DOI: 10.48011/asba.v2i1.1371

Sistema de identificação automática de características de filtros analógicos e substituição por filtros digitais implementados em SoC

Felipe Arnhold * **Jean Schmith** **
Rodrigo M. de Figueiredo ***

* *Faculdade de Engenharia da Computação, Universidade do Vale do Rio dos Sinos, RS (e-mail: farnhold@edu.unisinos.br).*
** *Faculdade de Engenharia Elétrica, Universidade do Vale do Rio dos Sinos, RS (e-mail: jschmith@unisinos.br)*
*** *Faculdade de Engenharia de Controle e Automação, Universidade do Vale do Rio dos Sinos, RS (e-mail: marquesf@unisinos.br)*

Abstract: Filters are the base of signal processing and nowadays digital filters are well-implemented in many different areas. Oftentimes, model an analog filter as a digital one can be very hard and time consuming. For this reason, this paper focuses on an implementation of an automatic system capable to copy the behavior of an analog filter. We used a System-on-Chip (SoC) composed by a Field Programmable Gate Array (FPGA) and a micro controller Arm to develop the system. The algorithm is capable of identifying which filter is in the black box (band-pass, band-stop, low-pass or high-pass), extract the characteristics and determine the parameters of an Infinite Impulse Response (IIR). We implemented classical equations to IIR filters and measurements on the SoC. The frequency response of both filter, analog and modeled, were compared to analyse the accuracy of modeled one. The system presented very low error (less than 3.3%) for frequencies parameters estimation, except for band-stop filter that presented 8.57%.

Resumo: Filtros são a base do processamento de sinais e, atualmente, os filtros digitais são amplamente aplicados em diversas áreas. Muitas vezes, a modelagem de um filtro analógico como um filtro digital pode ser muito complexo e demorado. Por essa razão, este trabalho foca na implementação de um sistema automático capaz de copiar o comportamento de um filtro analógico. Para o desenvolvimento do sistema, foi utilizado um *System-on-Chip* (SoC), composto por um *hardware* programável (FPGA) e um microcontrolador ARM. O algoritmo é capaz de identificar o tipo do filtro (passa banda, rejeita banda, passa baixa ou passa alta), extrair as características e determinar os parâmetros de um filtro de resposta infinito ao impulso (IIR) de segunda ordem. Nós implementamos equações clássicas do filtro IIR assim como as medidas no SoC. A resposta em frequência dos dois filtros, analógico e modelado, foram comparados para analisar a exatidão do modelado. O sistema apresentou um erro bem baixo (menos de 3,3%) na resposta em frequência para a estimativa dos parâmetros, exceto para o filtro rejeita banda que apresentou um erro de 8,57%.

Keywords: SoC; Digital Filter; IIR; Filter Identification; DSP; FPGA; ARM
Palavras-chaves: SoC; Filtros Digitais; IIR; Identificação de Filtros; DSP; FPGA; ARM.

1. INTRODUÇÃO

Filtros digitais são objetos de estudo desde a viabilidade dos sistemas digitais na década de 1960. Atualmente, estes filtros estão presentes em diversas aplicações, desde sistemas de comunicações até o processamento de sinais biomédicos. Um dos filtros mais conhecido é o de Resposta Infinita ao Impulso (IIR, do inglês *Inifinite Impulse Response*). Uma de suas aplicações é a modelagem de sistemas reais, através de funções de transferência. Comparado ao filtro de Resposta Finita ao Impulso (FIR, do inglês *Finite Impulse Response*), o IIR possui um número reduzido de coeficientes e tem melhor performance, conforme Banerjee and Sinha (2009) e ARFIA et al. (2009).

Os FPGAs (do inglês *Field Programmable Gate Array* e em português Arranjo de Portas Programáveis) são frequentemente utilizados para a implementação de filtros digitais, conforme Macpherson and Stewart (2007), Mehra (2011) e Gawande and Khanchandani (2015). Como podem ser reconfigurados, os FPGAs oferecem uma completa customização de hardware para o desenvolvimento de várias aplicações de processamento digital de sinais (DSP, do inglês *Digital Signal Processing*)(Mustafa et al., 2009),

⋆ Os autores agradecem a Xilinx pela doação da Zedboard.

incluindo tempo real (Rakic et al., 2015) ou simulações (Song et al., 2004).

Filtros analógicos de áudio são bons exemplos de sistemas que podem ser analiticamente modelados por um filtro IIR e implementados em um FPGA. Entretanto, determinar os coeficientes do IIR que melhor represente um filtro analógico é algo trabalhoso. Com equipamento adequado, pode ser fácil determiná-los, mas pode demandar muitas horas de trabalho. Dessa forma, um sistema simples para medição e clonagem de filtros analógicos apresenta uma grande variedade de aplicações.

Esse trabalho apresenta o desenvolvimento de um sistema capaz de analisar um filtro de áudio analógico e representá-lo digitalmente através de um filtro IIR de segunda ordem implementado em um *System-on-Chip* (SoC). O sistema proposto identifica quatro filtros de segunda ordem diferentes: passa baixas, passa altas, passa banda e rejeita banda. O SoC utilizado é o Zync 7020, da *Xilinx*. Ele é composto por um processador ARM e um FPGA. Para validação dos sistemas, foram modelados quatro filtros analógicos, um de cada tipo, e então a resposta em frequência de ambos (original e modelado) foram comparados para obter o erro absoluto médio.

2. DESCRIÇÃO DO ALGORITMO

O diagrama de blocos da técnica desenvolvida está representado na Fig. 1. Na parte de cima da Fig. 1 o sistema proposto inicia conectado em paralelo com o filtro analógico a ser analisado e modelado. O sistema implementado no SoC insere um sinal conhecido na entrada e faz a leitura dos dados na saída do filtro analógico. Na parte de baixo, o filtro analógico em análise é retirado, e o SoC assume a sua função. O sistema proposto implementado no SoC substitui o filtro analógico, calculando uma saída próxima a do analógico para uma mesma entrada.

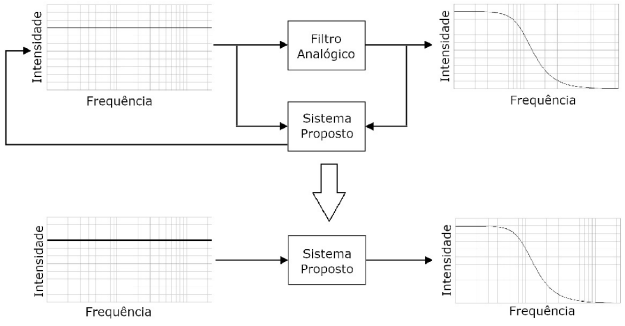

Figura 1. Diagrama de bloco do sistema proposto.

O sinal pode sofrer distúrbios devido as diferenças nas impedâncias de saída do SoC e entrada do filtro analógico. Para minimizar este problema, o SoC insere um sinal no filtro analógico e após faz a sua leitura. As leituras de entrada e saída do filtro analógico são utilizados para a modelagem digital.

2.1 Análise do filtro analógico

Para obter as características do filtro analógico, o sistema proposto gera um sinal de áudio e aplica na entrada do filtro analógico. O sinal aplicado é um sweep de 20 Hz a 20 kHz, com passos de 10 Hz, durante dois ciclos completos. Esse sinal apresenta um espectro em frequência *flat*, ou seja, possui a mesma amplitude em todas as frequências. Para cada incremento de frequência do sinal de entrada, o sistema faz a leitura do sinal de saída do filtro analógico. Através da equação de ganho (1), a resposta em frequência do filtro analógico é obtida, onde, $y[f]$ é a amplitude da saída do filtro na frequência f, enquanto que $x[f]$ é a amplitude da entrada, para a mesma frequência. O valor máximo de ganho apresentado durante a aplicação do sinal *sweep* é considerado o ganho do filtro analógico.

$$G[f]_{dB} = 20 \cdot log_{10}\left(\frac{y[f]}{x[f]}\right), 20 \leq f \leq 20000 \qquad (1)$$

Os filtros de segunda ordem apresentam frequência de corte (f_c) quando a resposta cai $-3dB$ em relação ao ganho máximo do filtro. O processo descrito pode ser observado na Fig. 2. No bloco "Encontrar Fc", o sinal lido gerado pelo *sweep* é analisado, e a frequência de corte é o ponto em que o ganho é $3dB$ menor que o ganho máximo encontrado. A caracterização do sistema depende de quantas frequências de corte são encontradas e em que parte do espectro elas aparecem, sendo possível identificar o tipo do filtro (passa baixas, passa altas, passa banda ou rejeita banda). A identificação de quantas frequências de corte o filtro possui e a posição delas no espectro de frequências segue uma lógica. Para o caso do filtro passa baixas, será identificado apenas uma frequência de corte no ponto onde a resposta de $-3dB$ aparece depois do ponto identificado com o ganho máximo do filtro. Para o filtro passa altas, o ponto onde a resposta decai em $-3dB$ aparece antes do ponto do ganho máximo do filtro. Se o filtro em análise é um passa banda, serão identificados duas frequências de corte, uma antes e outra depois do ganho máximo do filtro. Por último, em um filtro rejeita banda serão identificadas duas frequências de cortes, ambas entre o ganho máximo do filtro. Esta parametrização está resumida na tabela 1. Filtros com mais de duas frequências de corte ou não lineares estão fora do escopo deste projeto.

Tabela 1. Tipo do filtro relacionado ao ganho e frequência de corte.

.	Número de f_c	f_c em relação ao ganho máximo
Passa baixas	1	Depois
Passa altas	1	Antes
Passa banda	2	Antes e Depois
Rejeita banda	2	Entre

A partir das medições e com as características obtidas, os coeficientes do filtro IIR de segunda ordem podem ser calculados.

2.2 Construção do filtro IIR

De acordo com Hamilton (1834), qualquer sistema pode ser descrito com um conjunto de equações diferenciais de segunda ordem. Portanto, a metodologia é baseada em um filtro digital IIR de segunda ordem. A equação (2) implementa um filtro digital IIR, onde y representa a sequência de pontos da saída, x representa a sequência de

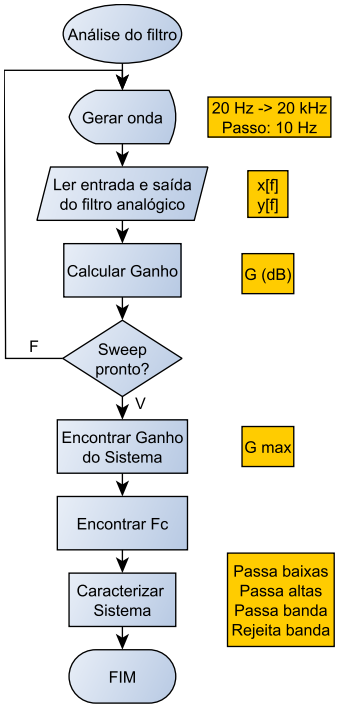

Figura 2. Fluxograma da primeira etapa de análise do filtro analógico.

pontos de entrada, n é a variável temporal e b_0, b_1, b_2, a_1 e a_2 são os coeficientes do filtro IIR.

$$y[n] = b_0x[n] + b_1x[n-1] + b_2.x[n-2] \\ -a_1y[n-1] - a_2y[n-2] \quad (2)$$

A tabela 2 apresenta as equações utilizadas para calcular os coeficientes para cada tipo de filtro, conforme Zölzer (2011). De acordo com estas equações, os parâmetros necessários para o cálculo são o tipo do filtro, a(s) frequência(s) de corte e para os filtros passa banda e rejeita banda, a largura de banda também é necessária.

Tabela 2. Equações dos coeficientes para cada tipo de filtro. Os valores de Q e K são representados pelas equações (3) e (4), respectivamente.

.	Passa baixa	Passa alta	Passa banda	Rejeita banda
b_0	$\frac{K^2Q}{K^2Q+K+Q}$	$\frac{Q}{K^2Q+K+Q}$	$\frac{K}{K^2Q+K+Q}$	$\frac{Q(1+K^2)}{K^2Q+K+Q}$
b_1	$\frac{2K^2Q}{K^2Q+K+Q}$	$-\frac{2Q}{K^2Q+K+Q}$	0	$\frac{2Q(K^2-1)}{K^2Q+K+Q}$
b_2	$\frac{K^2Q}{K^2Q+K+Q}$	$\frac{Q}{K^2Q+K+Q}$	$-\frac{K}{K^2Q+K+Q}$	$\frac{Q(1+K^2)}{K^2Q+K+Q}$
a_1	$\frac{2Q(K^2-1)}{K^2Q+K+Q}$	$\frac{2Q(K^2-1)}{K^2Q+K+Q}$	$\frac{2Q(K^2-1)}{K^2Q+K+Q}$	$\frac{2Q(K^2-1)}{K^2Q+K+Q}$
a_2	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$	$\frac{K^2Q-K+Q}{K^2Q+K+Q}$

O cálculo de Q é dependente do tipo de filtro. Para o passa baixas e passa altas, $Q = \frac{1}{\sqrt{2}}$. Para o passa banda e o rejeita banda, Q é determinado pela equação (3), onde f_{cc} é a frequência central de corte e f_b é a largura de banda entre as frequências de corte.

$$Q = \frac{f_{cc}}{f_b} \quad (3)$$

O valor de K é determinado pela equação (4), onde f é a frequência de corte (f_c) para os filtros passa baixas e passa altas ou a frequência central de corte (f_{cc}) para os filtros passa banda e rejeita banda. A frequência de amostragem é representada por f_s.

$$K = tan(\pi\frac{f}{f_s}) \quad (4)$$

3. IMPLEMENTAÇÃO

O projeto foi implementado em uma placa de desenvolvimento ZedBoard contendo um *System-on-Chip* (SoC), composto por um processador e uma unidade lógica programável. O processo de implementação consiste em desenvolver o algoritmo descrito na plataforma selecionada, particionando-o entre hardware e software.

3.1 Materiais e características do sistema

A placa de desenvolvimento ZedBoard possui o SoC *Zync 7020* da Xilinx. De acordo com Crockett et al. (2014), este circuito integrado é um sistema heterogêneo composto por um processador ARM Cortex-A9 (PS) e uma unidade lógica programável (PL), equivalente a um FPGA Artix-7. Um barramento é utilizado para a comunicação entre o PS e o PL. O processador possui um *clock* de $866MHz$ e o PL um *clock* de $100MHz$. Os softwares utilizados para o desenvolvimento foram o Vivado para a unidade lógica e o SDK para o processador.

A captura e escrita do áudio é realizada através do *codec* de áudio ADAU1761, presente na ZedBoard. O *codec* conta com um conversor analógico-digital (ADC) e um conversor digital-analógico (DAC), ambos estéreo (2 canais). A resolução dos conversores é de 24 bits, com frequência de amostragem configurada em $48kHz$. O sinal digital de áudio é transmitido através do protocolo I_2S e formatado em ponto fixo 1.23.

3.2 Particionamento Hardware/Software

No desenvolvimento de sistemas com SoC, um dos estágios é a partição entre o hardware e software, ou seja, determinar que parte deve ser executado no processador (PS) e que parte será executada em hardware (PL) Crockett et al. (2014). A Fig. 3 apresenta o sistema desenvolvido particionado entre o PL e PS. O programa principal está presente no PS, enquanto o PL é responsável pelo processamento do áudio, desde a captura pelo ADC até a escrita no DAC.

A comunicação entre o PL e o PS se dá por um barramento existente dentro do SoC. Os sinais de controle e demais dados compartilhados entre as partes são transferidos por esse barramento.

3.3 Parte do processador (Software)

O processador é responsável pelo controle de todo o sistema, assim como pela interface ao usuário. As principais

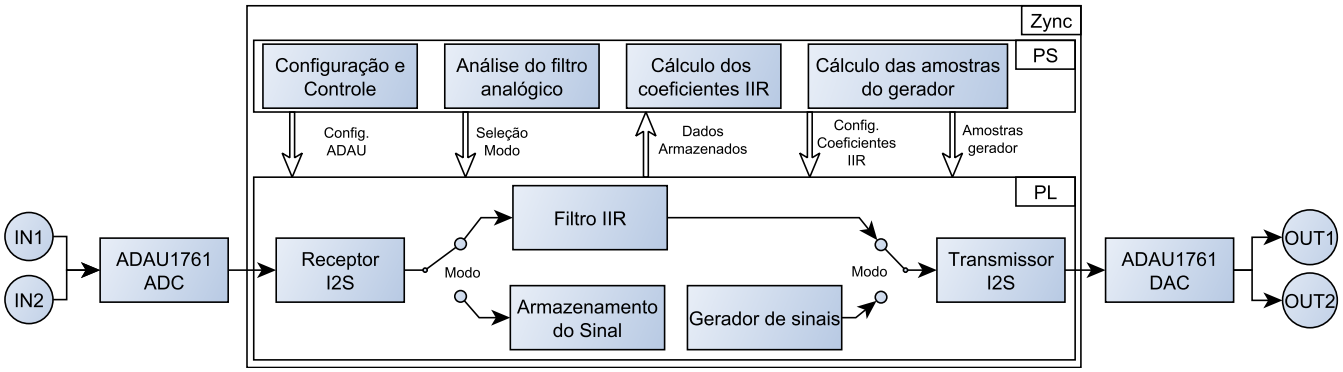

Figura 3. Fluxograma do sistema divido entre PS e PL.

funções executadas pelo processador estão descritas na Fig. 3. O bloco de configuração e controle é responsável tanto pela inicialização do sistema, como a configuração do *codec* de áudio através do protocolo SPI. Esse controle abrange a seleção do modo de operação em que o sistema está (análise do filtro e reprodução do filtro modelado) e o controle na passagem de dados entre o PS e PL. Na etapa de reprodução, o processador não possui função, portanto, os outros três blocos apresentados na Fig. 3 pertencem a primeira etapa de análise do filtro analógico. O bloco intitulado "Análise do filtro analógico"é responsável pelo algoritmo apresentado na Fig. 2. Esse bloco faz uso de outros dois. Um para o cálculo das amostras para a geração do sinal senoidal de 20 Hz a 20 kHz e o outro para o cálculo dos coeficientes IIR. Estes cálculos são facilitados pelo uso de bibliotecas matemáticas, motivo para mantê-las em software. O processador calcula e transfere à parte lógica para realizar de fato o processamento do áudio. O dado entregue pelo *codec* é no formato de ponto fixo 1.23, entretanto tanto os cálculos dos coeficientes quanto das amostras para o gerador de sinais são realizadas em ponto flutuante.

3.4 Parte lógica (Hardware)

A parte em hardware é responsável pelo processamento do sinal de áudio, lendo do *codec* através do protocolo I_2S, processando em um filtro IIR e escrevendo no DAC do *codec*. De acordo com a Fig. 3, o sinal de áudio é inserido através dos conectores no codec de áudio. O conversor analógico-digital presente no codec transforma o áudio em um *stream* de bits através do protocolo I^2S. Esse sinal é lido pelo SoC através do bloco "Receptor I2S". Dependendo do modo que o sistema estiver operando (análise do filtro analógico ou reprodução do filtro modelado), o sinal de áudio possui um fluxo diferente. Na etapa de análise do filtro analógico, os dois canais do ADC são lidos e direcionados para uma memória, construída com blocos de memória RAM (BRAM). O tamanho necessário desta memória, em kB, é descrita pela equação (5). O cálculo relaciona o número de amostras necessárias para se obter um período de uma determinada frequência com o tamanho de cada amostra. O número de amostras é dada pela relação da frequência de amostragem (F_s) e a frequência do sinal medido (f_{min}). A frequência de amostragem é fixa ($48\ kHz$) e quanto menor a frequência do sinal, maior a quantidade de pontos. Dessa forma, como o tamanho da BRAM deve ser fixo, deve-se utilizar o maior tamanho necessário, resolvendo em f_{min} igual a $20\ Hz$. O tamanho da amostra ($tam_{amostra}$) é de 32 bits e o número de períodos (n) utilizado é dois. A divisão por 8000 é para converter de bits para kB. Então, o tamanho utilizado para a BRAM é de 4800 posições, com $19,2\ kB$.

$$Tamanho[kB] = \frac{F_s}{f_{min} \cdot 8000} \cdot tam_{amostra} \cdot n \qquad (5)$$

Em paralelo a isso, o gerador de sinais também possui uma memória BRAM de mesmo tamanho para gerar o sinal senoidal na saída. As amostras geradas pelo processador são armazenadas nessa memória e o PL lê uma amostra e escreve no DAC do codec de áudio, através do transmissor I^2S. A leitura é feita na frequência de amostragem, garantindo a integridade do sinal.

Visto que os coeficientes e amostras calculados estão no formato de ponto flutuante de precisão simples, o sinal de áudio deve ser convertido do ponto fixo para *float* (ponto flutuante de precisão simples [32 bits]). Essa operação está implícita na função Receptor I2S, presente na Fig. 3. O mesmo ocorre para a conversão do ponto flutuante para ponto fixo na função de transmissão I2S.

No modo de reprodução do filtro modelado, o sinal de áudio passa pelo bloco de processamento, contendo o filtro IIR. O filtro IIR foi construído utilizando os módulos específicos para processamento digital de sinais (blocos de DSP) disponíveis no *Zync*. De acordo com a equação do IIR de segunda ordem (2), são necessárias apenas duas operações, soma e multiplicação, ambas disponíveis no software de desenvolvimento *Vivado*. Para evitar a subtração, os coeficientes a_1 e a_2 tem seu sinal invertido ainda no processador, tornando assim uma soma. São necessários dois valores, o de entrada ($x[n]$) e de saída ($y[n]$). Eles são facilmente obtidos adicionando dois *flip flop* do tipo D, sendo atualizados a cada nova amostra do sinal de áudio.

O protocolo I^2S é caracterizado por ser um *streaming* (Philips, 1996). O ADAU1761 envia dados continuamente, através da saída do ADC, independente se existe algum dispositivo lendo esses dados. O mesmo ocorre para a escrita do dado no DAC do *codec*. Para esse tipo de dado, o *Zync* possui um modo de transferência de dados na mesma modalidade de *stream*, onde pode haver apenas um transmissor e um receptor, chamado *AXI-Stream*. Dessa forma, um bloco só pode processar um dado após o

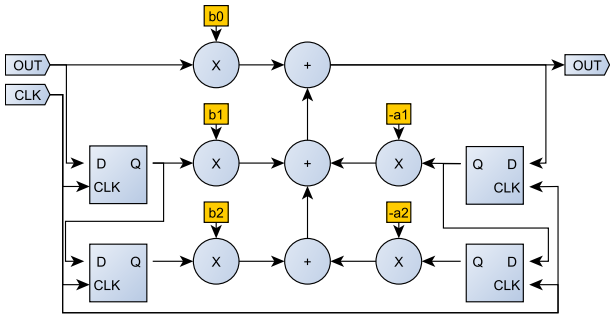

Figura 4. Fluxograma do sistema divido entre PS e PL.

próximo já ter lido o dado atual. Por exemplo, o receptor I^2S só captura uma nova amostra do ADC após o filtro IIR ter lido a amostra atual do receptor. Como o processo de transferência e processamento é muito mais rápido que o período de amostragem, o dado está disponível para transmitir antes da realização da captura de uma nova amostra.

4. TESTES E RESULTADOS

Para validação do sistema, foram construídos um filtro analógico para cada tipo. Para filtros dos tipos passa baixas (PB), passa altas (PA) e passa banda (PF), foi utilizado a aproximação de Butterworth de segunda ordem. Já para o rejeita banda (RF), foi utilizado um passa baixas e um passa altas de primeira ordem seguido por um somador. As características destes filtros analógicos estão presentes na tabela 3, onde f_c é a frequência de corte ou frequência central, G é o ganho e w_b a largura de banda dos filtros PF e RF.

Tabela 3. Características dos filtros analógicos para validação

	f_c (Hz)	G (dB)	w_b (Hz)
PB	1600	$6,17$	–
PA	2140	$5,43$	–
PF	1940	$6,34$	480
RF	370	0	370

O método de avaliação utilizado foi o erro médio absoluto percentual de todas as amostras e o erro absoluto em função das características. O erro médio absoluto percentual é descrito pela equação (6). O erro de cada amostra é calculado e então é feito a média dos erros em relação ao número total de amostras (N). A análise foi feita de 20 a 20 kHz, com passo de 10 Hz, totalizando 1999 amostras. Esse erro é dado pela diferença absoluta entre o ganho modelado ($\hat{y_i}$) e o ganho analógico (y_i). O erro sobre as características é calculado da mesma forma, através do erro absoluto relacionado aos valores.

$$Erro = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y_i}|}{|y_i|}\cdot 100\% \qquad (6)$$

O sistema calcula um conjunto de coeficientes diferente para cada filtro analógico em análise. Os valores dos coeficientes calculados para a equação 2 dos filtros digitais modelados a partir dos filtros analógicos estão descritos na tabela 4.

Tabela 4. Coeficientes calculados pelo sistema para cada tipo de filtro analógico em teste.

.	Passa baixa	Passa alta	Passa banda	Rejeita banda
b_0	0.009862	0.814854	0.030141	0.952633
b_1	0.019723	-1.62971	0	-1.90303
b_2	0.009862	0.814854	-0.03014	0.952633
a_1	-1.70009	-1.61102	-1.88657	-1.90303
a_2	0.739538	0.664286	0.939718	0.905266

4.1 Filtro passa baixas

As respostas em frequência dos filtros passa baixas analógico e modelado estão apresentadas na Fig. 5. Graficamente, o filtro modelado apresentou resultados satisfatórios. Em frequências perto do limite do ADC somado ao fato da atenuação do sinal ser muito alta, o filtro analógico apresenta certa instabilidade. O erro absoluto médio percentual é de $18,70\%$, justificado pela instabilidade no final da banda passante para sinais extremamente baixos. Para valores de frequência até 2000 Hz, o filtro teve um comportamento estável. Com frequência de corte em 1630 Hz, o erro foi de $1,88\%$, enquanto o ganho de $6,41$ dB teve erro de $3,89\%$. Para a maioria das aplicações esses valores são aceitáveis.

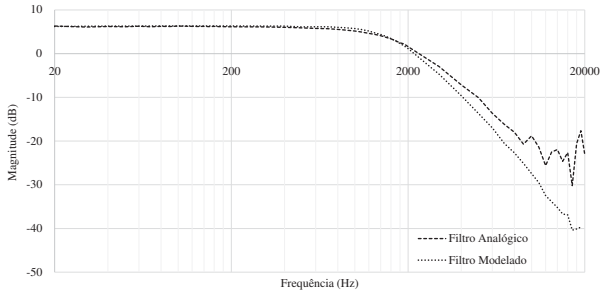

Figura 5. Resposta em frequência dos filtros passa baixas analógico e modelado.

4.2 Filtro passa altas

Diferente do passa baixas, o passa altas tem alta atenuação em frequências baixas e não apresentou a instabilidade significativa, como mostra o gráfico da Fig. 6. A frequência de corte obtida no sistema modelado é de 2210 Hz, obtendo um erro de $3,27\%$. O ganho, medido em $6,3$ dB, teve erro de $16,02\%$. Esse filtro apresentou erros bem mais significativos comparado ao passa baixas. O erro médio absoluto de $24,87\%$ mais uma vez é explicado pela grande diferença dada em níveis de sinal muito baixos, entre $20Hz$ e $400Hz$ com atenuações maiores que $-20dB$.

4.3 Filtro passa banda

Esse filtro teve a melhor resposta entre os testados. Três características são abordadas. Apresentado na Fig. 7, a frequência central (f_{cc}) obtida no filtro modelado é de 1940 Hz, com erro de apenas $0,26\%$. A largura de banda (w_b) medida foi exatamente a mesma do filtro analógico, no valor de 480 Hz, o que consequentemente resulta em $0,00\%$ de erro. O ganho também teve resultado muito satisfatório, no valor de $6,3$ dB e erro de $0,63\%$. O erro médio absoluto, medido em $11,51\%$, é o menor adquirido e as regiões onde a diferença entre o filtro analógico e modelado

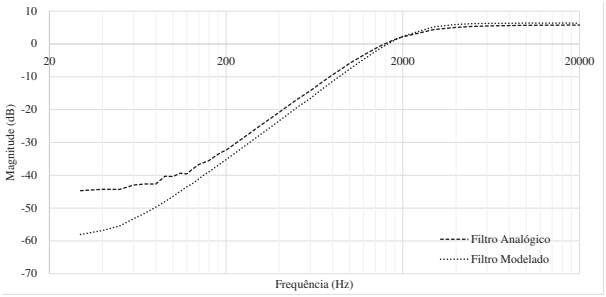

Figura 6. Resposta em frequência dos filtros passa altas analógico e modelado.

são mais notáveis são as regiões com grandes atenuações. Quando a atenuação é muito alta, neste caso a partir de $-20dB$, o sinal possui amplitude extremamente pequena, causando instabilidades e consequentemente aumentando o erro médio absoluto.

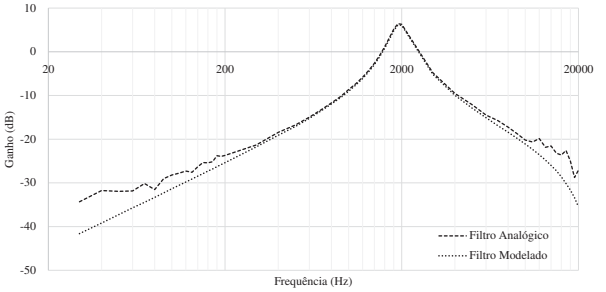

Figura 7. Resposta em frequência dos filtros passa banda analógico e modelado.

4.4 Filtro rejeita faixa

Para esse tipo foi usado uma combinação de dois filtros, um passa altas e um passa baixas de primeira ordem. Dessa forma, foge da especificação de modelagem de filtros de segunda ordem e por esse motivo apresenta erros mais significativos. Nas respostas em frequência apresentadas na Fig. 8, fica claro a diferença entre os filtros analógico e modelado. O erro médio absoluto ultrapassa os 336%, justamente pela grande diferença na atenuação realizada pelo filtro modelado. Entretanto o erro sobre a largura de banda ficou em $8,57\%$, no valor de 760 Hz e frequência central exatamente igual, 370 Hz e erro de $0,00\%$. O ganho ficou em $0,66$ dB, mas não é possível obter o erro absoluto percentual em vista que o ganho no filtro analógico é igual a zero.

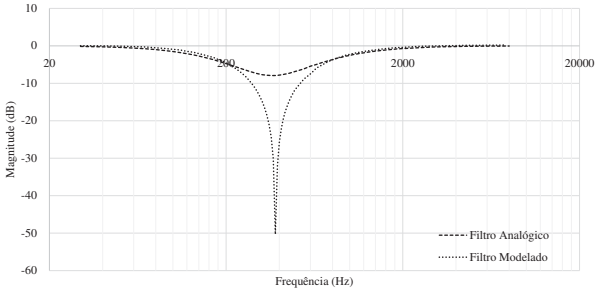

Figura 8. Resposta em frequência dos filtros rejeita banda analógico e modelado.

4.5 Tempo de execução e recursos do SoC

A utilização da lógica programável permite alta paralelização de funções, tornando o sistema proposto ideal para ser implementado nesses dispositivos. Com a paralelização das funções de soma e multiplicação dentro do filtro IIR, uma amostra precisa de aproximadamente 15 pulsos de clock para processar uma amostra do áudio. O processo de análise do filtro analógico demora cerca de 3 min, englobando a geração do *sweep*, determinação do tipo e cálculo dos coeficientes do filtro IIR.

A implementação do sistema proposto utilizou o 64% dos blocos memória RAM (BRAM) do SoC. Destes, $14,28\%$ são referentes aos blocos de armazenamento de sinal e geração do sinal *sweep*. A memória restante é usada para a comunicação entre a parte lógica e o processador. Ainda, 28% dos blocos de DSP foram utilizados e 14% das LUT (do inglês *Look Up Table*).

5. CONCLUSÃO E TRABALHOS FUTUROS

Foi desenvolvido um sistema automático implementado em um SoC, capaz de analisar e modelar digitalmente quatro tipos diferentes de filtros analógicos. Após a modelagem, o sistema implementado no SoC pode ser substituído pelo filtro analógico original. Os filtros testados foram: passa baixas, passa altas, passa banda e o rejeita banda de segunda ordem. O equacionamento de filtro digital utilizado como base é o IIR, cujos coeficientes foram calculados através de equações clássicas específicas para o tipo de filtro analisado. O sistema foi testado utilizando quatro filtros analógicos e tendo as respostas em frequência comparadas. Os filtros passa altas, passa baixas e passa banda apresentaram resultados satisfatórios, com erros relativamente baixos. Os filtros modelados divergiam dos analógicos em regiões onde as atenuações eram muito altas e com níveis de amplitude de sinais muito baixos, provavelmente abaixo da sensibilidade mínima do ouvido humano.

Este trabalho expôs ainda as limitações do sistema através do teste do filtro rejeita banda. O filtro analógico utilizado foi modelado a partir de dois primeira ordem e um somador, logo a atenuação apresentada na frequência central é bem menor do que seria apresentado em um filtro rejeita banda de segunda ordem. Visto que o filtro IIR utilizado é de segunda ordem, o erro absoluto médio esperado era grande.

Este sistema pode ser expandido para que vários filtros de segunda ordem possam ser utilizados concomitantemente, expandindo largamente os tipos de filtros suportados. Ainda, o sistema pode ser facilmente adaptado para outros tipos de sinais, dependendo apenas das limitações dos conversores A/D e D/A. Este sistema apresenta diversos campos de aplicações, como em processamento de imagens, sistemas de comunicações, controle e outras em que os filtros não são a parte importante da pesquisa e os pesquisadores necessitam de um sistema simples, rápido e eficiente para a condução de seus trabalhos.

AGRADECIMENTOS

Os autores agradecem a Xilinx pela doação da placa de desenvolvimento ZedBoard.

REFERÊNCIAS

ARFIA, F., Messaoud, M., and Mohamed, A. (2009). Nonlinear adaptive filters based on particle swarm optimization. *Leonardo Journal of Sciences*, 8.

Banerjee, S. and Sinha, A. (2009). Performance analysis of different dsp algorithms on advanced microcontroller and fpga. 609 – 613.

Crockett, L.H. et al. (2014). *The Zynq Book. Embedded Processing with the ARM Cortex-A9 on the Xilinx Zynq-7000 All Programmable SoC.* Strathclyde Academic Media.

Gawande, G.S. and Khanchandani, K. (2015). Efficient design and fpga implementation of digital filter for audio application. 906–910.

Hamilton, W. (1834). On a general method in dynamics. *Mathematical Papers*, 2, 247–308.

Macpherson, K. and Stewart, R. (2007). Area efficient fir filters for high speed fpga implementation. *Vision, Image and Signal Processing, IEE Proceedings* -, 153, 711 – 720.

Mehra, R. (2011). Modified pso based adaptive iir filter design for system identification on fpga. *International Journal of Computer Applications*, 22, 1–7.

Mustafa, R. et al. (2009). Design and implementation of least mean square adaptive filter on altera cyclone ii field programmable gate array for active noise control. volume 1, 479 – 484.

Philips, S. (1996). *'I_2S Specification.*

Rakic, A. et al. (2015). Fpga implementation of a high-speed, real-time, windowed standard deviation filter. *Electronics Letters*, 52.

Song, X. et al. (2004). Efficient algorithm for fpga board routing. *Electronics Letters*, 40, 469 – 470.

Zölzer, U. (2011). *DAFX: digital audio effects.* John Wiley & Sons.