H_o}{2} [x(n) \pm y_1(n)] + x(n). \quad (38)$$

Caso os requisitos do *shelving* exijam que o filtro passa-tudo seja de segunda ordem, podem ser usados os procedimentos apresentados em [18].

## 5.1 Seção Bump (Peak)

Conforme apresentado na Fig. 81, as seções *bump* atuam sobre as médias frequências, variando os parâmetros de frequência de corte ( $f_c$ ), largura de banda ( $f_b$ ) e ganho ( $H_o$ ). São utilizadas estruturas de segunda ordem cuja FT [22] é dada por:

$$H(z) = 1 + \frac{H_o}{2} [1 - A_2(z)] \quad (39)$$

onde  $A_2(z)$  é a função de transferência de um filtro passa-tudo de segunda ordem dada por:

$$A_2(z) = \frac{-a + (d - da)z^{-1} + z^{-2}}{1 + (d - da)z^{-1} - az^{-2}}. \quad (40)$$

Os parâmetros “ $a$ ” e “ $d$ ” são obtidos a partir das especificações de frequência de corte ( $f_c$ ), frequência de amostragem ( $f_s$ ), ganho ( $H_o$ ) e largura de banda ( $f_b$ ) do filtro, conforme detalhado em [18]. O diagrama de blocos é mostrado na Fig. 119.

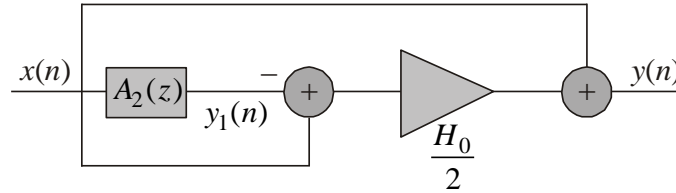


Fig. 119 Diagrama de blocos para a implementação do bump[11].

A equação discreta para a implementação do passa-tudo é dada por:

$$\begin{aligned} y_1(n) = & -ax(n) + d(1-a)x(n-1) + x(n-2) \\ & -d(1-a)y_1(n-1) + ay(n-2). \end{aligned} \quad (41)$$

Já a equação de saída da seção *peak* é dada por:

$$y(n) = \frac{H_o}{2} [x(n) - y_1(n)] + x(n). \quad (42)$$

## 6 RESULTADOS EXPERIMENTAIS

Os algoritmos aqui considerados foram implementados no Matlab e no ambiente do VisualDSP. Nos testes realizados, cada algoritmo foi submetido a arquivos de testes, gerando os correspondentes arquivos de saída. Os arquivos de testes e os algoritmos implementados encontram-se disponíveis para download no endereço apresentado em [23]. Deve-se observar que os arquivos foram gerados para uma frequência de amostragem de 8 kHz e 16bits de codificação e que podem ser manuseados pela ferramenta de edição de áudio ocenaudio [24].

## 7 REFERÊNCIAS BIBLIOGRÁFICAS

- [1] Hunter, D.; “Guitar Effects Pedals the Practical handbook”, Backbeat Book, 2004.
- [2] Loy, G ; “Musimathics- Volumes 1 e 2”, MIT Press, 2007.
- [3] Anderton, C.; “Eletronic Projects for Musicians”, Amsco Publications, 1980.
- [4] Noceti Filho, S.; “Filtros Seletores de Sinais”, Editora UFSC, 2010.
- [5] Time –Domain Synthesis of Linear Networks, Kendall. SU, Prentice-Hall, New Jersey, 1971
- [6] K. J. Gundry, “Constant-Q Graphic Equalizers,” *Journal of Audio Engineering Society*, vol. 34, no. 9, pp. 1-16, set. 1986
- [7] K. J. Gundry, *Adjustable Equalizers Useable in Audio Spectrum*, U.S. Patent 3.921.104, nov. 1975.
- [8] [www.analog.com](http://www.analog.com)
- [9] <http://www.analog.com/en/embedded-processing-dsp/blackfin/ADSP-BF533/processors/product.html>
- [10] [http://www.analog.com/en/embedded-processing-dsp/software-and-reference-designs/content/visualdsp\\_software\\_test\\_drive/fca.html](http://www.analog.com/en/embedded-processing-dsp/software-and-reference-designs/content/visualdsp_software_test_drive/fca.html)
- [11] ADSP-BF533 EZ-KIT Lite “Evaluation System manual” – Analog Devices.
- [12] VisuaDSP++ C/C++ Compiler and Library Manual for Blackfin Processors – Analog Devices.
- [13] VisualDSP++ Assembler and Preprocessor Manual – Analog Devices.
- [14] Orfanidis, S. J.; “Introduction to Signal processing”, Prentice Hall, 2009.
- [15] Bloom, P. J.; “High-Quality Digital Audio in the Entertainment Industry: An Overview of Achievements and Challenges”, *IEEE ASSP.Mag*, 2, October 1985.
- [16] Oppenheim, A., V.; “Applications of Digital Signal Processing”, , Prentice Hall, 1978.
- [17] Tomarakos, J.; Ledger, D.; “Using the Low-cost, High Performance ADSP-21065L Digital Signal Processor for Digital Audio Applications”, Analog Devices Applications, April 1998.
- [18] Zolzer, U.; “DAFX: Digital Audio Effects” John Wiley & Sons, Ltd – 2002.
- [19] Dattorro, J; “Effect Design, part 2: Delay-Line Modulation and Chorus” *J. Audio Eng. Soc*, October 1997.
- [20] Schroeder; M. R., “Natural Sounding Artificial reverberation” *J. Audio Eng. Soc*, 10; 1962.
- [21] Moorer; J. A.; “About This Reverberation Business”, 1979.
- [22] Zolzer, U; “Digital Audio Signal Processing” John Wiley & Sons, Ltd – 2008.
- [23] <http://www.linse.ufsc.br/research-public>.
- [24] <http://www.ocenaudio.com.br>