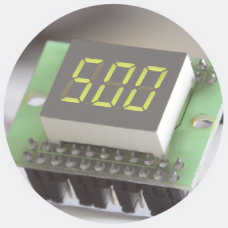
INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES






Aplicação das FPAA na realização de sistemas de ordem fraccionária

AMÉRICO ANTÓNIO GOMES SÃO ROQUE
Novembro de 2015

POLITÉCNICO
DO PORTO

APLICAÇÃO DAS FPAA NA REALIZAÇÃO DE SISTEMAS DE ORDEM FRACCIONÁRIA

Américo António Gomes São Roque
(Licenciado)

Dissertação para obtenção do grau de Mestre em

Engenharia Electrotécnica e de Computadores

Departamento de Engenharia Electrotécnica
Mestrado em Engenharia Electrotécnica e de Computadores
Área de especialização de Automação e Sistemas
Instituto Superior de Engenharia do Porto

2015

Dissertação realizada sob a orientação do
Professor Doutor Ramiro de Sousa Barbosa,
do Departamento de Engenharia Electrotécnica
Instituto Superior de Engenharia do Porto

Departamento de Engenharia Electrotécnica
Mestrado em Engenharia Electrotécnica e de Computadores
Área de especialização de Automação e Sistemas
Instituto Superior de Engenharia do Porto
2015

"Estou convencido de que cerca de metade do que separa os empreendedores de sucesso daqueles mal sucedidos é a pura perseverança."

Steve Jobs

À minha Esposa e Filho.

Agradecimentos

A realização desta dissertação só foi possível graças à colaboração e contributo, de forma direta e indireta, de várias pessoas às quais não posso deixar de dedicar algumas palavras de reconhecimento.

O maior agradecimento é devido à minha esposa e filho pelo apoio, compreensão e encorajamento que sempre manifestaram. É a eles que dedico este trabalho.

Agradeço ao meu orientador, Professor Doutor Ramiro de Sousa Barbosa pela forma como orientou o meu trabalho. O seu apoio, incentivo e disponibilidade foram constantes ao logo de toda orientação. A sua experiência, competência e profissionalismo contribuíram para o cumprimento dos objetivos propostos.

Um agradecimento à minha família pelo suporte e motivação que sempre me proporcionaram ao longo deste percurso.

Gostaria de agradecer ao ISEP as condições disponibilizadas para a realização deste trabalho e a todo o corpo docente do departamento de Engenharia Eletrotécnica da área de automação e sistemas pelo apoio dado ao longo do mestrado.

Agradeço também à Nanium, SA, em particular ao Engenheiro Jorge Rodrigues pela disponibilidade, motivação e compreensão ao longo de todo o mestrado.

Resumo

Esta tese de dissertação tem como principal objetivo a implementação de controladores fracionários utilizando diapositivos analógicos FPAA (*Field Programable Analog Array*). Embora estes dispositivos já não sejam um tecnologia recente, não tiveram grande aceitação comercial, daí não ter sido grande a sua evolução nesta última década. Mas para a elaboração de alguns circuitos analógicos, nomeadamente filtros, amplificadores e mesmo controladores PID (Proporcional-Integrativo-Derivativo) analógicos torna-se numa ferramenta que pode facilitar o projeto e implementação.

Para a realização deste estudo, utilizou-se a placa de desenvolvimento da Anadigm AN231K04-DVLP3 juntamente com o software disponibilizado pela mesma empresa, o AnadigmDesigner2. Para a simulação e observação dos resultados foi utilizada a DAQ (*Data Acquisition*) Hilink da Zelton juntamente com o software Matlab.

De forma a testar a implementação dos controladores fracionários nas FPAA foram realizados alguns circuitos no software e enviados para a FPAA comparando os resultados obtidos na simulação com os visualizados no osciloscópio. Por último foi projetado um controlador $PI^{\lambda}D^{\mu}$ recorrendo aos métodos de aproximação inteira descritos neste documento implementados na FPAA recorrendo ao uso de filtros de primeira e segunda ordem.

Palavras-Chave

Cálculo Fracionário, Controlo de ordem Fracionária, Fractor, Fractância, PID, FPAA, Hilink.

Abstract

This dissertation has as a main goal the implementation of fractional controllers by using FPAA analogical devices.
Although these devices are not a recent technology, they did not have great commercial acceptance yet, and their evolution during this decade has not been significant. However, in the creation of some analogical circuits, namely filters, amplifiers and even PID analogical controllers, it becomes a tool that can facilitate its design and implementation. For the realization of the study, it was used the development board AN231K04-DVLP3 together with the software provided by the same company, the AnadigmDesigner2. We used also the DAQ Hilink from Zelton together with the software Matlab for the simulation and results.
In order to test the implementation of the fractional controllers in the FPAA we have created some circuits in the software and sent them to the FPAA while comparing the results from the simulation with the ones displayed in the oscilloscope. Lastly, we have designed a $PI^{\lambda}D^{\mu}$ controller by using the approach methods described in this document and implemented in the FPAA using first and second order filters.

Keywords
Fractional Calculus, Fractional-Order Control, Fractor, Fractance, PID, FPAA, Hilink.

Indíce

Indíce de Figuras

Indíce de Tabelas

Acrónimos

ADC	*Analog to Digital Converter*
AMPOP	Amplificador Operacional
ASIC	*Application Specific Integrated Circuit*
bit	*Binary digit*
CAB	*Configurable Analog Block*
CAD	*Computer Aided Design*
CAM	*Configurable Analog Module*
CC	Corrente ContÃnua
CFE	*Continued Fraction Expansion*
CMOS	*Complementary Metal-Oxide-Semiconductor*
CRONE	*Commande Robuste d'Ordre Non Entier*
CT	*Continuos Time*
DAQ	*Data Acquisition*
dpASP	*dynamically programmable Analog Signal Processor*
EEPROM	*Electrical Erasable Programmable Read Only Memory*
EPAC	*Electrically Programmable Analogue Circuit*
FIR	*Finite Impulse Response*
FOPID	PID de ordem fracionária
FPAA	*Field Programable Analog Array*
FPGA	*Field Programable Gate Array*
IIR	*Infinite Impulse Response*
IMP	*International Microelectronics Products*
IPMC	*Ionic polymer-metal composites*
ispPAC	*In-System Programmable Analog Circuits*
LSI	*Large Scale Integration*
LUT	*Look-Up Table*

MOSFET	*Metal Oxide Semiconductor Field Effect Transistor*
OTA	*Operational Transconductance Amplifier*
PC	Computador Pessoal
PID	Proporcional-Integrativo-Derivativo
PLC	*programmable logic controller*
PLD	*Programmable Logic Device*
PSE	*Power Series Expansion*
PSoC	*Programmable System-on-Chip*
RASP	*reconfigurable analog signal processor*
S/C	*Switched Capacitor*
SAR	*Successive Approximation Register*
SRAM	*Static Random Access Memory*
TRAC	*Totally Reconfigurable Analog Circuit*
VMR	*Voltage Main Reference*

1. INTRODUÇÃO

A tecnologia de sistemas eletrónicos com *hardware* reconfigurável é uma área de ponta tecnológica que envolve não só o hardware mas também o software e pode ser utilizado em diversas áreas tais como engenharia biomédica, aeroespacial, entre outras. Por outro lado, a utilização do cálculo fracionário como ferramenta para controlo de processos tem um campo de aplicação muito vasto. Neste contexto a implementação de sistemas fracionários em FPAA representa um desafio de implementação.

1.1 CONTEXTUALIZAÇÃO

O uso de tecnologias de circuitos integrados com capacidade de reconfiguração surgiu há pouco mais de 20 anos. Porém, até recentemente apenas dispositivos digitais como os PLDs (*Programmable Logic Devices*) e os FPGAs (*Field Programable Gate Arrays*) estiveram disponíveis para o projeto de sistemas com a capacidade de reconfiguração. No final dos anos 90 novos dispositivos trouxeram esta capacidade para o mundo dos dispositivos analógicos comerciais, especialmente na forma dos chamados FPAAs. Estes dispositivos associados às correspondentes ferramentas de projeto, assim como seus equivalentes digitais, começam a trazer flexibilidade e rapidez no projeto de inúmeras aplicações da eletrónica analógica.

A tecnologia de sistemas eletrónicos com hardware reconfigurável é uma área de ponta tecnológica. Adicionalmente, as soluções analógicas têm-se tornado cada vez mais competitivas com circuitos digitais para aplicações de alta velocidade, densidade e baixo consumo em processamento de sinais de baixa precisão.

Uma vantagem importante dos circuitos digitais integrados tem sido a relativa facilidade de projeto quando comparados com os circuitos analógicos. Particularmente desde que o projeto científico de circuitos é adaptável à automação.

Na altamente competitiva indústria da eletrónica, a aplicação de técnicas CAD ao desenho de circuitos digitais integrados conduziu a ciclos de projeto mais curtos, aliviando algumas das pressões de tempo sentidas por quem desenvolve produtos comerciais.

Devido à grande variedade de funções necessárias nos sistemas eletrónicos e à complexidade dos sinais (frequência, tempo, nível de sinal, ruído, etc), o projeto de sistemas analógicos é muito especializado e suportado por um conjunto de diversas ferramentas CAD (*Computer Aided Design*) que são mais difíceis de integrar do que aquelas necessárias no projeto digital.

Por outro lado, o cálculo de ordem fracionário é uma das ferramentas que encontra aplicações em quase todas as áreas do conhecimento humano. O seu uso teve inicio na década de sessenta e desde então tem sido muitas as aplicações. A primeira aplicação com sucesso de controlo fracionário é reportada por Manabe (1961), no qual foi usada uma equação de ordem fracionária para síntese de controladores. Oustaloup (1975) seguiu-se e a sua aplicação consistiu na utilização do controlo fracionário para o controlo de sistemas de laser colorido. Em 1991 reporta a aplicação do controlo fracionário no controlo de sistemas dinâmicos e demonstra o desempenho superior do CRONE (*Commande Robuste d'Ordre Non Entier*) quando comparado com o PID. Por sua vez, Poldubny (1999) apresenta a adaptação do controlador PID fracionário o qual designou por $PI^{\lambda}D^{\mu}$, com um integrador de ordem λ e um diferenciador de ordem μ.

1.2 OBJETIVO

O objetivo desta tese de dissertação consiste na estudo e implementação de um controlador $PI^{\lambda}D^{\mu}$ num dispositivo analógico reconfigurável FPAA da Anadigm. A implementação deverá ser feita usando uma placa de desenvolvimento AN231E04 da Anadigm, em conjunto com a placa de aquisição de sinal HILINK da Zeltom. Para a simulação e teste deve ser usado o software Matlab/Simulink.

1.3 CALENDARIZAÇÃO

A Tabela 1.1 apresentada abaixo representa o conjunto de tarefas realizadas ao longo da execução deste projeto.

Tabela 1.1: Calendarização da Tese

Tarefa		MÊS													
Nº	Descrição	2014			2015										
		10	11	12	1	2	3	4	5	6	7	8	9	10	11
1	Preparação do template em latex														
2	Elaboração do relatório														
2.1	Introdução														
2.2	FPAA														
2.3	Cálculo Fracionário														
2.4	Arquitetura do sistema														
2.5	Implementação														
2.6	Conclusão														
3	Ensaios e testes														
4	Apresentação														

1.4 ORGANIZAÇÃO DO RELATÓRIO

Esta dissertação está dividida em seis capítulos.

No Capítulo 1, faz-se uma introdução à tese de dissertação onde é abordado o enquadramento do trabalho, objetivos que se pretendem alcançar e organização do presente documento.

Já no Capítulo 2, é feita uma introdução aos FPAA, considerando a sua evolução, dispositivos comerciais, arquitetura e tecnologia.

No Capítulo 3, é feito um estudo sobre os sistemas fracionários, a sua história e representação. São também apresentadas as principais formas de aproximação dos sistemas de ordem fracionária à ordem inteira e formas de implementação. Neste capítulo, apresenta-se ainda duas formas analógicas de implementação, designadas de fractor e fractância.

No Capítulo 4, tendo em vista o ensaio prático dos sistemas fracionários, é feita a descrição da arquitetura do sistema, bem como de todos os componentes nele utilizados, tais como: a placa de aquisição de sinais Hilink da Zeltom e a placa de desenvolvimento da Anadigm AN231K04-DVLP3. É também feita a apresentação da FPAA utilizada nesta placa, a AN231E04.

No Capitulo 5, são estudados vários tipos de controladores PID fracionários usados no controlo

da velocidade de um motor de corrente contínua.

Finalmente, no Capítulo 6 são apresentadas as conclusões relativamente aos resultados obtidos e indicações de trabalhos futuros.

2. *Field Programmable Analog Array (FPAA)*

A tecnologia de sistemas eletrónicos com hardware reconfigurável é uma área de ponta tecnológica que envolve não só o *hardware* mas também o software e pode ser utilizado em diversas áreas tais como engenharia biomédica, engenharia aeroespacial, entre outras. Na eletrónica digital o aparecimento dos FPGAs, despertaram bastante interesse académico e comercial. Já na eletrónica analógica os FPAA não tiveram tanto sucesso comercial mas foram âmbito de muitos artigos e desenvolvimentos académicos conseguindo chamar a atenção de vários grupos de investigação, graças à sua flexibilidade, possibilidade de simplificar os processos de desenvolvimento e acelerar a prototipagem de circuitos analógicos.

Neste capítulo apresenta-se os FPAAs considerando a sua evolução, dispositivos comerciais, arquitetura e tecnologia.

2.1 INTRODUÇÃO

Devido à rápida evolução dos circuitos eletrónicos e ao aumento do nível de integração (*Large Scale Integration* - LSI), o projeto e desenvolvimento dos circuitos integrados de aplicação específica (*Application Specific Integrated Circuit*s - ASICs) torna-se mais caro e bastante demorado. Isto faz com que haja uma necessidade crescente de circuitos de *hardware* reconfiguráveis, capazes de ser convertidos em diversos tipos de aplicações num curto espaço de tempo, bastando para tal um software de programação. Foi neste contexto que apareceram as FPGAs, causando um grande impacto no desenvolvimento de circuitos digitais personalizados, possibilitando reconfigurar o hardware com facilidade. Estes circuitos possuem elementos lógicos e interconexões configuráveis. Desde o seu aparecimento as FPGAs têm sido utilizadas maioritariamente em laboratórios e protótipos de hardware digital, permitindo o teste, depuração e reconfiguração baixando significativamente o tempo e custo de desenvolvimento, permitindo a produção de equipamentos com rápida colocação no mercado e com a facilidade de se poderem fazer atualizações mesmo depois de estarem implementadas. Com o mesmo objetivo foi desenvolvido no domínio da eletrónica analógica um dispositivo reconfigurável denominado de FPAA. Estes dispositivos associados às correspondentes ferramentas de projeto, assim como seus equivalentes digitais, começam a trazer flexibilidade e rapidez no projeto de inúmeras aplicações da eletrónica analógica.

2.2 RESUMO HISTÓRICO

O primeiro conceito de um circuito analógico integrado reconfigurável em campo designado por *Proto-Chip* foi descrito por Sivilotti em 1988. O Proto-Chip era constituído por blocos analógicos reconfiguráveis (*Configurable Analog Block* - CAB) interligados por uma rede em árvore. O seu âmbito de aplicação era a prototipagem analógica de redes neuronais (Sivilotti, 1988). Na sua constituição interna foram usadas portas CMOS (*Complementary Metal-Oxide-Semiconductor*) como elementos de comutação nas linhas de transmissão e foram disponibilizadas SRAM (*Static Random Access Memory*) para armazenamento dos estados dos elementos de comutação, mas não foi disponibilizada memória para armazenamento dos coeficientes do circuito (Gulak, 1995). Na Figura 2.1 está representada a estrutura do Proto-Chip descrito por Sivilotti.

Este tema suscitou muito interesse na comunidade científica e em 1991 Lee e Gulak desenvolveram um FPAA de baixo consumo baseado em técnicas de comutação abaixo do limiar de tensão (*Sub-threshold*) como representado na Figura 2.2. A FPAA era composta por transístores e estes eram controlados por memórias SRAM e usados para ativar elementos de comutação, que estavam conectados aos circuitos, tais como AMPOP (Amplificador

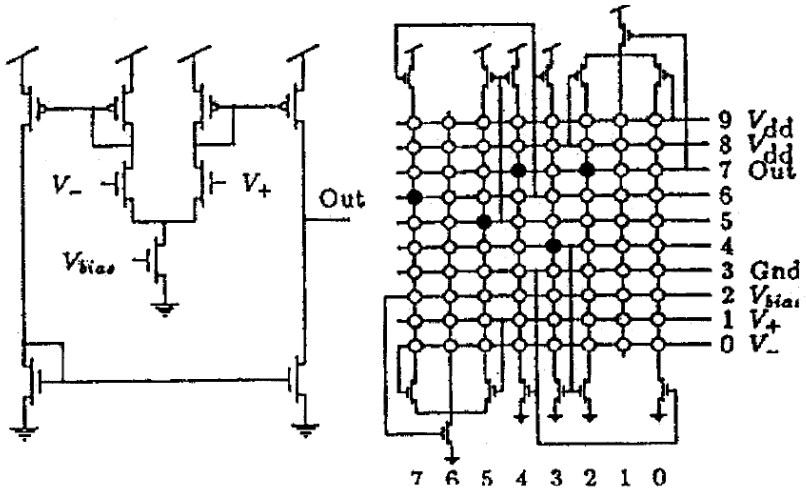
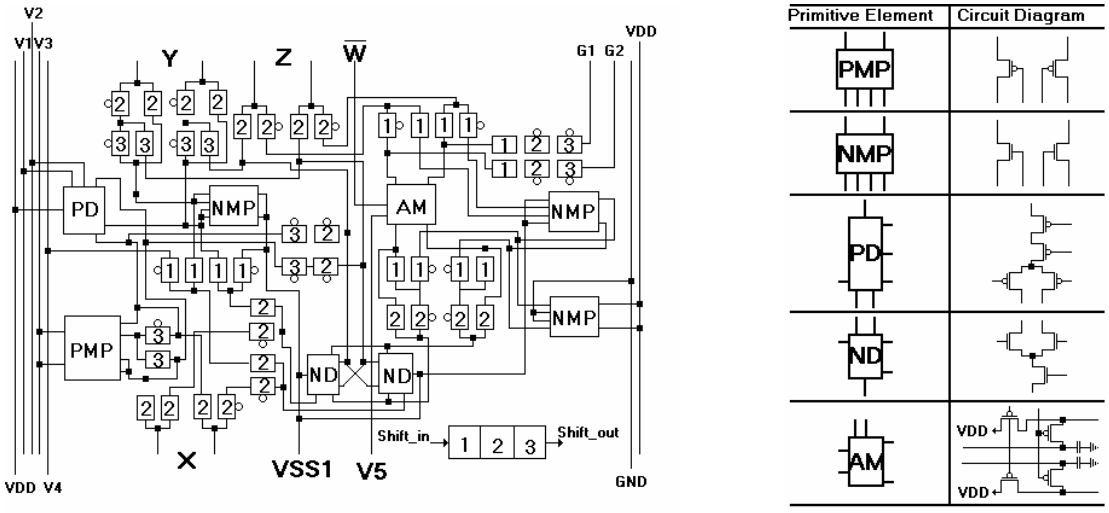


Figura 2.1: FPAA de Sivilotti - 1988 (D'Mello e Gulak, 1998)

Operacional). A CAB era programada por três bits (*Binary digits*) armazenados num Registador de deslocamento *shift register*. Cada bit configura o estado de um conjunto de *switches* que ligam aos elementos primitivos conforme a numeração na Figura 2.2(a). Na Figura 2.2(b) estão representados os elementos primitivos. As chaves controladas por cada bit, estão numeradas como 1, 2 e 3, os quais representam a posição do bit no *shift register* de programação (Lee e Gulak, 1991).



(a) Estrutura do CAB

(b) Elementos Primitivos

Figura 2.2: FPAA de Lee e Gulak - 1991 (Lee e Gulak, 1991)

Em 1992 Lee e Gulak, aprofundaram o seu conceito e apresentaram uma nova arquitetura baseada estrutura desenvolvida no trabalho anterior, trocando os transístores por MOSFETs (*Metal Oxide Semiconductor Field Effect Transistors*) usados como transcondutores na interconexão dos elementos primitivos (Figura 2.3). A principal vantagem sobre a versão

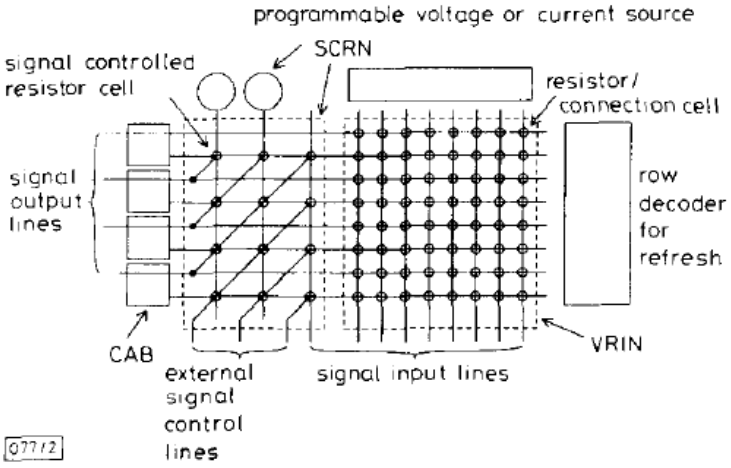


Figura 2.3: FPAA de Lee e Gulak - 1992 (Lee e Gulak, 1992)

anterior era o facto dos elementos de interconexão atuarem também como resistências que permitiam a configuração de blocos de ganho programáveis. Na sua constituição tinham dois barramentos cruzados de interconexão, um vetor de tensões ou correntes programáveis e um vetor de blocos analógicos configuráveis CABs (Lee e Gulak, 1992).

Em 1995 apresentam uma modificação na topologia dos transcondutores utilizados que foi feita com o fim de diminuir a área ocupada pelo FPAA dando origem a uma nova versão do dispositivo (Lee e Gulak, 1995). Esta estrutura consiste em quatro CABs e oito redes

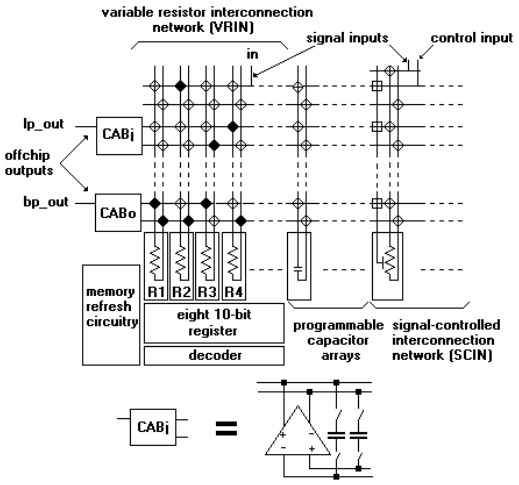
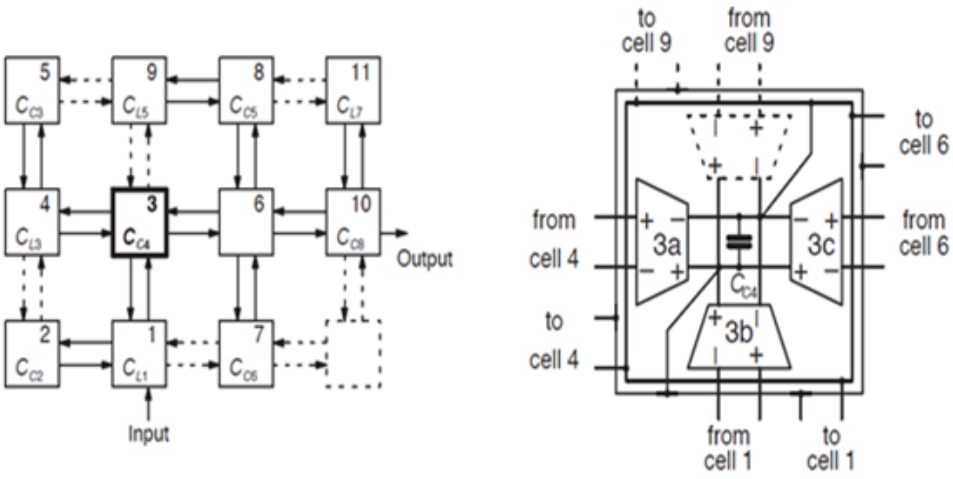


Figura 2.4: FPAA de Lee e Gulak - 1995 (Lee e Gulak, 1995)

de interligações contendo resistências variáveis. Cada CAB é composto por um AMPOP em montagem diferencial com condensadores que podem ser conectados ou desconectados do AMPOP por intermédio de *switches* conforme podemos verificar na Figura 2.4.

Ainda em 1995, Pierchala *et al.* apresentaram uma ideia semelhante à de Lee e Gulak, um circuito analógico programável eletronicamente denominado de EPAC (*Electrically*



(a) Estrutura do *Array* (b) Estrutura do CAB

Figura 2.5: FPAA de Pierzchala (EPAC) - 1995 (Pierchala *et al.*, 1995)

Programmable Analogue Circuit), representado na Figura 2.5.

Apesar do prototipo desenvolvido apresentar somente um circuito integrador, foi no entanto proposta uma estrutura de interligação local para assim evitar a limitação da largura de banda dos *switches* no encaminhamento da ligação (Pierchala *et al.*, 1995). A FPAA foi desenhada num *layout* em xadrez tal como representado na Figura 2.5(a).

Lee e Hui apresentaram em 1998 um FPAA também baseado em AMPOPs em montagem diferencial, com condensadores programáveis (Figura 2.6).

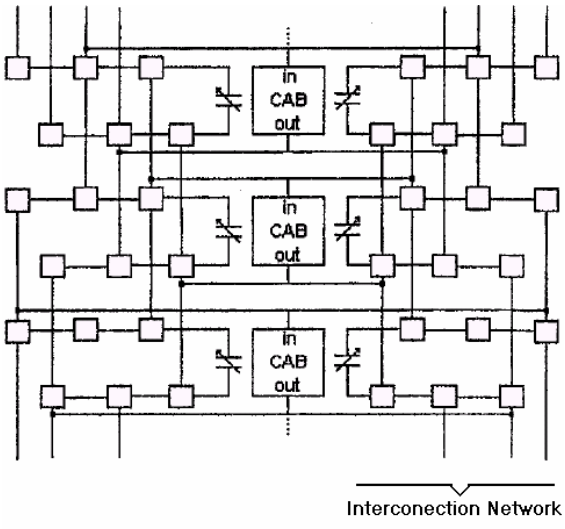


Figura 2.6: FPAA de Lee e Hui - 1998 (Lee e Hui, 1998)

A arquitetura proposta é composta por um conjunto de CABs em que na vizinhança existem dois bancos de condensadores programáveis. Uma rede de interligações permite que os CABs

e condensadores de cada linha sejam conectados aos CABs e condensadores de outras linhas na matriz configurável (Lee e Hui, 1998).

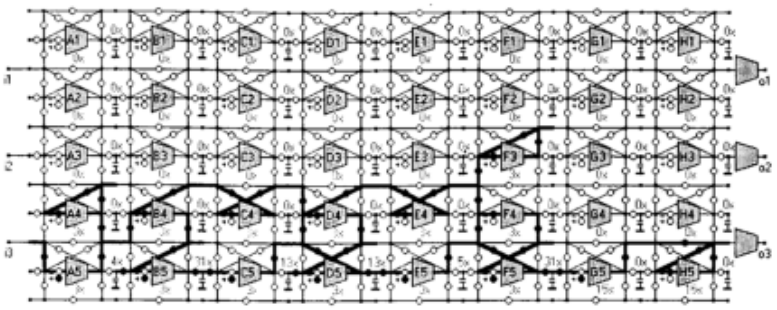
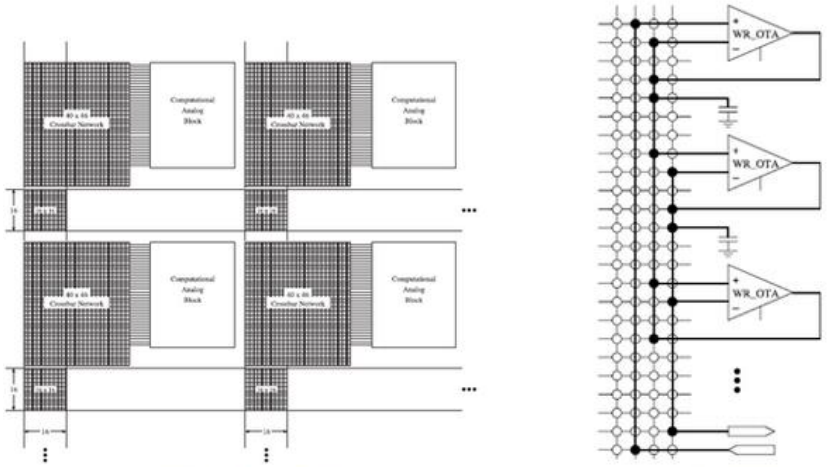


Figura 2.7: FPAA de Pankiewicz - 2002 (Pankiewicz *et al.*, 2002)

Em 2002, Pankiewicz *et al.* apresentaram uma FPAA de tempo contínuo (*Continuos Time* - CT) desenhada para aplicações de filtros ativos, usando um amplificador operacional programável de transcondutância (*Operational Transconductance Amplifier* - OTA) e uma matriz de condensadores programáveis (Figura 2.7). O dispositivo dispunha de 40 CABs e um condensador em cada OTA (Pankiewicz *et al.*, 2002).



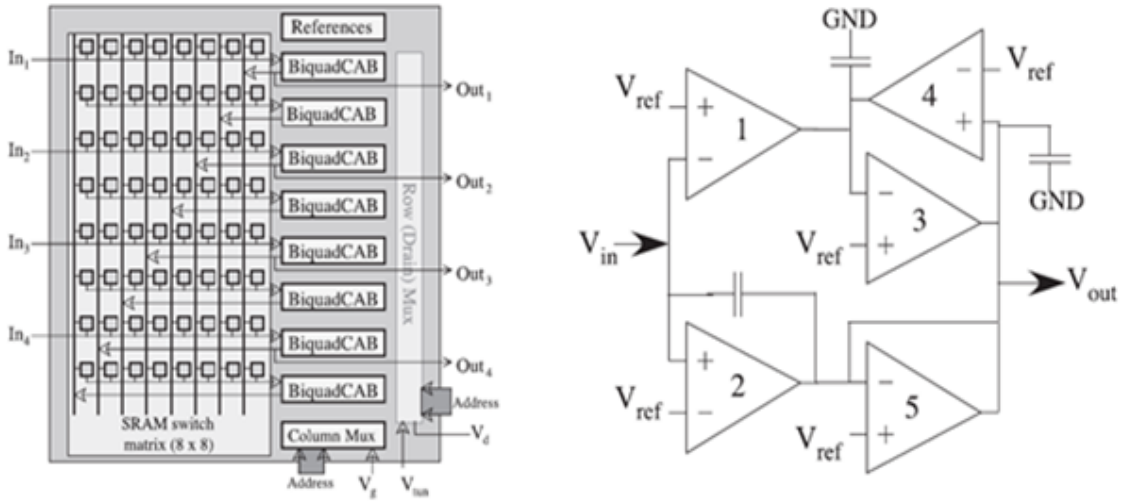
(a) *Routing Network* (b) OTA *Circuit*

Figura 2.8: FPAA de Hall - 2002 (Hall, Hasler e Aanderson, 2002)

Ainda em 2002, Hall, Hasler e Aanderson apresentaram o RASP (*reconfigurable analog signal processor*), representado na Figura 2.8 (Hall, Hasler e Aanderson, 2002).

Mas foi em 2005 que a segunda versão deste dispositivo apareceu com CABs conectados com uma interligação global. As portas flutuantes são usadas como *switches* na rede de interligação de encaminhamento e na sintonia dos elementos ativos. O projeto RASP tenta alcançar a complexidade e flexibilidade semelhante à dos FPGA incorporando elementos de alto nível,

tais como os filtros passa-banda de segunda ordem e um multiplicador de uma matriz de vetores quatro-por-quatro nos seus CABs (Hall *et al.*, 2005).



(a) *Routing Network* (b) OTA *Circuit*

Figura 2.9: FPAA de Hall - 2007 (Hasler e Twigg, 2005)

Em 2007, o mesmo grupo apresentou uma versão melhorada do RASP 2.9. Os CABs complexos foram substituídos por CABs mais simples cada um contendo um filtro de segunda ordem ajustável por portas flutuantes, como mostra a Figura 2.9(b). O dispositivo é conectado através do sistema de encaminhamento em rede usando interruptores de portas flutuantes, representado na Figura 2.9(a).

2.3 DISPOSITIVOS COMERCIAIS

Os fabricantes de semicondutores, já tentaram de algumas formas desenvolver circuitos analógicos reconfiguráveis, mas a estrutura dos dispositivos foi sempre baseada numa coleção de blocos reconfiguráveis ligados por uma rede de interligações. Por sua vez os blocos eram constituídos por componentes genéricos como transístores, resistências, condensadores e amplificadores operacionais. Foram também comercializados dispositivos nas duas tecnologias, tempo discreto e tempo contínuo.

2.3.1 IMP EPAC

Em 1995 a empresa IMP inc lançou no mercado o primeiro dispositivo FPAA designado por EPAC50E10 e de seguida o EPAC50E30. Ambos desenhados em tempo discreto baseado

na tecnologia da condensadores chaveados (Klein, 1996). Na Figura 2.10 está representado o diagrama do EPAC50E10. Este dispositivo era composto por um *multiplexer* analógico,

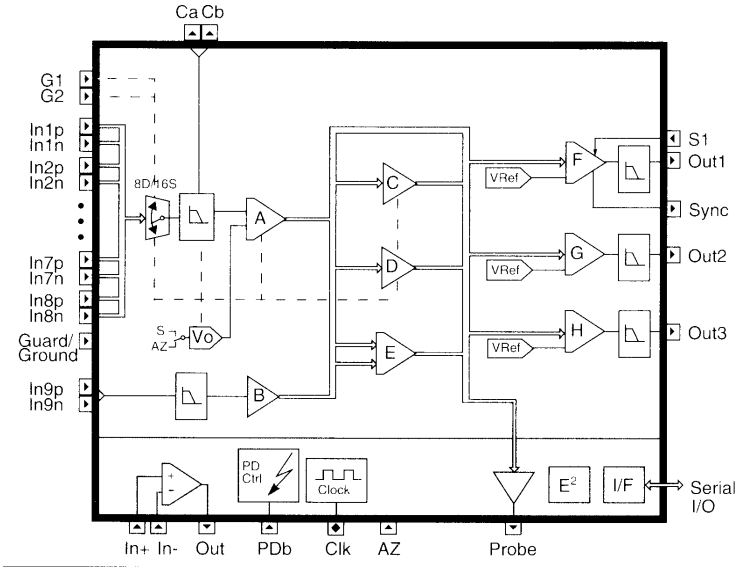


Figura 2.10: EPAC50E10 (Gaudet e Gulak, 1998)

amplificadores programáveis, barramento de interligação e módulos de saída. O *multiplexer* de entrada pode interligar 16 sinais independentes ou 8 diferenciais. O 50E10 era programado usando uma string de 200 bits, a largura de banda estava limitada a 125 kHz, devido à tecnologia de condensadores chaveados. A família de produtos EPAC foi descontinuada, tendo saído de comercialização (Gaudet e Gulak, 1998).

2.3.2 ZETEX TRAC

A Zetex apresentou a família TRAC (*Totally Reconfigurable Analog Circuit*), dos quais o TRAC020LH foi o primeiro. Tratava-se de uma FPAA em tempo contínuo baseado em transístores bipolares que operava até 4 MHz. Este dispositivo consiste em 20 CABs que podem ser interligados localmente (internamente) e de modo global (externamente). O CAB do TRAC permite a ligação de transístores bipolares de maneira a implementar várias funções, tais como a soma, inversão, logaritmo, anti-logaritmo, retificação e ganho. Na Figura 2.11 está representado o diagrama interno deste dispositivo.

2.3.3 MOTOROLA MPAA

Em 1996 a Pilkinton Microelectronics, apresentou um FPAA baseado na tecnologia em tempo discreto, denominado DPAD2, e pouco tempo depois uma versão posterior deste FPAA foi

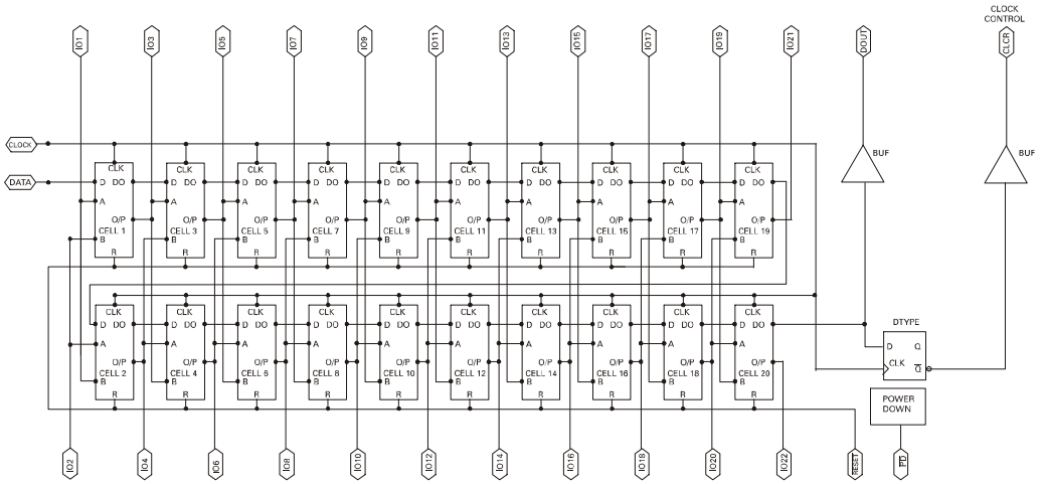


Figura 2.11: Diagrama TRAC020LH (Zetex, 1999)

denominada DPAD3. Em 1997 a Motorola comprou a empresa inglesa e fundou o Motorola Programmable Technologies Center (Clarke, 2000). O DPAD passou a ser comercializado pela Motorola com o nome de MPAA. O MPAA020 foi o primeiro dispositivo lançado pela Motorola

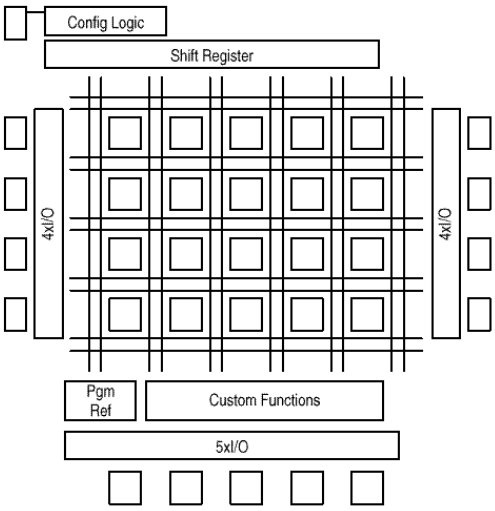
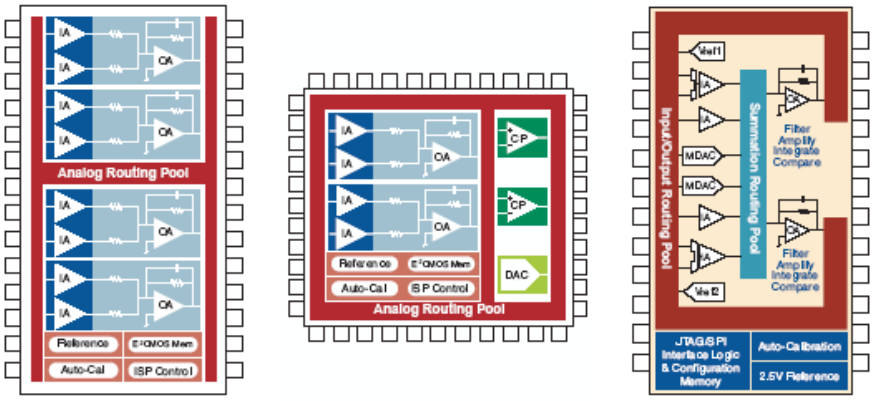


Figura 2.12: Arquitetura MPAA (Gaudet e Gulak, 1998)

com tecnologia CMOS em tempo discreto, a sua arquitetura estava organizada numa matriz de CABs em grupos de 5 distribuídos por 4 linhas, tal como representado na Figura 2.12. A sua configuração era feita usando uma *string* de 619 bits. Cada CAB realizava a função de um filtro de primeira ordem (Gaudet e Gulak, 1998).

2.3.4 LATTICE ISPPAC

A família ispPAC (*In-System Programmable Analog Circuits*) teve origem numa estrutura desenvolvida em 1995 pela IMP (*International Microelectronics Products*) hoje em dia pertencente à Lattice Semiconductors. Uma das principais características dos componentes da família ispPAC é o facto da memória de programação ser do tipo EEPROM (*Electrical Erasable Programmable Read Only Memory*). Os dispositivos ispPAC10, ispPAC20, ispPAC10



(a) ispPAC10 (b) ispPAC20 (c) ispPAC30

Figura 2.13: Lattice ispPAC

da família ispPAC possuem uma grande versatilidade podendo realizar funções analógicas como filtros, somadores, integradores, e amplificadores. Já os componentes ispPAC80, ispPAC81 são filtros programáveis de quinta ordem com topologia definida e não alterável. A programabilidade destes dois últimos componentes restringe-se aos parâmetros (ganho, frequência e fator de qualidade) e ao tipo dos filtros (Butterworth, Chebychev, Elíptico, entre outros). A diferença entre os componentes ispPAC80 e ispPAC81 é apenas na gama de frequência de operação. A Figura 2.13 mostra o diagrama interno dos dispositivos ispPAC10, ispPAC20 e ispPAC10.

2.3.5 CYPRESS PSOC

Por sua vez a Cypress apresentou da família PSoC (*Programmable System-on-Chip*), a série CY8C2XXXX. Os dispositivos PSoC possuem blocos analógicos em tempo contínuo e em tempo discreto, além de elementos digitais e mistos, como microcontrolador, memórias e conversores de dados. A representação em blocos dos componentes da família PSoC é mostrada na Figura 2.14. O microcontrolador do sistema varia, de acordo com a versão do dispositivo, M8C (PSoC 1), 8051 (PSoC 3) ou ARM cortex-M3 (PSoC 5).

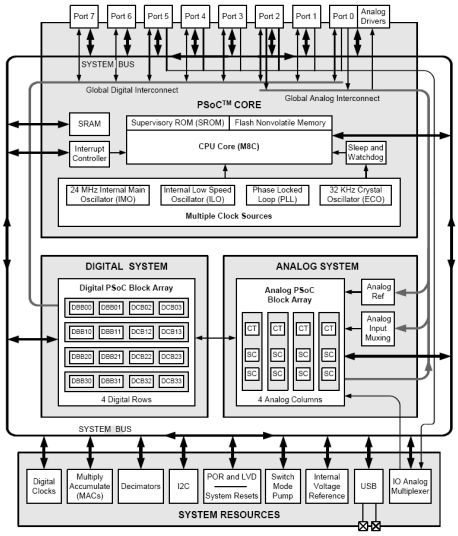


Figura 2.14: PSoC (Imp, 1995)

2.3.6 ANADIGM DPASP

Em janeiro de 2000 Macbeth, que na altura era responsável de desenvolvimento na Motorola junta-se com Ludwig Klingenbeck e funda a Anadigm, FPAA, comprando de imediato o portfólio das FPAA à Motorola. Lançam de seguida o seu primeiro FPAA designado por AN10E40 (Clarke, 2000). São muitas as semelhanças entre o MPAA020 e o atual

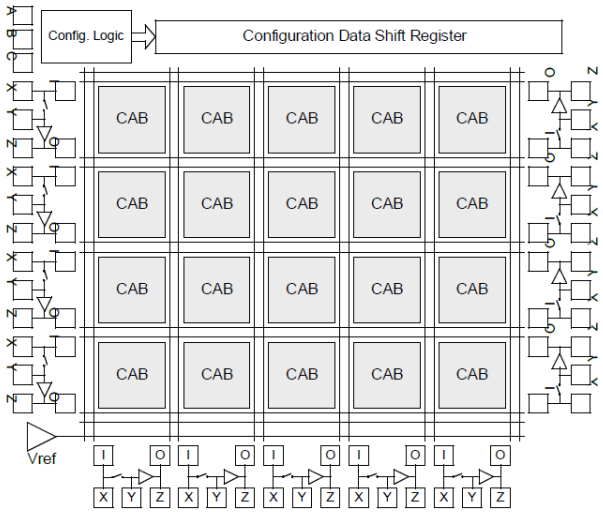


Figura 2.15: Anadigm dpASP AN10E40 (Anadigm, 2000)

N10E40, ambos possuem uma matriz de 4x5 CABs, 13 células de I/O, tem o mesmo esquema de interligações e o software de programação Anadigm Designer apresenta melhorias relativamente ao Easy Analog da Motorola, usado para a programação do MPAA020 (Anadigm, 2000). A representação por blocos do AN10E40 pode ser vista na Figura 2.15.

2.4 ARQUITETURA

A arquitetura de um FPAA é formada por blocos analógicos configuráveis CAB, células de entrada e saída, rede de interligações e registos de memória. A rede de interligações é responsável por fazer a ligação física entre os CAB e as células de entrada e saída, sendo estas o interface do FPAA com os sistemas externos e podem ser compostas por *buffers*, filtros *anti-aliasing* ou *smoothing*, entre outras funções de condicionamento de sinal. Os registos de memória é onde são armazenados os bits de configuração. A Figura 2.16 representa a

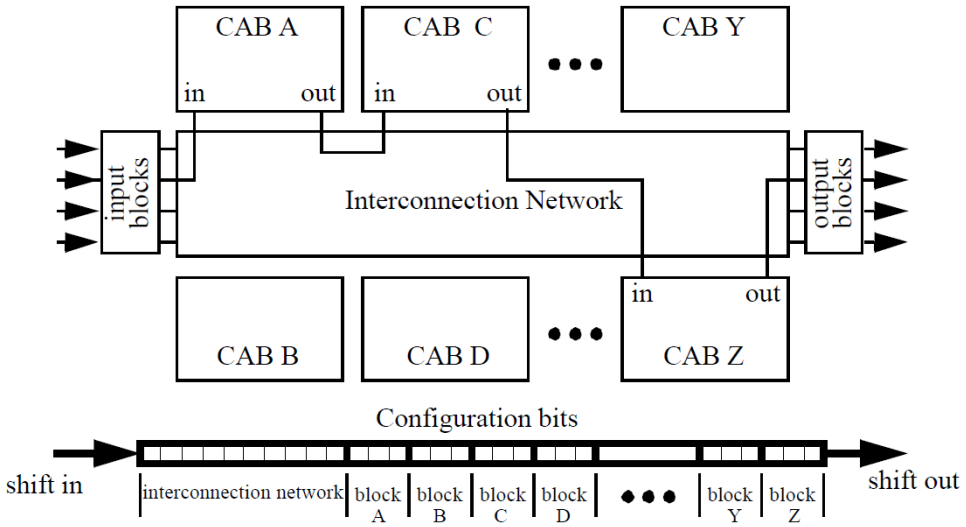


Figura 2.16: Arquitetura de um FPAA genérico (Gaudet e Gulak, 1998)

arquitetura de um FPAA genérico, contendo vários CABs ligados entre si através da rede de interligações. A *string* de configuração é armazenada num *shift register* e os bits configuram as ligações da rede de interligações e a funcionalidade de cada um dos CABs (Gaudet e Gulak, 1998).

2.4.1 CABs

O CAB é o bloco que executa a função num FPAA e pode ser configurado em diversas funções analógicas como amplificação, integração, diferenciação, adição, etc. Cada CAB é constituído por um conjunto de componentes analógicos programáveis, recursos de interligação local e global, blocos de comutação e pelo menos um AMPOP, tal como representado na Figura 2.17. Os componentes dentro da disposição podem ser implementados como simples fios, componentes passivos ou ativos ou partes mais complexas. Em geral os parâmetros programáveis dos CAB são amplificadores de ganho, valores de resistências e condensadores, e ajustar laços locais e globais de realimentação (Balen *et al.*, 2004). Na Figura 2.17 está

representado o diagrama de blocos de um CAB genérico.

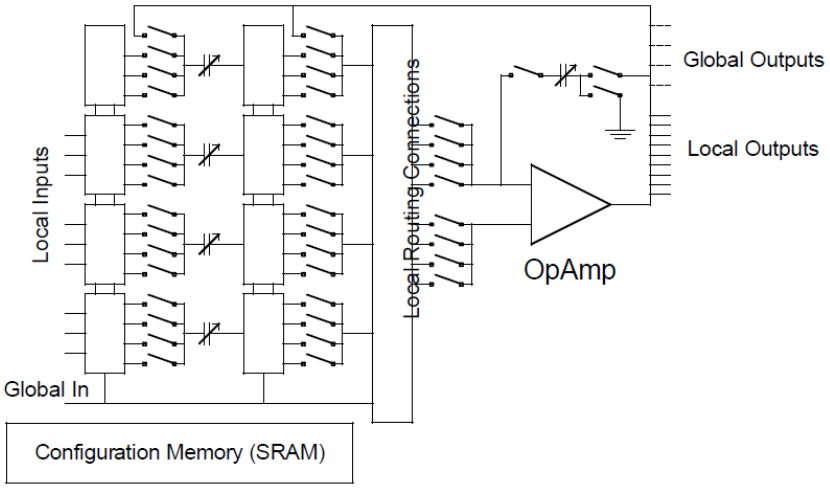


Figura 2.17: Estrutura de um CAB (Anadigm, 2000)

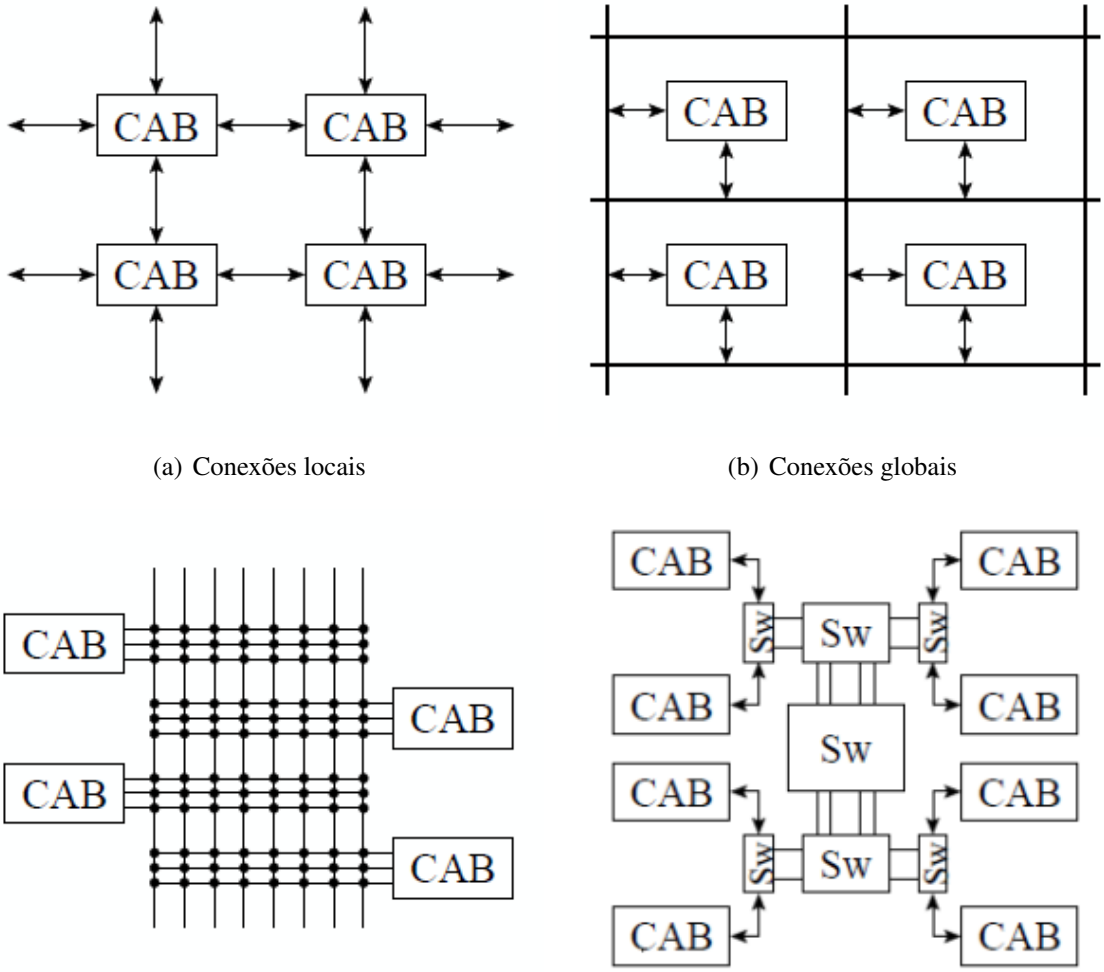
2.4.2 ESTRUTURA DE ENCAMINHAMENTO

A estrutura de encaminhamento, tal como os CAB, também desempenham um papel muito importante nos dispositivos FPAA, é através da configuração destas que o utilizador consegue aceder aos recursos disponíveis no dispositivo. De uma forma idêntica aos FPGA a implementação de um circuito analógico num FPAA necessita de um elevado número de interruptores que são programados de acordo com o circuito desejado.

Foram vários os tipos de estruturas de encaminhamento propostas ao longo dos anos, como são exemplo as conexões locais (Figura 2.18(a)), globais (Figura 2.18(b)) e em barramento (Figura 2.18(c)), e em árvore (Figura 2.18(d)).

As características principais das estruturas de encaminhamento são o grau de conetividade, integridade e potência do sinal e área utilizada. A arquitetura ideal destas estruturas apresentaria um grau de conetividade máximo de tal forma que poderia ligar os sinais provenientes de um CAB a qualquer outro CAB no sistema, os sinais não apresentariam degradação, a área ocupada seria mínima e a potência consumida zero. Dado que o seu tamanho ocupa grande parte da área da FPAA, a sua otimização tem vindo a ser alvo de vários estudos académicos.

Existem três tipos de estruturas de encaminhamento que estabelecem as ligações entre os componentes no mesmo CAB, na mesma coluna e em colunas diferentes, os barramentos que efetuam estas ligações são designados por locais, verticais e horizontais, respetivamente. Entre os CAB e os barramentos existem os interruptores que consoante a aplicação são configurados



(a) Conexões locais

(b) Conexões globais

(c) Conexões em barramento

(d) Conexões em árvore

Figura 2.18: Estruturas de encaminhamento (Hall *et al.*, 2005)

para estabelecem as ligações pretendidas, e são designados também como locais, verticais e horizontais consoante o barramento onde se encontrarem.

As ligações entre os interruptores e o barramento também podem ser estabelecidas de três formas diferentes:

- Tipo 1: Intra - CAB (Local) ligações entre componentes no mesmo CAB que usam interruptores do barramento local (Figura 2.19 Type 1).
- Tipo 2: Inter - CAB (Vertical) ligações entre componentes de CABs que usam interruptores dos três barramentos (Figura 2.19 Type 2).
- Tipo 1: Inter - Coluna (Horizontal) ligações entre componentes de CABs que usam interruptores do três barramentos (Figura 2.19 Type 3).

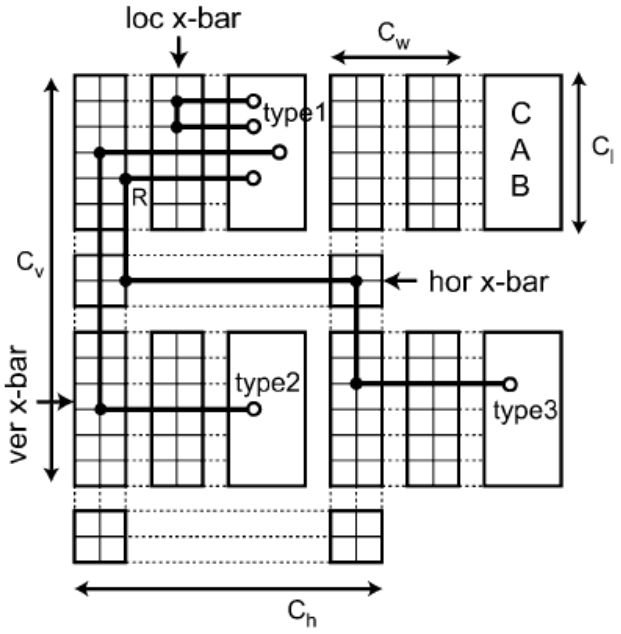


Figura 2.19: Estrutura de encaminhamento (Hall *et al.*, 2005)

2.4.3 CÉLULAS DE ENTRADA/SAÍDA

A FPAA conecta os sinais analógicos com as aplicações no seu exterior sejam elas cargas, circuitos geradores de sinal ou outros, através das células de entrada e saída. Cada uma destas células dependendo do tipo pode conter um conjunto de recursos configuráveis, filtro *anti-alising*, amplificador de ganho programável e um amplificador de ganho programável com estabilizador *chopper*, que lhe permite a conexão com o exterior sem recurso a componentes adicionais. Algumas destas células podem ainda ser bidirecionais, ou seja, podem ser configuradas como entradas ou saídas e para além disso ser multiplexadas, permitindo ligar mais do que um sinal na mesma célula. De forma a garantir o máximo de fidelidade do sinal de entrada, todo o tratamento do sinal no interior da FPAA é diferencial. Pode no entanto ser conectado um sinal simples na entrada positiva, neste caso um interruptor interno liga o terminal negativo da entrada diferencial à referência interna VMR (*Voltage Main Reference*).

Dependendo do fabricante, podemos encontrar três ou mais tipos de células de entrada e saída, a Anadigm na FPAA apresenta os listados a seguir.

CÉLULAS DE ENTRADA/SAÍDA CONFIGURÁVEIS

Este tipo de célula pode ser configurada como entrada ou saída de sinal, possui como recursos um filtro *antia-lising* de segunda ordem, um amplificador de ganho programável

e um amplificador de ganho programável com estabilizador *chopper*. Na Figura 2.20 está representado o diagrama desta célula.

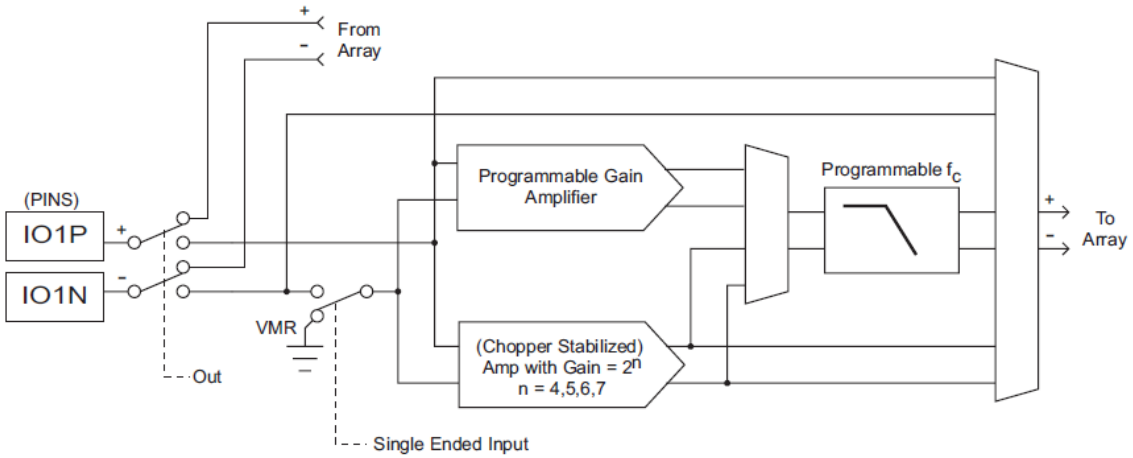


Figura 2.20: Célula de Entrada/Saída configurável (Anadigm, 2008)

O filtro *antia-lising* de segunda ordem pode ser configurado como um filtro passa-baixa, sendo para isso necessário configurar a sua frequência de corte. Este tipo de recurso é muito útil quando o sinal de entrada tem ruído associado de altas frequências. O amplificador de ganho programável é um recurso que se pode programar o ganho de entrada, na maioria dos casos configura-se este como *buffer*, ou seja, com ganho unitário. O amplificador de ganho programável com estabilizador *chopper*, difere do anterior pelo facto de possuir um *chopper* que reduz significativamente a tensão de *offset* do sinal de entrada e o ganho pode ser programado na forma de 2^n onde n pode tomar os valores entre 4 e 7. Este tipo de recurso pode ser muito útil em circuitos em que o sinal de entrada tenha uma amplitude muito baixa necessitando de um ganho muito elevado. A saída de ambos os amplificadores pode ser ligada diretamente ao exterior da célula ou fazendo passar pelo filtro *anti-aliasing*. Qualquer um destes recursos pode ser colocado em modo *bypass*, embora não seja aconselhável fazer o *bypass* total. Quando este tipo de células são usadas como saída a sua conexão é feita diretamente para o exterior, significando que não existe nenhum recurso ativo na célula.

CÉLULAS DE ENTRADA/SAÍDA MULTIPLEXADAS

Este tipo de células são em quase tudo igual às configuráveis, a diferença está na entrada, estas possuem normalmente quatro entradas diferenciais ligadas a um *multiplexer* de oito canais, podendo de igual modo ser configuradas como entrada ou saída de sinal, e possuem também como recursos um filtro *anti-aliasing* de segunda ordem, um amplificador de ganho programável e um amplificador de ganho programável com estabilizador *chopper*. Na Figura 2.21 está representado o diagrama desta célula.

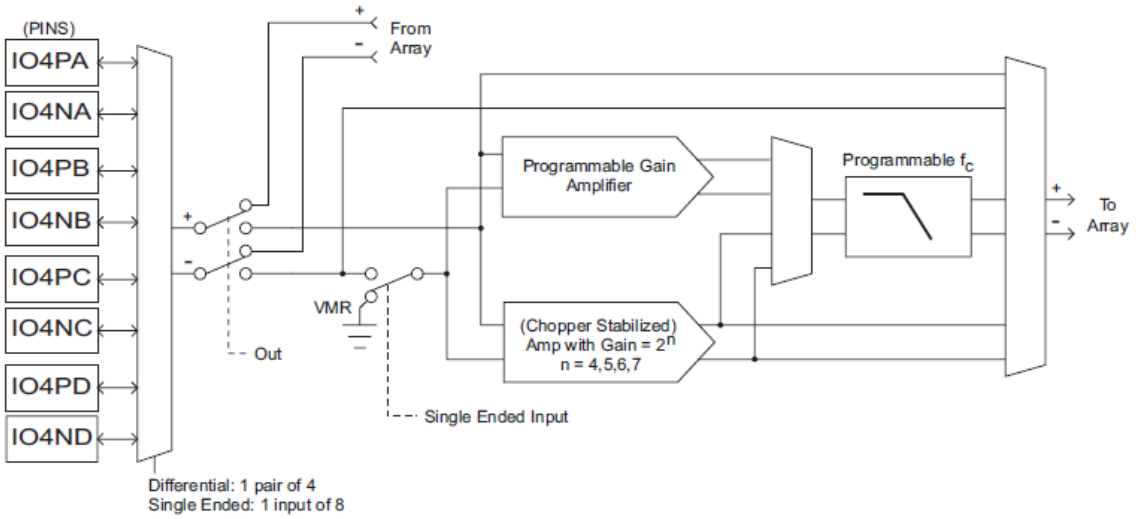


Figura 2.21: Célula de Entrada/Saída multiplexada (Anadigm, 2008)

CÉLULAS DE SAÍDA

As células de saída permitem disponibilizar na saída sinais digitais ou sinais analógicos diferenciais. Cada uma destas células possui, tal como as células de entrada anteriores, recursos que podem disponibilizar na saída como filtro *anti-aliasing* idêntico ao descrito anteriormente, mas que aqui serve como filtro de reconstrução de sinal de segunda ordem e um bloco conversor de diferencial para simples (DIFF2SINGLE). Nesta célula pode-se colocar em modo *bypass* todo o conjunto ou cada uma das entradas individualmente. Na Figura 2.22 está representado o diagrama desta célula.

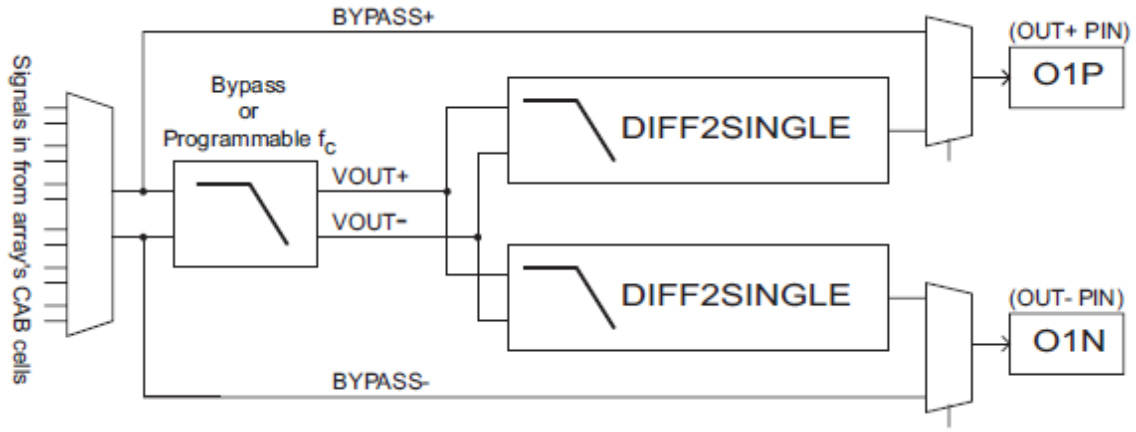


Figura 2.22: Célula de Saída (Anadigm, 2008)

2.5 TECNOLOGIA

Desde o seu aparecimento foram propostas algumas tecnologias diferentes de FPAA, mas que as podemos agrupar essencialmente em dois grupos, tempo contínuo e tempo discreto. Os FPAAs, do tipo contínuo, utilizam amplificadores operacionais de transcondutância OTA, com redes configuráveis de interligação e componentes passivos, proporcionando larguras de banda maiores assim como permitem a ausência de filtros *anti-aliasing* nas entradas e saídas de processos de sinal em tempo contínuo. Contudo, estas são mais difíceis de programar e podem originar maior distorção do sinal. Os FPAAs do tipo discreto recorrem a uma tecnologia de comutação de condensadores (*Switched Capacitor*s - S/Cs) para emular resistências. Contudo, esta tecnologia limita a sua largura de banda.

2.5.1 FPAA EM TEMPO CONTÍNUO

Os FPAAs em tempo contínuo são constituídos por uma matriz de componentes fixos, ligados por uma matriz de interruptores. Estes interruptores são controlados normalmente por registos binários, que podem ser configurados por um microcontrolador, permitindo à FPAA ser configurada em vários tipos de aplicações.

Nesta tecnologia, as resistências são realizadas utilizando transcondutores, em que o valor da transcondutância é programado utilizando uma tensão de referência gerada por um conversor de sinal.

VANTAGENS DAS FPAA EM TEMPO CONTÍNUO

- Não necessita de relógio *non-overlapping*;
- Não necessita de filtro Anti-Aliasing no recetor;
- Larga largura de banda;
- Boa linearidade.

DESVANTAGENS DAS FPAA EM TEMPO CONTÍNUO

- Necessita de memórias em cada célula.

Na literatura podemos encontrar três tipos diferentes de FPAA em tempo contínuo: os FPAA baseados em AMPOPs, os baseados em AMPOPs diferenciais, e os FPAAs baseados em OTA

com barramentos de interconexão.

FPAA BASEADOS EM AMPOPS

Este tipo de FPAA consiste num AMPOP ligado a uma estrutura de transcondutores. Estes transcondutores eram o elemento central do FPAA já que podiam ser configurados em resistências e condensadores e também eram usados no barramento de interconexão (Gulak, 1995).

FPAA BASEADOS EM AMPOPS DIFERENCIAIS

Neste tipo de FPAA os AMPOP possuem entradas diferenciais e condensadores na malha de realimentação, assim os CAB podem ser configurados como integradores ou diferenciadores. Através do controlo da tensão de *gate* dos transcondutores é possível configurar vários tipos diferentes de funções de transferência Gulak (1995).

FPAAS BASEADOS EM OTA COM BARRAMENTOS DE INTERCONEXÃO

Dos três tipos de FPAA em tempo contínuo os baseados em OTA são os que têm tido maior aceitação já que podem atingir os mesmos benefícios analógicos que os seus equivalentes digitais, FPGA. Possuem um amplificador de transcondutância OTA e um condensador programável, com isto é possível construir filtros numa gama de frequências muito alargada (Gulak, 1995).

2.5.2 FPAA EM TEMPO DISCRETO

Por sua vez os FPAAs em tempo discreto, são circuitos baseados na técnica da comutação de condensadores (*Switch Capacitor*), em que a tensão de entrada é amostrada através da abertura e fecho dos interruptores ligados aos condensadores. Os elementos constituintes destes dispositivos são AMPOPs e registos de deslocamento analógicos, que podem simular condensadores e resistências atuando na abertura e fecho dos interruptores. Contudo, esta tecnologia de matrizes de interruptores introduz algumas impedâncias parasitas nos sinais recebidos que limitam a largura de banda e introduzem ruídos e interferências no sistema (Lee e Gulak, 1991).

VANTAGENS DAS FPAA EM TEMPO DISCRETO

- Compatibilidade com a tecnologia CMOS;
- Imunidade à variação da temperatura;
- Linearidade.

DESVANTAGENS DAS FPAA EM TEMPO DISCRETO

- Necessita de relógio *non-overlapping*;
- A largura de banda do sinal é dez vezes inferior ao sinal de relógio.

2.5.3 TÉCNICA DO INTERRUPTOR CAPACITIVO

A técnica do interruptor capacitivo não é recente, James Maxwell usou-a em 1860 para a medição da resistência equivalente de um galvanómetro. A técnica é baseada na interrupção de condensadores, em que comutando periodicamente os interruptores entre dois nós do circuito é equivalente a uma resistência ligada a esses mesmos nós. O esquema representado na Figura 2.23 mostra um circuito paralelo do interruptor capacitivo. Quando se fecha o interruptor

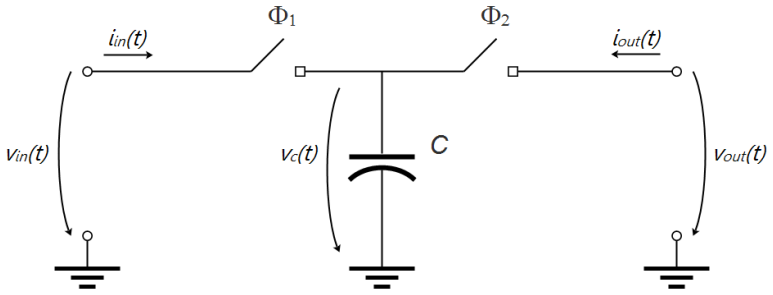


Figura 2.23: Diagrama de onda de duas fases sem sobreposição

Φ_1 circula pelo condensador uma corrente $i_{in}(t)$ que o carrega com uma tensão $v_c(t)$, abrindo Φ_1 e fechando Φ_2 o condensador vai descarregar a tensão armazenada.

A comutação dos interruptores deve ser feita considerando que não podem estar os dois ligados em simultâneo, ou seja que não existe sobreposição entre as ondas das fases Φ_1 e Φ_2 que os comutam, tal como representado na Figura 2.24. A matemática por trás desta técnica é simples, sabendo que a relação entre a carga e corrente num condensador é dada pela equação (2.1), e considerando que a variação de $V_{in}(t)$ e $V_{out}(t)$ para o período de tempo definido para a comutação dos interruptores é muito pequena e que pode ser desprezada, pode-se definir a

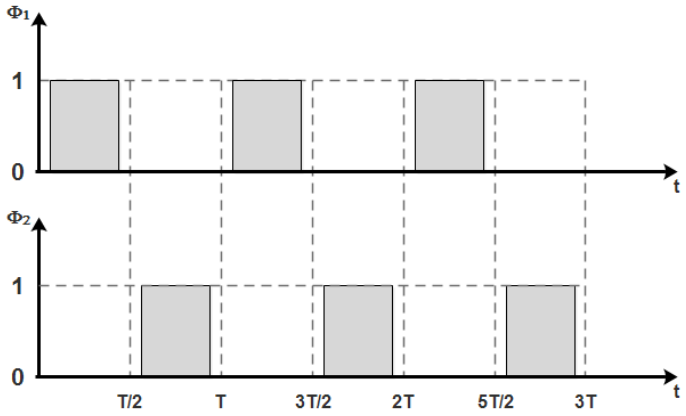


Figura 2.24: Circuito paralelo de interruptor capacitivo

corrente média no condensador pela expressão (2.2).

$$i(t) = \frac{dQ}{dt} = C\frac{dv_C}{dt} \tag{2.1}$$

$$i_{med} = \frac{1}{T}\int_0^{\frac{T}{2}} i(t)dt = \frac{1}{T}\int_0^{T} i(t)dt \tag{2.2}$$

Substituindo $i(t)$ na equação (2.2) pela equação (2.1) obtém-se

$$i_{med} = \frac{1}{T}\int_0^{\frac{T}{2}} \frac{dQ}{dt}dt = \frac{1}{T}\int_0^{\frac{T}{2}} C\frac{dv_C}{dt}dt \tag{2.3}$$

Resolvendo o integral vem

$$i_{med} = \left[\frac{v_C(t)}{T}\right]_0^{\frac{T}{2}} = \frac{Q(\frac{T}{2}) - Q(0)}{T} = \frac{C\left[v_C(\frac{T}{2}) - v_C(0)\right]}{T} \tag{2.4}$$

Contudo $v_C(T/2) = v_{in}(T/2)$ e $v_C(0) = v_{out}(0)$ e que o sinal máximo na entrada não excede dez vezes a frequência de comutação (f_C) então $v_{in}(t) \approx V_{in}$ e $v_{out}(t) \approx V_{out}$, então

$$i_{med} = \frac{C\left[v_{in}(\frac{T}{2}) - v_{out}(0)\right]}{T} \tag{2.5}$$

$$I = \frac{C\left[V_{in} - V_{out}\right]}{T} \Leftrightarrow \frac{\left[V_{in} - V_{out}\right]}{I} = \frac{T}{C} \tag{2.6}$$

$$R \approx \frac{T}{C} \tag{2.7}$$

Verifica-se assim que mantendo o condensador constante podemos obter uma resistência R de

qualquer valor fazendo variar somente a frequência de relógio do sistema. Esta técnica faz com que circuitos como placas de desenvolvimento se tornem dinâmicas e no caso das FPAA que se consiga usar resistências numa gama infindável de valores. Estes circuitos funcionam assim como controladores em tempo discreto, sem recorrer ao uso de conversores A/D ou D/A, com resultados mais fáceis de analisar recorrendo ao uso de transformadas z.

2.5.4 TRANSRESISTÊNCIA

Circuitos de transresistência são circuitos constituídos por redes de duas portas, onde aplicando tensão a uma das portas consegue-se controlar a corrente que flui na outra. Tipicamente uma das zonas está ao potencial zero (Terra virtual)(Allen, 2001).

CIRCUITO EQUIVALENTE DE TRANSRESISTÊNCIA POSITIVA

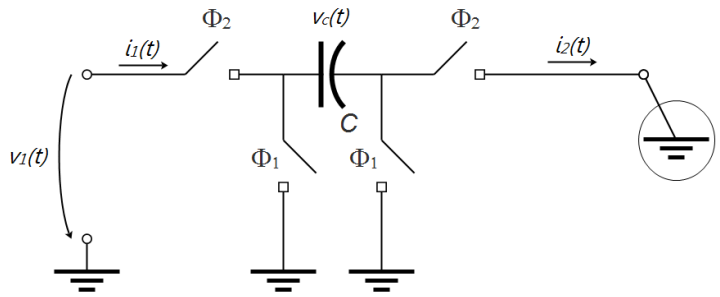


Figura 2.25: Circuito de transresistência positiva

A Figura 2.25 mostra um circuito de transresistência positiva que faz uso das duas fases Φ_1 e Φ_2 que estão definidas no esquema da Figura 2.26. Para a análise deste circuito considera-se

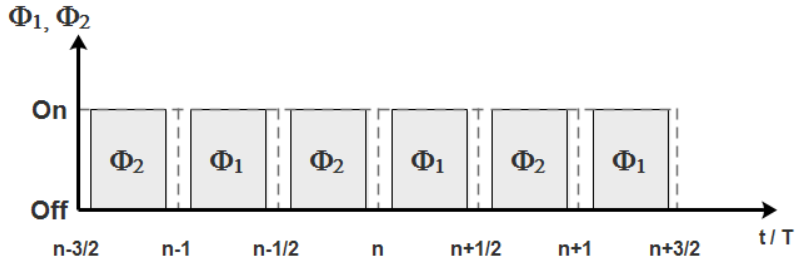


Figura 2.26: Diagrama de onda de duas fases

que a transresistência (R_T) é definida por

$$R_T = \frac{v_1(t)}{i_2(t)} = \frac{V_1}{I_{2med}} \tag{2.8}$$

considerando também $v_1(t)$ constante durante um período de relógio ($\Phi_1 + \Phi_2$), $i_2(t)$ é diferente de zero sempre que Φ_2 acontecer, desta forma pode-se dizer que

$$I_{2med} = \frac{1}{T}\int_{\frac{T}{2}}^{T} i_2(t)dt = \frac{q_2(T) - q_2(\frac{T}{2})}{T} = \frac{Cv_c(T) - Cv_c(\frac{T}{2})}{T} \tag{2.9}$$

nesta situação $v_c(T) = v_1$ e $V_c(T/2) = 0$, logo

$$I_{2med} = \frac{Cv_1}{T} \tag{2.10}$$

colocando na forma da equação (2.8)

$$R_T = \frac{T}{C} \tag{2.11}$$

CIRCUITO EQUIVALENTE DE TRANSRESISTÊNCIA NEGATIVA

Usando o mesmo circuito da Figura 2.25, mas trocando a ordem dos interruptores (Figura 2.27) obtém-se um circuito de transresistência negativa, que de igual modo faz uso das duas fases Φ_1 e Φ_2 que estão definidas no esquema da Figura 2.27. A análise deste circuito é em tudo idêntica

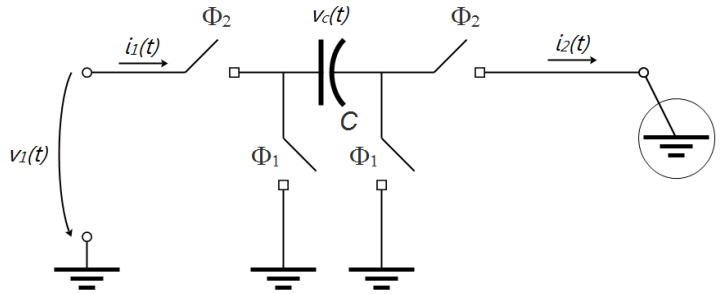


Figura 2.27: Circuito de transresistência negativa

ao circuito de transresistência positiva, mas nesta situação $v_c(T) = 0$ e $v_c(T/2) = v_1$, logo

$$I_{2med} = \frac{Cv_c(T) - Cv_c(\frac{T}{2})}{T} = \frac{-Cv_1}{T} \tag{2.12}$$

colocando na forma da equação (2.8)

$$R_T = -\frac{T}{C} \tag{2.13}$$

Desta forma verifica-se que com este tipo de técnica somente com um condensador consegue-se obter uma gama muito alargada de resistências, tendo para isso de fazer variar o período de amostragem, e trocando a ordem dos interruptores obtendo resistências negativas, o que no

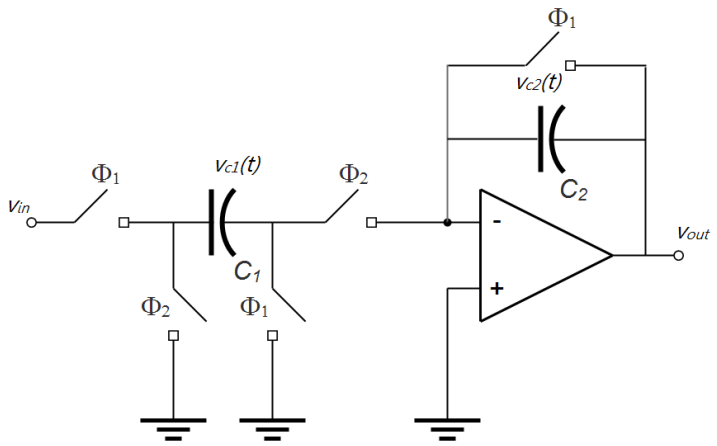


Figura 2.28: Amplificador não inversor

fundo corresponde a ter uma resistência seguida de um amplificador de ganho unitário negativo. Nas FPAAs esta técnica possibilita uma grande flexibilidade nos componentes passivos e uma grande diminuição no banco destes elementos.

AMPLIFICADOR NÃO INVERSOR

Com a técnica do interruptor capacitivo pode-se realizar todos os circuitos conhecidos usando AMPOPs. Como exemplo apresenta-se a seguir o circuito da Figura 2.28, que representa um amplificador não inversor.

Começa-se por analisar o intervalo de tempo $nT - T < t < nT - T/2$, que corresponde à fase Φ_1.

O condensador C_1 faz amostragem da tensão de entrada e fica com a carga

$$Q_{c1}(nT - T/2) = C_1 v_{in}(nT - T/2) \tag{2.14}$$

como a tensão de entrada v_{in} se mantém constante durante este período, então

$$v_{in}(nT - T/2) = v_{in}(nT - T) \Rightarrow Q_{c1}(nT - T/2) = C_1 v_{in}(nT - T) \tag{2.15}$$

já o condensador C_2 está curto-circuitado pelo interruptor o que faz com que a tensão armazenada seja igual à tensão do terminal negativo do AMPOP. Como este é uma terra virtual a tensão no condensador C_2 é igual a zero, sendo a sua carga igual a zero e consequentemente a tensão na saída também igual a zero.

$$Q_{c2}(nT - T/2) = C_2 v_{out}(nT - T/2) = 0 \tag{2.16}$$

No intervalo $nT - T/2 < t < nT$, que corresponde à fase Φ_2, o condensador C_1 vai ser ligado à terra pela placa que estava ligada a v_in,

$$Q_{c1}(nT) = 0 \tag{2.17}$$

como a resistência de entrada do AMPOP é muito elevada, a corrente vai circular de C_1 para v_{out} passando por C_2 fazendo com que este retenha a tensão amostrada em C_1 na fase anterior

$$Q_{c2}(nT) = C_2 v_{out}(nT) = C_1 v_{c1}(nT - T/2) = C_1 v_{in}(nT - T) \tag{2.18}$$

desta forma tem-se que

$$\begin{aligned} C_2 v_{out}(nT) &= C_1 v_{in}(nT - T) \\ v_{out}(nT) &= \left(\frac{C_1}{C_2}\right) v_{in}(nT - T) \end{aligned} \tag{2.19}$$

aplicando a transformada dos Z

$$v_{out}(z) = \left(\frac{C_1}{C_2}\right) z^{-1} v_{in}(z) \tag{2.20}$$

a função de transferência fica

$$H(z) = \frac{v_{out}(z)}{v_{in}(z)} = \left(\frac{C_1}{C_2}\right) z^{-1} \tag{2.21}$$

Na Tabela 2.1 apresenta-se a função de transferência em z de alguns circuitos implementados com esta técnica.

Tabela 2.1: Função de transferência

Circuito	Função de transferência
Amplificador não inversor	$H(z) = \left(\frac{C_1}{C_2}\right) z^{-1}$
Amplificador inversor	$H(z) = -\left(\frac{C_1}{C_2}\right)$
Integrador não inversor	$H(z) = \left(\frac{C_1}{C_2}\right) \frac{1}{z-1}$
Integrador inversor	$H(z) = -\left(\frac{C_1}{C_2}\right) \frac{z}{z-1}$

3. CÁLCULO FRACIONÁRIO

A utilização do cálculo fracionário como ferramenta para controlo de processos, não é uma teoria nova, e já foi explorada em muitos trabalhos teóricos, trabalhos estes que usam equações de diferenciação fracionária ou funções de transferência de ordem fracionária. O campo de aplicação do controlo fracionário é muito vasto, e a literatura estudada apresenta diversos resultados teóricos e experimentais que comprovam o sucesso desta técnica de controlo.

Neste capítulo vamos considerar as diferentes representações do cálculo fracionário integroderivativo, aproximações inteiras do operador s^{α}, controlo fracionário e métodos de implementação digital e analógica.

3.1 INTRODUÇÃO

As derivadas e integrais são usadas nas diversas áreas de engenharia e ciências. É usual encontrar operadores diferencias d^{-1}/dt^{-1}, d/dt, d^2/dt^2, d^3/dt^3 ou integrais $\int$, $\iint$, $\iiint$. Em qualquer um destes casos a ordem de diferenciação ou integração é sempre inteira.

Em 1695, L'Hospital colocou a questão sobre o significado de $d^n y/dx^n$ quando $n = 1/2$, ou

seja, "e se n for fracionário?". Leibniz respondeu "$d^{1/2}x$ será igual a $x\sqrt{(dx:x)}$". Esta questão levou ao aparecimento das derivadas e integrais de ordem não inteira. Durante quase três séculos foi tratado como um campo matemático puramente teórico. Foi durante o século XIX que esta teórica ganhou relevo com os trabalhos publicados por Liouville, Riemann, Weyl, Fourier, Abel, Lacroix, Leibniz, Grunwald e Letnikov. Apesar de ser tão antigo como o cálculo das integrais e derivadas de ordem inteira, o cálculo diferencial não ganhou muita popularidade na comunidade científica (Hilfer, 2000).

3.2 DEFINIÇÃO E REPRESENTAÇÃO

A representação de derivadas de ordem inteira é dada pela seguinte equação:

$$f^n(x) = \frac{d^n f(x)}{dx^n} = \lim_{h\to\infty} \frac{1}{h^n} \cdot \sum_{r=0}^{n} (-1)^r \cdot \binom{n}{r} \cdot f(x - r \cdot h), n \in \mathbb{Z} \tag{3.1}$$

onde:

$$\binom{n}{r} = \frac{n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot (n-r+1)}{r!}, n \in \mathbb{Z} \tag{3.2}$$

As derivadas de ordem fracionária não são mais que a generalização da expressão representada na equação (3.1) . Assim nestas derivadas, a ordem da derivada passa a ser representada por β, de tal forma que $\beta \in \mathbb{R}$ (Oldham e Spanier, 1974).

Para o cálculo das derivadas de ordem inteira é necessário determinar o domínio da função. Nas derivadas de ordem fracionária para além do domínio da função é necessário também definir o limite do domínio da derivada (Poldubny, 1999).

A representação da derivada de ordem fracionária é dada pela seguinte expressão:

$$f^\beta(x) = \frac{d^\beta f(x)}{dx^\beta} =_a \mathrm{D}_x^\beta f(x), \qquad \beta \in \mathbb{R} \tag{3.3}$$

Ao contrário das derivadas e integrais de ordem inteira, não se conhece a representação gráfica para as derivadas de ordem fracionária. Segundo Machado (2003) diversas interpretações plausíveis já foram propostas e reportadas, porém apresenta uma alternativa a partir de uma abordagem probabilística. Na Figura 3.1 apresenta-se a interpretação geométrica da derivada de ordem fracionária (Machado, 2003).

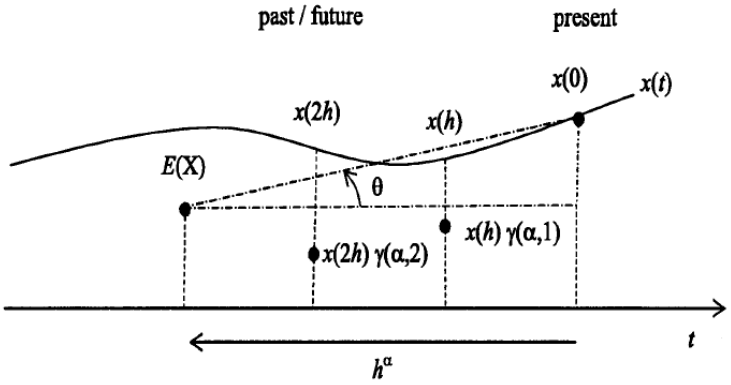


Figura 3.1: Intrepertação geométrica da derivada fracionária (Machado, 2003)

Existem várias definições para o cálculo de ordem fracionária representadas pela equação (3.3), seguidamente são apresentadas algumas das abordagens mais utilizadas.

G. W. LEIBNIZ (1695 - 1697)

Em cartas dirigidas a J. Wallis e J. Bernulli, em 1697, Leibniz apontou uma possível aproximação para as derivadas de ordem fracionária de tal forma que para valores não inteiros, de n a sua definição é a seguinte:

$$\frac{d^n e^{mx}}{dx^n} = m^n e^{mx} \tag{3.4}$$

L. EULER (1730)

L. Euler (1730) sugeriu usar esta relação para valores de n negativos ou não inteiros, ou seja, valores racionais:

$$\frac{d^n x^m}{dx^n} = m(m-1)...(m-n+1)x^{m-n} \tag{3.5}$$

usando a seguinte propriedade da função gama,

$$\Gamma(m+1) = m(m-1)...(m-n+1)x^{m-n} \tag{3.6}$$

obtém-se

$$\frac{d^n x^m}{dx^n} = \frac{\Gamma(m+1)}{\Gamma(m-n+1)}x^{m-n} \tag{3.7}$$

por sua vez a função gama é definida pela seguinte expressão:

$$\Gamma(z) = \int_o^\infty e^{-t} t^2 dt, \qquad Re(z) > 0 \tag{3.8}$$

J. B. J. FOURIER (1820 - 1822)

Através da seguinte representação integral

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} f(z)dz \int_{-\infty}^{\infty} \cos(px - pz)dp \tag{3.9}$$

escreveu

$$\frac{d^n f(x)}{dx^n} = \frac{1}{2\pi} \int_{-\infty}^{\infty} f(z)dz \int_{-\infty}^{\infty} \cos(px - pz + n\frac{\pi}{2})dp \tag{3.10}$$

N. H. ABEL (1823 - 1826)

Abel considerou a seguinte representação de integral para números inteiros

$$\int_0^x \frac{s'(\eta)}{(x-\eta)\alpha} d\eta = \psi(x) \tag{3.11}$$

para qualquer valor de α

$$s(x) = \frac{1}{\Gamma(1-\alpha)} \frac{d^{-\alpha}\psi(x)}{dx^{-\alpha}} \tag{3.12}$$

J. LIOUVILLE (1832 - 1855)

Liouville propôs três aproximações, a primeira considerando a representação exponencial de uma função

$$f(x) = \sum_{n=0}^{\infty} c_n e^{a_n x} \tag{3.13}$$

generalizou a aproximação de Leibniz representada na equação (3.4) na seguinte fórmula

$$\frac{d^\gamma f(x)}{dx^\gamma} = \sum_{n=0}^{\infty} c_n a_n^\gamma e^{a_n x} \tag{3.14}$$

A segunda aproximação considerou um integral fracionário

$$\int^{\mu} \Phi(x)dx^{\mu} = \frac{1}{(-1)^{\mu}\Gamma(\mu)} \int_0^{\infty} \Phi(x+\alpha)\alpha^{\mu-1}d\alpha \tag{3.15}$$

$$\int^{\mu} \Phi(x)dx^{\mu} = \frac{1}{\Gamma(\mu)} \int_0^{\infty} \Phi(x-\alpha)\alpha^{\mu-1}d\alpha \tag{3.16}$$

Substituído $\tau = x+\alpha$ na equação (3.15) e $\tau = x-\alpha$ na equação (3.16), obteve

$$\int^{\mu} \Phi(x)dx^{\mu} = \frac{1}{(-1)^{\mu}\Gamma(\mu)} \int_x^{\infty} (\tau-x)^{\mu-1}\Phi(\tau)d\tau \tag{3.17}$$

$$\int^{\mu} \Phi(x)dx^{\mu} = \frac{1}{\Gamma(\mu)} \int_x^{\infty} (x-\tau)^{\mu-1}\Phi(\tau)d\tau \tag{3.18}$$

A terceira aproximação de Liouville

$$\frac{d^{\mu}F(x)}{dx^{\mu}} = \frac{(-1)^{\mu}}{h^{\mu}} \left(F(x)\frac{\mu}{1}F(x+h) + \frac{\mu(\mu-1)}{1.2}F(x+2h) - ... \right) \tag{3.19}$$

$$\frac{d^{\mu}F(x)}{dx^{\mu}} = \frac{1}{h^{\mu}} \left(F(x)\frac{\mu}{1}F(x-h) + \frac{\mu(\mu-1)}{1.2}F(x-2h) - ... \right) \tag{3.20}$$

G. F. B. RIEMANN (1847 - 1876)

A representação de integral fracionário segundo Riemann é definida usando a série de Taylor

$$D^{-\nu}f(x) = \frac{1}{\Gamma(\nu)} \int_c^x (x-t)^{\nu-1}f(t)dt + \psi(t) \tag{3.21}$$

3.2.1 REPRESENTAÇÕES MAIS POPULARES

Apesar do impacto positivo das definições apresentadas em cima, foram três as representações que obtiveram maior aceitação e que são utilizadas com maior frequência nos trabalhos de investigação.

RIEMANN-LIOUVILLE

A definição de Riemann-Liouville é obtida partindo do integral de Cauchy que reduz o cálculo da primitiva. Desta forma a integração de multiplicidade n de uma função $f(t)$, é reduzida a uma simples integração simples do tipo convolução. Assim a definição de Riemann-Liouville para a derivada fracionária da função $f(t)$ de ordem $\alpha > 0$ é definida por:

$$ {}_a\mathrm{D}_t^{\alpha} f(x) = \frac{1}{\Gamma(n-\alpha)} \left(\frac{d}{dt} \right)^n \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha-n+1}} d\tau, \qquad (n-1) \leq \alpha < n \tag{3.22} $$

onde n representa um número inteiro e α um número Real.

M. CAPUTO (1967)

Caputo sugeriu uma notação diferente, mais restritiva que a da definição de Riemann-Liouville pois requer a integração absoluta da derivada de ordem n da função $f(t)$, mas com a vantagem de conduzir à formulação de condições iniciais dadas por derivadas inteiras da função $f(t)$. A derivada fracionária de Caputo de ordem $\alpha > 0$ é definida como:

$$ {}_a^C\mathrm{D}_t^{\alpha} f(x) = \frac{1}{\Gamma(n-\alpha)} \int_a^t \frac{f^{(n)}(\tau)}{(t-\tau)^{\alpha-n+1}} d\tau, \qquad (n-1) \leq \alpha < n \tag{3.23} $$

onde n representa um número inteiro e α um número Real.

GRÜNWALD-LETNIKOV

A definição de Grünwald-Letnikov é a que impõe menos restrições nas funções em que é aplicada e unifica num operador as noções de derivada e integral (Oldham e Spanier, 1974). A definição de Grünwald-Letnikov para a derivada e integral de ordem fracionária é dada por:

$$ {}_a\mathrm{D}_t^{\alpha} f(x) = \lim_{h \to 0} \frac{1}{h^{\alpha}} \sum_{j=0}^{\left[\frac{t-a}{h}\right]} (-1)^j \binom{\alpha}{j} f(t-jh), \qquad \left[\frac{t-a}{h}\right] \to Inteiro \tag{3.24} $$

3.3 APROXIMAÇÃO INTEIRA À ORDEM FRACIONÁRIA

A realização de hardware e software de sistemas fracionários só pode ser feita recorrendo a funções inteiras, desta forma a função de transferência de um sistema fracionário é

implementada aproximando-a a uma função inteira contínua ou discreta. A utilização de memória destes sistemas é ilimitada, sendo as aproximações casos particulares de memória limitada, o que os torna mais fáceis de implementar. Apesar da redução de memória as aproximações apresentam uma reprodução fidedigna do sistema dentro da largura de banda definida. Existem vários métodos encontrados na literatura para aproximar uma função de transferência fracionária a uma função de transferência de ordem inteira. De uma forma geral pode-se agrupar em aproximações contínuas e aproximações discretas. Torna-se assim claro que só através de aproximações inteiras é possível implementar funções de transferência fracionárias com componentes passivos (resistências, condensadores e bobines) ou funções (equação às diferenças e filtros digitais).

Nesta secção vamos apresentar alguns métodos destas duas formas de implementação.

3.3.1 APROXIMAÇÕES CONTÍNUAS

Dos vários métodos analisados para a obtenção de um modelo de aproximação contínua à ordem fracionária, dois deles são particularmente interessantes sobre o ponto de vista do controlo fracionário: os métodos baseados na expansão contínua de funções CFE (*Continued Fraction Expansion*) e os baseados no ajuste da curva de resposta. As aproximações de Carlson e Matsuda são exemplos de métodos baseados na CFE, enquanto as aproximações de Oustaloup e Chareff são exemplos de métodos baseados no ajuste da curva de resposta (Vinagre *et al.*, 2000).

MÉTODOS BASEADOS NA CFE

A expansão contínua de funções é um método para avaliar funções que convergem mais rapidamente que a expansão de potências PSE (*Power Series Expansion*), e no plano complexo num domínio mais alargado (Vinagre *et al.*, 2000). Desta forma o resultado da aproximação de uma função irracional, $G(s)$ pode ser expresso por:

$$\begin{aligned} G(s) &\simeq a_0 + \cfrac{b_1(s)}{a_1(s) + \cfrac{b_2(s)}{a_2(s) + \cfrac{b_3(s)}{a_3(s) + \ldots}}} \\ &= a_0(s) + \cfrac{b_1(s)}{a_1(s) +} \cfrac{b_2(s)}{a_2(s) +} \cfrac{b_3(s)}{a_3(s) +} \ldots \end{aligned} \tag{3.25}$$

A aplicação deste método gera uma função racional $\hat{G}(s)$ que é a aproximação da função irracional $G(s)$.

Método da CFE

Para a obtenção da aproximação da função irracional $G(s) = s^{-\alpha}, 0 < \alpha < 1$, pode-se aplicar diretamente o método CFE às funções:

$$\begin{aligned} G_h(s) &= \frac{1}{(1+sT)^{\alpha}} \\ G_l(s) &= \left(1+\frac{1}{s}\right)^{\alpha} \end{aligned} \tag{3.26}$$

Onde $G_h(s)$ representa a aproximação para as frequências altas ($w >> 1$) e $G_l(s)$ a aproximação para as frequências baixas ($w << 1$).

Método de Carlson

O método proposto por Carlson deriva da forma regular do processo de Newton usado para aproximações iterativas da raiz de ordem α. As seguintes relações são o ponto de partida para este método:

$$(H(s))^{1/\alpha} - (G(s)) = 0 \quad ; \quad H(s) = (G(s))^{\alpha} \tag{3.27}$$

definindo $\alpha = 1/q$ e $m = q/2$ em cada iteração, tendo para valor inicial $H_0(s) = 1$ obtém-se uma função racional aproximada da seguinte forma

$$H_i(s) = H_{i-1}(s)\frac{(q+m)(H_{i-1}(s))^2 + (q+m)G(s)}{(q-m)(H_{i-1}(s))^2 + (q-m)G(s)} \tag{3.28}$$

Método de Matsuda

O método apresentado por Matsuda de aproximar uma função irracional por uma racional, é obtido por CFE e ajustando a função original a um conjunto de pontos pertencentes a uma curva logarítmica (Vinagre *et al.*, 2000). Assumindo que os pontos ajustados são $s_k, k = 0, 1, 2, ...$, a aproximação toma a forma:

$$H(s) = a_0 + \frac{s-s_0}{a_1+}\frac{s-s_1}{a_2+}\frac{s-s_2}{a_3+}..., \tag{3.29}$$

onde

$$a_i = v_i(s_i), \qquad v_0(s) = H(s), \qquad v_{i+1}(s) = \frac{s-s_i}{v_i(s)-a_i} \tag{3.30}$$

MÉTODOS BASEADOS NO AJUSTE DA CURVA DE RESPOSTA

Estes métodos baseiam a sua forma de implementação no ajuste da largura de banda e assim reduzir a memória total usada pela aproximação. Através destas aproximações consegue-se obter funções racionais a partir das funções irracionais originais.

Método de Oustaloup

O filtro recursivo de Oustaloup dá uma aproximação muito boa de um elemento de ordem fracionária dentro de uma largura de banda limitada. Se considerarmos w_b como o limite inferior da largura de banda e w_h como o limite superior, pode-se definir o filtro como

$$G_f(s) = K\prod_{-N}^{N}\frac{s+w_k^{'}}{s+w_k} \tag{3.31}$$

Onde γ representa a ordem do diferenciador, $2N+1$ a ordem do filtro, e (w_b, w_h) a largura de banda.

$$w_k^{'} = w_b\left(\frac{w_h}{w_b}\right)^{\frac{k+N+1/2(1-\gamma)}{2N+1}} \quad ; \quad K = w_h^{\gamma} \quad ; \quad w_k = w_b\left(\frac{w_h}{w_b}\right)^{\frac{k+N+1/2(1+\gamma)}{2N+1}} \tag{3.32}$$

Para a implementação desta aproximação usa-se a função ousta_fod, que implementa a aproximação de Oustaloup referida na equação (3.31), em Matlab. Esta função está definida da seguinte forma:

Filtro de Oustaloup em Matlab

```
function G=ousta_fod(Gamma,N,wb,wh)
k=1:N;
wu=sqrt(wh/wb);
wkp=wb*wu.^((2*k-1-Gamma)/N);
wk=wb*wu.^((2*k-1+Gamma)/N);
G=zpk(-wkp,-wk,wh^Gamma);
G=tf(G);
```

Método de Chareff

Este método é muito semelhante ao proposto por Oustaloup, a sua aproximação é definida como:

$$H(s) = \frac{1}{\left(1+\frac{s}{p_T}\right)^{\alpha}} \tag{3.33}$$

Representando a mesma equação por um quociente de polinómios com s fatorizado, vem:

$$\hat{H}(s) = \frac{\prod_{i=0}^{n=1}\left(1+\frac{s}{z_i}\right)}{\prod_{i=0}^{n=1}\left(1+\frac{s}{p_i}\right)} \tag{3.34}$$

Onde os coeficientes são calculados de forma a obter o desvio máximo da magnitude original da resposta em frequência em y dB:

$$p_0 = p_T\sqrt{b}, \qquad p_i = p_0(ab)^i, \qquad z_i = ap_0(ab)^i \tag{3.35}$$

O número de polos e zeros está relacionado com a largura de banda desejada, com um erro dado pela seguinte expressão:

$$N = \left\lceil \frac{log\left(\frac{w_{max}}{p_0}\right)}{log(ab)} \right\rceil + 1 \tag{3.36}$$

3.3.2 APROXIMAÇÕES DISCRETAS

A discretização do operador fracionário $s^{\pm\alpha}, (\alpha \in R)$, pode ser expresso pela função $s = w(z^{-1})$. Através desta substituição é possível determinar a forma da expansão bem como os seus coeficientes. As aproximação discretas são mais usadas que as aproximações contínuas, já que facilmente podem ser implementadas em hardware e software digital. Existem dois métodos para conseguir as aproximações discretas, o método direto e o indireto. O método indireto de discretização é conseguido em dois passos, o primeiro passo consiste em encontrar a aproximação contínua da função de transferência seguidamente procede-se à discretização dessa função. Quanto ao método direto é conseguido aplicado a PSE, CFE ou a expansão de MacLaurin com a função geradora apropriada. A relação ou conversão do domínio contínuo para o discreto ($s \to z$) é conhecido como a função geradora (Barbosa, Machado e Silva, 2006).

$$H^{\alpha}(z) = \left(\frac{1}{\lambda T}\frac{1-z^{-1}}{\gamma+(1-\gamma)z^{-1}}\right)^{\alpha} \tag{3.37}$$

Na Tabela 3.1 estão apresentadas as funções geradoras mais populares de $s \approx H\left(z^{-1}\right)$, resultantes da inversão da função geradora proposta por Smith representada na equação (3.37), considerando diferentes valores para o parâmetro γ e um valor fixo para $\lambda = 1$.

Tabela 3.1: Funções geradoras $s \approx H\left(z^{-1}\right)$ com $\lambda = 1$

γ	$s \to z$	Método
0	$s \approx \left(\frac{1}{T}\frac{1-z^{-1}}{z^{-1}}\right)^{\alpha}$	Regra de Euler à frente
$\frac{1}{2}$	$s^{\alpha} \approx \left(\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}\right)^{\alpha}$	Tustin
$\frac{7}{8}$	$s^{\alpha} \approx \left(\frac{8}{7T}\cdot\frac{1-z^{-1}}{1+z^{-1}/7}\right)^{\alpha}$	Al-Alaoui
1	$s^{\alpha} \approx \left(\frac{1-z^{-1}}{T}\right)$	Regra de Euler a atrás
$\frac{3}{2}$	$s^{\alpha} \approx \left(\frac{2}{3T}\cdot\frac{1-z^{-1}}{1-z^{-1}/3}\right)^{\alpha}$	Adams de Segunda Ordem

3.4 CONTROLO FRACIONÁRIO

O controlo fracionário tal como o controlo clássico utiliza equações diferenciais, mas só que de ordem não inteira. Os processos onde este tipo de controlo pode ser aplicado são inúmeros e ao longo do tempo foram muitas as aplicações onde a sua utilização teve resultados teóricos e experimentais muito positivos.

Manabe (1961) reporta pela primeira vez a aplicação de uma equação diferencial de ordem fracionário para síntese de controladores com realimentação.

Oustaloup (1975) seguiu-se e a sua aplicação consistiu na utilização do controlo fracionário para o controlo de sistemas de laser colorido. Em 1991 reporta a aplicação do controlo fracionário no controlo de sistemas dinâmicos e demonstra o desempenho superior do CRONE quando comparado com o PID.

Por sua vez Poldubny (1999) apresenta a adaptação do controlador PID fracionário o qual designou por $PI^{\lambda}D^{\mu}$, com um integrador de ordem λ e um diferenciador de ordem μ.

Vinagre *et al* apresentam em 2000 uma aproximação no domínio das frequências usando controladores PID.

Pommier *et al*, em 2002 aplicam o controlo fracionário a atuadores hidráulicos e componentes da suspensão automóvel.

Barbosa, Machado e Jesus (2008), apresentam um estudo teórico descrevendo uma abordagem

baseada em diagramas de Bode.

3.4.1 CONTROLADOR PID FRACIONÁRIO

O conceito por trás do controlador fracionário é o cálculo fracionário. A Figura 3.2 mostra um controlador PID fracionário genérico.

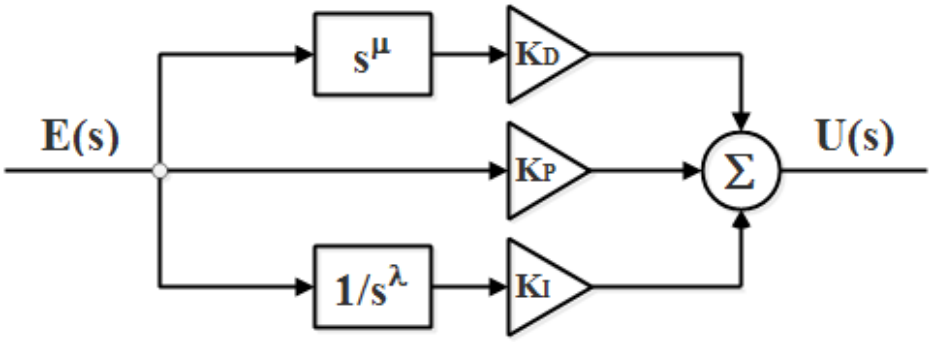


Figura 3.2: Controlador fracionário genérico

A representação proposta por Poldubny (1999) é a mais comum, que considera um integrador de ordem λ e um diferenciador de ordem μ. A função de transferência de um controlador $PI^{\lambda}D^{\mu}$ pode ser dado pela seguinte expressão,

$$G_c(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s^{\lambda}} + K_d s^{\mu} \qquad (\lambda, \mu > 0) \tag{3.38}$$

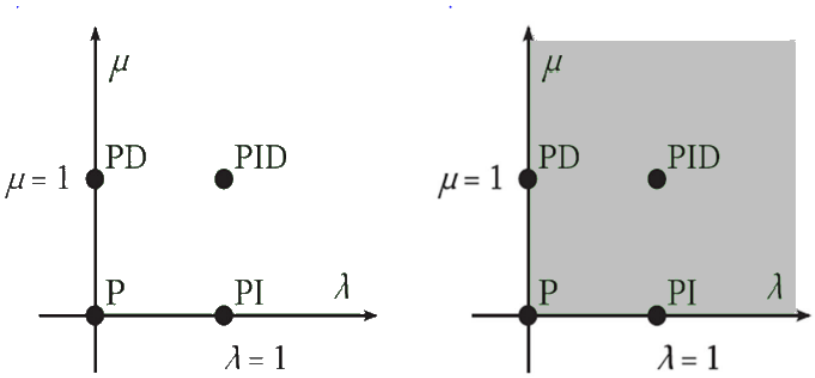
ou de outra forma,

$$G_c(s) = \frac{U(s)}{E(s)} = K_p \left(1 + \frac{1}{T_i s^{\lambda}} + T_d s^{\mu}\right) \qquad (\lambda, \mu > 0) \tag{3.39}$$

Onde $G_c(s)$ representa a função de transferência do controlador, $E(s)$ o erro, e $U(s)$ a saída. Os parâmetros K_p, K_i, K_d representam os ganhos: proporcional, integrativo e derivativo, respetivamente. Por sua vez o sinal $u(t)$ pode ser expresso no domínio dos tempos como

$$u(t) = K_p e(t) + K_i D^{-\lambda} e(t) + K_d D^{\mu} e(t) \tag{3.40}$$

O FOPID (PID de ordem fracionária) permite a realização de um número infinito de controladores, atuando nos ganhos K_p e K_i e também na ordem do integral (λ) e derivada (μ). Se estas ordens forem inteiras, obtêm-se os controladores convencionais P, PI, PD, PID, mas estas podem ser reais o que torna muito flexível o projeto deste tipo de controlo. Desta forma faz-se a expansão do ponto para o plano, tal como representado na Figura 3.3. Na Tabela 3.2 apresenta-se alguns dos principais controladores que podem ser obtidos através da configuração



(a) Convencional (b) Fracionário

Figura 3.3: Comparação entre o controlador convencional e fracionário

dos parâmetros do controlador fracionário.

Tabela 3.2: Principais controladores fracionários

Controlador	Descrição	Parâmetros
P	P Convencional	$K_d = 0,\ K_i = 0$
PI	PI Convencional	$K_d = 0,\ K_i \neq 0,\ \lambda = 1$
PD	PD Convencional	$K_d \neq 0,\ K_i = 0,\ \mu = 1$
PID	PID Convencional	$K_d \neq 0,\ K_i \neq 0,\ \lambda = 1,\ \mu = 1$
PI^{λ}	PI Fracionário	$K_d = 0,\ K_i \neq 0,\ 0 < \lambda < 1$
PD^{μ}	PD Fracionário	$K_d \neq 0,\ K_i = 0,\ 0 < \mu < 1$
PID^{μ}	PID Integrativo inteiro	$K_d \neq 0,\ K_i \neq 0,\ \lambda = 1,\ 0 < \mu < 1$
$PI^{\lambda}D$	PID Derivativo inteiro	$K_d \neq 0,\ K_i \neq 0,\ 0 < \lambda < 1,\ \mu = 1$
$PI^{\lambda}D^{\mu}$	PID Fracionário	$K_d \neq 0,\ K_i \neq 0,\ 0 < \lambda < 1,\ 0 < \mu < 1$

3.5 IMPLEMENTAÇÃO

De uma forma geral podemos implementar controladores fracionários de duas formas: implementação digital com um *software* ou *firmware* implementando funções de transferência ou filtros IIR (*Infinite Impulse Response*) ou FIR (*Finite Impulse Response*) a correr num computador, microcontrolador, PLC (*programmable logic controller*) ou num FPGA (Vinagre *et al.*, 2000). Existe também a implementação analógica usando fractores, fractâncias em montagens com integradores e diferenciadores e filtros bilineares ou biquadráticos.

3.5.1 CONTROLADOR FRACIONÁRIO DIGITAL

Para a realização digital é necessário a equação às diferenças da função de transferência do filtro digital.

IMPLEMENTAÇÃO USANDO PSE

Uma das formas mais fáceis de implementação é usando o método direto de discretização, utilizando um comprimento de memória finita, e aplicando a PSE. Desta forma usando a equação (3.24) e efetuando a PSE da função geradora de Euler da regra atrás representada na Tabela 3.1 chega-se à representação da derivada fracionária de ordem α:

$$Y(z) = T^{\pm\alpha} PSE \left\{ \left(\frac{1-z^{-1}}{T} \right)^{\pm\alpha} \right\} F(z) \tag{3.41}$$

Utilizando o princípio de memória curta, descrito por Poldubny (1999), a equação equivalente discreta do operador integro-diferencial ($w(z^{-1})$) tem a seguinte forma:

$$D^{\pm\alpha}(z) = (w(z^{-1}))^{\pm\alpha} = T^{\pm\alpha} z^{-[L/T]} \sum_{j=0}^{[L/T]} (-1)^j \begin{pmatrix} \pm\alpha \\ j \end{pmatrix} z^{[L/T]-j} \tag{3.42}$$

onde T representa o período de amostragem, L o comprimento da memória e $(-1)^j \binom{\pm\alpha}{j}$ os coeficientes da binomiais $c_j^{\pm\alpha}$, $(j = 0, 1, \cdots)$ representado por:

$$c_0^{(\pm\alpha)} = 0, \qquad c_j^{(\pm\alpha)} = \left(1 - \frac{1+(\pm\alpha)}{j} \right) c_{j-1}^{(\pm\alpha)} \tag{3.43}$$

Desta forma a função de transferência do $PI^{\lambda}D^{\mu}$ na forma discreta é dada pela seguinte expressão:

$$G_c(z^{-1}) = \frac{U(z^{-1})}{E(z^{-1})} = K_p + K_i (w(z^{-1}))^{-\lambda} + K_d (w(z^{-1}))^{\mu} \tag{3.44}$$

Substituindo $(w(z^{-1}))$ representado na equação (3.44) pela equação (3.42) obtém-se o controlador digital de ordem fracionária representado pela seguinte equação às diferenças:

$$u(k) = K_p e(k) + \frac{K_i}{T^{-\lambda}} \sum_{j=v}^{k} q_j^{-\lambda} e(k-j) \frac{K_d}{T^{\mu}} \sum_{j=v}^{k} q_j^{\mu} e(k-j) \tag{3.45}$$

A equação representada em (3.45) faz uso de toda a história, e para tal era necessário uma memória infinita. Para que esta seja passível de ser implementada recorre-se ao princípio da memória curta onde $v = 0$ para $k < (L/T)$ ou $v = k - (L/T)$ para $K > (L/T)$. A implementação da equação (3.45) num dispositivo sob a forma de *firmware* pode ser feita de acordo com o seguinte pseudo-código listado no Anexo A. Para implementação com microcontrolador pode ser usado o esquema da Figura 3.4.

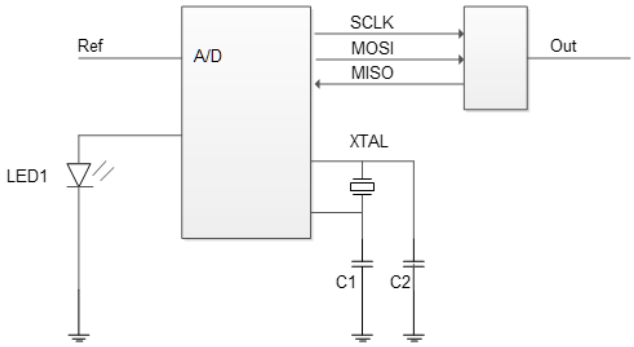


Figura 3.4: Implementação com Microcontrolador

IMPLEMENTAÇÃO USANDO CFE

De uma forma geral a implementação de um controlador fracionário digital usando a CFE é baseado num filtro IIR, que pode ser definido segundo a seguinte expressão:

$$F(z^{-1}) = \frac{U(z^{-1})}{E(z^{-1})} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_M z^{-M}}{a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_N z^{-N}} \tag{3.46}$$

Desta forma a função de transferência do $PI^{\lambda}D^{\mu}$ na forma discreta é dada pela seguinte expressão:

$$H_{PI^{\lambda}D^{\mu}}(z) = K_p + K_i H_I^{\lambda}(z) + K_d H_D^{\mu}(z), \qquad 0 \leq \lambda, \mu \leq 1 \tag{3.47}$$

Onde K_p, K_i e K_d representam os ganhos proporcional, integrativo e derivativo do controlador, $H_I^{\lambda}(z)$ representa o aproximação discreta do integrador fracionário de ordem λ, $H_D^{\mu}(z)$ representa aproximação discreta do derivador fracionário de ordem μ.

O controlador fracionário na forma de um filtro IIR (equação (3.46)) pode ser implementado via *firmware* diretamente em qualquer microcontrolador, ou via *software* para correr num computador. A implementação usando a forma canónica representada na Figura 3.5, é feita considerando uma entrada $e(k)$ e uma saída $u(k)$ e uma excursão do sinal de entrada entre 0 e U_{FOC}. O pseudo-código desta implementação está listado no Anexo B.

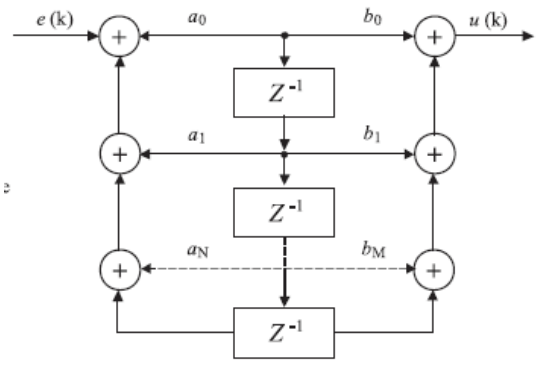


Figura 3.5: Representação canónica de um filtro IIR

3.5.2 CONTROLADOR $PI^{\lambda}D^{\mu}$ ANALÓGICO

A realização de controladores PID fracionários analógicos, é feita da mesma forma que os de ordem inteira, mas substituindo os condensadores do integrador e derivador por impedâncias fracionárias, sendo elas fractores ou fractâncias, da ordem pretendida, que serão descritos mais em pormenor nas secções seguintes.

Na Figura 3.6 está representado o esquema de um controlador $PI^{\lambda}D^{\mu}$ analógico implementado com AMPOPs, resistências e fractores ou fractâncias. A impedância $Z(s)_d$ representa o derivador fracionário de ordem λ, e $Z(s)_i$ o integrador fracionário de ordem μ. O ganho K_p pode ser calculado por $K_p = \frac{R_3}{R_4}$, a constante de integração por $T_i = \frac{Z(s)_i}{R_i}$ e a constante de derivação por $T_d = \frac{R_d}{Z(s)_d}$.

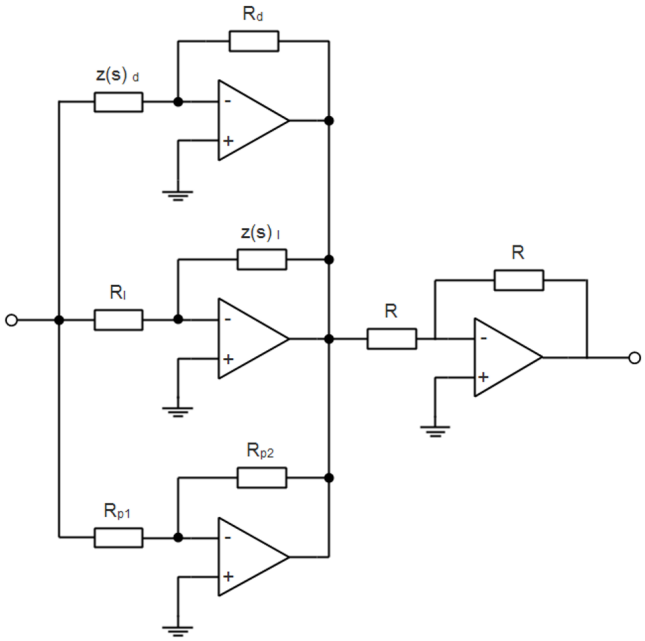


Figura 3.6: Controlador $PI^{\lambda}D^{\mu}$ analógico (Yang, Petráš e Dingyu, 2009)

3.6 FRACTOR

Um fractor é um elemento eletrónico passivo de dois pinos semelhante a um condensador. Todos os condensadores apresentam um comportamento fracionário (Westerlund e Ekstam, 1994) e devem ser modelados segundo a equação representada em (3.48). Apesar deste comportamento o valor de α é muito próximo de um, o que faz com se despreze o comportamento fracionário.

$$Z_C = \frac{1}{s^{\alpha} C} \tag{3.48}$$

Apesar de ainda não existir um elemento que implemente analogicamente o comportamento fracionário, já existem alguns estudos sobre a implementação deste tipo de dispositivo. Jesus e Machado (2009) apresentam um estudo descrevendo um processo eletrolítico na perspetiva dos condensadores fracionários. Na Figura 3.7 está representado o esquema proposto.

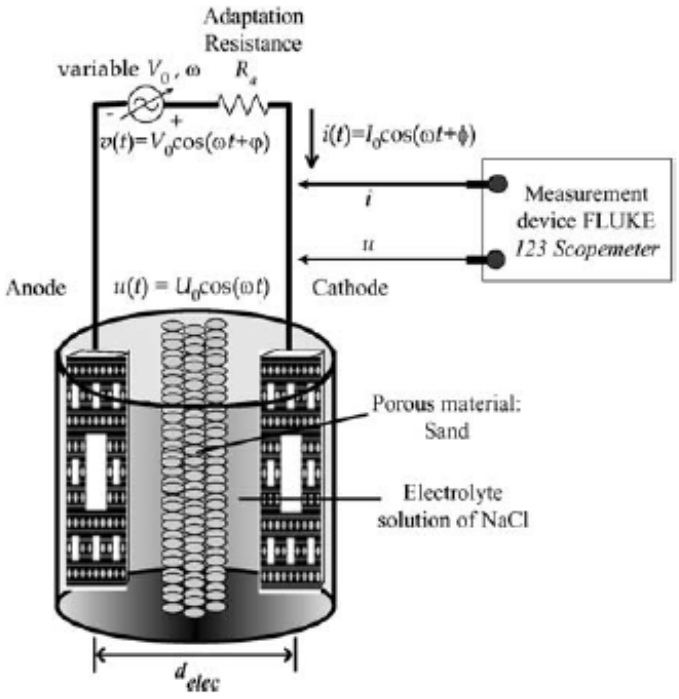
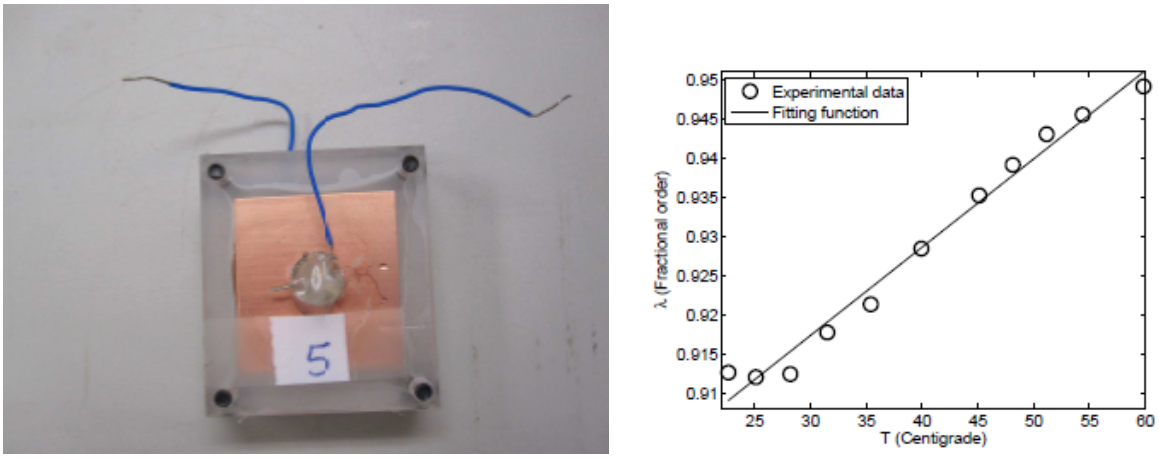


Figura 3.7: Processo eletrolítico (Jesus e Machado, 2009)

Bohannan (2008) propôs um elemento fracionário feito à mão com tamanho semelhante a um condensador com comprimento e largura de 3,5 cm e 1,0 cm de espessura, representado na Figura 3.8(a). O comportamento do fractor está modelado segundo a equação (3.49) em que K representa a impedância do fractor à frequência $w = 1/\tau$:

$$Z_C = \frac{K}{(jw\tau)^{\lambda}}, \qquad \lambda \in (0,1) \tag{3.49}$$

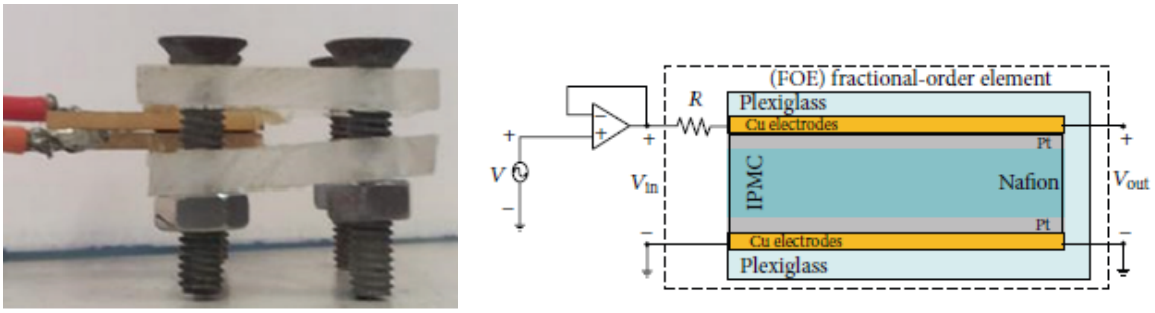
Tal como se pode observar na Figura 3.8(b) a ordem do operador fracionário λ varia linearmente com a temperatura, e com um controlo adequado da temperatura consegue-se assim fixar o valor de λ e logo controlar também a dinâmica do sistema.



(a) Fractor (b) Variação de λ com a temperatura

Figura 3.8: Protótipo do Fractor de Bohannan (Bohannan, 2008)

Bonomo *et al.* (2007) por sua vez propõem um fractor baseado em IPMCs (*Ionic polymer-metal compositess*) representado na Figura 3.9(a). Este circuito apresenta uma linearidade limitada dependendo das gamas de frequências usadas. Nas duas experiências efetuadas o autor conclui que para uma ordem de 0,05 o fractor é linear entre 1 e 100 Hz, já para para uma ordem de 0,3 o fractor apresenta linearidade entre 1 e 10 Hz. Em ambas as experiências o desvio de fase apresentado corresponde a um comportamento fracionário. A Figura 3.9(b) mostra o esquema de implementação do IPMC.



(a) Fractor (b) Esquema de implementação

Figura 3.9: Protótipo do Fractor de Bonomo (Bonomo *et al.*, 2007)

3.7 FRACTÂNCIA

Designa-se por fractância ao elemento passivo que implementa um comportamento de ordem fracionária.

De uma forma geral encontra-se na literatura três tipos de implementações diferentes de fractâncias, a mais utilizada é a estrutura em escada, também muito usadas são as estruturas em árvore e por fim as linhas de transmissão. Partindo de uma função no domínio de Laplace é fácil fazer a sua conversão em impedâncias ou admitâncias de forma a conseguir fazer a sua implementação recorrendo a elementos passivos R, L ou C. A realização destes circuitos pode ser feita tal como descrito por Podlubny *et al.* (2002), ou usando a CFE.

A implementação usando a CFE é direta e não necessita de cálculos adicionais. Os valores dos elementos elétricos da fractância são obtidos dos coeficientes da equação contínua quando estes são positivos, se alguns coeficientes forem negativos é necessário recorrer a conversores de resistência negativa (Podlubny *et al.*, 2002).

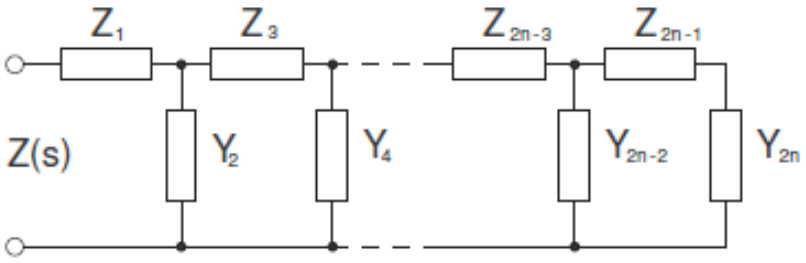


Figura 3.10: circuito da aproximação em escada finita

A estrutura em escada consegue aproximar o operador de ordem fracionária a uma estrutura analógica finita passível de ser implementada. Na Figura 3.10, está representada a forma genérica destas estruturas, onde $Z_{2k-1}(s)$ e $Y_{2k}(s)$ com $K = 1, \cdots, n$ correspondem às impedâncias do respetivo circuito. A impedância total do circuito $Z(s)$ pode ser calculada de acordo com a equação:

$$Z(s) = Z_1(s) + \cfrac{1}{Y_2(s) + \cfrac{1}{\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots + \cfrac{1}{Y_{2n-2}(s) + \cfrac{1}{Z_{2n-1}(s) + \cfrac{1}{Y_{2n}(s)}}}}} \tag{3.50}$$

4. ARQUITETURA

Atualmente o PC (Computador Pessoal) é a plataforma mais utilizada para efetuar sistemas de aquisição, processamento e tratamento de dados. Entre as principais razões para a sua popularidade pode destacar-se o baixo custo, a flexibilidade, desempenho e a facilidade de utilização. Em comparação com os sistemas tradicionais de medição, os sistemas DAQ baseados em PC exploram a capacidade de processamento, produtividade, sistemas de visualização e recursos de conectividades dos computadores padrão da indústria. Nos sistemas mais evoluídos a aquisição de dados não se fica apenas pela medição e recolha de dados, compreende também o controlo das aplicações em causa.

Neste capítulo é feita a descrição da arquitetura do sistema, bem como todos componentes nele utilizados como a placa de aquisição de sinais Hilink da Zeltom e a placa de desenvolvimento da Anadigm AN231K04-DVLP3, sendo também feita a apresentação da FPAA utilizada nesta placa, a AN231E04.

4.1 ARQUITETURA DO SISTEMA

O sistema apresentado nesta secção tem por base o computador portátil e o *software* Matlab. Este PC comunica com a placa de aquisição de sinais Hilink da Zeltom através da porta RS232, tendo sido disponibilizado pela Zeltom uma biblioteca com um conjunto de blocos configuráveis que permite a interligação entre estes dois componentes do sistema. Para a implementação na FPAA foi usado o kit de desenvolvimento AN2321E04-DVLP3 da Anadigm, juntamente com o software de configuração, o Anadigm Designer. Para a realização de testes necessários à verificação do funcionamento da placa Hilink, foi usado também um gerador de sinais simples baseado na DAC AD9850 da Analog. Para aferição do sistema foi considerado um motor de corrente contínua. A Figura 4.1 mostra a arquitetura geral do sistema.

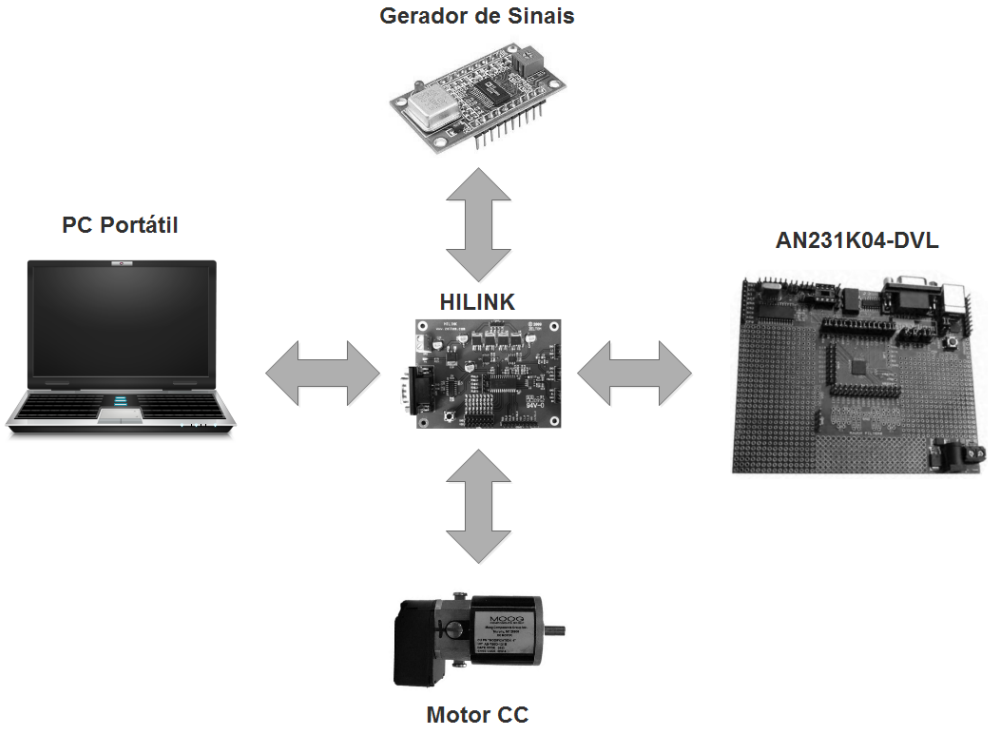


Figura 4.1: Arquitetura do sistema

4.2 PLACA DE DESENVOLVIMENTO AN231K04-DVLP3

A placa de desenvolvimento da Anadigm representada na Figura 4.2 fornece um mecanismo simples para a prototipagem, avaliação, depuração e teste dos de circuitos analógicos. Esta placa de desenvolvimento está equipada com a FPAA, programador, EEPROM, porta RS232 e USB para programação, pinos de acesso às entradas e saídas, terminais de controlo e uma área para implementação de hardware externo.

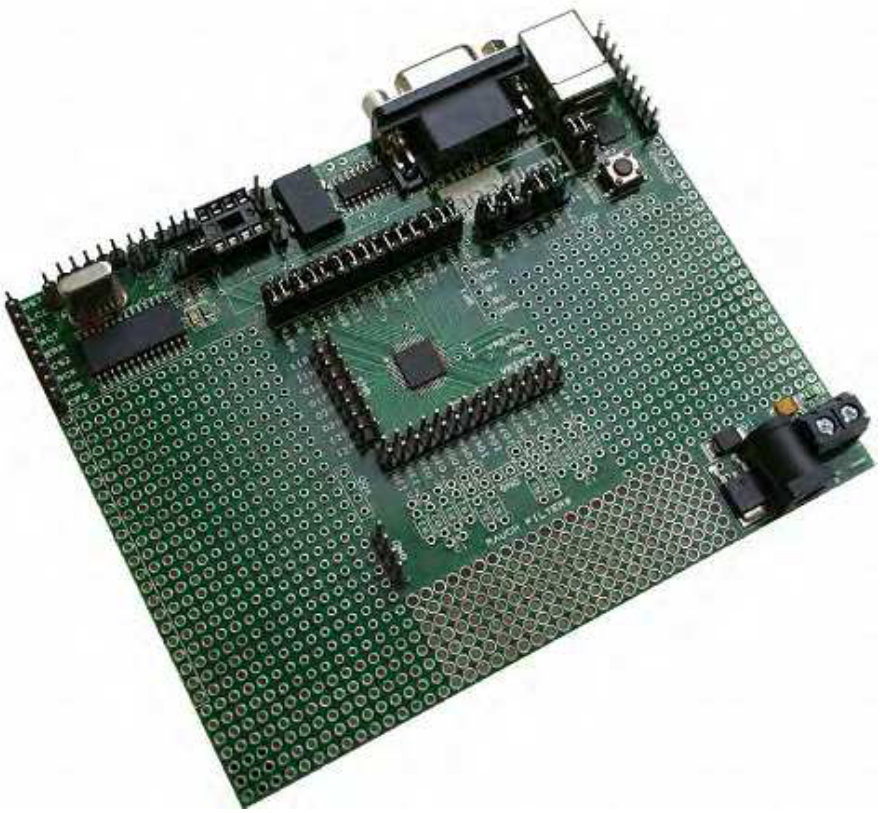


Figura 4.2: Anadigm AN231K04-DVLP3 (Anadigm, 2008)

Esta placa apresenta as seguintes caraterísticas principais:

- **USB** para comunicação com o PC;
- **UART RS232** para comunicação com o PC;
- **UART SPI** interface série, pelo protocolo SPI, que permite ligar a outros periféricos;
- **Área de montagem** *breadboard* para construção de circuitos;
- **Oscilador** *on-board* com frequência de 16 MHz;
- **EEPROM** para armazenamento de dados;
- **Programador** para transferência do programa para o dpASP (*dynamically programmable Analog Signal Processor*) através de um microcontrolador PIC.

A placa de desenvolvimento usada, AN231K04-DVLP3, permite a utilização imediata da FPAA, sendo possível integrar circuitos analógicos na própria placa. A configuração do dispositivo é feita no software de desenvolvimento Anadigm Designer e após completo é enviado via USB. O módulo de oscilação de 16 MHz para funcionamento dos CAB é outra das caraterísticas desta placa, bem como pinos nas entradas e saídas da FPAA. Uma outra característica, é o facto de poder ser acoplada a outras placas do mesmo tipo, sendo possível

criar uma cascata de FPAA, o que permite implementar sistemas mais complexos, e assim contornar diversas limitações inerentes à utilização de um único dispositivo (Anadigm, 2008).

A Figura 4.3 mostra a disposição dos componentes na placa.

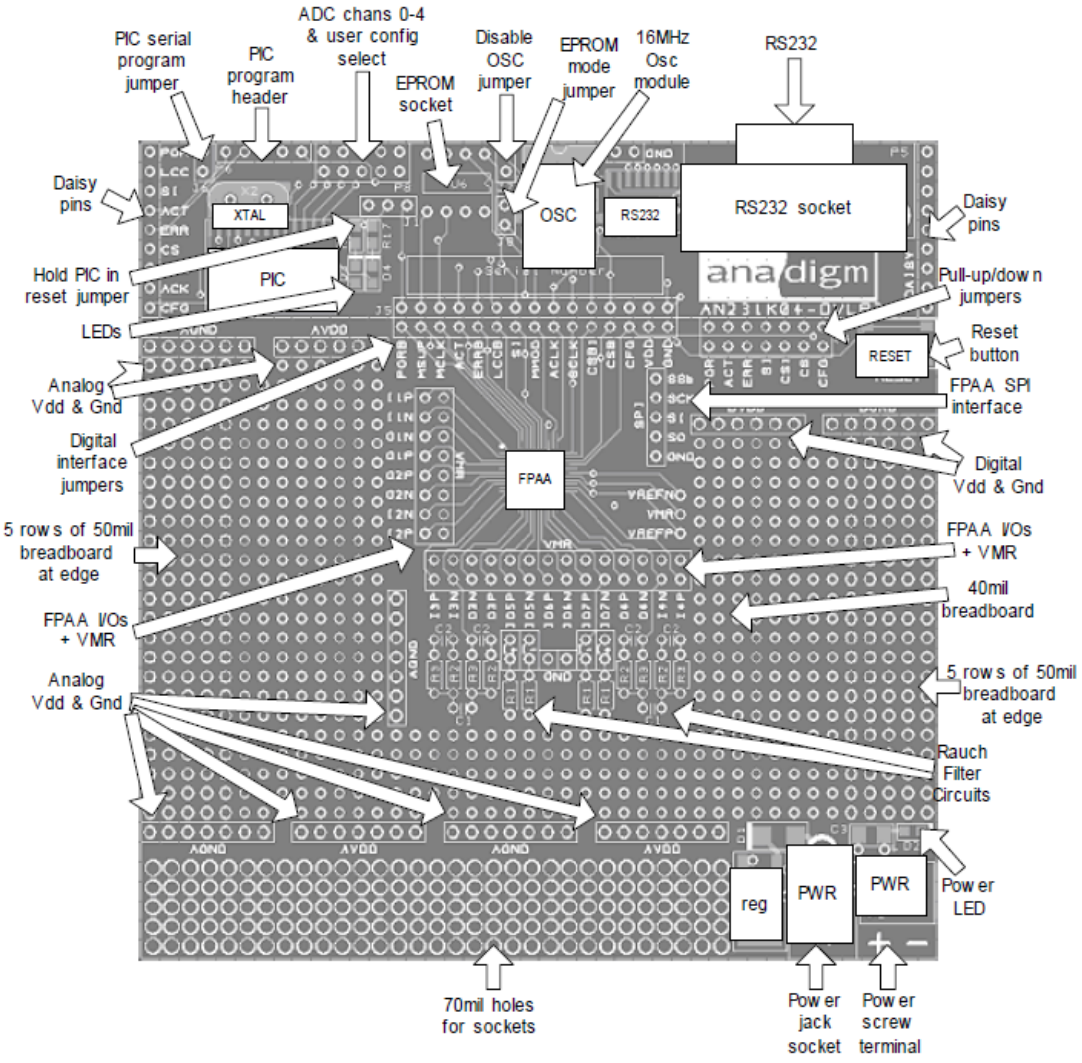


Figura 4.3: Anadigm AN231K04-DVLP3 *Layout* (Anadigm, 2008)

4.2.1 FPAA AN231E04

O diapositivo AN231E04, pertence à terceira geração de dpASP da Anadigm designada por *AnadigmApex*. A arquitetura deste dispositivo tem por base a segunda geração de FPAA da Anadigm, mas com maior número de células de entrada/saída e estas com maior capacidade de configuração e recursos disponíveis. A tensão de alimentação passou de 5 V para 3,3 V e houve uma redução na corrente de consumo duplicando a largura de banda. Na Figura 4.4 apresenta-se

a arquitetura deste dispositivo. Esta FPAA possui uma matriz de 2x2 CABs, e sete células

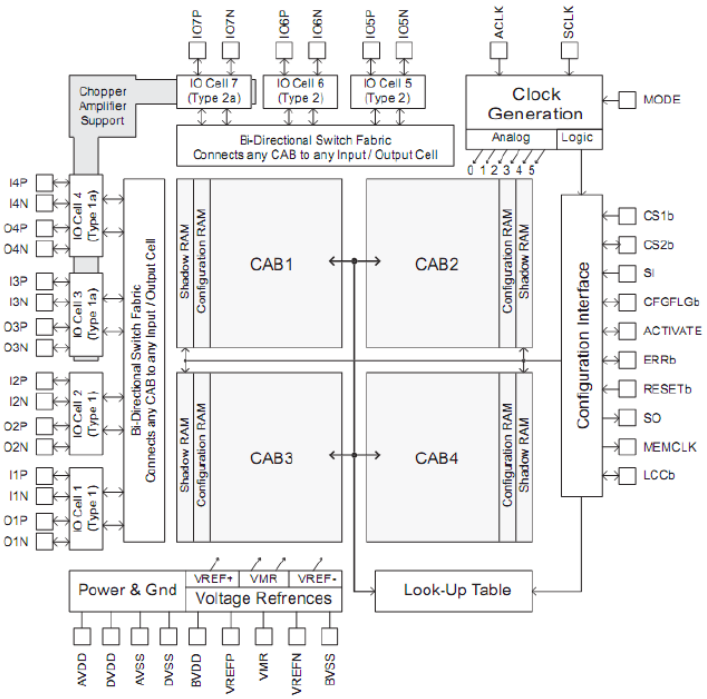


Figura 4.4: Arquitetura da Anadigm AN231E04 (Anadigm, 2006)

analógicas de entrada/saída. O processamento de sinal analógico é efetuado nas CABs através de circuitos de interrutor capacitivo. As CABs estão ligadas a uma LUT (*Look-Up Table*), que permite ajustar cada elemento que possua uma resposta dependente de um sinal, ou de uma base de tempo. A LUT pode ser usada para implementar funções de transferência arbitrárias entrada-saída, gerar sinais aleatórios, entre outros.

CARACTERISTICAS PRÍNCIPAIS

Como características principais possui, quatro CABs organizados numa matriz de 2x2, sete células de entrada e saída de dados, uma LUT, um conversor ADC (*Analog to Digital Converter*) do tipo SAR (*Successive Approximation Register*) em cada CAB, um bloco de alimentação de referência, um bloco gerador de sinal de relógio e um bloco de configuração de interface.

Existem três regiões de memória SRAM volátil dentro do dispositivo. A primeira, memória partilhada designa-se por *Shadow* SRAM, é a memória que recebe o código do programa durante a configuração ou reconfiguração. Esta memória serve como uma área de armazenamento temporário para os dados de configuração antes da sua transferência para a memória de configuração designada por *Configuration* SRAM. A segunda região, memória de

configuração, controla o comportamento dos circuitos de processamento de sinal analógico. A transferência da memória partilhada para a memória de configuração acontece num único ciclo de *clock*, minimizando a perturbação dos trajetos dos sinais analógicos. A terceira região de memória é a própria tabela de busca LUT, esta fornece valores de reposição para as zonas das memórias de configuração. A combinação CAB- LUT pode ser usada para criar funções não-lineares, como síntese de forma de onda arbitrária e tabela com funções de linearização de sensores.

Um bloco gerador de tensão de referência fornece as tensões de referência para cada um dos CABs do dispositivo e possui pinos externos para ligação de condensadores, excluindo dessa forma a necessidade de qualquer circuito externo de referência externa.

Os sinais analógicos são direcionados para dentro e fora do dispositivo pelas células de entrada/saída disponíveis, sendo que duas são do Tipo 1, duas do Tipo 1a, duas do Tipo 2 e uma do Tipo 2a. Tipo 1 e Tipo 1a são células de entrada/saída que contêm circuitos passivos e ativos que permitem: a entrada e saída direta de sinais, construir filtros ativos, circuitos de *Sample-and-Hold*, entradas e saídas digitais. Tipo 2 e Tipo 2a são células de Entrada/Saída mais simples que podem implementar entradas e saídas diretas de sinais, saída de tensão de referência e entrada e saída digital. Qualquer uma das células Tipo 1a ou Tipo 2a pode ter acesso ao amplificador do tipo *chopper* especializado, que permite a amplificação com precisão de sinais de entrada com tensão muito baixa.

4.3 SOFTWARE DE DESENVOLVIMENTO

Os softwares de desenvolvimento trazem inúmeras vantagens aos projetistas, tais como redução de tempo para implementação, rapidez na alteração de sistemas, entre outras. Seguindo este princípio foi desenvolvido pela Anadigm o software AnadigmDesigner2 (Figura 4.5) que é utilizado para o desenvolvimento dos circuitos analógicos que posteriormente serão enviados para FPAAs e assim programados na memória interna do circuito integrado.

Neste software são disponibilizados vários circuitos preconcebidos designados por CAMs (*Configurable Analog Modules*) tais como filtros bilineares ou biquadráticos, multiplicadores, somadores, amplificadores, comparadores, retificadores, integradores, etc. Todos estes circuitos usam recursos internos da FPAA, recursos esses que podem ser visualizados na tabela "*Resource Panel*" do mesmo software (Figura 4.5). O AnadigmDesigner2 possui ainda duas ferramentas específicas, uma para a criação de filtros e outra para sistemas PIDs representadas na Figura 4.9 e Figura 4.10, respetivamente. Estas ferramentas possibilitam que sejam desenvolvidos sistemas apenas com a passagem de parâmetros, o diagrama do circuito é criado automaticamente, bastando apenas transferir o diagrama para o AnadigmDesigner2.

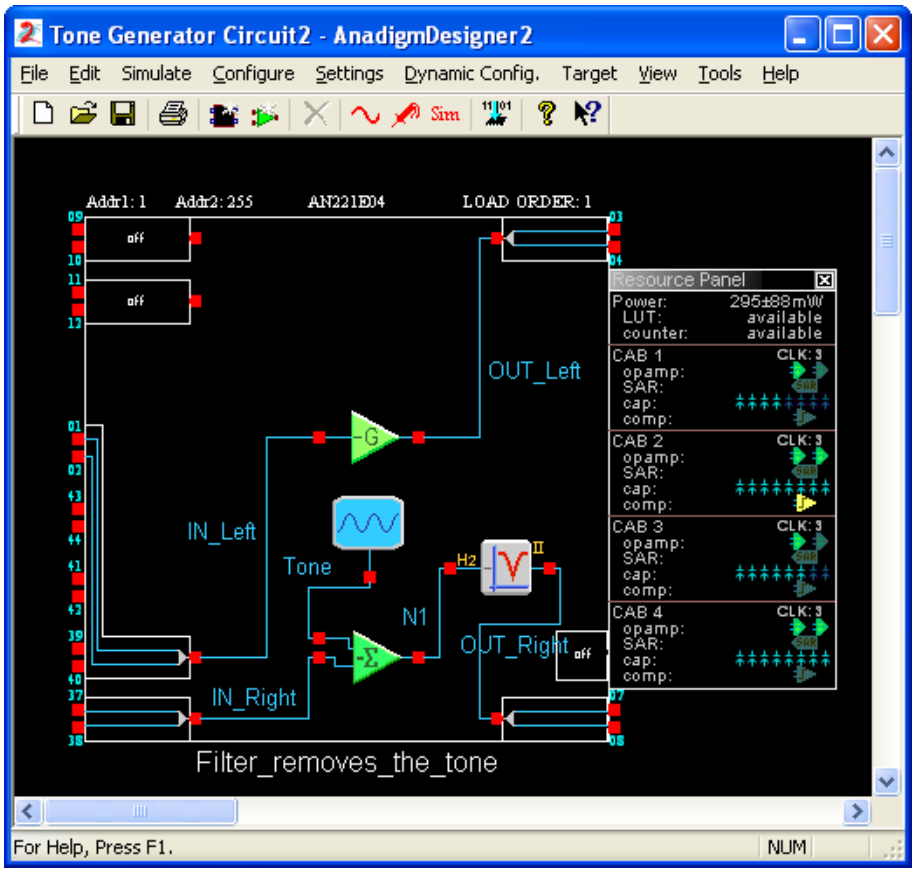


Figura 4.5: Software AnadigmDesigner2

Existem inúmeras CAM para os mais diversos tipos de aplicações. Na Figura 4.6 estão listadas as disponíveis na FPAA AN231E04.

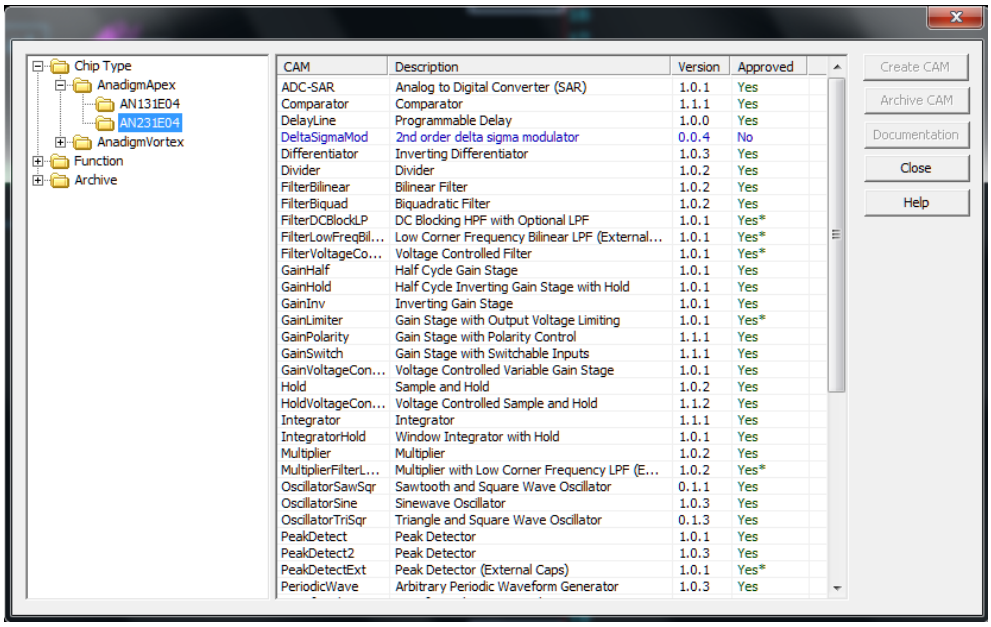


Figura 4.6: Lista de CAM da FPAA AN231E04

Para a implementação de sistemas fracionários existem duas que merecem mais atenção, as quais são descritas a seguir.

FILTRO BILINEAR

Esta CAM cria um filtro bilinear de um polo, podendo ser passa-baixa, passa-alta e passa-tudo, podendo ainda ser do tipo polo-zero (Figura 4.7). Este filtro tem como parâmetros de configuração a frequência de corte, ganho e passa-banda. Para a aplicação em causa usa-se o filtro do tipo polo-zero.

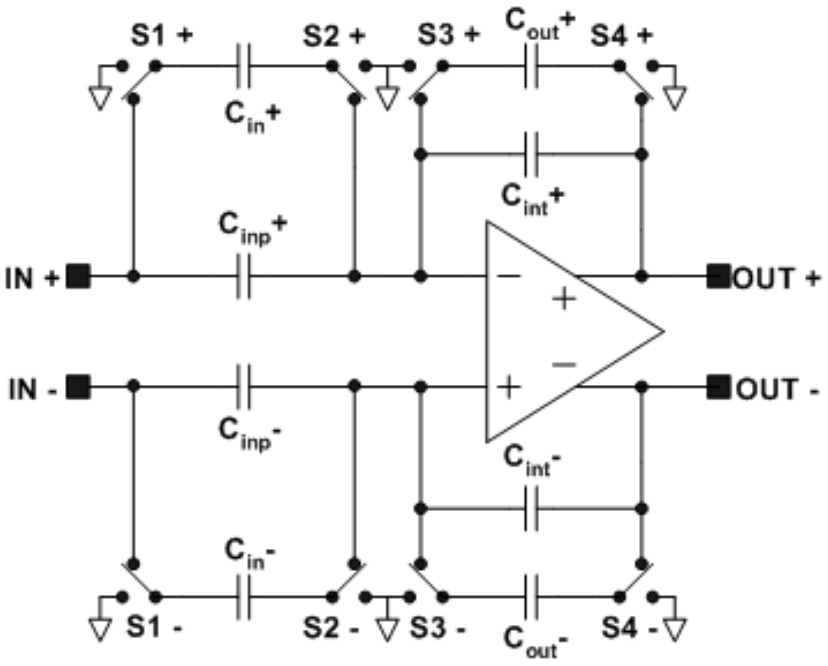


Figura 4.7: Cicuito da CAM Bilinear Filter

Na equação (4.1) está representada função de transferência do filtro polo-zero desta CAM:

$$\frac{V_{Out}(s)}{V_{In}(s)} = -\frac{G_H\,(s+2\pi f_z)}{s+2\pi f_p} \tag{4.1}$$

O ganho de corrente contínua (G_L) é função das frequências do polo e zero, bem como do ganho às altas frequências, como representado na equação (4.2).

$$G_L = G_H\left(\frac{f_z}{f_p}\right) \tag{4.2}$$

FILTRO BIQUADRÁTICO

Esta CAM cria um filtro biquadrático de dois polos, podendo ser passa-baixa, passa-alta, passa-banda e rejeita-banda, podendo ainda ser do tipo polo-zero (Figura 4.8). Este filtro tem como parâmetros de configuração a frequência de corte, ganho, e fator de qualidade. Para a aplicação em causa utiliza-se o filtro do tipo polo-zero.

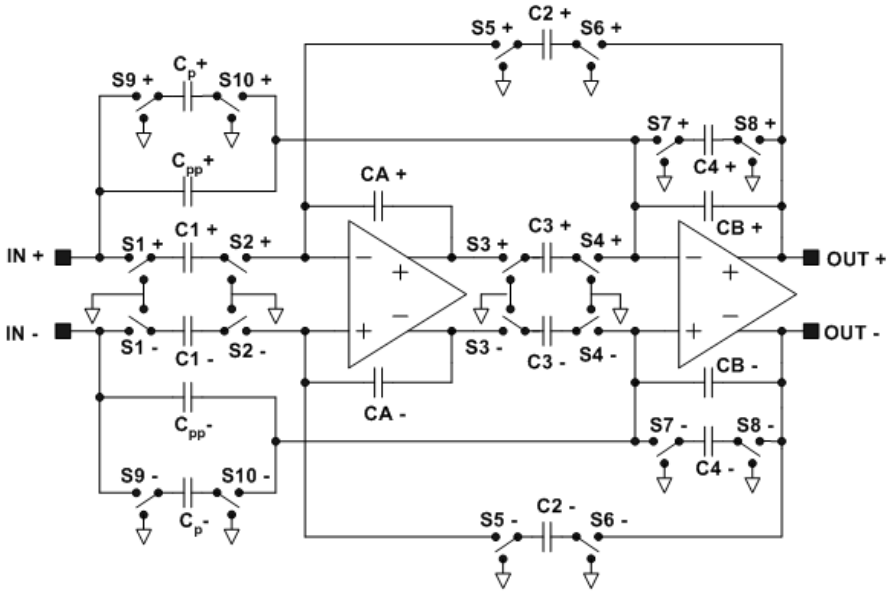


Figura 4.8: Cicuito da CAM Biquad Filter

Na equação (4.3) está representada função de transferência do filtro polo-zero desta CAM:

$$\frac{V_{Out}(s)}{V_{In}(s)} = -\frac{G_H\left(s^2+\frac{2\pi f_z}{Q_z}s+4\pi^2 f_z^2\right)}{s^2+\frac{2\pi f_p}{Q_p}s+4\pi^2 f_p^2} \tag{4.3}$$

O ganho de corrente contínua (G_L) é função das frequências dos polos e zeros, bem como do ganho às altas frequências, como representado na equação (4.4).

$$G_L = G_H\left(\frac{f_z}{f_p}\right)^2 \tag{4.4}$$

A ferramenta AnadigmFilter (Figura 4.9) permite desenvolver e implementar filtros de uma forma bastante simples e rápida, bastando para tal selecionar o tipo de filtro, ajustar os parâmetros num diagrama de Bode, e por fim enviar para o AnadigmDesigner2, sendo o filtro gerado automaticamente.

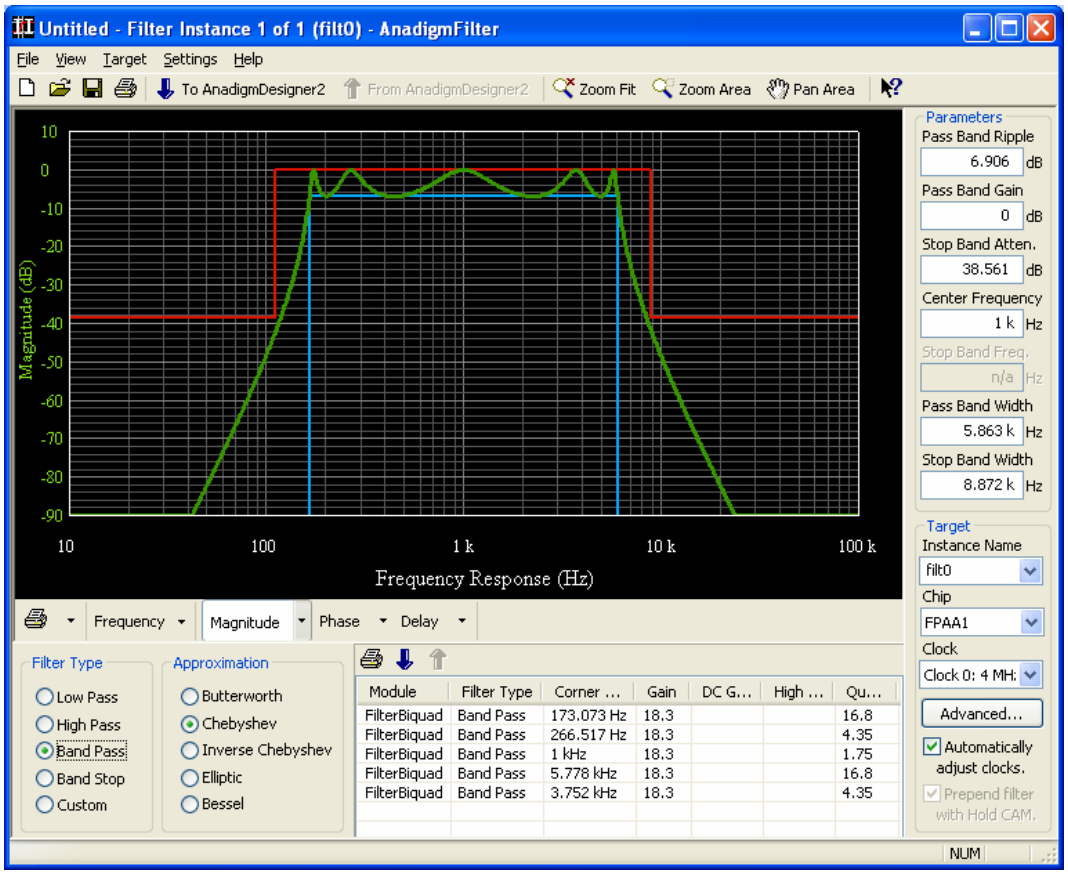


Figura 4.9: Ferramenta AnadigmFilter

PASSOS PARA REALIZAR UM FILTRO:

- Selecionar o tipo de filtro;
- Selecionar a aproximação;
- Configurar os parâmetros ou arrastar as linhas do diagrama de Bode para o valor pretendido;
- Transferir o filtro para o AnadigmDesigner2;
- Transferir o filtro para a FPAA.

Por sua vez a ferramenta AnadigmPID (Figura 4.10) permite realizar controladores PID analógicos tendo para isso o projetista de selecionar o tipo de controlador (P, PI, PD ou PID), e ajustar os parâmetros (K_p, K_i, K_d), e por fim enviar para o AnadigmDesigner2, sendo o controlador gerado automaticamente.

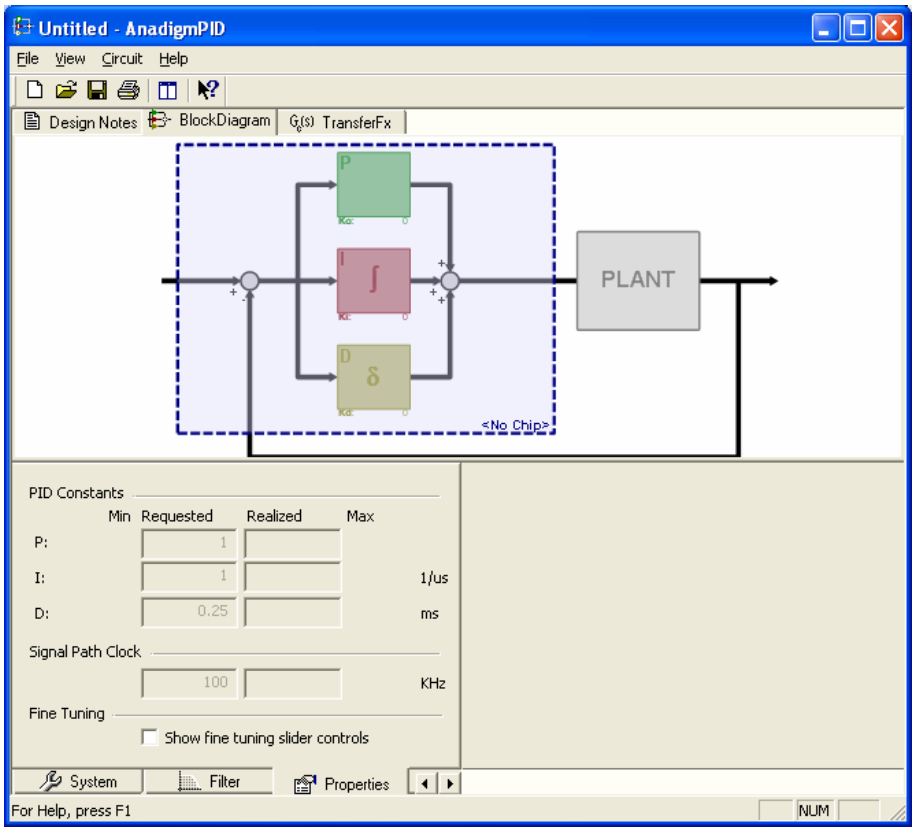


Figura 4.10: Ferramenta AnadigmPID

PASSOS PARA REALIZAR UM CONTROLADOR PID:

- Selecionar o tipo de controlador:
  - P, PI, PD, ou PID
- Ajustar os parâmetros:
  - K_p, K_i, K_d
- Transferir o controlador para o AnadigmDesigner2;
- Transferir o controlador para a FPAA.

4.3.1 VERIFICAÇÃO DO SOFTWARE

Para validar a funcionalidade do software AnadigmDesigner2, foram simulados dois circuitos representados em baixo, sendo eles o amplificador inversor e o filtro bilinear. Estas simulações foram efetuadas no simulador da aplicação.

AMPLIFICADOR DE GANHO NEGATIVO

Neste circuito pretende-se fazer a simulação de um amplificador inversor, através da utilização da CAM *Inverting gain stage*. O ganho da CAM pode ser configurado, neste caso foi de -2, como se pode ver na Figura 4.11.

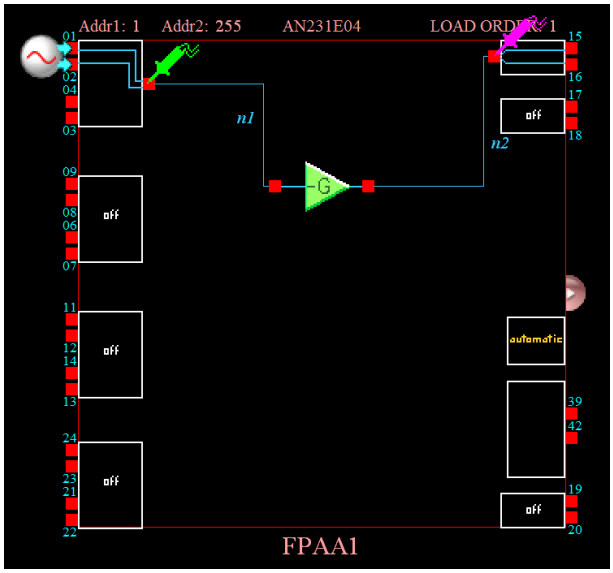


Figura 4.11: Esquema de um amplificador com ganho negativo

Na entrada do dispositivo é aplicada uma onda com uma determinada frequência e amplitude, e na saída irá aparecer a mesma onda desfasada 180º. Ou seja, quando o valor de pico da onda de entrada é máximo positivo, o valor de pico da onda de saída será máximo negativo. Constate-se que tanto o valor de amplitude como o de frequência se mantêm (Figura 4.12).

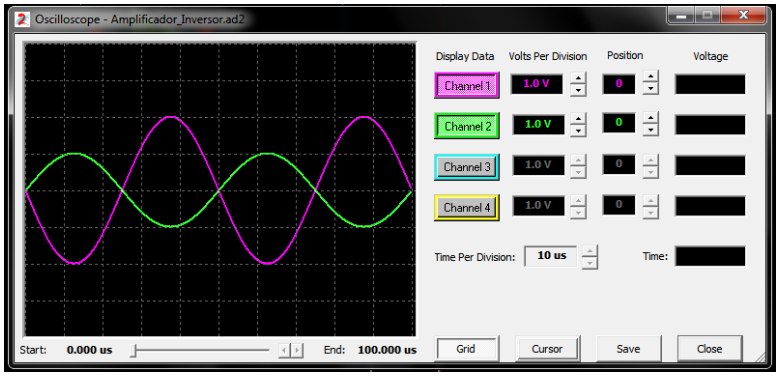


Figura 4.12: Resultado da simulação de um amplificador com ganho negativo

FILTRO PASSA-BAIXO

Para a realização desta simulação procedeu-se à utilização da CAM FilterBilinear, estando esta a funcionar no modo inversor e com uma frequência de corte de 100 kHz. Foi utilizado também um gerador de sinais com uma onda quadrada (Figura 4.13).

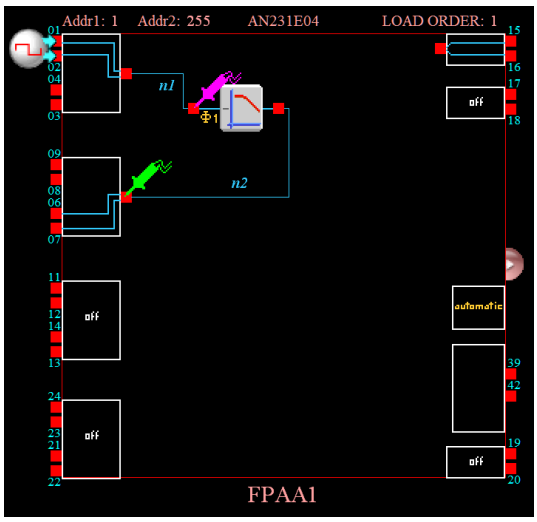


Figura 4.13: Esquema de um integrador

Dado que o sinal de entrada tem a forma de uma onda quadrada logo este possui uma frequência fundamental e vários harmónicos. Se a frequência fundamental e os harmónicos estiverem dentro da banda passante, então a onda quadrada terá aproximadamente a mesma forma na saída, tal como acontece na experiência (Figura 4.14). Atente-se que existe um atraso na onda de saída em relação à de entrada. Dado que o sinal de entrada tem a forma de uma onda quadrada logo este possui uma frequência fundamental e vários harmónicos. Se a frequência fundamental e os harmónicos estiverem dentro da banda passante, então a onda quadrada terá aproximadamente a mesma forma na saída.

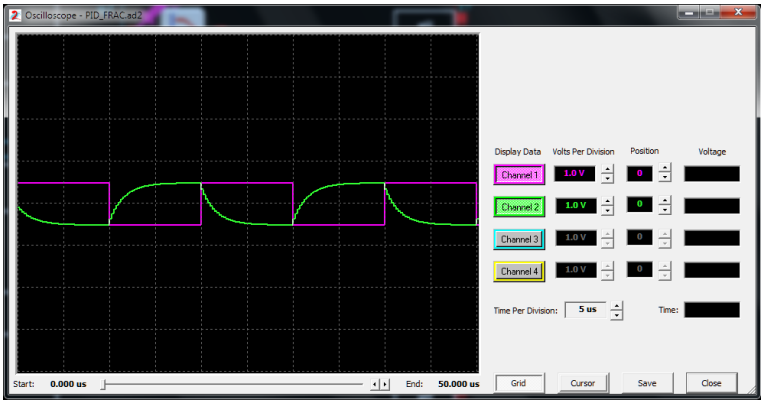


Figura 4.14: Resultado da simulação de um integrador

4.4 PLACA DE AQUISIÇÃO HILINK

A placa de aquisição HILINK constitui um interface entre o meio externo e o Matlab/Simulink, permitindo que rapidamente se desenvolvam sistemas controlo em tempo real quer para teste em laboratório quer para utilizações em aplicações industriais. Funciona como um sistema de aquisição de dados DAQ. A placa é conectada a um PC com Matlab via RS232 através de um cabo Null Modem. Com esta placa é disponibilizada uma biblioteca que permite a interligação entre o Simulink e as entradas/saídas disponíveis no hardware. A alimentação da placa é feita através de uma fonte DC externa de 12 V/5 A. Na Figura 4.15 está representado o *layout* desta placa.

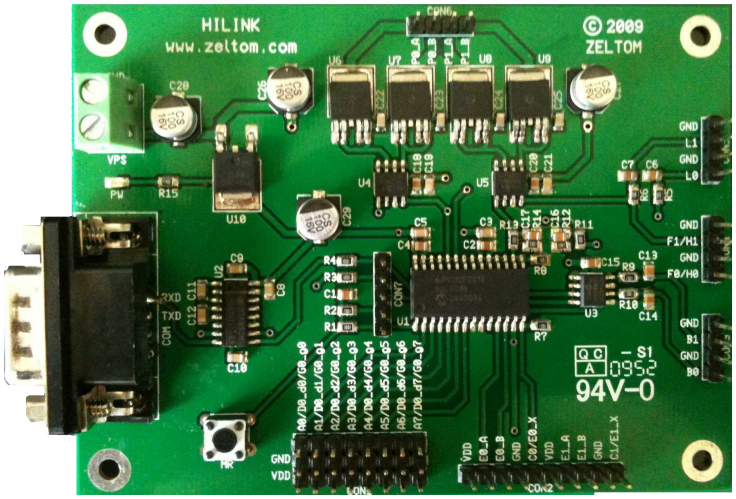


Figura 4.15: Layout da placa Hilink (Zeltom, 2011)

4.4.1 HARDWARE

O hardware da placa de aquisição Hilink, está desenvolvido tendo como base o microcontrolador dsPIC30F2012 da Microchip. Este dsPIC trabalha a 16 bit e tem 12 kB de memória *flash* para programa e 1 kB de memória SRAM para dados. A sua frequência de relógio está configurada no oscilador interno para 7,37 x 106 x 16 = 117,92 MHz. É um microcontrolador com muitos periféricos o que o torna ideal para aplicações em tempo real. A plataforma de desenvolvimento Hilink possui oito entradas analógicas de 12 bit (A0 - A7), duas entradas do tipo captura de sinal de 16 bit (C0 - C1), duas entradas *encoder* de 16 bit (E0 - E1), uma entrada digital de 8 bit (D0_d0 - D0_d7), duas saídas analógicas de 12 bit (B0 - B1), duas saídas de frequência (F0 - F1), duas saídas tipo pulso (H0 - H1) e uma saída de 8 bit digital (G0_g0 - G0_g7). Na Figura 4.16 está representado o diagrama de blocos da placa Hilink.

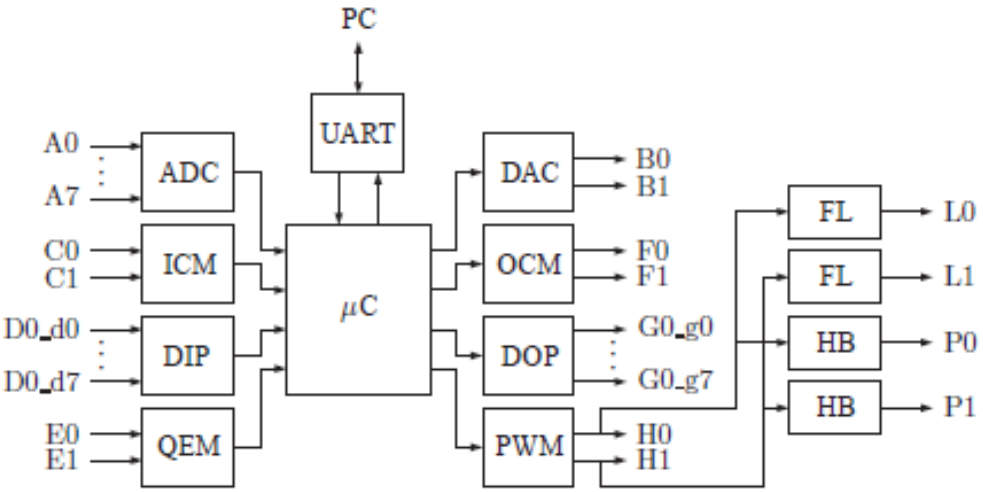


Figura 4.16: Diagrama de blocos Hilink (Zeltom, 2011)

4.4.2 SOFTWARE

O software da placa Hilink foi desenvolvido sob a plataforma Matlab/Simulink, e logo totalmente compatível com esta. Vem com uma biblioteca de blocos para Simulink que estão diretamente ligados à correspondente entrada e saída da placa. Esta biblioteca é constituída por um bloco de entrada analógica (A0), bloco de entrada de captura de sinal (C0), bloco de entrada *encoder* (E0), bloco de entrada digital (D0), bloco de saída analógica (B0), bloco de saída em frequência (F0), bloco de saída tipo pulso (H0) e por fim o bloco de saídas digitais (G0). A Figura 4.17 apresenta os blocos constituintes desta biblioteca.

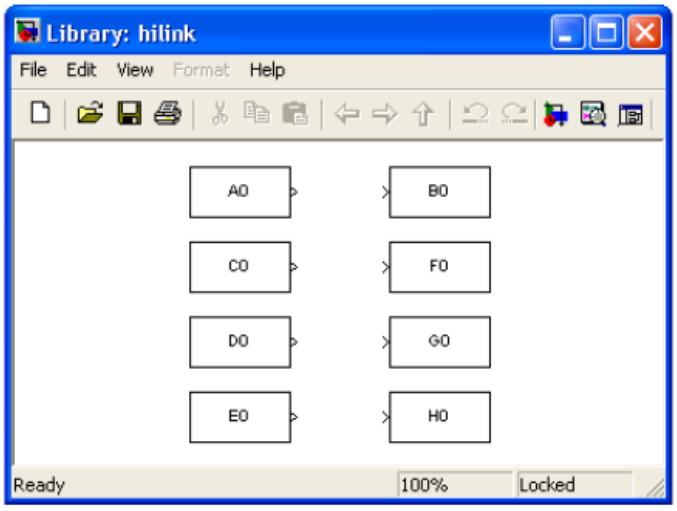


Figura 4.17: Biblioteca de blocos Simulink (Zeltom, 2011)

4.4.3 APLICAÇÕES

A crescente evolução dos sistemas informáticos, em particular dos computadores faz com que a aquisição de dados seja hoje em dia a plataforma mais utilizada nos sistemas de aquisição de dados. O baixo custo associado ao desempenho, a flexibilidade, e facilidade de utilização são fatores que contribuem para a sua ampla utilização em diversos campos de ação como na indústria, construção civil, mecânica e mesmo nos laboratórios científicos. A sua aplicação pode ir desde a monitorização, medição, simulação, diagnóstico e controlo de processos. Lista-se a seguir algumas aplicações possíveis de implementar com a placa de aquisição Hilink:

- Ajuste e otimização de parâmetros;
- Modelação, análise e desenho de sistemas de controlo;
- Sistemas de controlo em tempo real;
- Simulação de circuitos em malha fechada;
- Prototipagem de sistemas de controlo em tempo real;
- Realização de experiências de aprendizagem em laboratórios;
- Aquisição de sinais em tempo real.

4.4.4 TAXA DE AMOSTRAGEM

Com a placa de aquisição Hilink é possível de implementar sistemas de controlo em tempo real, com taxas de amostragem até 3,8 kHz. O valor da taxa de amostragem a implementar em cada projeto embora dependa da performance do PC, depende essencialmente do número de entradas e saídas em uso. Este valor é calculado considerando a equação representada em (4.5).

$$f = \frac{11520}{2max(n_i, n_o) + 1} \tag{4.5}$$

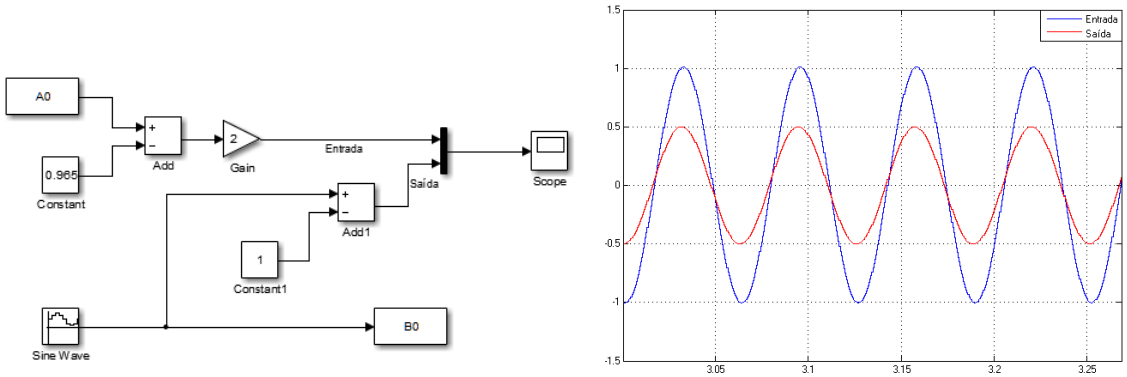
Onde $n_i \leqslant 8$ representa o número de entradas usadas no projeto, $n_o \leqslant 8$ o número de saídas e f a frequência de amostragem. Nos blocos do Simulink é necessário definir o período de amostragem, considerando que $T = 1/f$. Se considerar como exemplo $n_i = 4$ e $n_o = 4$ a taxa de amostragem máxima que se pode obter é de $f = 1280$ Hz, nesta situação é utilizado como taxa de amostragem o valor 1024 Hz que corresponde a um período de $T = 1/1024 = 976,5625$ μs. A taxa de amostragem máxima atingível é de 650 Hz quando no projeto são usadas oito entradas e oito saídas.

4.4.5 VERIFICAÇÃO DA CONECTIVIDADE E TESTE

Para validar a funcionalidade da placa de desenvolvimento HILINK, foi efetuada a simulação e teste descrito a seguir.

AMPLIFICADOR NÃO INVERSOR

O objetivo do circuito apresentado na Figura 4.18(a) foi verificar a conectividade da placa HILINK. Para tal usou-se um gerador de sinais do Matlab ligado à saída analógica B_0, à entrada A_0 ligou-se um osciloscópio. A onda resultante está apresentada na Figura 4.18(b), verificando-se que está em fase com a onda injetada na placa apesar de um ligeiro atraso resultante da comunicação RS232.



(a) Diagrama de blocos Simulink®

(b) Formas de onda da entrada e saída

Figura 4.18: Amplificador não inversor

4.5 SISTEMA SERVO

Um atuador comum nos circuitos de controlo é o motor CC. É possível simular o comportamento destes componentes no Simulink. Para isso basta ter um modelo que represente o seu comportamento dinâmico. Este é por norma, uma simplificação do sistema real e a sua complexidade é a suficiente para capturar as dinâmicas relevantes. Para a realização desta tese de dissertação foi utilizado o sistema servo modular da Inteco®, cujo os blocos constituintes estão ilustrados na Figura 4.19. Este sistema consiste em vários módulos, montados numa régua metálica e ligados entre si através de pequenos acopladores. Os módulos são ligados em cadeia. O motor DC com o gerador taquimétrico aparece no topo da frente e o sistema de engrenagens com o disco de saída no topo de trás da régua metálica, conforme mostra a Figura 4.20.

Figura 4.19: Sistema servo modular da Inteco®

O motor DC pode ser acoplado com os seguintes módulos: módulo de inércia, módulo de travagem magnética, módulo de folga e módulo de engrenagens com disco de saída. O deslocamento angular do veio do motor é medido através de um codificador incremental. O codificador pode ser colocado entre dois quaisquer módulos para medir o ângulo de rotação. O tacogerador está ligado diretamente ao motor e gera uma tensão proporcional à velocidade angular.

4.5.1 MODELO MATEMÁTICO DO SISTEMA SERVO

O modelo linear do sistema servo está ilustrado na Figura 4.21. Assume-se que a indutância da armadura do motor é desprezável. Do mesmo modo, despreza-se quaisquer atritos assim como a saturação. Com base nestas considerações, as equações elétrica e mecânica do modelo do sistema são, respetivamente, dadas por:

$$v(t) = Ri(t) + K_e w(t) \tag{4.6}$$

$$J\dot{w}(t) + Bw(t) = K_t i(t) \tag{4.7}$$

Onde as variáveis envolvidas estão listadas na Tabela 4.1.

Na Figura 4.21 estão representadas esquematicamente as grandezas que definem o comportamento do motor.

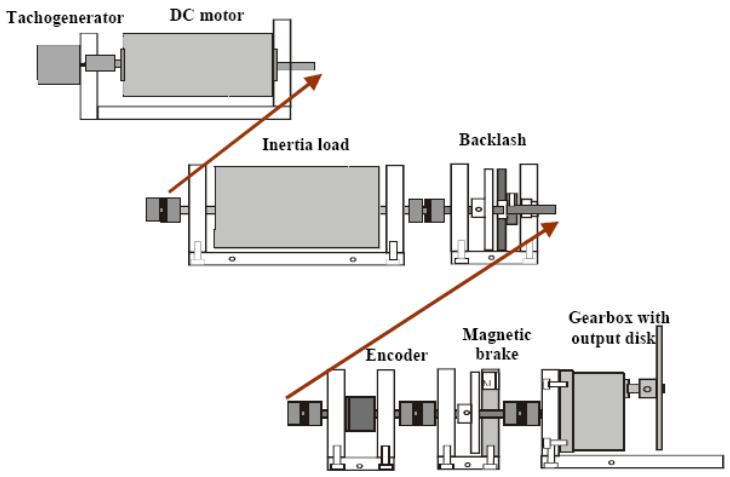


Figura 4.20: Disposição dos módulos do servo na régua metálica

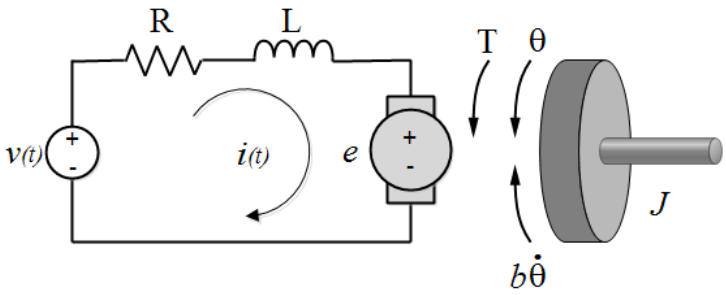


Figura 4.21: Diagrama elétrico de um motor CC

Combinando as expressões (4.6) e (4.7), obtém-se:

$$T_s\dot{w}(t) + w(t) = K_{sm}v(t) \tag{4.8}$$

em que a constante de tempo do motor T_s e o ganho do motor K_{sm} são dados respetivamente por:

$$T_s = \frac{RJ}{BR + K_eK_t} \tag{4.9}$$

$$K_{sm} = \frac{K_t}{BR + K_eK_t} \tag{4.10}$$

Tabela 4.1: Parametros do motor

Parametro	Descrição	Unidades
$v(t)$	Tensão aplicada	V
$i(t)$	Corrente da armadura	A
$w(t)$	Velocidade angular do rotor	rad/sec
R	Resistência do enrolamento da armadura	Ω
J	Momento de inércia das partes rotativas	kg.m^2
B	Coeficiente de atrito devido ao atrito viscoso	N.m.s/rad
$K_e w(t)$	Força contra-electromotriz (f.c.e.m.)	V
$T = Ki(t)$	Binário electromecânico	N.m

A função de transferência da velocidade do motor é então dada na forma:

$$G(s) = \frac{w(s)}{V(s)} = \frac{K_{sm}}{T_s s + 1} \tag{4.11}$$

Considerando $K_s = K_{sm} v_{max}$, a função de transferência que relaciona a velocidade de saída $w(s)$ e a entrada do sistema $U(s)$ é dada por:

$$G(s) = \frac{w(s)}{U(s)} = \frac{K_s}{T_s s + 1} \tag{4.12}$$

em que as constantes T_s e K_s são os parâmetros do modelo do sistema servo.

4.6 SINTONIA DE CONTROLADORES

Existem inúmeros critérios de sintonia de controladores PID: métodos baseados na resposta ao degrau, na resposta em frequência, em medidas integrais de desempenho, etc. Os mais conhecidos baseiam-se em heurísticas, como são exemplos os métodos propostos por Ziegler e Nichols, Cohen e Coon e Shinskey. Estes métodos podem ser aplicadas com ou sem o conhecimento do modelo matemático do processo. Se o modelo é conhecido, então podem ser usadas várias técnicas de projeto para a determinação dos parâmetros do PID que satisfazem as especificações da resposta transitória e em regime permanente do sistema em malha fechada, mas se o modelo não é conhecido, então recorre-se a métodos experimentais. Ziegler e Nichols propuseram em 1942 dois métodos que permitem criar um conjunto de regras, o primeiro partindo da resposta ao degrau de um sistema em malha aberta, o segundo avaliando a resposta do sistema em malha fechada.

4.6.1 REGRAS DE ZIEGLER-NICHOLS EM MALHA ABERTA

Em 1942, Ziegler e Nichols verificaram que a resposta ao degrau de um largo número de processos é representada por uma curva em forma de "S". Esta curva é característica de muitos sistemas de ordem superior que podem ser aproximados a sistemas de primeira ordem com a adição de um atraso de L segundos:

$$\frac{Y(s)}{U(s)} = \frac{Ke^{-Ls}}{Ts+1} \tag{4.13}$$

Tabela 4.2: Regras de ajuste de Ziegler-Nichols, em que $R = K/T$

Controlador	K_p	T_i	T_d
P	$\frac{1}{RL}$	∞	0
PI	$\frac{0,9}{RL}$	$\frac{L}{0,3}$	0
PID	$\frac{1,2}{RL}$	$2L$	$0,5L$

Na Tabela 4.2 estão listados os parâmetros do controlador propostos por Ziegler e Nichols para ajuste do ganho proporcional K_p, a constante de integração T_i e de derivação T_d. Depois de determinar T_i e T_d é possível calcular os ganhos K_i e K_d:

$$K_i = \frac{K_p}{T_i}, \qquad K_d = K_p T_d \tag{4.14}$$

Tal como referido, o método de Ziegler-Nichols em malha aberta pode ser aplicado quando a curva de resposta ao degrau unitário de entrada apresentar o aspeto de uma curva em "S". Essa curva de resposta ao degrau unitário pode ser gerada experimentalmente ou a partir de uma simulação.

A curva com o formato em "S"pode ser caracterizada por duas constantes, o atraso L e a constante de tempo T e por um ganho K. Os parâmetros L e T são determinados desenhando uma linha tangente no ponto de inflexão da curva com o formato de "S"e determinando a intersecção da linha tangente com o eixo do tempo. Já K representa o valor final da resposta do sistema (Figura 4.22). O ajuste do controlador PID, segundo este método, introduz no sistema dois zeros em $s = -1/L$ se o projeto for realizado em tempo contínuo.

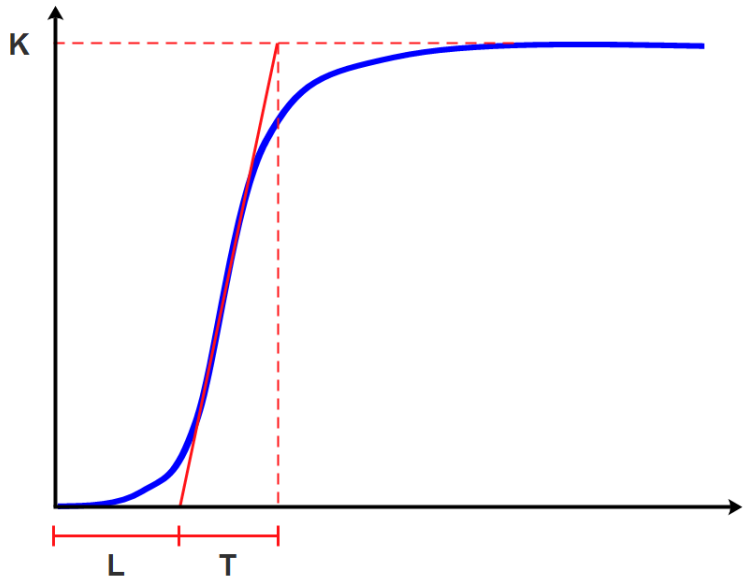


Figura 4.22: Resposta ao Degrau

Determinação dos parâmetros K, L e T

Para o desenvolvimento deste trabalho começou-se por implementar o diagrama de blocos em Simulink correspondente à aplicação das heurísticas de Ziegler-Nichols de malha aberta representado na Figura 4.23.

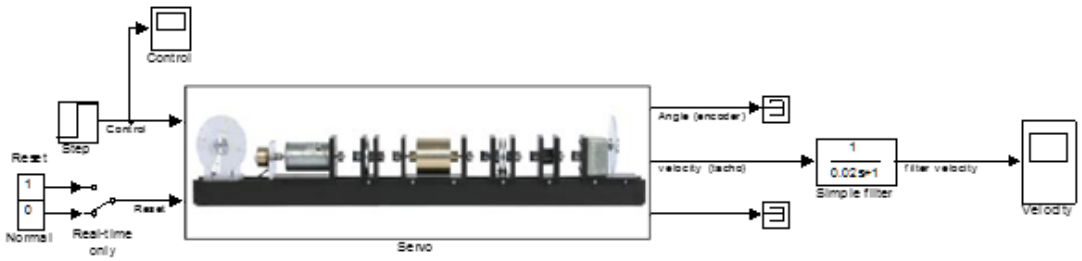


Figura 4.23: Sistema de controlo em malha aberta

A partir do gráfico da simulação da resposta do sistema servo em malha aberta em Matlab/Simulink (Figura 4.24) determinam-se os parâmetros K, L e T, aplicando o procedimento atrás descrito. No Anexo C apresenta-se a função `zn_ma` desenvolvida em Matlab para cálculo dos parâmetros do modelo do sistema servo.

Foram assim obtidos em Matlab os valores dos parâmetros K, L e T:

$$K = 187,2106; \qquad L = 0,1753; \qquad T = 1,1841 \tag{4.15}$$

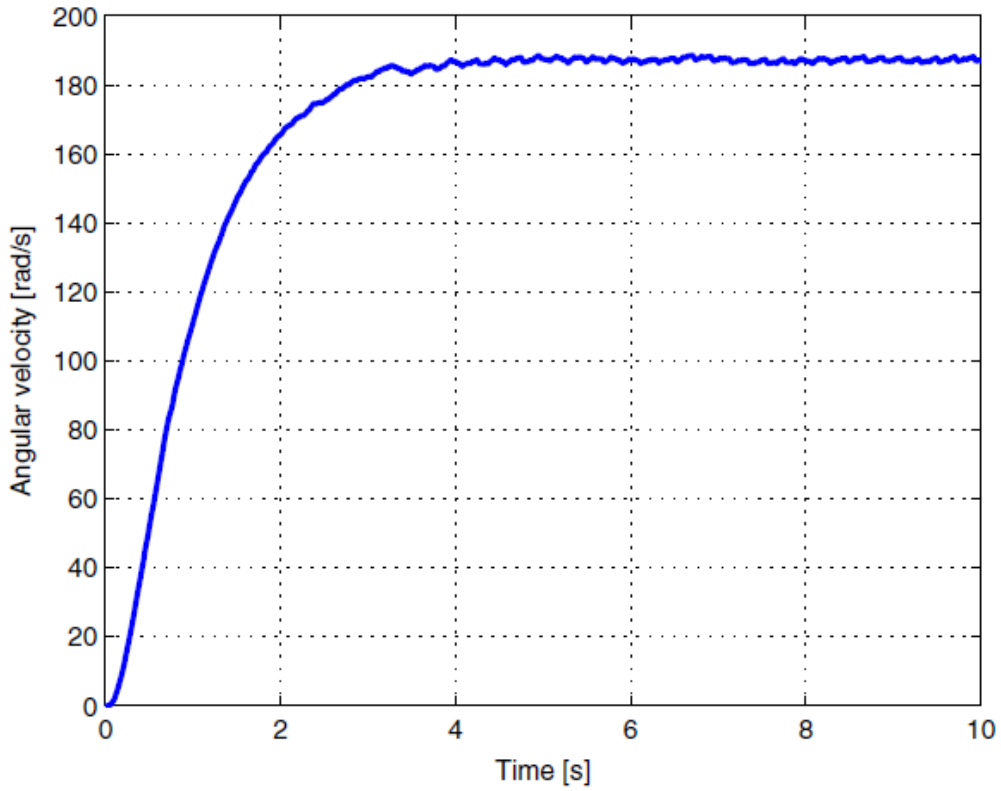


Figura 4.24: Resposta do sistema servo em malha aberta a um degrau unitário

A curva característica aproximada deste sistema é representada por:

$$\frac{Y(s)}{U(s)} = 187,2106\frac{e^{-0,1753*s}}{1,1841s+1} \tag{4.16}$$

Para fazer a sintonia dos diversos controladores em malha aberta, Ziegler-Nichols apresentou a Tabela 4.2, em que tendo como entrada os valores K, L e T calculados, apresenta como saída os parâmetros do controlador K_p, T_i e T_d.

Na Tabela 4.3 são apresentados os controladores P, PI, PID definidos pelas regras de Ziegler e Nichols.

Tabela 4.3: parâmetros dos controladores P, PI, PID

Controlador	K_p	T_i	T_d
P	0,0361	∞	0
PI	0,0325	0,0556	0
PID	0,0433	0,1235	0,0038

5. SIMULAÇÃO E TESTE

Como já foi descrito nos capítulos anteriores existem várias formas de implementar sistemas fracionários, neste estudo o foco foi a implementação destes sistemas nos FPAAs. Os FPAAs apesar de implementarem circuitos analógicos não o fazem de uma forma direta, toda a sua implementação é feita em circuitos preconcebidos designados por CAM, assim não é possível a implementação recorrendo a fractâncias nas malhas de realimentação dos AMPOPs. Do estudo realizado, verifica-se que uma das formas de implementar sistemas fracionários em FPAA é recorrendo a aproximações inteiras do operador s^{α} e implementando-as sob a forma de filtros passivos bilineares ou biquadráticos.

Neste capítulo são estudados vários tipos de controladores PID fracionários usados no controlo da velocidade de um motor de corrente contínua. A simulação e teste destes controladores é feita recorrendo ao diagrama apresentado na Figura 4.1 em conjunto com o Matlab/Simulink.

5.1 IMPLEMENTAÇÃO NUM FPAA

Como foi dito nos capítulos anteriores os FPAAs permitem implementar circuitos analógicos através da configuração dos módulos CAMs. A implementação de circuitos fracionários

através de sua aproximação inteira usa essencialmente duas destas CAMs, sendo elas o filtro bilinear e o filtro biquadrático. A ligação em cascata destas CAMs permite implementar funções de ordem superior a dois. A Anadigm AN231E04 possui quatro CABs, o que limita a implementação ao máximo de quatro filtros biquadráticos. Além dos filtros também é necessário o ganho, o que limita o máximo de filtros biquadráticos sem esgotar os recursos da FPAA a três. Para o âmbito desta tese considera-se como suficiente uma aproximação de terceira ordem, mas para efeito de testes e aproveitando os recursos da FPAA usa-se aproximações de quarta ordem.

O primeiro passo na implementação de sistemas fracionários nos FPAAs é a escolha da ordem do operador fracionário. Nestes testes escolhe-se um integrador e derivador de ordem quatro, assim considera-se a implementação de um integrador $s^{-0,4}$. Para a frequência baixa (wb) e alta (wh) são atribuídos os valores de 1×10^3 rad/s e 1×10^6 rad/s, os quais foram escolhidos de forma a conseguir implementar os filtros dentro dos limites da FPAA. Para o valor da ordem da aproximação (N) considera-se então o valor quatro, e para a ordem do operador fracionário (γ) o valor de $\gamma = -0,4$. A implementação em Matlab fica da seguinte forma:

Filtro de Oustaloup em Matlab

```
Matlab
gamma=-0.4;
N=4;
wb=1000;
wh=1e06;
G = ousta_fod(Gamma,N,wb,wh)
```

O que resulta na seguinte expressão:

$$H(s) = \frac{0,003981s^4 + 2881s^3 + 3,134 \times 10^8 s^2 + 5,749 \times 10^{12} s + 1,585 \times 10^{16}}{s^4 + 3,627 \times 10^5 + 1,977s^3 \times 10^{10} s^2 + 1,818 \times 10^{14} s + 2,512 \times 10^{17}} \tag{5.1}$$

Seguidamente aplica-se a função $tf2sos(n,d)$ representada abaixo, que converte em equivalentes de segunda ordem a função de transferência dada.

Divisão em dois filtros: H1 e H2 e ganho: A

```
Matlab
[n,d] = tfdata(G,'v');
[sos,A] = tf2sos(n,d);)
H1 = tf(sos(1,1:3),sos(1,4:6))
H2 = tf(sos(2,1:3),sos(2,4:6))
```

H_1 e H_2 representam os filtros de segunda ordem resultantes da decomposição da função de transferência representada na equação (5.1):

$$H(s) = A \times H_1(s) \times H_2(s) \tag{5.2}$$

$$\begin{gathered} A = 0,004 \\ H_1(s) = \frac{s^2 + 7,02 \times 10^5 s + 6,31 \times 10^{10}}{s^2 + 3,52 \times 10^5 s + 1,59 \times 10^{10}} \; ; \; H_2(s) = \frac{s^2 + 2,22 \times 10^4 s + 6,31 \times 10^7}{s^2 + 1,11 \times 10^4 s + 1,59 \times 10^7} \end{gathered} \tag{5.3}$$

As duas funções de transferência apresentadas $H_1(s)$ e $H_2(s)$ correspondem a filtros biquadráticos do tipo polo-zero, e podem ser implementados na FPAA recorrendo à equação genérica representada em (4.3), desta forma tem-se para o filtro $H_1(s)$:

$$\begin{gathered} 4\pi^2 f_p^2 = 1,59 \times 10^{10} \leftrightarrow f_p = \sqrt{\frac{1,59 \times 10^{10}}{4\pi^2}} = 20,04 \text{ kHz} \\ \frac{2\pi f_p}{Q_p} = 3,52 \times 10^5 \leftrightarrow Q_p = \frac{2\pi \cdot 20,04}{3,52 \times 10^5} = 0,358 \\ 4\pi^2 f_z^2 = 6,31 \times 10^{10} \leftrightarrow f_z = \sqrt{\frac{6,31 \times 10^{10}}{4\pi^2}} = 39,98 \text{ kHz} \\ \frac{2\pi f_z}{Q_z} = 7,02 \times 10^5 \leftrightarrow Q_z = \frac{2\pi \cdot 39,98}{7,02 \times 10^5} = 0,358 \end{gathered} \tag{5.4}$$

Da mesma forma calcula-se os parâmetros do filtro $H_2(s)$:

$$\begin{gathered} f_p = \sqrt{\frac{1,59 \times 10^7}{4\pi^2}} = 0,6336 \text{ kHz} \qquad Q_p = \frac{2\pi \cdot 0,6336}{1,11 \times 10^4} = 0,358 \\ f_z = \sqrt{\frac{6,31 \times 10^7 \times 10^4}{4\pi^2}} = 1,264 \text{ kHz} \qquad Q_z = \frac{2\pi \cdot 1,264}{2,22 \times 10^4} = 0,358 \end{gathered} \tag{5.5}$$

Para a implementação do filtro de quarta ordem, são utilizados dois filtros de segunda ordem. A execução destes num FPAA é direta bastando os parâmetros calculados (Tabela 5.1).

A aplicação pode assim ser desenvolvida no *software* AnadigmDesigner2, usando para tal duas CAM *FilterBiquad* e uma *GainLimiter*. Na Figura 5.1 está representado o circuito desta implementação.

A configuração do filtro é feita diretamente na janela da CAM, preenchendo os campos com os valores da Tabela 5.1, tal como verificado na Figura 5.2.

Tabela 5.1: Parâmetros dos filtros $H_1(s)$ e $H_2(s)$

Filtro Biquadrático $H_1(s)$	Filtro Biquadrático $H_2(s)$
$G = 1$	$G = 1$
$fp = 20,04$	$fp = 0,6336$
$Qp = 0,358$	$Qp = 0,358$
$fz = 39,98$	$fz = 1,264$
$Qz = 0,358$	$Qz = 0,358$

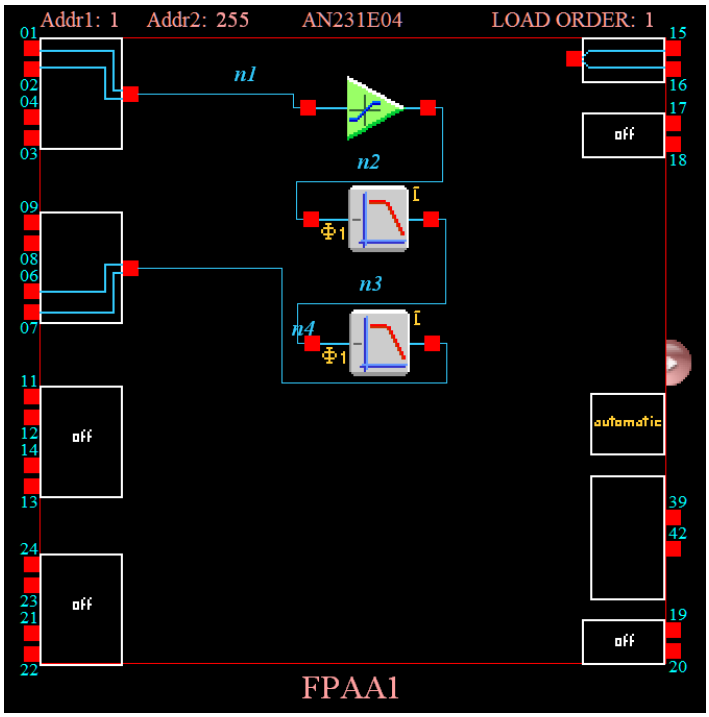
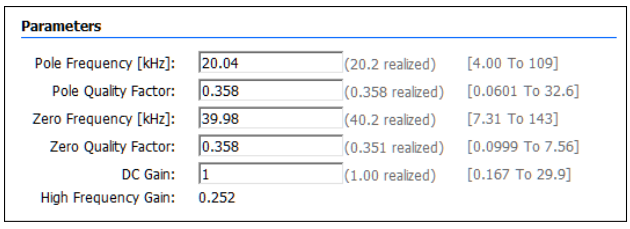
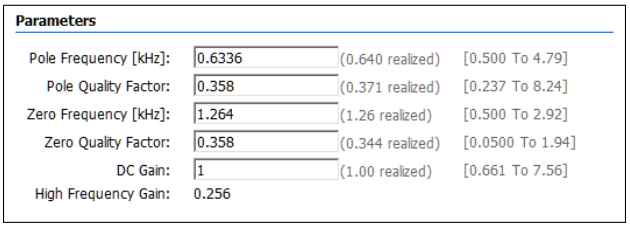


Figura 5.1: Implementação do filtro no AnadigmDesiner2



(a) Filtro H_1



(b) Filtro H_2

Figura 5.2: Configuração dos filtros na FPAA

5.1.1 Recolha de dados em Simulink

Para a recolha e análise dos dados foi criado o diagrama de blocos da Figura 5.3, em que o bloco "Simulink"implementa o controlador, o bloco "Servo"a função de transferência do motor, o bloco "FPAA"a entrada e saída para a comunicação com o *hardware* e o *Driver* converte a tensão do degrau em velocidade para a comparação com a saída do motor.

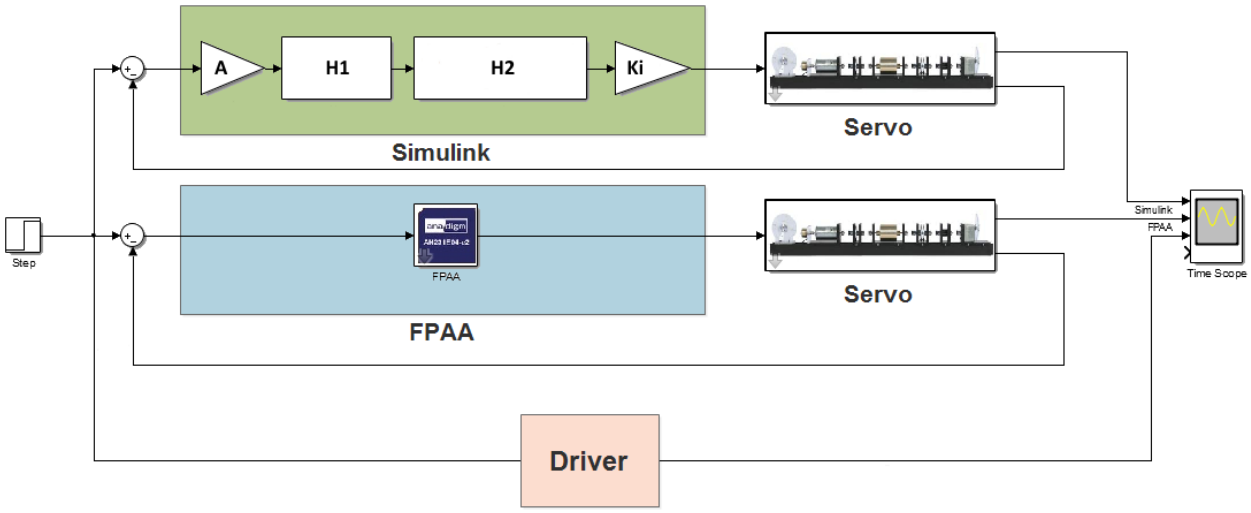


Figura 5.3: Diagrama de blocos em Simulink

Bloco Controlador

Este bloco implementa a função de transferência do controlador, é constituído pela função de transferência da aproximação obtida e, dependendo do controlador, pelos restantes blocos que o compõem. Para o controlador $K_i s^{-0,4}$ esta implementação tem a forma da equação (5.6) e o diagrama representado na Figura 5.4.

$$H(s) = A \times H_1(s) \times H_2(s) \times H_2(s) \times K_i \tag{5.6}$$

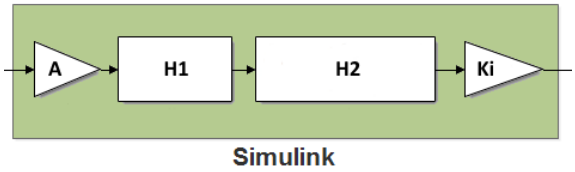


Figura 5.4: Diagrama de blocos do controlador $K_i s^{-0,4}$

BLOCO SERVO

O bloco servo implementa a função de transferência de primeira ordem do motor obtida pelas regras de Ziegler-Nichols, tal como apresentada na equação (4.16). A Figura 5.5 apresenta o aspeto deste bloco.

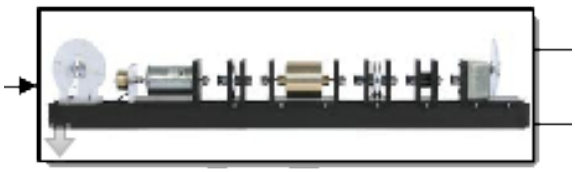


Figura 5.5: Bloco servo

O diagrama de blocos é constituído pela função de transferência $\frac{K}{Ts+1}$ multiplicado pelo atraso e^{-Ls}. A Figura 5.6 apresenta o diagrama de blocos correspondente.

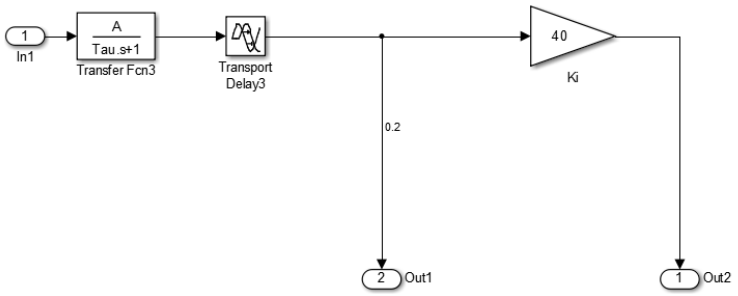


Figura 5.6: Diagrama de blocos do servo

Na Figura 5.7 estão apresentados os valores que configuram o servo, obtidos em (4.15).

BLOCO FPAA

O bloco FPAA trata da comunicação com o Hilink, é constituído pelos blocos de saída B_0, pelas entradas A_0 e A_1, uma saturação, um valor de *offset* e dois blocos somadores. A Figura 5.8 apresenta o aspeto deste bloco.

A saída B_0 envia os sinais gerados no Simulink para a placa de aquisição Hilink, que por sua vez envia para a FPAA (placa de desenvolvimento AN231K04). Nesta está carregada em memória a função de transferência do controlador por onde passa o sinal e é devolvido em modo diferencial para a placa Hilink, que por sua vez envia para o Simulink através das entradas A_0 e A_1. O Bloco

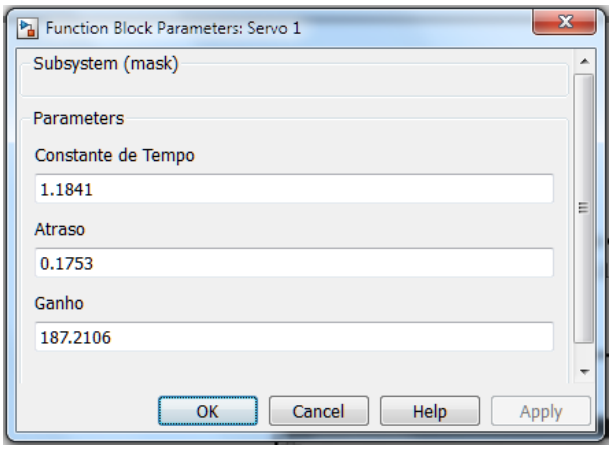


Figura 5.7: Valores dos parâmetros do servo

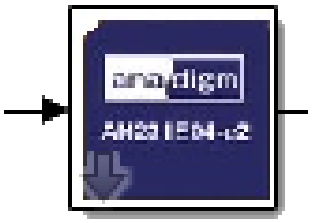


Figura 5.8: Bloco FPAA

saturação limita a saída da FPAA entre 0 e 5 V. Foi notada em experiências realizadas uma tensão residual na saída da FPAA, que para o derivador não tinha qualquer interferência mas para o integrador era o suficiente para alterar a resposta. Para eliminar este efeito foi colocado um bloco de *offset* que elimina esta tensão residual. O seu valor foi verificado medindo a saída da FPAA quando na entrada está presente um sinal de valor zero. Na Figura 5.9 está representado o diagrama deste bloco.

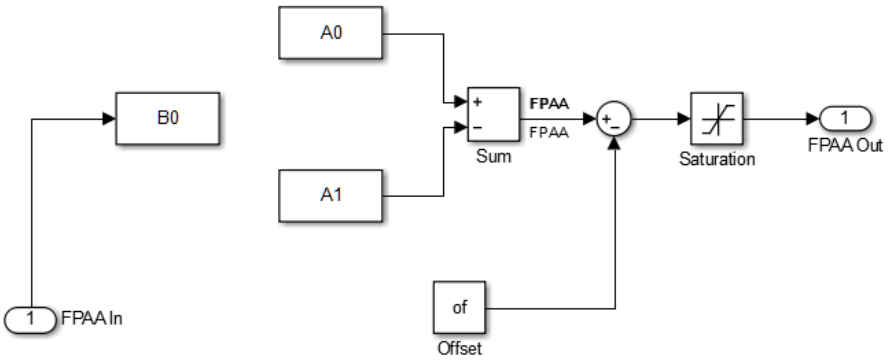


Figura 5.9: Diagrama de blocos da FPAA

Na Figura 5.10 apresenta-se a imagem desta arquitetura.

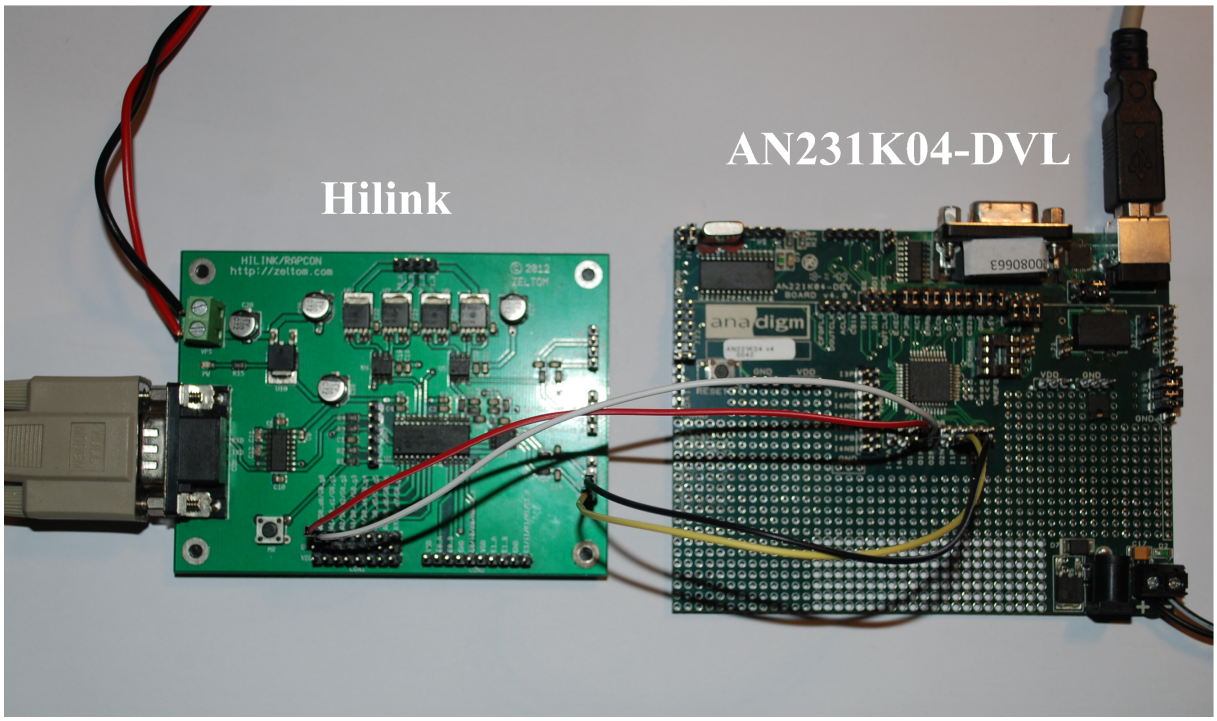


Figura 5.10: *Hardware* Hilink e AN231K04

5.2 DERIVADOR FRACIONÁRIO - D^{μ}

Para a implementação de um derivador fracionário usa-se a aproximação do operador s^{μ} descrita no capítulo 3. Considerando a função $s^{0,4}$ e aplicando a aproximação $N = 4$, $wb = 1000$ rad/s e $wh = 1 \times 10^6$ rad/s obtém-se a seguinte aproximação:

$$H(s) = \frac{251,2s^4 + 9,112 \times 10^7 s^3 + 4,967 \times 10^{12} s^2 + 4,567 \times 10^{16} s + 6,31 \times 10^{19}}{s^4 + 7,238 \times 10^5 s^3 + 7,872 \times 10^{10} s^2 + 1,444 \times 10^{15} s + 3,981 \times 10^{18}} \tag{5.7}$$

Dividindo em duas funções de segunda ordem, obtém-se:

$$\begin{gathered} A = 251,2 \\ H_1(s) = \frac{s^2 + 3,516 \times 10^5 s + 1,585 \times 10^{10}}{s^2 + 7,016 \times 10^5 s + 6,31 \times 10^{10}};\ H_2(s) = \frac{s^2 + 1,112 \times 10^4 s + 1,585 \times 10^7}{s^2 + 2,219 \times 10^4 s + 6,31 \times 10^7} \end{gathered} \tag{5.8}$$

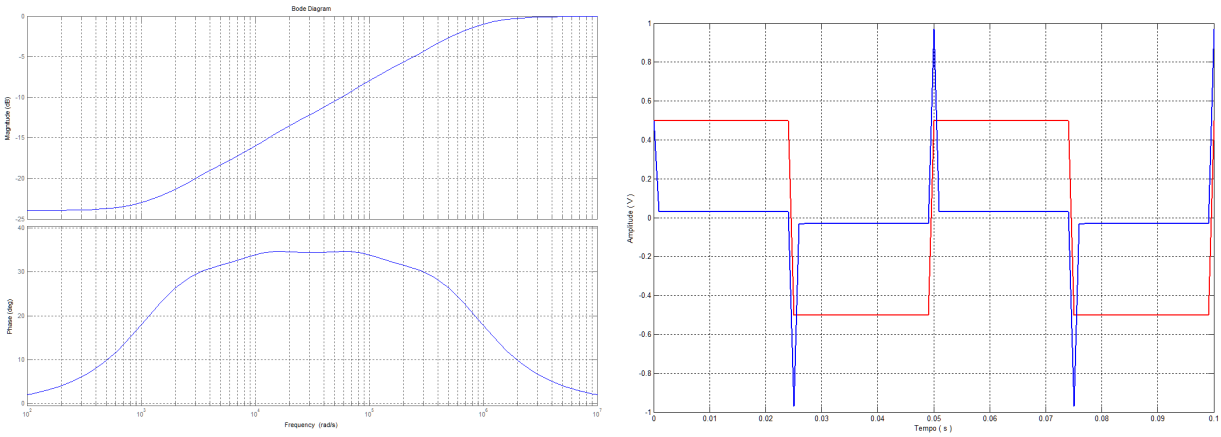
Por sua vez o filtro reconstruido é igual a:

$$H(s) = A \times H_1(s) \times H_2(s) \tag{5.9}$$

Na Tabela 5.2 são apresentados os parâmetros a inserir nos filtros da FPAA, na configuração representada na Figura 5.1. Na Figura 5.11(a) apresenta-se o diagrama de fase e magnitude desta aproximação, já na Figura 5.11(b) está representada a resposta à onda quadrada.

Tabela 5.2: Parâmetros dos filtros de $s^{0,4}$

Filtro Biquadrático 1	Filtro Biquadrático 2
$G = 1$	$G = 1$
$fp = 39,98$	$fp = 1,264$
$Qp = 0,358$	$Qp = 0,358$
$fz = 20,04$	$fz = 0,6336$
$Qz = 0,358$	$Qz = 0,358$



(a) Magnitude (b) Resposta á onda quadrada

Figura 5.11: Respostas do controlador D^{μ}

Para comparação das respostas entre Simulink e FPAA foi implementado o circuito da Figura 5.12.

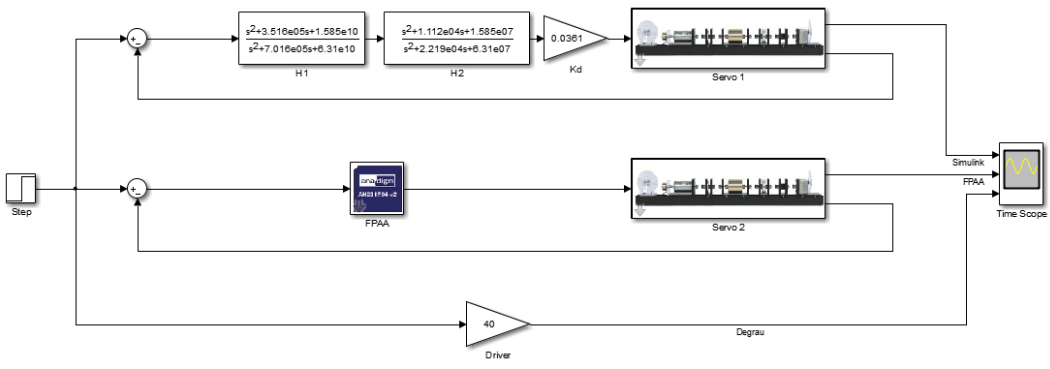


Figura 5.12: Diagrama Simulink dos controladores FPAA e Simulink

Na Figura 5.13 está representada a resposta em Simulink do controlador $s^{0,4}$, já na Figura 5.14 está representada a resposta do mesmo controlador implementado na FPAA.

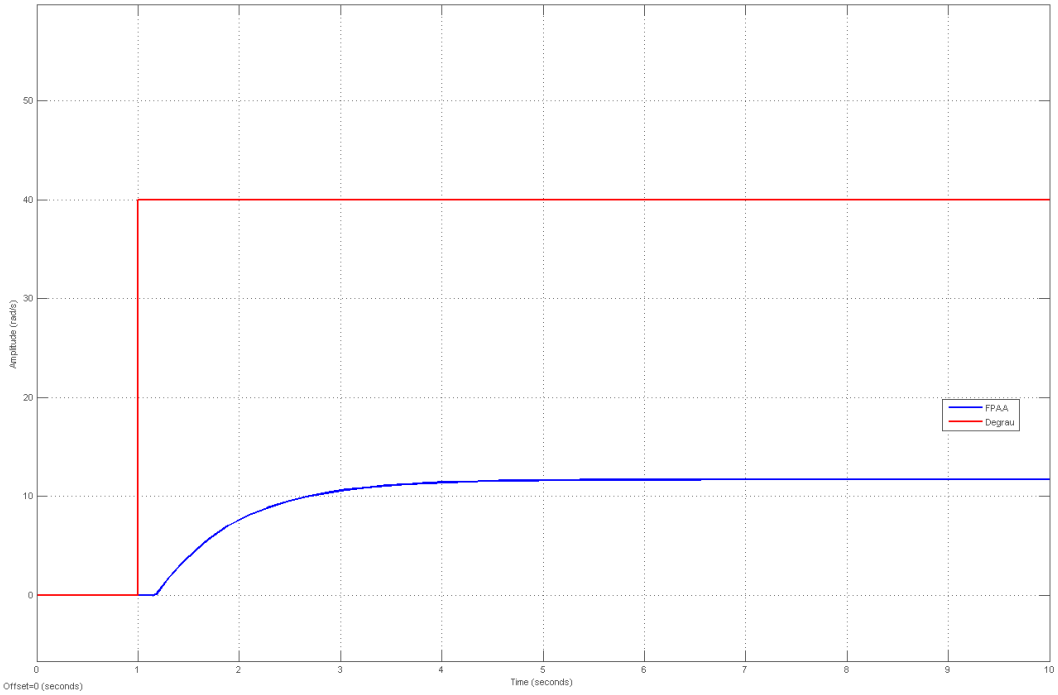


Figura 5.13: Resposta ao degrau do controlador implementado em Simulink

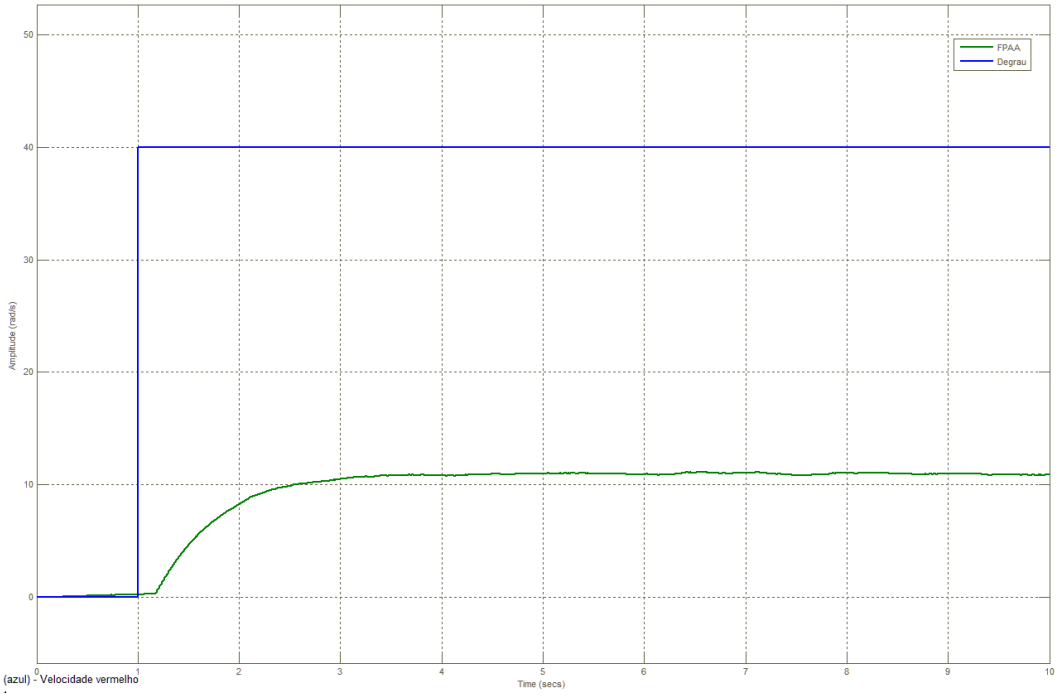


Figura 5.14: Resposta ao degrau do controlador implementado na FPAA

Como se pode verificar as figuras são muito semelhantes existindo só uma pequena diferença devido ao ruído médio introduzido pela FPAA e anulado pela tensão de *offset*.

A função de transferência do controlador fracionário D^{μ} é dada pela seguinte expressão:

$$G_c = K_d s^{\mu}, \qquad \mu > 0 \tag{5.10}$$

Este controlador foi projetado adotando para o ganho derivativo K_d o valor do ganho K_p obtido pelas regras de Ziegler e Nichols, $K_d = 1/RL = 0.0361$. Desta forma foram experimentados três derivadores fracionários de ordem $\mu = \{0,2;0,4;0,6\}$. A resposta de cada um dos controladores pode ser verificada na Figura 5.15.

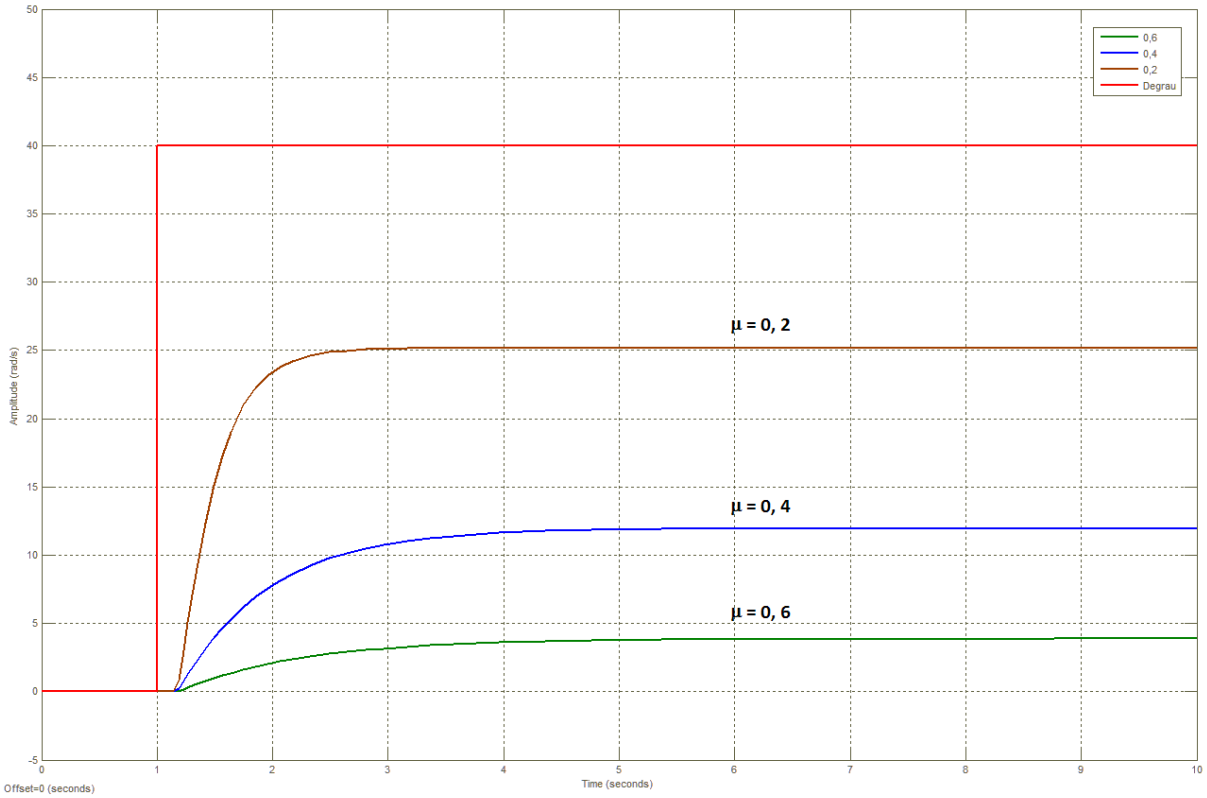


Figura 5.15: Resposta ao degrau dos controladores $\mu = \{0,2;0,4;0,6\}$

A Figura 5.15 mostra que o erro em regime permanente diminui quando a ordem de μ diminui. Aumentando o ganho K_d o erro em regime permanente diminui mas aumenta a instabilidade do sistema. Nas Figuras 5.16 e 5.17 estão representadas as respostas dos controladores implementados em Simulink e na FPAA quando se multiplica por 20 o valor de K_d, sendo este igual a $0,7$. As respostas são muito semelhantes, embora a resposta da FPAA estabilize mais rapidamente.

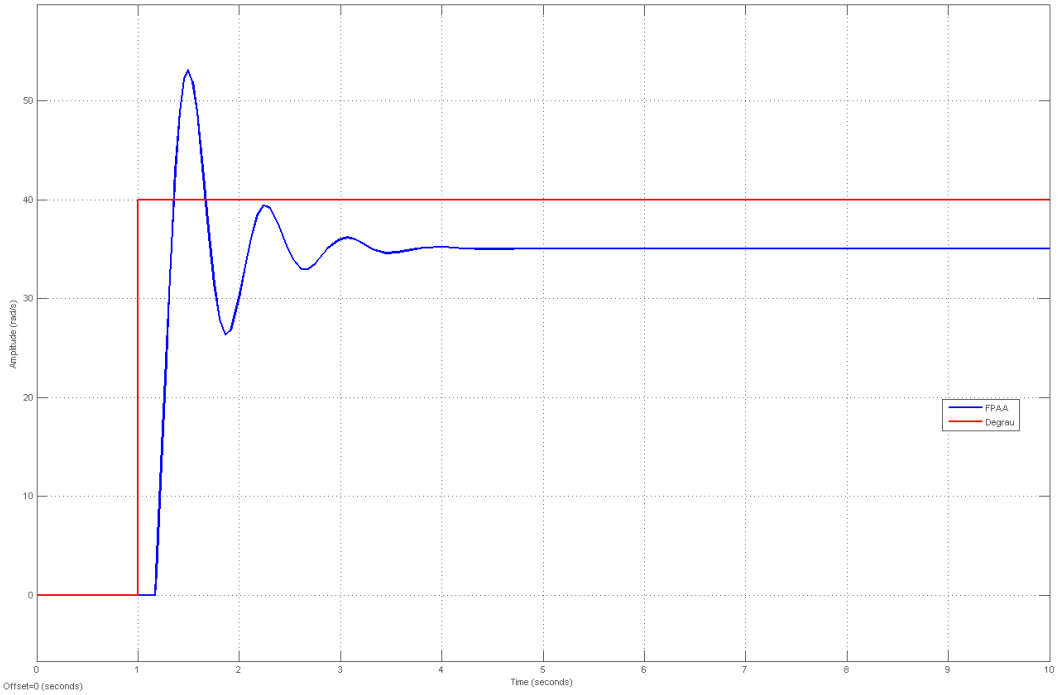


Figura 5.16: Resposta em Simulink com $K_d = 0,7$

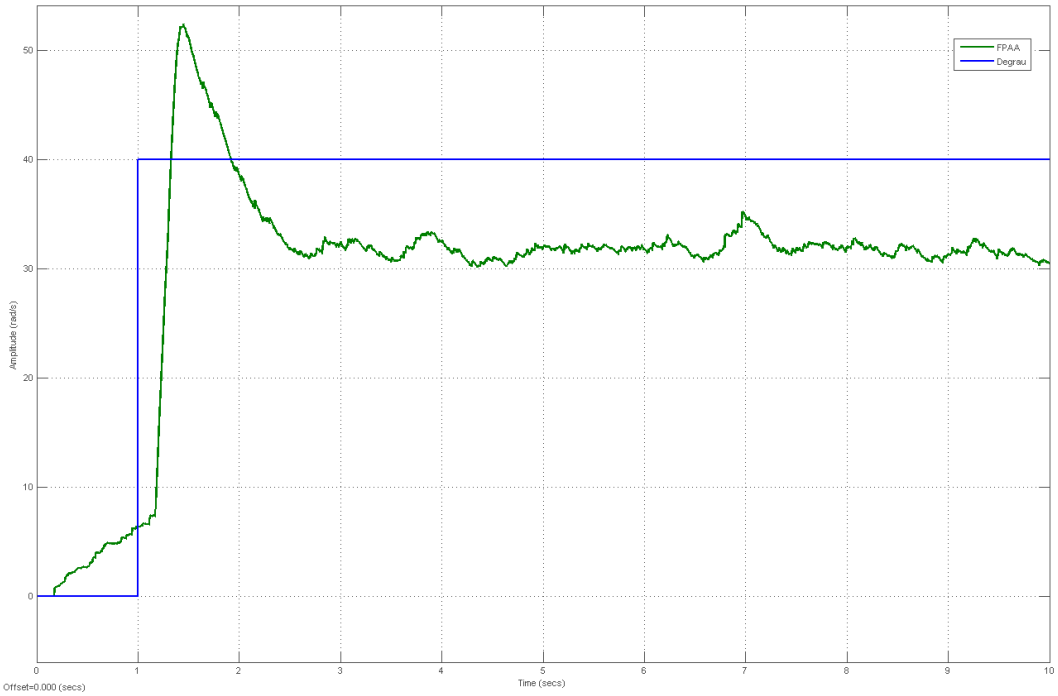
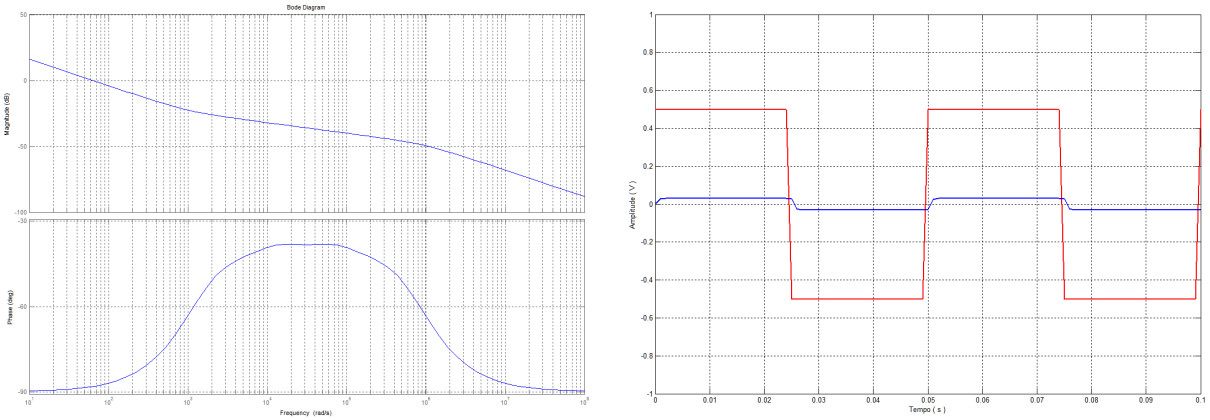


Figura 5.17: Resposta da FPAA com $K_d = 0,7$

5.3 INTEGRADOR FRACIONÁRIO - I^{λ}

Para a implementação de um integrador fracionário usa-se a aproximação do operador $s^{-\lambda}$ descrita no capítulo 3. Considerando a função $s^{-0,4}$ e aplicando a aproximação com $N = 4$, $wb = 1000$ rad/s e $wh = 1 \times 10^6$ rad/s obtém-se a aproximação da equação 5.1. Os filtros H1(s) e H2(s) estão apresentados na equação 5.3. A Tabela 5.1 mostra os parâmetros a inserir nos filtros da FPAA, na configuração representada na Figura 5.1. Na Figura 5.18(a) apresenta-se o diagrama de fase e magnitude desta aproximação, já na Figura 5.18(b) está representada a resposta à onda quadrada.



(a) Magnitude (b) Resposta à onda quadrada

Figura 5.18: Respostas do controlador I^{λ}

A implementação gera um erro em regime permanente que pode tornar-se elevado. De forma a obter-se um erro em regime permanente aceitável ou nulo o termo $1/s^{\lambda}$ deve ser implementado usando um integrador inteiro, assim o controlador I^{λ} modificado fica da seguinte forma:

$$G_c = K_i \frac{s^{1-\lambda}}{s} = K_i \frac{1}{s} s^{1-\lambda}, \qquad \lambda > 0 \tag{5.11}$$

Desta forma o controlador modificado implementado na FPAA tem a seguinte forma:

$$G_c = K_i \frac{1}{s} \times H(s) = K_i \frac{1}{s} \times A \times H_1(s) \times H_2(s) \qquad \lambda > 0 \tag{5.12}$$

Para simplificar a implementação os ganhos A e K_i, são representados por um só bloco amplificador de tal forma que $K = A \times K_i$. Na Figura 5.19 está representada a implementação na FPAA.

Para esta implementação há necessidade então de calcular uma nova aproximação $s^{1-\lambda} = s^{0,6}$. Na Tabela 5.3 estão apresentados os parâmetros destes filtros.

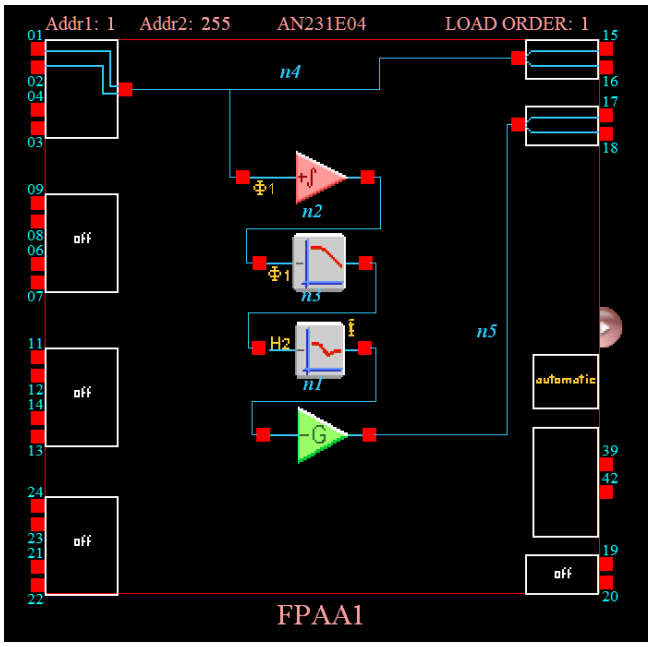


Figura 5.19: Diagrama do controlador no AnadigmDesigner

Tabela 5.3: Parâmetros dos filtros $s^{0,6}$

Filtro Biquadrático 1	Filtro Biquadrático 2
$G = 1$	$G = 1$
$fp = 47,51$	$fp = 1,503$
$Qp = 0,358$	$Qp = 0,358$
$fz = 16,86$	$fz = 0,5339$
$Qz = 0,358$	$Qz = 0,358$

Para comparação das respostas entre Simulink e FPAA foi implementado o circuito da Figura 5.20.

A resposta do controlador $K_i s^{-0,4}$ implementado na FPAA comparativamente ao controlador implementado no Simulink está apresentado nas Figuras 5.21 e 5.22.

Como se pode observar as respostas são muito semelhantes, embora o controlador implementado na FPAA tenha um atraso muito superior ao controlador implementado no Simulink. Uma das causas deste atraso, é devido á comunicação bidirecional entre o computador e a placa de aquisição de sinal(Hilink).

Este controlador foi projetado adotando para o ganho derivativo K_i o valor do ganho K_p obtido pelas regras de Ziegler e Nichols, $K_i = 1/RL = 0,0361$. Desta forma foram experimentados

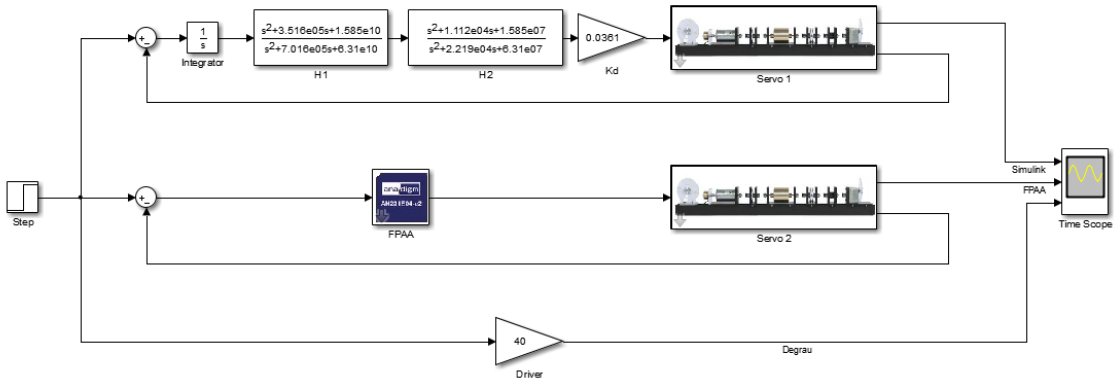


Figura 5.20: Diagrama Simulink dos controladores FPAA e Simulink

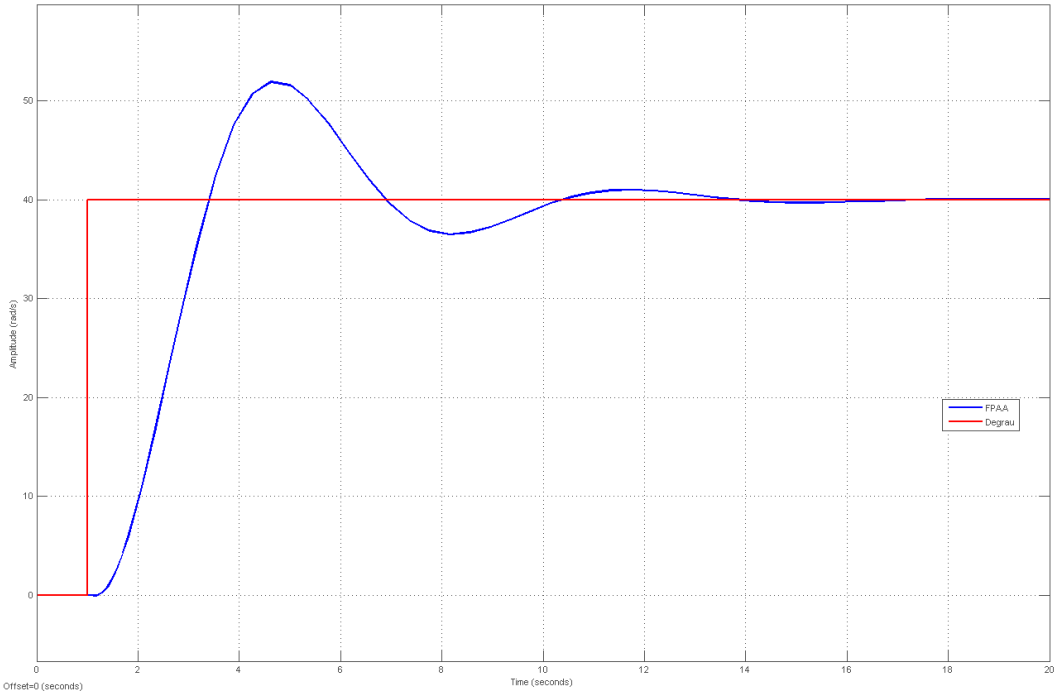


Figura 5.21: Resposta ao degrau do controlador implementado em Simulink

três integradores fracionários de ordem $\lambda = \{0,2;0,4;0,6\}$. A resposta de cada um dos controladores pode ser verificada na Figura 5.23 A Figura 5.23 mostra que o erro em regime permanente é praticamente nulo. Verificou-se também que a ordem não inteira de λ é muito útil no ajuste da dinâmica do sistema. Aumentando o ganho K_i o erro em regime permanente diminui mas aumenta a instabilidade do sistema. Nas Figuras 5.24 e 5.25 estão representadas as respostas dos controladores implementados em Simulink e na FPAA quando se multiplica por 2 o valor de K_i, sendo este igual a $0,7$. O ruído em alguns tipos de controladores pode ter impacto na dinâmica do sistema, como é o caso do controlador I^{λ} em que este impacto interfere no atraso, como se pode observar nas Figuras 5.22 e 5.25: A anulação do ruído através da tensão de *offset* é um fator a ter em conta.

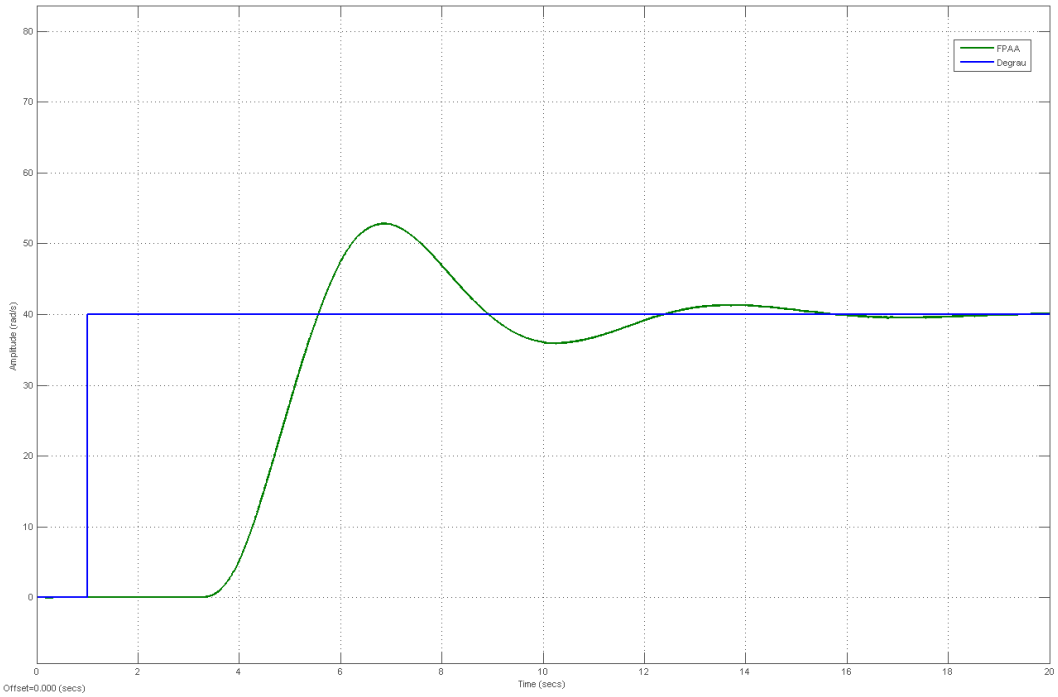


Figura 5.22: Resposta ao degrau do controlador implementado na FPAA

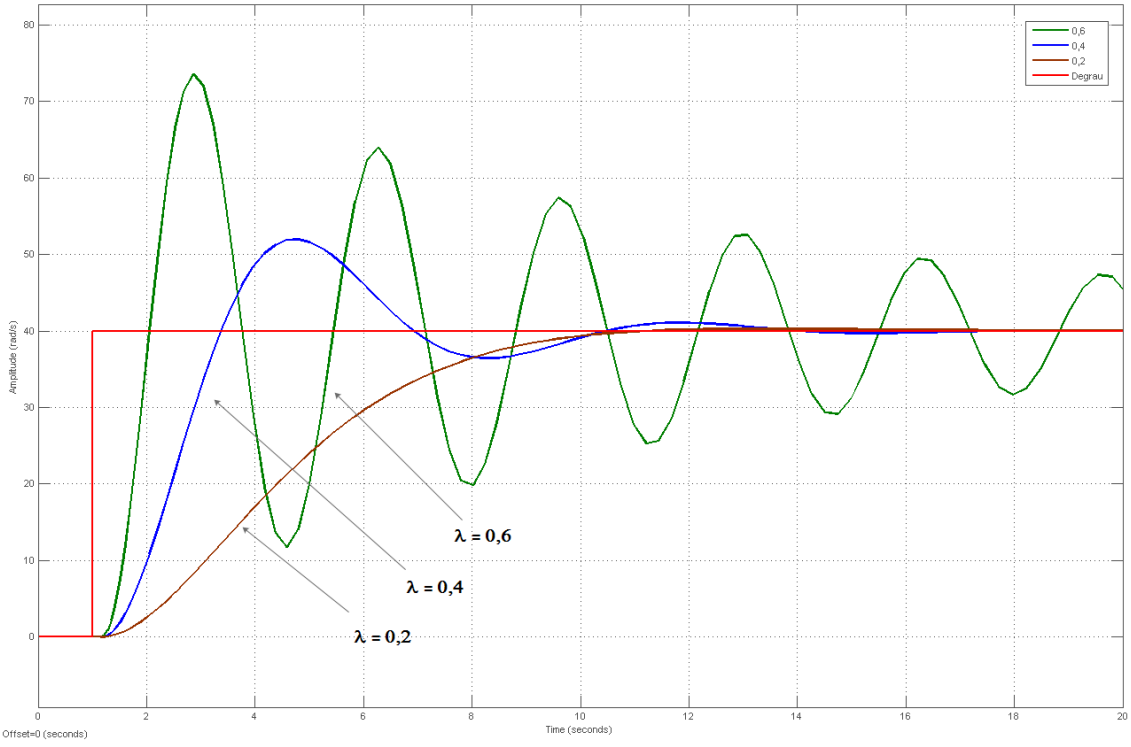


Figura 5.23: Resposta ao degrau dos controladores $\lambda = \{0,2; 0,4; 0,6\}$

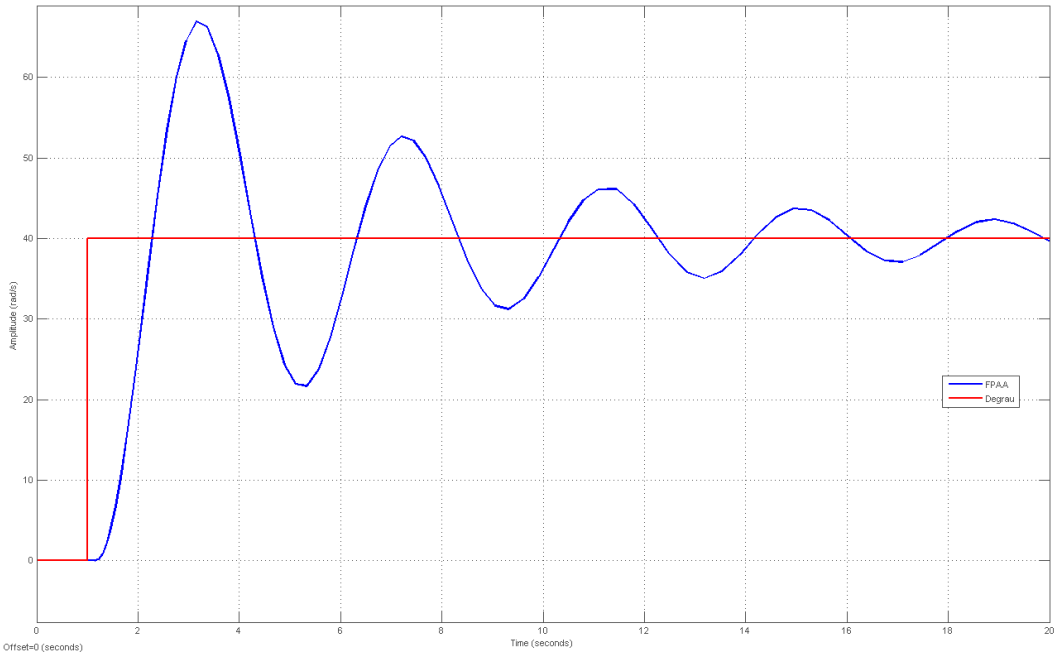


Figura 5.24: Resposta em Simulink com $K_i = 0,7$

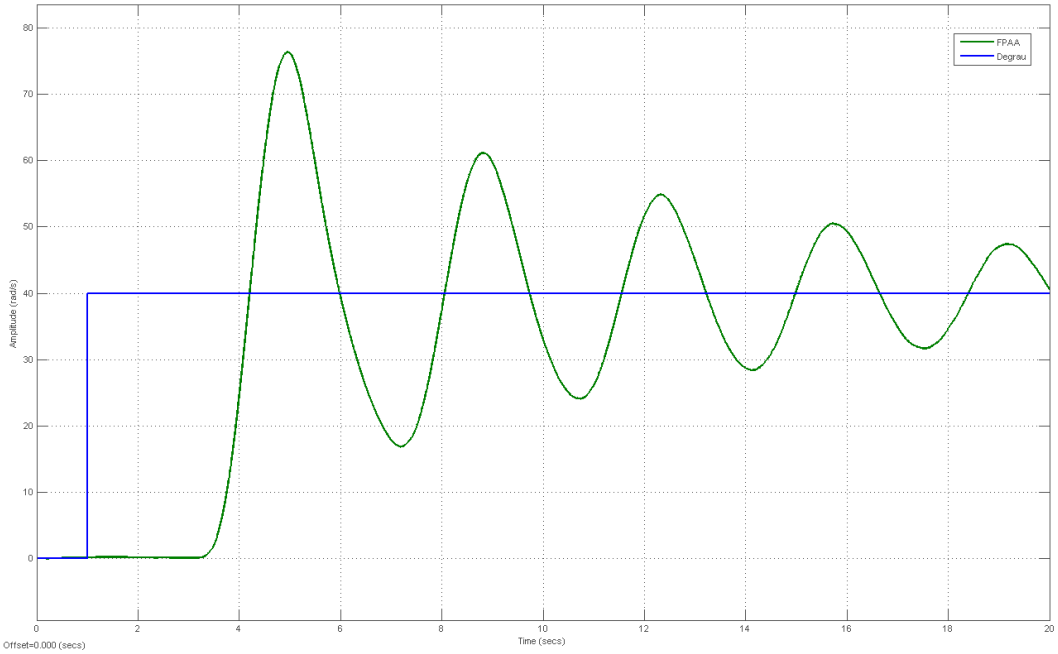


Figura 5.25: Resposta da FPAA com $K_i = 0,7$

5.4 CONTROLADOR PI^{λ}

A função de transferência do controlador PI^{λ} é dado pela seguinte expressão:

$$G_c(s) = K_p + \frac{K_i}{s^{\lambda}} \tag{5.13}$$

Onde o ganho proporcional K_p, ganho integrativo K_i e a ordem do controlador λ são os parâmetros a ser ajustados. De notar ainda que o termo s^{λ} é implementado de acordo com (5.11). Os parâmetros do controlador fracionário adotados foram obtidos do ajuste efetuado segundo as regras Ziegler e Nichols para o controlador PI, $K_p = 0,9/RL = 0,0325$ e $K_i = 0,3K_p/L = 0,0556$.

Devido à complexidade do controlador exceder os recursos da FPAA, foi reduzida a ordem da aproximação, assim considerando a função $s^{-0.4}$ e aplicando a aproximação considerando $N = 3$, $wb = 1000$ rad/s e $wh = 1 \times 10^6$ rad/s obtém-se a seguinte aproximação:

$$H(s) = \frac{3981s^3 + 7,004 \times 10^8 s^2 + 1,11 \times 10^{13} s + 1,585 \times 10^{16}}{s^3 + 7,004 \times 10^5 s^2 + 4,419 \times 10^{10} s + 2,512 \times 10^{14}} \tag{5.14}$$

Dividindo numa função de segunda ordem e outra de primeira ordem, obtém-se:

$$A = 3981; \qquad H_1(s) = \frac{s^2 + 1,743 \times 10^4 s + 2,512 \times 10^7}{s^2 + 6,941 \times 10^4 s + 3,981 \times 10^8}; \qquad H_2(s) = \frac{s + 1,585 \times 10^5}{s + 6,31 \times 10^5} \tag{5.15}$$

Por sua vez o filtro reconstruido é igual a:

$$H(s) = A \times H_1(s) \times H_2(s) \tag{5.16}$$

Tabela 5.4: Parâmetros dos filtros de $I^{0,4}$

Filtro Bilinear	Filtro Biquadrático
$G = 1$	$G = 1$
$fp = 100,4$	$fp = 3,176$
$fz = 25,22$	$Qp = 0,2875$
	$fz = 0,7977$
	$Qz = 0,2875$

Na Tabela 5.4 são apresentados os parâmetros a inserir nos filtros da FPAA.

A aplicação pode assim ser desenvolvida no *software* AnadigmDesigner2, usando para tal uma CAM *FilterBiquad*, uma *FilterBilinear*, duas *Gaininv* e uma *Suminv*. Na Figura 5.26 está representado o circuito desta implementação.

A resposta do controlador $K_p + K_i s^{-0,4}$ implementado na FPAA comparativamente ao controlador implementado no Simulink está apresentado nas Figuras 5.27 e 5.28.

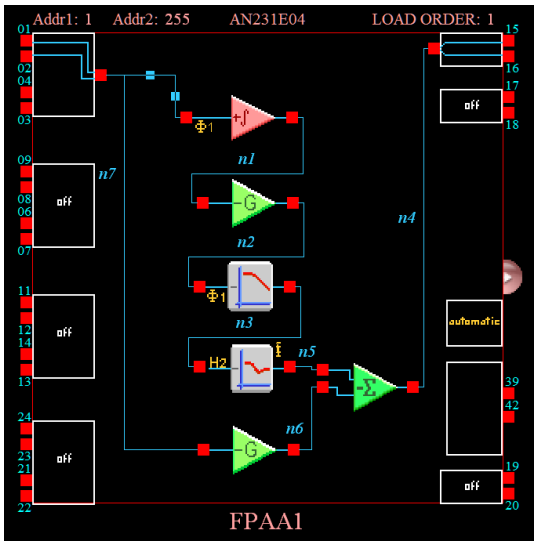


Figura 5.26: Implementação do filtro no AnadigmDesigner2

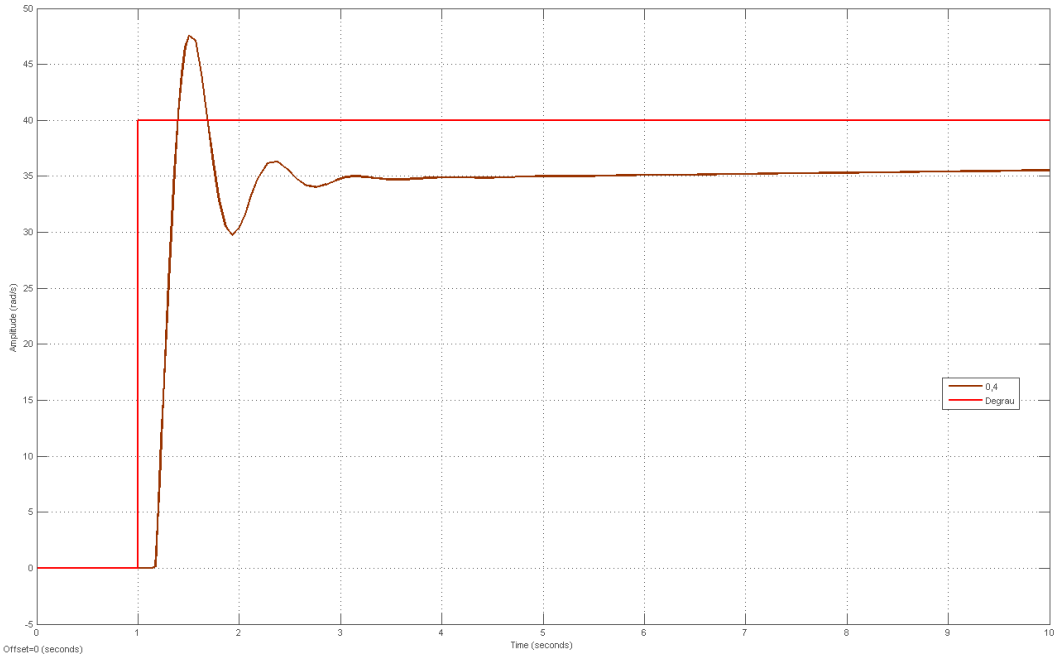


Figura 5.27: Resposta ao degrau do controlador implementado em Simulink

Da análise das Figuras 5.27 e 5.28 verifica-se que tal como no controlador I^{λ}, a introdução do controlador K_p no sistema tornou-o mais rápido a atingir o *stepoint*, mas aumentou o erro em regime permanente.

Na Figura 5.29 pode-se observar a resposta dos três fracionários implementados de ordem $\lambda = \{0,2; 0,4; 0,6\}$. Como se pode observar na Figura 5.29 a variação de λ tem pouco impacto no tempo de subida e no *overshoot*, mas afeta o erro em regime permanente.

A Figura 5.30 apresenta a resposta do controlador aumentando K_i para dez vezes o seu valor

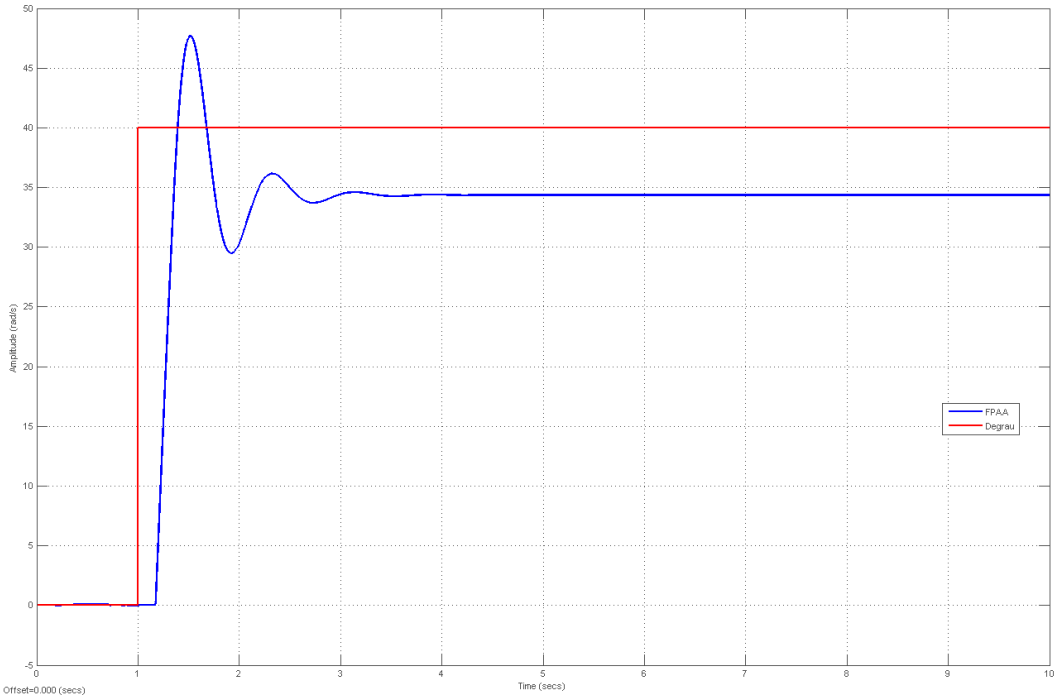


Figura 5.28: Resposta ao degrau do controlador implementado na FPAA

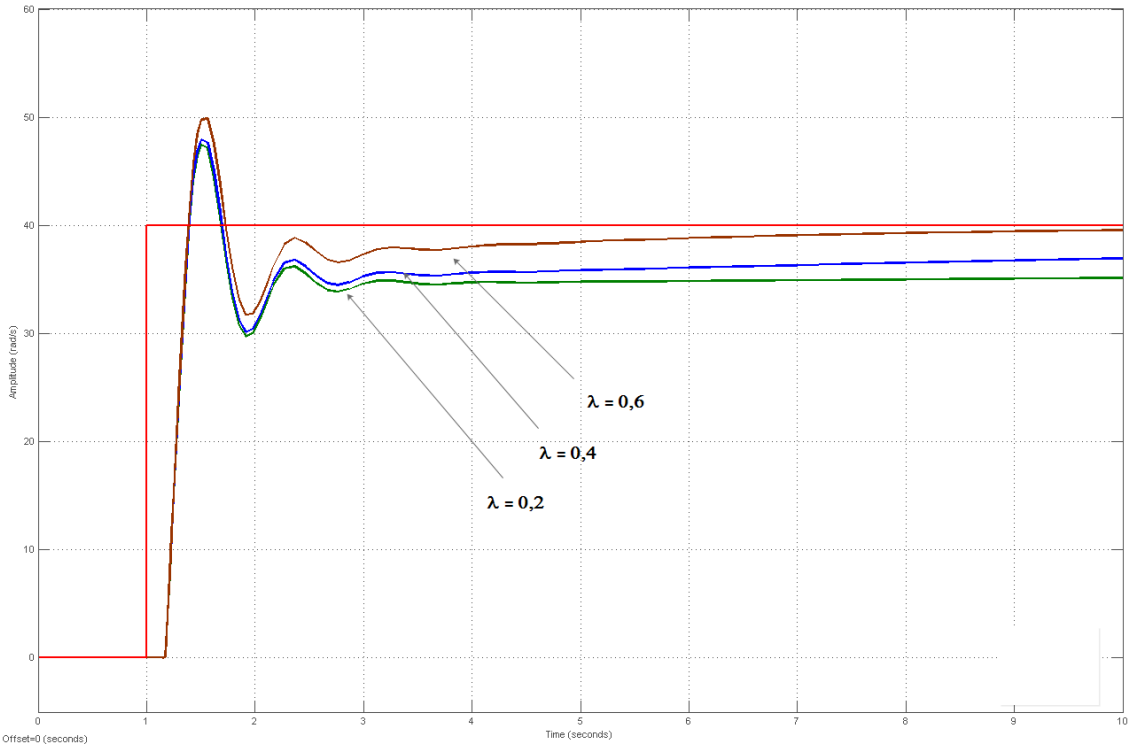


Figura 5.29: Resposta ao degrau dos controladores $\lambda = \{0,2; 0,4; 0,6\}$

($K_i = 10 \times 0,0556 = 0,556$), nota-se que o impacto é pouco significativo. Aumentando K_p para duas vezes o seu vaor ($K_p = 2 \times 0,0325 = 0,065$), aumenta a instabilidade do sistema, como pode ser observado na Figura 5.31.

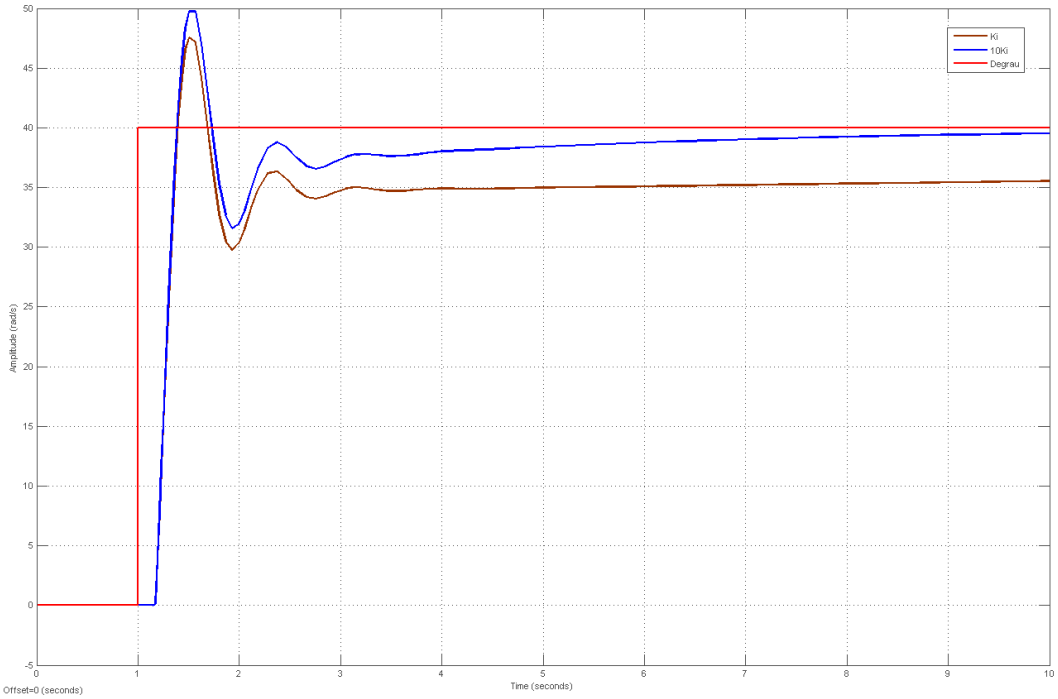


Figura 5.30: Resposta do controlador com $10K_i$

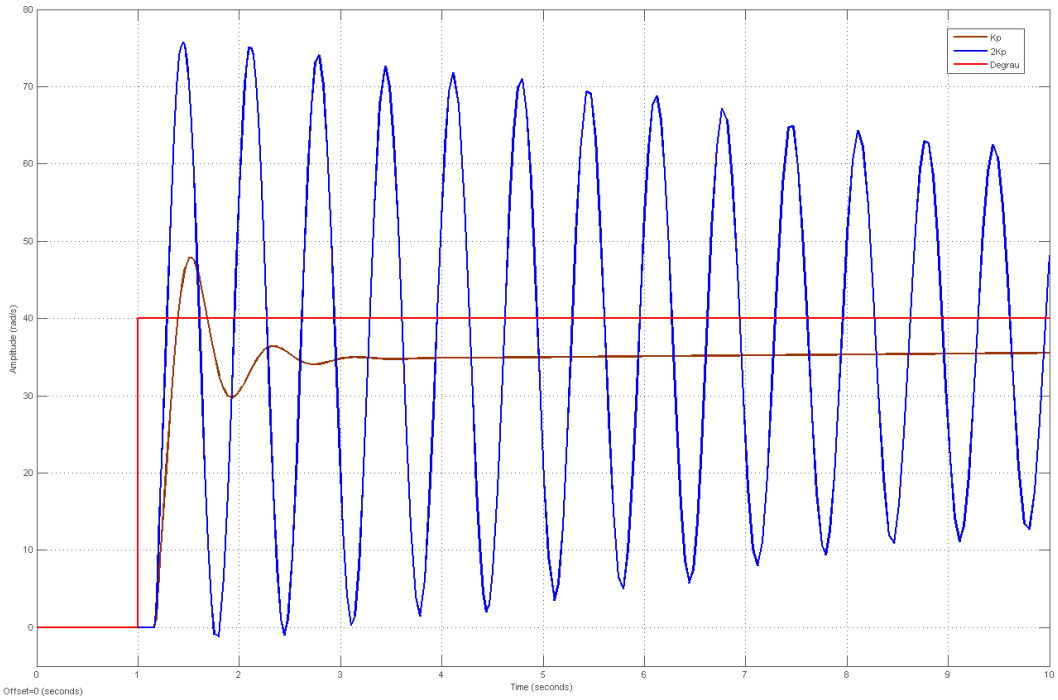


Figura 5.31: Resposta do controlador com $2K_p$

5.5 CONTROLADOR $PI^{\lambda}D$

A função de transferência do controlador $PI^{\lambda}D$ é dado pela seguinte expressão:

$$G_c(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d s, \qquad \lambda > 0 \tag{5.17}$$

Onde o ganho proporcional K_p, ganho integrativo K_i, ganho derivativo K_d e a ordem do integrador λ são os parâmetros a ser ajustados. De notar ainda que o termo $s^{-\lambda}$ é implementado de acordo com (5.11). Os parâmetros do controlador fracionário adotados foram obtidos do ajuste efetuado segundo as regras Ziegler e Nichols para o controlador PID, $K_p = 1,2/RL = 0,0433$, $K_i = K_p/2L = 0,1235$ e $K_d = 0,5LK_p = 0,0038$.

A implementação deste controlador não é possível utilizando somente uma FPAA, sendo necessárias duas FPAA. Desta forma apresenta-se como proposta de implementação o circuito da Figura 5.32.

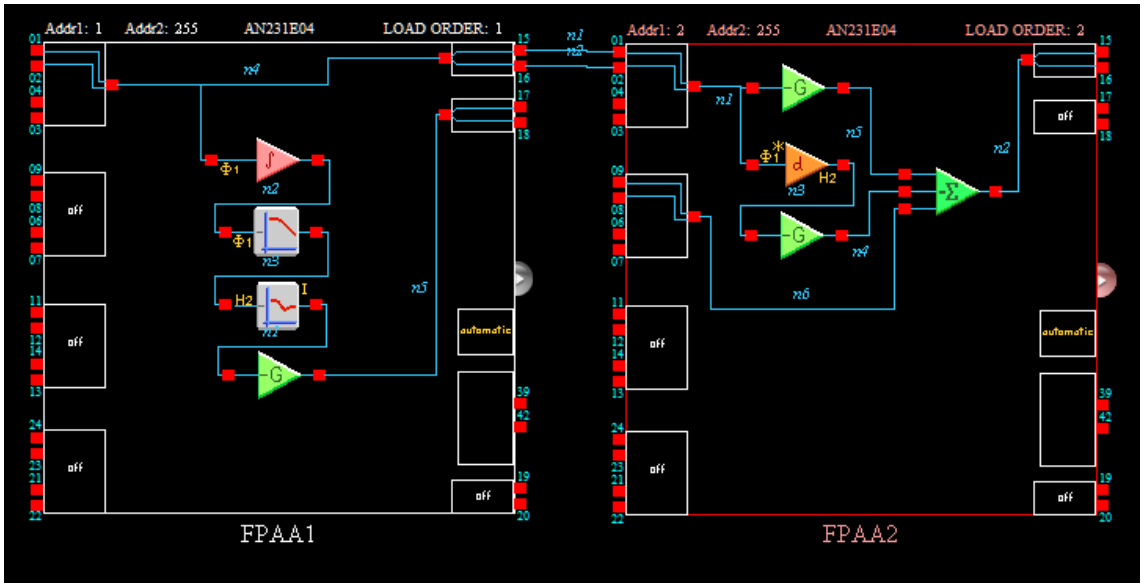


Figura 5.32: Implementação do filtro no AnadigmDesigner2 com dois FPAA

A aplicação pode assim ser desenvolvida no *software* AnadigmDesigner2, usando para tal uma CAM *FilterBiquad*, uma *FilterBilinear* e duas *Gaininv* para a realização do integrador fracionário (I^{α}), uma CAM *differentiator* e um *Gaininv* para a realização do derivador inteiro (D), uma CAM para o ganho proporcional K_P e uma *Suminv* para realizar a soma dos três componentes.

A função de transferência implementada é a mesma do controlador PI^{λ} representada na equação (5.14) os filtros e ganho na equação (5.15). Na Tabela 5.4 são apresentados os parâmetros a inserir nos filtros bilinear e biquadrático da FPAA, na configuração representada

na Figura 5.1.

Para análise do comportamento do sistema foram experimentados três controladores fracionários de ordem $\lambda = \{0,2;0,4;0,6\}$ e $\mu = 1$. A resposta de cada um dos controladores pode ser verificada na Figura 5.33.

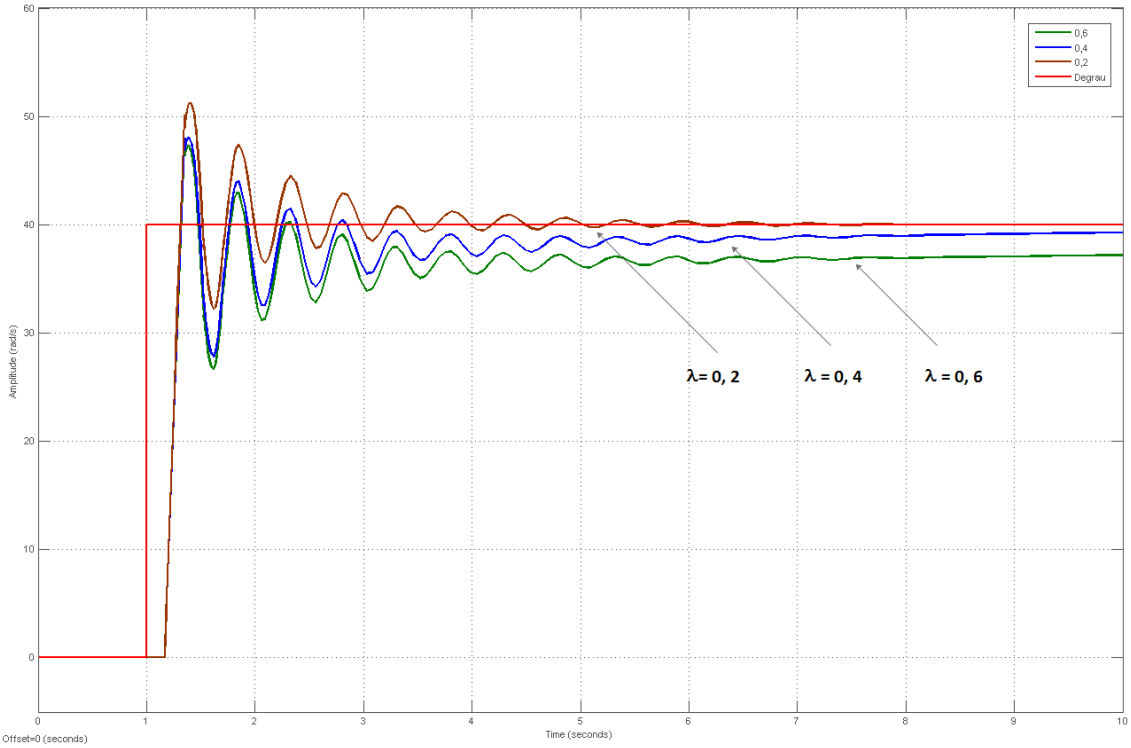


Figura 5.33: Resposta ao degrau dos controladores $\lambda = \{0,2;0,4;0,6\}$

Como se pode verificar pela análise da Figura 5.33, a variação da ordem do integrador faz variar o erro em regime permanente, de tal forma que para $\lambda = 0,2$ o erro é praticamente nulo. A variação dos ganhos K_p, K_i, K_d torna o sistema mais instável, tal como se pode observar nas Figuras 5.34, 5.35 e 5.36.

5.6 CONTROLADOR $PI^{\lambda}D^{\mu}$

A função de transferência do controlador $PI^{\lambda}D^{\mu}$ é dado pela seguinte expressão:

$$G_c(s) = K_p + \frac{K_i}{s^{\lambda}} + K_d s^{\mu} \tag{5.18}$$

Onde o ganho proporcional K_p, ganho integrativo K_i, ganho derivativo K_d, a ordem do integrador λ e a ordem do derivador μ são os parâmetros a ser ajustados. De notar ainda que o termo Ki/s^{λ} é implementado de acordo a equação (5.11). Os parâmetros do controlador

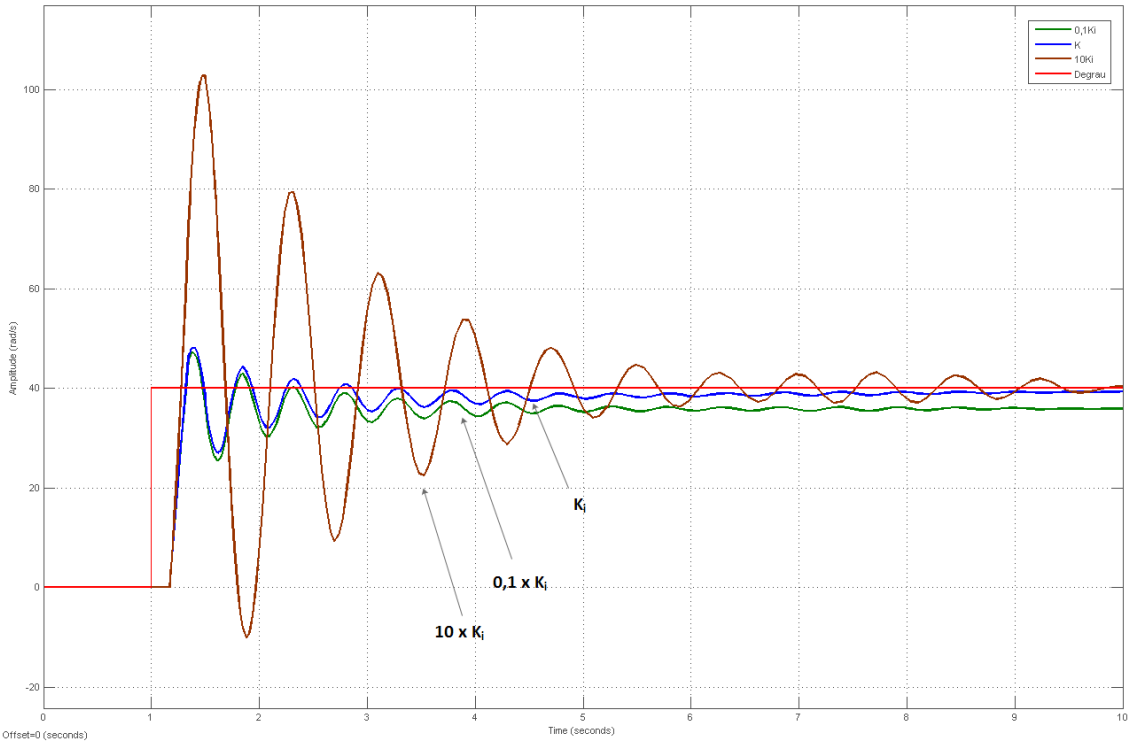


Figura 5.34: Resposta do controlador com $0,1K_i; K_i; 10K_i$

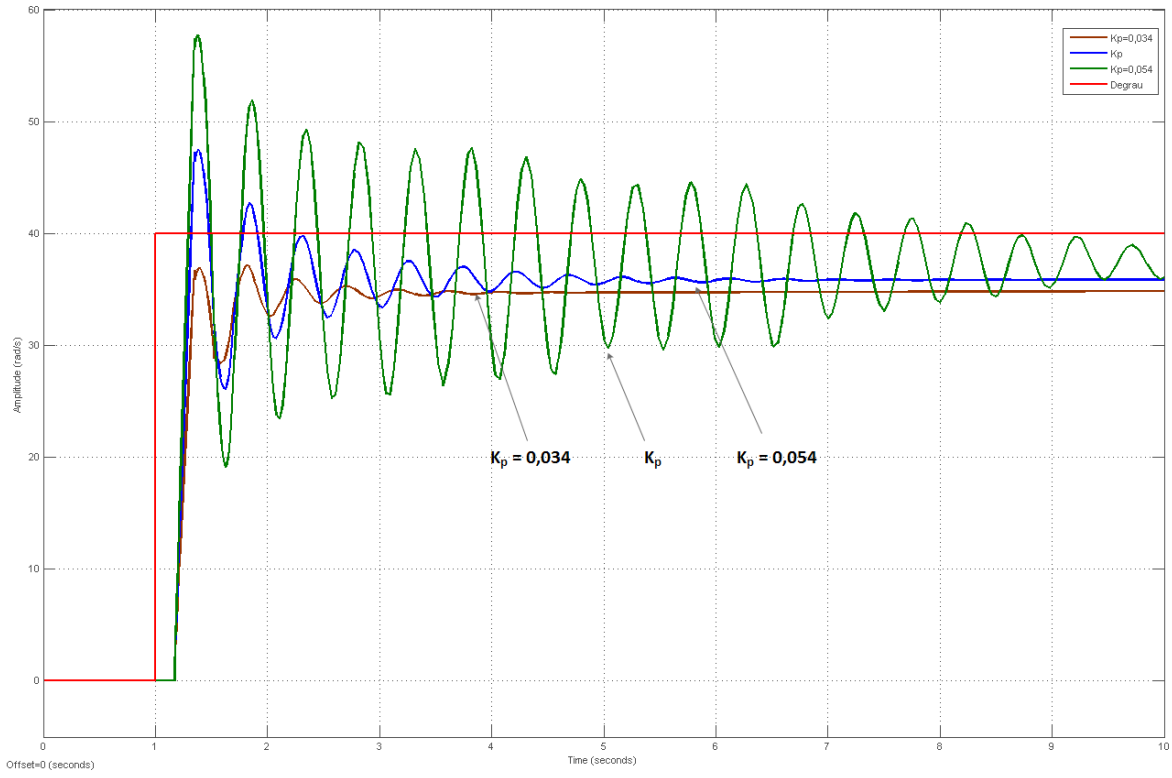


Figura 5.35: Resposta do controlador com $K_p = 0,034; Kp; K_p = 0,54$

fracionário adotados foram obtidos do ajuste efetuado segundo as regras Ziegler e Nichols para

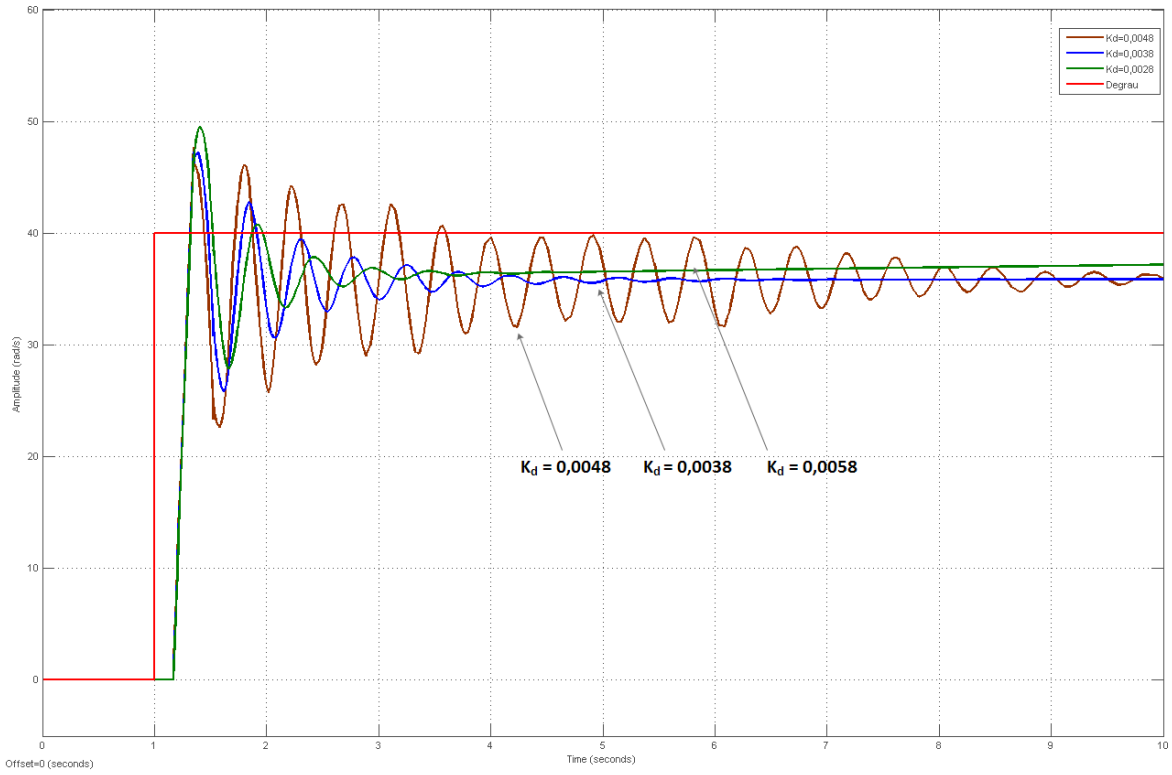


Figura 5.36: Resposta do controlador com $K_d = 0,0028; Kp = 0,0038; K_p = 0,0058$

o controlador PID, $K_p = 1,2/RL = 0,0433$, $K_i = K_p/2L = 0,1235$ e $K_d = 0,5LK_p = 0,0038$.

A implementação deste controlador não é possível utilizando somente uma FPAA, sendo necessárias três FPAA. Desta forma apresenta-se como proposta de implementação o circuito da Figura 5.37.

A aplicação pode assim ser desenvolvida no *software* AnadigmDesigner2, usando para tal uma CAM *FilterBiquad*, uma *FilterBilinear*, duas *Gaininv* para a realização do integrador fracionário (I^{α}), uma CAM *differentiator* e um *Gaininv* para a realização do derivador inteiro (D^{μ}), uma CAM para o ganho proporcional K_P e uma *Suminv* para realizar a soma dos três componentes.

A função de transferência implementada é a mesma do controlador PI^{λ} representada na equação (5.14) e equação (5.15). Na Tabela 5.4 são apresentados os parâmetros a inserir nos filtros bilinear e biquadrático da FPAA, na configuração representada na Figura 5.1.

Na Figura 5.38 estão apresentados os resultados da simulação em Matlab.

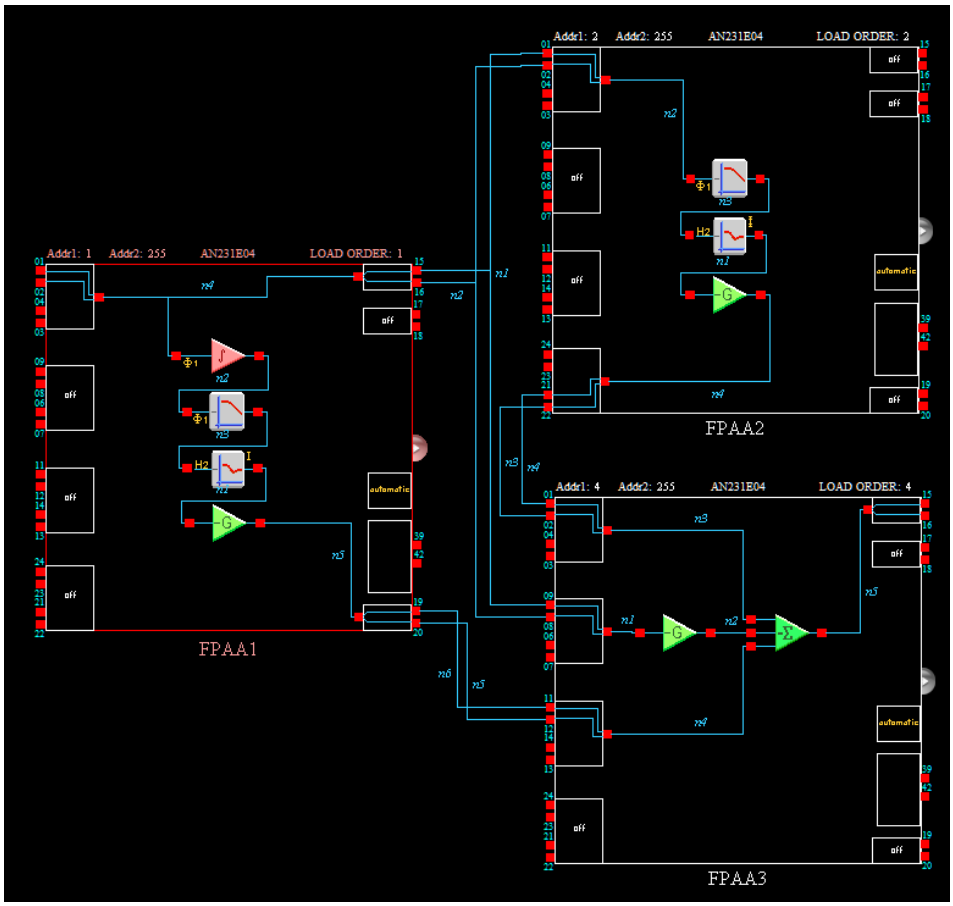


Figura 5.37: Implementação do filtro no AnadigmDesiner2 com dois FPAA

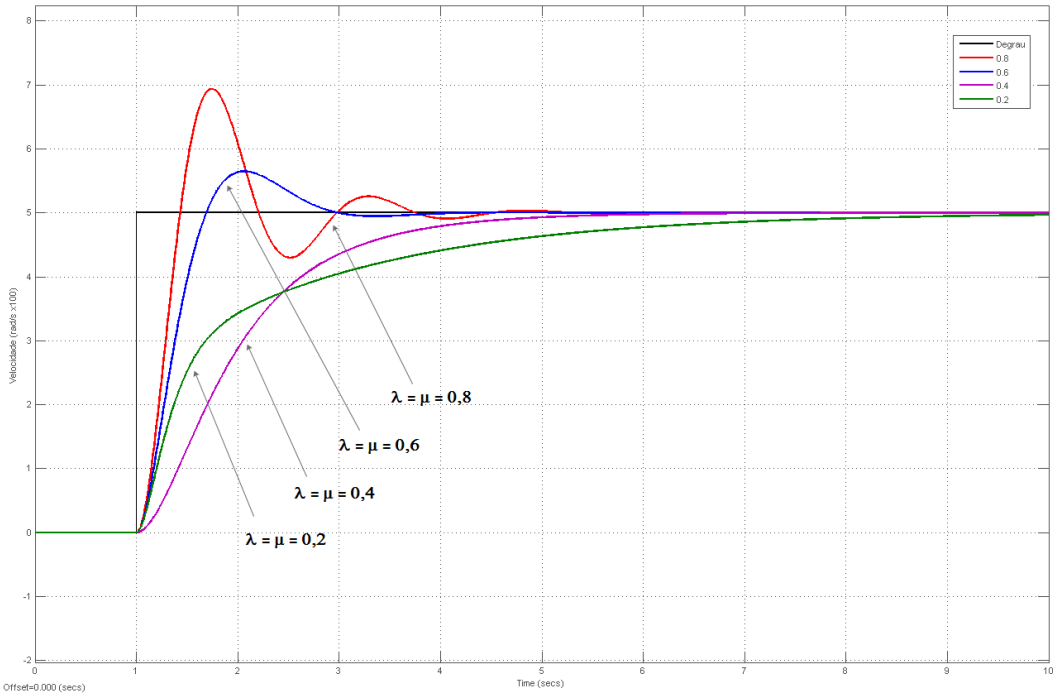


Figura 5.38: Resposta ao degrau dos controladores $\lambda = \mu = \{0,2; 0,4; 0,6\}$

5.7 DISCUÇÃO DOS RESULTADOS

Ficou comprovado nos ensaios realizados que é possível implementar controladores fracionários em FPAA. No entanto é necessário ter atenção a algumas restrições e cuidados a ter na sua implementação. O ruído é um fator que pode ter muita influência nos resultados obtidos onde é imperativo ter o sistema bem isolado de fontes perturbadoras, bem como usar cabos com malha ligada à terra. Outro fator é o valor pico a pico da entrada e saída da FPAA, considerando que estas são diferenciais, logo limita o valor do sinal a 2,5 V de amplitude máxima. Nesta implementação, outra condicionante foi o sistema de aquisição Hilink, embora para grande parte das simulações fosse suficiente, a sua largura de banda não permitiu observar as variações quase instantâneas dos sinais como no caso da resposta do derivador D^{μ} à onda quadrada. Outra limitação foi a frequência dos filtros Bilinear e Biquadrático que na FPA só pode ser implementada em kHz, este facto obrigou a subir o valor mínimo e máximo (w_l e w_h) da frequência na aproximação de Oustaloup para obter coeficientes suficientemente elevados de forma a conseguir a sua implementação.

Os resultados obtidos foram consistentes apresentado ligeiros desvios comparativamente aos modelos implementados em Matlab/Simulink. No derivador D^{μ} a resposta á onda quadrada foi semelhante à obtida em Simulink, com a exceção de não ser possível observar os picos das transições, facto explicado anteriormente. Na simulação com $K_d = 0,7$ obteve-se muita instabilidade e ruído de tal forma que a onda esteja somente aproximada à obtida por simulação.

A resposta do controlador I^{λ} à onda quadrada do sistema ao degrau também teve resultados consistentes com a simulação em Simulink. A variação de λ faz variar o tempo de resposta do sistema, variando também o *overshhot* e o tempo de subida, mantendo baixo o erro em regime permanente.

Os restantes controladores foram implementados e simulados somente em Matlab/Simulink, devido á necessidade de CAMs extras, o que implicaria à necessidade de FPAAs extras. Mas mesmo para estes, os resultados são bastante consistentes, ficando a recomendação para implementação dos circuitos com mais do que uma FPAA.

6. CONCLUSÃO

Apesar dos FPAA serem constituídos por componentes passivos, em particular por condensadores, não é permitido o acesso a estes na execução dos projetos, estando o projetista limitado aos circuitos pré-configurados designados por CAM. Este aspeto limita a utilização destes dispositivos, tornando-os pouco versáteis. O desempenho dos CAB depende muito das frequências de funcionamento selecionadas já que a gama de ganhos disponível depende desta, o que por vezes torna necessário decidir entre a rapidez e o nível de controlo. O número de CAM possíveis de implementar num só dispositivo é reduzido, o que faz com que só aplicações simples sejam possíveis de implementar num só FPAA, para aplicações mais complexas é necessário colocar várias em cascata. Apesar destas desvantagens, a FPAA apresenta bastantes vantagens, tais como o tempo de implementação e simulação dos sistemas ser bastante reduzido, ser fácil de programar ou de reprogramar. Aliado a isto, existe a vantagem de ser programada no próprio circuito, sem que para isso haja necessidade de ser removida. Tudo isto faz com que a FPAA seja um dispositivo com bastante viabilidade para a realização de circuitos analógicos, mas principalmente para o seu desenvolvimento.

Com a utilização da FPAA foi possível notar algumas vantagens, tais como alterar o sistema em plena utilização, o não envelhecimento dos componentes, alteração de parâmetros sem alteração física do circuito bem como a sua utilização ser rápida, poupando tempo.

Neste projeto foram analisadas três hipóteses de implementação dos circuitos fracionários nas FPAA, a implementação com amplificadores operacionais, via *Look Up Table*, e por filtros. Somente a última hipótese foi viável já que, tal como referido anteriormente, não é possível o acesso direto aos componentes da CAM.

Ficou demonstrado que o calculo fracionário é uma ferramenta bastante útil para o controlo de sistemas, podendo ser sintonizados pelas mesmas regras ou métodos que os sistemas inteiros. Os resultados obtidos neste trabalho confirmam a possibilidade de implementação de sistemas fracionários em dispositivos FPAA.

Os controladores implementados na FPAA obtiveram resultados, muito aproximados aos implementados em Matlab/Simulink, no entanto tem algumas restrições e é necessário alguns cuidados na sua utilização.

Como perspetiva de melhoramento futuro deste trabalho, propõe-se a implementação de um controlador fracionário $PI^{\lambda}D^{\mu}$ em quatro dispositivos FPAA, com filtros de quarta ordem implementados em dois filtros biquadráticos, desta forma o desempenho do controlador seria bastante melhorado dentro da gama de frequências testadas.

Referências Bibliográficas

ALLEN, P.E. (2001) - Analog CMOS Circuit Deisgn. 2001, Short Course Resources

ANADIGM (2000) - AN10E40 Field Programmable Analog Array. 2000, Datasheet

ANADIGM (2006) - AnadigmApex dpASP Family User Manual. 2006, Datasheet

ANADIGM (2008) - AN231K04-DVLP3 - AnadigmApex Development Board. 2008, Datasheet

BALEN, T. *et al.* (2004) - Testing the configurable analog blocks of field programmable analog arrays. In Test Conference, 2004. Proceedings. ITC 2004. International. 2004, p. 893–902

BARBOSA, R.; MACHADO, J.A.T.; JESUS, I.S. (2008) - On the Fractional PID Control of a Laboratory Servo System. In Proceedings of the 17th World Congress. The International Federation of Automatic Control (IFAC), 2008, p. 15273–15279

BARBOSA, Ramiro S.; MACHADO, J. A. Tenreiro; SILVA, Manuel F. (2006) - Time Domain Design of Fractional Differintegrators Using Least-squares. Signal Process. Vol. 86, 2006 nº 10, p. 2567–2581

BOHANNAN, Gary W. (2008) - Analog Fractional Order Controller in Temperature and Motor Control Applications. Journal of Vibration and Control, Vol. 14, 2008, p. 1487–1498

BONOMO, C *et al.* (2007) - A nonlinear model for ionic polymer metal composites as actuators. Smart Materials and Structures, Vol. 16, 2007, p. 1–2

CLARKE, P. (2000) - Startup rolls switched-capacitor analog array. EETIMES Electronic Magazine, Vol. [Em linha]., 2000, ˙[Consult. 2015–04–20]. Disponível em www: ⟨URL: `http://www.eetimes.com/document.asp?doc_id=1229253`⟩

D'MELLO, D.R. ; GULAK, P.G. (1998) - Design Approaches to Field-Programmable Analog Integrated Circuits. Analog Integrated Circuits and Signal Processing, Vol. 17, 1998, p. 7–34

GAUDET, Vincent C. ; GULAK, P.G. (1998) - IMPLEMENTATION ISSUES FOR HIGH-BANDWIDTH FIELD-PROGRAMMABLE ANALOG ARRAYS. Journal of Circuits, Systems and Computers, Vol. 08, 1998, p. 541–558

GULAK, P.G. (1995) - Field-Programmable Analog Arrays: Past, Present and Future Perspectives. Proc. IEEE Region 10 Int. Conf. Microelectron. VLSI, 1995, p. 123–126

GULAK, P.G. ; D'MELLO, D.R. (1996) - Review of field-programmable analog arrays. In High-Speed Computing, Digital Signal Processing, and Filtering Using Reconfigurable Logic. Vol. 2914, 1996, p. 152–169

HALL, T.S. *et al.* (2005) - Large-scale field-programmable analog arrays for analog signal processing. Circuits and Systems I: Regular Papers, IEEE Transactions on, Vol. 52, 2005, p. 2298–2307

HALL, Tyson S.; HASLER, Paul; AANDERSON, David V. (2002) - Field-Programmable Analog Arrays: A Floating-Gate Approach. In Proceedings of the Reconfigurable Computing Is Going Mainstream, 12th International Conference on Field-Programmable Logic and Applications. Springer-Verlag, 2002, p. 424–433

HASLER, P.E. ; TWIGG, C.M. (2005) - An OTA-based Large-Scale Field Programmable Analog Array (FPAA) for faster On-Chip Communication and Computation. Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on, 2005, p. 177–180

HILFER, R. (2000) - Applications of Fractional Calculus in Physics. World Scientific Publishing 2000

IMP, Inc (1995) - Preliminary Product Information - IMP50E10 EPAC (Electrically Programmable Analog Circuit). 1995, Datasheet

JESUS, Isabel S. ; MACHADO, Tenreiro J.A. (2009) - Development of fractional order capacitors based onÂ electrolyte processes. Nonlinear Dynamics, Vol. 56, 2009, p. 45–55

KLEIN, Hans W. (1996) - The EPAC architecture: an expert cell approach to field programmable analog circuits. In Circuits and Systems, 1996., IEEE 39th Midwest symposium on. Vol. 1, 1996, p. 169–172

LEE, E.K.F. ; GULAK, P.G. (1991) - A CMOS Field-Programmable Analog Array. IEEE Journal of Solid State Circuits, Vol. 26, 1991, p. 1860–1867

LEE, E.K.F. ; GULAK, P.G. (1992) - Field Programmable Analogue Array Based On MOSFET Transconductors. Electronics Letters, Vol. 28, 1992, p. 28–29

LEE, E.K.F. ; GULAK, P.G. (1995) - A transconductor-based field-programmable analog array. In Solid-State Circuits Conference, 1995. Digest of Technical Papers. 41st ISSCC, 1995 IEEE International. 1995, p. 198–199

LEE, E.K.F. ; HUI, Wai L. (1998) - A Novel Switched-Capacitor Based Field-Programmable Analog Array Architecture. Analog Integr. Circuits Signal Process. 1998, p. 35–50

MACHADO, J.A.T (2003) - Probabilistic Interpretation of the fractional-order differentiation. Fractional Calculus & Applied Analysis, Vol. 6, 2003 nº 1, p. 73–80

MANABE, S. (1961) - The non-integer integral and its application to control systems. Japanese Institute of Electrical Engineers, Vol. 6, 1961, p. 83–87

OLDHAM, Keith B. ; SPANIER, Jerome (1974) - The fractional calculus : theory and applications of differentiation and integration to arbitrary order. Academic Press, 1974, Mathematics in science and engineering

OUSTALOUP, A. (1975) - Etude et realization d'un systéme d'asservissement d'odre 3/2 de la fréquence d'un laser à colorant. Ph.D. Thesis, Universite Bordeaux I, France, 1975

PANKIEWICZ, B. *et al.* (2002) - A field programmable analog array for CMOS continuous-time OTA-C filter applications. IEEE Journal of Solid State Circuits, Vol. 37, 2002, p. 125–136

PIERCHALA, E. *et al.* (1995) - Current Mode amplifier/integrator for field programmable analog array. IEEE International, 1995, p. 196–197

PODLUBNY, I. *et al.* (2002) - Analogue realizations of fractional order controllers. In Nonlinear Dynamics. Vol. 29, 2002, p. 281–296

POLDUBNY, I. (1999) - Fractional-order systems and $PI^{\lambda}D^{\mu}$ - controllers. Automatic Control, IEEE Transactions on Vol. 44, 1999 nº 1

RADWAN, A. G.; SOLIMAN, A. M.; ELWAKIL, A. S. (2008) - FIRST-ORDER FILTERS GENERALIZED TO THE FRACTIONAL DOMAIN. Journal of Circuits, Systems and Computers, Vol. 17, 2008 nº 01, p. 55–66

SIVILOTTI, Massimo A. (1988) - A Dynamically Configurable Architecture for Prototyping Analog Circuits. In Proceedings of the Fifth MIT Conference on Advanced Research in VLSI. MIT Press, 1988, p. 237–258

VINAGRE, B. M. *et al.* (2000) - Some approximations of fractional order operators used in control theory and applications. Fractional Calculus & Applied Analysis, Vol. 3, 2000, p. 231–248

WESTERLUND, S. ; EKSTAM, L. (1994) - Capacitor theory. Dielectrics and Electrical Insulation, IEEE Transactions on, Vol. 1, 1994 nº 5, p. 826–839

YANG, Quan Chen; PETRÁŠ, I.; DINGYU, Xue (2009) - Fractional order control - A tutorial. In American Control Conference, 2009. ACC '09. 2009, p. 1397–1411

ZELTOM (2011) - Real-Time Hardware-In-The-Loop Control Platform For MATLAB/SIMULINK. 2011, Datasheet

ZETEX (1999) - Tottaly Re-Configurable Analog Circuit - TRAC. 1999, Datasheet

ZIEGLER, J.G. ; NICHOLS, N.B. (1942) - Optimum Settings for Automatic Controllers. Trans. ASME, Vol. 64, 1942, p. 759–768

Anexo A. *Código $PI^{\lambda}D^{\mu}$ PSE*

Pseudo-Código

```
K=0; Kp=0,4; Ki=0,9; Kd=1;
Lbda=0,5; Delta=0,5;
Ref=40; L=1; T=0,01; M=L/T, V=0;

*Iniciar o vector dos coeficientes*
q[0]=1, d[0]=1
For j=1 to L-1
  q[j]=[1-(1-lbda)/j].q[j-1]
  d[j]=[1-(1+delta)/j].d[j-1]
next j
u=0       *u=saída
Erro[k]=Ref-u *Ref=referência

***** loop principal *****
Loop
  p=Kp.erro[k]
  i=Ti/(T.exp(-lbda)).soma_i
  d=Td/(T.exp(delta)).soma_d
  u=p+i+d

  Reorganização da memória
  if k<M
    k=k+1
    Erro[k]=Ref-u
    V=0
  Else
    For n=0 to M-2
      Erro[n]=erro[n+1]
    Next n
    Erro[k]=Ref-u
    V=k-M
  endif
End loop
Somatório:
  Soma_q=0, soma_d=0
  For j=v to k
    Soma_q=soma_q+q[j].erro[k-j]
    Soma_d=soma_d+d[j].erro[k-j]
  next j
end
```

Anexo B. *Código $PI^{\lambda}D^{\mu}$ CFE*

Pseudo-Código

```
(* initialization code *)
scale := 32752; % input and output
order := 5; % order of approximation
U_FOC := 10; % input and output voltage range:
% 5[V], 10[V], ...
a[0] := ...; a[1] := ...; a[2] := ...;
a[3] := ...; a[4] := ...; a[5] := ...;
b[0] := ...; b[1] := ...; b[2] := ...;
b[3] := ...; b[4] := ...; b[5] := ...;
loop i := 0 to order do
  s[i] := 0;
endloop
(* loop code *)
in := (REAL(input)/scale) * U_FOC; feedback := 0;
  feedforward := 0; loop i:=1 to order do
  feedback := feedback - a[i] * s[i];
  feedforward := feedforward + b[i] * s[i];
endloop
s[0] := in + a[0] * feedback;
out := b[0] * s[1] + feedforward;
loop i := order downto 1 do
  s[i] := s[i-1];
endloop
output := INT(out*scale)/U_FOC;
```

Yang, Petráš e Dingyu (2009)

Anexo C. *Ziegler e Nichols, Função que determina K, L, T*

Função que determina K, L, T

Matlab

```
function [K, L, T, G] = zn_ma(t, v)
  s=tf('s');
  l=length(t);
  K=v(l);
  n=1;
  Dmax=0;
  while n<l-1
    Dv=(v(n+1)-v(n))/(t(n+1)-t(n));
    if Dmax<Dv
      Dmax=Dv;
      PDmax=n;
    end
    n=n+1;
  end
  m=Dmax;
  x=t(PDmax);
  y=v(PDmax);
  b=y-m*x;
  L=-b/m;
  tk=(K-b)/m;
  tk=tk*100;
  tk=round(tk)/100;
  Pk=1;
  while t(Pk)<tk
    Pk=Pk+1;
  end
  T=t(Pk)-L;
  G=K*exp(-L*s)/(T*s+1);
```