Universidade Federal da Bahia
Escola Politécnica da Bahia
Programa de Pós-Graduação em Mecatrônica

Vitor Leão Filardi

DESENVOLVIMENTO DE UM SISTEMA PARA NAVEGAÇÃO E TELEMETRIA DE AERONAVES NÃO-TRIPULADAS

Salvador
2006

Vitor Leão Filardi

DESENVOLVIMENTO DE UM SISTEMA PARA NAVEGAÇÃO E TELEMETRIA DE AERONAVES NÃO-TRIPULADAS

Trabalho apresentado ao Programa de Pós-graduação em Mecatrônica, programa conjunto entre o Departamento de Engenharia Mecânica e o Departamento de Ciência da Computação, da Universidade Federal da Bahia como requisito parcial para obtenção do grau de Mestre em Mecatrônica.

Orientador: Prof. Dr. *Leizer Schnitman*

Co-orientador: Prof. Dr. *Carlos Arthur Mattos Teixeira Cavalcante*

Co-orientador: Prof. Dr. *Iuri Muniz Pepe*

Salvador
2006

TERMO DE APROVAÇÃO

VITOR LEÃO FILARDI

DESENVOLVIMENTO DE UM SISTEMA PARA NAVEGAÇÃO E TELEMETRIA DE AERONAVES NÃO-TRIPULADAS

Dissertação aprovada como requisito parcial para obtenção do grau de Mestre em Mecatrônica, Universidade Federal da Bahia, pela seguinte banca examinadora:

Carlos Arthur Mattos Teixeira Cavalcante
Doutor em Engenharia de Produção, Universidade de São Paulo (USP).
Universidade Federal da Bahia

Iuri Muniz Pepe
Doutor em Física Nuclear,Universite Catholique de Louvain, U.C.L., Bélgica.
Universidade Federal da Bahia

Leizer Schnitman - Orientador
Doutor em Engenharia Eletrônica e Computação, Instituto Tecnológico de Aeronáutica (ITA).
Universidade Federal da Bahia

Marcelo Ricardo Stemmer
Doutor em Automação Industrial, Rheinisch-Westfalischen Technischen Hochschule/Aachen, R.W.T.H.A., Alemanha.
Universidade Federal de Santa Catarina

Paulo Eigi Miyagi
Doutor em Engenharia de Controles, Tokyo Institute of Technology, TITECH, Japão.
Universidade de São Paulo (USP)

Salvador, 22 de setembro de 2006.

Dedico aos meus pais, irmãos e Rebeca

AGRADECIMENTOS

Antes de entrar no mérito daqueles que conviveram comigo durante todos os anos de pesquisa, quero agradecer a Deus por todos os caminhos e riscos em que me colocou, e com muita fé pude discenir entre as verdades e as mentiras, onde e quando podia seguir ou não.

Quero agradecer aos meus pais, irmãos e a Rebeca por me terem dado a educação, a fé, o empenho e a coragem de ir e vencer. Alguns deles próximos, outros à distância, mas sempre presentes.

Gostaria de agradecer ao meu orientador, Prof. Dr. Leizer por ter acreditado e me confiado tal trabalho. Ao Prof. Dr. Iuri pelo seu bom humor e genuíno gosto pelo ensino (pelo próprio ato de ensinar às pessoas). Ao Prof. Dr. Arthur pelo entusiasmo a cada etapa concluída e a vontade de acompanhar tudo de perto. Muito obrigado por me ajudarem a realizar este sonho.

Aos amigos Marcelo Cad, Waslon Terllizzie, Luiz Carlos vulgo Nescau, Robson Veras, Tiago Fróes e a todos dos laboratórios (LaPO, SGPA e AERO-UFBA) e colegas do mestrado, sempre curiosos e prestativos quando eu precisava de alguma ajuda. Gostaria de citar os nomes, mas poderia acabar omitindo alguém, por isso achei melhor não arriscar. Valeu, pessoal.

"É melhor tentar e falhar,
que preocupar-se e ver a vida passar;
é melhor tentar, ainda que em vão,
que sentar-se fazendo nada até o final.
Eu prefiro na chuva caminhar,
que em dias tristes em casa me esconder.
Prefiro ser feliz, embora louco,
que em conformidade viver ..."

—MARTIN LUTHER KING

RESUMO

A pesquisa para o desenvolvimento do conjunto de circuitos eletrônicos (*hardware*) para aquisição de dados e sistemas de controle de vôo para aeronaves não-tripuladas (UAV -*Unmanned Aerial Vehicle*) tem gerado novos conhecimentos aeronáuticos, com aplicações em monitoramento ambiental e controle de áreas agrícolas e de pragas. Neste trabalho, foi concebido o desenvolvimento de todo o aparato eletrônico necessário para um UAV, sendo este desenvolvimento parte integrante do projeto UAV-UFBA. Apresenta-se aqui um detalhamento do funcionamento dos sensores utilizados para o sistema de navegação inercial, bem como todas as equações necessárias para a conversão dos valores de tensão adquiridos pelos sensores em suas respectivas unidades de medida: aceleração, pressão, velocidade do angular e outros. São também apresentados os filtros analógicos e digitais desenvolvidos e os circuitos de condicionamento de sinal de cada sensor. O sistema de processamento é composto por dois microcontroladores da família MCS-51, como módulo de interface entre o sistema de navegação e a aeronave. A arquitetura desenvolvida é flexível, podendo ser utilizada de várias formas e expandida de acordo com a necessidade. Para validar o equipamento foram feitos diferentes tipos de ensaio. As respostas obtidas tiveram erros inferiores a 4%, levando em consideração que se esperava erros maiores, próximos a 20%, visto que a equipe não tinha experiência nesta área, nem outros trabalhos como referência. Desta maneira mostra que com a continuidade e o aprofundamento da pesquisa, pode-se obter resultados ainda melhores.

Palavras-chaves: Aeronave Não-Tripulada, Monitoramento Aéreo Autônomo, Aquisição de Dados em Tempo-Real, Sistema de Navegação Inercial.

ABSTRACT

Hardware development for data acquisition and navigation system are the basis of autonomous flight control in unmaned aerial vehicles (UAV). These subjects are relatively new for the national aeronautics and they have applications in environmental monitoring, agriculture and plague control. This work was is concerned the development of all electronic apparatus used in an UAV and it is a part of the UAV-UFBA project. This thesis presents a detailed description of the overall system as well as the sensors which were used in the Inertial Navigation System. Furthermore, all necessary equations for conversion of the voltage values acquired by the sensors in its corresponding units of measure: acceleration, pressure, angle velocity, and so on. The analog and digital filter circuitry for signal conditioning of each sensor are also presented. The system is composed of two microcontrollers of the MCS-51 Family which work as an interface between the navigation system module and the aircraft. The proposed architecture is flexible and may be used for others applications or expanded according to new requeriments. In order to validate the equipment, some experiments have been performed. The gotten answers had had inferior errors the 4%, leading in consideration that if waited bigger errors, near to 20%, since the team did not have know-how in this area, nor other works as reference. Showing that with the continuity and the deepening of the research, they can be gotten resulted still better.

Keywords: UAV, Control System for UAV, Real-Time Data Aquisition, Inercial System Navigation.

SUMÁRIO

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

A/D	Analógico/Digital
ARARA	Aeronaves de Reconhecimento Assistidas por Rádio e Autônomas
AROD	Airborne Remotely Operated Device
AURORA	Autonomous Unmanned Remote Monitoring Robotic Airship
AVATAR	Autonomous Vehicle Aerial Tracking and Retrieval
DAAV	Dirigible Autonomous Aerial Vehicle
DC	Corrente contínua
DEM	Departamento de Engenharia Mecânica
E/S	Entrada/Saída
EMBRAPA	Empresa Brasileira de Pesquisa Agropecuária
EPUFBA	Escola Politécnica da UFBA
FFT	Fast Fourier Transform
GPS	Global Position System
LCD	Light Cristal Display
MEMS	Micro Eletronic Mechanical System
NTC	Negative Temperature Coeficient
PEE	Programa de Especialização em Engenharia
PID	Proporcional-Integral-Derivativo
PPGM	Programa de Pós-Graduação em Mecatrônica
PWM	Pulse Width Modulation
R/C	Rádio Controle
RC	Resistor-Capacitor
RPH	Remotely Piloted Helicopter
RPV	Remotely Piloted Vehicle
RTLinux	Real-Time Linux
SAE	Society of Automotive Engineers
SI	Sistema Internacional
SMD	Surface-Mountable-Device
SPV	Self Piloted Vehicle
UAV	Unmanned Aerial Vehicle
UCN	Unidade de Controle e Navegação
UMA	Unmanned Aircraft
UTA	Unmanned Tactical Aircraft

UVS Unmanned Vehicle System

LISTA DE SÍMBOLOS

α_a	Aceleração
$\ddot{z}_p$	Aceleração no eixo p
$\dot{v}_p$	Velocidade no eixo p
λ	Constante de decaimento térmico
ω	Velocidade angular
ρ_o	Densidade volumétrica do ar
θ	Ângulo qualquer
$\triangle V$	Diferença de potencial elétrico
A	Área
Alt	Altitude
b_a	Coeficiente de viscosidade
F	Força
F_a	Força de aceleração
g	Aceleração da gravidade
Gnd	Terra do circuito
k_a	Constante da mola
L	Comprimento da corda
m	Massa
m_a	Massa acelerada
P	Pressão
P_{mar}	Pressão ao nível do mar
Ps	Peso
R	Constante do Ar - Lei Geral dos Gases
$refA_{inicial}$	Tensão do sensor quando em repouso
Rf	Resistores da parte flexível
Rr	Resistores da parte rígida
t	Período de um ciclo
T	Temperatura
T_A	Temperatura de aferição dos acelerômetros
T_o	Temperatura ambiente em Kelvin
Tr	Tração da corda
U_{+g}	Tensão da gravidade positiva
U_{-g}	Tensão da gravidade negativa
V	Tensão elétrica
v	Velocidade linear

v_a	Velocidade da massa
V_{lido}	Valor de tensão lido pelo A/D
V_{ref}	Valor de tensão por dígito convertido
V_{out}	Tensão de saída
$Valor_{conv}$	Valor digital convertido
Vcc	Tensão de alimentação
VDD	Tensão de alimentação
w_c	Freqüência de corte
z_a	Deslocamento da massa

CAPÍTULO 1

INTRODUÇÃO

Nos últimos anos, diversas universidades em todo o mundo vêm fazendo pesquisas relacionadas a aeronaves não-tripuladas (*Unmanned Aerial Vehicle*-UAV) de pequeno e médio portes. As pesquisas desenvolvidas nessas universidades [(NERIS, 2001),(AEROSONDE, 2001), (AVATAR, 2004) e outras] têm permitido gerar novos conhecimentos nas diversas áreas relacionadas à aeronáutica, tais como *design*, fabricação, instrumentação e sistemas para vôo autônomo.

Na literatura são encontrados vários termos que definem uma aeronave não-tripulada. A sigla UAV é comumente associada a *Unmanned Aerial Vehicle*, podendo ainda ser associada a *Uninhabited Aerial Vehicle* e *Unoccupied Air Vehicle*. Denominações como UMA *Unmanned Aircraft*, UVS *Unmanned Vehicle System*, SPV *Self Piloted Vehicle*, UTA *Unmanned Tactical Aircraft* e RPV *Remotely Piloted Vehicle* também são comumente aplicadas. Esta última designa aeronave não-tripulada que é pilotada remotamente como os aeromodelos controlados por radiocontrole (R/C) [(HALLBERG; KAMINER; PASCOAL, 1999), (DITTRICH, 2002), (DIXON; WICKENS, 2003) e (ELFES; MAETA, 1998)].

Encontram-se, também, termos específicos de acordo com o tipo e a utilização da aeronave: RPH (*Remotely Piloted Helicopter*) para helicópteros, DAAV (*Dirigible Autonomous Aerial Vehicle*) para dirigíveis e AROD (*Airborne Remotely Operated Device*) para qualquer dispositivo aéreo remotamente operado. Para simplificar, o termo UAV compreende qualquer aeronave não-tripulada com ou sem asas, assistida por um operador terrestre ou aéreo (NERIS, 2001).

A idéia de aeronaves tipo UAV não é nova. Durante a 2ª Guerra Mundial, já se pensava na utilização dessas aeronaves que, além de evitar perigo aos pilotos, permitem também controlar o avião em condições adversas. Em não havendo um piloto a bordo, os UAV podem realizar manobras e/ou suportar melhor, por exemplo, a influência da gravidade; podem voar em alturas elevadas sem necessidade de qualquer sistema para manutenção de vida humana; ou voar em missões muito mais longas sem se preocupar com a fadiga do piloto. Podem ser usados em vôos de reconhecimento dentro do território inimigo sem expor a vida humana. Tais veículos podem também ser construídos segundo um *designer* que otimiza a aerodinâmica, sem que haja a preocupação com a ergonometria interna, como, por exemplo, o assento do piloto. (DIXON; WICKENS, 2003), e (DITTRICH, 2002).

A possibilidade dos UAV realizarem missões de reconhecimento em campos de batalha motivou um crescente interesse militar. Depois de uma demonstração bem sucedida em um lançamento de míssil por um Predator (Figura 1.1), que acertou um tanque em terra no dia 21 de fevereiro de 2001, os UAV evoluíram de um mero recurso experimental não letal para uma arma de guerra considerada mortal e altamente precisa (AIRFORCE..., 2006).

Figura 1.1. Predator com mísseis Hellfire, Airforce Technology (AIRFORCE..., 2006).

Uma vez que as aplicações bélicas foram bem sucedidas, os UAV atraíram também a atenção para aplicações civis como: controle da área de queimadas, monitoramento de pragas, monitoramento de áreas de desastres, observação do tempo e a pesquisa

científica [(ELFES; MAETA, 1998), (HILL et al., 2004), (MONTGOMERY, 2000), (RAMOS et al., 2001) e (PICCOLO, 2003)].

Assim, com o crescente uso deste tipo de aeronave em aplicações civis, um maior número de pesquisadores nesta área do conhecimento, mais recursos para financiamento e um maior interesse do setor produtivo vêm incentivando a pesquisa nacional. O uso do UAV nas áreas de monitoração ambiental e monitoração de áreas agrícolas de cultivo são potenciais aplicações a serem exploradas em nosso país.

O barateamento e miniaturização dos componentes necessários ao controle de vôos de aeronaves permitiram o desenvolvimento de UAV de baixo custo. No Brasil, os projetos de maior destaque desenvolvidos nesta área são: ARARA (Aeronaves de Reconhecimento Assistidas por Rádio e Autônomas), uma parceria da USP-São Carlos com a EMBRAPA e o projeto AURORA do ITI de Campinas (MAETA, 2001).

Figura 1.2. Aeronave UAV - Projeto ARARA (NERIS, 2001)

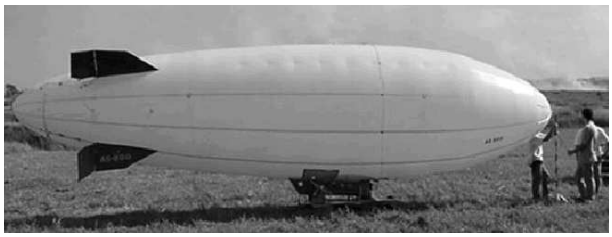

Figura 1.3. Dirigível UAV - Projeto AURORA (MAETA, 2001)

A partir da iniciativa de um grupo de professores e alunos do Departamento de Engenharia Mecânica (DEM) da UFBA em participar da Competição SAE BRASIL AeroDesign em São José dos Campos - São Paulo, nasceu o Projeto AERO-UFBA (SAE BRASIL, 2004). Concebido como um projeto permanente, objetivando a médio e longo prazos a implantação da engenharia aeronáutica na Escola Politécnica da UFBA, o Projeto AERO-UFBA conferiu visibilidade às ações então desenvolvidas. Isto inseriu a Bahia no roteiro de seleção de novos talentos da EMBRAER e implantou as sementes de uma "cultura aeronáutica" na UFBA, cultura esta que tem motivado alunos e professores a prosseguirem com tal iniciativa.

O ano de 2003 foi marcado pela implantação do Programa de Pós-Graduação em Mecatrônica (PPGM) da UFBA, seguido de uma grande concentração de esforços para a sua consolidação. Realizado conjuntamente pelo Departamento de Engenharia Mecânica (DEM/EPUFBA) e pelo Departamento de Ciências da Computação (DCC/IM), este programa trouxe espontaneamente novas demandas e contribuições para o Projeto AERO-UFBA.

De um lado, vários pesquisadores perceberam neste projeto a oportunidade da aplicação tecnológica do produto de suas pesquisas acadêmicas - em especial, aqueles dedicados às áreas de teoria de controle e métodos formais em ciências da computação. Por outro lado, a chegada desses novos estudantes do mestrado do PPGM criou a demanda do apoio acadêmico e laboratorial junto ao Projeto AERO-UFBA, fazendo com que este projeto fosse dividido em duas frentes de trabalho, o UFBA-AERODESIGN e o UAV-UFBA.

a) O projeto UFBA-AERODESIGN é hoje responsável pela construção de aeromodelos, visando à competição da SAE-BRASIL AeroDesign (SAE BRASIL, 2004). O domínio dos aspectos referentes ao projeto de uma aeronave em escala (aeromodelo) permitirá o desenvolvimento de um aeromodelo com caracterísitcas apropriada para um UAV de monitoramento. Atualmente, a aeronave construída para a competição SAE-BRASIL já representa uma possível base de testes para o UAV-UFBA.

b) Por sua vez, o UAV-UFBA visa a desenvolver um aeromodelo autônomo capaz de monitorar reservas ecológicas e áreas de desmatamento, auxiliar no controle de pragas, além de prover novas bases de conhecimento para a pesquisa científica.

De modo geral, são objetivos do Projeto AERO-UFBA:

a) estudo, desenvolvimento, implantação e difusão de temas e linhas de pesquisa relativos à área de engenharia aeronáutica em nossa universidade;

b) implantação de aspectos básicos da engenharia aeronáutica entre as atividades curriculares dos cursos de graduação em engenharia;

c) orientação e integração de estudantes de graduação e de pós-graduação no desenvolvimento de estudos científicos e tecnológicos em temas aeronáuticos e suas aplicações;

d) reunião de competências científicas locais para viabilizar suporte técnico a empreendimentos voltados para o desenvolvimento de produtos e componentes aeronáuticos inovadores, bem como para sua aplicação em segmentos diversos.

Atualmente, o projeto UAV-UFBA está em fase de especificação. Inicialmente, foram levantadas 14 etapas para a conclusão total do projeto:

1) desenvolvimento de um algoritmo para controle da atitude de vôo de aeronaves em escala (aeromodelos);

2) seleção e/ou desenvolvimento de dispositivos mecatrônicos para o vôo autônomo de aeronaves em escala (aeromodelos);

3) desenvolvimento de um algoritmo para a navegação autônoma por pontos previamente selecionados de aeronaves em escala (aeromodelos);

4) testes de bancada dos dispositivos e instrumentos reais para o vôo autônomo aplicável aos algoritmos desenvolvidos;

5) *design* e projeto de uma aeronave em escala que comporte a instrumentação e o sistema computacional embarcado para o vôo autônomo;

6) construção de uma aeronave em escala segundo as especificações técnicas;

7) integração do sistema de vôo autônomo com a aeronave;

8) desenvolvimento de um sistema de transmissão de dados para monitoramento de dados enviados pela aeronave de forma *on-line*;

9) desenvolvimento de um sistema de transmissão de vídeo para acompanhar em terra as imagens captadas pela aeronave;

10) estudo do reconhecimento de imagens para módulo de visão computacional da aeronave;

11) adição do módulo de visão computacional ao sistema computacional embarcado na aeronave;

12) desenvolvimento de um módulo de inteligência artificial, para tomada de decisões durante o vôo;

13) aprimoramento do sistema de vôo autônomo para suportar decolagens e aterrisagens;

14) desenvolvimento de instrumentos/componentes de vôo específicos para UAV que não são encontrados no mercado, que são de tecnologia proprietária ou são muito caros.

Este trabalho propõe o desenvolvimento de um sistema microcontrolado (*hardware*) capaz de fazer o controle de vôo, a navegação e a aquisição de dados de uma aeronave não-tripulada. Tal esforço se insere na 2ª e uma parte da 4ª etapa do projeto UAV-UFBA.

1.1 PROBLEMA ABORDADO

Os problemas abordados foram: determinação do conjunto de sensores, desenvolvimento de uma eletrônica embarcada (*hardware*) capaz de fazer a navegação inercial e telemetria, além de fornecer o suporte computacional necessário para os ensaios, e validação dos dados adquiridos.

Priorizou-se um projeto de baixo custo, tentando sempre utilizar a tecnologia disponível no mercado nacional. Entre outros aspectos, o hardware desenvolvido se baseou nas seguintes premissas:

i) ter o número de entradas/saídas (E/S), analógicas ou digitais, compatíveis com os elementos sensores e atuadores definidos para o devido controle da aeronave;

ii) ser o mais leve possível, pois peso é não desejado;

iii) ser o menor possível, pois espaço não é abundante;

iv) baixo consumo de bateria. Considerando o peso usual de baterias, um consumo de energia exagerado pode limitar a autonomia do vôo;

v) ter uma comunicação por rádio de tal forma a:

- permitir ao operador a definição do modo de controle vôo: autônomo / manual (por radiocontrole);
- receber informações de pontos de referência (trajetória) definida pelo operador;
- transmitir informações dos seus sensores/atuadores para uma estação em solo.

A proposta acima, segue a concepção utilizada pelos autores (HALLBERG; KAMINER; PASCOAL, 1999) e (KAHN; KELLOG, 2003), onde a utilização de uma arquitetura aberta e flexível possibilita a expansão e o aprimoramento do sistema.

Desta forma, a arquitetura proposta do sistema a ser implementado pode ser visualizada na Figura 1.4. O processador central utilizado nesta unidade foi um microcontrolador da família MCS-51. Este será responsável pela aquisição de dados dos sensores, envio dos sinais de controle e interface entre o módulo de controle e a Unidade de Controle e Navegação (UCN) da aeronave, sendo a que a UCN pode estar ou não embarcada na aeronave.

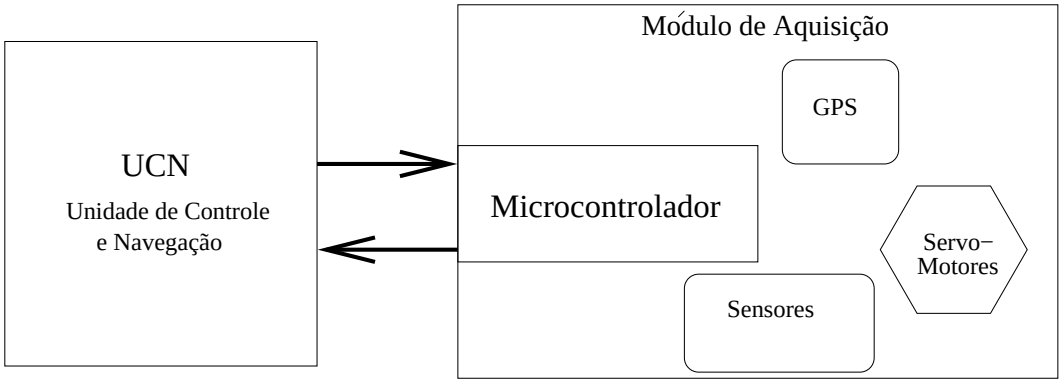

Figura 1.4. Arquitetura básica do sistema desenvolvido

1.2 ORGANIZAÇÃO DO TEXTO

No capítulo 2, são abordados o conceito de navegação aérea e as forças envolvidas no vôo de uma aeronave. Detalha-se o sistema de navegação inercial e seus aspectos conceituais como: princípio de funcionamento dos sensores e dos atuadores de controle de uma aeronave em escala (aeromodelo).

No capítulo 3, são apresentados o panorama atual dos dispositivos comerciais existentes e algumas pesquisas correlatas feitas no Brasil e no mundo. Inspirados nestes sistemas algumas caracterísitcas foram desenvolvidas para o Projeto AERO-UFBA.

O Capítulo 4 mostra os módulos desenvolvidos, bem como os circuitos eletrônico e os cálculos envolvidos para cada sensor. Desta maneira, pretende-se detalhar e proporcionar melhor compreenção da arquitetura proposta, assim como a estratégia de aquisição de dados e dos softwares envolvidos.

O Capítulo 5 descreve o ambiente de ensaio em laboratório, bem como os testes aplicados e seus resultados. Ainda neste capítulo são mostrados os resultados obtidos em situações reais de utilização.

O Capítulo 6 traz a conclusão sobre o sistema desenvolvido e resultados obtidos, bem como algumas perspectivas para a continuidade do Projeto AERO-UFBA.

CAPÍTULO 2

PROJETO DE UM SISTEMA DE NAVEGAÇÃO INERCIAL

2.1 SISTEMA DE NAVEGAÇÃO

Formalmente, navegação é o processo de conduzir um objetvo móvel qualquer de um ponto a outro usando algum método de orientação (SMITH, 1993). Um sistema de navegação tem como principal objetivo adquirir a posição relativa do veículo e processá-la de tal forma a prover as informações de direção e sentido.

Um sistema de navegação pode obter informações das mais variadas formas. A informação básica fornecida por todo sistema de navegação é a posição absoluta, geralmente dada em coordenadas geográficas (latitude, longitude e altitude). Existem outras formas de navegação por posição relativa, e que podem ser classificadas como (NERIS, 2001):

- Correção de Posição: consiste na correção da posição de um veículo em relação a um ponto de referência. É o método mais simples e mais antigo. Existem três regras básicas: navegação celestial, leitura de mapas e de distância e direção de pontos identificáveis;

- *Dead Reckoning*: com base na última informação conhecida do veículo (a média da velocidade e a direção), estima-se a sua futura posição a partir dessas informações até se obter uma nova informação;

- Navegação Inercial: é um sistema que determina a posição e a velocidade através da medida da aceleração. A aceleração é obtida através de um sensor interno e a velocidade é calculada usando-se um processo de integração. Uma segunda

integração sobre os dados de velocidade determina a posição do veículo com base na sua posição e velocidade inicial.

- RadioNavegação: determina-se a posição do veículo com base no tempo de viagem de uma onda eletromagnética transmitida da estação-base até o veículo.

  GPS - *Global Position System*: é um sistema de radionavegação baseado em satélites artificiais. O aparelho receptor obtém a sua posição por um conjunto de satélites artificiais em relação a uma referência inercial e à sua posição em relação a este conjunto. Geralmente este sistema consiste em 24 satélites artificiais em 6 órbitas circulares de 20.200 km acima da crosta terrestre, em um período de 12 horas.

Para que uma aeronave seja capaz de executar uma determinada missão é necessário que ela consiga determinar a sua posição geográfica de partida, ou de um ponto de referência, assim como os outros pontos de sua rota, a fim de alcançar o seu destino final. Desta maneira, um sistema de navegação completo contempla as seguintes variáveis: tempo, velocidade, distância entre os pontos, latitude, longitude, direção do pólo magnético, direção e velocidade do vento e posições relativas de pontos de referência terrestres. O conjunto de variáveis acima não é absolutamente necessário para que um sistema de navegação funcione. De uma forma geral, sempre é possível a utilização de um conjunto reduzido destas informações, desde que este conjunto forneça informações suficientes para o deslocamento da aeronave de um ponto desejado a outro.

2.2 RECURSOS NECESSÁRIOS PARA UM VÔO AUTÔNOMO

Conforme foi visto na Secção 2.1, a navegação inercial consiste em um sistema capaz de determinar posição e velocidade através da aceleração. É sabido que estas grandezas estão diretamente relacionadas, bastando para tanto que sejam aplicados operadores derivativos ou integrais. Sendo uma forma simples e barata para um sistema de navegação. Porém, para um vôo autônomo, existem outras variáveis envol-

vidas que não apenas a posição e a velocidade da aeronave. Assim, velocidade do ar, ângulo de arfagem, ângulo de rolagem e ângulo de guinada, conforme ilustrado na Figura 2.1, são informações relacionadas ao controle da aeronave. Ou seja, outros sensores serão necessários para um sistema de vôo autônomo.

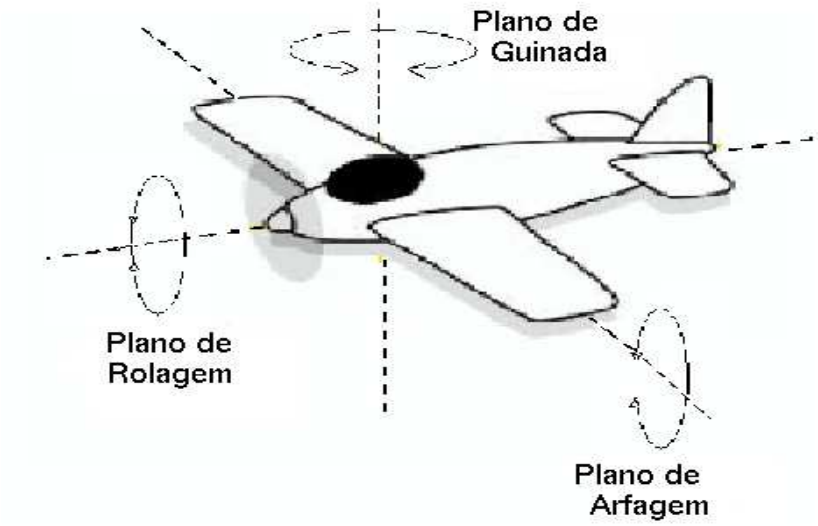

Figura 2.1. Principais planos de movimento de um avião.

O conhecimento desses ângulos (arfagem, rolagem e guinada) é utilizado para determinar as forças que compõem a dinâmica da aeronave, como exemplificado na Figura 2.2. O sistema de vôo autônomo deve ser capaz de efetuar as medidas desses ângulos, colocando-os à disposição do sistema de controle. Os sensores necessários para o sistema de vôo autônomo inercial proposto são:

- Acelerômetro - responsável pela medida da aceleração linear da aeronave;
- Giroscópio - responsável pela medida dos ângulos dos planos de movimento;
- Barômetro - responsável pela medida de altitude e velocidade do ar.

Figura 2.2. Principais forças que compõem a dinâmica da aeronave.

Assim, as forças que compõem a dinâmica da aeronave são definidas como:

- **Força de Tração** - É a força produzida pelo propulsor da aeronave, fazendo com que este móvel se desloque para frente. Quase sempre esta força é paralela ao eixo longitudinal da aeronave.
- **Força de Sustentação** - É a força gerada pelo empuxo aerodinâmico da aeronave quando esta se desloca por uma massa de ar. O ponto por onde passa a linha de ação da força de sustentação é chamado de centro de sustentação, ou centro de pressão. Sua direção é sempre perpendicular ao fluxo de ar relativo.
- **Força-Peso** - É a força de atração gravitacional que a Terra exerce sobre a massa da aeronave. É representada sempre a partir de um ponto imaginário, como se toda a massa estivesse concentrada este ponto, também chamado de centro de gravidade ou centro de massa.
- **Força de Arrasto** - É a resistência causada pelas componentes das forças aerodinâmicas geradas pela aeronave quando se desloca através de uma massa de ar. Sua direção é paralela à direção ao fluxo de ar e geralmente em sentido oposto ao sentido de deslocamento da aeronave.

Existem ainda outras três forças que são: força centrífuga (esta força aparece quando o avião está descrevendo uma curva), efeito de torque do motor (tende a

impor um movimento circular no mesmo sentido de rotação da hélice) e o fator de carga (o esforço tensional sobre a estrutura do avião, devido à distribuição de carga ao longo de sua estrutura, é uma força que tende a tirar a aeronave de sua trajetória retilínea) (NERIS, 2001).

2.3 SENSORES UTILIZADOS

A partir da revisão bibliográfica, foi possível tomar conhecimento de um certo número de sistemas comerciais e dos sensores normalmente utilizados neste tipo de aplicação (PICCOLO, 2003), (HALLBERG; KAMINER; SILVESTRE, 1998), (HALLBERG; KAMINER; PASCOAL, 1999),(JONHSON; FORTAINE, 2001) e (KAHN; KELLOG, 2003).

2.3.1 Sensor de Pressão

Os sensores de pressão empregados no sistema servem para determinar a altitude da aeronave e a sua velocidade em relação ao ar, através de um tubo de Pitot. A altitude também poderia ser determinada pelo receptor GPS, entretanto sensores de pressão proporcionam usualmente uma maior confiabilidade que o receptor de GPS (PICCOLO, 2004).

Princípio de Funcionamento

Para se ter um sensor capaz de medir a pressão são necessários dois componentes essenciais. Em primeiro lugar, é preciso dispor de uma área (A) conhecida, associada a um transdutor sensível à força aplicada (F) sobre esta área. Ambos os componentes podem ser fabricados em silício, mas para isso se faz necessário ter um diafragma com área (A) onde será montado um par de elementos piezo-resistivos. Este diafragma

consiste em uma fina camada de sílicio que age como material elástico, onde pode ser crescido por litografia, ataque químico e métodos de deposição de dopantes; e os resistores piezo-resistivos. Desta forma, estes diafragmas são comumente construídos em configuração de ponte de *Wheatstone*. Este tipo de resistor é extremamente sensível à temperatura (T), sendo então necessária a montagem de circuitos de compensação capazes de manter a ponte em equilíbrio quando (T) varia (FRADEN, 1996).

Dessa forma, os resistores Rr da Figura 2.3 formam a parte rígida do sensor, enquanto os Rf são montados no diafragma flexível. Quando o sensor sofre uma pressão (P) sobre a parte flexível, haverá uma deformação do diafragma e, portanto, uma variação da sua resistência. Esta variação causa um desequilíbrio da ponte de *Wheatstone*, o que gera uma diferença de potencial ($\triangle$V), em função da pressão aplicada.

Observando a Figura 2.3, nota-se a presença de dois termistores do tipo NTC (*Negative Temperature Coefficient*) montados em cada um dos ramos da ponte. Estes dispositivos fazem a compensação de temperatura, estabilizando o funcionamento do sensor de pressão.

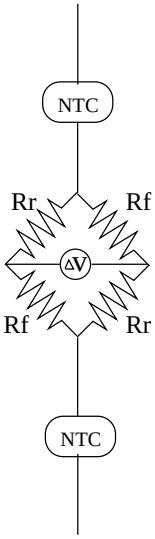

Figura 2.3. Esquema interno do circuito do sensor de pressão.

O encapsulamento final do sensor (Figura 2.4) possui outros elementos que não foram detalhados nesta breve exposição e são ditos com o estado da arte deste sensor,

assim afirma o fabricante em sua nota técnica (MOTOROLA, 2001) .

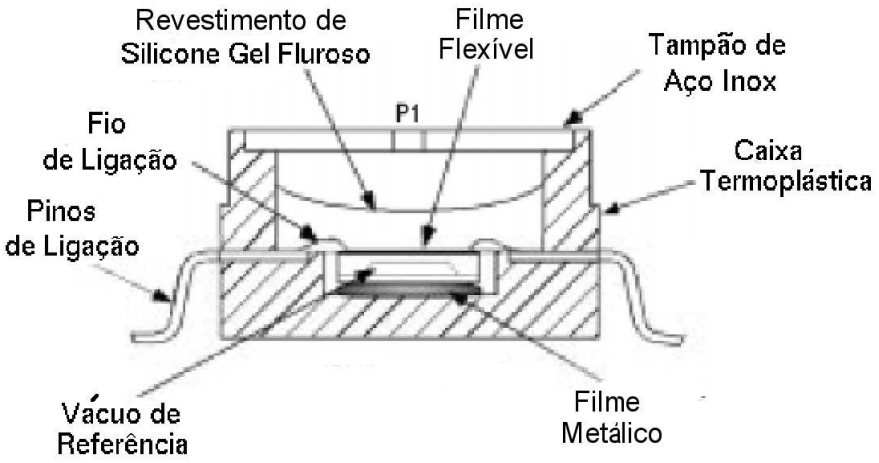

Figura 2.4. Esquema interno do sensor de pressão (MOTOROLA, 2001).

Descrição geral e caraterísticas técnicas

Para o sistema desenvolvido, foi utilizado o sensor da Motorola modelo MPX4115A. Este sensor piezo-resistivo mede a pressão de forma barométrica e utilizado pela PICOLLO (PICCOLO, 2003) e por Kahn (KAHN; KELLOG, 2003). Dentro deste sensor, circuitos integrados fazem o condicionamento do sinal elétrico da ponte de *Wheatstone* Os dados técnicos mais relevantes estão representados na Tabela 2.1 abaixo:

Tabela 2.1. Dados técnicos do sensor de pressão ($T_A = 25^oC$ e $V_{DD} = 5V$) (MOTOROLA, 2001).

Parâmetros	**Valores Típicos**
Faixa de Medida	15 a 115 kPa
Sensibilidade	45,9 mV/kPa
Tempo de Resposta	1 ms
Tensão de alimentação	4,85 a 5,35V
Temperatura de Operação	-40ºC a 125ºC

onde T_A é a temperatura ambiente e V_{DD} é a tensão de alimentação.

2.3.2 Acelerômetro

Os acelerômetros empregados neste trabalho medem a aceleração da aeronave ao longo dos eixos coordenados. Para a escolha deste sensor foram levados em consideração aspectos técnicos como: precisão, largura de banda, ruído característico, confiabilidade, além de outras características físicas. O acelerômetro adotado toma-se como base características de trabalhos pesquisados como o Picollo (PICCOLO, 2003), Projeto ARARA (NERIS, 2001), Micropilot (MICROPILOT, 2005) e Kahn (KAHN; KELLOG, 2003).

No sistema desenvolvido, foram utilizados os acelerômetros da Motorola, modelos MMA1200D e MMA2300D ((MOTOROLA, 2004a) e (MOTOROLA, 2004b)) que medem a aceleração de forma indireta, a partir de um sistema massa-mola.

Princípio de Funcionamento

Os acelerômetros utilizam o princípio da primeira Lei de Newton,

$$F_a = m_a \cdot \alpha_a,$$

A partir dessa expressão, obtém-se a força F_a resultante da aceleração α_a de um objeto de massa m_a conhecida. Existem diversas maneiras de determinar o módulo de uma força; o método comumente utilizado é medir o deslocamento de uma massa padrão, suspensa por molas. O sistema massa-mola-amortecedor está representado na Figura 2.5.

Quando o sistema massa-mola-amortecedor está em equilíbrio, as seguintes forças estão em ação:

$$F_a = m_a \cdot \alpha_a = b_a \cdot v_a + k_a \cdot z_a,$$

onde b_a é o coeficiente de viscosidade, v_a é velocidade da massa, k_a, a constante da mola e z_a, o deslocamento da massa , $\alpha_a = \dfrac{\partial^2 z_a}{\partial t^2}$ e $v_a = \dfrac{\partial z_a}{\partial t}$.

Fora da freqüência de ressonância do sistema, o efeito de amortecimento pode ser ignorado e então a aceleração pode ser dada por:

$$\alpha_a = \frac{k_a}{m_a} \cdot z_a$$

ou seja, α_a é a função do deslocamento da massa padrão. Um dos métodos utilizados para medir z_a é obtido pela aplicação de um sensor capacitivo. Este sensor mede a variação da capacitância entre uma placa fixa e uma placa móvel.

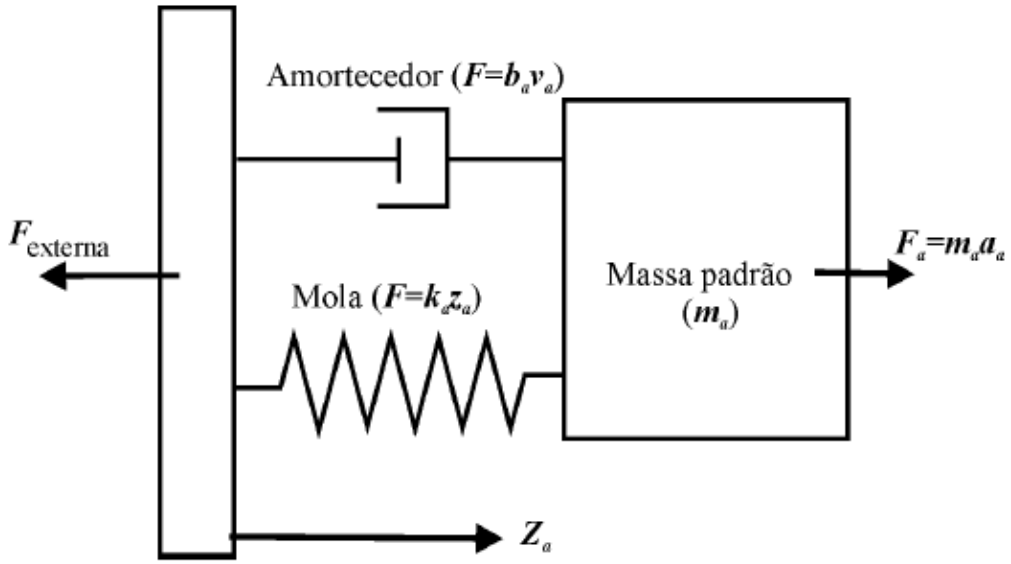

Figura 2.5. Esquema do sistema massa-mola-amortecedor

Os sensores utilizados podem ser usados para medir tanto a aceleração dinâmica (típica ou vibração) como estática (força de inércia, gravidade ou inclinação). Neste trabalho, o sensor é utilizado para medir a aceleração dinâmica do sistema.

Ambos os acelerômetros (MMA1200D e MMA2300D) são vistos como um sistema completo de medida de aceleração, assumindo a forma de um circuito integrado que combina o sensor mecânico, o circuito de condicionamento de sinal com um filtro passa-baixa de quarta ordem e um compensador de temperatura. O sistema massa-mola é do tipo MEMS (*Micro Eletronic Mechanical System*), junto ao qual atua um sensor de posição por diferença capacitiva (MOTOROLA, 2004a) e (MOTOROLA, 2004b).

A Figura 2.6 representa uma vista esquemática do sensor. O eixo sensível à aceleração está no plano do dispositivo. A massa-padrão é feita em silício, sendo sustentada por molas também de silício. Estas molas se contrapõem à força criada pela aceleração da massa-padrão. Como se vê na Figura 2.6, a deformação da estrutura pode ser determinada pela medida diferencial capacitiva. Esta diferença de capacitância produz uma tensão de saída proporcional à aceleração.

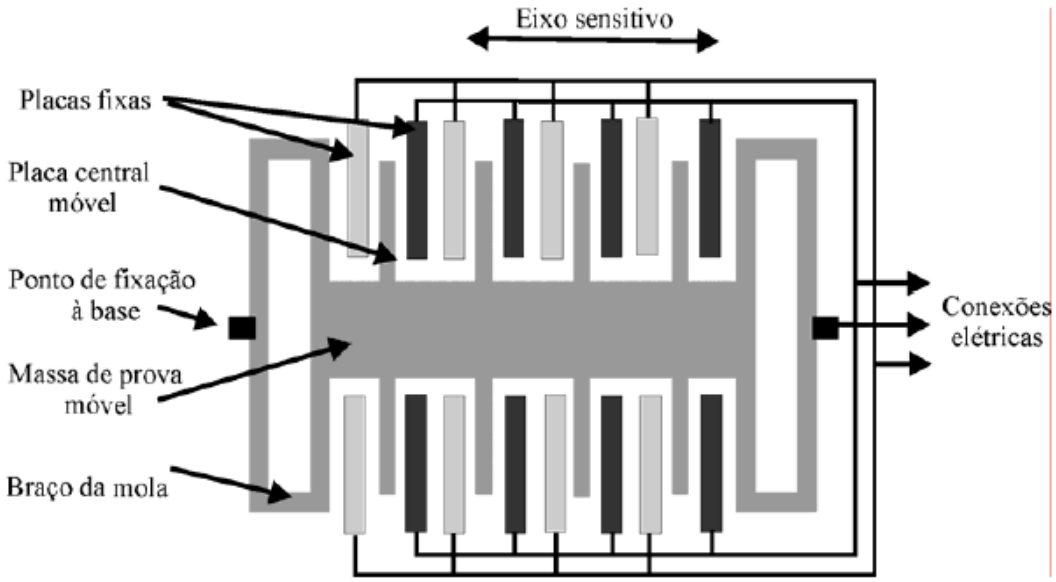

Figura 2.6. Vista superior esquemática do sensor.

Nas Figuras 2.7 e 2.8, tem-se o diagrama do sensor em repouso e de quando uma aceleração externa é aplicada ao dispositivo, respectivamente. Os valores capacitivos são dados pelos capacitores C_1 e C_2.

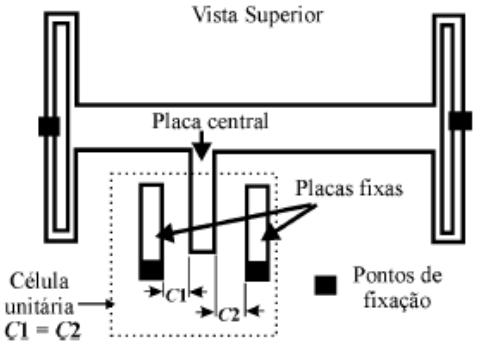

Figura 2.7. Diagrama do sensor em repouso.

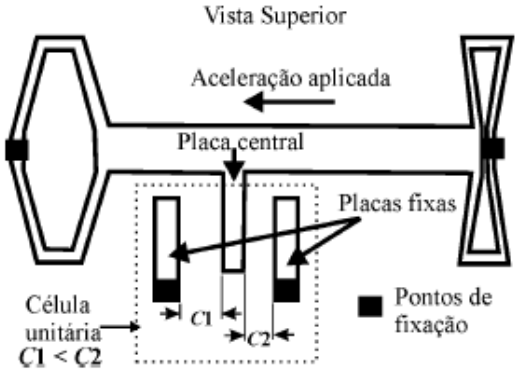

Figura 2.8. Diagrama do sensor quando uma aceleração é aplicada.

Descrição geral e caraterísticas técnicas

Os acelerômetros utilizados MMA1200D e MMA2300D são transdutores que medem a aceleração em um único eixo. Para o projeto foram usados dois MMA2300D para os eixos x e y e um MMA1200D para o eixo z uma vez que todos estão em um mesmo plano. Os dados técnicos mais relevantes estão apresentados na Tabela 2.2:

Tabela 2.2. Dados técnicos do acelerômetro ($T_A = 25^oC$, $V_{DD} = 5V$ e $acel. = 0g$.)

Parâmetros	**Valores Típicos**	
	MMA1200D	**MMA2300D**
Faixa de Medida	± 250 g	±250 g
Sensibilidade	$8mV/g$	$8mV/g$
Ruído	$110\mu V/\sqrt{hz}$	$110\mu V/\sqrt{hz}$
Ruído com Banda passante até 1kHz	2,8 mV rms	2,8 mV rms
Tensão de alimentação	4,75 a 7 V	4,75 a 7 V
Temperatura de Operação	-40 °C a 125 °C	-40 °C a 125 °C

onde g é a aceleração da gravidade local, foi determinada na seção 5.1.

2.3.3 Giroscópio

O giroscópico especificado é um sensor capaz de medir o movimento de revolução ou a velocidade angular de um objeto sob a influência de um campo de Coriolis. Mesmo em objetos não pontuais, caracterizados por uma distribuição de massa ao longo de sua estrutura, este tipo de sensor pode ser aplicado, uma vez que seu funcionamento independe da distância entre a sua posição de instalação e o centro de revolução do objeto. Neste trabalho, este sensor é utilizado para a obtenção dos ângulos de controle, ou seja, arfagem, guinada e rolagem da aeronave. Estes ângulos estão ilustrados na Figura 2.1.

Princípio de Funcionamento

O giroscópio de estado sólido é constituido por osciladores piezo-elétricos, também chamados de ressonadores piezo-elétricos. Em 1989, a empresa Murata lançou no mercado o modelo Gyrostar que usa um vibrador de estrutura triangular (ABE, 2005). Este dispositivo funcionava de forma totalmente diferente dos vibradores de estrutura quadrada que tradicionalmente se desenvolviam. Os osciladores triangulares caracterizam-se pelo fato de que as forças que levam à oscilação da estrutura peizoelétrica não têm que estar necessáriamente em ângulo reto em relação à força a ser detectada. Como resultado desta modificação de conceito, obteve-se uma substancial simplicação dos circuitos de transdução, além de um ajuste de freqüências muito mais simplificado e aumento da sensibilidade em relação aos giroscópios convencionais.

A partir de 1997, este tipo de giroscópio foi progressivamente reduzido de tamanho e, finalmente, colocado em um encapsulamento do tipo SMD (*surface-mountable-device*). A sua precisão também foi melhorada, o que expandiu muito as suas áreas de aplicação (ABE, 2005).

O giroscópio especificado utiliza um fenômeno físico denominado força de Coriolis (DOMELEN, 2006). O pêndulo de Foucault, assim chamado em referência ao físico

francês Jean Bernard Léon Foucault, é usado freqüentemente para pôr em evidência esta força. Conforme ilustrado na Figura 2.9, quando um pêndulo de massa (m) está ao mesmo tempo oscilando num dado plano e fazendo uma revolução em um outro plano qualquer, com velocidade angular ω, a força de Coriolis aparece e pode ser expressa como:

$$F_{Coriolis} = -2 \cdot m \cdot v_r \times \omega \tag{2.1}$$

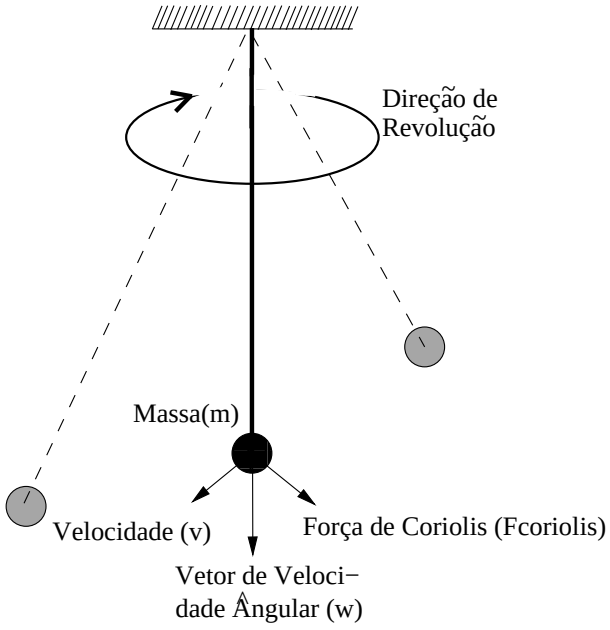

Figura 2.9. Pêndulo de Foucault

O movimento do pêndulo é determinado pela componente tangencial da força-peso ($m \cdot g \cdot sen\theta$), devido ao fato de a massa m ter sido tirada da sua posição de equilíbrio. Esta componente da força-peso é a única que contribui para o movimento, pois a componente radial é contrabalançada pela reação do apoio ou tração do fio. Em conseqüência, o movimento pendular terá sempre uma direção: a da componente tangencial da força-peso. As componentes radial e tangencial definem um plano no qual o pêndulo é obrigado a oscilar: não pode fugir dele, ainda que o chão esteja em movimento. Se este referencial é a Terra, o pêndulo sofrerá uma rotação em função da

força de Coriolis (DURRAN, 1993).

Um exemplo simplificado do uso da força de Coriolis para determinação da velocidade angular pode ser dado pela seguinte situação hipotética: quando o eixo de oscilação de um pêndulo é paralelo ao eixo de rotação da Terra, ou seja, o pêndulo encontra-se em um dos pólos terrestres, observa-se que seu plano de oscilação gira 360º em 24 horas, ou, exatamente, 15º por hora. Neste caso, o movimento da Terra e o movimento do pêndulo estão praticamente desacoplados. Quando se sai do pólo em direção ao Equador, o movimento do pêndulo sofre a ação do novo referêncial onde ele se encontra. Observando-se o pêndulo por alguns instantes, é possível identificar uma discordância na velocidade de giro do plano de oscilação em relação à Terra. Este efeito parece atrasar a velocidade aparente de deslocamento deste plano, considerando-se que, no Equador, o pêndulo não sofre rotação.

Sendo assim, para o giroscópio piezo-elétrico oscilante tem-se que sua freqüência de oscilação é ajustada pela geometria de suas barras cerâmicas peizoelétricas. Esta oscilação corresponde à oscilação do pêndulo usado no exemplo acima. Se este sistema vibrante, instalado em um objeto móvel, for excitado por uma dada velocidade angular ω, a força de Coriolis gerada, na direção perpendicular à direção de vibração das cerâmicas piezo-elétricas, dará origem a sinais elétricos proporcionais à variação da velocidade angular deste objeto.

Osciladores cerâmicos

A estrutura de um vibrador cerâmico é mostrada na Figura 2.10. Este dispositivo é composto por duas barras cerâmicas fixadas uma à outra. Estas barras estão presas entre si, em oposição de direção de polarização; assim, quando aplicada uma tensão externa, é produzida uma curvatura do conjunto, visto que uma se dilata enquanto a outra se contrai. Dessa maneira, as barras são levadas a oscilar.

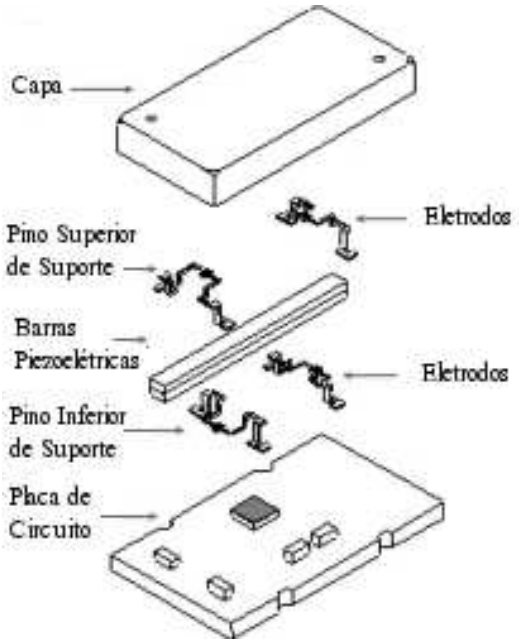

Figura 2.10. Oscilador cerâmico bimórfico (ABE, 2005)

No exemplo do vibrador cerâmico, os eletrodos servem ao mesmo tempo para aplicação da tensão de vibração das barras e para a detecção dos sinais induzidos pela ação da força de Coriolis sobre este conjunto oscilante.

Descrição geral e caraterísticas técnicas

O giroscópio Gyrostar da empresa Murata é um transdutor que mede a velocidade angular num único eixo (MURATA, 1999). Para esta aplicação, é necessária a utilização de três giroscópio, sendo um para cada eixo coordenado. Os dados técnicos principais estão apresentados na Tabela 2.3.

Tabela 2.3. Dados técnicos do giroscópio (MURATA, 1999).

Parâmetros	**Valores Típicos**
Faixa de Medida	$\pm 300^o/s$
Sensibilidade	$0,67mV/^o/s$
Tempo de resposta	20 ms
Valor de saída em repouso ($\omega = 0$)	1,35 V
Tensão de alimentação	2,7 a 5,5 V
Temperatura de operação	-5°C a 75 °C

2.4 ATUADORES

Os atuadores utilizados normalmente em aeromodelismo são servo-motores. Normalmente, se utiliza um potenciômetro acoplado ao seu eixo de rotação. A grandeza a ser realimentada pode ser a posição angular, a velocidade angular ou o sentido de rotação do seu eixo. O termo micro-servo-motor é aqui usado para designar um tipo específico de servo-motor muito utilizado em aplicações de aeromodelismo, como ilustra a Figura 2.11, que tem como principal objetivo o controle do posicionamento do seu eixo.

Figura 2.11. Exemplo de servo-motores (FUTABA, 2003).

O sinal de controle deste servo-motor impõe uma posição angular que o circuito de controle deve posicionar o eixo do motor. Um micro-servo-motor típico consiste em um motor DC, um conjunto de engrenagens de redução, um potenciômetro para a realimentação da posição e um circuito que, a partir do sinal do potenciômetro, coloca o motor na posição desejada.

Outra característica comum a este tipo de dispositivo é a existência de um limitador mecânico que restringe o seu campo de posicionamento angular. Os servo-motores utilizados foram da empresa Futaba modelo FP-S3003. Este tipo de servo-motor pode ser encontrado em diversos veículos controlados por rádio como aeromodelos, automodelos e nautimodelos. Existem outros fabricantes deste tipo de motor, como, por

exemplo, Hobbico, Hitec, JR, Tower Hobbies, Airtronics, Cox.

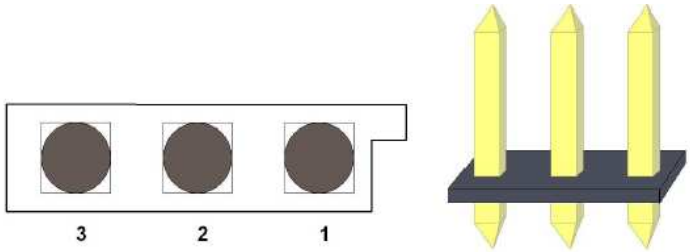

Figura 2.12. Conector Futaba padrão *Header*.

Tabela 2.4. Conexão para micro-servo-motores padrão Futaba.

Pino	Sinal
1	PWM
2	Vcc
3	Gnd

Variações em tamanho, torque, velocidade, tipos de posicionamento do eixo e forma de construção definem uma grande variedade de preços. O servo-motor da empresa Futaba, modelo S3003, pode posicionar seu eixo em até 180 graus. A ligação elétrica deste dispositivo é feita por meio de três pinos ilustrados na Figura 2.12. Um fio (fio vermelho no Futaba S3003) leva a alimentação de 5 V DC ao motor. Outro fio (fio preto no Futaba S3003) leva o sinal de terra. O terceiro fio (branco) leva o sinal de controle de posicionamento ao dispositivo. O sinal de controle considerado é um sinal periódico cuja largura é modulada (PWM- *Pulse Width Modulation*). A conexão de um micro-servo-motor é feita através de um conector fêmea de três pinos do tipo *header* (ver Figura 2.12). Este conector possui um chanfro de polarização para evitar conexões invertidas do motor. Este tipo de conector e pinagem são padronizados para todos os fabricantes de dispositivos de modelismo.

2.4.1 Controle de Posição Angular

Conforme citado anteriormente, o controle da posição angular do eixo é feito por um sinal PWM. Dois parâmetros caracterizam um sinal PWM: a freqüência e a razão do ciclo ($\frac{x}{t}$), onde t é o período (ver figura 2.13). A freqüência é a razão entre o número de ciclos ou repetições de um dado sinal e o tempo (ciclos por segundo). Para o Futaba S3003, a freqüência de trabalho é 33Hz ou 30 ms de período. A razão do ciclo é o percentual de tempo que um sinal está em nível alto (largura de pulso) em

relação ao período total.

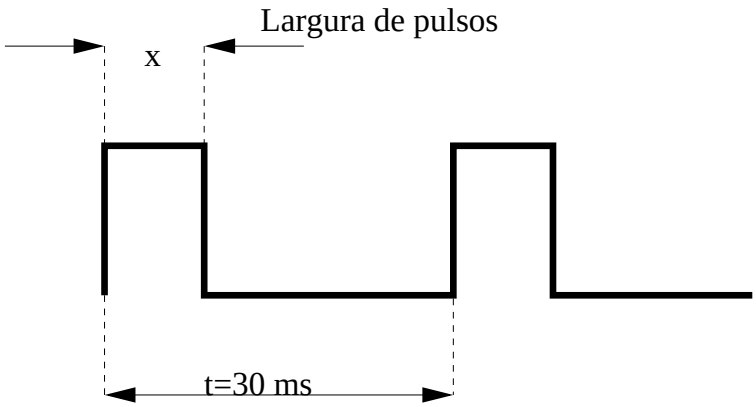

Figura 2.13. Sinal PWM para os micro-servo-motores Futaba 3003.

Neste tipo de servo-motor, a largura do pulso (nível alto) determina a posição do eixo. A freqüência de trabalho requerida por estes motores (33 Hz) do sinal não necessita de uma grande precisão, podendo variar em torno desse valor. Testes com o S3003 indicam que a menor largura de pulso que pode ser aplicada a este motor é de 0,1ms. Este sinal levará o eixo totalmente para a direita, enquanto um pulso com largura de 2,3 ms inverte o eixo todo para a esquerda. Um pulso com largura de 1,2 ms posiciona o eixo no centro de seu campo de ação. Deslocar o eixo de um extremo a outro demora cerca de 0,75s.

Tabela 2.5. Largura de pulso x posição do eixo.

Largura do Pulso (ms)	0,1	0,46	0,65	0,83	1,2	1,57	1,75	1,93	2,3
Posição (^{o})	-90	-60	-45	-30	0	30	45	60	90

A Tabela 2.5 relaciona alguns valores de largura do pulso com a posição angular para o micro-servo-motor Futaba S3003. Valores intermediários aos mostrados na Tabela 2.5 deslocarão a posição do eixo de forma proporcional. Tais valores podem ser aplicados a dispositivos similares de outros fabricantes.

CAPÍTULO 3

MÓDULOS COMERCIAIS E PESQUISAS CORRELATAS

Nos UAV, o módulo de aquisição de dados para navegação e telemetria pode ser considerado a "alma" deste equipamento. Este módulo fornece todas as informações para um vôo autônomo. Nos últimos anos, o barateamento dos componentes eletrônicos e os avanços tecnológicos de miniaturização, principalmente no que se diz respeito à capacidade de processamento de microcontroladores, estimularam diversas universidades e empresas a desenvolverem módulos comerciais para UAV (NERIS, 2001). Neste capítulo são descritos alguns destes módulos comerciais, além de outros trabalhos correlatos.

3.1 PESQUISAS CORRELATAS

Os projetos discutidos neste capítulo foram aqueles que, de certa forma, obtiveram algum destaque dentro dos projetos pesquisados e serviram de base para o projeto aqui apresentado.

3.1.1 Projeto ARARA

O projeto ARARA (Aeronaves de Reconhecimento Assistidas por Rádio e Autônomas) tem por base o uso de aeronaves em escala reduzida, para monitoramento ambiental. Este projeto teve como principal objetivo a substituição de aeronaves convencionais utilizadas para a aquisição de fotos aéreas para monitoramento e controle de áreas ecológicas e de grandes plantações (NERIS, 2001),(SOUZA, 1999) e (TRINDADE, 2002).

Este projeto foi desenvolvido pelo Laboratório de Alto Desempenho do Departamento de Ciências de Computação e Estatísticas da USP, campus São Carlos, em parceria com a EMBRAPA - CNPDIA.

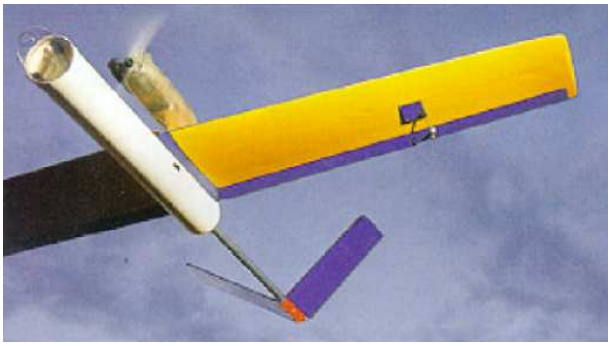

Figura 3.1. Aeronave UAV - Projeto ARARA (SOUZA, 1999)

O projeto ARARA foi dividido nas seguintes fases:

- Fase 1 - Fazer fotos aéreas de forma manual e radiocontrolada;
- Fase 2 - Monitoramento do vôo, colocação de sensores para a aquisição de dados de vôo ainda controlado por radiocontrole;
- Fase 3 - Inserção do módulo de controle autônomo de navegação e controle;
- Fase 4 - Fazer reconhecimento de imagens em tempo-real para navegação de forma autônoma.

3.1.2 Projeto AURORA

O projeto AURORA (*Autonomous Unmanned Remote Monitoring Robotic Airship*), desenvolvido no Laboratório de Robótica e Visão do Centro de Pesquisas Renato Archer (CENPRA), antigo Instituto de Automação do Instituto Nacional de Tecnologia da Informação (LRV/IA/ITI), em Campinas, teve como objetivo desenvolver um UAV do tipo dirigível como plataforma de aquisição de fotos, dados metereológicos e da biodiversidade.

A aeronave do projeto AURORA conta com um série de dispositivos que possibilitam a navegação autônoma em todas as fases de operação incluindo diagnóticos de falhas, avaliação dos sensores em tempo real e o replanejamento das tarefas a serem feitas. O dirigível escolhido pela a equipe é o modelo AS800 da *Airships*, ilustrado na Figura 3.2.

Desta forma, os subsistemas de navegação e controle da missão são responsáveis pela aquisição de dados e pelo controle dos atuadores. O hardware do dirigível é composto por um computador IBM/PC, microcontroladores, sensores e atuadores.

Figura 3.2. Dirigível AS800 do Projeto AURORA (MAETA, 2001).

Para o sistema de comunicação do dirigível com a estação-base, são utilizados dois enlaces de comunicação comercial. Um é responsável pela comunicação dos dados e comandos durante todo o vôo e o outro, pela transmissão das imagens adquiridas pela câmera embarcada (ELFES; MAETA, 1998), (ELFES et al., 1998), (MAETA, 2001), (RAMOS et al., 1999), (RAMOS et al., 1998) .

3.1.3 Projeto AEROSONDE

No projeto AEROSONDE, foi construída uma aeronave autônoma de longo alcance, podendo fazer trajetos de milhares de quilômetros e com duranção de alguns dias em funcionamento, para monitoramento ambiental (AEROSONDE, 2001). Esta aeronave pode ser adaptada para outras aplicações, possuindo um pequeno comparti-

mento para carga útil. O seu principal objetivo é o reconhecimento metereológico e do ambiente oceânico, podendo até investigar furacões e tempestades, conforme ilustrado na Figura 3.3.

Figura 3.3. Aeronave de Longo Alcance - AEROSONDE (AEROSONDE, 2001).

Esta aeronave é capaz de decolar e pousar de forma completamente autônoma. Mas, em caso de mau funcionamento, existe um mecanismo que possibilita o controle da aeronave por radiocontrole (R/C). Este enlace de comunicação com a estação-base pode chegar a 150 km (AEROSONDE, 2001) e (RAMOS et al., 2001). No presente projeto (AERO-UFBA), também foi prevista a função de vôo rádio-assistido; assim, caso haja um mau funcionamento do sistema de navegação, a aeronave pode ser resgatada por radiocomando.

3.1.4 Projeto AVATAR

O projeto AVATAR (*Autonomous Vehicle Aerial Tracking and Retrieval*) foi desenvolvido na Universidade da Califórnia do Sul (*University of Southern California*) pelo Laboratório de Pesquisa em Robótica (*Robotics Research Laboratory*). O veículo desenvolvido por este grupo foi um helicóptero, conforme ilustra a Figura 3.4.

Inicialmente, foram utilizados dois processadores 68332 da Motorola, posteriormente substituídos por um computador IBM/PC que executa o controle, a navegação

Figura 3.4. Aeronave do projeto AVATAR (AVATAR, 2004)

e o processamento de imagens. Além disso, o *hardware* embarcado conta ainda com um módulo GPS e um sistema de sensores inerciais para a navegação e telemetria. O computador de bordo recebe os dados de altitute, latitude, longitude, velocidade horizontal e vertical, ângulo de guinada e taxa de variação da velocidade e decide sobre a forma da manobra a ser efetuada (AVATAR, 2004) e (MONTGOMERY, 2000).

3.2 MÓDULOS COMERCIAIS

Nos trabalhos pesquisados na literatura, muitas vezes o detalhamento técnico é omitido ou muito resumido. Para suprir esta deficiência no desenvolvimento técnico deste projeto, foram usados os parâmetros de equipamentos comerciais e as referências utilizadas por outros grupos de pesquisa. Em seguida, serão discutidos os modelos comerciais que contribuíram para o projeto, desde a seleção dos sensores até a arquitetura final desenvolvida.

3.2.1 U-NAV

U-NAV é um sistema de controle modular desenvolvido pela empresa Silvertone Electronics (MARQUETTE, 2004). Este sistema permite o controle de UAV e RPV. O

(a) Módulo PDC20.

(b) Módulo PDC10.

(c) Módulo PDC25.

Figura 3.5. Módulos de Controle U-NAV (PDC20, PDC10, PDC25) (MARQUETTE, 2004).

sistema U-NAV é capaz de transformar um aeromodelo padrão em um UAV com controle de altitude e velocidade combinado com um sistema de geoposicionamento. Os três módulos deste sistema foram projetados para serem utilizados de forma independente ou integrada sobre o comando de um receptor R/C padrão. Quando combinados, formam um sistema completo de piloto automático.

Os três módulos são os seguintes:

- PDC20 - Controlador de Altitude;
- PDC10 - Unidade de Orientação - GPS;
- PDC25 - Controle de Velocidade.

Apesar do seu baixo custo em relação aos outros módulos comerciais, conforme Tabela 3.1, este sistema apresenta a desvantagem de não pôr à disposição do usuário as informações de vôo. Isto o torna inviável para aplicações onde se pretende criar e testar vários tipos de controladores e algoritmos de navegação.

Tabela 3.1. Sistema U-NAV.

Módulo	**Preço (US$)**
PDC10	199,00
PDC20	249,00
PDC25	249,00

3.2.2 Micropilot - MP1000/MP2000

A Micropilot é fabricante de diversos controladores destinados a aplicações UAV (MICROPILOT, 2005). Dentre eles, destacam-se as séries MP1000 e MP2000 que foram projetadas para controlar um grande número de aeromodelos do tipo escala[1]. O piloto automático é composto por um sistema de manutenção de altitude e velocidade. O sistema fornece ainda dados de coordenação de curvas (raio de curvatura do padrão de manobras), GPS, um sistema de pouso e decolagem automáticos e um conjunto de sensores para o controle total da aeronave. Existem ainda trinta e dois controladores PID (Proporcional-Integral-Derivativo) ajustáveis pelo usuário, que trabalham de forma independente.

Os sistemas da Micropilot também possuem a capacidade de telemetria, que pode enviar um pacote de informações a cada 200 ms para uma estação-base. Este sistema de telemetria é composto por 5 pacotes diferentes de informação, de tal modo que os 5 pacotes podem ser lidos em um segundo. A Tabela 3.2 apresenta os valores de cada bloco funcional e a Figura 3.6 mostra o sistema apresentado.

Tabela 3.2. Tabela dos valores do sistema MP1000/MP2000 (MICROPILOT, 2005).

Descrição	**Valor (US$)**
MP1100 - Módulo Piloto Automático com receptor, antena e conector de GPS	1800,00
MP1100UAV - Sistema completo para vôo	4300,00
MP1100VID - Sistema completo para vôo com link de vídeo	5800,00
MP1100UP - Módulo de upgrade do MP1000 para MP2000	4000,00
MP2000 - Módulo Piloto Automático com receptor, antena e conector de GPS	5000,00
MP2000CON - Módulo de conexão	75,00
MP2000AGL - Módulo do sensor ultra-sônico	500,00
MP2000SV - Módulo de expansão de 8 servos	300,00
MP2000ANT - Antena de GPS	60,00
MP2000UAV - Sistema completo para vôo	7500,00
MP2000VID - Sistema completo para vôo com link de vídeo	9000,00

Alguns conceitos deste sistema foram aproveitadas para o sistema proposto, tais

[1] réplicas em escala de um avião.

como: módulo de expansão de servo-motores e concepção de uma arquitetura modular. A série MP1000/MP2000 também foi usada como orientação na escolha dos sensores, assim como a faixa de resposta dos mesmos. Considerando uma evolução natural do projeto AERO-UFBA, os sistemas da Micropilot podem continuar sendo fonte de inspiração.

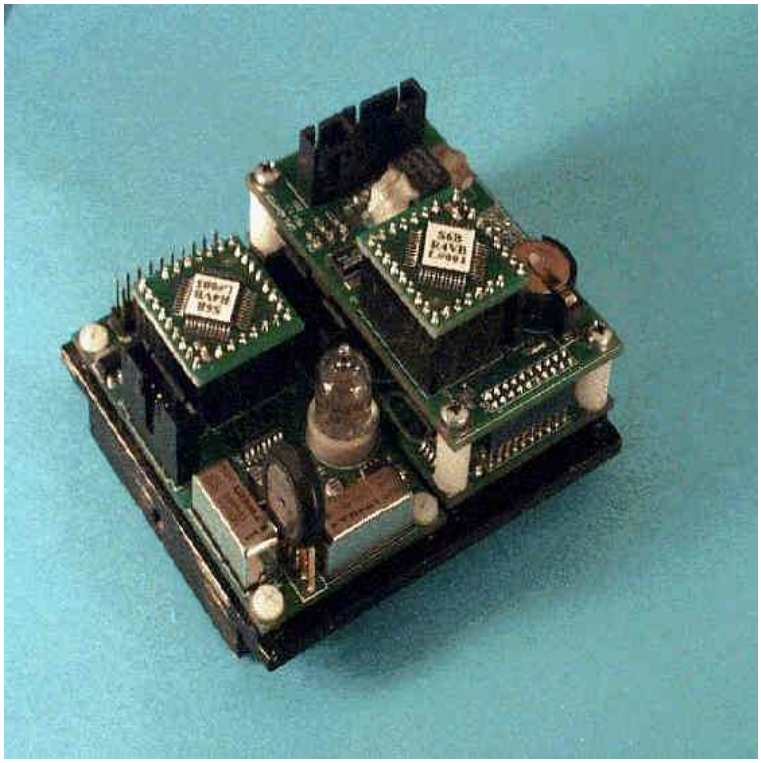

Figura 3.6. Sistema Micropilot MP1000 Fonte:(MICROPILOT, 2005)

3.2.3 Piccolo

O sistema Piccolo, fabricado pela Cloud Cap Technology, possui todas as funcionalidades esperadas para um sistema de vôo autônomo (PICCOLO, 2003). Ele não está dividido em módulos básicos e avançados, é formado por um único sistema relativamente pequeno e leve, quando comparado aos módulos de outros fabricantes. Este sistema é capaz de equipar uma grande quantidade de aeromodelos (ver Figura 3.7).

O Piccolo vem equipado com todos os sensores de navegação inercial e dois sensores barométricos. Na Tabela 3.3, tem-se o preço deste sistema.

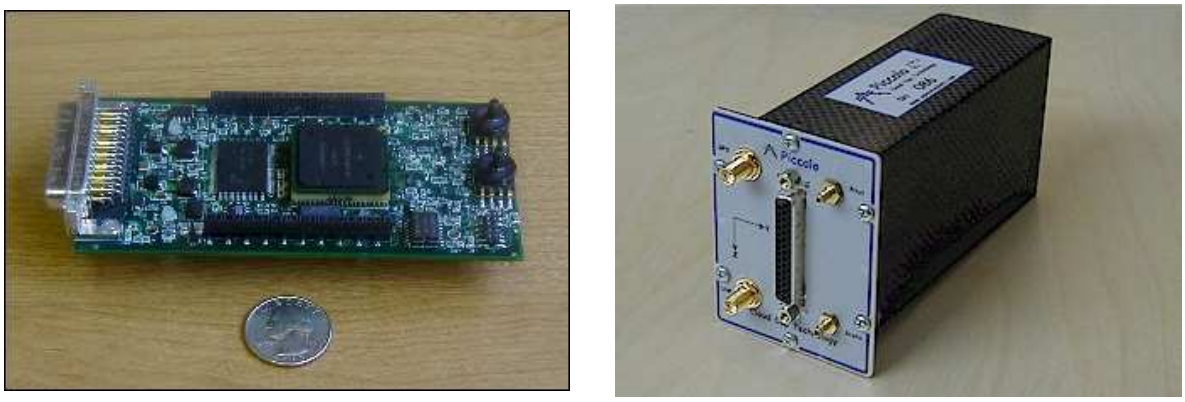

Figura 3.7. Sistema Piccolo de controle UAV (PICCOLO, 2003)

Tabela 3.3. Tabela dos valores do sistema Piccolo (PICCOLO, 2003).

Descrição	Valor (US$)
PiccoloPlus Autopilot- Módulo piloto automático com receptor, antena e módulo de GPS	7000,00
Piccolo II Autopilot - Módulo piloto automático com receptor, antena, módulo de GPS e pacotes de sensores adicionais	8.600,00
Piccolo Ground State Kit	8.600,00
Piccolo Ground State Kit Portable	Sob encomenda

O portfólio do fabricante mostra que este sistema é utilizado por diversos grupos de pesquisas, tais como: *Georgia Tech Research Institute* (GTRI), *Idaho State University*, *Johns Hopkins Applied Physics Lab*, *Massachusetts Institute of Technology (MIT)*, *Mississipi State University*, *North Carolina State University*, *Oklahoma State University*, *Penn State University*, *Texas A&M University*, *Aerospace Vehicles Systems Institute*, *University of Arizona*, *University of Arizona Aerial Robotics Club*, *UC Berkeley (ONR/AINS STTR program)*, *UCLA*, *(ONR/AINS STTR program)*, *University of Colorado - Boulder*, *University of Illinois*, *University of Pennsylvania*, *US Air Force Academy e US Naval Academy.* Algumas delas organizam campeonatos anuais que utilizam como base o sistema Piccolo (HILL et al., 2004).

É importante destacar que arquitetura do Piccolo serviu como inspiração para este

trabalho, como se verá no Capítulo 4. Um grande número dos sensores escolhidos são os mesmos usados no sistema Piccolo. Pode-se dizer que o sistema aqui proposto traz consigo muitos aspectos similares a um dos sistemas de navegação inercial comercial bastante utilizado por outros grupos de pesquisa.

3.3 COMENTÁRIOS DO CAPÍTULO

Neste capítulo, foram discutidos alguns dos módulos comerciais existentes no mercado que contribuíram para a concepção da arquitetura deste projeto. Foram discutidos também outros desenvolvimentos de UAV, tanto no Brasil quanto no mundo.

Com a intenção de mostrar a diversidade destes veículos não-tripulados, foram abordados os três tipos mais comuns de aeronave (aviões, helicópteros e dirigíveis).

O próximo capítulo abordará as características do sistema de navegação inercial aqui proposto, bem como os sensores, atuadores e funções eletrônicas desempenhadas pelo sistema embarcado.

CAPÍTULO 4

PLATAFORMA DESENVOLVIDA

Neste capítulo, é abordada a arquitetura desenvolvida no projeto, assim como os módulos para o sistema de navegação inercial e telemetria que são: o módulo de aquisição de dados, o módulo dos sensores e o módulo de interface e controle. Ainda neste capítulo, são detalhados a lógica de funcionamento desses módulos, os filtros analógicos aplicados aos sinais dos sensores, a chave eletrônica do modo de operação e o protocolo de comunicação.

4.1 ARQUITETURA

Para o atendimento dos recursos necessários para um vôo autônomo (ver Seção 2.2), deste projeto. O sistema desenvolvido seguiu algumas importantes características que foram encontradas em sistemas comerciais ((MICROPILOT, 2005) e (PICCOLO, 2003)). Além disso, os projetos relatados na literatura também serviram de base para este projeto (ver Capítulo 3). Suas características são:

- sistema modular, que facilita a montagem e a manutenção;
- possibilitar a expansão de dispositivos, aumentando o uso do sistema;
- possuir diferentes tipos de interfaces de comunicação (paralela ou serial);
- colocar à disposição do projetista vários tipos de atuadores (servo-motores, motores de passo e elementos biestáveis (Liga/Desliga));
- possibilitar o vôo autônomo ou por R/C (radiocontrole);

O módulo de interface e controle é capaz de gerenciar a aquisição dos dados, gerar os sinais dos atuadores e comunicar estas informações à estação-base através de um protocolo.

Observando a Figura 4.1, é possível visualizar dois canais de comunicação estação-base e a aeronave. O primeiro, mais a esquerda no canto superior, é responsável pelo vôo por R/C, pois este é o canal habitualmente utilizado para as manobras de pouso e decolagem do UAV. Este canal também poderá ser usado em casos de mau funcionamento do sistema de vôo autônomo, uma vez que não interfere no sistema autônomo de vôo do UAV. O segundo canal de comunicação permite ao operador na estação-base ou na unidade de controle e navegação (UCN) ajustar os parâmetros do sistema de controle, além de mudanças na trajetória do UAV em pleno vôo.

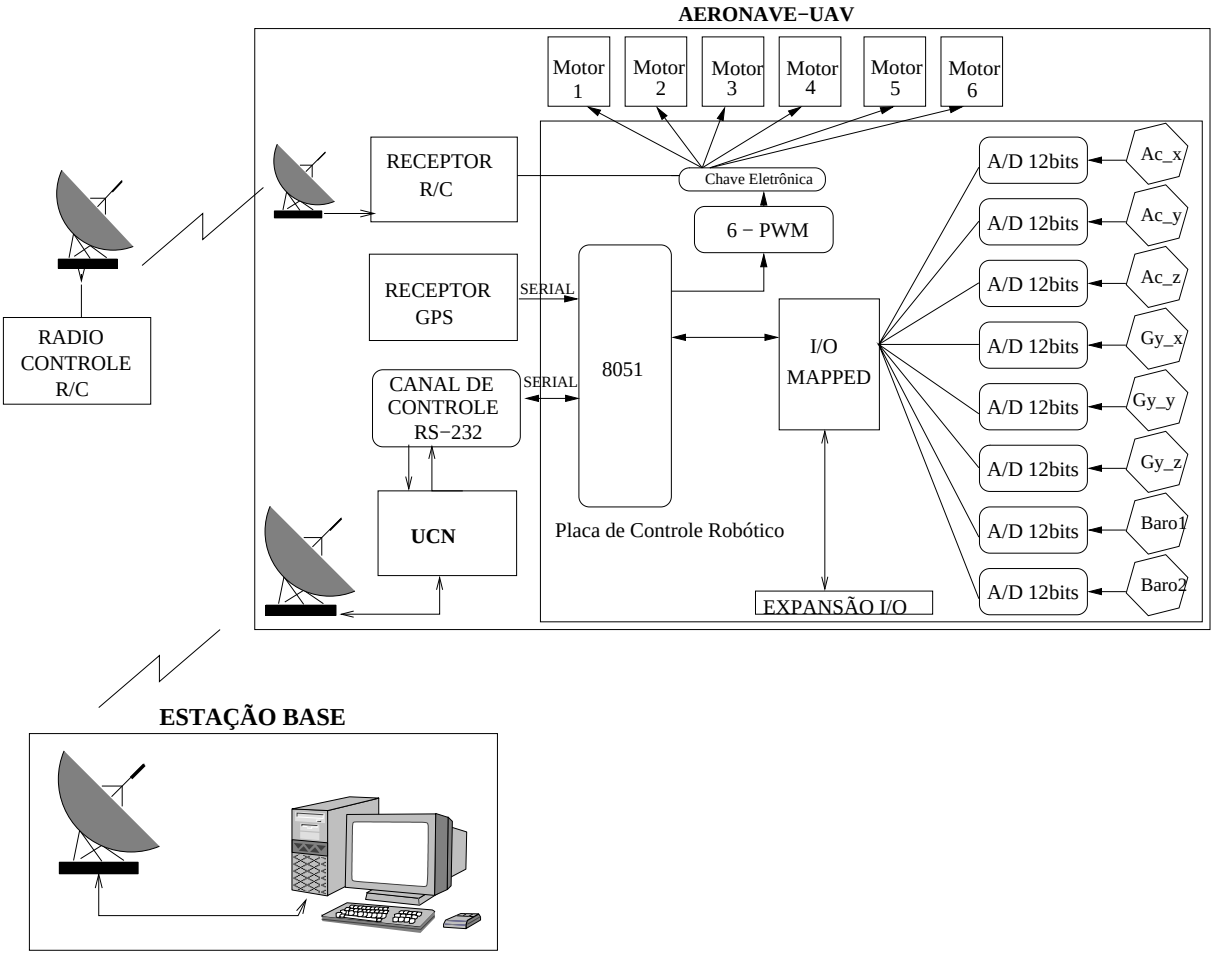

Figura 4.1. Arquitetura do sistema UAV-AERO-UFBA.

Na arquitetura Piccolo a unidade central utilizada é o microcontrolador da empresa Motorola modelo MPC555; neste projeto foram usados microcontroladores da empresa Intel, famíla MSC-51, mais conhecidos pela designação de 8051.

O 8051 é um microcontrolador de pequeno porte e baixo custo, facilmente encontrado no mercado brasileiro e com uma vasta documentação disponível ((CORPORATION, 1994), (AYALA, 1991), (NICOLOSI, 2000), (GIMENEZ, 2002) e (SILVA, 1998)). Este *chip* pode ser considerado o "coração" do *hardware* sendo responsável pela aquisição, processamento e transmissão das informações de vôo (sensores, atuadores e GPS) usando para isso o segundo canal de comunicação. O fato de o algoritmo de controle e navegação não se encontrar no *firmware* deste microcontrolador torna o sistema mais flexível. Desta maneira, os algoritmos de controle da UCN serão processados externamente ou em uma outra placa microprocessada ou na própria estação-base em solo. Assim, a utilização deste *hardware* em outras aplicações de robótica móvel também é possível.

Em um desenvolvimento futuro, a placa microprocessada da UCN pode ser substituída, como, por exemplo, uma placa Stargate (STARGATE, 2004). Esta placa possui como sistema operacional o RTLinux (*Real-Time Linux*) desenvolvido especialmente para gerenciar sistemas de tempo-real. Outros autores já utilizaram este *hardware* em suas propostas de sistema UAV (HALL, 2001). O Stargate poderia comunicar-se com o *hardware* desenvolvido através do canal de comunicação serial e com a estação-base através da sua saída *ethernet* utilizando uma rede *wireless*. Desta forma, a estação-base apenas monitora os dados, fazendo atualização dos mesmo quando necessário.

No futuro, o *hardware* desenvolvido poderá vir a ser utilizado em uma grande quantidade de aeromodelos comerciais, uma vez que a modelagem matemática do controle da aeronave não se encontra embarcada e sim na estação-base ou na UCN. Assim, é na UCN que se concentram todas as informações e processamento de controle da aeronave. Esta flexibilidade na arquitetura resulta na possível aplicação de outros dispositivos

processados, tais como: *HandHeld* embarcado, o kit Stargate (citado anteriormente), ou até mesmo a utilização de células do sistema de telefonia móvel (telefones celulares) para a comunicação de dados entre a estação-base e o *hardware* desenvolvido.

Outra característica do sistema desenvolvido, como mencionado no ínicio deste Capítulo, é a sua modularidade, isto é, o sistema é composto pelos seguintes módulos:

- **Módulo de Aquisição** - Este módulo é responsável pela conversão de sinais analógicos em digitais. Para atender à demanda de sensores são utilizados dois destes módulos, podendo os mesmos serem utilizados de forma independente do sistema, como placas de aquisição de dados para uso geral;
- **Módulo de Sensores e filtros analógicos**- Neste módulo concentram-se os sensores do sistema de navegação inercial (acelerômetros, giroscópio e sensores de pressão barométricos), bem como os filtros analógicos de condicionamento de sinal destes sensores;
- **Módulo Interface**- Este módulo é responsável pela aquisição dos dados, montagem do protocolo de comunicação, chaveamento do modo de vôo (R/C ou autônomo), interface com os atuadores e pela expansão de dispositivos de Entrada/Saída (E/S) endereçáveis de 8 bits.

4.2 MÓDULO DE AQUISIÇÃO DE DADOS

Foram usados conversores de sinal Analógico/Digital (A/D) de 12 bits da empresa BURR-BROWN modelo ADS1286, ilustrados na Figura 4.2 (INSTRUMENTS, 1998). Estes dispositivos são utilizados para adquirir os dados dos 8 sensores do sistema. Dentro da arquitetura desenvolvida, os valores lidos nos sensores são adquiridos e convertidos diretamente, obtendo assim uma característica de sincronismo. Isto é possível devido a dois fatores:

- O pulso de início de conversão e relógio são acionados ao mesmo tempo para

todos os conversores A/D.

- O emprego da técnica de E/S mapeada na memória consiste em acessar dispositivos (conversor A/D, buffers, LCD (*Light Cristal Display*) e outros) como se fossem endereços de memória. Assim, o microcontrolador seleciona exclusivamente um único endereço de cada vez para efetuar a escrita e a leitura.

Figura 4.2. Conversor A/D de 12 bits (INSTRUMENTS, 1998).

O conversor ADS1286 coloca à disposição a informação convertida numa porta serial, ou seja, os bits são lidos, um a um, num mesmo pino do *chip* e, posteriormente, "monta-se" o valor lido em uma palavra de n *bits*. Para que esta informação possa ser convertida e em seguida extraída, é necessária a utilização de dois sinais de controle do conversor A/D, um pulso para inicializar a conversão($\overline{CS}/SHDN$), seguido de n pulsos de relógio (DCLOCK) para a extração do dado. Observando-se a Figura 4.3, tem-se o diagrama de estado dos sinais de controle.

Para efetuar uma aquisição de dados, segue-se o procedimento:

1º) $\overline{CS}/SHDN = 0$

2º) Dois pulsos de relógio (DCLOCK) para amostrar o sinal (T_{SMPL})

3º) Doze pulsos de relógio para conversão do sinal (T_{CONV})

4º) Doze pulsos de relógio para a leitura do sinal (T_{DATA})

5º) $\overline{CS}/SHDN = 1$

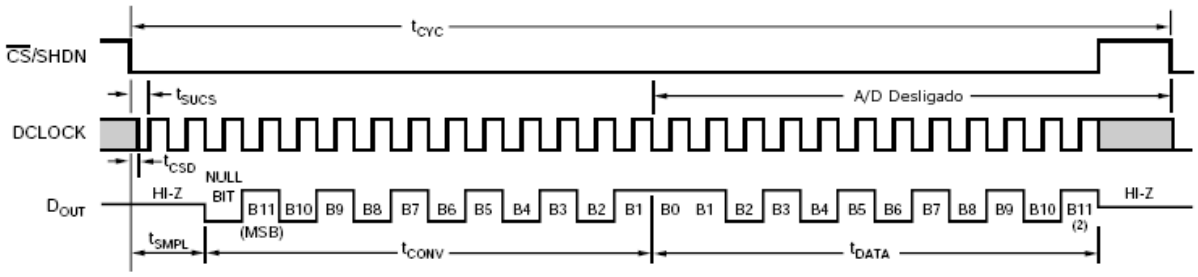

Figura 4.3. Diagrama de estados dos pulsos de controle (INSTRUMENTS, 1998).

O conversor utilizado pode trabalhar com uma taxa de aquisição de até 20 khz (50 μs). Cada módulo de aquisição é composto por 4 canais, podendo funcionar conectado diretamente ou com um microcontrolador ou usando a porta paralela de um computador (LPT) como uma placa de aquisição de dados autônoma. Este módulo permite, ainda, a conexão de mais um outro módulo igual, formando assim uma placa de aquisição de 8 canais. As microchaves (*dipswitches*) na placa - o bloco central da Figura 4.4 - possibilitam a configuração do modo de expansão.

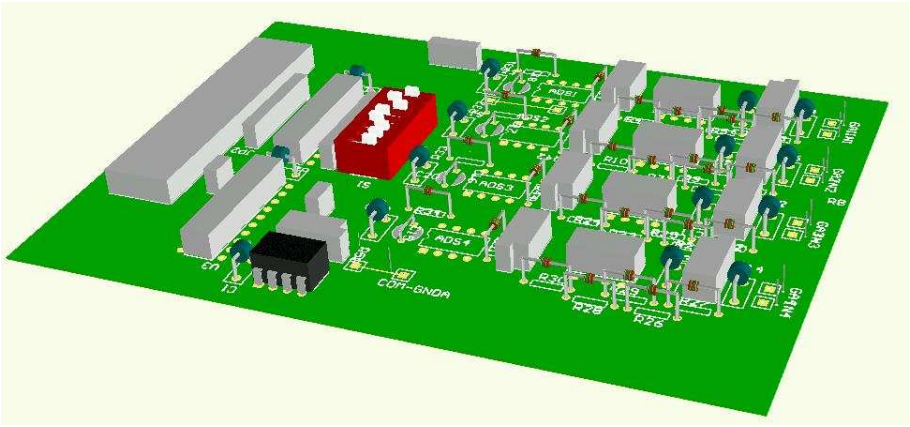

Figura 4.4. Placa de aquisição com 4 canais.

4.2.1 Quantização

O ADS1286 trabalha com tensão máxima de entrada de 5 V, tendo resolução de 12 bits. Assim, o passo de quantização, ou seja, a resolução do A/D, é dado por:

$$V_{ref} = \frac{5}{2^{12}} = \frac{5}{4096} = 1,2207mV \tag{4.1}$$

Desta forma, um valor lido (V_{lido}[V]) pelo conversor será o produto do valor de referência (V_{ref}=1,2207 mV) pelo valor convertido ($Valor_{conv}$).

$$V_{lido}[V] = 1,2207 \cdot 10^{-3} \cdot Valor_{conv} \tag{4.2}$$

Como apresentado no Capítulo 2 (ver Tabelas 2.1, 2.2, 2.3), a sensibilidade do giroscópio fica muito aquém da resolução do A/D. Este problema foi resolvido pela adição de um circuito de condicionamento de sinal e ajuste de ganho (ilustrado na Figura 4.5). Desta forma, um ganho de duas vezes aplicado no sinal do giroscópio é suficiente para que ele possa fornecer a informação requerida. Este mesmo circuito permite, ainda, um ajuste da tensão de *offset* do sensor, caso seja necessário. Maiores detalhes do circuito se encontram no apêndice A.

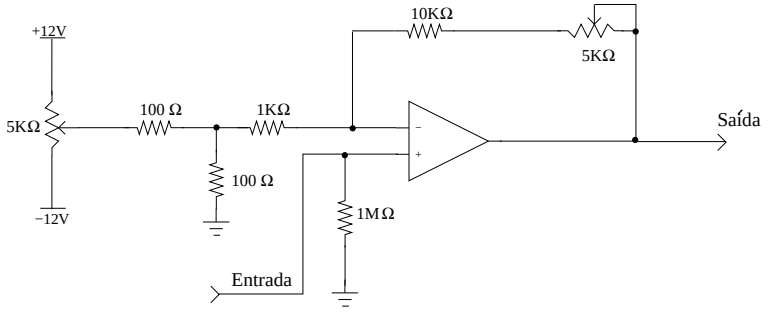

Figura 4.5. Circuito de condicionamento de sinal e ajuste de ganho.

4.3 MÓDULO DOS SENSORES

A seguir, serão mostrados os circuitos desenvolvidos, os filtros analógicos e o equacionamento necessário para leitura da informação de cada sensor.

4.3.1 Pressão

O circuito desenvolvido para o sensor de pressão é apresentado na Figura 4.6. A saída do sinal analógico é ligada na entrada do circuito de condicionamento e ajuste de ganho, mostrado anteriormente na Figura 4.5.

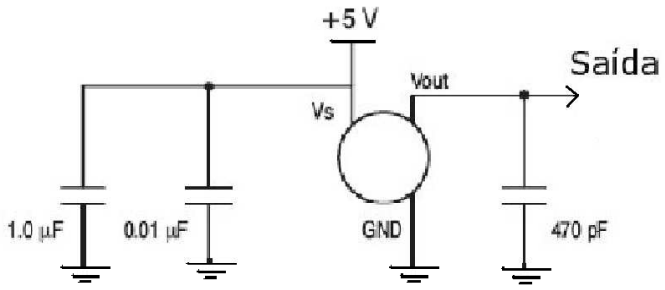

Figura 4.6. Circuito do sensor de pressão.

Cálculo da altura em função da pressão

Para efetuar o cálculo da altitude usando sensor de pressão barométrica, utiliza-se a equação 4.3 de ajuste, fornecida pelo fabricante (MOTOROLA, 2001). A pressão (P) é dada em unidades de kPa.

$$V_{out} = 0,045 \cdot P - 0,475 \tag{4.3}$$

Substituindo a equação 4.2 em 4.3, tem-se a pressão (P) em função do valor con-

vertido ($Valor_{conv}$):

$$V_{out} = 0,045 \cdot P - 0,475 = 1,2207 \cdot 10^{-3} \cdot Valor_{conv} \Rightarrow P = \frac{1,2207 \cdot 10^{-3} \cdot Valor_{conv} + 0,475}{0,045}$$

$$P = 0,0271267 \cdot Valor_{conv} + 10,5556 \tag{4.4}$$

A relação entre pressão (P) em kPa e altitude (Alt) em metros é dada pela equação 4.5, usando o sistema internacional (SI).

$$R = \frac{P_{mar}}{T_o \cdot \rho_o} \tag{4.5}$$

$$Alt = \frac{T_o}{\lambda}\left[1 - \left(\frac{P}{P_{mar}}\right)^{\lambda \cdot R}\right] \tag{4.6}$$

onde, P_{mar} = 101,29 kPa é a pressão ao nível do mar, R é a constante do ar para a Lei Geral dos Gases, λ=0,0064997 K/m a constante de decaimento térmico, ρ_o=1,2251 kg/m^3 a densidade volumétrica do ar e T_o é a temperatura ambiente medida em Kelvin. Substituindo-se as equações 4.4 e 4.5 em 4.6, tem-se:

$$Alt = \frac{T_o}{0,0064997}\left[1 - \left(\frac{0,0271267 \cdot Valor_{conv} + 10,5556}{101,29}\right)^{0,0064997 \cdot \frac{101,29}{T_o \cdot 1,2251}}\right] \tag{4.7}$$

4.3.2 Acelerômetro

O circuito montado para acelerômetro sugerido pelo fabricante Motorola está ilustrado na Figura 4.7. O sinal de saída deste circuito é aplicado ao circuito de condicionamento e ajuste de ganho similar ao ilustrado na Figura 4.5.

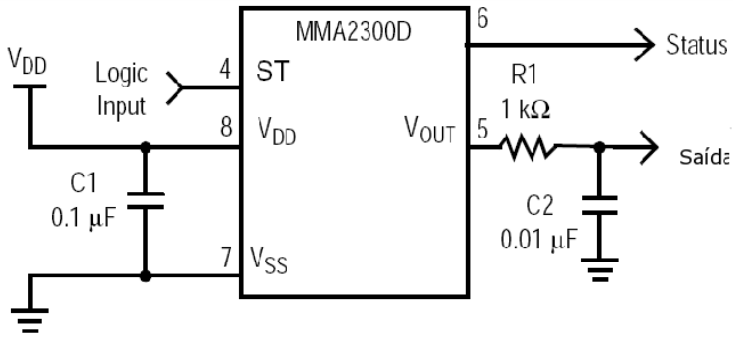

Figura 4.7. Circuito dos acelerômetros.

Cálculo da aceleração inercial

Para efeito de calibração dos sensores de aceleração, é necessário determinar qual é a sensibilidade que o módulo de aquisição é capaz de medir.

Na prática, o valor real da sensibilidade dos sensores é medido da seguinte forma:

1º) O eixo sensível do acelerômetro é apontado para o solo, sendo então lida a sua tensão de saída (U_{+g}); nesta posição, o sensor fica sensível apenas à força da gravidade, no sentido dito positivo;

2º) O eixo sensível do acelerômetro é então rotacionado a 180º em relação ao solo, e sua tensão de saída (U_{-g}), corresponde à gravidade, no sentido negativo;

3º) Utilizando essas leituras, determina-se a real sensibilidade do sensor, sendo dada pela média dos dois valores:

$$sensibilidade = \frac{U_{+g} - U_{-g}}{2} \left[\frac{V}{g}\right] \tag{4.8}$$

A Tabela 4.1 apresenta a sensibilidade real dos três acelerômetros, lembrando que g denota a aceleração da gravidade. A determinação do valor absoluto de g é feita na Seção 5.1

Tabela 4.1. Valor real da sensibilidade dos acelerômetros.

Acelerômetro	U_{+g}(V)	U_{-g}(V)	Sensibilidade [V/g]
$Eixo_x$	2,5412	2.5241	0,0085
$Eixo_y$	2,4906	2,4724	0,0091
$Eixo_z$	2,5075	2,4903	0,0086

O valor da aceleração inercial medida nos três eixos $\ddot{z}_p(t)(p = 1, \ldots, 3)$ é dado por:

$$\ddot{z}_p(t) = \frac{\left(refA_{inicial_p} - Valor_{lido_p}\right) g}{sensibilidade_p} \tag{4·9}$$

sendo $refA_{inicial}$ o valor de tensão do sensor em repouso e a unidade de aceleração é m/s^2.

Cálculo da velocidade

Inicialmente tentou-se determinar a velocidade fazendo a integração numérica do sinal de aceleração, mas, ao se executar este procedimento, o sinal da velocidade apresentou um erro acumulativo (de deriva) pois existe uma componente contínua DC como resultado da calibração de referência.

Este problema de deriva pode ser resolvido com a utilização de um filtro digital passa-alta antes da integração do sinal da velocidade ((GAVIN, 2001) e (GAVIN; MORALES; REILLY, 1998)). Desta forma, o valor da velocidade estimada $\dot{z}(t)$ é calculado a partir da integração do sinal de aceleração $\ddot{z}_p(t)$, após ter passado pelo filtro passa-alta, com freqüência de corte w_c, ilustrado na Figura 4.8. Para a aplicação utilizou-se o valor de w_c=8.

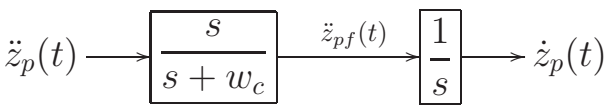

Figura 4.8. Diagrama de blocos em S (Laplace) para o cálculo da velocidade.

A partir das funções de transferência no domínio de Laplace, o integrador e o filtro passa-alta foram discretizados.

As funções de transferência obtidas no domínio discreto (Z), com freqüência de amostragem Freq, estão indicados na Figura 4.9.

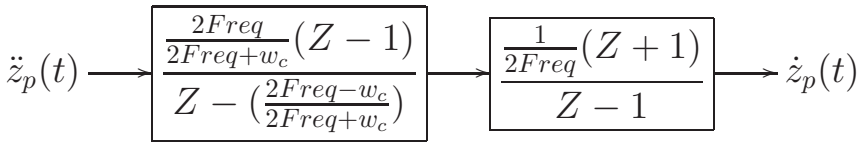

Figura 4.9. Diagrama de blocos em Z (discreto) para o cálculo da velocidade.

Convencionando-se que Z^{-1} é um operador de atraso, isto é, para um instante de tempo k:

$$Z^{-1}f(k) = f(k-1).$$

foi então obtida a equação da diferença para o filtro e o integrador, que é dada por:

$$\dot{v_p}(t) = \frac{2Freq - w_c}{2Freq + w_c} \cdot \dot{v_p}(t-1) + \frac{1}{2Freq + w_c}\left(\ddot{z}_p(t) + \ddot{z}_p(t-1)\right) \tag{4.10}$$

onde $\ddot{z}_p(t)$ e $\dot{v_p}(t)$ representam, respectivamente, a aceleração medida e a velocidade estimada para o acelerômetro p(1,2 e 3) nos três eixos coordenados no instante t=k.

4.3.3 Giroscópio

O circuito montado para este sensor é ilustrado na Figura 4.10. Conforme sugerido pelo fabricante Murata, é necessária a utilização de alguns filtros do tipo rejeita-faixa para suprimir os ruídos e garantir a leitura das alterações na frequência de oscilação do osciladores piezo-elétricos. A saída deste sinal é ligada a um circuito de condicionamento e ajuste de ganho (ver Figura 4.5).

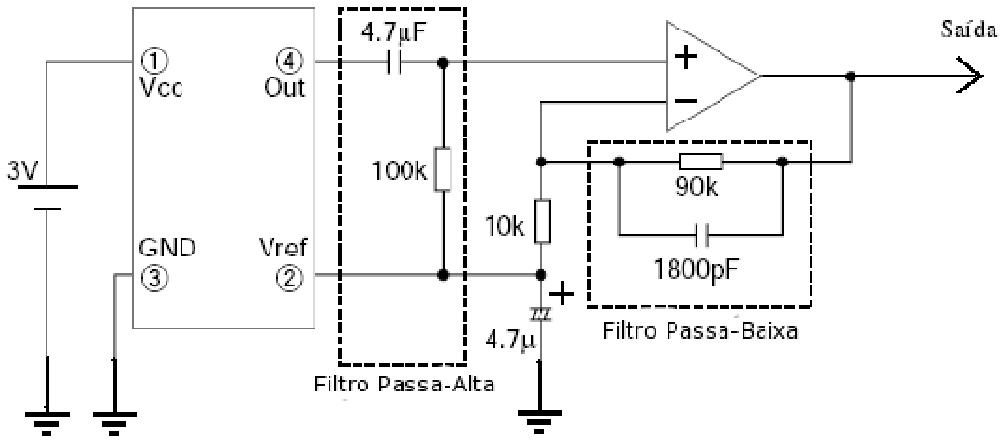

Figura 4.10. Circuito do giroscópio.

Cálculo do ângulo percorrido

Segundo PuttKamer (PUTTKAMER, 2000), para o uso adequado do giroscópio deve-se fazer uma rotina de aquecimento do sensor (*warm up*), que deve levar um pouco mais de 10 minutos. Isto é necessário para que o erro de leitura seja inferior a 1º/min. Feita esta rotina de aquecimento, o sensor deve ser recalibrado a cada 2 minutos, para diminuir o erro acumulativo do processo de integração e da sua variação em função da temperatura.

Assim, a obtenção do ângulo percorrido num intervalo $\triangle\ t$ é dado pela a integração do sinal lido pelo A/D, conforme equação 4.11:

$$\alpha = \int_{t_1}^{t2} (Valor_{lido} - Offset_{ref})\ \delta t \tag{4.11}$$

Substituindo o termo $Valor_{lido}$ pela equação 4.2 tem-se:

$$\alpha = \int_{t_1}^{t2} (1,2207 * 10^{-3} \cdot Valor_{conv} - Offset_{volts})\ \delta t \tag{4.12}$$

4.4 MÓDULO INTERFACE E CONTROLE

Este módulo de interface e controle é responsável pela aquisição das informações dos sensores, envio dos sinais de PWM aos atuadores e o interfaceamento da comunicação com a UCN.

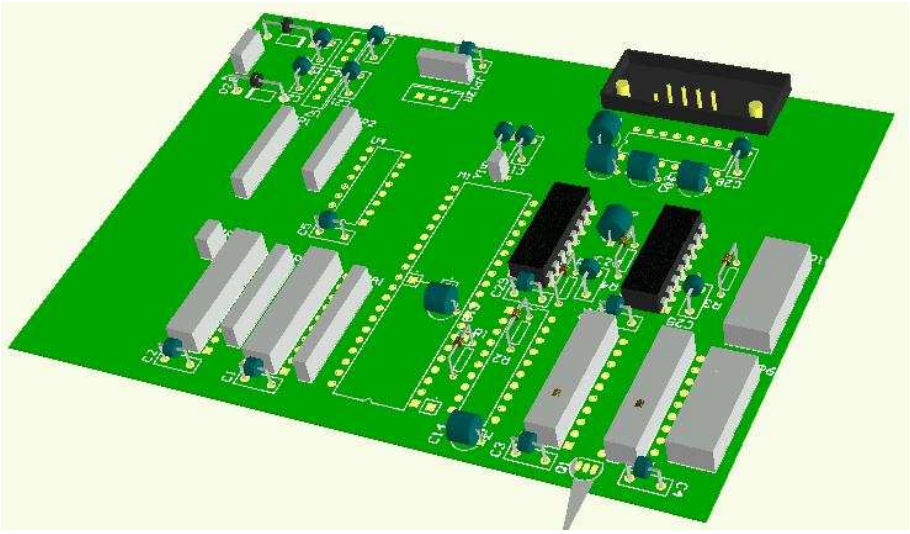

Figura 4.11. Placa de Interface e Controle.

Além das funções descritas anteriormente, este módulo é também responsável pelo fornecimento e condicionamento das tensões de alimentação do sistema. Usando três diferentes reguladores de tensão (LM7812, LM7912 e LM7805), esta placa provê tensões de 12 V, -12 V e 5 V, respectivamente. As tensões 12 V e -12 V são usadas no circuito de condicionamento e ajuste de ganho (ver Figura 4.5). Os circuitos restantes são alimentados com 5Vcc.

Na comunicação serial foi utilizado o conversor bidirecional MAX232, que converte o nível de tensão da interface padrão RS-232 em TTL e vice-versa.

Para possibilitar a expansão de atuadores ou a leitura de outros dispositivos no projeto desta placa, utilizou-se a técnica de E/S mapeada na memória, onde um decodificador de endereço de 3 bits (74LS138) faz o endereçamento de 8 endereços de E/S. Esta técnica requer um barramento de endereço e um outro barramento de da-

dos. A partir de um dado endereço válido, é selecionado o dispositivo que será lido ou onde será escrito um dado *byte* de informação. No presente sistema, o primeiro endereço está reservado para a leitura dos conversores A/D, enquanto os sete restantes ficam livres para o uso geral. Para a utilização destes outros endereços é necessário o desenvolvimento de uma placa de expansão, além da atualização do *firmware* do microcontrolador Intel modelo AT89C51, para que o mesmo possa ler ou escrever nestes novos endereços.

A placa de interface e controle possui ainda o circuito de chaveamento de modo de vôo. Este circuito (ver Figura 4.12) seleciona o sinal PWM que controla os atuadores pelo R/C ou pela a UCN (modo autônomo). Mesmo que esteja em modo R/C, a interface de comunicação continua ativa e enviando os dados adquiridos para a UCN ou para a estação-base. O circuito chaveador aciona o modo autônomo de forma automática, caso esteja fora do alcance do R/C. Para utilizar este recurso basta ligar o canal 5 (normalmente destinado ao trem de pouso) do R/C na entrada do circuito. Deve-se tomar cuidado na hora de chavear do modo automático para o modo R/C, pois o sinal de controle passará a ser a posição dos *joysticks* do R/C.

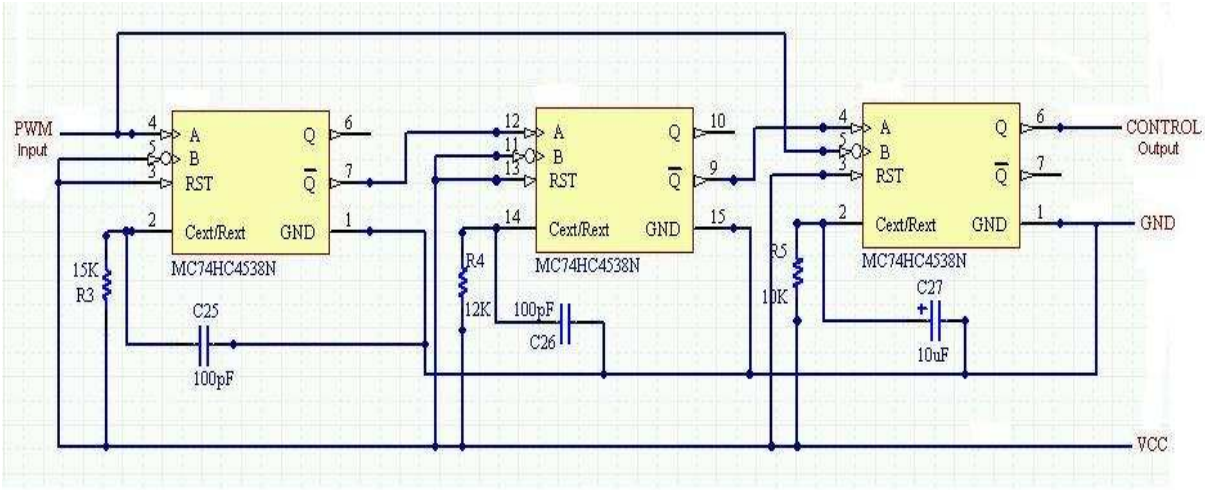

Figura 4.12. Circuito de chaveamento RC/Autônomo.

Este módulo possui dois microcontroladores. Um deles, o AT89C2051 (MP1), é responsável pela geração dos sinais de PWM para o acionamento dos micro servo-motores e o recebimento das informações de posição de cada motor. Ao final de cada pacote de informação, é gerado, pelo MP1, um pulso de sinalização para o AT89C51 (MP2), responsável pela aquisição dos dados e pelo envio das informações para a UCN ou a estação-base a cada sinalização de MP1.

Foram utilizados dois microcontroladores, primeiro para garantir a geração contínua dos sinais de PWM, uma vez que o PWM é gerado por interrupções do *timer* do microcontrolador, ou seja, os sinais de PWM são gerados por software. Por outro lado, tem-se a necessidade de mais uma entrada de comunicação serial UART para a interação com o GPS (ilustrado na Figura 4.13), uma vez que cada microcontrolador possui apenas uma porta serial. Foi testada uma segunda porta de comunicação serial por *software* em um dos microcontroladores. Desta forma, se evitaria a utilização de dois microcontroladores. Entretanto, a taxa de erro de comunicação foi muito alta e, portanto, esta solução não foi satisfatória para a aplicação requerida.

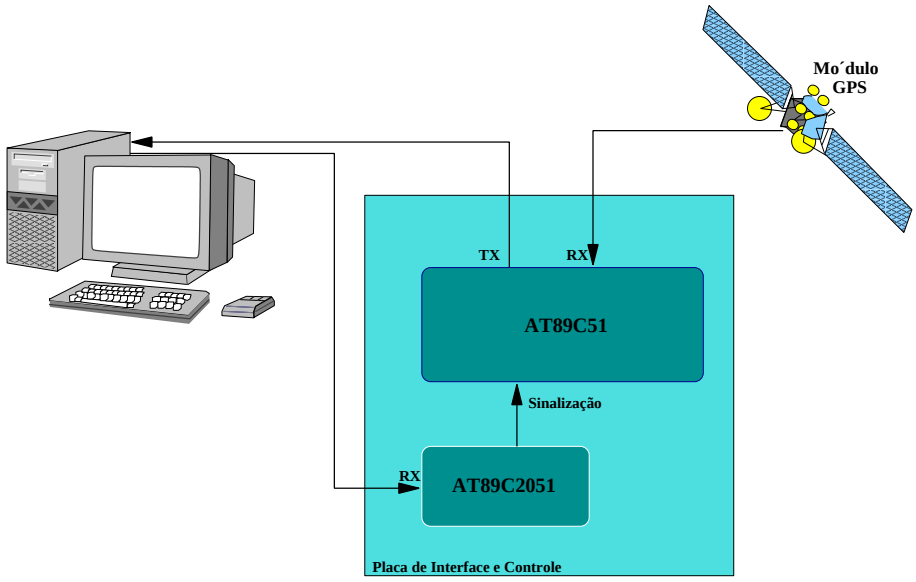

Figura 4.13. Esquema de comunicação com a interface.

4.4.1 Protocolo de Comunicação

A forma de comunicação usada é do tipo assíncrona, ou seja, não se utilizam sinais para o controle de fluxo. Isso permite uma maior simplicidade sem alterar sua robustez, uma vez que, em havendo perda de comunicação momentânea com a UCN, a comunicação pode ser reestabelecida a partir do primeiro próximo pacote de informação detectado. A princípio, não foi descartada a possibilidade da utilização de outros protocolos de comunicação já existentes, porém avaliação individual de cada um não faz parte do objetivo deste estudo.

Para testar esta hipótese de trabalho, foi feito o seguinte teste de robustez do protocolo de comunicação: em um dado instante, escolhido de forma aleatória, foi retirado o cabo de comunicação durante um determinado tempo (t) e em seguida recolocado. A comunicação foi restabelecida de pronto, assim que o primeiro pacote de informação foi detectado pela UART do microcontrolador. Apartir deste ensaio verificou-se que o protocolo desenvolvido é capaz de fornecer informações e robustez para proposta desenvolvida. Mesmo sabendo que a interface RS-232 é antiga, ela ainda é um padrão de entrada para muitos rádios-enlaces comerciais.

Existem vários protocolos de comunicação para RS-232, os quais não foram explorados neste trabalho, da mesma forma que não se abordou o tratamento de erro de comunicação. Trabalhos futuros podem investigar protocolos de comunicação adequados ao sistema.

No controle do atuador do tipo servo-motor, a UCN envia para o *hardware* valores inteiros entre 2 a 27. O valor 2 corresponde à posição de fim de curso para a direita, enquanto o valor 27 corresponde ao final de curso à esquerda. Este valores são interpretados pelo microcontrolador (MP1) como posições angulares destes micro-servos. As posições determinadas pelos códigos de 2 a 27 são conseguidas por deslocamentos angulares sucessivos de 10,8º. Esta precisão pode ser aumentada pela associação de dispositivos mecânicos ao eixo do motor.

Para transmitir estes valores de posição para o *hardware* de atuação, é montado um vetor de 10 *bytes* contendo o valor angular desejado para cada motor (ver Figura 4.14). Para sinalização de início de informação, é usado o caracter (;) e, para a finalização, o caracter (#). Ao final de cada pacote válido de informação, o MP1 sinaliza para o MP2 a necessidade do envio dos próximos dados a serem lidos pela UCN.

;	Motor 1	Motor 2	Motor 3	Motor 4	Motor 5	Motor 6	Motor 7	Motor 8	#

Figura 4.14. Palavra de controle dos servo-motores.

O pacote de informação que transita entre o MP2 e a UCN possui 35 palavras em código ASCII dispostas da seguinte forma: início do pacote; marcado por dois caracter (;), dois pares de *bytes* para cada um dos conversores A/D codificados em hexadecimal e totalizando 32 *bytes* de informação, sendo o pacote finalizado por um caracter (#). Este vetor está ilustrado na Figura 4.15.

;	;	$AD1_{alta}$	$AD1_{baixa}$	$AD2_{alta}$	$AD2_{baixa}$	$AD3_{alta}$	...	$AD8_{alta}$	$AD8_{baixa}$	#

Figura 4.15. Palavra de envio dos dados lidos.

Considerando-se o seguinte vetor de dados:

; ; 0A 12 02 DD 02 03 59 02 03 59 02 03 59 02 03 59 #

A informação passada para os diferentes conversores é:

$$\begin{aligned} AD1 = &\quad 0A12h = &\quad 2578 \\ AD2 = &\quad 02DDh = &\quad 733 \\ \vdots = &\quad \vdots = &\quad \vdots \\ AD8 = &\quad 0359h = &\quad 857 \end{aligned}$$

Ou seja, o valor convertido, em base decimal, por cada um dos conversores A/D é: AD1 = 2578, AD2 = 733, $\cdots$, AD8 = 857. De posse desses valores e de uma lei de conversão de base e unidade para cada sensor, é possível obter as diferentes grandezas físicas sensoriadas nas unidades próprias de tais grandezas.

Tal arquitetura permite a substituição padrão (*standard*) tanto dos sensores quanto do canal de aquisição utilizado. Esta substituição pode ser feita sem que seja necessário alterar o *firmware* do módulo de interface e controle. Será necessário simplemente ajustar o equacionamento da lei de conversão de base e unidade na UCN.

4.5 CALIBRAÇÃO DO ACELERÔMETRO

Durante o processo de inicialização da unidade de controle e navegação (UCN), são feitas amostragens do valor lido pelos acelerômetros com a aeronave ainda em repouso. Esta medida serve como calibração deste sensor.

A seguir, é discutida a equação da recorrência que permite calcular o valor médio atual (no presente instante t_{n+1}) em função do valor obtido para a média no instante imediatamente anterior a este (t_n). Isto porque os acelerômetros são sensores com elevado nível de ruído, muito sensíveis e possuem uma micro-mecânica (MEMS), conforme explicado na Seção 2.3.2, necessitando de alguns minutos para que os seus sinais de saída e a sua micro-mecânica se estabilize, permitindo uma medida mais precisa.

Sabendo-se que a média aritmética dos dados adquiridos no instante atual ($\overline{t}(k)$) é dada por:

$$\overline{t}(k) = \frac{\sum_{i=1}^{k} t_i}{k} \tag{4.13}$$

então o valor da média aritmética dos dados adquiridos no instante seguinte $\overline{x}(n+1)$ é dado por:

$$\overline{t}(k+1) = \frac{\sum_{i=1}^{k+1} t_i}{k+1} = \frac{\left(\sum_{i=1}^{k} t_i\right) + t_{n+1}}{k+1} \tag{4.14}$$

em que t_i são os valores lidos e k, a quantidade de valores lidos.

Substituindo-se a expressão 4.13 em 4.14, obtém-se:

$$\overline{t}(k+1) = \frac{k\overline{t}(k) + t_{k+1}}{k+1} = \frac{k\overline{t}(k) + \overline{t}(k) - \overline{t}(k) + t_{k+1}}{k+1} = \tag{4.15}$$

$$\overline{t}(k+1) = \overline{t}(k) + \frac{1}{k+1}\left(t_{k+1} - \overline{t}(k)\right).$$

Reescrevendo a equação 4.15 em função do valor de referência, tem-se:

$$refA_{inicial}(k+1) = refA_{inicial} + \frac{1}{k+1}\left(Valor_{atual} - refA_{inicial}\right). \tag{4.16}$$

O cálculo do valor de referência, ou valor de tensão para aceleração inercial nula ($refA_{inicial}$), é fornecido pela expressão 4.16. Para que a calibração dos acelerômetros também leve em consideração a sua orientação em relação aos eixos coordenados, cada um dos acelerômetros deve ser alinhado à vertical, de tal forma que sofra a ação apenas da aceleração gravitacional local, conforme descrito na Seção 4.3.2 pela equação 4.8.

CAPÍTULO 5

TESTES DO SISTEMA DESENVOLVIDO

Neste capítulo, são apresentados a descrição dos ensaios submetidos e os resultados dos ensaios obtidos com o sistema desenvolvido. Os ensaios estão divididos em três grupos: ensaios em solo para aferição do tempo de resposta e a calibração dos sensores, ensaio dentro de um automóvel de passeio para validação do protótipo e ensaio do sistema em vôo para aquisição dos dados em condições reais de funcionamento.

5.1 ENSAIO DO SISTEMA EM SOLO

Nos ensaios em solo, foi montado um pêndulo simples instalado no teto do laboratório da UFBA, no intuito de aferir os sensores de aceleração do sistema desenvolvido.

Um pêndulo simples ou pêndulo matemático é constituído por um fio inextensível preso a uma estrutura rígida e indeformável, sendo que, na extremidade desse fio, é fixada uma massa muitas vezes maior que a massa do fio. Sob a ação do campo gravitacional, o conjunto fio-massa ficará em posição vertical quando abandonado em repouso. Quando tirado de sua posição de equilíbrio, atua sobre a massa a força-peso, dividida em duas componentes; uma, na mesma direção do fio (P_y), e outra (P_x), perpendicular à componente (P_y). Sob a ação da força restauradora (P_x), o pêndulo oscilará em um plano vertical; o movimento é periódico e harmônico (HALLIDAY; RESNICK, 1984).

A Figura 5.1 mostra um pêndulo de comprimento L e massa m. Como se pode ver, o fio forma um ângulo θ com a vertical. As forças que atuam sobre m são: o peso ($Ps = m \cdot g$) e a tração da corda Tr. O movimento é em torno de um arco de círculo de raio de L, tomando os eixos de referência de forma que um deles seja radial e o

outro tangente ao círculo. O peso Ps pode ser decomposto, como dito anteriormente, numa componente radial de módulo $P_y = m \cdot g \cdot cos\theta$ e numa componente tangencial $P_x = -m \cdot g \cdot sen\theta$. A componente radial da resultante é a força centrípeta (P_y) que mantém a partícula na trajetória circular. A componente tangencial é a força restauradora (P_x), onde o sinal negativo indica que F se opõe ao fato de o pêndulo ter sido tirado de sua posição de equilíbrio θ=0.

$$F = -m \cdot g \cdot sen(\theta) \tag{5.1}$$

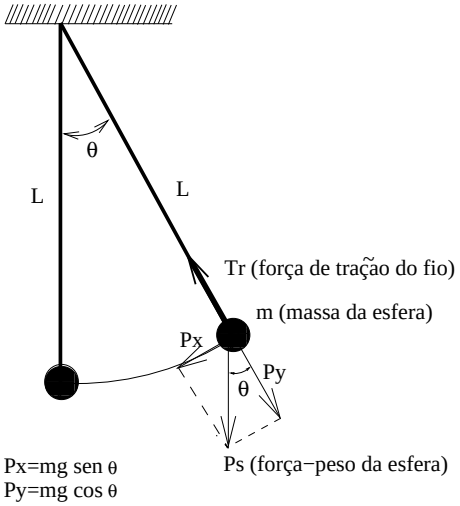

Figura 5.1. Pêndulo simples e as forças que atuam sobre a esfera de massa m.

A partir da equação 5.1, observa-se que a força restauradora (P_x) não é proporcional ao deslocamento angular θ e sim a sen(θ). O movimento, portanto, não é harmônico simples. Para efeito de tratamento matemático e em primeira aproximação, se o ângulo θ for suficientemente pequeno, sen(θ) será aproximadamente igual a θ. Note-se que a diferença numérica dos resultados, uma vez feita a aproximação proposta, é de cerca de 0,1%. Já o deslocamento ao longo do arco será $x = L \cdot \theta$ e, para ângulos pequenos, ele será aproximadamente retilíneo (HALLIDAY; RESNICK, 1984). Sendo assim, é possível escrever:

$$P_x = F \cong -m \cdot g \cdot \theta = -m \cdot g \cdot \frac{x}{L} = -\frac{m \cdot g}{L} * x \tag{5.2}$$

Como se vê na equação 5.2, a força restauradora P_x é proporcional ao deslocamento e tem sentido oposto. Esta é exatamente a condição para se ter movimento harmônico simples. De fato, a equação 5.2 tem a mesma forma que a equação, $F = -k \cdot x$, onde o termo $\frac{m \cdot g}{L}$ assemelha-se à constante k da lei de Hook. Para pequenas amplitudes, o período t (tempo de um ciclo) pode ser obtido fazendo-se k = $\frac{m \cdot g}{L}$

$$t = 2\pi \cdot \sqrt{\frac{m}{k}} = 2\pi \cdot \sqrt{\frac{m}{m \cdot g/L}} \rightarrow t = 2\pi \cdot \sqrt{\frac{l}{g}} \tag{5.3}$$

O pêndulo simples (ver a equação 5.3) pode ser usado para a medição do valor da aceleração gravitacional local g. É possível determinar L e t usando um cronômetro e uma trena e assim obter uma precisão melhor que 1% (HALLIDAY; RESNICK, 1984).

$$g = 4\pi^2 * \frac{L}{t^2} \tag{5.4}$$

Note-se que o período t é independente da massa m do corpo suspenso.

Quando a amplitude de oscilação não é pequena, o sistema pode ser resolvido pela expansão em série das equações, sendo que a equação geral para o período pode ser descrita como (HALLIDAY; RESNICK, 1984):

$$t = 2\pi\sqrt{\frac{L}{g}} \cdot (1 + \frac{1}{2^2} \cdot sen^2\frac{\theta_m}{2} + \frac{1}{2^2} \cdot \frac{3^2}{4^2} \cdot sen^4\frac{\theta_m}{2} + \ldots) \tag{5.5}$$

onde θ_m é o deslocamento angular máximo e os termos consecutivos da série, com expoente superior a dois, serão cada vez menores. O período pode ser calculado com um grau de precisão ainda melhor, dependendo do número de termos utilizados na série. Como o período de um pêndulo simples, mesmo em regime de grandes ângulos,

é praticamente independente da amplitude, o mesmo permanece inalterado ao longo do experimento quando o intervalo de tempo é de algumas oscilações do pêndulo simples(HALLIDAY; RESNICK, 1984).

5.1.1 Descrição do Ensaio com Pêndulo Simples

O objetivo principal deste ensaio é verificar as respostas em freqüência dos sensores de aceleração, testar a interface de comunicação paralela e uma vez que o acelerômetro expressa suas medidas em função de g é necessário se determinar o valor de g local, para efeito de calibração. Tal ensaio foi montado no laboratório de Física Experimental-II, na sala 211 do Instituto de Física da UFBA.

Devido ao fato dos acelerômetros utilizados terem uma escala muito grande para o propósito deste trabalho ($\pm$250g), se faz necessário conhecer a sua resposta em freüência para baixas acelerações e verificar se este sensor é capaz de efetuar as medidas de forma correta para um sistema de navegação.

O sistema desenvolvido foi colocado dentro de uma cesta plástica junto com um peso de 5 kg e uma fonte de alimentação, conforme a Figura 5.2. A montagem final pode ser vista nas Figuras 5.2 e 5.3.

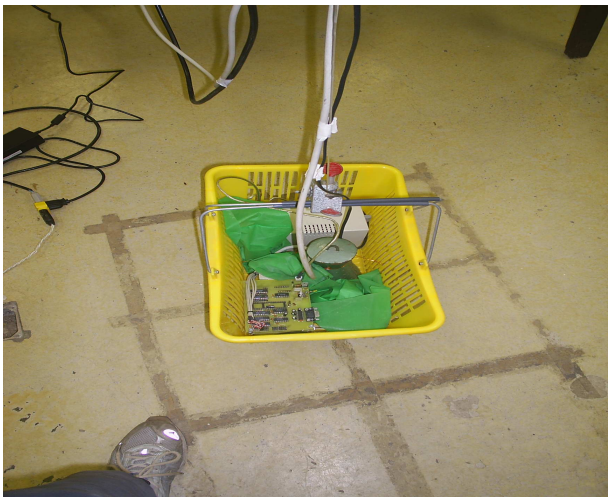

Figura 5.2. Montagem do sistema em teste para o ensaio de pêndulo simples.

Figura 5.3. Montagem final do ensaio de pêndulo simples.

Para garantir a precisão do valor do período t, mediu-se o tempo de 10 oscilações três vezes, conforme Tabela 5.1 e o ângulo θ inicial utilizado foi de aproximadamente de 25º.

Tabela 5.1. Períodos medidos.

Medida	Tempo para 10 oscilações (s)
1º	38,04
2º	38,06
3º	38,02
Média para 10 oscilações	**38,04**

De posse do valor do período $t = 3{,}804 \pm 0.002$s e do comprimento $L = 3{,}57 \pm 0.02$m, é possível calcular o valor de g local aplicando-se a equação 5.6:

$$g = 4\pi^2 \cdot \frac{3,57}{t^2} \rightarrow g = 4 \cdot 3,1415^2 \frac{3,57}{14,47042} = 9,739 \pm 0.06 \ m/s^2 \qquad (5.6)$$

Fazendo-se a comparação do valor de g encontrado por medição (9,73915 m/s^2) com o valor de g local para Salvador [2] (9,7823 m/s^2), o erro relativo entre essas duas grandezas é de 0,44%. Este será o erro assumido para o conjunto das medições de aceleração pelo sistema de navegação e telemetria.

Como ilustrado na Figura 5.4, os dados aparecem associados a ruídos. O fabricante especifica este nível de ruído como sendo de 2,8 mV_{rms} entre 10 Hz e 10 kHz (MOTOROLA, 2004b) e (MOTOROLA, 2004a). Mesmo após a aplicação de um filtro passa-baixa de primeira ordem RC (resistor-capacitor) antes da etapa de amplificação e condicionamento do sinal, não se conseguiu uma melhora significativa. Para se conseguir efetuar a leitura dos dados, foi necessário aumentar o ganho da etapa de amplificação e condicionamento do sinal em 100 vezes e reajustar a tensão de *offset*.

[2] valor estabelecido pelo Departamento de Geofísica do Instituto de Geociências da UFBA - Comunicação Privada

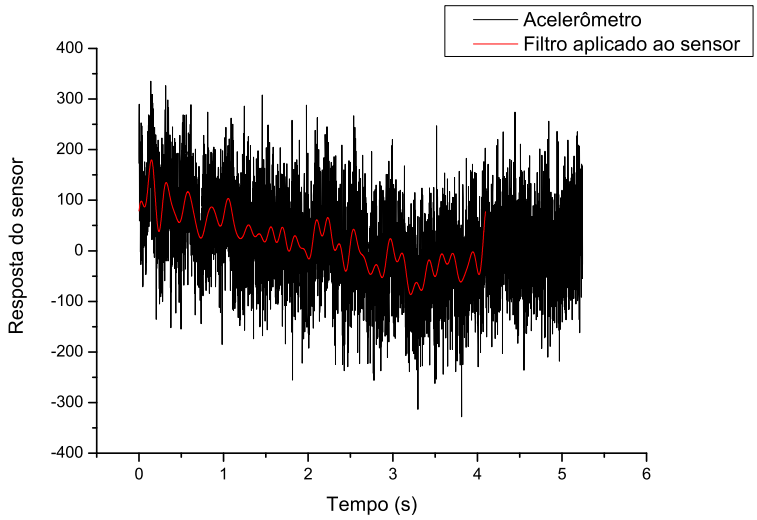

Figura 5.4. Dados amostrados com e sem aplicação do filtro.

O sinal em destaque na Figura 5.4 foi obtido após a utilização do filtro passa-baixa digital. A aplicação do filtro e da transformação rápida de Fourier (FFT - *Fast Fourier Transform*) foi feita usando-se o programa de pós-análise *Origin* (versão 7.0 da empresa OriginLab. Corporation) . Os dados da resposta em freqüência dos sensores após a FFT podem ser visualizados nas Figuras 5.5 e 5.6.

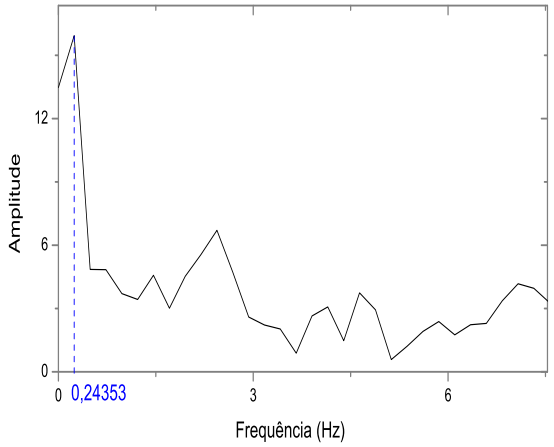

Figura 5.5. Resposta em Freqüência dos sensores após aplicação da FFT sem filtro digital.

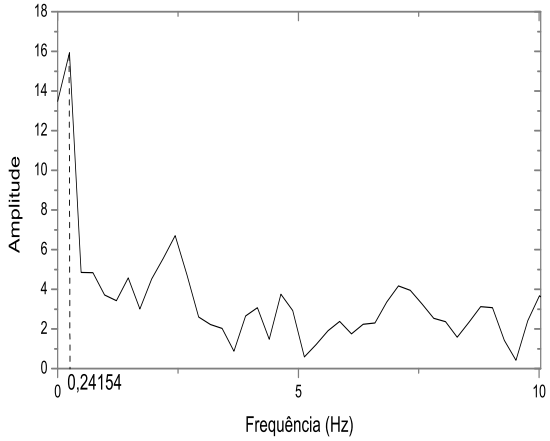

Figura 5.6. Resposta em Freqüência dos sensores após aplicação da FFT com filtro digital.

Outros ensaios deste tipos foram feitos. Muitas vezes, ao soltar o pêndulo, o mesmo tendia a sair da sua trajetória retilínea, inutilizando os dados adquiridos. Apesar deste desalinhamento do pêndulo em relação à sua trajetória, foi possível aferir um erro menor que 7,7% entre os dados experimentais com o cronômetro (0,26288 Hz) e os resultados obtidos pela aquisição e tratamento de dados dos acelerômetros (0,24353 Hz sem filtro e 0,24153 Hz com filtro digital), satisfazendo assim o propósito deste ensaio. A utilização deste sensor para aplicações de baixa aceleração é desaconselhável, pois o mesmo, apesar de sensível, tem uma baixa relação sinal/ruído nesta faixa de trabalho.

5.2 ENSAIO DO SISTEMA A BORDO DE UM AUTOMÓVEL

Para validar o sistema desenvolvido, antes de sua aplicação em navegação e telemetria de uma aeronave não-tripulada, é necessário que a informação obtida pelos acelerômetros tenha qualidade e precisão suficientes, para que possa ser usada na determinação indireta da velocidade e da distância percorrida. Assim, utilizando um automóvel de passeio e respectivos instrumentos de painel (odômetro e velocímetro), é possível confrontar a velocidade e a distância percorrida medidas pelos acelerômetros e armazenadas na memória de um IBM/PC. Sabe-se que a norma do CONTRAM para os odômetros automotivos admite um erro de até ±10 m para cada quilômetro percorrido, mas o contador de leitura do odômetro só permite distinguir ± 100m ou 0,1 km na distância percorrida, ao passo que são admitidos até 3% de erro para a velocidade medida por um velocímetro, e tais instrumentos devem ser capazes de medir mudanças de aceleração de acima de 2 m/s^2 (CONTRAN, 1999).

5.2.1 Descrição do Ensaio Submetido

Os principais objetivos deste ensaio foram testar os sensores de aceleração e o método de integração para o cálculo da velocidade e da distância.

O sistema desenvolvido estava acoplado a um conjunto de baterias de alimentação, o que significa um peso total de 4,4±0,001 kg. Esta massa confere uma certa inércia ao sistema, o que diminui a transmissão de vibrações do próprio carro para o sistema de medição.

Durante este ensaio, o móvel partiu do repouso com velocidade zero até atingir uma velocidade constante de 80±0,1 km/h, velocidade esta medida com o auxílio do velocímetro do painel do automóvel. Foram percorridos aproximadamente 4 km, numa duração total de ensaio de 190±1s. A cada meio quilômetro percorridos foram registrados a distância percorrida e o tempo transcorrido (ver Tabela 5.2). Considerando que, após o período de aceleração, o móvel passa a ter um movimento uniforme, ou

seja, aceleração igual a zero, foi possível montar a coluna de velocidade a partir dos dados obtidos por cronometragem, como aparece na Tabela 5.2. Desta forma, obtém-se uma melhor precisão na determinação da velocidade do automóvel, de forma que seja possível comparar os valores de cronometragem manual com os dados obtidos pelo sistema de navegação e telemetria.

Observando-se a Tabela 5.2, tem-se um intervalo médio de 23,75±3,13s para cada meio quilômetro percorrido. Considerando-se que, apartir de 81s da Tabela 5.2 tem-se a entrada do automóvel em regime de velocidade constante. Pode-se determinar que a velocidade média do automóvel é de 23,10±0,8 m/s ou 83,18±2,5 km/h.

Tabela 5.2. Dados coletados manualmente durante a primeira etapa do ensaio.

Tempo (s)	Distância (m)	Velocidade (m/s)
0	0	0
35,9	500	13,93
60	1000	20,75
81	1500	23,81
104	2000	21,74
126	2500	22,73
147	3000	23,81
169	3500	22,73
190	4000	23,81

Durante o ensaio, foram coletados 3.420 pontos pelo sistema de navegação e telemetria, ou seja, 18 pontos por segundo, em média. Na Figura 5.7, são apresentados os gráficos de aceleração e velocidade do móvel. O gráfico de aceleração foi montado a partir dos dados do acelerômetro; já os valores de velocidade foram obtidos pela integração numérica desses mesmos dados. Não foi utilizado nenhum recurso computacional de filtro ou suavização.

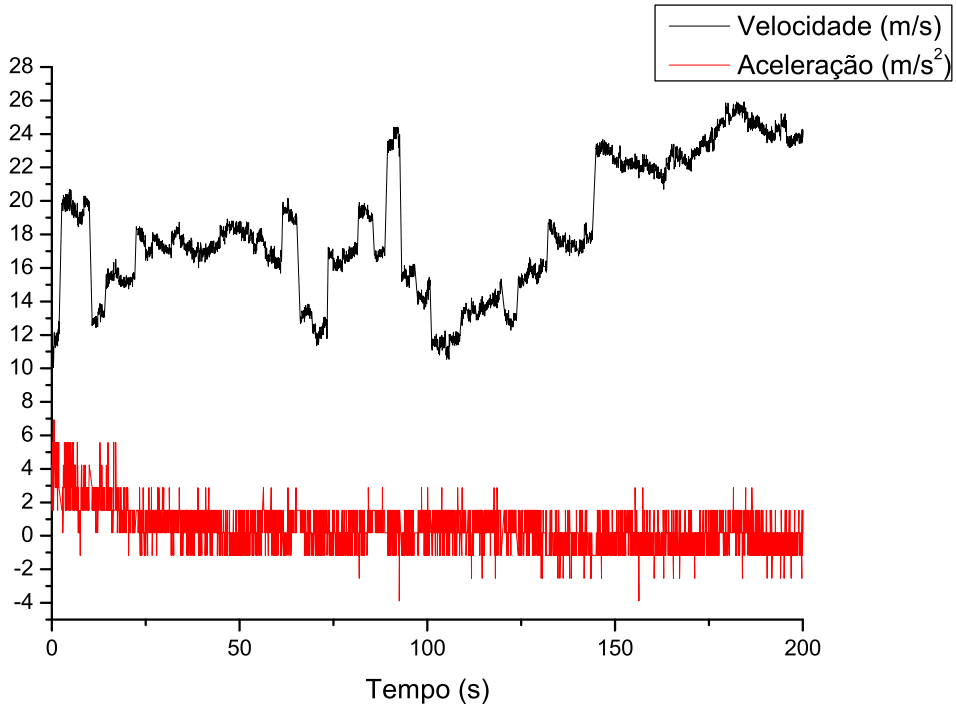

Figura 5.7. Dados aquiridos durante todo o ensaio no intervalo de tempo 0-190s.

Vale ressaltar que todos os métodos de integração numérica possuem um erro intrínseco; para o equipamento desenvolvido, este erro é de 3,3% na determinação da velocidade e de 5% na determinação da distância percorrida. Este erro foi estimado a partir da derivação das equações usadas na integração, além da aplicação de técnicas de propagação de erro, tanto para determinar o erro da velocidade (integral simples da aceleração), quanto na determinação do erro da distância (integral dupla da aceleração).

Para as Figuras 5.8 e 5.9, foi utilizado um filtro passa-baixa para suavizar os dados adquiridos. A suavização aplicada pelo programa Origin 7.0 é feita através de um filtro do tipo FFT, com uma janela de dados dos 20 pontos mais próximos a cada ponto. A Figura 5.8 apresenta a velocidade média de 19,84±0,65 m/s a partir de $t = 81s$, enquanto, com os valores da Tabela 5.2 neste mesmo intervalo, obtém-se uma média de 22,94±1,02 m/s, ou seja, o erro relativo é de aproximadamente 13%.

Na Figura 5.9 é mostrada a aceleração do carro sobre todo o intervalo. Neste gráfico, pode-se perceber duas diferentes regiões. A primeira tem início em $t = 0$ e se estende até $t = 50s$. Neste intervalo, é possível observar um máximo de aceleração que chega a aproximadamente 4,6 m/s^2 quando o móvel inicia o seu movimento; em seguida, a aceleração cai progressivamente até o valor zero em $t = 50s$. A segunda parte deste gráfico mostra uma aceleração média $(3,03 \pm 0,01 \cdot 10^{-2} m/s^2)$, apresentando um comportamento compatível com um regime de aceleração nula.

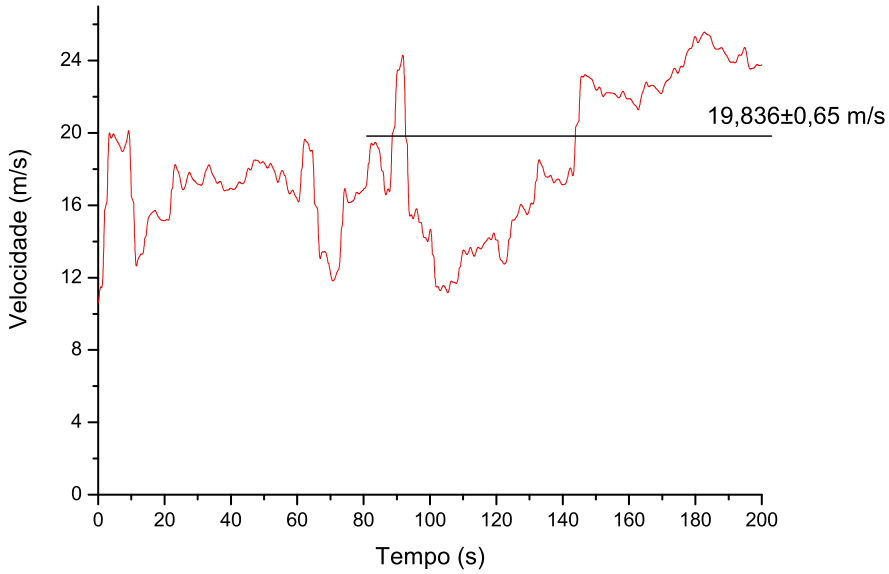

Figura 5.8. Aplicação da FFT nos dados adquiridos da velocidade.

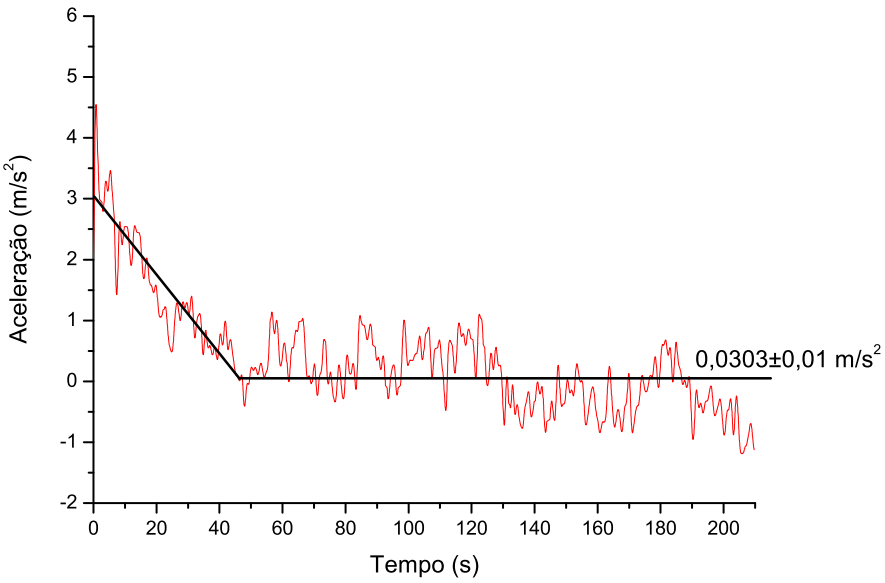

Figura 5.9. Aplicação da FFT nos dados adquiridos da aceleração.

Para o cálculo da distância percorrida a partir da velocidade, utilizou-se o programa Origin 7.0. O valor obtido através da integral foi de 3,89±0,20 km, enquanto o odômetro do automóvel marcou 4,0 ±0,1 km, ou seja, um erro de 2,9%, sem levar em consideração que o erro do odômetro pode ser maior para pequenos trajetos.

Na Figura 5.10, é possível ver os instantes em que foram feitas as trocas de marcha do automóvel no início do percurso teste. Esta Figura 5.10 é um bom exemplo de quanto o sistema desenvolvido é sensível e capaz de ser usado em outras aplicações que não apenas para aeronaves não-tripuladas.

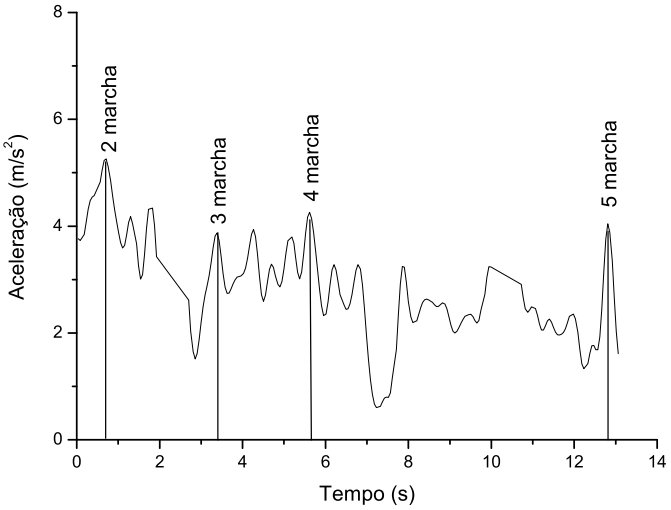

Figura 5.10. Dados adquiridos no intervalo de tempo 0-12s.

Neste desenvolvimento, não foi objeto de estudo a aplicação de outros métodos de integração para efeito comparativo. Mas, mesmo sem utilizar um método de integração mais adequado, o sistema desenvolvido apresentou resultados relativamente satisfatórios para testes iniciais. Verificou-se que, para os dados adquiridos, os erros percentuais foram relativamente baixos mesmo para um método de integração relativamente simples, e a taxa de amostragem (18 pontos por segundo) foi satisfatória para a aplicação considerada do sistema, que é uma aeronave não-tripulada.

Acredita-se que os resultados finais deste desenvolvimento podem ser melhorados à medida que sejam usados acelerômetros sensíveis na faixa de ±15g, ao invés dos atuais sensíveis na faixa de ±250g. Como se vê na Seção 5.1, a relação sinal/ruído característica desses acelerômetros impossibilita o seu uso para a aplicação proposta, já que, para tanto, é indispensável um tratamento matemático, conforme utilizado nos tratamentos dos resultados obtidos, o que requer maior capacidade de processamento na UCN.

5.3 ENSAIO DO SISTEMA EM VÔO PARA AQUISIÇÃO DE DADOS

Uma vez testado o sistema desenvolvido e aferida a qualidade dos dados adquiridos, optou-se por um ensaio em condições reais de vôo. O projeto AERO-UFBA desenvolve anualmente uma aeronave de carga de competição, segundo as normas estabelecidas pela SAE-BRASIL (ver Figura 5.11). Esta aeronave tem um compartimento de carga capaz de realizar o ensaio requerido, possuindo também características de portabilidade importante, podendo carregar até 9 kg. Sendo assim, o objetivo deste ensaio foi coletar informações da aeronave em vôo através de um módulo de comunicação serial entre a UCN e o sistema de navegação inercial por rádio.

Figura 5.11. Foto da aeronave do ensaio.

5.3.1 Descrição do ensaio

Para a realização deste ensaio em vôo foi necessário desenvolver um módulo de comunicação serial entre a UCN e o sistema de navegação inercial (ver foto na Figura 5.12) . Foram adquiridos dois pares de módulos híbridos da empresa Radiometrix,

modelos TX2 e RX2. Estes módulos são dotados externamente por uma blindagem completa, assim como de filtros internos anti-radiação espúria segundo o fabricante (RADIOMETRIX, 1997). Possui, também, um bit ultra-rápido (700μs) de sinalização para a detecção da portadora, estando apto a recepção de informação em menos de 1ms. Estes módulos têm alcance próximo de 300m em áreas abertas, taxa de transmissão de dados de até 28.800bps e consumo relativamente baixo de corrente (14 mA por módulo) (RADIOMETRIX, 1997).

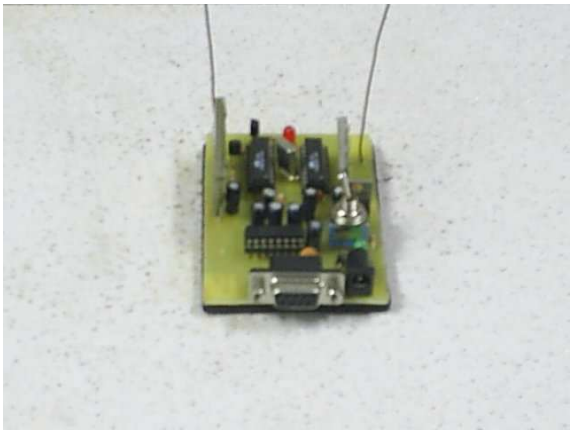

Figura 5.12. Módulo de comunicação da UCN com o sistema desenvolvido.

Devido à necessidade de uma infra-estrutura no local de ensaio, optou-se pelo aeroporto da cidade de Feira de Santana, BA. Em Salvador, os aeroclubes utilizados para vôos de aeromodelos não possuem energia elétrica ou espaço suficiente para alocação dos equipamentos de montagem, manutenção e teste em local coberto. Em Feira de Santana, BA, entretanto, tem-se à disposição um hangar completo no aeroclube.

Por causa do deslocamento imposto à equipe do AERO-UFBA, foi necessário montar uma logística de preparação da viagem. Esta preparação previu desde caixas com parafusos diversos até os equipamentos que seriam necessários a um eventual reparo da aeronave. Este ensaio envolveu toda a equipe, sendo preciso uma divisão em frentes de trabalho: um grupo para a montagem mecânica e regulagem da aeronave e outro grupo para preparar a alocação e a montagem do equipamento eletrônico (ver Figuras 5.13 e 5.14).

Figura 5.13. Grupo de montagem mecânica.

Figura 5.14. Grupo de montagem eletrônica.

Uma vez montada a aeronave, os ensaios foram divididos em 2 etapas:

1ª-Etapa: Vôo de trimagem[3] e verificação das condições de vôo, com carga útil de 3 kg;

2ª-Etapa: Vôo de aquisição de dados, teste de alcance do rádio e interferências de comunicação, com carga útil de 4,4 kg.

A primeira etapa foi o vôo de teste da aeronave, verificando-se a dirigibilidade em diferentes manobras e trimagem dos comandos. Esta é uma parte importante para o ensaio, pois, somente com a aeronave em vôo,é que o piloto pode perceber o seu real comportamento quando submedido as condições de climáticas do local. Assim, para esta primeira etapa, a aeronave decolou com uma carga útil de 3 kg, pois é o peso mínimo projetado para o mesmo.

Durante o vôo, o piloto fez os ajustes finos dos *links* de comando (trimagem do avião), para que se pudesse realizar um vôo sem tendências aerodinâmicas. Um aeromodelo trimado e em condições de vento favorável é capaz de realizar um vôo reto-nivelado sem a intervenção do piloto ou auxílio de qualquer equipamento.

Depois da operação de trimagem, a aeronave retornou ao solo após 5 minutos de

[3]O termo trimagem ou trimar corresponde ao ato de efetuar ajustes finos no aileron, profundor e leme através do R/C do aeromodelo, de forma que a aeronave não tenha tendências de movimento quando em vôo reto nivelado

vôo. Este limite foi imposto pela capacidade do reservatório de combustível. A etapa foi concluída com sucesso.

A segunda etapa foi o vôo para aquisição de dados, teste do alcance do rádio e interferências de comunicação. Os dados adquiridos foram tratados mais tarde, no laboratório da UFBA, com aplicação de diferentes filtros e métodos de suavização.

O equipamento embarcado com baterias seladas (chumbo ácido) tem um peso total de aproximadamente 4,4 kg, como já mencionando na Seção 5.2. Este conjunto foi instalado no compartimento de carga da aeronave, como pode ser visto na Figura 5.15.

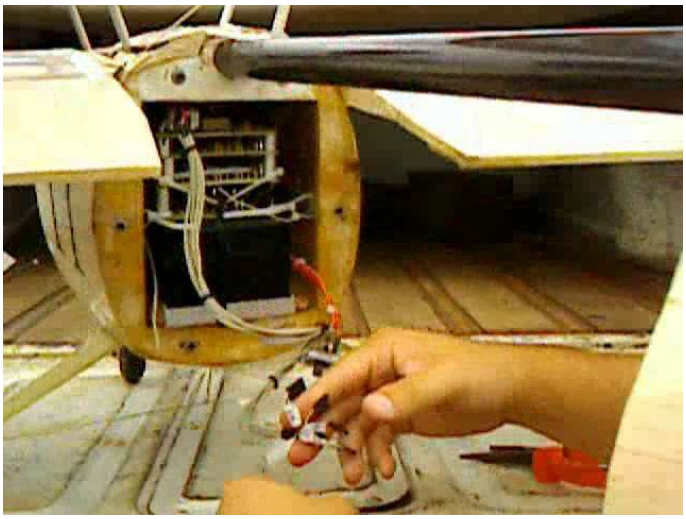

Figura 5.15. Compartimento de carga da aeronave com o sistema desenvolvido.

O piloto da equipe foi instruído a fazer vôos circulares próximos à UCN, porém o rádio não apresentou o alcance esperado. O alcance obtido foi de alguns metros e não de algumas centenas de metros. Foi então exigida habilidade por parte do piloto para passar o mais próximo possível à UCN, possibilitando assim a aquisição de dados, ainda que parcial. A melhor "janela" de dados adquiridos, aquela que apresentou um maior número de pontos correlatos, está resumida nos gráficos das Figuras 5.16 e 5.17.

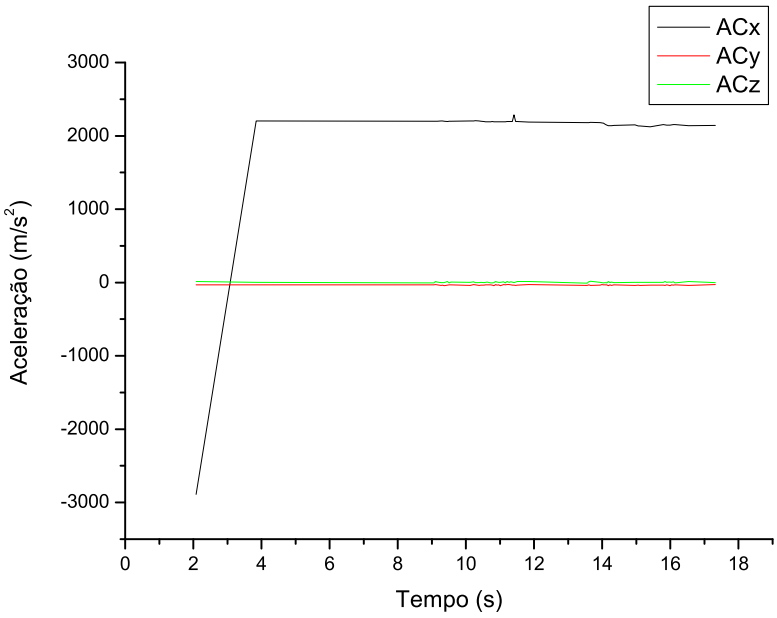

Figura 5.16. Dados do vôo para aceleração.

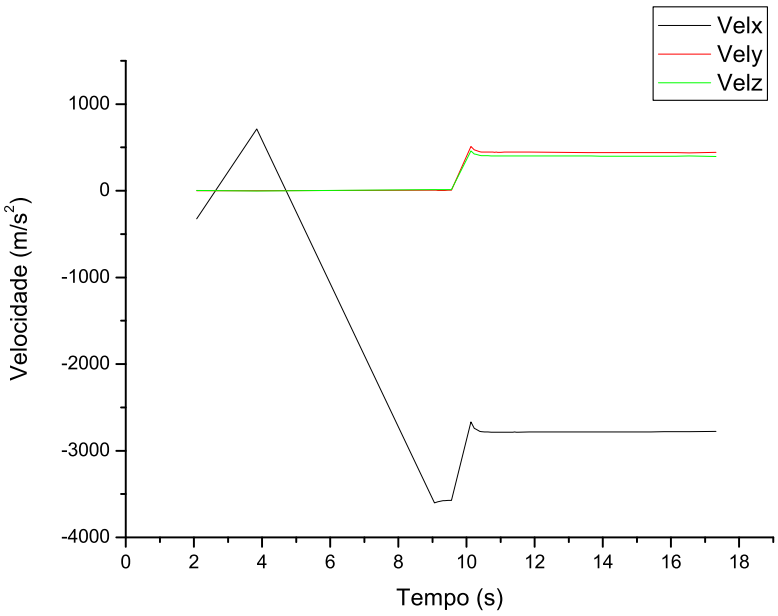

Figura 5.17. Dados do vôo para velocidade.

Ao final da "janela" de dados, o valor transmitido dos dados do sensor de aceleração do eixo X apresentou um acentuado aumento. Na prática, isso sugere a ocorrência de um mau contato elétrico entre o módulo de aquisição e o módulo dos sensores. Este fato só pôde ser comprovado quando do retorno ao laboratório na UFBA. Provavelmente, a causa deste mau contato está ligada às condições de transporte e/ou aos inúmeros reparos e procedimentos de montagem antes de cada ensaio. Outro fator para o mau funcionamento do rádio foi o tempo chuvoso nos dias em que foram realizados os ensaios.

Infelizmente, não foi possível obter nas informações de vôo com a qualidade esperada. O objetivo deste ensaio foi a medição de grandezas como velocidade de decolagem, desaceleração da aeronave no pouso, velocidade de *stol* e uma série de outras informações importantes para o suporte dos futuros desenvolvimentos pela equipe AERO-UFBA. Tais informações, por exemplo, poderiam ser utilizadas pela equipe do AERO-DESIGNER no desenvolvimento da próxima aeronave de competição para este ano.

CAPÍTULO 6

CONCLUSÃO

No presente trabalho, foi desenvolvido um sistema de navegação e telemetria aplicável a um veículo aéreo não-tripulado (UAV). Para tanto, foi feita a revisão da bibliografia, onde foram consultados não só trabalhos anteriores desenvolvidos no Brasil e no mundo, mas também foi feita uma investigação de alguns sistemas comerciais. Esta revisão serviu para "balizar" o projeto do equipamento. Tal sistema foi desenvolvido de maneira a somar esforços com o grupo de pesquisa do programa AERO-UFBA, que desenvolve atualmente o primeiro protótipo de UAV da Universidade Federal da Bahia.

Nesta dissertação, foi apresentado o princípio de funcionamento de cada sensor, além dos modelos matemáticos necessários ao trabalho de aquisição e análise dos dados adquiridos. Foi demonstrada a necessidade da aplicação de um filtro digital aos dados fornecidos por um conjunto de três acelerômetros para o cálculo da velocidade e do deslocamento da aeronave. A partir do sinal do acelerômetro, é possível conhecer a velocidade, com erro relativo de cerca de 3,3%, enquanto o deslocamento é conhecido com uma precisão de cerca de 5%.

Cada um dos módulos que formam o sistema de navegação desenvolvido foi projetado para trabalhar de forma independente. Assim, assegura-se maior flexibilidade e compactação do equipamento, permitindo que testes parciais sejam feitos durante o desenvolvimento dos circuitos propriamente ditos. Infelizmente, não foi atingido o nível ideal de compactação para a utilização em aeromodelos comerciais, como previsto. O equipamento final, entretanto, pôde ser instalado em uma aeronave com caracterísitcas e especificações para carga (neste caso, de acordo com as normas técnicas do concurso AERO-DESIGN da SAE-BRASIL).

O equipamento desenvolvido apresenta duas formas de comunicação de dados, entre a unidade de controle e navegação (UCN) e a interface desenvolvida, comunicação serial via rádio ou cabo e comunicação de dados paralelos via cabo. Assim, este equipamento pode ser utilizado tanto em aeromodelos como em robôs móveis, sistemas de navegação em automóveis convencionais ou experimentais, aviões etc.

Os módulos de aquisição de dados resultantes deste trabalho também podem ser aplicados como uma interface de aquisição de dados de baixo custo. Isto deve-se ao fato de que este módulo possui os circuitos necessários ao condicionamento e ajuste dos sinais de entrada, não necessitando, para tanto, de circuitos adicionais.

No ensaio do pêndulo foi possível determinar a gravidade local ($9,7392 m/s^2$), valor este que comparado com $9,7823 m/s^2$, que é o valor da aceleração gravitacional em Salvador-BA, apresenta um erro percentual de 0,44% . Este ensaio serviu para a calibração dos acelerômetros. O tempo de resposta (18 pontos por segundo) e a sensibilidade obtida são compatíveis com os dados do fabricante. É recomendada, porém, a utilização de um acelerômetro de menor fundo de escala, o que certamente melhorará a relação sinal/ruído destes transdutores.

O ensaio no automóvel foi de grande valia para a etapa de desenvolvimento. Foi possível testar não só as respostas do equipamento às excitações físicas (aceleração, velocidade, deslocamento e mudança de marcha) geradas pelo automóvel, como também aferir os valores adquiridos pelo sistema.

Infelizmente, o ensaio real na aeronave não foi considerado como bem sucedido devido ao alcance e ruído nos enlaces de rádio prejudicados pelo mau tempo em que se realizou os ensaios, o que não invalida o experimento. Apesar de não fazer parte dos objetivos deste trabalho, também foi desenvolvida uma interface gráfica capaz de apresentar ao usuário e armazenar todos os dados adquiridos.

6.1 CONTINUIDADE DA PESQUISA

Abaixo, enumeram-se algumas atividades que indicam potencial para melhor explorar esta solução tecnológica e abordar novos problemas:

- a aquisição de um enlace de rádio comercial de maior alcance.
- o refinamento dos algoritmos desenvolvidos;
- a investigação de um protocolo de comunicação que mais se adeque ao sistema;
- a otimização do funcionamento da UCN, reduzindo o tempo de resposta da mesma;
- a miniaturzação do conjunto de módulos que formam o sistema de navegação e telemetria;
- a confecção das placas impressas em processo industrial, melhorando os maus contatos;

REFERÊNCIA

ABE, H. Applications expand for downsized piezoelectric vibrating gyroscopes. *Dataweek Electronics & Communication Technology*, Novembro 2005.

AEROSONDE. *AEROSONDE*. 2001. Disponível em: <http://www.aerosonde.com>. Acesso em: 21 junho 2006.

AIRFORCE Technology. 2006. SPG Media Limited. Disponível em: <http://www.airforce-technology.com/projects/predator/>. Acesso em: 21 de junho de 2006.

AVATAR. *AVATAR - Autonomous Vehicle Aerial Tracking and Retrieval*. 2004. Disponível em: <http://www-robotics.usc.edu/ avatar/>. Acesso em: 05 de outubro de 2005.

AYALA, K. J. *The 8051 MICROCONTROLLER -Architecture, Programmig and Applications*. [S.l.]: West Publishing Company, 1991.

CONTRAN. *RESOLUÇÃO Nº 92*. Maio 1999. CONSELHO NACIONAL DE TRÂNSITO.

CORPORATION, I. *MCS@51 MICROCONTROLLER FAMILY USER'S MANUAL*. [S.l.]: Intel Corporation, 1994.

DITTRICH, J. S. *Design and Integration of an Unmanned Aerial Vehicle Navigation System*. Dissertação (Mestrado) — Georgia School of Aerospace Engineering, 2002.

DIXON, S.; WICKENS, C. D. *Imperfect Automation in Unmanned Aerial Vehicle Flight Control*. [S.l.], Agosto 2003.

DOMELEN, D. J. V. *Getting Around The Coriolis Force*. 2006. The Ohio State University. Disponível em: <http://www.physics.ohio-state.edu/ dvandom/Edu/newcor.html>. Acesso em: 21 de junho de 2006.

DURRAN, D. R. Is the coriolis force really responsible for the inertial oscillation? *Bulletin of the American Meteorological Society*, v. 74, p. 2179–2184, 1993.

ELFES, A.; MAETA, S. M. A semi-autonomous robot airship for environment monitoring missions. In: *IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION*. [S.l.: s.n.], 1998. p. 3449–3455.

Projeto AURORA. 79-84 p.

FRADEN, J. *Handbook of Modern Sensors*. Segunda edição. [S.l.]: AIP Press, 1996.

FUTABA. *Futaba Corporation of America*. 2003. FUTABA R/C Hobbies. Disponível em: <http://www.futaba.com>. Acesso em: 21 de março de 2005.

GAVIN, H. P. Control of seismically-excited vibration using eletrorheological materials and lyapunov methods. *IEE Transactions on Automation Control*, v. 9, n. 1, p. 27–37, 2001.

GAVIN, H. P.; MORALES, R.; REILLY, K. Drift free. *Review of scientific Instruments*, v. 69, n. 5, p. 2171–2175, 1998.

GIMENEZ, S. P. *Microcontroladores 8051*. [S.l.]: Editora Pearson Education do Brasil, 2002.

HALL, C. E. J. A real-time linux for autonomous navigation and flight attitude control of an uninhabited aerial vehicle. *IEEE Digital Avionics Systems*, p. 1–9, 2001.

HALLBERG, E.; KAMINER, I.; PASCOAL, A. Flight test system for unmanned air vehicles. *IEEE Control Systems*, n. PAS-101, p. 55–65, 1999.

HALLBERG, E.; KAMINER, I.; SILVESTRE, C. Trajectory tracking for autonomous vehicles: An integrated approach to guidance and control. *AIAAA Jornal of Guidance, Control and Dynamics*, v. 21, n. 1, p. 29–38, 1998.

HALLIDAY, D.; RESNICK, R. *Física*. [S.l.]: LTC - Livros Técnicos e Científicos, 1984.

HILL, W. J. et al. Association for unmanned vehicle systems international. In: TEXAS A&M UNIVERSITY - TEXAS BUZZARD. [S.l.], 2004.

INSTRUMENTS, T. *Datasheet ADS1286*. Outubro 1998. BURR-BROWN Produts for Texas Instruments.

JONHSON, E.; FORTAINE, S. Use of flight simulation to complement flight testing of low-cost uavs. *AIAA Modeling and Simulation Technology Conference*, n. 4059, 2001.

KAHN, A. D.; KELLOG, J. C. Low complexity, low-cost, altitude/heading hold flight control system. *IEEE AESS System Magazine*, p. 14–18, Abril 2003.

MAETA, S. M. *Desenvolvimento da Infra-estrutura Embarcada do Projeto AURORA/IC*. Dissertação (Mestrado) — UNICAMP, 2001.

MARQUETTE, J. *U-NAV - Modular UAV Control System*. 2004. Disponível em: <http://www.silvertone.com.au/pdc.htm>. Acesso em: 21 de junho de 2006.

MICROPILOT. *MicroPilot Control System*. 2005. Disponível em: <http://www.micropilot.com>. Acesso em: 21 de junho de 2006.

MONTGOMERY, J. F. *The USC Autonomous Flying Vehicle (AFV) Project: Year 2000 Status*. [S.l.], 2000.

MOTOROLA. *MPX4115: Integrated Silicon Pressure Sensor for Mainfold Abosulte Pressure, Altimeter or Barometer Applications On-Chip Signal Conditioned, Temperature Compensated and Calibrated*. Junho 2001. MOTOROLA.

MOTOROLA. *MMA1200D: Z Axis Sensitivity Micromachined Acelerometer*. Junho 2004. MOTOROLA.

MOTOROLA. *MMA2300D: X Axis Sensitivity Micromachined Acelerometer*. Junho 2004. MOTOROLA.

MURATA. *Piezoelectric vibrating gyroscope ENC03*. Março 1999. MURATA.

NERIS, L. de O. *Um piloto automático para as Aeronaves do Projeto ARARA*. Dissertação (Mestrado) — ICMC/USP, São Carlos, 2001.

NICOLOSI, D. E. C. *Microcontrolador 8051 Detalhado*. [S.l.]: Editora Érica, 2000.

PICCOLO. *Cloud Cap Technology*. 2003. Disponível em: <http://www.cloudcaptech.com/>. Acesso em: 05 de outubro de 2005.

PICCOLO. *Piccolo System User Guide*. 1.1.9. ed. [S.l.], Fevereiro 2004.

PUTTKAMER, E. von. *Usage of a Vibrating Gyroscope of Orientation Estimation*. [S.l.], 2000.

RADIOMETRIX. *Datasheet TX2 and RX2*. 1997. Radiometrix.

RAMOS, J. J. G. et al. Project aurora: Autonomous unmanned remote monitoring robotic airship. In: *2ND INTERNATIONAL AIRSHIP CONVENTIONAL AND EXHIBITION*. Bedford, UK: [s.n.], 1998. p. 91–103.

RAMOS, J. J. G. et al. A software environment for an autonomous unmanned airship. In: *PROCEEDINGS OF THE 1999 IEE/ASME INTERNATIONAL CONFERENCE ON ADVANCED INTELLIGENT MECHATRONICS*. Atlanta, USA: [s.n.], 1999.

RAMOS, J. J. G. et al. The aerosonde robotic aircraft: A new paradigm for environmental observation. *Bulletin of the American Meteorological Society*, v. 82, n. 5, p. 889–902, 2001.

SAE BRASIL. *SAE BRASIL*. 2004. SAE BRASIL - Sociedade de Engenheiros da Mobilidade. Disponível em: <http://www.saebrasil.org.br/>. Acesso em: 21 de junho de 2006.

SILVA, V. P. da. *Aplicações Práticas do Microcontrolador 8051*. [S.l.]: Editora Érica, 1998.

SMITH, S. G. Automatic navigation in the air and at the sea. *Aeronautical Journal*, v. 97, n. 966, p. 183–194, Junho 1993.

SOUZA, P. N. *Sistemas Alternativos para a Obtenção de Imagem Aérea e sua Aplicação na Agricultura*. Dissertação (Mestrado) — ICMC/USP, São Carlos, 1999.

STARGATE. *Datasheet Stargate Developer's Guide*. 2004. Crossbow.

A Mission Planner and Navigation System For The ARARA Project.

APÊNDICE A

ESQUEMAS ELETRÔNICOS DESENVOLVIDOS

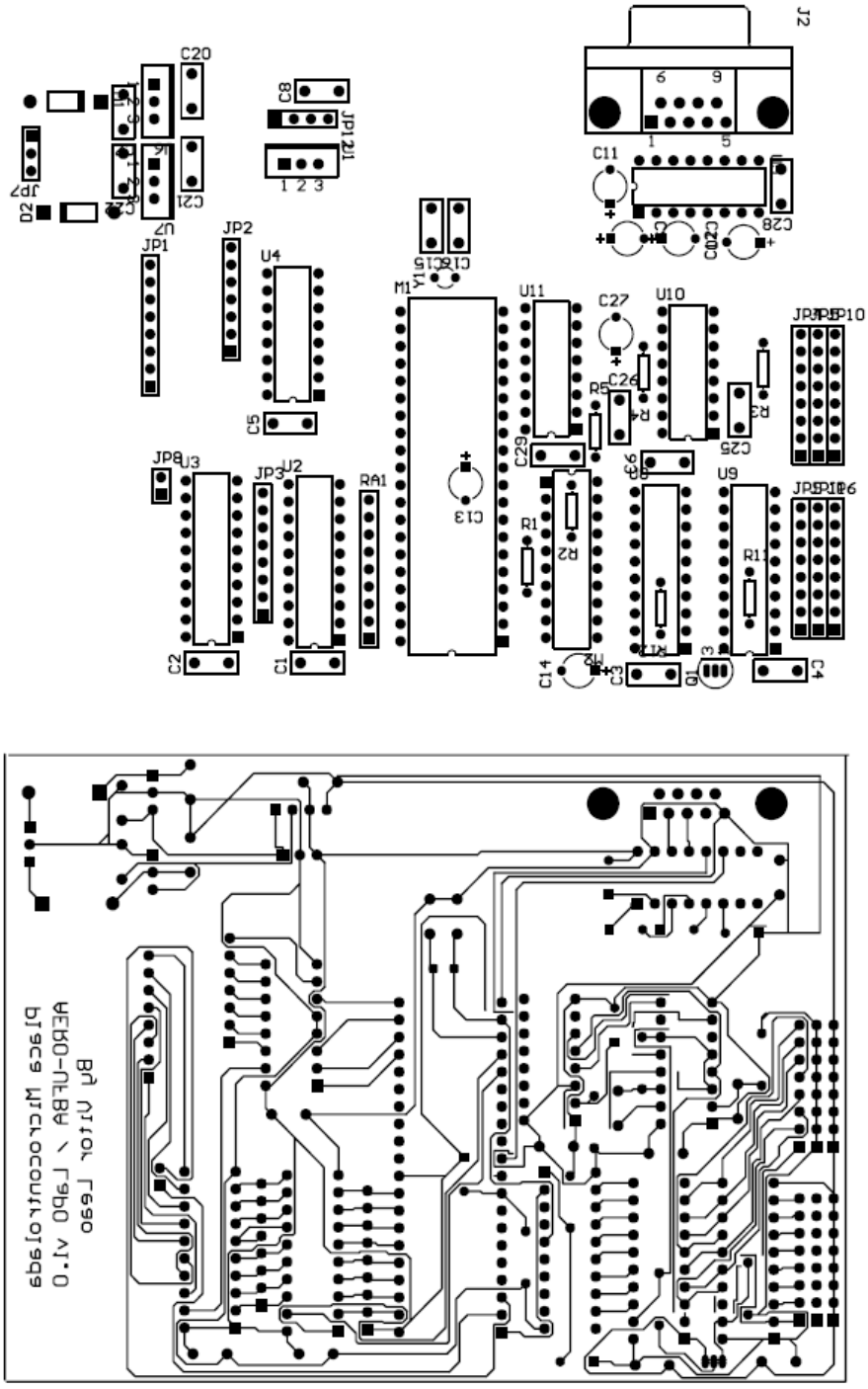

Figura A.1. Layout eletrônico da placa microcontroladora

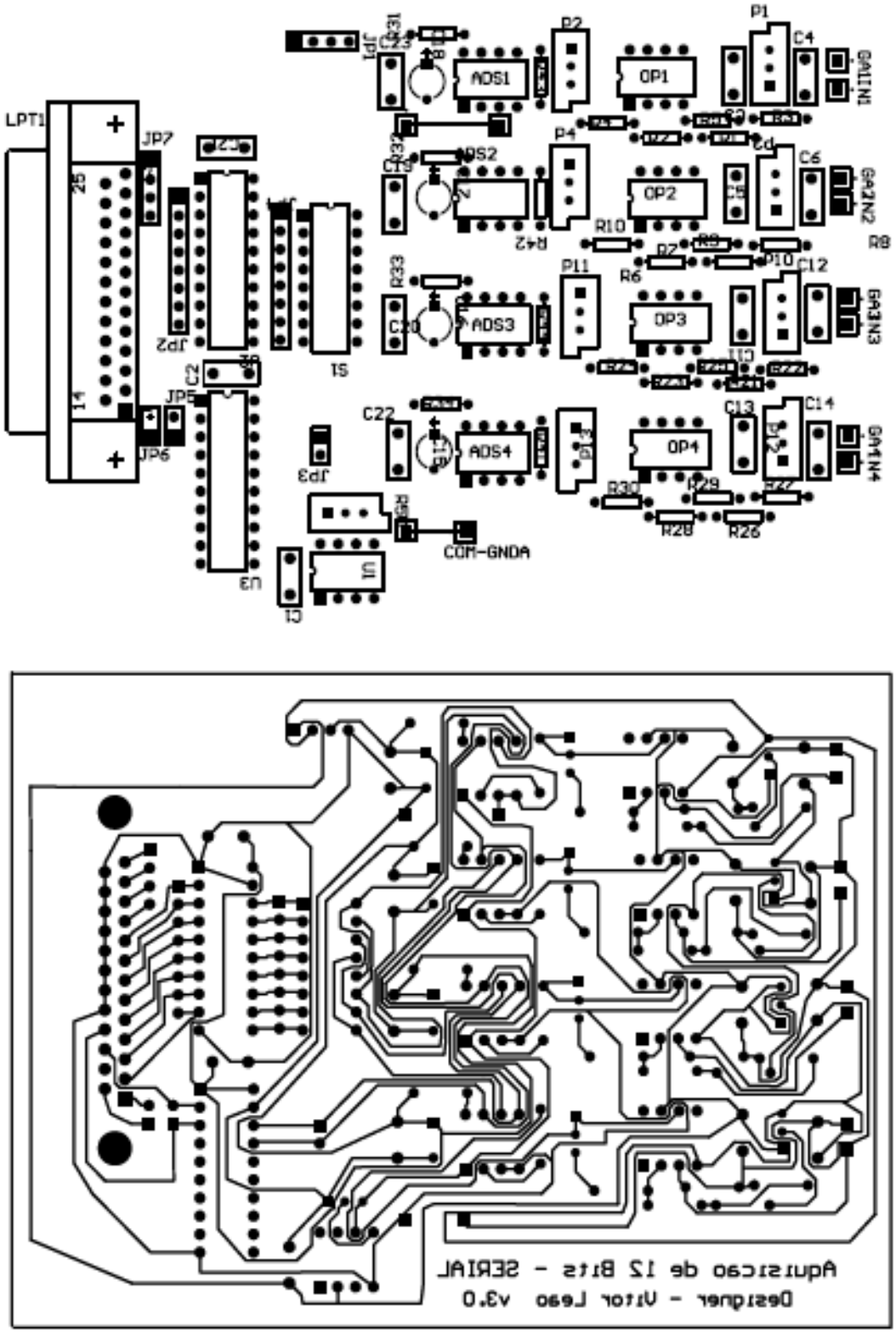

Figura A.2. Layout eletrônico da placa de aquisição de dados 12 bits

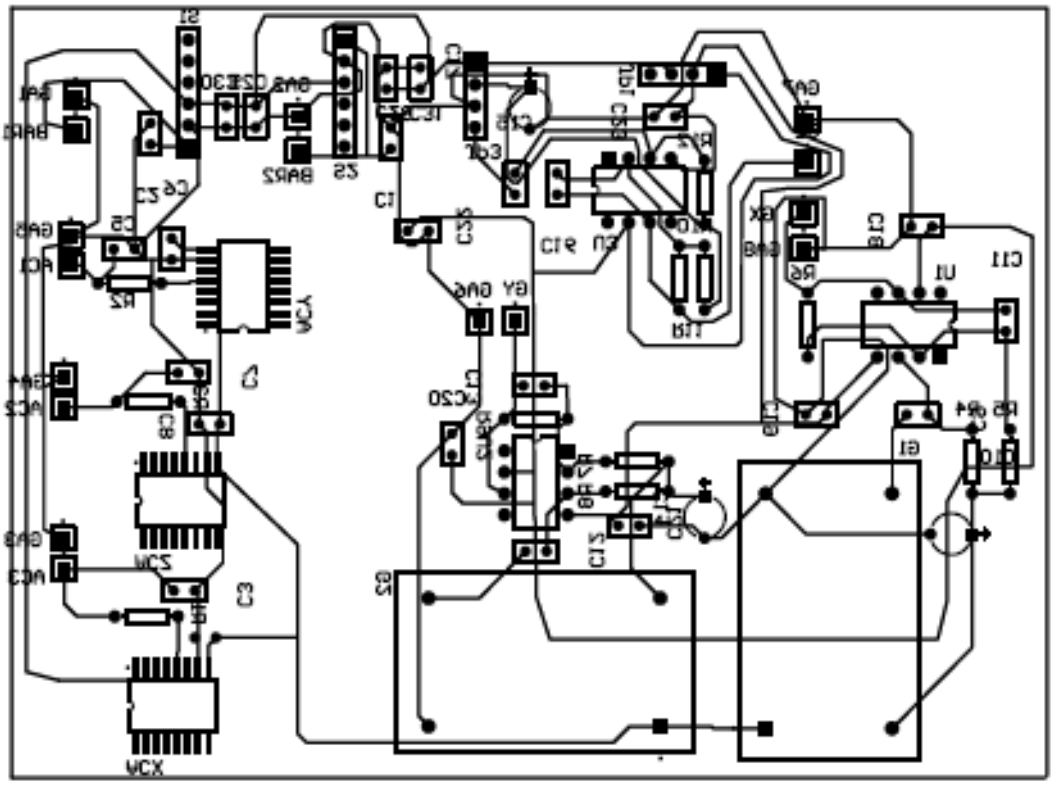

Figura A.3. Layout eletrônico da placa dos sensores

APÊNDICE B

PROGRAMAS DESENVOLVIDOS

B.1 INTERFACE PARALELA DE AQUISIÇÃO DE DADOS

Listagem do arquivo: main.cpp

```
#include <qapplication.h>
#include "aquisicao12c4.h"

int main( int argc, char ** argv )
{
    QApplication a( argc, argv );
    Aquisicao12serial w;
    w.show();
    w.initTimer();
    a.connect( &a, SIGNAL( lastWindowClosed() ), &a, SLOT( quit() ) );
    return a.exec();
}
```

Listagem do arquivo: aquisicao12c4.ui.h

```
#include <qtimer.h>
#include <math.h>
#include <qmessagebox.h>
#include <qpainter.h>
#include <qpen.h>
#include <qpoint.h>
#include <qdatetime.h>
```

```
#include <qsize.h>
#include "adcserial.h"
#include <qfiledialog.h>

QTimer *t;
QTime tempo;
ADC conversor;

int Maior_A,Maior_B,Menor_A,Menor_B,pontos=0,Npags=0;
float  Matriz_Valores[100][3490][9];

void Aquisicao12serial::initTimer()
{
    t = new QTimer( this );
}

void Aquisicao12serial::saveFile()
{

    // Dialogo que define o nome do arquivo //
    QString fileFilters = tr("Dados Adquiridos (*.dat)");
    QString fileName = QFileDialog::getSaveFileName(".dat", fileFilters, this,"Salvar Arquivo"
,"Defina um nome para o arquivo ser salvo");
    QFileInfo fi(fileName);
    QString name = fi.fileName();
 //   NomeFile->setText(name);

    // Teste se o arquivo pode ser salvo //
    QFile f( fileName );
    QTextStream stream(&f);

    if ( !f.open( IO_WriteOnly ) ) {
 QMessageBox::critical(this, tr("Aviso"), tr("Não foi possível iniciar a gravação
 do arquivo."),QMessageBox::Ok,QMessageBox::NoButton);
            return;
```

```
    }


     // Loop de salvamento dos dados //
    for(int i=0;i<=Npags;i++){
if (i==Npags){
    for(int j=0;j<pontos;j++){
for(int l=0;l<9;l++){
    stream << Matriz_Valores[i][j][l] << "\t";
    if(l==8) stream<<endl;
}
    }
}
else {
    for(int j=0;j<3490;j++){
for(int l=0;l<9;l++){
    stream << Matriz_Valores[i][j][l] << "\t";
    if(l==8) stream<<endl;
}
    }
}
    }
   //  Salva o arquivo no local indicado anteriormente //
    QMessageBox::information(this, tr("Aviso"), tr("Dados salvos com sucesso!"),
                            QMessageBox::Ok,QMessageBox::NoButton);
    f.close();

}

void  Aquisicao12serial::initLeitura()
{
  int ret=0;

    ret = ioperm(0x378, 3, 1);
    if (ret == -1)
```

```
    {
      QMessageBox::critical(this, tr("Aviso"), tr("Não foi possível o acesso a porta
 paralela!!\n","Favor checar suas permissões de acesso."),
QMessageBox::Ok,QMessageBox::NoButton);
       close();
   }

   conversor.inicializar_adc(); // Inicializa o modo epp
   connect( t, SIGNAL( timeout() ), SLOT( leitura_dados() ) );
   t->start( 0, FALSE );
   tempo.start();

   But_Inicio->setEnabled(0);
   But_parar->setText("&Parar");

}

void Aquisicao12serial::leitura_dados()
{
    int *valores;

    conversor.limpar_adc();
    valores=conversor.getValue();


    if (Channel_A->isChecked())
           {
// AQUISICAO DE DADOS DO CANAL A
               Valor_A->setText(QString("%1").arg(valores[0]));
// DESENHA UM PONTO NA INTERFACE
Plotar(pontos,(valores[0]+(-1*Offset_A->value()))*Ganho_A->value(),Cor_A->currentItem());
           }
    else{
            Valor_A->setText("--");
valores[0]=0;
```

```
}

    /* Faz a leutura do canal B*/
     if (Channel_B->isChecked())
           {
// AQUISICAO DE DADOS DO CANAL B
                Valor_B->setText(QString("%1").arg(valores[1]));
// DESENHA UM PONTO NA INTERFACE
 Plotar(pontos,(valores[1]+(-1*Offset_B->value()))*Ganho_B->value(),Cor_B->currentItem());
    }
     else {
 Valor_B->setText("--");
 valores[1]=0;
     }

  /* Faz a leutura do canal C*/
      if (Channel_C->isChecked())
           {
// AQUISICAO DE DADOS DO CANAL C
                Valor_C->setText(QString("%1").arg(valores[2]));
// DESENHA UM PONTO NA INTERFACE
Plotar(pontos,(valores[2]+(-1*Offset_C->value()))*Ganho_C->value(),Cor_C->currentItem());
           }
    else{
            Valor_C->setText("--");
            valores[2]=0;
}

    /* Faz a leutura do canal D*/
     if (Channel_D->isChecked())
           {
// AQUISICAO DE DADOS DO CANAL D
                Valor_D->setText(QString("%1").arg(valores[3]));
// DESENHA UM PONTO NA INTERFACE
Plotar(pontos,(valores[3]+(-1*Offset_D->value()))*Ganho_D->value(),Cor_D->currentItem());
```

```
           }
    else{
            Valor_D->setText("--");
            valores[3]=0;
          }
       /* Faz a leutura do canal E*/
     if (Channel_E->isChecked())
           {
// AQUISICAO DE DADOS DO CANAL E
               Valor_E->setText(QString("%1").arg(valores[4]));
// DESENHA UM PONTO NA INTERFACE
Plotar(pontos,(valores[4]+(-1*Offset_E->value()))*Ganho_E->value(),Cor_E->currentItem());
           }
    else{
            Valor_E->setText("--");
            valores[4]=0;
          }

    /* Faz a leutura do canal F*/
     if (Channel_F->isChecked())
           {
// AQUISICAO DE DADOS DO CANAL F
               Valor_F->setText(QString("%1").arg(valores[5]));
// DESENHA UM PONTO NA INTERFACE
Plotar(pontos,(valores[5]+(-1*Offset_F->value()))*Ganho_F->value(),Cor_F->currentItem());
           }
    else{
            Valor_F->setText("--");
            valores[5]=0;
          }

    /* Faz a leutura do canal G*/
     if (Channel_G->isChecked())
           {
// AQUISICAO DE DADOS DO CANAL G
```

```
                Valor_G->setText(QString("%1").arg(valores[6]));
// DESENHA UM PONTO NA INTERFACE
Plotar(pontos,(valores[6]+(-1*Offset_G->value()))*Ganho_G->value(),Cor_G->currentItem());
           }
    else{
            Valor_G->setText("--");
            valores[6]=0;
          }

       /* Faz a leutura do canal H*/
     if (Channel_H->isChecked())
           {
// AQUISICAO DE DADOS DO CANAL H
                Valor_H->setText(QString("%1").arg(valores[7]));
// DESENHA UM PONTO NA INTERFACE
Plotar(pontos,(valores[7]+(-1*Offset_H->value()))*Ganho_H->value(),Cor_H->currentItem());
           }
    else{
            Valor_H->setText("--");
            valores[7]=0;
          }


     /* INFORMAÇÕES ADICIONAIS DA AQUISIÇÃO */
     Npag->setText(QString("%1").arg(Npags));
     PT->setText(QString("%1").arg(pontos));
     TempoD->setText(QString("%1").arg(tempo.elapsed()*0.001));
     Matriz_Valores[Npags][pontos][0]=tempo.elapsed()*0.0001;
     Matriz_Valores[Npags][pontos][1]=valores[0];
     Matriz_Valores[Npags][pontos][2]=valores[1];
     Matriz_Valores[Npags][pontos][3]=valores[2];
     Matriz_Valores[Npags][pontos][4]=valores[3];
     Matriz_Valores[Npags][pontos][5]=valores[4];
     Matriz_Valores[Npags][pontos][6]=valores[5];
     Matriz_Valores[Npags][pontos][7]=valores[6];
```

```
     Matriz_Valores[Npags][pontos][8]=valores[7];

     pontos++;
     if (pontos>3490)
     {
           pontos=0;
           Npags++;
           paintRefresh();
       }

     usleep(SpinDelay->value()*1000);
    delete valores;

}

void Aquisicao12serial::parar()
{
   if (But_Inicio->isEnabled())
        close();
    else
   {
          t->stop();
         //retira as permissão
          conversor.finaliza_adc();
          ioperm(0x378, 3, 0);

          But_parar->setText("&Sair");
          Valor_A->setText("--");
          Valor_B->setText("--");
          Valor_C->setText("--");
          Valor_D->setText("--");
          Valor_E->setText("--");
          Valor_F->setText("--");
          Valor_G->setText("--");
          Valor_H->setText("--");
```

```
          But_Inicio->setEnabled(1);
      }
}

void Aquisicao12serial::limpatela()
{
   pontos=0;
   Npags=0;
   paintRefresh();
   TempoD->setText("0.000");
   Npag->setText(QString("%1").arg(Npags));
   PT->setText(QString("%1").arg(pontos));
   tempo.restart ();
}

void Aquisicao12serial::paintEvent( QPaintEvent * )
{

    QPainter paint( this );
    paint.setBrush(colorGroup().foreground() );
    drawGrid( &paint );
}

void Aquisicao12serial::drawGrid(QPainter *paint )
{
    paint->save();

    paint->setWindow( -900,-900, 8000,8000);

    QRect v = paint->viewport();
    paint->setViewport(0, 0, 1500,1500);


    paint->save();
    paint->drawRect(-880,-690,3500,3410);     // BACKGROUND
```

```
    QBrush b1( Qt::white, Qt::SolidPattern  );
    QPen pen (Qt::DotLine);

    paint->setBrush(b1);
    paint->drawRect(-860,-700,3500,3400);
    paint->setBrush(b1);
    paint->setPen(pen);
    paint->drawLine(-860,1000,2640,1000);
    paint->restore();

}

void Aquisicao12serial::Plotar(int pontoX, int pontoY, int cor)
{
   QPainter paint(this);
   paint.save();
   paint.setWindow( -900,-900,8000,8000);
   QRect v = paint.viewport();
   paint.setViewport(0, 0, 1500,1500);
   paint.save();



    switch (cor) {
        case 0:
            paint.setPen(yellow);
            break;
        case 1:
            paint.setPen(blue);
            break;
        case 2:
            paint.setPen(black);
            break;
        case 3:
```

```
            paint.setPen(green);
            break;
        case 4:
            paint.setPen(red);
            break;
        case 5:
            paint.setPen(gray);
            break;
        case 6:
            paint.setPen(magenta);
            break;
        case 7:
            paint.setPen(cyan);
            break;

        }
   paint.scale(1,0.74);
   paint.drawPoint (pontoX-850,1350-pontoY);
   paint.restore();

}

void Aquisicao12serial::paintRefresh()
{

   QPainter paint(this);
   paint.save();


    paint.setWindow( -900,-900, 8000,8000);

    QRect v = paint.viewport();
    paint.setViewport(0, 0, 1500,1500);
```

```
    paint.save();
    paint.drawRect(-880,-690,3500,3410);     // BACKGROUND

    QBrush b1( Qt::white, Qt::SolidPattern  );
    QPen pen (Qt::DotLine);

    paint.setBrush(b1);
    paint.drawRect(-860,-700,3500,3400);
    paint.setBrush(b1);
    paint.setPen(pen);
    paint.drawLine(-860,1000,2640,1000);
    paint.restore();
}
```

Listagem do arquivo: adcserial.h

```
#ifndef ADCSERIAL_H
#define ADCSERIAL_H
#include <unistd.h>
#include <sys/io.h>
#include <qapplication.h>

class ADC {
public:
          int *getValue(void);
           void limpar_adc(void);
           int Maior(int , int);
           int Menor(int , int);
                          void inicializar_adc(void);
           void finaliza_adc(void);
private:

};
#endif
```

Listagem do arquivo: adcserial.cpp

```
#include "adcserial.h"
#include <math.h>
#include <stdio.h>

/* Placa de Aquisição de Dados 12 Bits Serial */

/* DEFINICOES DA PORTA */
#define end1 0x378
#define end2 0x379
#define end3 0x37A

/* DEFINICOES DA PLACA DE ADC */
#define CS            4
#define DCLOCK  8
#define EPP         32

/*Inicializa a Porta Paralela para o Modo EPP*/
void ADC::inicializar_adc(void)
{
   outb(EPP,end3);
   return;
}

/*Finaliza a porta do modo EPP*/
void ADC::finaliza_adc(void)
{
   outb(0,end3);
   return;
}

/*COLOCA OS AD'S EM HI-Z*/
```

```
void ADC::limpar_adc(void)
{
   outb(CS|EPP,end3);
   return;
}

/* FAZ A AQUISIÇÃO DOS DADOS */
int  * ADC::getValue(void)
{

   int i, j,a,b,*dado;
   unsigned int CANAL_A[12];
   dado=new int[8];


     for(i=0;i<13;i++)
     { CANAL_A[i]=0; }

   /* LIGANDO O ADC  - HI-Z */
     outb(0|DCLOCK|EPP,end3);

   /* INICIANDO A AQUISIÇÃO DOS DADOS */
   for(i=1;i<15;i++)
     {
outb(DCLOCK|EPP,end3);
outb(0|EPP,end3);
     }
      /* FAZENDO A LEITURA DOS DADOS */
   for(i=0;i<12;i++)
     {
outb(DCLOCK|EPP,end3);
outb(0|EPP,end3);
CANAL_A[i]=inb(end1);
     }
//    /* FIM DA LEITURA DO CANAL A */
```

```
       outb(CS|EPP,end3);

  /* CANAL A,B,C, D, E, F e G - AGRUPAMENTO DOS BITS LIDOS */
       for(j=0;j<8;j++){
  dado[j]=0;
for(i=0;i<12;i++) {
                    dado[j] =dado[j]+(((CANAL_A[i]&(int)pow(2,j))>>j)*pow(2,i));
}
dado[j]=dado[j]-2048;
 }

       return (dado);
}



int ADC::Maior(int valorA, int valorB)
{
  int Maior=0;
    if (valorA > valorB)
Maior=valorA;
    else
Maior=valorB;

    return (Maior);
}

int ADC::Menor(int valorA, int valorB)
{
    int Menor=0;

    if (valorA < valorB)
Menor=valorA;
    else
Menor=valorB;
```

```
    return (Menor);
}
```

B.2 INTERFACE SERIAL DE COMUNICAÇÃO DE DADOS

Listagem do arquivo: main.cpp

```
#include <qapplication.h>
#include "fControlador.h"

int main( int argc, char ** argv )
{
    QApplication a( argc, argv );
    frmControlador w;
    w.show();
    w.initTimer();
    a.connect( &a, SIGNAL( lastWindowClosed() ), &a, SLOT( quit() ) );
    return a.exec();
}
```

Listagem do arquivo: fControlador.ui.h

```
#include <qfiledialog.h>
#include <qtimer.h>
#include <math.h>
#include <qmessagebox.h>
#include <qpainter.h>
#include <qpen.h>
#include <qpoint.h>
#include <qdatetime.h>
#include <qsize.h>
#include "serial.h"
#include "joy.h"
#include "fuzzy.h"
```

```
#include "sensores.h"
#include <stdlib.h>
#include <string.h>


//#include "filesave.xpm"
#define ADC_ref 0.0012207
#define ganho 10
#define g 9.73915 // aceleração da gravidade


Serial s;
Joy joy;
Fuzzy ConjFuzzy;
Sensores sensor;
QTimer *t;
QTime tempo;
int pontos=0,Npags=0;
float Matriz_Valores[100][2120][13],veloc[100][2120][6],filtro[100][2120][3];
//float Scalibrado[3];
// Sensibilidade dos Acelerometros (G/Volts)
float Repouso[3]={2.4932, 2.4580 , 2.4820};
float Seb[3]={0.0084, 0.0084, 0.0084};

void frmControlador::saveFile()
{

    // Dialogo que define o nome do arquivo //
    QString fileFilters = tr("Dados Adquiridos (*.dat)");
    QString fileName = QFileDialog::getSaveFileName(".dat", fileFilters, this,"Salvar
 Arquivo","Defina um nome para o arquivo ser salvo");
    QFileInfo fi(fileName);
    QString name = fi.fileName();
    NomeFile->setText(name);
```

```
    // Teste se o arquivo pode ser salvo //
    QFile f( fileName );
    QTextStream stream(&f);

    if ( !f.open( IO_WriteOnly ) ) {
 QMessageBox::critical(this, tr("Aviso"), tr("Não foi possível iniciar a
gravação do arquivo."), QMessageBox::Ok,QMessageBox::NoButton);
             return;
    }


   // stream << "#********** Sistema de Aquisicao UAV
 **************" << endl;
 stream << "#TEMPO  AD1 AD2 AD3 AD4 AD5  AD6  AD7 AD8
 Profundor Desejada Erro Derivada " << endl;


     for(int i=0;i<=Npags;i++){
if (i==Npags){
    for(int j=0;j<pontos;j++){
for(int l=0;l<13;l++){
    stream <<Matriz_Valores[i][j][l] << "\t";
    if(l==12) stream<<endl;
}
    }
}
else {
    for(int j=0;j<2120;j++){
for(int l=0;l<13;l++){
    stream << Matriz_Valores[i][j][l] << "\t";
    if(l==12) stream<<endl;
}
    }
}
    }
```

```
     //  Salva o arquivo no local indicado anteriormente //
   // stream << "#******* por Vitor Leao **********************"
 << endl;
    QMessageBox::information(this, tr("Aviso"), tr("Dados salvos com sucesso!"),
                           QMessageBox::Ok,QMessageBox::NoButton);
    f.close();

    // Salva os dados de aceleração e velocidade
    // Teste se o arquivo pode ser salvo //
   QFileInfo fo(fileName);
   name = fo.baseName();
   name.append("_vel.dat");
   NomeFile->setText(name);
   QFile fx( name );
   QTextStream streamx(&fx);

    if ( !fx.open( IO_WriteOnly ) ) {
 QMessageBox::critical(this, tr("Aviso"), tr("Nãoo foi possível iniciar a
gravação do arquivo."),QMessageBox::Ok,QMessageBox::NoButton);
             return;
    }


   // stream << "#******* Sistema de Aquisicao UAV ***********" <<
 endl;
 streamx << "#TEMPO  Velx Vely Velz ACx ACy ACz" << endl;


     for(int i=0;i<=Npags;i++){
if (i==Npags){
    for(int j=0;j<pontos;j++){
streamx << Matriz_Valores[i][j][0] << "\t";
for(int l=0;l<6;l++){
    streamx << veloc[i][j][l] << "\t";
```

```
    if(l==5) streamx << endl;
}
    }
}
else {
    for(int j=0;j<2120;j++){
    streamx << Matriz_Valores[i][j][0] << "\t";
for(int l=0;l<6;l++){
    streamx << veloc[i][j][l] << "\t";
    if(l==5) streamx << endl;
}
    }
}
    }

     //  Salva o arquivo no local indicado anteriormente //
   // stream << "########### por Vitor Leao **************" <<
 endl;
    QMessageBox::information(this, tr("Aviso"), tr("Dados salvos com sucesso!"),
                           QMessageBox::Ok,QMessageBox::NoButton);
    fx.close();
    limpatela();

}

void frmControlador::initTimer()
{
    t = new QTimer( this );
}

void frmControlador::parar()
{
   if (Bt_Start->isEnabled())
        close();
```

```
    else
   {
          t->stop();
          Sair->setText("&Sair");
          Bt_Start->setEnabled(1);
          s.closePort();
      }
}

void frmControlador::connectSerial(){
    int resp = s.openPort(comboBox1->currentText());
    s.setPort();

    if (resp == 0)
    {
Sair->setText("&Parar");
Bt_Start->setEnabled(0);

   // Loop que limpa a matriz dos dados //
    for(int i=0;i<100;i++){
for(int j=0;j<2120;j++){
    for(int l=0;l<13;l++){
Matriz_Valores[i][j][l] =0;
    }
}
    }
           connect( t, SIGNAL( timeout() ), SLOT( leitura_dados() ) );
            t->start( 0, FALSE );
           tempo.start();


    }
    else
    {
QMessageBox::critical(this, tr("Aviso"), tr("Não foi possível o acesso a
```

```
porta serial!!\n","Favor checar suas permissões de acesso."),
QMessageBox::Ok,QMessageBox::NoButton);
    }
   resp=0;

    resp = ioperm(0x378, 3, 1);
    if (resp == -1)
    {
      QMessageBox::critical(this, tr("Aviso"), tr("Não foi possível o acesso a porta
 paralela!!\n",  "Favor checar suas permissões de acesso."),
               QMessageBox::Ok,QMessageBox::NoButton);
       close();
   }

        joy.On_Joy();  //Liga o Joystick
}

void frmControlador::leitura_dados(){

    int ret = 0,i,j,dado[8],*controle,temp[2][8];
    char MatSaida[11], MatEntrada[35],wbyte[5],*endptr;
   unsigned char ch;
   float erro=0,derivada=0,profundor=0,deltaT=0,temperatura,a,b,c,d;

   // Leitura do Joystick
   controle=joy.le_dado();
   for(j=3;j<5;j++){
       for(i=0;i<8;i++){
   temp[j-3][i]=(controle[j]&(int)pow(2,i))>>i;
       }
   }
  // Acinamendo dos Motores por Joystick
   // Motor 1
   if (temp[0][4]==0)       dialM1->addLine();
   if (temp[0][6]==0)       dialM1->subtractLine();
```

```
   // Motor 2
   if (temp[0][5]==0)       dialM2->addLine();
   if (temp[0][7]==0)       dialM2->subtractLine();

   // Motor 3
   if (temp[1][4]==0)       dialM3->addLine();
   if (temp[1][6]==0)       dialM3->subtractLine();

   // Motor 4
   if (temp[1][5]==0)       dialM4->addLine();
   if (temp[1][7]==0)       dialM4->subtractLine();

   // Motor 5
   if (temp[1][3]==0)       dialM5->addLine();
   if (temp[1][1]==0)       dialM5->subtractLine();

   // Motor 6
   if (temp[1][2]==0)       dialM6->addLine();
   if (temp[1][0]==0)       dialM6->subtractLine();

   // Motor 7
   if (temp[0][0]==0)       dialM7->addLine();
   if (temp[0][3]==0)       dialM7->subtractLine();


//   //    MatSaida[0]='@';
    MatSaida[0]=dialM1->value();
    MatSaida[1]=dialM2->value();
    MatSaida[2]=dialM3->value();
    MatSaida[3]=dialM4->value();
    MatSaida[4]=dialM5->value();
    MatSaida[5]=dialM6->value();
    MatSaida[6]=dialM7->value();
    MatSaida[7]=dialM8->value();
```

```
    MatSaida[8]='\0';


// Envio dos Comandos para os servos
ret = s.sendCommand(MatSaida);
if (ret == -1) // erro no envio do comando
    QMessageBox::critical(this, tr("Aviso"), tr("Problemas na
 Comunicaçãoo!!!!!."),QMessageBox::Ok,QMessageBox::NoButton);

for(i=0;i<35;i++){
    ch=s.readByte();
    MatEntrada[i] = ch;
}

for(i=0;i<8;i++)dado[i]=0;

// /* Montagem dos dados Lidos */
j=0;
if (MatEntrada[0]==';' & MatEntrada[1]==';' & MatEntrada[34]==63){
    for(i=2;i<32;i=i+4){
wbyte[0]=MatEntrada[i];
wbyte[1]=MatEntrada[i+1];
wbyte[2]=MatEntrada[i+2];
wbyte[3]=MatEntrada[i+3];
wbyte[4]='\0';
dado[j]=strtol(wbyte,&endptr,16)-2047;
//printf("%d /t",dado[j]);
j++;
  }
    txtSerial->setText("SERIAL->OK!");
    temperatura=(dado[1]+2047)*ADC_ref*9;
    TempoD->setText(QString("%1").arg(tempo.elapsed()*0.001));
    Npag->setText(QString("%1").arg(Npags));
    PT->setText(QString("%1").arg(pontos));
    labTemp->setText(QString("%1").arg(temperatura ,0, 'f', 4));
```

```
    labAltitude->setText(QString("%1").arg(sensor.altitude
(dado[0]+2047,temperatura),0, 'f', 4));
    labAltitude2->setText(QString("%1").arg(sensor.altitude
(dado[4]+2047,temperatura),0, 'f', 4));

// Controlador Fuzzy
    if (rbtnAtiva->isChecked()){
erro=sensor.altitude(dado[0]+2047,temperatura)-spDesejada->value();
// Calculo da derivada
deltaT=tempo.elapsed()*0.001-Matriz_Valores[Npags][pontos-1][0];
derivada=(erro-Matriz_Valores[Npags][pontos-1][11])/erro;
// Calculo Fuzzy
profundor=ConjFuzzy.FuzzyFunc(erro,derivada);
dialM2->setValue(profundor);
labAltitudes->setText(labAltitude->text());
labProfundor->setText(QString("%1").arg(profundor));
labErro->setText(QString("%1").arg(erro));
labDerivada->setText(QString("%1").arg(derivada));
    }

// Aplica o filtro nos acelerômetros e calcula a velocidade
    if ((pontos>=1)||((pontos==0)&&(Npags>=1))) {
          for (i=5;i<8;i++){
// tempo decorrido
a=tempo.elapsed()*0.001;
b=Matriz_Valores[Npags][pontos-1][0];
// Valores atual e passado
c=(dado[i]+2047)*ADC_ref-Repouso[i-5]; // Valor em tensão
d=(Matriz_Valores[Npags][pontos-1][i+1]+2047)*ADC_ref-Repouso[i-5];
// Valor em tensão
filtro[Npags][pontos][i-5]=sensor.filtro_PA(c,d,filtro[Npags]
[pontos-1][i-5], 1./ (a-b));
veloc[Npags][pontos][i-5]=sensor.filtro_PA_Veloc(filtro[Npags]
[pontos][i-5],filtro[Npags][pontos-1][i-5],
veloc[Npags][pontos-1][i-5], 1./ (a-b),Seb[i-5]);
```

```
        }
      }
    else{
                        for (i=5;i<8;i++){
// tempo decorrido
a=tempo.elapsed()*0.001;
b=0;
// Valores atual e passado
c=(dado[i]+2047)*ADC_ref-Repouso[i-5]; // Valor em tensï¿½
d=0; // Valor em tensï¿½
filtro[Npags][pontos][i-5]=sensor.filtro_PA(c,d,0, 1./ (a-b));
veloc[Npags][pontos][i-5]=sensor.filtro_PA_Veloc(filtro[Npags]
[pontos][i-5],0,0, 1./ (a-b),Seb[i-5]);
         }
    }


    Matriz_Valores[Npags][pontos][0]=tempo.elapsed()*0.001;
    Matriz_Valores[Npags][pontos][1]=dado[0];
    Matriz_Valores[Npags][pontos][2]=temperatura;
    Matriz_Valores[Npags][pontos][3]=dado[2];
    Matriz_Valores[Npags][pontos][4]=dado[3];
    Matriz_Valores[Npags][pontos][5]=dado[4];
    Matriz_Valores[Npags][pontos][6]=dado[5];
    Matriz_Valores[Npags][pontos][7]=dado[6];
    Matriz_Valores[Npags][pontos][8]=dado[7];
    Matriz_Valores[Npags][pontos][9]=profundor;
    Matriz_Valores[Npags][pontos][10]=spDesejada->value();
                Matriz_Valores[Npags][pontos][11]=erro;
                Matriz_Valores[Npags][pontos][12]=derivada;
                veloc[Npags][pontos][3]=((dado[5]+2047)*ADC_ref-Repouso[0])/Seb[2]*g;
    veloc[Npags][pontos][4]=((dado[6]+2047)*ADC_ref-Repouso[0])/Seb[2]*g;
    veloc[Npags][pontos][5]=((dado[7]+2047)*ADC_ref-Repouso[0])/Seb[2]*g;

                P1->setText(QString("%1").arg(dado[0]));
```

```
    GYx->setText(QString("%1").arg((dado[1]+2047)*ADC_ref,0, 'f', 4));
    GYy->setText(QString("%1").arg(dado[2]));
    GYz->setText(QString("%1").arg(dado[3]));
    P2->setText(QString("%1").arg(dado[4]));
        ACx->setText(QString("%1").arg((((dado[5]+2047)*
ADC_ref-Repouso[0])/Seb[2]*g), 0, 'f', 4));
    ACy->setText(QString("%1").arg((((dado[6]+2047)*
ADC_ref-Repouso[1])/Seb[2]*g), 0, 'f', 4));
    ACz->setText(QString("%1").arg((((dado[7]+2047)*
ADC_ref-Repouso[2])/Seb[2]*g), 0, 'f', 4));
    // Dados de AquisiÃ§Ã£o
    Lat->setText(QString("%1").arg((dado[5]+2047)*ADC_ref, 0, 'f', 4));
    Long->setText(QString("%1").arg((dado[6]+2047)*ADC_ref, 0, 'f', 4));
    Status->setText(QString("%1").arg((dado[7]+2047)*ADC_ref, 0, 'f', 4));
    NS->setText(QString("%1").arg(Repouso[0], 0, 'f', 4));
    WE->setText(QString("%1").arg(Repouso[1], 0, 'f', 4));
    Data->setText(QString("%1").arg(Repouso[2], 0, 'f', 4));
    Curso->setText(QString("%1").arg(Seb[0], 0, 'f', 4));
    Vel->setText(QString("%1").arg(Seb[1], 0, 'f', 4));
    Hora->setText(QString("%1").arg(Seb[2], 0, 'f', 4));

        // DESENHA UM PONTO NA INTERFACE
    if (rbVeloc->isChecked()){
for (i=0;i<3;i++){
    Plotar(pontos,abs(veloc[Npags][pontos][i]),i); }
ACx->setText(QString("%1").arg((veloc[Npags][pontos][0])*3.6, 0, 'f', 4));
ACy->setText(QString("%1").arg((veloc[Npags][pontos][1])*3.6,0, 'f', 4));
ACz->setText(QString("%1").arg((veloc[Npags][pontos][2])*3.6,0, 'f', 4));
    }
    else{
for (i=0;i<8;i++){
Plotar(pontos,dado[i],i);}
    }
```

```
      // Controle da Interfase dos GrÃ¡ficos
    pontos++;

    if (pontos>2120){
pontos=0;
Npags++;
paintRefresh();
    }

}
// }
else{
    txtSerial->setText("SERIAL-ERRO");
}


 usleep(500); // Delay de 50us
}

void frmControlador::paintEvent( QPaintEvent * )
{

    QPainter paint( this );
    paint.setBrush(colorGroup().foreground() );
    drawGrid( &paint );
}

void frmControlador::drawGrid(QPainter *paint )
{
    paint->save();

    paint->setWindow(-900,-1650, 5000,5000);

    QRect v = paint->viewport();
    paint->setViewport(0, 0, 1500,1500);
```

```
    paint->save();
    paint->drawRect(-870,-695,2115,885);     // BACKGROUND

    QBrush b1( Qt::white, Qt::SolidPattern  );
    QPen pen (Qt::DotLine);

    paint->setBrush(b1);
    paint->drawRect(-885,-700,2120,880);
    paint->setBrush(b1);
    paint->setPen(pen);
    paint->drawLine(-885,-260,1235,-260);
    paint->restore();

}

//void frmControlador::Plotar(int pontoX1, int pontoY1, int
 pontoX, int pontoY, int cor)
void frmControlador::Plotar(int pontoX, int pontoY, int cor)
{
   QPainter paint(this);
   paint.save();
   paint.setWindow(-900,-1650, 5000,5000);
   QRect v = paint.viewport();
   paint.setViewport(0, 0, 1500,1500);
   paint.save();



    switch (cor) {
        case 0:
            paint.setPen(yellow);
            break;
        case 1:
```

```
            paint.setPen(blue);
            break;
        case 2:
            paint.setPen(black);
            break;
        case 3:
            paint.setPen(green);
            break;
        case 4:
            paint.setPen(red);
            break;
        case 5:
            paint.setPen(gray);
            break;
        case 6:
            paint.setPen(magenta);
            break;
        case 7:
            paint.setPen(cyan);
            break;

        }
   paint.scale(1,0.20);
   paint.drawPoint (pontoX-885,(-1*pontoY)-1300);
 //    paint.drawLine
// (pontoX1-885,(-1*pontoY1)-1300,pontoX-885,(-1*pontoY)-1300);

   paint.restore();

}

void frmControlador::paintRefresh()
{

   QPainter paint(this);
```

```
    paint.save();

    paint.setWindow(-900,-1650, 5000,5000);

    QRect v = paint.viewport();
    paint.setViewport(0, 0, 1500,1500);



    paint.save();
    paint.drawRect(-870,-695,2115,885);     // BACKGROUND

    QBrush b1( Qt::white, Qt::SolidPattern  );
    QPen pen (Qt::DotLine);

    paint.setBrush(b1);
    paint.drawRect(-885,-700,2120,880);
    paint.setBrush(b1);
    paint.setPen(pen);
    paint.drawLine(-885,-260,1235,-260);
    paint.restore();

}

void frmControlador::Calibrar()
{
    int i,k,tipo=5;
    float  valores_medios[6];
    char temp[100];
    // Zera a matriz dos valores de saida
    for (k=0;k<6;k++){
       valores_medios[k]=0;
    }
    s.setPort();
```

```
    for (k=0;k<6;k++){

switch (k) {
case 0:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
 sistema para cima"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=6;
    break;

case 1:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
 sistema para baixo"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=6;
     break;

case 2:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
 sistema virado p/ esquerda"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=7;
    break;

case 3:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
 sistema virado p/ direita"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=7;
     break;

case 4:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
 sistema ao plano"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=8;
     break;

case 5:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
```

```
 sistema gire 180"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=8;
    break;
}
   Npags=0;
               pontos=0;
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Aguarde a
 aquisição de 200 pontos e tecle OK"),QMessageBox::NoButton);
for(i=0;i<200;i++){
          valores_medios[k]=(valores_medios[k]+(Matriz_Valores[0][i][tipo]+2047)*
ADC_ref);
      }
    sprintf(temp,"k=%d %2.4f\n",k,valores_medios[k]/200);
    QMessageBox::information(this, tr("Valores lidos"),temp,QMessageBox::NoButton);

}

    Seb[0]=((valores_medios[0]/200)-(valores_medios[1]/200))/2;
    Seb[1]=((valores_medios[2]/200)-(valores_medios[3]/200))/2;
    Seb[2]=((valores_medios[4]/200)-(valores_medios[5]/200))/2;

    sprintf(temp,"ACx %2.4f\n ACy %2.4f\n ACz %2.4f\n",Seb[0], Seb[1], Seb[2]);

    QMessageBox::information(this, tr("Sensibilidade dos AcelerÃ´metros"),temp,QMessageBox::Ok,QM
    for (k=0;k<3;k++){

switch (k) {
case 0:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
 sistema para cima"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=8;
    break;

case 1:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
```

```
 sistema ao plano"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=6;
     break;

case 2:
    QMessageBox::information(this, tr("Processo de Calibração"), tr("Coloque o
 sistema ao plano"),QMessageBox::Ok,QMessageBox::NoButton);
    tipo=7;
    break;
}

Npags=0;
            pontos=0;
            valores_medios[k]=0;
QMessageBox::information(this, tr("Processo de Calibração"), tr("Aguarde a
 aquisição de 100 pontos e tecle OK"),QMessageBox::NoButton);
for(i=0;i<100;i++){
          valores_medios[k]=(valores_medios[k]+(Matriz_Valores[0][i][tipo]+2047)*
ADC_ref);
      }
    sprintf(temp,"%2.4f\n",valores_medios[k]/100);
    QMessageBox::information(this, tr("Valores lidos"),temp,QMessageBox::NoButton);

}

    Repouso[0]=valores_medios[1]/100; // Valor de repouso - EIXO X
    Repouso[1]=valores_medios[2]/100; // Valor de repouso - EIXO Y
    Repouso[2]=valores_medios[0]/100; // Valor de repouso - EIXO Z

    sprintf(temp,"VRepX %2.4f\n VRepY %2.4f\n VRepZ %2.4f\n",Repouso[0], Repouso[1],
 Repouso[2]);

    QMessageBox::information(this, tr("Valor de
 Repouso"),temp,QMessageBox::Ok,QMessageBox::NoButton);
}
```

```
void frmControlador::limpatela()
{
   pontos=0;
   Npags=0;
   paintRefresh();
   TempoD->setText("0.000");
   Npag->setText(QString("%1").arg(Npags));
   PT->setText(QString("%1").arg(pontos));
   tempo.restart ();
}
```

Listagem do arquivo: sensores.h

```
#ifndef Sensores_H
#define Sensores_H

#include <math.h>


class Sensores
{
public:
 float Sensores::altitude(int valor, float temperatura);
 float Sensores::filtro_PA(float yDado, float yDado1, float yDadoK1, float freq);
 float Sensores::filtro_PA_Veloc(float yDadoV, float yDado1V, float yDadoK1V,
float freq, float sb);
};
#endif //Sensores_h
```

Listagem do arquivo: sensores.cpp

```
/************************************************************
*****  Cálculos para os sensores do sistema
*****  Por: Vitor Leão
```

```
*****  Data: 18/11/2005
*****  Implementados:
*****               Sensor de Pressão
*****        Cálculo da Frequencia de Corte e da Integral:
*****                      - Acelerômetro
*****                      - Velocidade
************************************************************/
#include "sensores.h"

#define fc 8 // Frequencia de corte do filtro passa alta
#define g 9.73915 // aceleração da gravidade



float Sensores::altitude(int valor, float temperatura){
   float R,lambda=0.0064997,Po=101.29,P,altura=0;
   temperatura=temperatura+273.15;
   P=0.0271267*valor+10.5555556;
   R=Po*1000/(temperatura*1.2251);
   altura=temperatura/lambda*pow((1-P/Po),lambda*R);

    return(altura);
}

float Sensores::filtro_PA(float yDado, float yDado1, float yDadoK1, float freq){
    // Filtro Passa-Alta dos acelerômetros //
       float yDadok,af,bf;

       af=(2*freq-fc)/(2*freq+fc);
       bf=(2*freq)/(2*freq+fc);

       yDadok = af * yDadoK1 + bf * (yDado - yDado1);

       return(yDadok);
}
```

```
float Sensores::filtro_PA_Veloc(float yDadoV, float yDado1V, float yDadoK1V, float freq,
 float sb){
    // Calculo da Integral para as velocidades //
       float yDadokV,bf=1/(2*freq);

       yDadokV = yDadoK1V + (bf*(yDadoV - yDado1V) *g)/sb;

       return(yDadokV);
}
```

Listagem do arquivo: serial.h

```
#ifndef SERIAL_H
#define SERIAL_H

#include <termios.h>
#include <fcntl.h>
#include <sys/signal.h>
#include <qstring.h>
#include <sys/ioctl.h>

#define TRUE            1
#define FALSE          0
#define NONBLOCK  0
#define BLOCK         1

class Serial
{
public:
    struct termios oldtio, newtio;
    struct sigaction saio;
    int s_port;
    char port_name[32];
```

```
    Serial();
    ~Serial();
    int openPort(QString portName);
    void setPort();
    void closePort();
    unsigned char readByte();
    int readNum();
    int sendCommand(QString text);
};
#endif //serial_h
```

Listagem do arquivo: serial.cpp

```
#include "serial.h"

Serial::Serial() {}

Serial::~Serial() {}

int Serial::openPort(QString portName)
{
    s_port = open(portName, O_RDWR | O_NOCTTY );
    if (s_port == -1) return(-1);

    return(0);
}

void Serial::setPort()
{
//    int status;

  // tcgetattr(s_port, &oldtio);
 //newtio=oldtio;
    /*
          BAUDRATE: Set bps rate. You could also use cfsetispeed and cfsetospeed.
          CRTSCTS : output hardware flow control (only used if the cable has
```

```
                    all necessary lines. See sect. 7 of Serial-HOWTO)
          CS8     : 8n1 (8bit,no parity,1 stopbit)
          CLOCAL  : local connection, no modem contol
          CREAD   : enable receiving characters
        */
   newtio.c_cflag = B9600 | CLOCAL | CREAD;
   newtio.c_cflag &= ~PARENB;
   newtio.c_cflag &= ~CSTOPB;
   newtio.c_cflag &= ~CSIZE;
   newtio.c_cflag |= CS8;
   newtio.c_cflag &= ~CRTSCTS;


//         tcflush(s_port, TCIFLUSH);
         tcsetattr(s_port,TCSAFLUSH,&newtio);

}

//lê um caracter
unsigned char Serial::readByte()
{
    int res;
    unsigned char buf[2];

    res = read(s_port, buf, 1);
    buf[1] = '\0';

    if ( res == 1 ) return(buf[0]);
    else return(0);
}

int Serial::sendCommand(QString text)
{
  int n=0;
```

```
     n = write(s_port, text,strlen(text));
 //    printf("%d\n",strlen(text));
     if (n < 0) return(-1);
     else
return(0);
}

void Serial::closePort()
{
    close(s_port);
}

int Serial::readNum()
{
    unsigned char one, two, three, four;

    one = readByte();
    if ( one == 0 ) return(-1);
    two = readByte();
    if ( two == 0 ) return(-1);
    if ( two==' ' || two == '\r' ) return( (int) ( one - '0' ) );

    three = readByte();
    if ( three == 0 ) return -1;
    if ( three == ' ' || three == '\r' ) return( (int) ( one - '0' ) * 10 + ( two - '0' ) );

    four = readByte();
    if ( four == 0 ) return -1;
    return( (int) ( one - '0' ) * 100 + ( two - '0' ) * 10 + ( three - '0' ) );
}
```

Listagem do arquivo: joy.h

```
#ifndef Joy_H
#define Joy_H
```

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/io.h>
#include <math.h>


class Joy
{
public:
 int *Joy::le_dado(void);
  void Joy::On_Joy(void);
  void Joy::Off_Joy(void);
};
#endif //Joy_h
```

Listagem do arquivo: joy.cpp

```
/*****************************************************************************
*****  Programa de Controle da Porta Paralela para Leitura de um Joystick PSX
*****  Por: Vitor Leão
*****  Data: 22/09/2005
*****************************************************************************/
#include "joy.h"

/* DEFINICOES DA PORTA */
#define end1 0x378   // DATA
#define end2 0x379 // STATUS
#define end3 0x37A // CONTROL

/* Comando para o Joystick */
#define COMMAND 1  // PINO 2  - DATA
#define ATT         2  // PINO 3 - DATA
#define CLOCK     4  // PINO 4 - DATA
#define VCC     248  // PINO 5,6,7,8,9 - DATA
```

```
#define DATA    64  // PINO 10 - STATUS
#define ACK      32  // PINO 12 - STATUS
#define SAIR        0 // Desligar o Joystick

int * Joy::le_dado(void)
{
   int i,valor[5]={0x01,0x42,0xFF,0xFF,0xFF},saida,*dado;

   dado=new int[8];

   outb(0|COMMAND|VCC|CLOCK|ATT,end1); // INICIALIZAÇÃO DO JOYSTICK
   outb(0|COMMAND|VCC|CLOCK,end1);

   for (int j=0;j<5;j++){
       dado[j]=0;
       for(i=0;i<8;i++) {
   saida=((valor[j]&(int)pow(2,i))>>i);        // MONTA O VALOR DE COMANDO
   outb(0|saida|VCC,end1);                        // CLOCK BAIXO
               outb(0|saida|VCC|CLOCK,end1); // ESCREVE UM DO COMANDO
   dado[j]=((inb(end2)&DATA)>>6)*(int)pow(2,i)+dado[j];           // LÊ DADO
   }
     usleep(8);
   }

       outb(0|COMMAND|VCC|CLOCK|ATT,end1); // INICIALIZAÇÃO DO JOY
       return (dado);
}
/************** Liga o Joystick ************************/
void Joy::On_Joy(void){
  outb(0|VCC|ATT,end1); // Liga o joystick
  return;
}

/************** Desliga o Joystick ************************/
void Joy::Off_Joy(void){
```

```
outb(SAIR,end1); // Liga o joystick
return;    }
```