Projeto de iniciação científica – PIBIC 2003

Bolsista:	Carlos Alberto Cascão Júnior	Matricula:	00/15776
Curso:	Engenharia Mecatrônica		
Orientador:	Geovany Araújo Borges	Matricula:	999636

Titulo do projeto: Estação de trabalho para experimentação de controle de processos em tempo real

Titulo do plano de trabalho: Estação de trabalho para experimentação de controle de processos em tempo real

Período: Agosto/2003 a Julho/2004

Laboratório de Controle e Visão por Computador – LCVC
Departamento de Engenharia Elétrica – ENE
Faculdade de Tecnologia – Universidade de Brasília
Caixa Postal 04591 – Asa Norte
CEP 70910-900
Brasília - DF

1. Apreciação do bolsista pelo orientador

Como orientador do aluno Carlos Alberto Cascão Júnior, acompanhei de perto suas atividades de iniciação científica neste projeto. O bolsista apresentou desenvoltura para apreender conceitos que somente lhe seriam passados em disciplinas mais adiantadas do curso, ou mesmo de nível de mestrado. Em momentos de dificuldade, tinha a iniciativa de tentar primeiramente resolver o problema por si. Caso não tivesse sucesso, o bolsista me apresentava o problema e conjuntamente chegávamos a uma solução. Seu entusiasmo e motivação não cessaram neste período, o que permitiu o bolsista alcançar o elevado nível de conhecimento sobre o problema em questão. Assim, o bolsista Carlos Alberto Cascão Júnior mostrou ter uma boa capacidade de aprendizado técnico rápido, bem como ter afinidade com atividades científicas, apesar da literatura consultada não ser muito vasta.

2. Introdução

O projeto tem como proposta a criação de uma estação de trabalho baseada em microcomputador para o controle de processos físicos em tempo real, implementação de técnicas de controle digital e construção de um módulo experimental de processo. Neste caso, o pêndulo invertido foi escolhido como processo a ser implementado. Este projeto possui grande relevância por ser comumente empregado em processos industriais, por apresentar uma implementação em arquitetura aberta, por procurar reduzir os custos de fabricação da placa de interface e, ainda, por representar um ótimo meio de contato com a prática de controle no meio acadêmico.

No projeto, faz-se necessária a concepção de um módulo de E/S, este módulo permite ao computador gerenciar a tarefa de aquisição de dados e controle do processo. O diagrama simplificado da estação de trabalho pode ser visto na Figura 1. Tal placa apresenta as seguintes características:

- 8 entradas analógicas de resolução 12 *bits* limitadas ao intervalo de –2,5V a +2,5V (pouco mais de 1mV de resolução);
- 4 saídas analógicas de resolução 12 *bits* limitadas ao intervalo de –2,5V a +2,5V;
- 2 saídas moduladas em largura de pulso (PWM) a 10kHz e 10 *bits* de resolução;
- 1 porta de entrada digital de 8 *bits*;
- 1 porta de saída digital de 8 *bits*;
- Acesso rápido pelo computador.

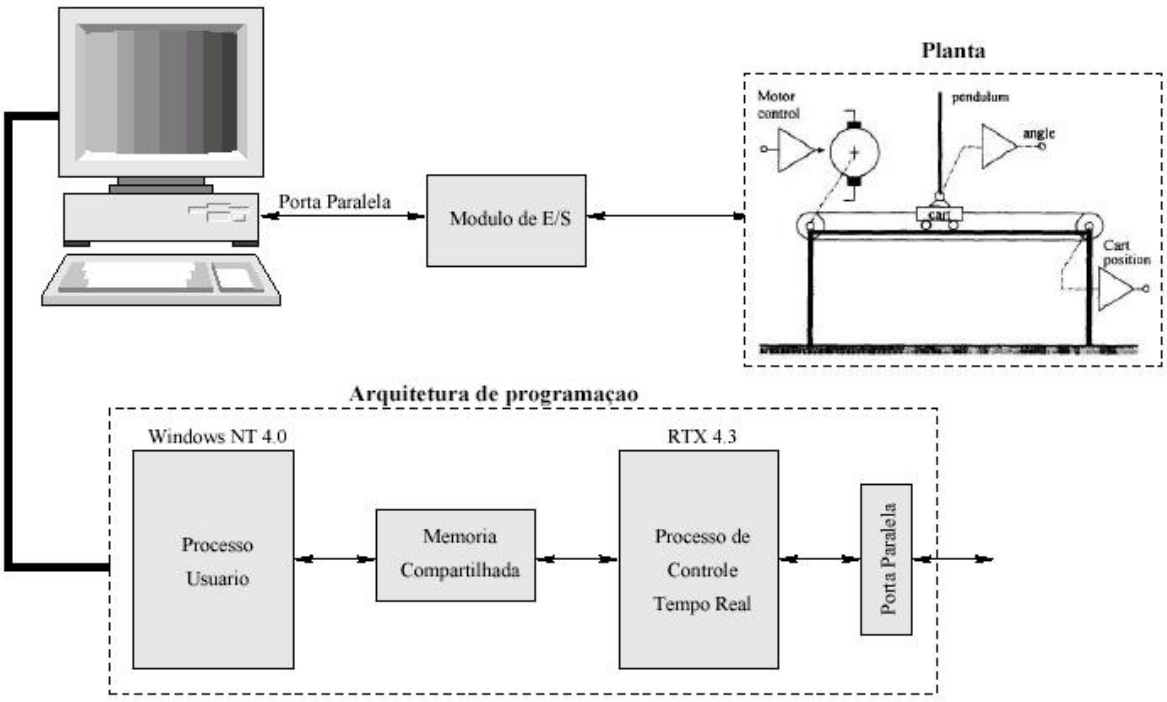

Figura 1 - Diagrama simplificado da estação de trabalho.

O módulo experimental presta-se à simulação de um processo industrial em escala reduzida. Para o controle de tal processo serão utilizadas técnicas de identificação em malha fechada para obtenção dos parâmetros do modelo. O projeto da estação de trabalho, como foi apresentado, permite a conexão de diferentes plantas por ser modular. No entanto, cada planta deve possuir seus próprios atuadores, sensores e alimentação, sendo disponibilizados apenas os sinais necessários para seu controle.

Em médio prazo, deverão ser desenvolvidas pesquisas em controle não-linear preditivo usando análise por intervalo que considera a presença de restrições na síntese de controladores.

A implementação deste trabalho buscou amenizar as imperfeições físicas encontradas em todo processo real. Foram utilizados rolamentos, visando a diminuição do atrito mecânico. Foram colocados sensores de presença de fim de curso para delimitar o espaço de atuação do carrinho. Foram introduzidos capacitores, como filtro de ruído, em todos os circuitos implementados. Os esforços foram válidos para diminuição das imperfeições, mesmo assim, elas continuam presentes.

3. Revisão bibliográfica

A revisão bibliográfica se ateve apenas à literatura técnica pois o trabalho realizado nesta fase não exigiu ir além, mesmo assim serão apresentadas as bibliografias referentes à fonte de inspiração do projeto e a bibliografia referente à plataforma e extensão utilizada.

A inspiração para o projeto do pêndulo invertido foi o artigo: [1]"*Laboratory Experiments for Control Theory Courses: A Survey*", que apresentava entre outros projetos o controle do pêndulo invertido e do ângulo de ataque e orientação de um helicóptero. Este artigo fala sobre a importância que a experimentação desempenha na formação acadêmica e da distância entre a prática e a teoria. Apresenta uma introdução à programação em tempo real com MATLAB e a descrição de vários projetos.

Para utilização da plataforma em tempo real foram consultados os manuais de introdução ao sistema Linux (Curso Linux Básico), o manual especifico da versão utilizada [2](Slackware Linux Essentials), e os manuais de instalação e utilização do RTAI [3](RTAI Programming Guide 1.0). O manual do Slackware apresenta uma introdução ao sistema Linux, abordando seu sistema de diretórios, comandos básicos e alguns passos para instalação de seus componentes. O manual do RTAI apresenta como programar, mostrando, ainda, uma breve explicação sobre suas funções e seus argumentos.

4. Metodologia

4.1. Projeto mecânico

O projeto mecânico constitui a implementação física do modelo em escala reduzida do processo industrial, que neste caso é o pêndulo invertido. Como o projeto mecânico foi detalhadamente explicado no relatório parcial do projeto de iniciação cientifica, aqui será dada apenas uma visão geral para que o leitor se familiarize com o processo de montagem. A Figura 2 apresenta uma visão geral do pêndulo.

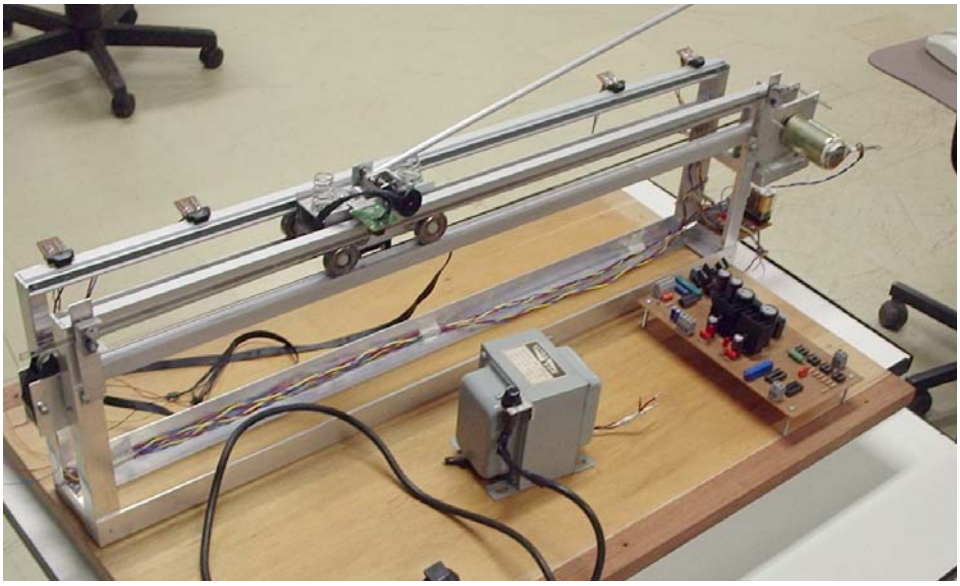

Figura 2 - Visão geral do pêndulo invertido.

Todos os materiais utilizados na montagem da parte física do pêndulo foram aproveitados de equipamentos encontrados na área de materiais reutilizáveis do SG11, provindos de sucatas e equipamentos defeituosos.

Os trilhos são constituídos de metais de divisórias de ambientes sobrepostas de maneira a impedir que o carrinho, que sustenta a haste do pêndulo, saia do trilho. Foram posicionados sensores de área útil e de fim de curso, os dói primeiro para que o controlador tome conhecimento que o carrinho alcançou o limite da área controlável e possa tomar uma atitude na tentativa de retornar o carrinho para a área útil sem perder o controle do equilíbrio da haste, Os outros dois para proteção das partes mecânicas do projeto, sendo que estes sensores acionam o circuito de segurança que impede o controle do carrinho pelo controlador e freia o motor atuador, evitando que o carrinho se choque nas extremidades do trilho. Foi colocada uma fita gradua da ao longo do trilho, para que fosse medida a posição do carrinho no trilho.

O carrinho apresenta um *codificador óptico incremental* retirado de uma impressora HP para a contagem dos pulsos da fita graduada e conseqüente medição da posição do carrinho. Um potenciômetro multi-voltas foi acoplado à base da haste no centro do carrinho para se medir a angulação da mesma. O carrinho ainda possui rolamentos para diminuir o

atrito com os trilhos e molas para absorver o impacto em caso de queda da haste. A montagem do carrinho pode ser vista na Figura 3.

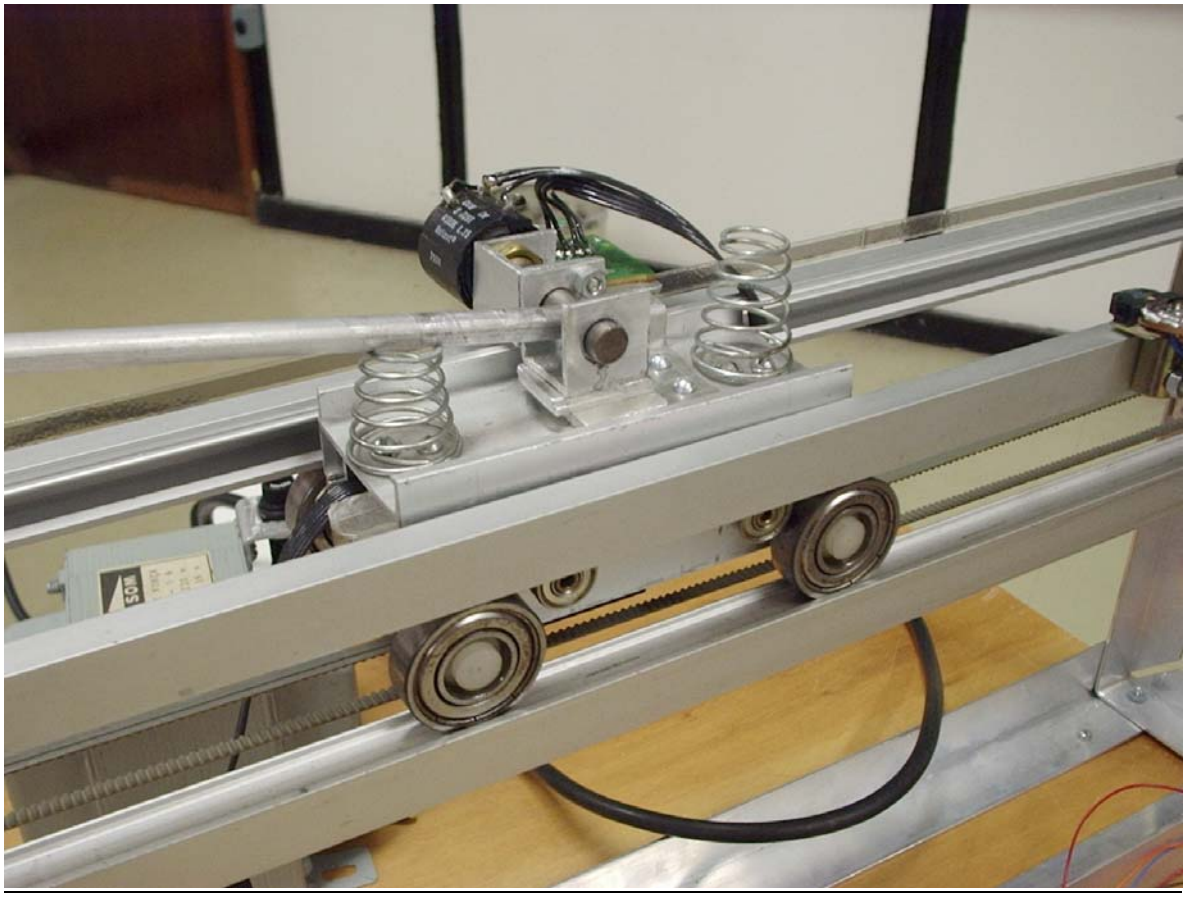

Figura 3 - Visão do carrinho.

O motor atuador encontra-se posicionado na extremidade do trilho e apresenta uma caixa de redução para garantir uma relação apropriada de velocidade e torque, que neste caso é de 5:1. A potência é transmitida através de uma correia tencionada por mola na extremidade oposta do motor. Toda a estrutura foi fixada em um suporte de madeira.

4.2. Projeto elétrico

O primeiro circuito projetado foi feito com a preocupação de garantir a integridade dos componentes mecânicos, o circuito de proteção anula o controlador e coloca os terminais do motor atuador em contato para que o mesmo seja freado. A Figura 4 apresenta o circuito de proteção.

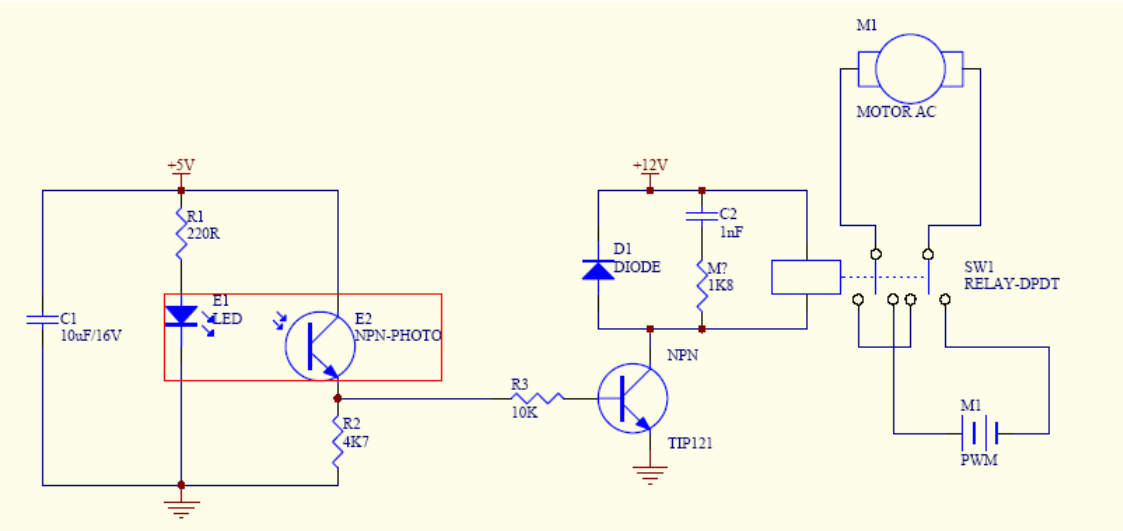

Figura 4 - Circuito de proteção.

4.2.1. Sensoriamento e condicionamento de sinais

Para a resolução do problema do controle do pêndulo invertido duas variáveis devem ser medidas, a posição do carrinho no trilho e a angulação da haste. Para a medição da posição do carrinho foi utilizado um *codificador óptico incremental* e para a angulação um potenciômetro, os sinais provindos dos sensores devem ser tratados antes de serem disponibilizados para a placa de interface.

O sinal vindo do potenciômetro passa por um comparador de tensão para que possa ser ajustado o ângulo zero no ponto desejado, o ajuste deste ângulo é feito através de um *trimpot*, o sinal também é amplificado. A Figura 5 apresenta o circuito de condicionamento do potenciômetro de angulação.

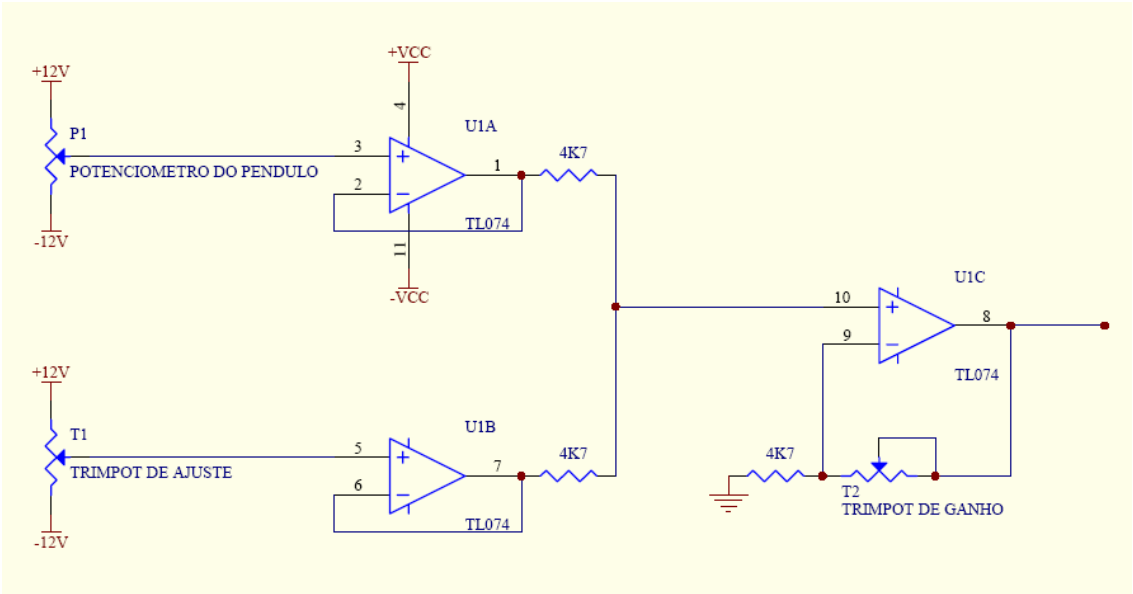

Figura 5 - Circuito de condicionamento do sinal de angulação.

O trem de pulsos que determina a contagem da graduação da fita passa por um flip-flop tipo D para determinação da direção do movimento do carrinho e o sinal dos dois canais (canal A e B defasados de 90º) passam por uma porta XOR para dobrar a freqüência dos pulsos e aumentar a precisão da medição. A Figura 6 apresenta o circuito de determinação do sentido do movimento e a Figura 7 apresenta os canais A e B da saída do sensor.

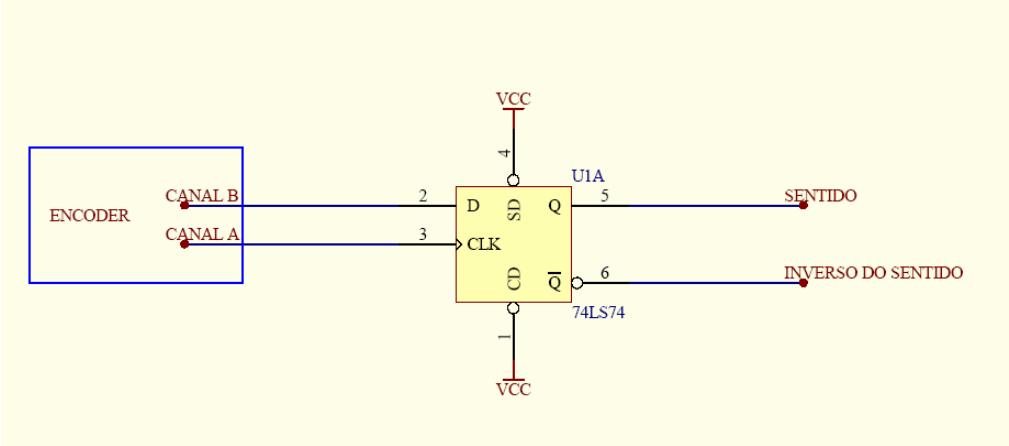

Figura 6 - Circuito de determinação de sentido do movimento.

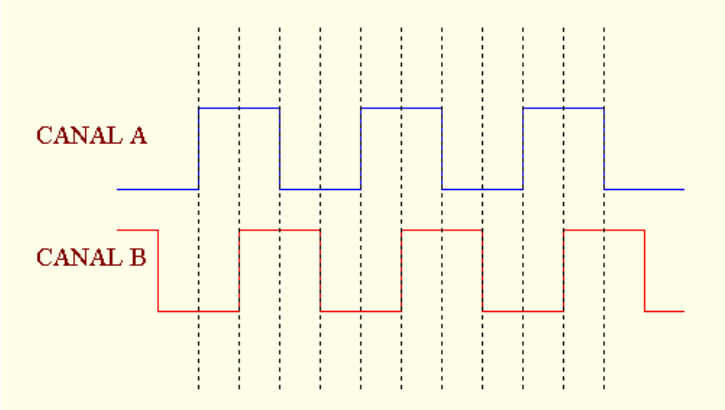

Figura 7 - Canais de saída do sensor de posição.

4.2.2. Placa de interface

A placa de interface apresenta um barramento de 8 bits para compatibilidade com o PC, entre suas diversas funcionalidades encontra-se a capacidade de conversão de sinais analógicos para digitais através de um conversor A/D, a geração de pulsos PWM através de um contador (que se presta à substituição do conversor D/A) e a possibilidade de ler e escrever dados digitais através das entradas/saídas digitais. A Figura 8 apresenta o circuito da placa de interface.

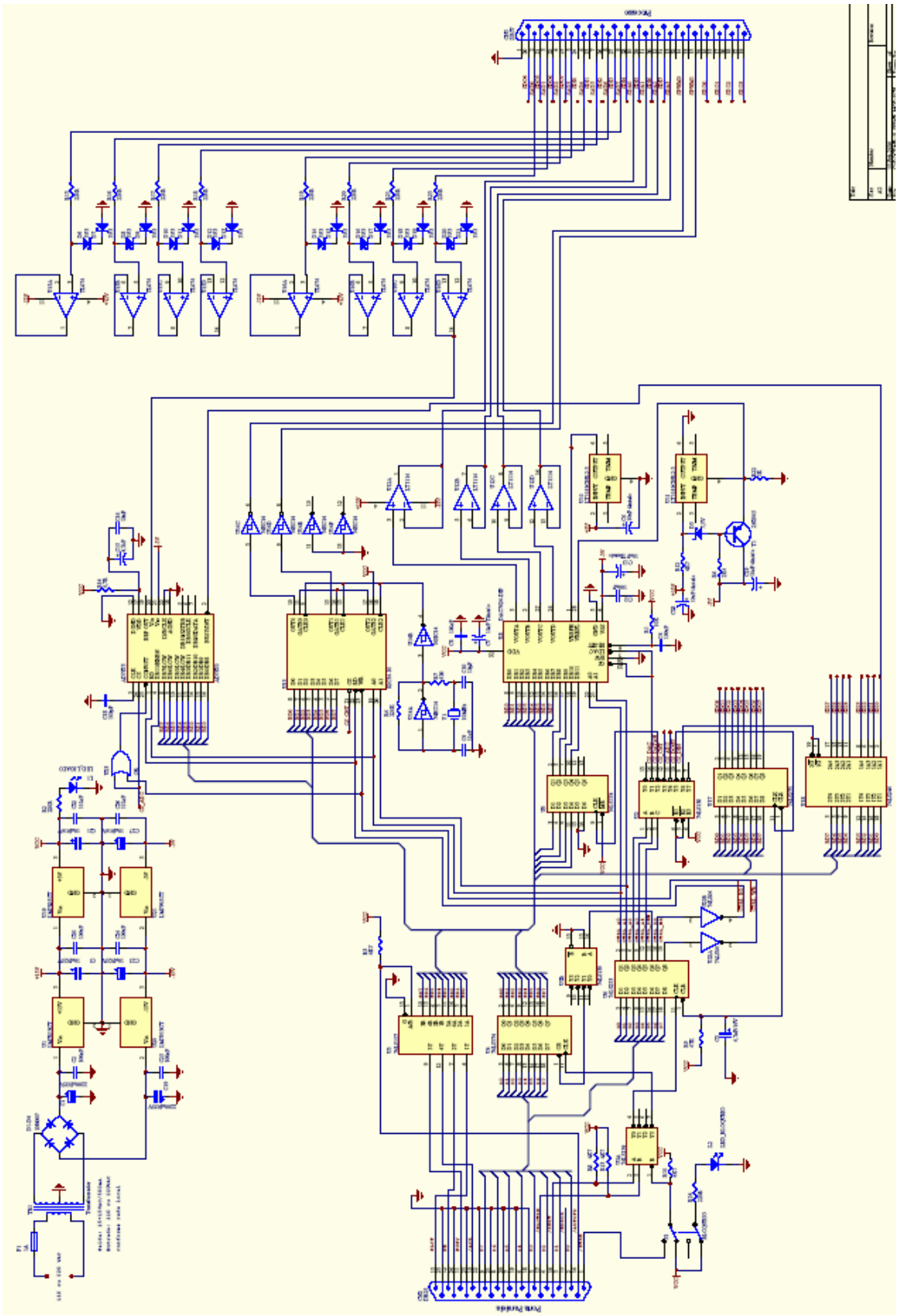

Figura 8 - Circuito da placa de interface.

Para um maior detalhamento dos circuitos e da montagem poderá ser consultado o relatório parcial de PIBIC deste projeto.

4.3. Plataforma de programação

A plataforma escolhida para implementação do controlador foi o Slackware Linux com extensão em tempo real RTAI, mas como será visto nos resultados obtidos, a escolha de uma plataforma de código aberto apresenta algumas desvantagens como a falta de documentação.

4.3.1. Estrutura do programa

A estrutura do programa se divide em duas partes, a primeira parte é a parte em tempo real que cuida do controle e do envio de dados ao usuário, a segunda parte em tempo normal que apresenta os dados obtidos do processo em tempo real e possibilita que o usuário interaja com o processo.

4.3.1.1 Módulo em tempo real

O modulo em tempo real tem a finalidade de controlar o processo, então este modulo deve, necessariamente, apresentar maior prioridade em relação aos outro módulos, tal prioridade é justificada pelo fato do controle do processo ser o objetivo principal das aplicações desta aplicação, em quanto que a apresentação dos dados do processo ao usuário é secundária e necessária apenas por questões de estética, pois os dados podem ser gravados e consultados posteriormente. A Figura 9 apresenta o módulo em tempo real.

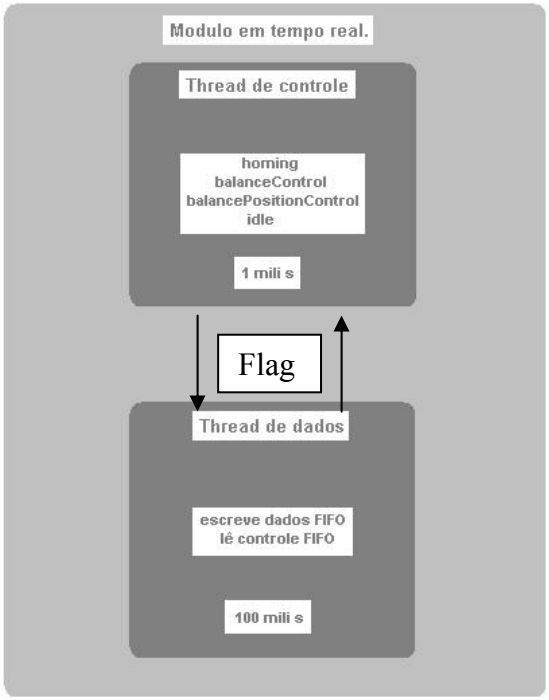

Figura 9 - Módulo tempo real.

Como foi escolhido apresentar os dados ao usuário em uma interface gráfica amigável o módulo em tempo real apresenta duas *threads*. A primeira presta-se exclusivamente ao controle do processo, esta *thread* aplica o tipo de controle desejado de acordo com uma *flag* presente no programa e que pode ser alterada pelo usuário durante o processo. A *thread* de controle apresenta o modo *homing*, que ignora a angulação da haste e promove o correto posicionamento do carrinho no início da área útil (que é o ponto de referencia do posicionamento do carrinho), o modo *balanceControl* que promove o controle da angulação da haste sem se preocupar com a posição do carrinho no trilho, modo *balancePositionControl* que promove o controle da angulação da haste e da posição do carrinho simultaneamente e o modo *idle* que simplesmente não implementa nenhuma ação. O controle aplicado pode ser modificado pelo usuário no decorrer da aplicação para todos os modos de controle. O controlador é executado a cada milisegundo e promove a medição das variáveis do sistema para o calculo e aplicação do controle.

O usuário pode selecionar dois tipos de controladores, PID e controle adaptativo. O controle PID apresenta suas constantes fixas, previamente testadas para se obter uma boa resposta.

A *thread* de dados promove a troca de dados entre o módulo usuário e o módulo tempo real, como esta troca de dados é menos prioritária que o controle esta tarefa é executada a cada 100 milisegundos, o que diminui a competição por recursos do sistema. O papel desempenhado por esta *thread* é o de obter todas as variáveis e estados disponíveis no sistema, como tempo, ângulo da haste, posição do carrinho, estado dos sensores do trilho, sentido do movimento, modo de controle e algoritmo do controlador, para que a interface de usuário possa apresentar o estado do sistema amigavelmente em uma interface gráfica e ainda salvar em arquivo os dados para que possam ser analisados posteriormente com o *Sci Lab* ou programa compatível. Para a troca de informações entre os processos em tempo real e usuário são usadas duas FIFOs (first in first out), que são fluxos de dados com restrição de acesso do tipo o primeiro dado a entrar é o primeiro dado a sair. A primeira FIFO presta-se à troca de dados do processo sendo que o processo em tempo real tem permissão apenas para escrita na FIFO e o processo usuário tem permissão apenas para leitura da FIFO (FIFO de dados), a segunda FIFO presta-se ao encaminhamento da intenção do usuário em modificar o controle ou o controlador (FIFO de controle), nesse caso o processo usuário tem permissão de escrita apenas em quanto o processo em tempo real tem permissão de leitura apenas. Quando qualquer dado é escrito na FIFO de controle o processo executa um procedimento que recupera essa informação da FIFO e seta os *flags* necessários para a modificação do controle.

4.3.1.2 Módulo usuário

O módulo usuário presta-se a apresentar uma interface de usuário amigável para apresentação do estado do sistema, gravação dos dados em arquivo e modificação dos algoritmos e modos de controle. A Figura 10 apresenta o módulo usuário.

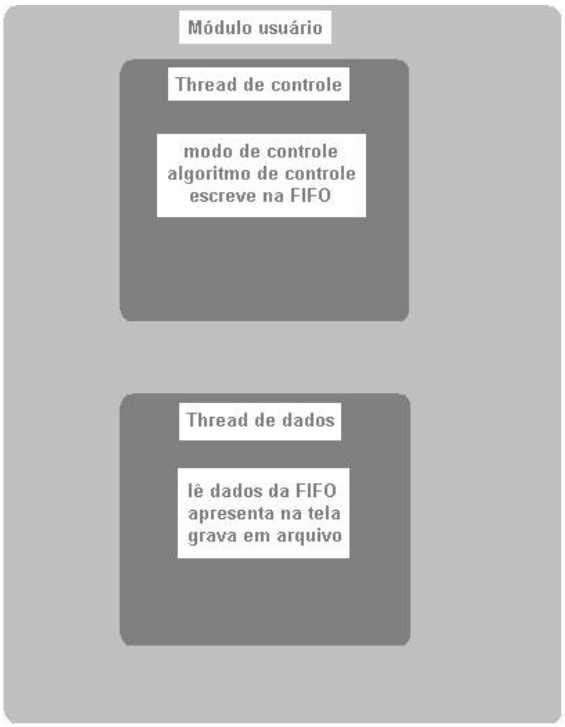

Figura 10 - Módulo usuário.

Este módulo também apresenta duas threads que cuidam de aspectos diferentes da comunicação entre os processos. A primeira thread monitora a intenção do usuário em modificar o modo de controle e/ou o algoritmo de controle, inserindo os dados inerentes à modificação na FIFO de controle. A segunda thread presta-se à apresentação dos dados em forma gráfica e a gravação do arquivo de dados em disco, para tanto a thread verifica a existência de dados na FIFO de dados e promove a sua extração, selecionando os dados que serão apresentados na tela e gravando toda a informação em arquivo. A apresentação dos dados na tela se dá em forma de um desenho do processo, onde se pode ver facilmente a posição do carrinho e o ângulo da haste, bem como o estado dos sensores do trilho.

A Figura 11 apresenta o programa completo com suas trocas de dados.

Figura 11 - Programa completo.

5. Resultados Obtidos no período

Os módulos do projeto foram implementados progressivamente, sendo que a cada módulo escrito se procedia a uma bateria de testes para garantir o correto funcionamento do módulo. Antes de fazer os módulos foram feitos as bibliotecas de acesso às funcionalidades da placa de interface, as funções de escrita, de leitura, de PWM e de conversão. Foram feitos testes suficientes para atestar a funcionalidade da biblioteca.

5.1. Controle e aquisição de dados

Com a biblioteca pronta procedeu-se á escrita do módulo em tempo real, primeiramente a parte de controle, foi implementado um controlador PID que no modo *balanceControl*, que preocupa-se apenas com a inclinação da haste do pêndulo. Os resultados obtidos com esta implementação mostram que a haste é mantida na posição vertical com uma pequena oscilação em torno do ponto central, esse resultado foi bem próximo do obtido com um controlador analógico implementando um PID. Neste teste ficou claro a eficiência da plataforma RTAI em tornar o processo prioritário em tempo real, durante a experimentação foram abertos alguns programas no Linux para que outros processos solicitassem parte do tempo do processador e verificou-se que o processo em tempo real é prioritário mesmo em relação aos processos do sistema operacional, inclusive mais prioritário que o mouse, e ainda assim o escalonador consegue encontrar tempo para rodar os processos menos prioritários evitando que os processos menos prioritários parem de rodar por "*starvation*". Não foi realizada nenhuma filmagem do processo com controle em tempo real, mas o controle analógico pode ser visto no sitio abaixo:
http://www.ene.unb.br/~gaborges/academico/orient/ic/etctr/pendulo_invertido_1.mpg.

Posteriormente foi implementado, ainda em controle PID, o modo *balancePositionControl*, que alem da angulação da haste do pêndulo leva em consideração a posição do carrinho nos trilhos. Nesta implementação, devido às imperfeições do projeto, o controlador PID se mostrou ineficiente no controle das duas variáveis. O sinal de controle total era composto pelo sinal de controle da angulação da haste do pêndulo e pelo sinal de controle da posição do carrinho no trilho, sendo que a orientação vertical da haste era controlada por um tempo com o carrinho divergindo do centro do trilho mas quando a posição do carrinho atingia uma posição próxima do final da área útil do trilho o acumulo de erro de posicionamento se tornava apreciável e o sinal de controle se tornava predominantemente o sinal de controle da posição fazendo com que o carrinho se deslocasse para o centro do trilho e perdesse o controle da angulação da haste do pêndulo. Também foram implementados os modos *homing* e *idle*, o primeiro modo utilizava um controle de posição PID para posicionar o carrinho próximo a uma das extremidades da área útil, utilizando um dos sensores de posição do trilho, o carrinho se movimentava com velocidade constante para o lado do atuador até que o último sensor de posição de área 'til fosse acionado, então este ponto era considerado o zero de posição e o controle de posição entrava em ação, posicionando o carrinho corretamente. O modo *idle* não implementa ação alguma.

A implementação da aquisição de dados chamava o modo *homing* e depois colocava o carrinho para se deslocar com velocidade constante para o outro lado até atingir o final de curso, sendo que as informações do processo eram transferidas para o módulo usuário para gravação em arquivo. A partir desta implementação foi possível levantar a curva característica do sistema a uma entrada constante. Foram testadas as respostas do sistema a várias velocidades constantes começando no limiar da zona morta do motor com tal de 10% a 100% com incrementos de 10%, o arquivo resultante foi manipulado em *Matlab* para se obter a curva características do sistema. O Gráfico 1 apresenta a curva característica do sistema no domínio do tempo em relação à posição, à velocidade e à aceleração com tal de 70%. Os dados coletados do processo serão usados em técnicas de determinação de parâmetros. Como os módulos de controle e aquisição de dados foram testados separadamente, cada módulo foi implementado com apenas uma *thread* em tempo real, que funcionou como esperado.

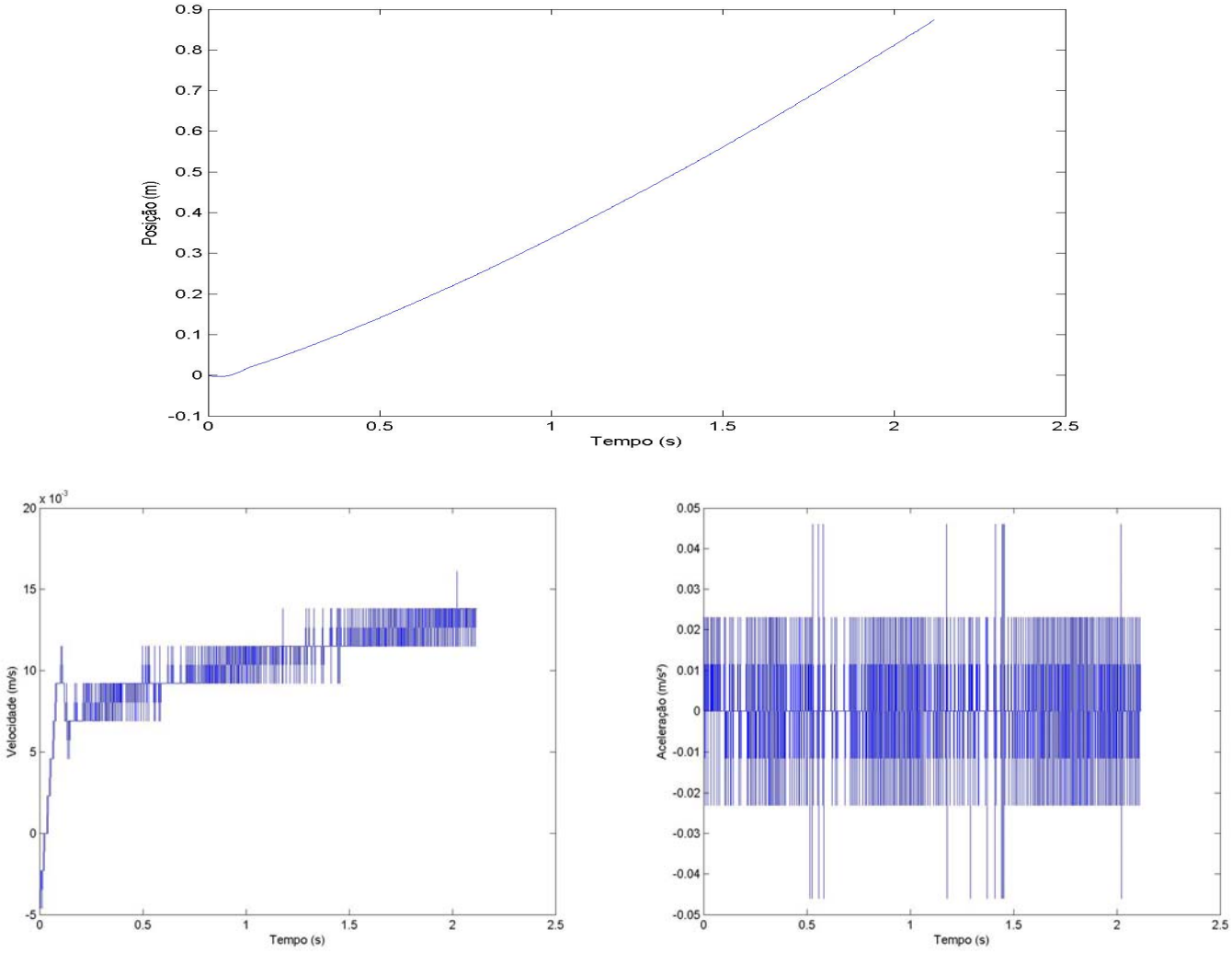

Gráfico 1 - Posição, velocidade e aceleração com tal de 70%.

5.2. Gravação de dados e envio de intenção do usuário

O módulo usuário apresenta uma *thread* que testa se a FIFO de dados contem dados e procede à retirada dos dados até que a FIFO esteja vazia, quando a FIFO está vazia a *thread* espera 100 milisegundos e volta a procurar dados na FIFO, o procedimento é terminado quando um *flag* de término é modificado através de uma função. A outra *thread* do módulo usuário espera que o usuário digite a opção desejada através do teclado e insere na FIFO de controle os dados correspondentes ao modo de controle desejado, caso o modo digitado seja válido.

5.3. Integração dos módulos

Depois de implementados e testados todos os módulos separadamente procedeu-se à integração dos módulos, para tanto deveriam ser criadas duas *threads* em tempo real

(*kernel*) e outras duas normais (usuário). O programa com o código contendo os módulos integrados foi escrito, inclusive com o devido bloqueio ao acesso ao hardware para se evitar *deadlock*, mas ao se executar este código o sistema se tornava instável e parava de responder, apontando uma referência perdida a um ponteiro do sistema, com suas devidas especificações (endereço de memória). Tal problema tem causa desconhecida visto que também ocorreu ao se rodar um código fornecido junto com o produto (RTAI) e que também continha mais de uma *thread* em tempo real rodando ao mesmo tempo. Foram feitas várias tentativas de contornar o erro, como reinstalação de todo o sistema operacional e da plataforma RTAI, reinstalação de versões mais recentes e mais antigas da plataforma RTAI, todas as tentativas acabaram com o mesmo resultado. Como a plataforma RTAI é de código aberto e não possui documentação completa nem suporte técnico as etapas finais do projeto foram comprometidas, pois tais etapas dependiam exclusivamente da programação para serem concluídas.

6. Conclusões

O objetivo de se construir uma plataforma de testes para experimentação de controle foi plenamente atingido, o modelo em escala reduzida do processo industrial apresenta-se em funcionamento, a placa de interface apresenta as funcionalidades necessárias para o controle do processo e também se encontra em pleno funcionamento, foi feita uma biblioteca de manipulação das funcionalidades da placa de interface, para que os alunos que vierem a utilizar a plataforma não precisem conhecer o funcionamento da placa para fazer a aquisição de dados, podendo se concentrar nas atividades de controle.

A implementação do projeto em uma plataforma aberta se mostrou um verdadeiro desafio, porque vários conhecimentos avançados de Linux foram necessário, como exemplo dos conhecimentos que tiveram que ser adquiridos está a re-compilação do *kernel* do Linux com aplicação dos *patchs* inerentes ao RTAI. A falta de suporte e documentação dificultou a solução do problema de se rodar mais de uma *thread* em tempo real simultâneas. A plataforma RTAI encontra-se em desenvolvimento e está em constante modificação, é possível que na implementação utilizada a utilização de *threads* tempo real simultâneas não funcione corretamente, ou mesmo que a re-compilação do *kernel* e aplicação dos *patchs* do RTAI tenha sido feita de maneira errônea.

A pretensão de se implementar um algoritmo de controle adaptativo para que o processo fosse completamente controlado (controle da angulação da haste e da posição do carrinho) foi impossibilitada pela dificuldade encontrada na programação tempo real, espera-se que com o tempo a plataforma RTAI seja melhorada e seja facilitado seu método de instalação.

7. Bibliografia

[1]. Peter Horáček. Laboratory experiments for control theory courses: A survey. *Anual Reviews in Control*, 24:151-162, 2000.

[2]. David Cantrell Logan and Johnson Chris Lumens. Slackware Linux Essentials. The Official Guide To Slackware Linux.

[3]. RTAI Programming Guide 1.0 Lineo, Inc. Lindon, Utha 84042, USA. September 2000. http://www.lineo.com/

[4]. Dual full-bridge driver – L298 – Datasheet.

[5]. Analog Devices – Conversor analógico/digital – AD7870_5_6_b – Datasheet – http://www.analog.com

[6]. Porta lógica XOR –74HS86 – Datasheet.

[7]. Amplificador Operacional –TL074 – Datasheet.

8. Avaliação pelo bolsista

8.1. Auto-avaliação

Acredito que a dedicação despendida ao projeto apresentou plena conformidade com a necessidade solicitada no contrato de PIBIC, visto que as matérias cursadas juntamente com a realização do projeto de pesquisa apresentam prioridade superior à do projeto. O interesse apresentado pelo bolsista corroborou para o bom andamento do projeto, apresentando os resultados previstos nas datas estabelecidas pelo cronograma de desenvolvimento do projeto, salvo o problema encontrado com a plataforma.

8.2. Avaliação da orientação

A presença do orientador foi imprescindível na realização do projeto, onde fez-se necessário conhecimentos científicos que iam além dos adquiridos no curso de graduação. Além da motivação oferecida nos momentos de dificuldade, mostrando que os desafios devem ser superados com paciência e persistência.

8.3. Avaliação das condições de trabalho

Apesar da escassez de recursos devido a poucos incentivos à pesquisa científica, outras formas de recursos se fizeram presentes, tais como materiais reutilizáveis aproveitados de equipamentos desativados e componentes fornecidos pelo grupo de apoio técnico do SG11, o que viabilizou a implementação do projeto mesmo que com restrições.

9. Atividades do bolsista

Participou da X Semana de Ciência e Tecnologia, realizada no Centro Comunitário da UnB, como expositor, apresentando o projeto desenvolvido durante o PIBIC de Agosto/2003 a Julho/2004.