Capítulo IV: Projeto de Filtro Digital tipo FIR implementado em um microcontrolador

Luis Gustavo Ferrareto Espontão [6]
Renato Kazuo Miyamoto [7]
Carolina Alves Bianchini [8]
Layhon Roberto Rodrigues Dos Santos [9]
Wesley Candido da Silva [10]
Fabio Rodrigo Milanez [11]

RESUMO

Um filtro digital utiliza recursos de computação afim de implementar a ação de filtragem a ser executada a um sinal contínuo, para isso, o sinal é convertido em uma sequência finita de amostras, e após a filtragem, realiza-se a reconstrução do sinal (HAYKIN, 2001). Idealmente esses filtros são aplicados em sistemas embarcados com o objetivo de tratar os sinais, tornando possível a análise do sistema de maneira mais assertiva. A utilização de filtros digitais dispensa o uso de circuitos eletrônicos externos e possibilita a alteração do seu funcionamento de modo dinâmico, considerando que o filtro é programável. Assim, esse trabalho apresenta o projeto de um filtro digital tipo FIR passa-baixas aplicado a um sensor de luminosidade, para isso será utilizado ferramentas computacionais para determinar os parâmetros do filtro, para que seja possível implementá-lo em um sistema microcontrolado. As amostras são comparadas com os sinais coletados sem o filtro digital.

Palavras-chave: Filtro. Digital. Microcontrolador. Processamento Digital de Sinais. Filtro FIR.

[6] Email: luis.espontao@sesisenaipr.org.br
[7] Email: renato.miyamoto@sistemafiep.org.br
[8] Email: carolina.bianchini@sistemafiep.org.br
[9] Email: layhon.santos@sistemafiep.org.br
[10] Email: wesley.candido@sistemafiep.org.br
[11] Email: fabio.milanez@sistemafiep.org.br

FIR type Digital Filter project implementing in a microcontroller

ABSTRACT

A digital filter uses computing resources in order to implement the filtering action to be performed on a continuous signal, for this, the signal is converted into a finite sequence of samples, and after filtering, the signal is reconstructed (HAYKIN). , 2001). Ideally, these filters are applied in embedded systems in order to treat the signals, making it possible to analyze the system in a more assertive way. The use of digital filters does not require the use of external electronic circuits and makes it possible to change its operation dynamically, considering that the filter is programmable. Thus, this work presents the design of a low-pass FIR digital filter applied to a light sensor, for which computational tools will be used to determine the parameters of the filter, so that it is possible to implement it in a microcontrolled system. The samples are compared with the signals collected without the digital filter.

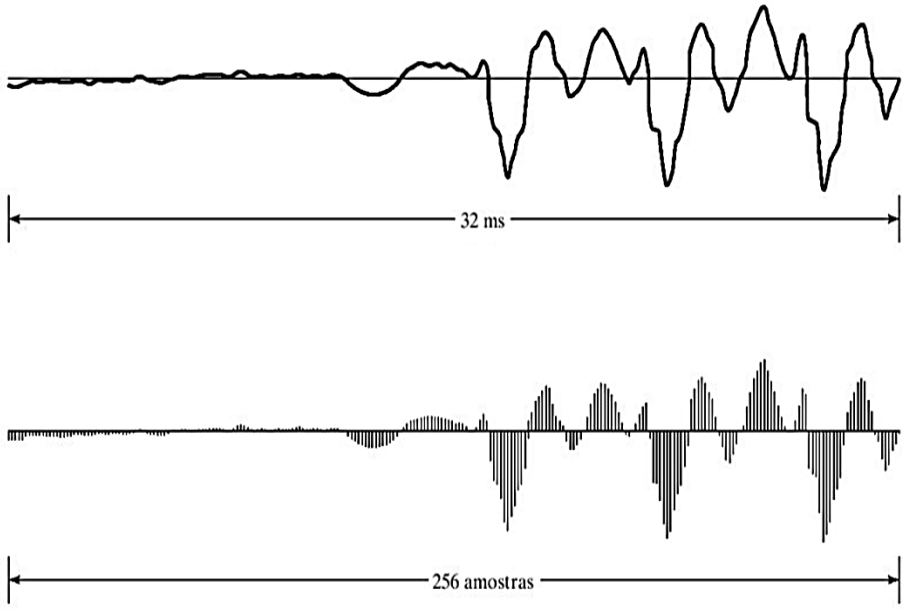
Key words: Filter. Digital. Microcontroller. Digital signal processing. FIR filter.

1. INTRODUÇÃO

O processamento de sinais lida com a representação, transformação e manipulação de sinais e da informação que os sinais contêm, podemos separar sinais que foram combinados por alguma operação (OPPENHEIN, 2012).

Para OPPENHEIN (2012), processamento digital de sinais fundamenta-se no processamento de sequências numéricas indexadas sobre variáveis inteiras, num plano de amostras, em vez de funções de uma variável independente continua no eixo do tempo, para tal é utilizado computação digital, conforme ilustrado pela Figura 1.

Figura 1 – Exemplo do sinal contínuo e discreto



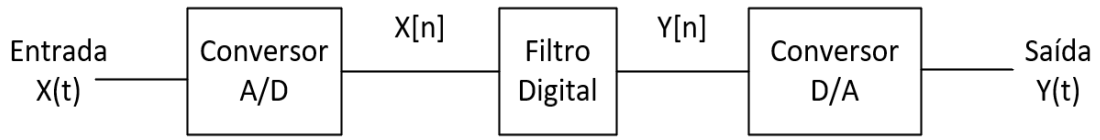
Fonte: Adaptado de OPPENHEIN, 2012.

A evolução de computadores e microprocessadores digitais para conversão de sinais analógicos para digital (A/D) e digital para analógico (D/A), tornaram possível a aplicação de sistemas de aquisição de variáveis no tempo contínuo de maneira digital, para serem posteriormente tratadas e replicadas para analogicamente.

Geralmente um sinal amostrado digitalmente é tratado através de um filtro para remover sinais indesejados, ou amplificar determinado sinal. Para essa finalidade utiliza-se de filtros digitais.

Um filtro digital utiliza computação afim de implementar a ação de filtragem a ser executava a um sinal contínuo, convertido em uma sequência finita de amostras, para após a filtragem ser reconstruído em um sinal contínuo (HAYKIN, 2001).

Figura 2 – Sistema de Filtro de sinal contínuo



Fonte: Do autor.

Dessa forma esse sistema pode ser empregado a um microcontrolador embarcado com a finalidade de realizar a filtragem de um sinal amostrado. Com esse objetivo, será projetado um filtro digital através de ferramentas matemáticas computacionais, acerca das ferramentas disponíveis no software e na literatura.

Obtendo o modelo do filtro projetado, será implementado em uma aplicação em um microcontrolador, para leitura de uma variável de intensidade luminosa para filtrar os ruídos gerados por sinais indesejados somados ao sinal fundamental.

2. FUNDAMENTAÇÃO TEÓRICA

2.1. FILTRO DIGITAL FIR

Filtros do tipo FIR (*Finite Impulse Response)* são característicos por uma resposta ao impulso finita, devido a essa característica, possuem memória finita, portanto, qualquer transitório tem duração limitada (HAYKIN, 2001). Sua operação é regida por equações lineares de diferenças com coeficientes constantes de natureza não-recursiva e sua função de transferência pode ser representada de acordo com a equação 1.

$$H(\mathcal{Z}) = \sum_{n=0}^{M} b_n \mathcal{Z}^{-n} \quad (1)$$

A metodologia mais simples de projeto filtro FIR é por meio do *método de janelamento*. Esse método se inicia com uma resposta em frequência desejada ideal, expressa pela equação 2 (OPPENHEIN, 2012).

$$H_d(e^{j\omega}) = \begin{cases} e^{-j\omega n_d}, & |\omega| \leq \omega_c \\ 0, & \omega_c < |\omega| \leq \pi \end{cases} \quad (2)$$

A resposta ao impulso da equação 2, é dada pela equação 3.

$$h_d[n] = \frac{\sin \omega_c (n - n_d)}{\pi(n - n_d)}, -\infty < n < \infty \quad (3)$$

A resposta impulsiva tem duração infinita e é não-causal, para solucionar esse problema, deve-se truncar a resposta impulsiva.

Admita-se que M denote a ordem de filtro, correspondente a um comprimento de filtro $M + 1$.

$$h[n] = \begin{cases} h_d[n], & 0 \leq n \leq M = 2n_d = N - 1 \\ 0, & caso\ contrário \end{cases} \quad (4)$$

Equivalendo a uma multiplicação da equação 3, por uma janela de resposta infinita.

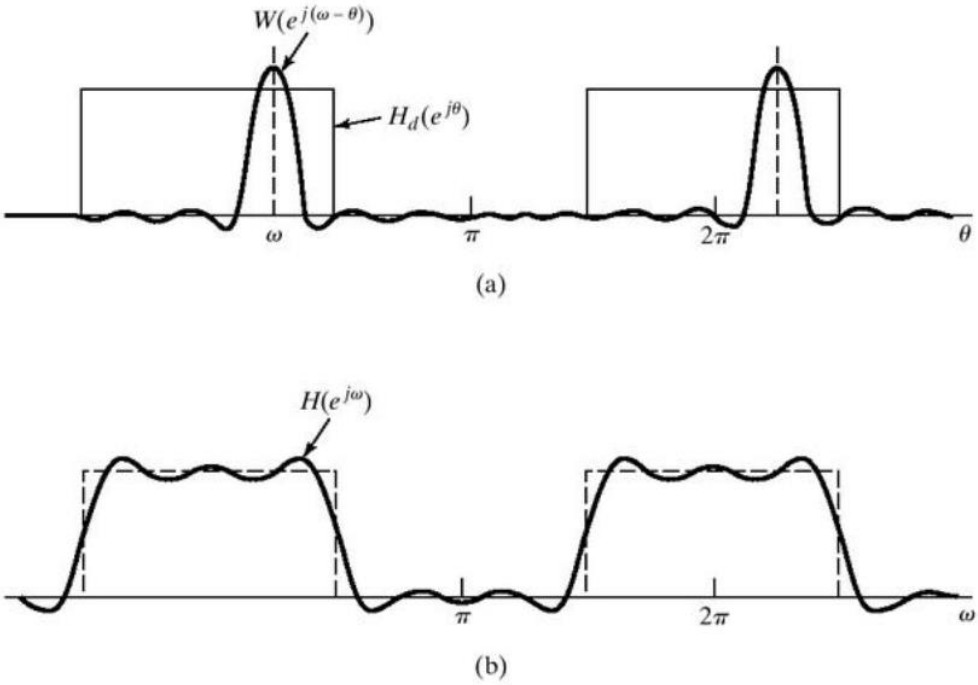
$$h[n] = h_d[n] \cdot w[n] \quad (5)$$

No caso para um truncamento simples, a janela de resposta infinita se trata de uma janela retangular, expressa pela equação 6.

$$w[n] = \begin{cases} 1, & 0 \leq n \leq M = N - 1 \\ 0, & caso\ contrário \end{cases} \quad (6)$$

O efeito do janelamento é evidente no domínio da frequência, onde tem-se a convolução periódica entre a resposta em frequência ideal e o espectro da janela, ilustrado pela Figura 3.

Figura 3 – Efeitos Janelamento



Fonte: Adaptado de OPPENHEIN, 2012.

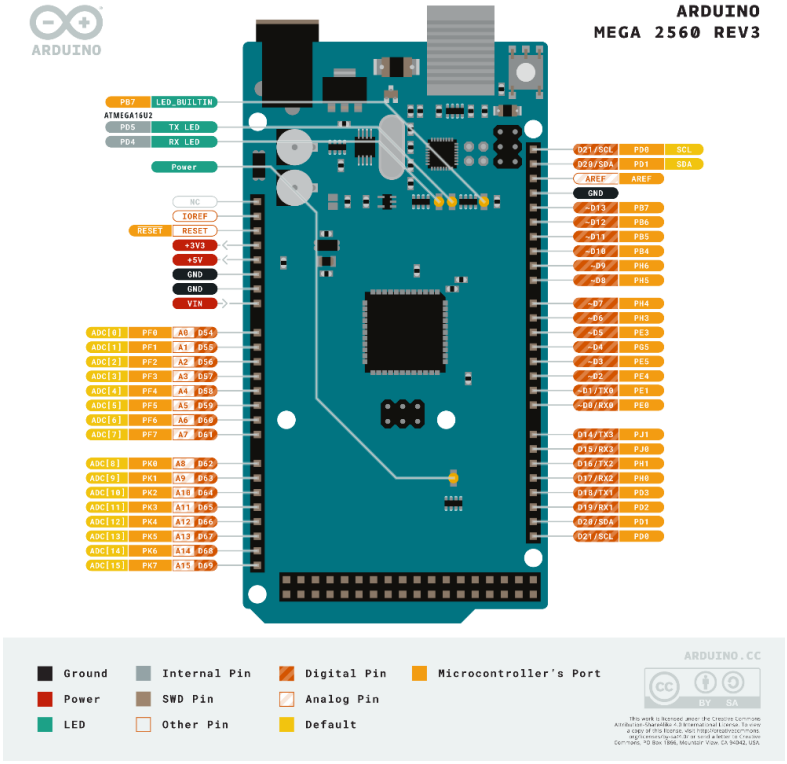
A ilustração (a) se refere ao processo da convolução decorrente do truncamento da resposta ao impulso ideal ($W(e^{-j(\omega-\theta)})$), a figura (b) demonstra a aproximação típica resultante do janelamento da resposta ao impulso ideal (OPPENHEIN, 2012). Dessa forma, a escolha da janela irá influenciar a resposta em frequência do filtro obtido. Um filtro digital pode ser aplicado a sistemas digitais microcontrolados, para isso é necessário compreender sobre os aspectos relacionados a microcontroladores. Nessa pesquisa, utilizou-se o microcontrolador ATmega2560 embarcado na plataforma de prototipagem Arduino MEGA 2560 R3.

2.2. PLATAFORMA ARDUINO MEGA 2560 R3

Arduino se trata de uma plataforma de desenvolvimento de código livre. O Arduino MEGA utilizado nesse trabalho emprega o microcontrolador ATmega2560 que possui entradas e saídas para sensores. A programação é desenvolvida utilizando uma linguagem de programação específica e um ambiente de desenvolvimento. A

disposição dos pinos da placa Arduino utilizada para o experimento está ilustrada na Figura 4 (ARDUINO, 2022).

Figura 4 – Arduino MEGA 2560 R3



Fonte: ARDUINO, 2022.

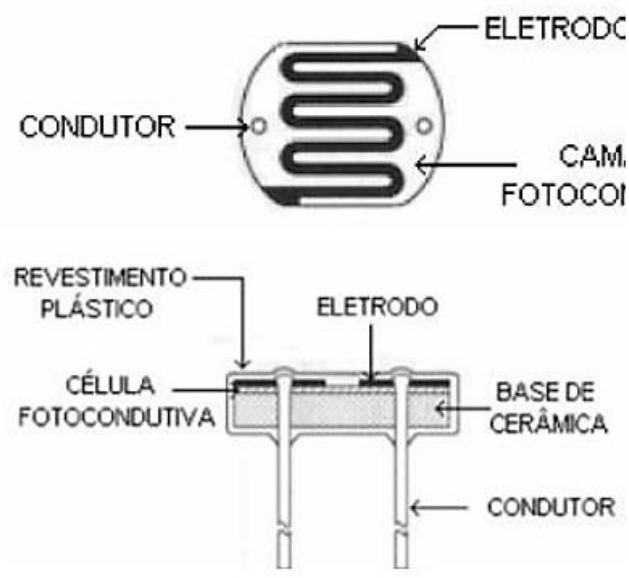
Em termos práticos “um Arduino é um pequeno computador que você pode programar para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele” (MCROBERTS, 2011, p. 22). O Arduino pode receber e enviar dados por meio das portas, que são programadas para funcionarem como entrada ou saída. A entrada pode ser analógica ou digital, e para esse projeto, utiliza-se uma entrada analógica para a aquisição de sinais de luminosidade através de um sensor termo resistivo.

2.3. RESISTOR DEPENDENTE DE LUZ (*LDR*)

Os semicondutores possuem características que reagem com a incidência de luminosidade e aplicação de temperatura, que influencia diretamente sua capacidade de conduzir corrente elétrica (GHELLERE, 2009). O LDR (*Light Depedent Resistor*)

ou Resistor Dependente de Luz, é um dispositivo semicondutor eletrônico que possui dois terminais e a características de alterar sua resistência de acordo com a incidência de luz sobre seu encapsulamento conforme ilustra a Figura 5.

Figura 5 – Visa superior e frontal LDR



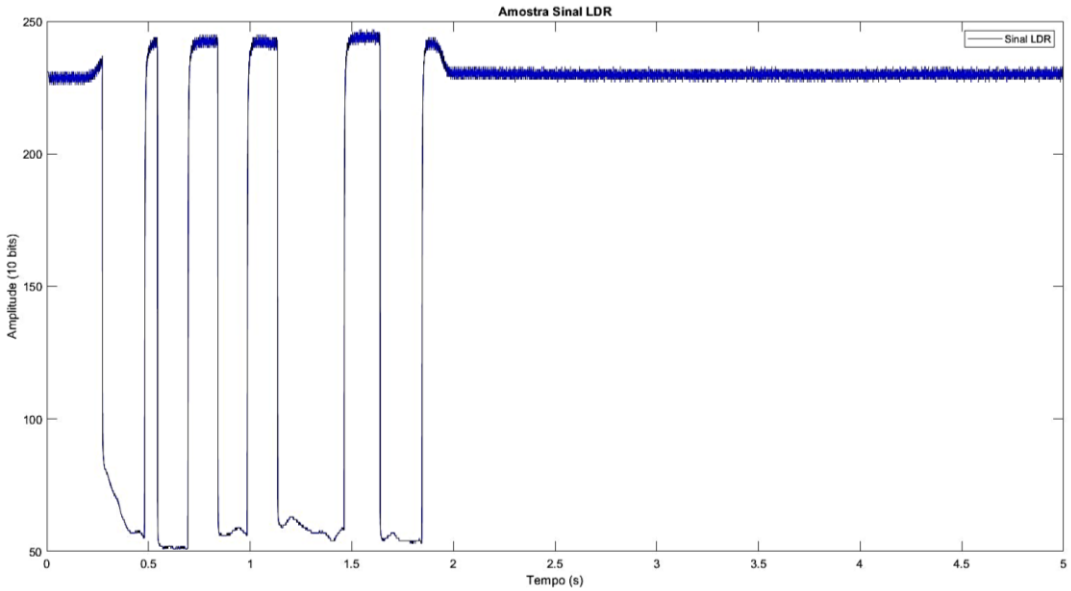
Fonte: GHELLERE, 2009.

Seu encapsulamento consiste basicamente no material semicondutor na sua forma pura, protegido por um invólucro transparente a quase todas as cores do espectro visível, de modo que possa absorver melhor a luz (GHELLERE, 2009). A Seção 3 apresenta os aspectos metodológicos utilizados nesse trabalho.

3. METODOLOGIA

Para o desenvolvimento do filtro digital, é necessário o conhecimento dos dados a serem amostrados para identificar os parâmetros de atenuação e banda de passagem do filtro. Desta forma foi realizada a coleta do sinal de intensidade luminosa, através do Arduino e um módulo contendo o sensor LDR, enviando os dados lidos via serial para o software de modelagem matemática. Os dados amostrados são representados na Figura 6.

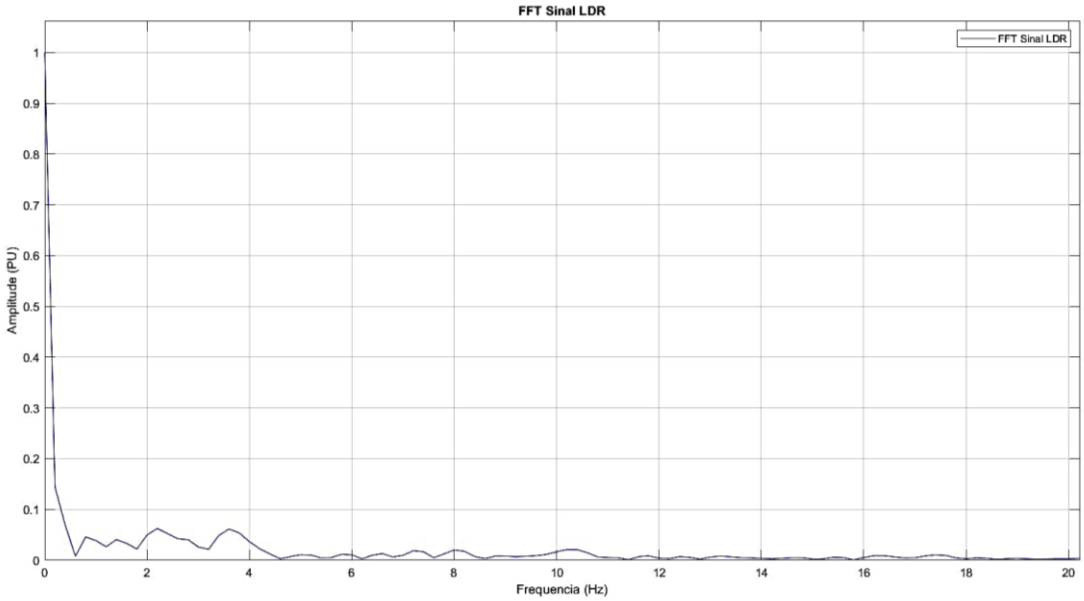
Figura 6 – Amostragem LDR



Fonte: Do autor.

No momento da coleta, a incidência de luz no LDR, foi variada com o intuito de identificar os transitórios do sensor. Na sequência foi realizada a FFT (*Fast Fourier Transform*) do sinal da Figura 7.

Figura 7 – FFT Sinal LDR

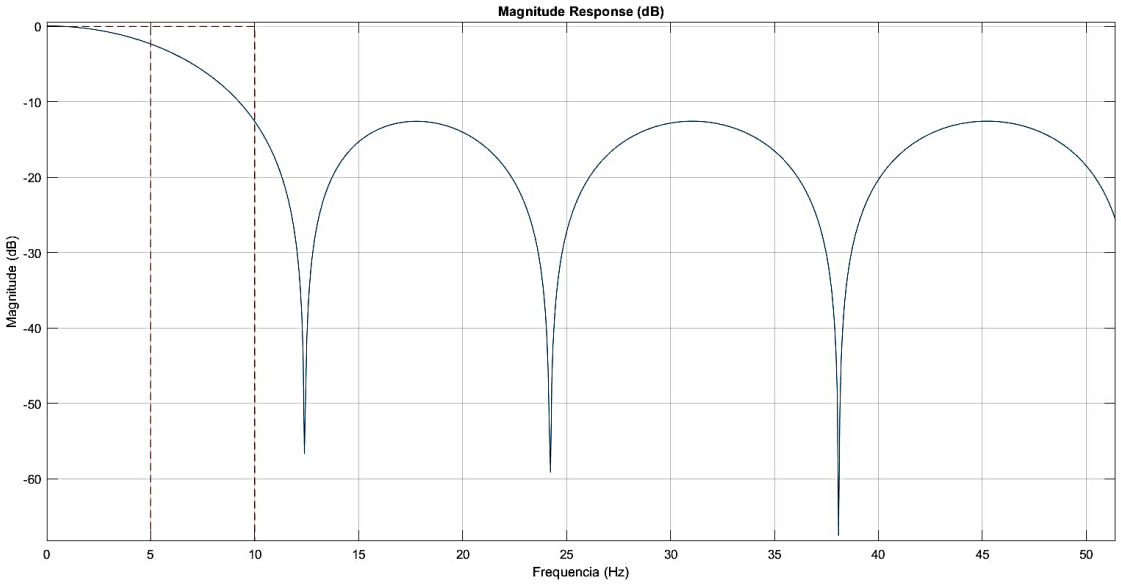


Fonte: Do autor.

Ao analisar as amplitudes do sinal, ilustradas pela Figura 7, podemos identificar algumas amplitudes de sinais na ordem de 4 Hz até próximo de 12 Hz.
Dessa forma o filtro deverá atenuar estas frequências, a fim de reduzir os ruídos não desejados ao sinal, dessa forma será projeto um filtro passa-baixas, que deverá atenuar frequências altas a partir de uma determinada frequência de corte.

O filtro será projetado utilizando a ferramenta *designfilt*, disponível no software de modelagem matemática, onde ao inserir os parâmetros de tamanho do janelamento, frequência de corte, banda de transição e a frequência de amostragem de nosso microcontrolador, que no caso é próxima aos 1,5 KHz. Os valores foram empiricamente ajustados conforme a resposta do filtro a um impulso, ilustrado na Figura 8.

Figura 8 – Resposta Magnitude em função da Frequência

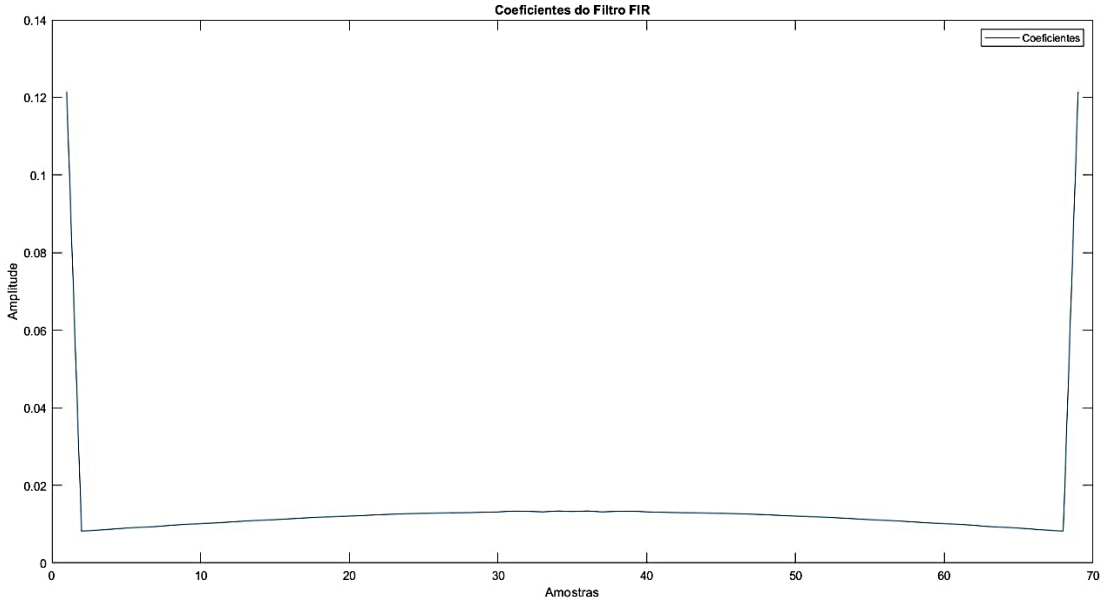


Fonte: Do autor.

Onde podemos ver o ganho dos sinais pela frequência, pode-se identificar na Figura 8, que a frequência de corte que deu os resultados mais satisfatório foi de 5Hz, onde o filtro passa a atenuar os sinais acima de 3 Hz, e em 5 Hz apresenta 70,7% de seu sinal, ou -3dB de atenuação.

A banda de transição selecionada foi de 5Hz, para garantir a estabilização da atenuação do filtro próximo a -12dB de ganho. Dessa forma o janelamento o sistema é constituído de 69 coeficientes ilustrados na Figura 9.

Figura 9 – Coeficientes Filtro Passa-Baixa



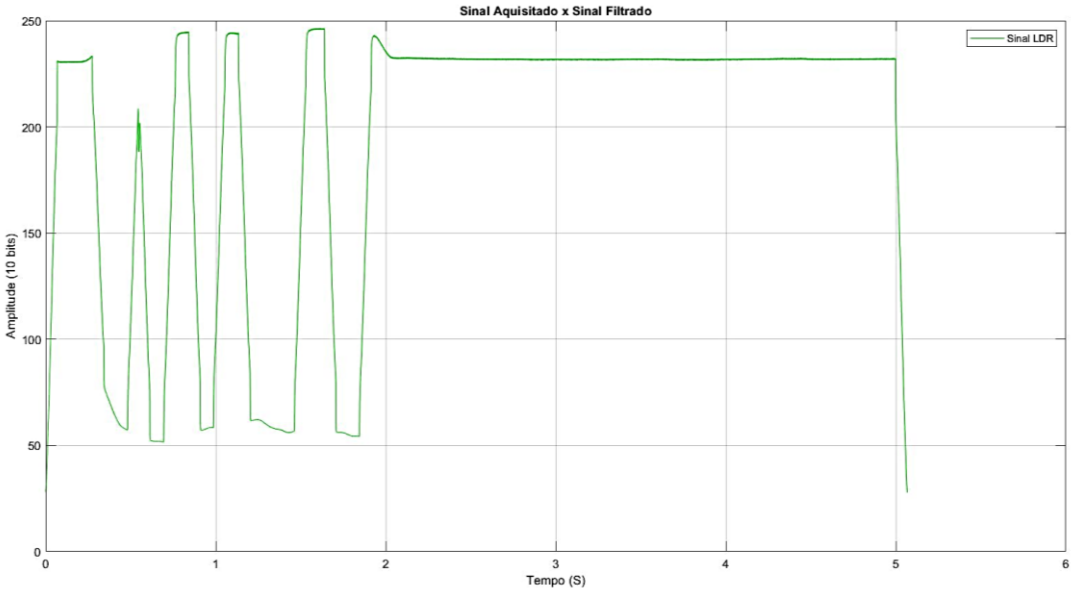
Fonte: Do autor.

Dessa forma será realizado a convolução destes coeficientes com o sinal amostrado, ilustrado na Figura 6, para realizar a filtragem das frequências demonstradas na Figura 8.

4. APRESENTAÇÃO E DISCUSSÃO DOS RESULTADOS

Os resultados obtidos através da convolução do filtro desenvolvido na metodologia seguem ilustrados na Figura 10. Pode-se verificar a atenuação dos ruídos presentes no sinal amostrado (Figura 6). A convolução dos coeficientes do filtro ocorre em tempo real e foi implementado no microcontrolador ATmega2560. Dessa forma observa-se que o filtro realizou de maneira coerente o tratamento do sinal, de forma a tornar um sinal ruidoso para um sinal limpo, adequado para ser aplicado em um sistema embarcado.

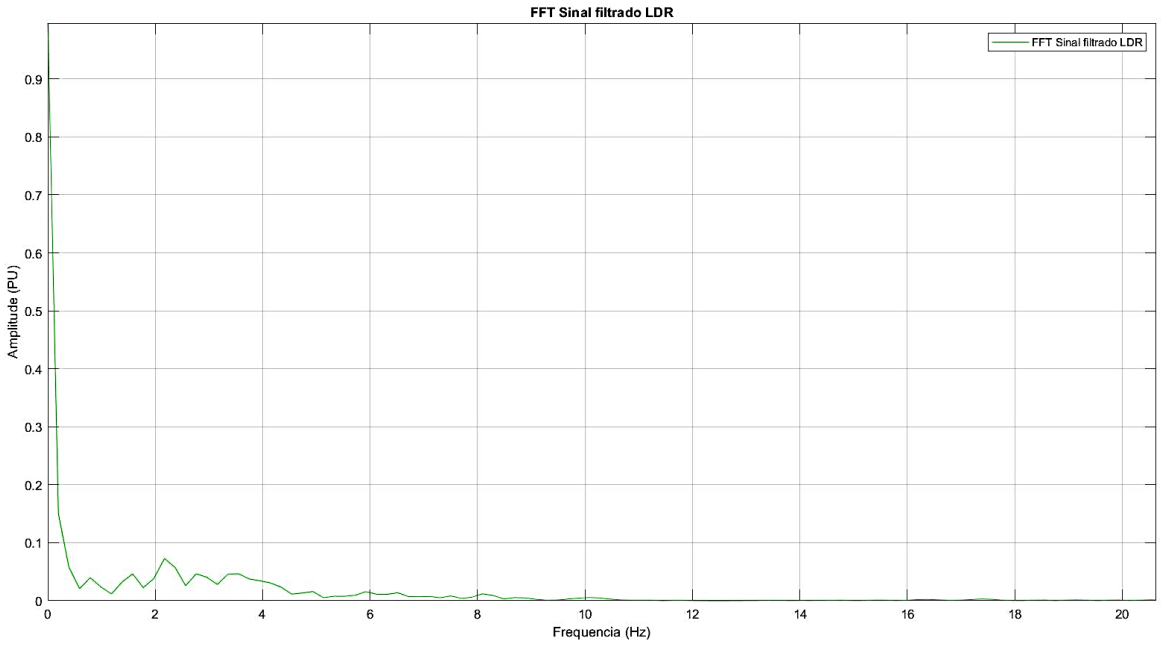
Figura 10 – Sinal Filtrado



Fonte: Do autor.

A Figura 11 ilustra a FFT (*Fast Fourier Transform*) do sinal obtido pela filtragem, para identificar as frequências que foram atenuadas em relação as apresentadas na Figura 7.

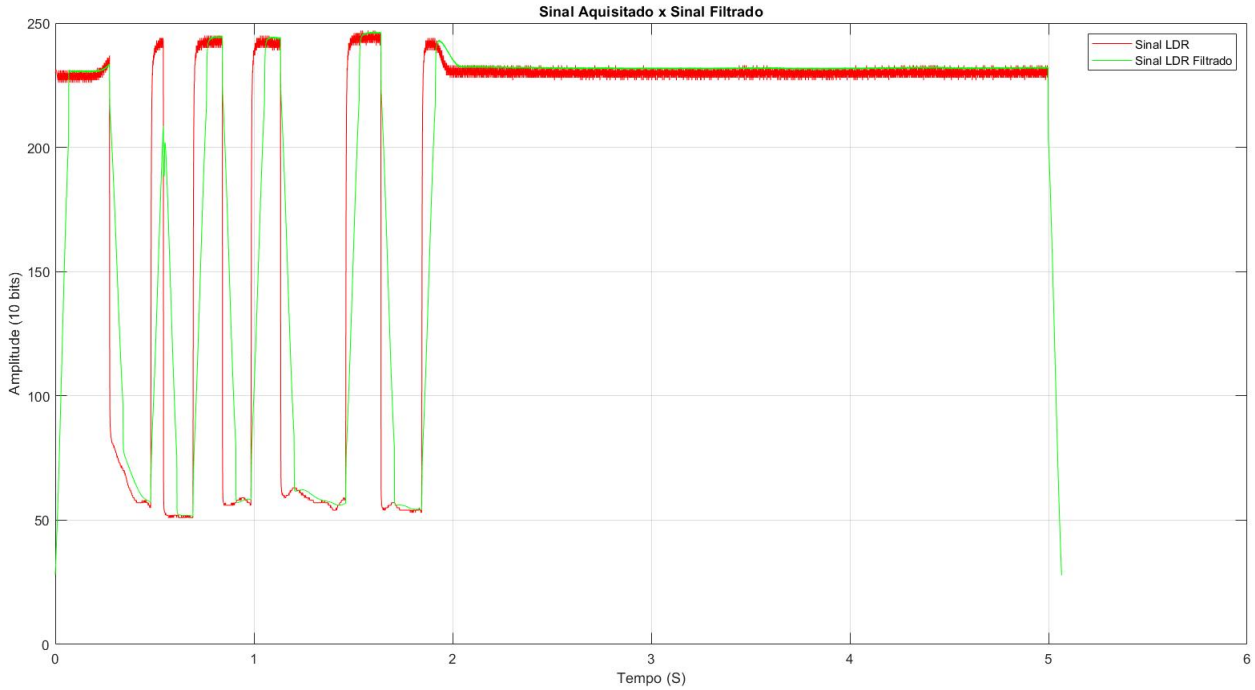
Figura 11 – FFT Sinal Filtrado



Fonte: Do autor.

Analisando a Figura 11 em relação a Figura 7, identifica-se que houve atenuações de sinais acima de 4 Hz, que compõem os principais ruídos do sinal amostrado. A Figura 12, ilustra a sobreposição dos sinais, em que fica evidenciado a correção das amostragens do filtro.

Figura 12 – Sobreposição do Sinal amostrado x Sinal filtrado

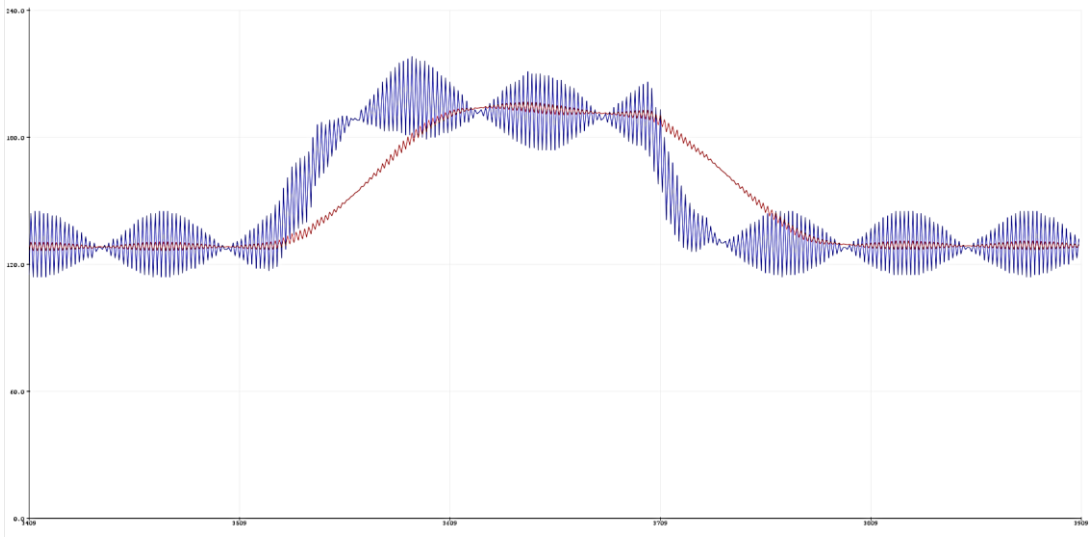


Fonte: Do autor.

No Arduino utilizou-se a biblioteca FIR que já possui funções voltadas para a aplicações de filtros digitais. O script utilizado para o tratamento das amostras consta em Anexo nesse artigo. Para identificar a atuação do filtro, foi utilizado a serial plotter, presente na interface de programação do Arduíno, que possibilita realizar a plotagem dos dados em tempo real. Foi realizado a plotagem do sinal de entrada do sensor LDR, e a plotagem do sinal filtrado.

A Figura 13 ilustra o resultado obtido pelo filtro digital FIR. Na figura, o sinal plotado em azul se refere ao lido em tempo real sem a aplicação da filtragem. O sinal plotado em vermelho refere-se ao sinal filtrado.

Figura 13 – Implementação Filtro Arduino



Fonte: Do Autor.

Analisando a resposta do filtro, vemos um certo atrasado devido a amostragem e os coeficientes, porém a resposta está bem agradável para este tipo de sinal.

5. CONSIDERAÇÕES FINAIS

O intuito deste trabalho é demonstrar como utilizar de maneira objetiva as ferramentas disponíveis em softwares de modelagem matemática, para realizar o projeto de filtros digitais, e aplicação em sistemas embarcados.

No exemplo deste trabalho foi a coleta de intensidade luminosa, que pode ser usada para várias aplicações da indústria, como contagem de peças de um processo, ou verificação de luminosidade incidida em painéis solares.

Pode-se aplicar para qualquer sinal que necessite deste tipo tratamento, seja para remoção de ruídos, ou para coletar determinada característica presente no sinal, como por exemplo, filtrar as frequências de um sinal de áudio, que representam um ruído presente na gravação.

REFERÊNCIAS

ARDUINO. **Arduino MEGA 2560 R3**. 2022. Disponível em: https://docs.arduino.cc/hardware/mega-2560 .Acesso em: 09 de junho, 2022.

MCROBERTS, M. **ARDUINO BÁSICO**. São Paulo: Novatec, 2011.

GHELLERE, G. LDR **Light Dependent Resistor: Resistor Variável de acordo com incidência de luz**. Foz do Iguaçu: 2009. Disponível em: http://www.foz.unioeste.br/~lamat/downcompendio/compendiov7.pdf . Acesso em: 09 de junho, 2022.

OPPENHEIN A.V., SCHAFER R.W., **PROCESSAMENTO DE SINAIS EM TEMPO DISCRETO**, 3ª Edição, 2012.

HAYKIN, S. **SINAIS E SISTEMAS**. Ed. Bookman, 2001.

ANEXO A – CÓDIGO ARDUINO

```
#include <FIR.h>

FIR<float, 68> fir_lp;

int sensorPin = 0;
int LDR = 0;
void setup()
{
  Serial.begin(9600);

  float coef_lp[69] = {
0.121475311702895,0.00817690691154585,0.00844382222120648,0.00869474782
810398,0.00895556041027448,0.00921511872152930,

0.00940148993964046,0.00969100933249147,0.00991365032104916,0.010143139
3003249,0.0103394929802585,0.0105530810397007,

0.0107730684208698,0.0109803607212739,0.0111863404930232,0.011383240203
0097,0.0115792079187627,0.0117667523065124,0.0119354448267821,

0.0121067740678013,0.0122606850398663,0.0124251725906382,0.012552522431
9711,0.0126713639569128,0.0127668966458493,0.0128578634908869,

0.0129519838401939,0.0129920266404429,0.0131000541368050,0.013170702285
7531,0.0133465373798627,0.0133021972298821,0.0131702524492663,

0.0133803823253790,0.0132622210004390,0.0133803823253790,0.013170252449
2663,0.0133021972298821,0.0133465373798627,0.0131707022857531,

0.0131000541368050,0.0129920266404429,0.0129519838401939,0.012857863490
8869,0.0127668966458493,0.0126713639569128,0.0125525224319711,

0.0124251725906382,0.0122606850398663,0.0121067740678013,0.011935444826
7821,0.0117667523065124,0.0115792079187627,0.0113832402030097,

0.0111863404930232,0.0109803607212739,0.0107730684208698,0.010553081039
7007,0.0103394929802585,0.0101431393003249,0.00991365032104916,

0.00969100933249147,0.00940148993964046,0.00921511872152930,0.008955560
41027448,0.00869474782810398,0.00844382222120648,0.00817690691154585,0.
121475311702895      };

  // Set the coefficients
  fir_lp.setFilterCoeffs(coef_lp);

  // Set the gain
  Serial.print("Low Pass Filter Gain: ");
```

```
  Serial.println(fir_lp.getGain());
}

void loop() {
  // Run through our simulated data in "real time" and compare the outputs.
  // You can paste the output in your favorite graphing program if you want
  // to see how the different filters modify the output.
  LDR = analogRead(sensorPin);
  Serial.print(LDR);
  Serial.print(",");
  Serial.println(fir_lp.processReading(LDR));
  delayMicroseconds(1);

}
```

ANEXO B – CÓDIGO OCTAVE

```
clear; clc; close all;

load('LDR.mat');

fs = 1000; %Frequencia de amostragem
Ts = 1/fs; %Tempo de amostragem

figure(1);
plot(x,y,'b');
title('Amostra Sinal LDR');
legend('Sinal LDR');
xlabel('Tempo (s)');
ylabel('Amplitude (10 bits)');
%% FFT Sinal

% Frequencia de analise FFT sinal
[R,C] = size(y);
freq = (0:C-1)/(C * Ts);
%freq = freq';
Yfft = fft(y);
%Normalização dos valores
ampspec_Y = abs(Yfft);
base_Y = max(ampspec_Y);
ampY_PU = ampspec_Y/base_Y;
%------------------------------

figure(2);
plot(freq,ampY_PU,'b');
grid on
axis([0 200 0 1.5]);
title('FFT Sinal LDR');
legend('FFT Sinal LDR');
xlabel('Frequencia (Hz)');
ylabel('Amplitude (PU)');

%% Projeto FIR - DesignFilt - Projeto

M = 68;
f1 = 5;
banda = 5;

d = designfilt('lowpassfir','filterOrder',M,...
'PassbandFrequency', f1, 'StopbandFrequency', f1+banda,...
```

```
'SampleRate', fs);

fvtool(d);

figure(3)
plot(d.Coefficients);
title('Coeficientes do Filtro FIR');
legend('Coeficientes');
xlabel('Amostras');
ylabel('Amplitude');

Fitro_FIR = d.Coefficients;

Y_FIR = conv(y,Fitro_FIR);

%% Frequencia de analise FFT sinal filtrado
[R1,C1] = size(Y_FIR);
freq1 = (0:C1-1)/(C1 * Ts);
t1 = (0:C1-1) * Ts;
freq1 = freq1';

Y_FIR_fft = fft(Y_FIR);

%Normalização dos valores
ampspec_Y_FIR_fft = abs(Y_FIR_fft);
base_Y_FIR_fft = max(ampspec_Y_FIR_fft);
ampY_FIR_PU = ampspec_Y_FIR_fft/base_Y_FIR_fft;
%------------------------------

figure(4);
plot(x,y,'r');
plot(freq1,ampY_FIR_PU,'g');
axis([0 200 0 1.5]);
grid on
title('FFT Sinal filtrado LDR');
legend('FFT Sinal filtrado LDR');
xlabel('Frequencia (Hz)');
ylabel('Amplitude (PU)');

%%
figure(5);
%plot(x,y,'r');
%hold on
plot(t1,Y_FIR,'g');
grid on
title('Sinal Aquisitado x Sinal Filtrado');
legend('Sinal LDR','Sinal LDR Filtrado');
```

```
xlabel('Tempo (S)');
ylabel('Amplitude (10 bits)');
```