Sistemas Embarcados

Desenvolvimento de uma plataforma de monitoramento de propulsores iônicos do tipo PHALL durante testes de longa duração.

Helbert de Oliveira Coelho Júnior
14/0142851
Universidade de Brasília - FGA
hoc.junior@gmail.com

***Abstract*— Este projeto terá como objetivo o desenvolvimento de uma plataforma integrada de monitoramento das condições de operação de um propulsor iônico do tipo PHALL. Tal plataforma, composta de uma Raspberry Pi como seu cérebro, será um servidor remoto que fornecerá ao usuário informações sobre o estado do propulsor durante testes de longa duração.**

Keywords*— *RaspberryPI; ADC; PHALL; Datalog; SPI; Aquisição de dados;

I. INTRODUÇÃO

O propósito desse projeto é o de criar um sistema que permita o monitoramento das condições de um propulsor PHALL durante um teste de longa duração.

Propulsores iônicos do tipo PHALL são propulsores desenvolvidos para operar por longos períodos de tempo, fornecendo um empuxo reduzido mas um grande impulso específico. Abaixo podemos ver o propulsor que está sendo desenvolvido no Laboratório Associado de Plasmas, na Universidade de Brasília.

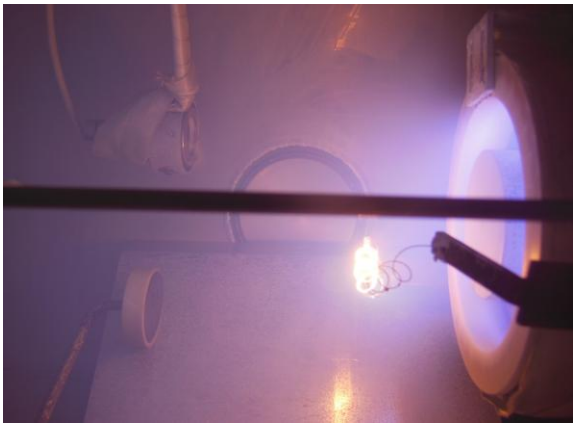

Figura 1. Propulsor iônico sendo desenvolvido na Universidade de Brasília.

Durante o teste de tal propulsor, queremos conhecer os valores de tensão e corrente aplicados ao anodo (Va, Ia) e ao cátodo (Vc, Ic). Além disso, para estimar o valor do empuxo gerado, utilizamos uma sonda de Langmuir. Tal sonda necessita de uma rampa de tensão para que possamos calcular os parâmetros desejados do plasma da pluma do propulsor.

Uma vez que essa curva foi obtida, temos que retirar alguns valores para se realizar a devida caracterização do plasma. A primeira análise da curva de Langmuir consiste em calcular a sua derivada e plotar o gráfico em escala logarítmica. A derivada serve para encontrarmos os pontos de inflexão da curva, enquanto que a escala logarítmica nos dá o Potencial de Plasma. As figuras abaixo mostram os pontos de inflexão bem como uma curva na escala logaritimica, calculados em janeiro desse ano.

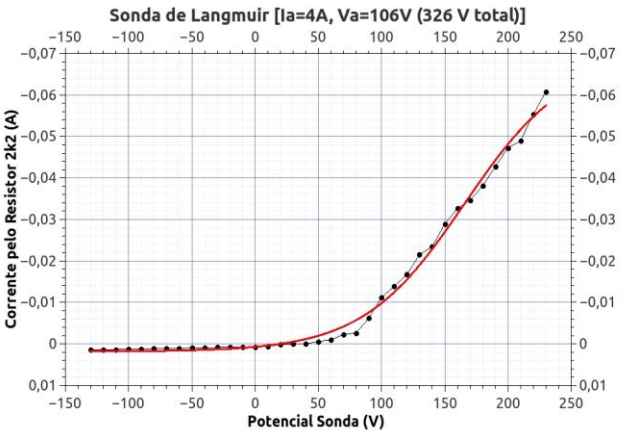

Figura 2. Curva de Langmuir colhida pelo aluno e utilizada nos testes do protótipo.

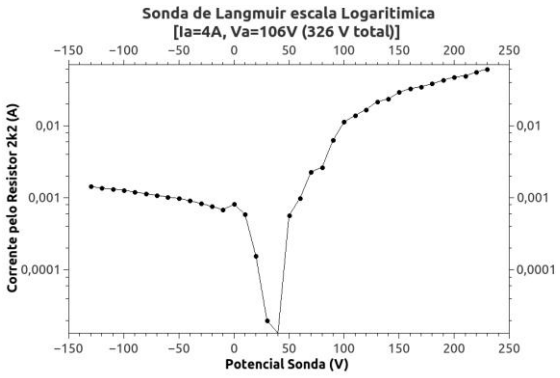

Figura 3. Gráfico com escala logarítmica da curva acima. O ponto em que a curva cruza o eixo X é chamado de Potencial de Plasma.

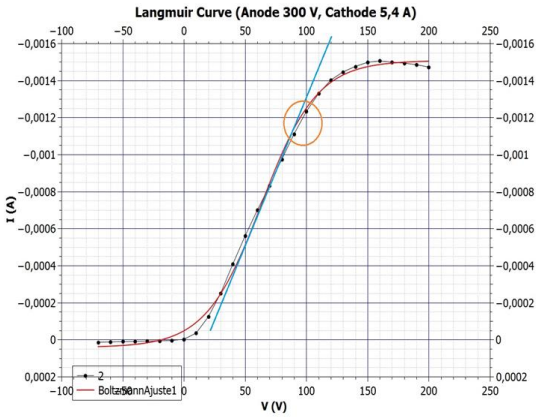

Figura 4.Ponto de inflexão.

Uma vez que os pontos de inflexão foram encontrados, caracterizados por uma mudança elevada no valor da derivada, as características do plasma podem ser calculadas. Utilizamos o valor de tensão e corrente do gráfico nesses pontos para se calcular a densidade e a temperatura do gás ionizado.

A temperatura de elétrons pode ser calculada com a seguinte fórmula, cuja região podemos ver no gráfico abaixo:

$$\ln I_e = \ln I_0 - \frac{e|U|}{kT_e}$$

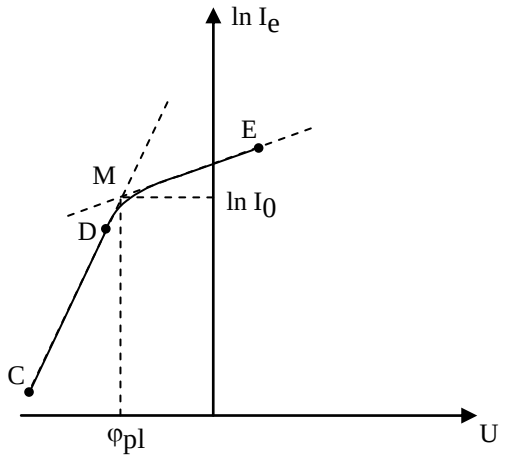

Figura 5. Região da inflexão. Cálculo da temperatura de elétrons.

Já a densidade de elétrons pode ser calculada através da fórmula abaixo:

$$N_e = \frac{4I_o}{eV_{th}S}$$

Além desses valores, precisamos da corrente e da tensão no anodo, massa do gás que passa pelo propulsor e algumas constantes para se obter o valor do empuxo de forma indireta.

Portanto, para se obter tais valores o sistema de monitoramento deve ser capaz de obter quatro valores de tensão, Tensão no Anodo, Tensão na fonte da sonda, Tensão no resistor de shunt para se obter a corrente na sonda e Tensão no resistor do anodo para se obter a corrente que passa pelo mesmo.

II. DESENVOLVIMENTO

Para conseguir ler todos esses valores de tensão, devemos utilizar algum conversor AD que se comunique com a Raspberry Pi e transmita esses dados. Antes de colocar o sinal no conversor AD temos que condicionar o mesmo para valores saudáveis no MSP430. Dessa forma temos um circuito de condicionamento do sinal antes de enviar o mesmo para o microcontrolador. O msp430 irá se comunicar através do protocolo I2C com a Raspberry Pi.

Uma plataforma que contenha um sistema operacional nos permite realizar ações mais complexas de forma simples, pois podemos utilizar as diversas abstrações do sistema operacional, como protocolos de rede, interface gráfica, capacidade de processamento e multi-tasking (vários processos e threads).

O diagrama de blocos abaixo ilustra o sistema básico necessário para a aquisição dos dados.

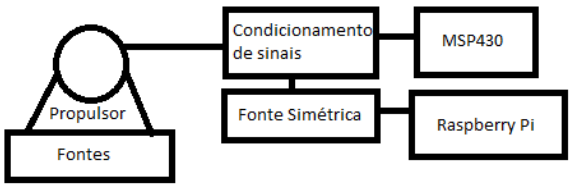

Figura 6. Diagrama de Blocos simplificado.

Uma vez que a aquisição do sinal tenha sido feita de forma correta, os valores serão gravados em um arquivo e o processo pai deverá esperar por 15 minutos antes de fazer uma nova medida.

Durante o processo de medida, o processo pai deverá comunicar-se com o msp430 a cada 300 ms, de forma a gravar os quatro valores de tensão.

A primeira dificuldade que obtive foi nesse ponto, de utilizar 4 ADCs no msp430.

Depois de obter uma curva, o processo pai criará um processo filho e este será responsável por realizar os cálculos necessários. Ao término da execução, o processo filho se encerrará, restando o processo pai.

O motivo de se criar um novo processo com a instrução fork() é para permitir que o processo pai continue a coletar dados enquanto os cálculos estão sendo efetuados.

Como objetivo adicional, o processo filho deveria enviar os resultados obtidos para o operador. Pensei em utilizar GSM, Ethernet, Servidor Web, entre outras possibilidades. Porém, essa comunicação externa não foi implementada com sucesso.

A cada curva nova, o processo pai deverá criar um arquivo com a data e a hora, de modo que o mesmo arquivo não seja acessado repetidas vezes e os dados não sejam sobrescritos.

Na implementação do algoritmo para se retirar os valores da curva, diversos erros foram encontrados e um resultado satisfatório não foi obtido.

III. DESCRIÇÃO DO HARDWARE

O sistema possui alguns circuitos separados trabalhando em conjunto para se obter os dados do propulsor. A lista a seguir enumera os principais itens do sistema, bem como alguns esquemáticos e fotos das placas projetadas e montadas pelo aluno.

-Raspberry Pi. Essa plataforma possui o papel de cérebro do circuito, enviando instruções para o msp430 (ou outros conversores AD), para as fontes (ou para o operador do teste), realizando os cálculos necessários e mantendo o datalogger. Não se viu a necessidade de aplicar threads à execução do programa.

-MSP430. Esse microcontrolador é responsável por realizar a conversão AD dos diversos sinais necessários.

-Placa de Condicionamento de dados. Esse circuito consiste em aplificadores operacionais e de instrumentação para proteger o circuito lógico sensível à altas tensões e correntes que temos na operação do propulsor. Atualmente utiliza 5 ampops por canal, bem como um divisor de tensão e um filtro passivo.

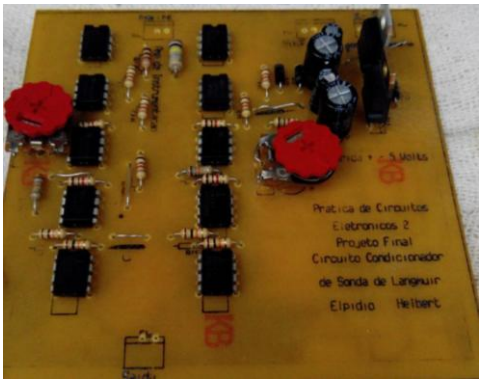

Figura 7. Placa de aquisição de dados.

-Fonte simétrica. A alimentação do circuito ativo é feita por essa fonte.

-Sonda de Langmuir. Essa sonda fica dentro da câmara de vácuo e funciona como um sensor para o plasma do propulsor.

-Fontes de alimentação do sistema PHALL. Por ser um propulsor elétrico, diversas fontes são necessárias para se manter a descarga elétrica.

IV. DESCRIÇÃO DE SOFTWARE

Aqui o projeto desandou. No início e meio do desenvolvimento, os diversos sistemas estavam sendo construídos e testados sozinhos. À medida que o mesmo foi sendo integrado no mesmo programa, diversos problemas foram sendo encontrados e, devido à falta de organização do aluno, não tive tempo hábil para consertar os erros. Dessa forma, não obtive nenhum resultado útil e o projeto se encontra na teoria.

Porém, o sistema proposto consistia em um processo pai, que criaria processos filhos para realizar as ações. O fator tempo de execução não é algo que deve ser mantido à risca, não sendo necessário utilizar módulos de kernel ou sistemas em tempo real.

Em suma, o processo pai iria criar um arquivo com a data e hora atuais. Esse mesmo processo deveria enviar endereços à msp430, esperar 100 us, e ler os valores do ADC de 10 bits da msp430. Como precisamos de 4 valores de tensão diferentes, o msp430 deveria realizar essa medida 4 vezes. Após 300 ms, a Raspberry Pi deveria recomeçar o ciclo de leituras até que fossem lidos 1000 pontos de cada uma das curvas. O código em anexo contém esse algoritmo para uma curva apenas, sendo possível escalar para 4 sinais através de chamadas de funções criadas pelo aluno.

Após o a leitura dos 4 sinais, o processo principal iria criar um processo filho utilizando da função fork(). Como o processo filho herda do pai as variáveis já criadas e etc, o filho teria acesso à esses quatro vetores. Ele deveria então criar um arquivo com a data e hora e gravar os quatro vetores nesse arquivo. Ao término, o processo filho realizaria as contas necessárias através de chamadas de funções e gravaria esses resultados no final do arquivo criado, encerrando a sua execução.

Enquanto isso, o processo pai esperaria 15 minutos antes de iniciar um novo ciclo de leitura de dados repetindo todo o processo. Dessa forma, mesmo que um processo filho demore um pouco mais a execução de suas instruções, o processo pai pode continuar a ler os valores do msp430.

Dependendo do resultado obtido com o tratamento matemático da curva, o processo filho deveria avisar ao usuário que detectou um erro.

V. RESULTADOS

Os resultados obtidos ficaram bem aquém do proposto para o projeto final. Diversos fatores influenciaram, criando diversos erros nos testes realizados anteriores a entrega deste relatório.

Primeiro, o Laboratório de Física dos Plasmas possui um propulsor iônico, porém o mesmo estava esperando peças para poder funcionar. Dessa forma não seria possível testar a aquisição dos sinais, função esta que representava a motivação do projeto.

O professor propôs simular as curvas obtidas utilizando um Arduino. Pesquisei um pouco sobre isso, utilizando uma onda PWM e um filtro passa-baixas para criar uma onda similiar àquela que teria na saída do sistema de condicionamento de sinais.

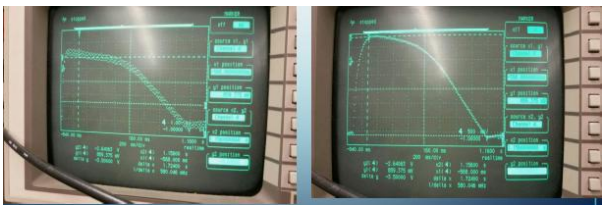

Figura 8. Sinal na saída da placa de aquisição de dados. Essa curva deveria ser recriada com o arduino para se realizar o teste sem o propulsor.

Não consegui recriar de forma satisfatória a onda utilizando o arduino.

Segundo, o sistema dependia de um conversor AD para ler os sinais. Adquiri da China alguns módulos ADC parecidos com os vendidos pela ADAFRUIT. Os mesmo ainda não chegaram. Decidi utilizar o msp430 para ler os sinais. Tive dificuldades para receber o sinal de 10 bits através de dois pedaços de 8 bits, bem como utilizar mais de uma porta ADC do msp430. Com o intuito de pelo menos provar o conceito, o programa em anexo foi criado com base em 1 sinal apenas.

Terceiro, não consegui construir de forma iterativa o algoritmo que utilizo para calcular os parâmetros necessários do propulsor. Os valores obtidos na lógica utilizada na plataforma Labview não foram condizentes com os valores que encontrei realizando o cálculo de forma manual. Estava fazendo na plataforma Labview com o intuito de acelerar o desenvolvimento do algoritmo, sendo passível a sua conversão para a linguagem “C” após a comprovação de sua eficácia.

Quarto, e possivelmente o mais importante, não me organizei direito e deixei para terminar o projeto na semana da entrega. Com isso, os diversos erros de lógica, compilação e execução no código principal me sobrejulgaram. Não consegui terminar o projeto. Envio o código incompleto junto com este relatório.

VI. CONCLUSÃO

A proposta desse projeto final é uma excelente ferramenta para que o aluno consiga aplicar os conteúdos teóricos vistos em sala, desde que o mesmo se organize e não deixe para última hora.

A automatização desses testes básicos em propulsores de plasma permitem um maior controle bem como mais liberdade para o operador durante os testes de vida, sendo possível de verificar algumas prováveis anomalias.

O algoritmo proposto aqui era um esboço inicial, sendo necessário um tempo de aquisição de dados menor, quase que durante todo o período de operação do mesmo. Para tal, uma boa quantidade de memória e de soluções anti travamento ou interrupção da aquisição dos dados se fazem necessárias.

Acredito que, corrigindo os erros encontrados o sistema proposto pode ser sim utilizado para o monitoramento nesses tipos de testes.

Infelizmente o aluno não concluiu um protótipo funcional.