Osciloscópio baseado em FPGA

IISE – Projeto 4

Diogo Miguel Cunha Fernandes, PG47150
José Tomás Lima de Abreu, PG47386

Orientação:
Professor Doutor Jorge Cabral
Professor Doutor Rui Machado
Professora Sofia Paiva

Projeto Integrador em Eletrónica Industrial e Computadores
Universidade do Minho 2021/2022

CENTROALGORITMI

FCT Fundação para a Ciência e a Tecnologia

Índice

Problema e contexto

01 Problema e contexto

02 Cenários de aplicação

03 Objetivos e resultados esperados

04 Requisitos e restrições

Problema e contexto

- **Aumento da complexidade dos sistemas digitais [2]:**
  - Maior dificuldade no desenho de *hardware;*
  - Maior tempo de desenvolvimento.

- **Algoritmos de processamento digital de sinal (PDS):**
  - Complexos;
  - Difíceis de implementar em *hardware.*

Cenários de aplicação

- Criptografia baseada em *hardware*;
- Acelerador de unidade de processamento de visão por computador;
- Controlo e atuação de um braço robótico;
- **Aplicações que exijam processamento digital de sinal:**
  - Aplicações de interface com sensor – Ex: Filtragem de sinal;
  - Cancelamento de eco acústico;
- **Osciloscópio digital:**
  - Aquisição de sinal;
  - Filtragem digital de sinal;
  - Suporte gráfico.

Objetivos e resultados esperados

- Explorar técnicas de PDS recorrendo a *High Level Synthesis* (HLS);

- **Comparação do desenvolvimento de *hardware* recorrendo a HLS com o desenvolvimento utilizando *Hardware Description Languages* (HDL): [4]**
  - Qualidade do *hardware* gerado;
  - Tempo de desenvolvimento;

- **Osciloscópio básico capaz de:**
  - Amostrar sinais e aplicar filtros digitais;
  - Apresentação dos sinais num *display;*

Requisitos e restrições

- **Requisitos:**
  - Aquisição de sinal;
  - Implementação de filtros digitais;
  - Apresentação do sinal filtrado num *display*.

- **Restrições:**
  - Usar FPGA-SoC Zybo Z7-10;
  - Usar HLS;
  - Equipa de 2 membros;
  - Entrega de projeto no final do semestre.

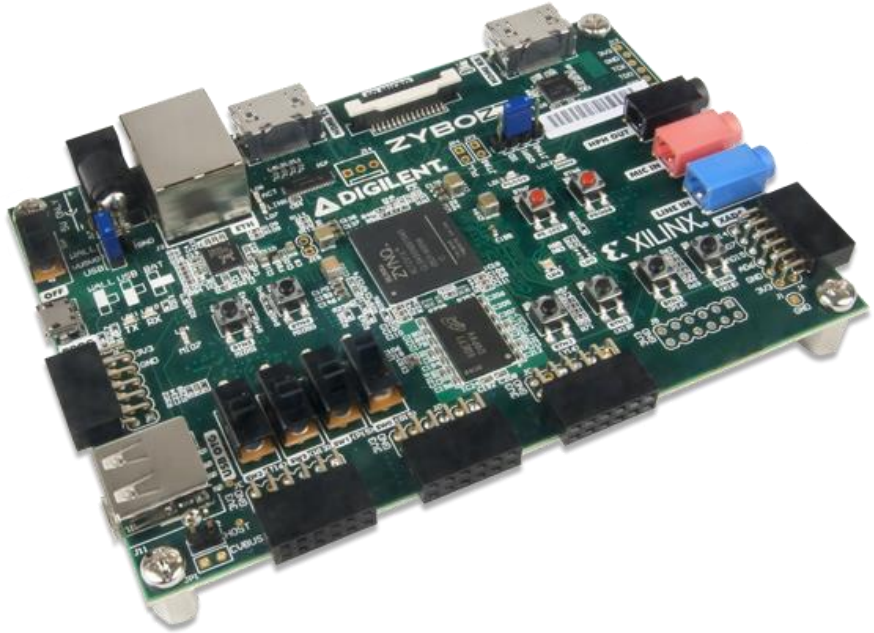


Figura 1 – Zybo Z7-10, da Digilent.

Estado da arte

01 Algoritmo de desenho de FPGA

02 HLS – *High Level Synthesis*

03 Amostragem

04 Filtros digitais

05 Aritmética de *Fixed Point*

Algoritmo de desenho de FPGA

1. Criar os **ficheiros HDL**;
2. Para cada módulo deve ser **verificado o RTL** (*Register Transfer Level)* gerado;
3. Criação de **ficheiros *testbench*** para testar e validar as fases do desenho;
4. **Simulação comportamental** do sistema;
5. **Síntese** – mapeamento do RTL para uma tecnologia específica;
6. **Implementação** – definir onde colocar a lógica definida;
7. **Geração do *bitstream*** – configurar a lógica FPGA;
8. Verificação das **regras de *design***;
9. **Programação da FPGA**;

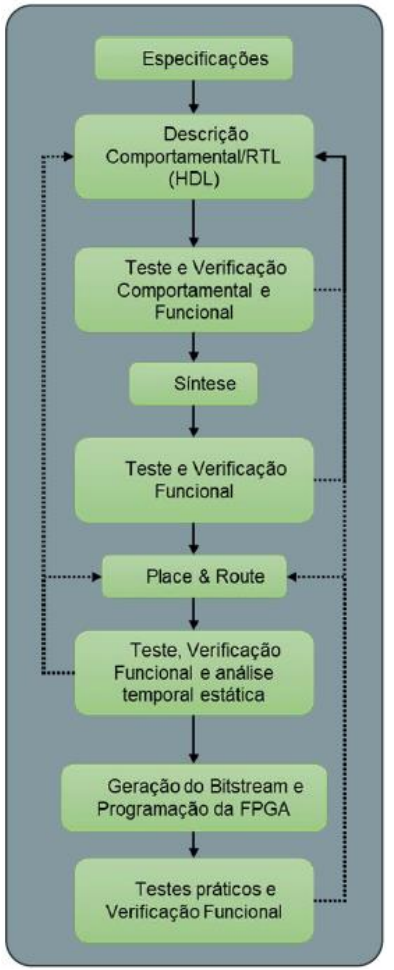


Figura 2 – Algoritmo de desenho de FPGA.

HLS – *High Level Synthesis*

- **Processo de desenho em alto nível:**
  - Elevado nível de abstração;
  - Automatiza processos que seriam realizados manualmente por um designer de RTL;
  - Utilização de linguagens de especificação de alto nível como C, C++;

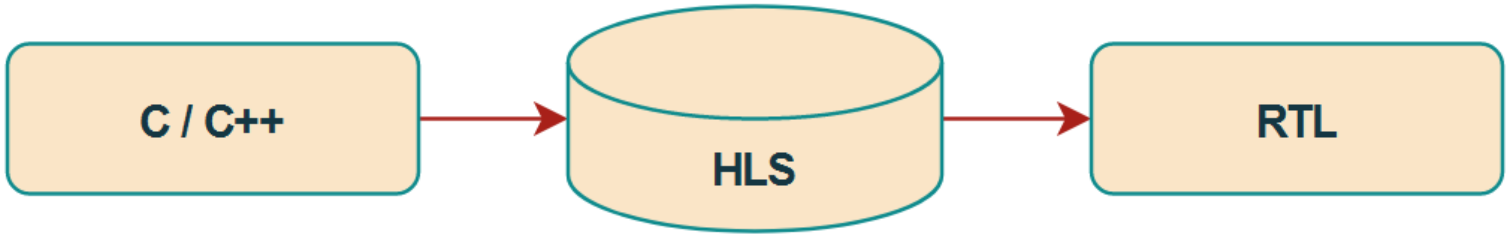


Figura 3 – Função do HLS no processo de desenho de *hardware*.

Amostragem

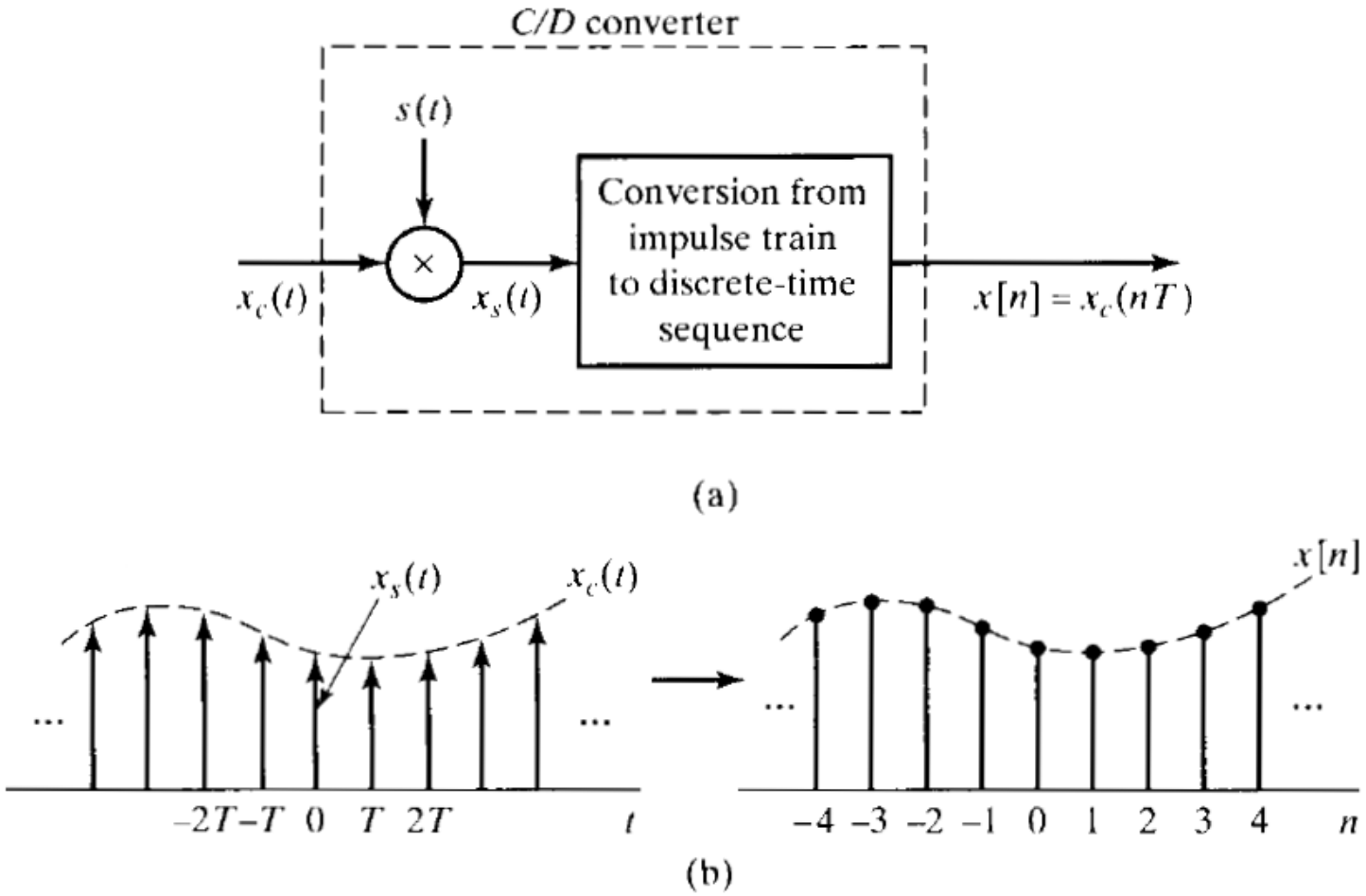


Figura 4 - Processo da amostragem: a) Conversor tempo contínuo - tempo discreto.

b) Sinal contínuo à esquerda e respetivo sinal discreto à direita.

(adaptado do livro [1] página 142)

Amostragem - Aliasing

- **Teorema de Nyquist:** $f_s \geq 2.f_M$

- **Aliasing:** Resultado do sinal ter sido sub-amostrado;

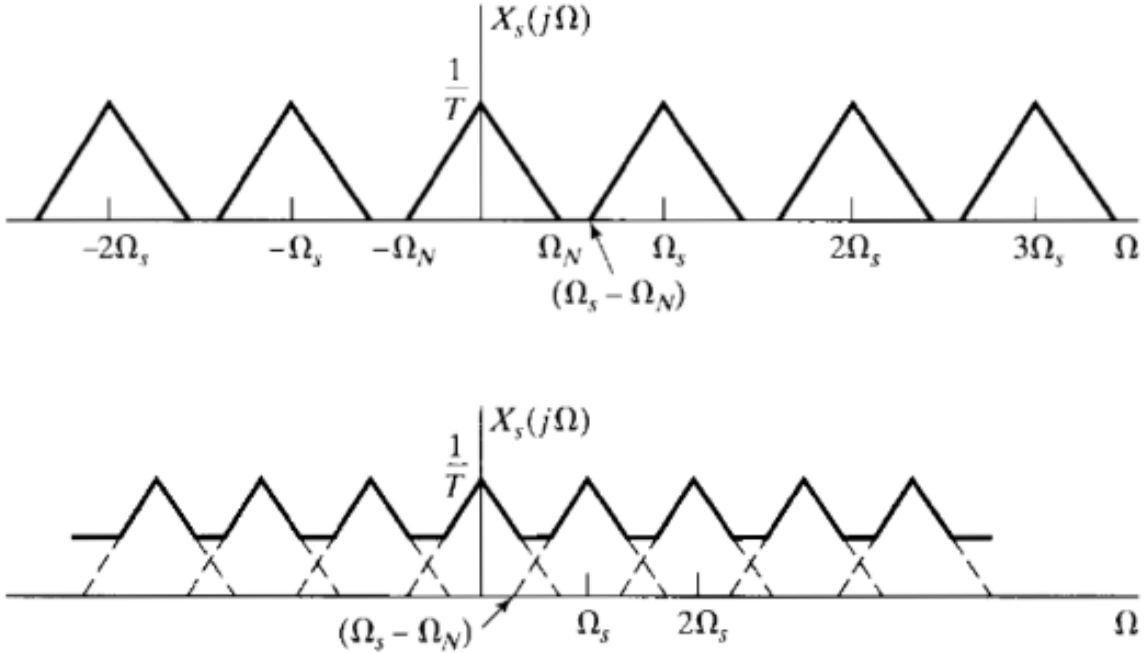


Figura 5 - Efeito do aliasing no domínio das frequências. Em cima: Espetro do sinal amostrado com fs > 2.fM

Em baixo: espetro do sinal amostrado com fs < 2.fM (adaptado do livro [1] página 144)

Filtros digitais

- **Equação de diferenças de um filtro digital:**

$$y[n] = a_1 y[n-1] + \ldots + a_N y[n-N] + b_0 x[n] + b_1 x[n-1] + \ldots + b_M x[n-M]$$

- **Filtros *Infinite Impulse Response* (IIR):**
  - Saída calculada com base nos $x[n-k]$ e em $y[n-k]$
  - Saída instável;
  - Desenho a partir de filtros analógicos.

Filtros digitais

- **Equação de diferenças de um filtro digital:**

$$y[n] = a_1 y[n-1] + ... + a_N y[n-N] + b_0 x[n] + b_1 x[n-1] + ... + b_M x[n-M]$$

- **Filtros *Finite Impulse Response* (FIR):**
  - Coeficientes a_k são nulos;
  - Saída calculada com base nos $x[n-k]$
  - Saída estável;
  - Resposta em frequência com fase linear;
  - Maior tempo de cálculo; [5]

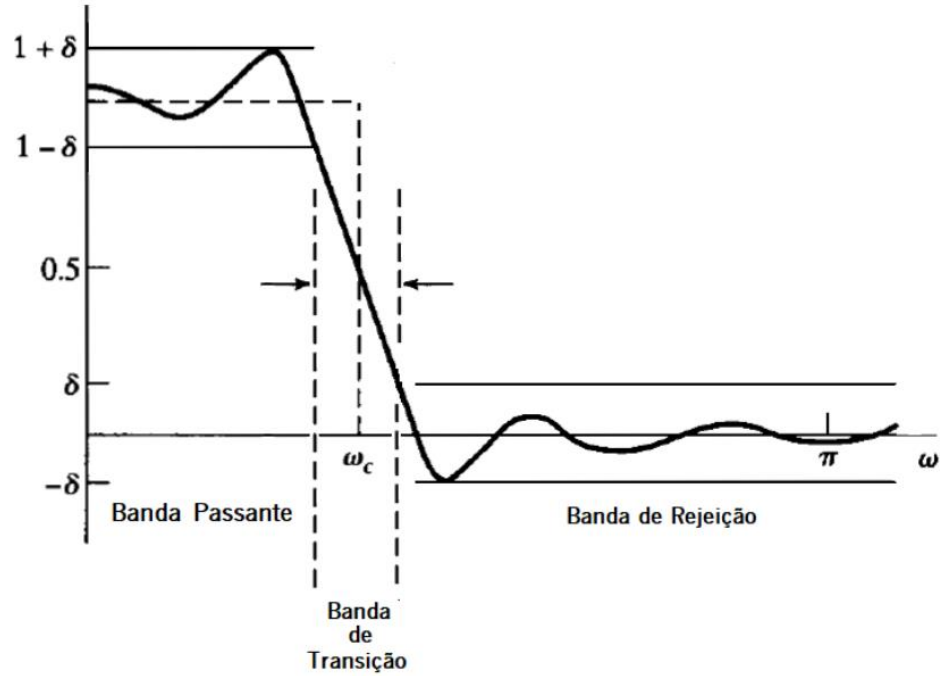


Figura 6 - Resposta em frequência típica de um filtro passa baixo, em que wc é a frequência de corte do filtro. (adaptado do livro [1] página 473)

Filtros digitais – Método das Janelas

- **Gerar uma janela de comprimento** $M+1$
  - Filtro é obtido pela multiplicação da janela pela resposta impulsional de um LPF ideal, limitando o seu comprimento e forma [1]

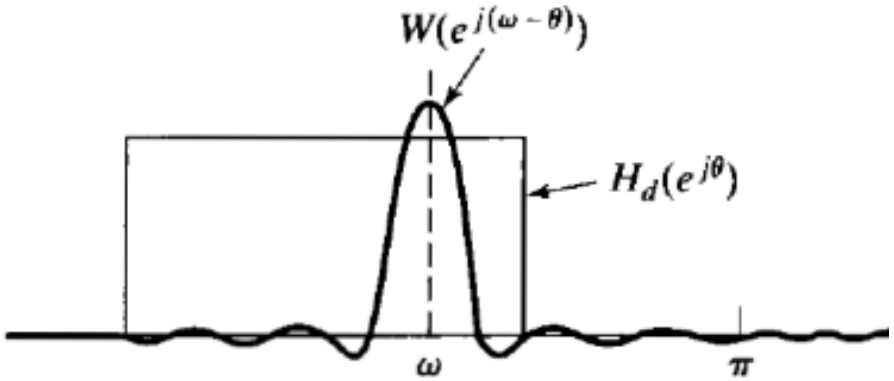


Figura 7 - Convolução da janela, W, com a resposta impulsional do LPF ideal, Hd. (adaptado do livro [1] página 467)

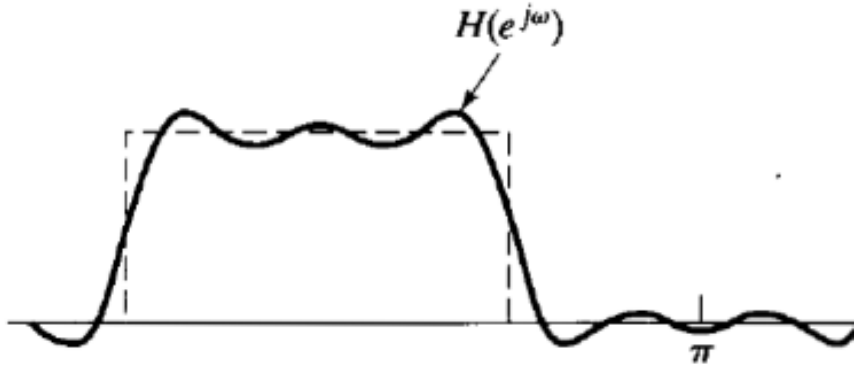


Figura 8 - Aproximação do resultado da aplicação de uma janela a uma resposta impulsional ideal. (adaptado do livro [1] página 467)

- **Janela de Kaiser:** flexível e evita procedimentos de tentativa e erro;

Aritmética de *Fixed Point*

- **Forma de representar um número real através de um número inteiro:**
  - Número fixo de dígitos para a parte decimal;
  - Perda de precisão;
  - Menos complexa e custosa em termos de desempenho (na ausência de uma FPU);

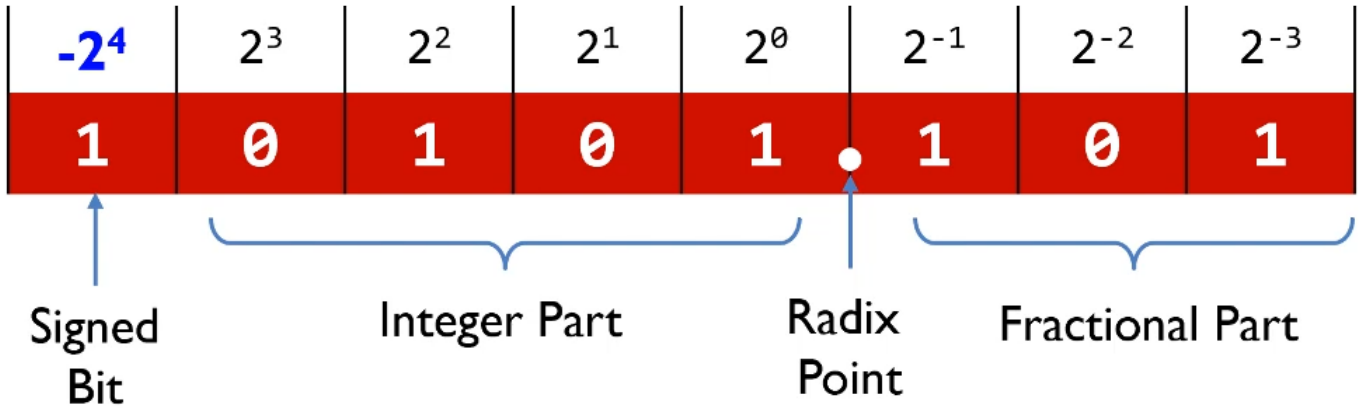


Figura 9 – Aritmética de *fixed-point* para um número real de 4-bits de parte fixa e 3-bits para parte fracionária [6].

Proposta de solução

Diagrama de blocos do sistema

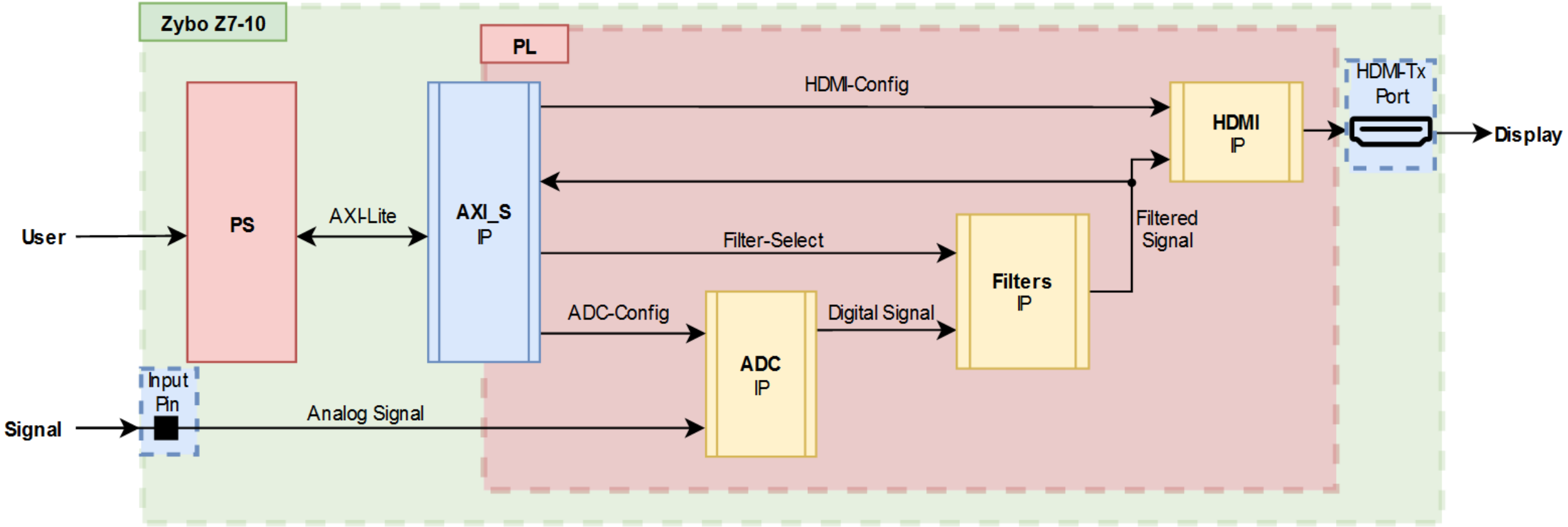


Figura 10 – Diagrama de blocos geral do sistema.

Proposta de solução

Diagrama de blocos do sistema

- **PS – *Processing System:*** para *debug* e interface de configuração para o utilizador;
- **AXI_S – *slave* AXI*:*** interface entre a PS e a PL (*Programming Logic*), recorrendo ao protocolo AXI (*Advanced eXtensible Interface*);
- **ADC IP:** amostragem de um sinal analógico;
- **Filters IP:** aplicação do filtro digital escolhido pelo utilizador ao sinal de entrada;
- **HDMI IP:** apresentação do sinal filtrado numa interface HDMI;

Proposta de solução

Diagrama de blocos do Filters IP

- **Filters IP:** aplicação do filtro digital escolhido pelo utilizador ao sinal de entrada;

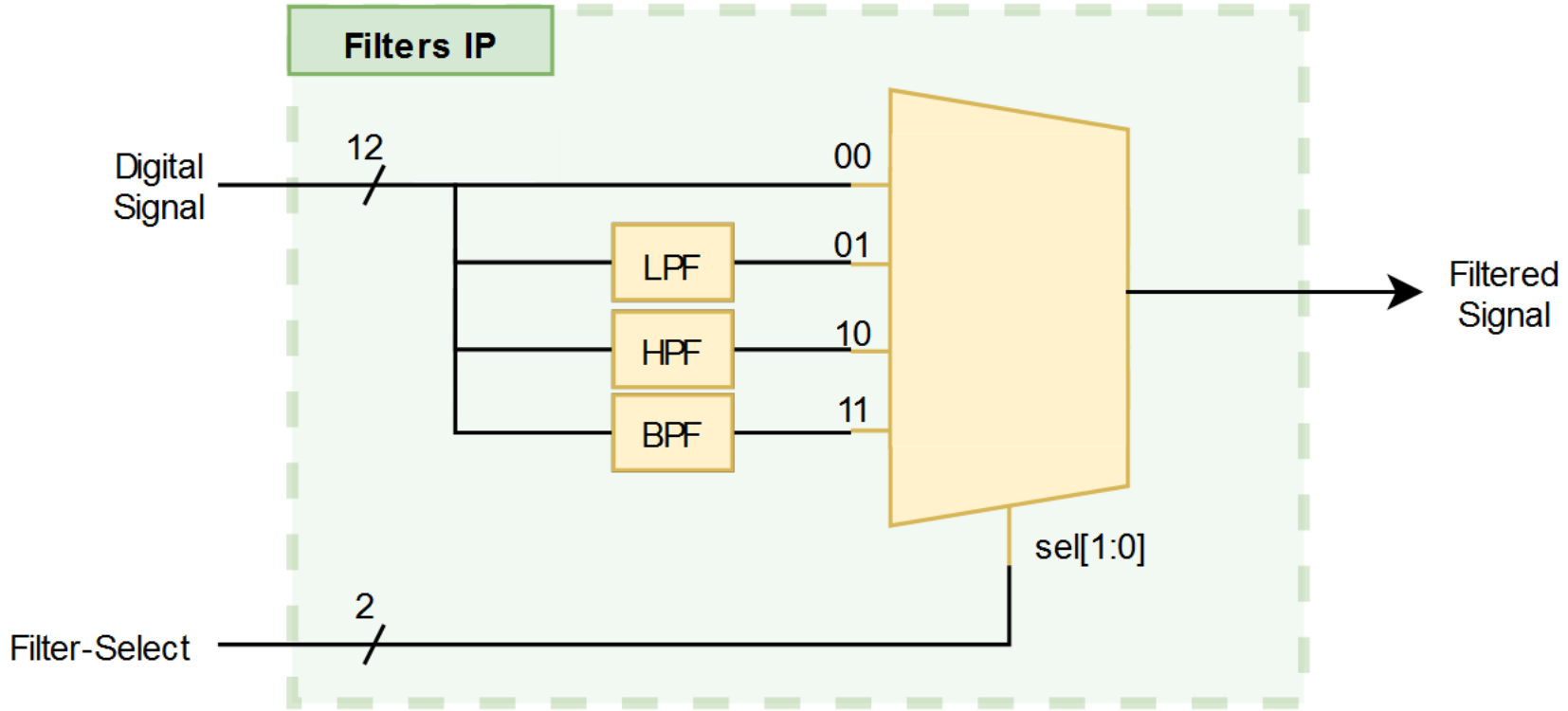


Figura 11 – Diagrama de blocos do Filters IP.

Plano de implementação

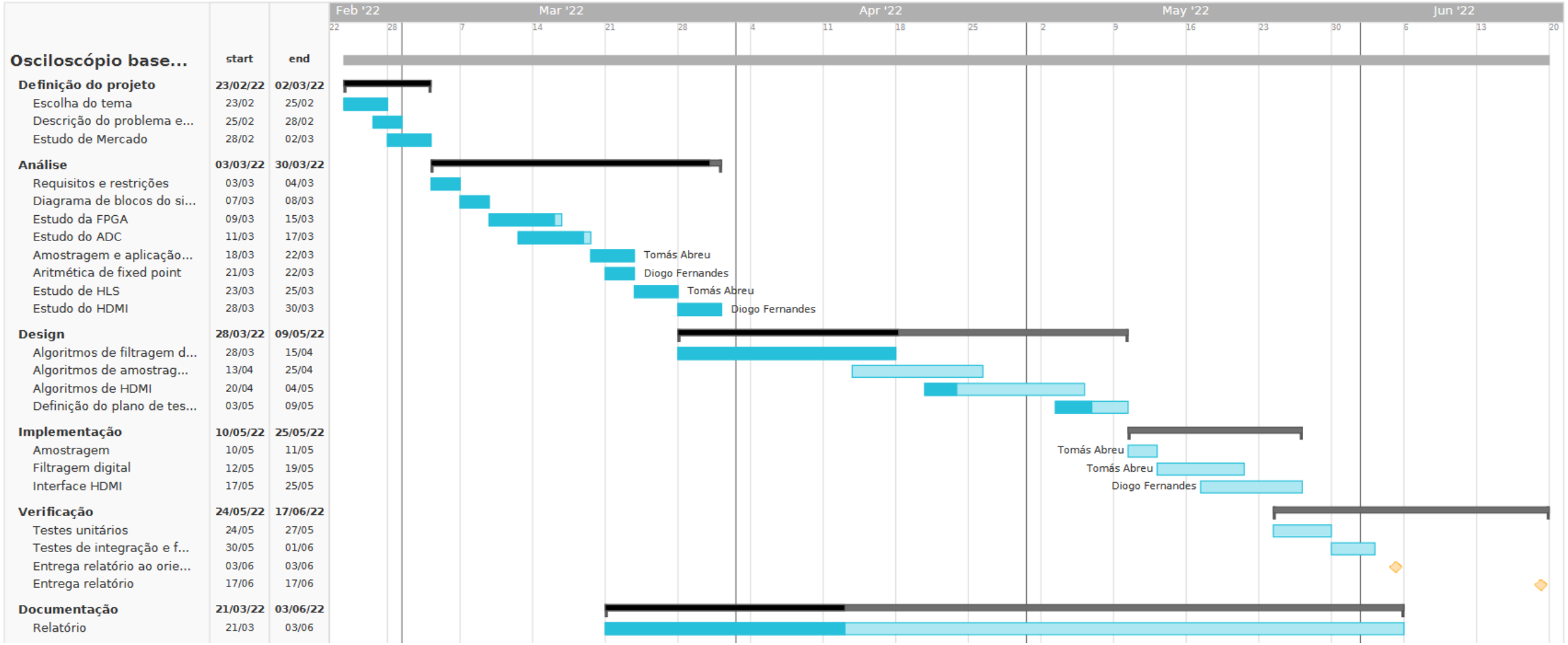
1. Estudo dos filtros digitais e dos seus coeficientes;
2. Implementação de filtros digitais na STM32 (filtro passa-baixo, passa-alto e passa-banda);
3. Desenvolvimento de aplicação *bare metal* na Zybo para interface com o utilizador;
4. Estudo e desenvolvimento de uma interface HDMI na Zybo;
5. Implementação da amostragem e filtros digitais recorrendo a HLS;
6. Integração e verificação do sistema;
7. Análise do *hardware* gerado recorrendo a HLS;

Plano de testes

1. Testes dos filtros digitais (na STM32);
2. Teste da interface HDMI;
3. Teste do ADC;
4. Teste da amostragem e filtragem digital (na Zybo);
5. Teste da aplicação de interface com o utilizador;

Calendário de tarefas

Diagrama de Gantt



Tarefas realizadas

Estudo do número de coeficientes de um Filtro FIR

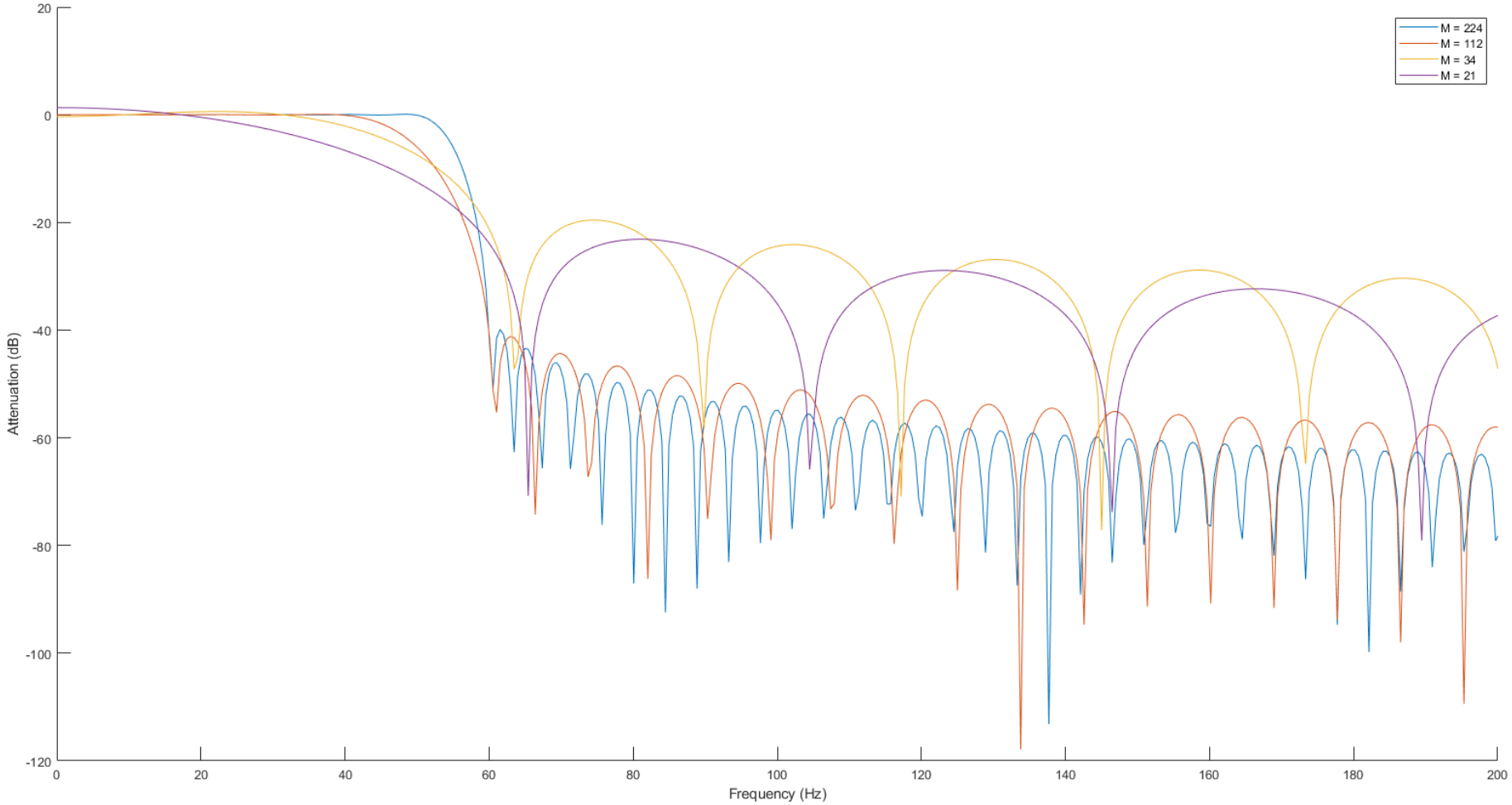


Figura 12 - Resposta em frequência de um LPF para diferentes ordens.

Tarefas realizadas

Desenho e cálculo de um filtro FIR

```
%------------- Low Pass Filter (LPF)
% sampling frequency [Hz]
fsamp = 1000;
% stopband and passband frequencies [Hz]
fcuts = [25 50];
% ripples
devs = [0.01 0.01];

% low pass filter
mags = [1 0];

% get kaiser window
[n,Wn,beta,ftype] = kaiserord(fcuts,mags,devs,fsamp);
% calculate coefficients
hh = fir1(n,Wn,ftype,kaiser(n+1,beta),'noscale');
```

Figura 13 - Algoritmo para Desenho de um LPF em Matlab.

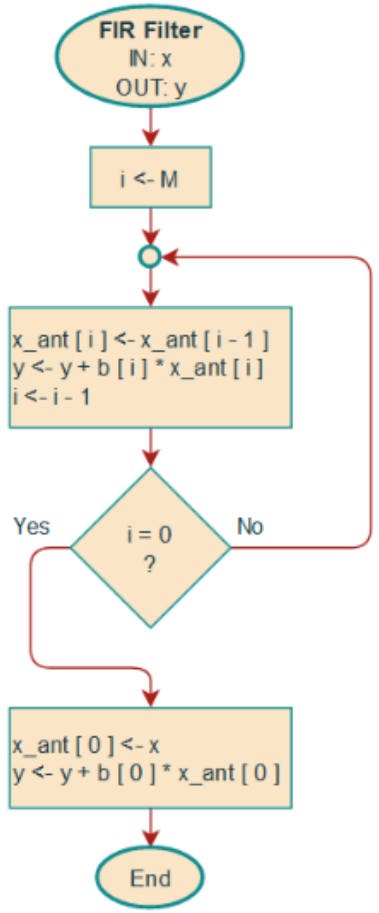


Figura 14 - Algoritmo de um filtro FIR.

Tarefas realizadas

Resposta em frequência de um filtro FIR passa-baixo

- Frequência de corte do filtro 50 Hz;
- Saída do filtro (a azul) para sinais de entrada (a amarelo) de frequência:

(a) **20 Hz** (b) **40 Hz** (c) **100 Hz**

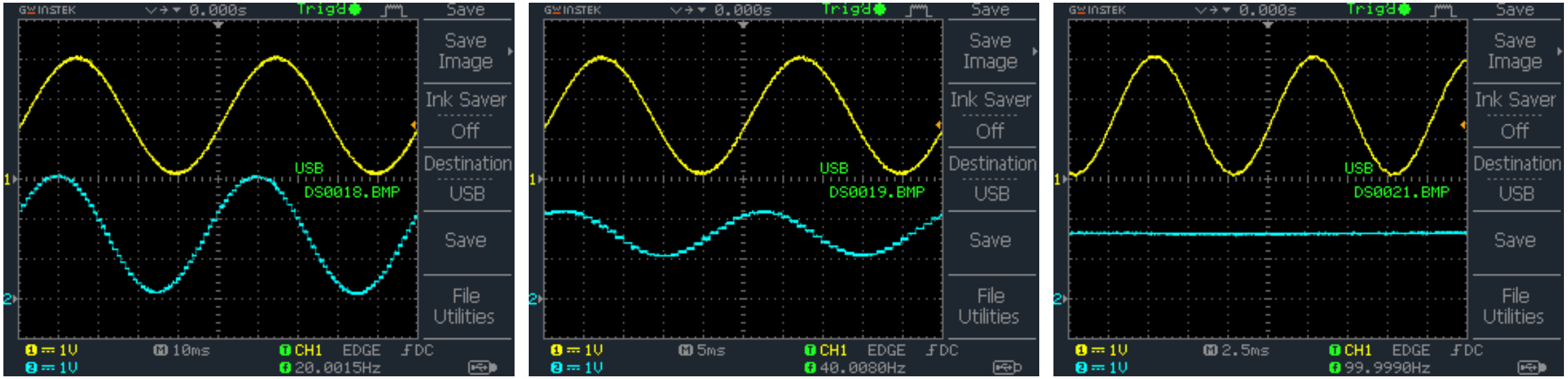


Figura 15 – Aplicação de um filtro passa-baixo a um sinal de entrada sinusoidal (a amarelo) com frequência:

(a) 20 Hz; (b) 40 Hz; (c) 100 Hz.

Tarefas realizadas

Resposta em frequência de um filtro FIR passa-alto

- Frequência de corte do filtro 50 Hz;
- Saída do filtro (a azul) para sinais de entrada (a amarelo) de frequência:

(a) **30 Hz** (b) **60 Hz** (c) **100 Hz**

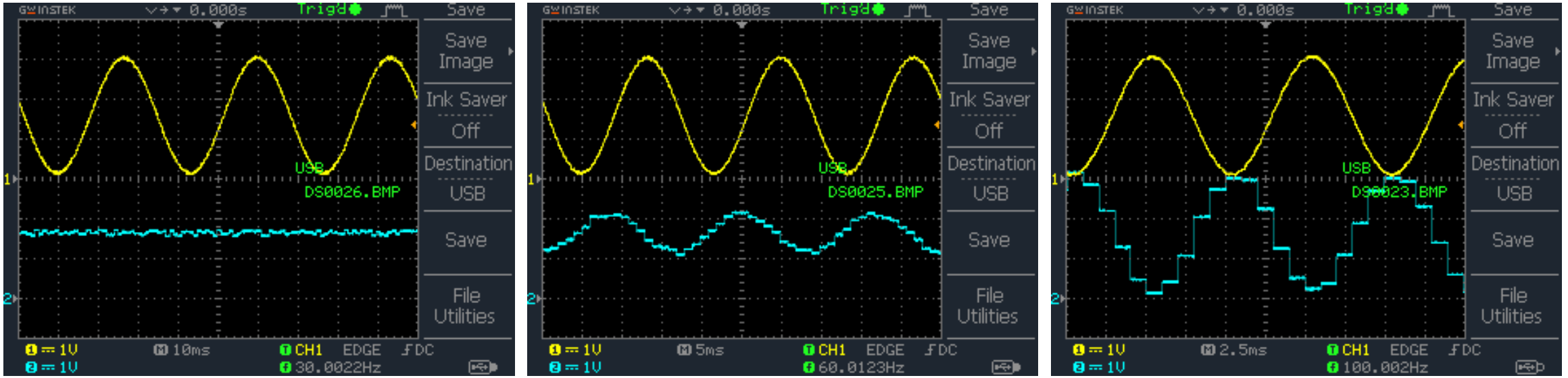


Figura 16 – Aplicação de um filtro passa-alto a um sinal de entrada sinusoidal (a amarelo) com frequência:

(a) 30 Hz; (b) 60 Hz; (c) 100 Hz.

OBRIGADO!

QUESTÕES?

Referências bibliográficas

1. Alan V. Oppenheim, Ronald W.Schafer, John R. Buck, Discrete-Time Signal Processings, 2nd ed. New Jersey: Prentice Hall, 1999
2. D. H. Sarah L. Harris, Digital Design and computer architecture: RISC-V edition. Morgan Kaufmann, 2021, ch. 4
3. Digilent, "Zybo Z7," acedido em 16 abril 2022. [Online]. Disponível em: https://digilent.com/reference/programmable-logic/zybo-z7/start
4. Ryan Kastner, Janarbek Matai, and Stephen Neuendorffer, "Parallel programming for fpgas," 2018, acedido em 1 março 2022. [Online]. Available: https://kastner.ucsd.edu/wp-content/uploads/2018/03/admin/pp4fpgas.pdf
5. R. Oshana, DSP For Embedded And Real-Time Systems. Elsevier, 2012, ch. 1
6. Dr. Yifeng Zhu, Embedded Systems with ARM Cortex-M Microcontrollers in Assembly Language and C, 3rd ed. : E-Man Press LLC, 2017