MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

DESCRIÇÃO FUNCIONAL DO SUBSISTEMA DE CONTROLE E RASTREIO DA ESTAÇÃO MULTIMISSÃO DE NATAL (EMMN)

Kurios Iuri Pinheiro de Melo Queiroz (UFRN, Bolsista PIBIC/CNPq)
E-mail: kurios@crn.inpe.br

Manoel Jozeane Mafra de Carvalho (INPE, Orientador)
E-mail: manoel@crn.inpe.br

COLABORADORES

Dr. Francisco das Chagas Mota (DCA/UFRN)
Eng. José Marcelo lima Duarte (INPE/UFRN)

Resumo

Este trabalho tem como objetivo a descrição funcional do Subsistema de Controle e Rastreio (SCR) da estação multimissão de Natal (EMMN). Sua concepção vai desde o acionamento e proteção do sistema (*hardware*), até as técnicas de controle empregadas no rastreio de satélites (*software*). O sistema tem como base a estação SACI (atualmente desativada), cuja estrutura física (cabos, antena, motores, inversores, etc.) foi totalmente aproveitada. Toda a parte de software foi desenvolvida utilizando-se a plataforma GNU/Linux e os pacotes do projeto *Comedi*, responsáveis pelos *drives* da placa conversora analógica-digital (AD/DA), e pelas bibliotecas de programação na linguagem C.

Sumário

Lista de Figuras

Lista de Tabelas

1. Introdução

Este relatório visa à descrição funcional do subsistema de controle e rastreio (SCR) da estação multimissão de Natal (EMMN). O documento em questão é dividido basicamente em duas partes: uma que aborda detalhadamente todo o *hardware* da estação (seções de 2 a 6), e outra que descreve o *software* responsável pelo controle e rastreio da EMMN (seções de 7 a 13). Este último é ainda seccionado em dois blocos funcionais, o que trata das técnicas de controle empregadas, e o que realiza o ajuste das efemérides durante o rastreio. Tal ajuste é fundamental, pois o sistema de referência adotado para a estação, não corresponde ao sistema de coordenadas das efemérides fornecidas. A seção 12 explica o sistema de coordenadas adotado para antena, além de alguns problemas e suas respectivas soluções encontrados no decorrer do desenvolvimento do sistema.

Todos os componentes básicos do sistema (como cabos, motores, fontes, etc.) foram herdados da antiga estação de rastreio SACI (Satélite Avançado de Comunicações Interdisciplinares), atualmente desativada. Basicamente, o *hardware* é composto por quatro módulos: o módulo posicionador, o de potência (gaveta de potência – *puissance*), o de comando (gaveta de *pilotage)*, e o de controle (computador). Cada um destes itens será abordado detalhadamente nas próximas seções, apresentando seus componentes internos e suas principais funcionalidades. A figura 1 mostra a estrutura de *hardware* da estação, com as suas respectivas interações entre os módulos.

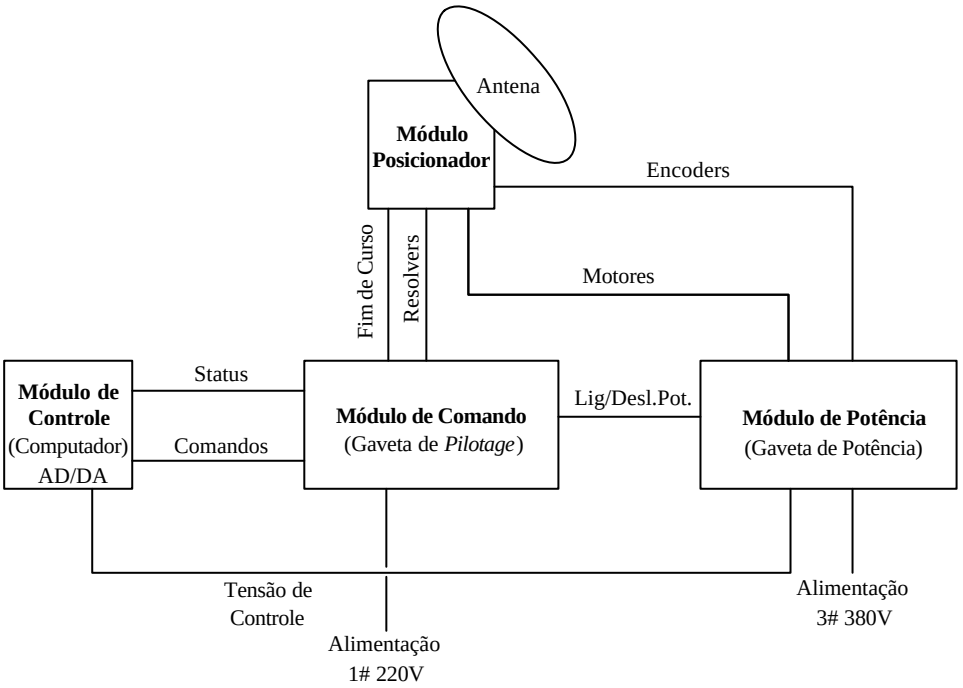

Figura 1: Estrutura de *Hardware* da estação EMMN.

O módulo posicionador é composto basicamente pelos motores que movimentam a antena, pelos redutores coaxiais, pelos sensores de fim de curso, e pelos sensores de posição (*resolvers)*. Os motores são equipados com *encoders*[1], que fornecem periodicamente informações para os inversores na gaveta de potência,

[1] Sensores de posição angular.

responsáveis pelo acionamento e controle de velocidade desses motores. O movimento da antena é monitorado *on-line* pelo módulo de controle, através dos *resolvers* acoplados em cada um dos eixos da antena (azimute e elevação). A comunicação entre os módulos de controle e comando é realizada através de uma placa conversora AD/DA, por onde são coletados dados sobre o *status* do sistema, bem como realizada a emissão de comandos.

A gaveta de *pilotage* engloba todos os circuitos de comando, além de todas as fontes de alimentação e *LEDs* para indicação visual de eventos, como por exemplo, botão de emergência acionado. Ela tem a função de informar ao módulo de controle (computador) a situação atual do sistema, e também de transformar os comandos recebidos em ações.

O *software* de rastreio que trabalha com esta configuração de *hardware* é o *CITO0v2* (versão 0.2), desenvolvido em C/C++, exclusivamente com ferramentas de *software* livre.

2. A Teoria de Controle

O problema de controle no sistema consiste no posicionamento adequado da antena, nos seus dois eixos, a cada intervalo de amostragem. No controle de posição, para uma referência do tipo degrau, não é necessária a inclusão de um controlador que melhore o regime permanente. Porém, como a referência é do tipo rampa, em virtude do rastreio, um controlador do tipo proporcional integrativo (PI) foi utilizado. A figura 2 apresenta o digrama de blocos do sistema.

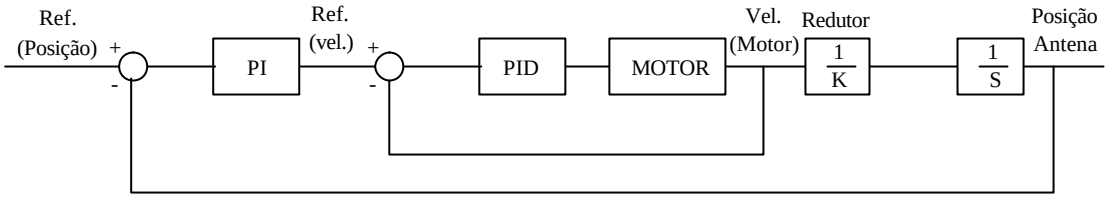

Figura 2: Diagrama de blocos do sistema.

Além do controle de posição envolvido, existe o controle de velocidade dos motores, que é realizado pelo módulo de potência, através de um controlador PID (Proporcional, Integrativo e Derivativo) presente nos inversores. Estes geram a freqüência e intensidade das correntes de alimentação para os motores, de modo que a velocidade desejada seja atingida. A realimentação da malha de velocidade é realizada por dois *encoders*, acoplados um em cada motor.

Um controlador digital do tipo PI, funcionando no módulo de controle, compara a posição atual da antena com a posição desejada. A partir desta diferença e dos demais cálculos envolvidos no PI, determina-se o sinal de controle para o inversor. Este sinal de controle é analógico, sezndo gerado pela placa conversora AD/DA, e corresponde a velocidade de rotação desejada para o motor. O período de amostragem deste controlador é de 1s, ou seja, a leitura da saída da planta (posição da antena), seguida dos cálculos de controle é feita em intervalos de 1s.

3. Módulo Posicionador

O posicionador da antena permite que a mesma se movimente nos dois eixos, azimute e elevação, de forma independente. Cada eixo é movido por um conjunto motor mais redutor coaxial. Os motores são do tipo autossíncronos (imãs permanentes), com *encoders* oriundos de fábrica. Dispositivos de fim de curso, mecânicos e elétricos, instalados nos dois eixos, limitam o deslocamento da antena parabólica, aos valores limites de segurança. O motor responsável pelo deslocamento no sentido de elevação possui um freio eletromagnético para garantir o seu travamento nas paradas. A figura 3 apresenta a antena parabólica da estação.

Figura 3: Antena parabólica da estação EEMN.

4. Módulo de Potência

Na gaveta de potência (*puissance*) estão instalados os dois inversores, modelo UMV 4301 da *Leroy Somer*, responsáveis pelo acionamento e controle de velocidade dos motores. Os inversores (ou *drives*) recebem um sinal de controle, proveniente do módulo de controle, entre -10V e +10V. Uma tensão de +10V equivale a velocidade nominal do motor (3000 rpm) num sentido, -10V no sentido contrário, e 0V a 0 rpm. O sistema de controle interno do inversor garante a convergência da velocidade real do motor com a velocidade de referência.

A alimentação dos dois inversores é controlada através de uma chave contactora, acionada pelo módulo de comando. O sinal para cortar a alimentação dos inversores, pode ser gerado devido ao acionamento de alguma proteção, como a de fim de curso, ou da chave que desliga o sistema. As figuras 4 e 5 apresentam respectivamente, o esquema elétrico do módulo de potência, e sua a estrutura física em gaveta industrial.

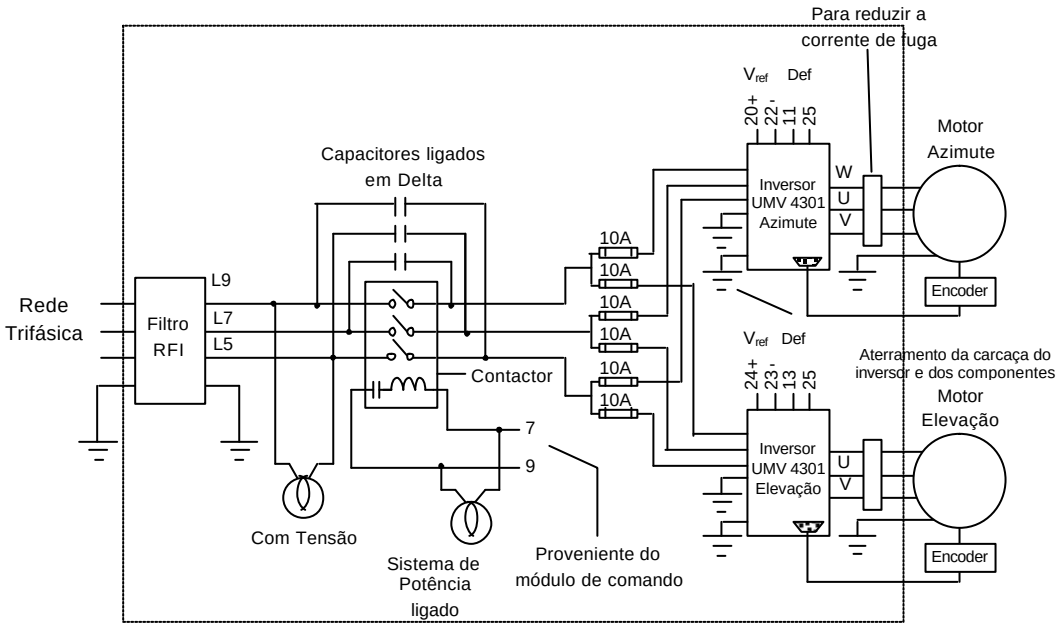

Figura 4: Módulo de potência.

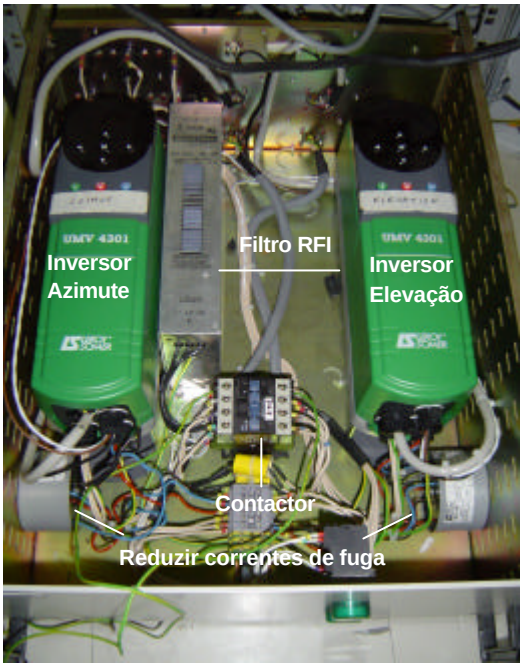

Figura 5: Foto da gaveta de potência (módulo de potência).

As tabelas 1 e 2 apresentam a pinagem dos conectores S25 e S26, utilizados no módulo de potência. Eles são identificados por uma letra e um número, da mesma forma como na estação SACI, uma vez que sua gaveta industrial pertencia ao antigo sistema.

Pino	Função	Fio
1	Bobina da chave contactora	7
2	Bobina da chave contactora	9
3	Defeito Inversor Azimute	11
4	Defeito Inversor Elevação	13

Tabela 1: Pinagem do conector S25 da gaveta de potência.

Pino	Função	Fio
1	+ da tensão de controle para o inversor azimute.	20
2	- da tensão de controle para o inversor azimute.	22
4	- da tensão de controle para o inversor elevação.	23
5	+ da tensão de controle para o inversor elevação.	24

Tabela 2: Pinagem do conector S26 da gaveta de potência.

Ao se ligar um inversor pela primeira vez em conjunto com o motor, alguns procedimentos básicos devem ser realizados, como por exemplo, o "faseamento" (*autotune*) motor-inversor. O procedimento de *autotune* permite ao inversor, o conhecimento de parâmetros fundamentais do motor para execução do controle de velocidade interno (PID). A cada troca de motor ou inversor, este procedimento deve ser realizado para perfeito funcionamento de ambos. Caso contrário, o erro ENC PH9 será mostrado no visor do *drive*, acusando problemas no *encoder*. O inversor necessita ainda de mais algumas informações (parâmetros), que devem ser fornecidas pelo o usuário (quantidade de pólos do motor, freqüência de alimentação, tipo do motor, etc.)

através de seu teclado. Para mais detalhes sobre os *menus* do inversor e sua configuração, favor consultar manual operacional da estação EMMN.

A configuração dos inversores é feita de duas formas, digitalmente através da entrada de parâmetros pelo seu teclado, e pelas ligações físicas nos seus *borners.* As figuras 6 e 7 apresentam essas ligações para cada um dos inversores.

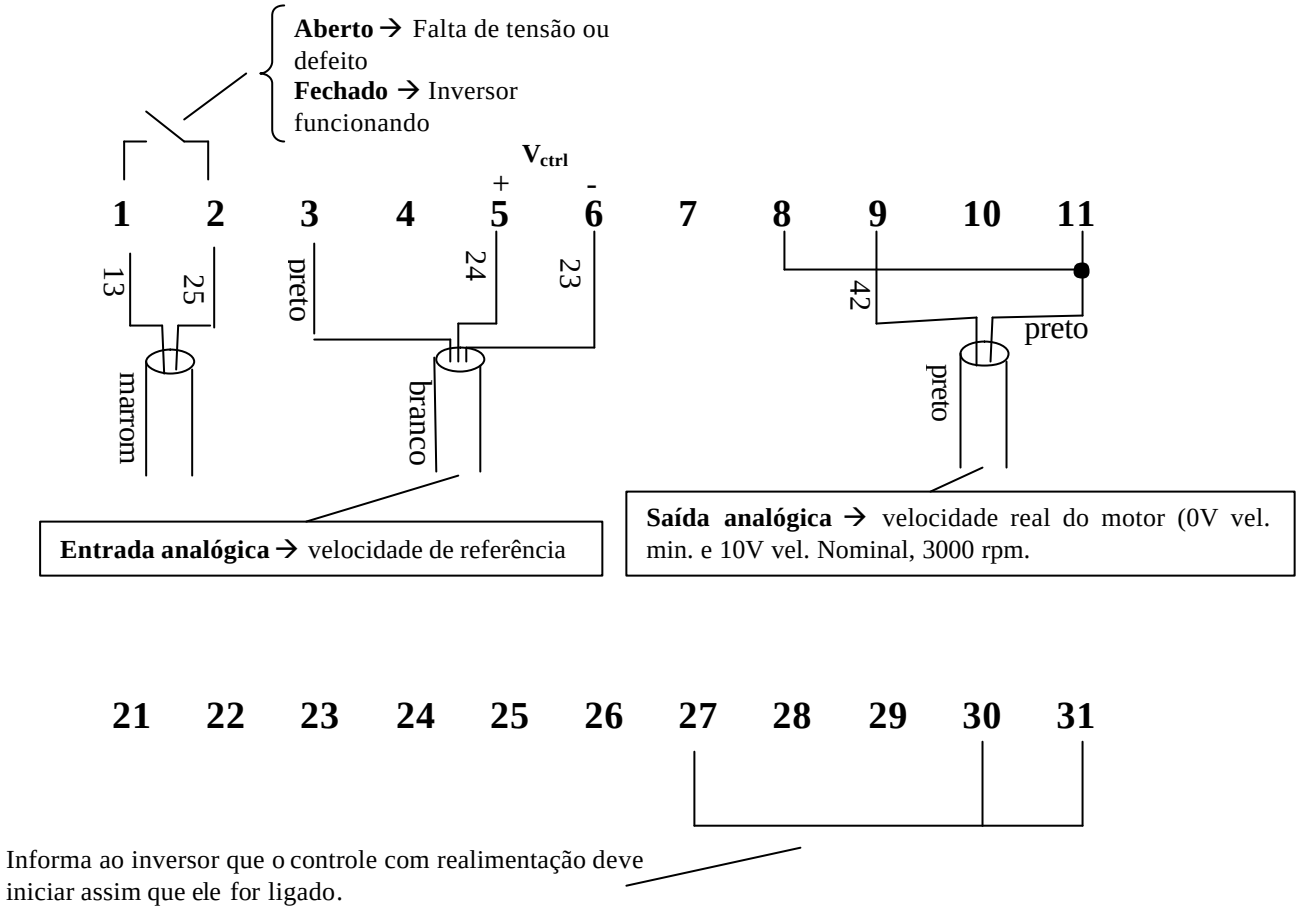

Figura 6: Configuração dos *bornies* da inversor azimute.

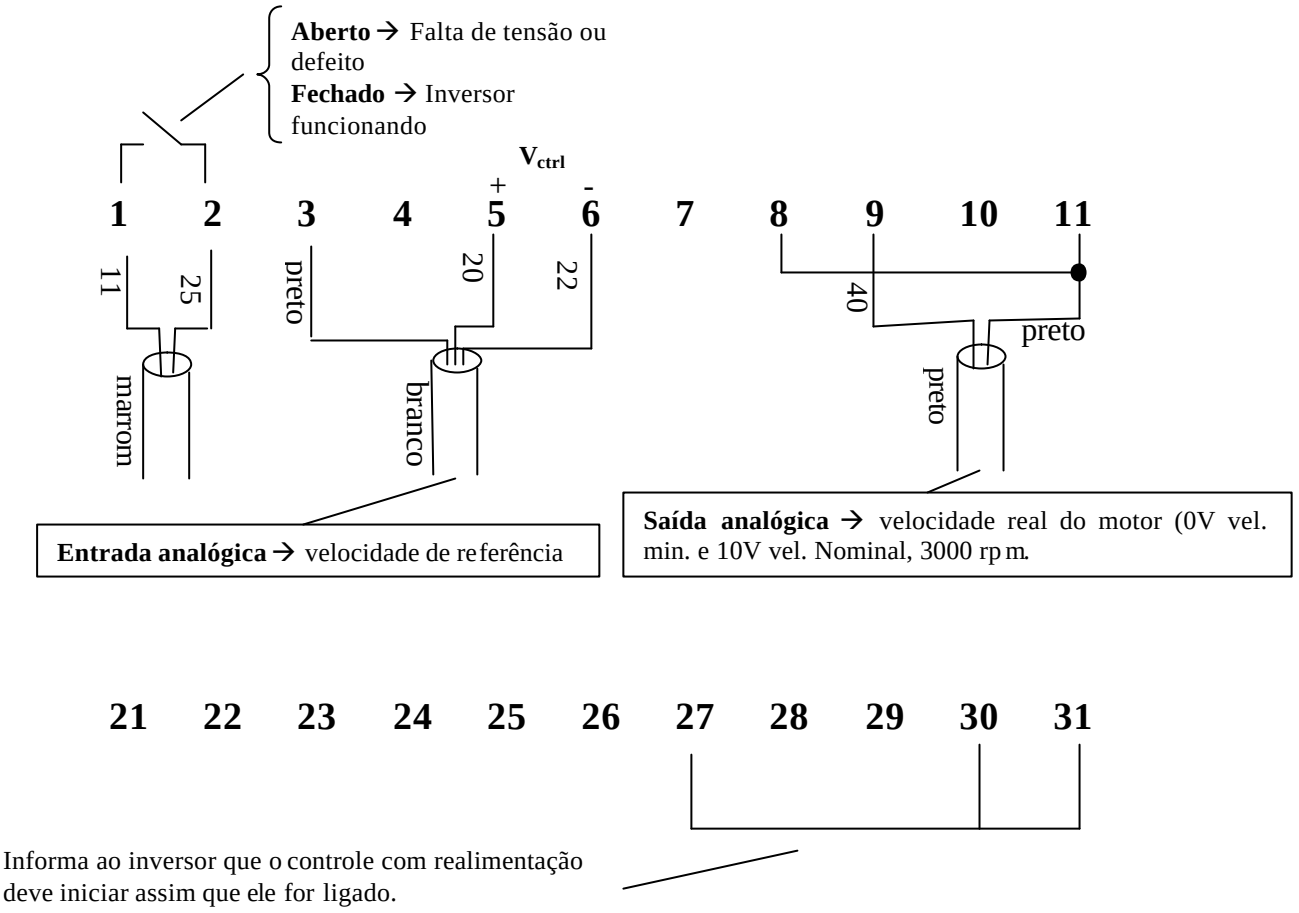

Figura 7: Configuração dos *bornies* do inversor elevação.

5. Módulo de Comando

O módulo de comando (gaveta de *pilotage*) suporta todos os circuitos de comando para a alimentação dos inversores, a placa conversora SOTEREM 2266-1, os LEDs para visualização de eventos, e as fontes de alimentação do sistema. Ela também abriga um autotransformador (230V/48V), que alimenta o circuito de aquecimento do módulo posicionador. Isso não é necessário no Brasil, em virtude do nosso clima quente, mas foi mantido apenas por compatibilidade com a antiga estação. Este circuito é fundamental em países de clima muito frio, onde os componentes do posicionador podem ser seriamente danificados por congelamento.

Logo após as devidas configurações no módulo de controle (calibração da placa AD/DA, configuração e inicialização do sistema), a chave principal na gaveta de *pilotage* deve ser acionada. Com isso, as fontes de alimentação são ligadas, e o *software* de rastreio poderá ser utilizado. O primeiro procedimento a ser executado antes de desligar o sistema, corresponde a desligar sua chave principal.

A interface do módulo de comando (com os demais módulos) é realizada através de saídas (conectores), identificados por uma letra e um número na parte de trás da gaveta de *pilotage*. Esta nomenclatura é semelhante à apresentada pela gaveta de potência. A figura 8 mostra a vista frontal e traseira da gaveta de *pilotage*.

(a) (b)

Figura 8: Gaveta de *pilotage*: (a) vista frontal e (b) vista traseira.

5.1 Placa SOTEREM 2266-1

A placa SOTEREM 2266-1 é responsável pela conversão dos sinais dos *resolvers* numa saída digital. Estes sensores de posição fornecem sinais analógicos, referentes ao seno e co-seno da posição angular, para cada eixo em que se encontram acoplados. O principal componente desta placa é o circuito integrado RDC 19220 (*Resolver to Digital Converter*), que transforma tais sinais numa palavra de 16 bits. Na placa se encontram instalados dois desses CIs, um para cada sensor.

A placa apresenta duas portas: uma para se comunicar com os dois *resolvers*, e outra para se comunicar com o computador e fontes de alimentação. Esta última é dividida em dois lados, que fornecem as posições no formato digital (16 *bits*) para os eixos, elevação (A) e azimute (B). A figura 9 apresenta um esquema básico desta placa.

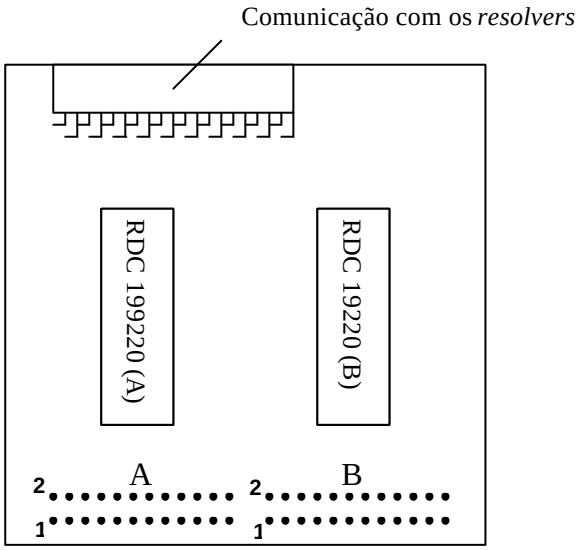

Figura 9: Esquema básico da placa SOTEREM 2266-1.

É interessante ressaltar que a pinagem dessa placa segue a descrição da figura 8, em que os pinos ímpares se encontram na parte de baixo (ver indicação do pino 1) e os pares na parte de cima. Esta identificação é de suma importância para que a placa funcione perfeitamente, evitando assim a danificação total ou parcial de seus componentes.

Os pinos 25A e 19B permitem controlar os instantes de leitura da posição da antena. Quando essas entradas se encontram em nível lógico alto (+5V), os dois RDCs 19220 ficam continuamente realizando o processo de conversão dos sinais analógicos. No entanto, o resultado dessas conversões não pode ser lido, porque os pinos de saída ficam com uma impedância alta. Em nível lógico baixo (0V), a conversão de dados é inibida, e o resultado da ultima conversão disponibilizado para leitura (os pinos de saída ficam em baixa impedância).

Em virtude da grande quantidade de *bits* para cada posição, seriam necessários 32 canais digitais para a placa conversora AD/DA no módulo de controle. Além destes canais reservados para a leitura, outros ainda seriam utilizados na execução de algumas tarefas (acender LEDs, acionar módulo de potência, etc.). Com a finalidade de reduzir essa quantidade de canais, os 16 *bits* de cada posição (azimute e elevação) são ligados em paralelo, e depois lidos em dois blocos. Inicialmente o pino 25A é habilitado[2] e o 19B desabilitado, deixando o lado A com a saída de dados em baixa impedância, e o B em alta. No instante seguinte, os estados são invertidos e uma nova leitura executada.

Para realizar as conexões necessárias na placa SOTEREM nesta configuração, e para fornecer uma interface de comunicação com o computador e fontes de alimentação (+-12V e +5V), a placa INPE-01 foi confeccionada (ver apêndice C). Ela é composta apenas por dois conectores (um DB20 e outro do tipo *phoenix*), e encontra-se acoplada a placa 2266-1 através das partes A e B. A figura 10 mostra uma foto das duas placas, uma sobre a outra, montadas na gaveta de comando. As tabelas 3, 4 e 5 apresentam informações sobre a pinagem da placa 2266-1.

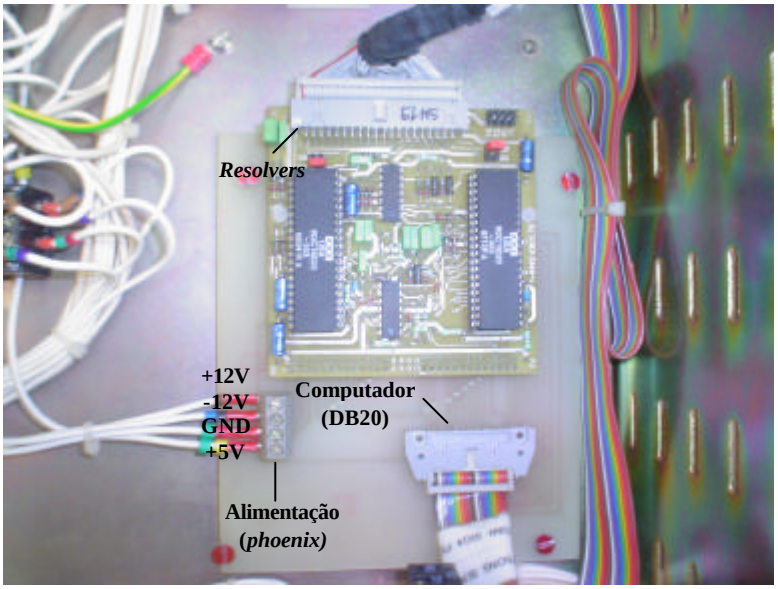

Figura 10: Vista superior da placa SOTEREM 2266-1 acoplada a placa INPE-01.

[2] Os pinos 25A e 19B são habilitados em nível lógico baixo.

Alimentação	
Pinos	**Tensão**
1A	-12V
2A	+12V
3A 4A 21B 22B	GND
5A 6A 23B 24B	+5V

Tabela 3: Alimentação da placa SOTEREM 2266-1.

Descrição	Lado A	Lado B
MSB	21	15
A14	22	16
A13	19	13
A12	20	14
A11	17	11
A10	18	12
A09	15	09
A08	16	10
A07	13	07
A06	14	08
A05	11	05
A04	12	06
A03	09	03
A02	10	04
A01	07	01
LSB	08	02

Tabela 4: Saída de dados da placa SOTEREM 2266-1.

Porta de Entrada		
Função	**Resolver A**	**Resolver B**
+REF	40	20
-REF	25	21
+C	06	04
-C	39	24
+S	05	03
-S	19	23

Tabela 5: Entrada de dados (*resolvers*) da placa SOTEREM 2266-1.

5.2 Circuitos de Comando

Todos os circuitos de comando para acionamento do módulo de potência encontram-se alojados na gaveta de *pilotage*. Eles estão ligados a circuitos externos, responsáveis pelo sistema de segurança da estação (fim de curso e botão de emergência). A figura 11 apresenta a vista superior do módulo de comando.

Figura 11: Vista superior da gaveta de *pilotage*.

A alimentação da gaveta de potência é realizada através do conector S9, onde o pino 1 corresponde ao neutro (L6), o pino 2, a fase (L7), e o pino 3, ao terra. Dentro da gaveta, os fios são identificados por dois números[3], e os seus componentes (fontes, relés, fusíveis, etc.) por rótulos. Os esquemas apresentados nesta seção utilizarão esta configuração para análise. Alguns dos fios terminam em conectores, que são devidamente identificados com sua referência e pinos utilizados. Sempre que um deles é encontrado, seu respectivo cabo é informado entre colchetes (por exemplo: S9[J9] – conector S9 e cabo J9). A figura 12 mostra um exemplo da representação adotada nos esquemas.

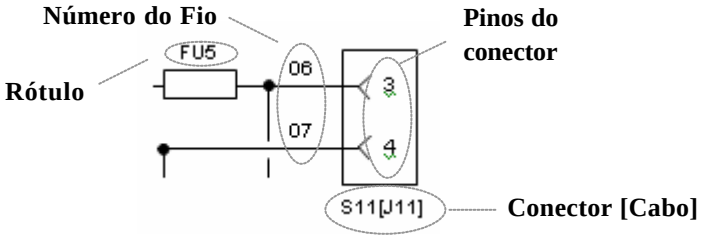

Figura 12: Exemplo de representação dos componentes nos esquemas.

A figura 13 apresenta o esquema geral de alimentação interna da gaveta de *pilotage*. Quando a chave principal do módulo de comando é acionada, todos os seus contatos 3-4 são fechados (ver figuras 13 e 14), permitindo a alimentação das fontes AL1 e AL2. Conseqüentemente, a bobina do relé KA3 é também alimentada (figura 15) e seu contato 1-3 fechado (figura 13), pois a fonte AL2 passa a fornecer 24V aos seus

[3] Exceto os fios L6, L7 e alguns dos fios ligados a *leds*, que são identificados por três números.

terminais 2-10. Com a chave principal ligada, caso o botão de emergência não esteja pressionado (pinos 5 e 6 do conector S11 em curto), a bobina do relé KA1 é energizada[4] (figura 14).

O contato B4-B5 referente à placa SOTEREM 2483 (ver figura 14) é responsável pelo acionamento do módulo de potência, e comutado apenas através do módulo de controle, por meio da placa conversora AD/DA. Se o fim de curso não estiver acionado (pinos 1 e 2 do conector S11 em curto) e B4-B5 fechado, a bobina KA2 é energizada[5], fechando seus contatos 1-3 da figura 13 e 6-7 da figura 15. Dessa forma, o módulo de potência é ligado e o freio do motor elevação retirado.

A bobina do relé KA2 é desenergizada quando uma das seguintes condições ocorrer: fim de curso acionado, contato B4-B5 em aberto, ou bobina de KA1 desenergizada. A primeira ocorrerá quando a antena chegar em um de seus fins de cursos, provocando um circuito aberto entre os pinos 1 e 2 do conector S11. O segundo é de responsabilidade do módulo de controle, e o terceiro é provocado pela abertura da chave principal, ou pelo acionamento do botão de emergência. O *status* das bobinas KA1 e KA2 é transmitido ao módulo de controle, através dos pinos 5 e 18 do conector S7 (figura 16).

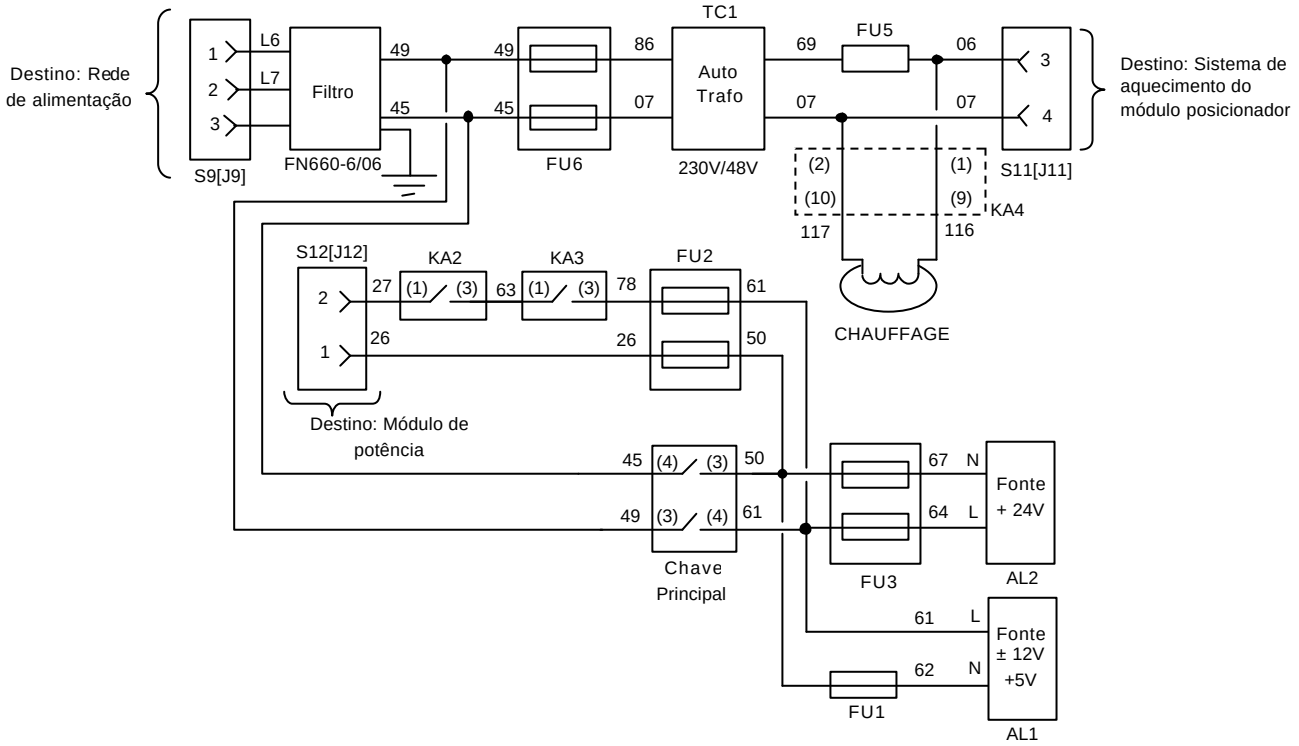

Figura 13: Circuito de comando referente a alimentação interna da gaveta de *pilotage*.

[4] Os contatos 3-1 e 7-6 da figura 14 são fechados quando a bobina do relé KA1 é energizada.

[5] Supondo a bobina do relé KA1 previamente energizada

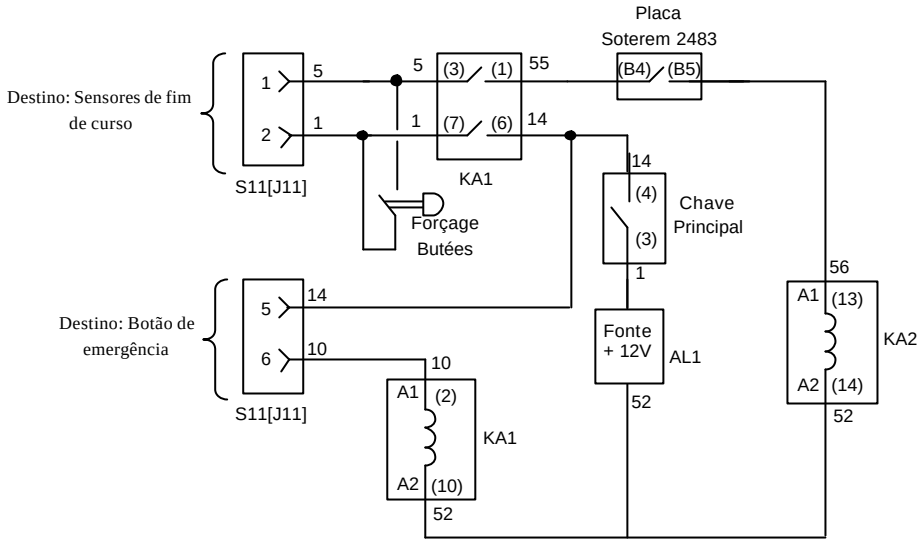

Figura 14: Circuito de comando referente a segurança da gaveta de *pilotoge*.

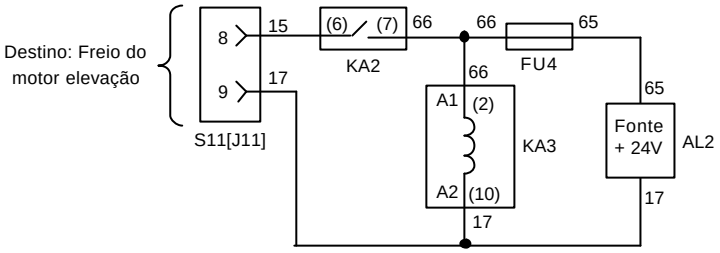

Figura 15: Circuito de comando referente ao freio do motor elevação.

A bobina do relé KA4 é energizada somente quando o botão *test voyants* for pressionado (figura 16), transferindo a alimentação do *led Chauffage* de 48V AC para 5V DC (figura 17). Com isso, seu contato 11-7 é fechado, sinalizando ao módulo de controle que o evento "teste de *leds*" foi solicitado pelo usuário. O computador como resposta, envia comandos para acender todos os *leds* da gaveta de *pilotage*. Sua ligação com o módulo de controle é realizada através do pino 6, do conector S7, por onde é transmitido o estado do relé KA4.

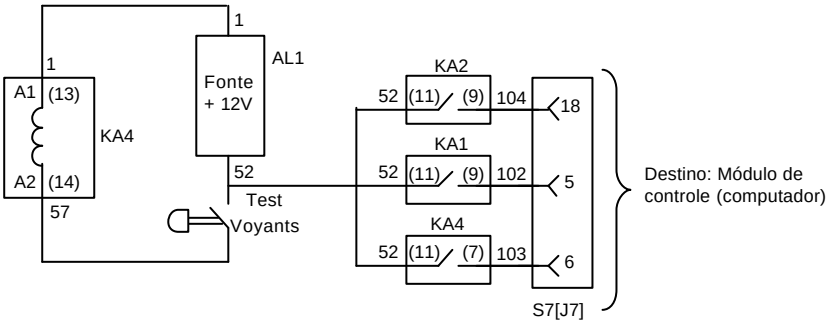

Figura 16: Circuito de comando referente ao evento "teste de *leds*.".

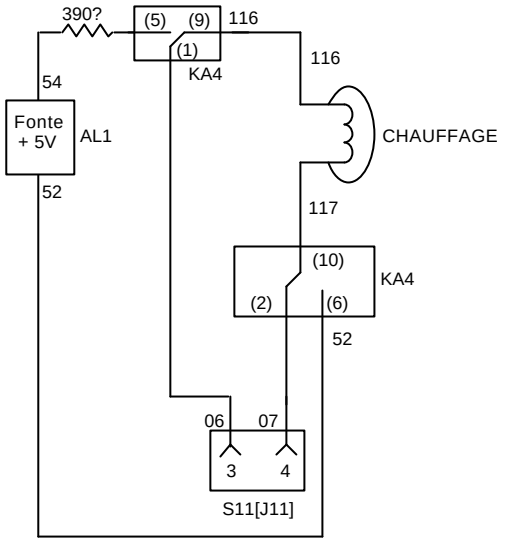

Figura 17: Circuito de comando referente a mudança na alimentação do *led Chauffage*.

Um dos componentes importantes do módulo de comando corresponde à placa SOTEREM 2483. Ela é composta basicamente por dois resistores, um relé, um transistor e um diodo, dispostos de acordo com o esquema da figura 18. Sua função é permitir o acionamento do módulo de potência através do computador.

Uma das saídas digitais da placa AD/DA é ligada ao pino 23, do conector S7, e satura o transistor da placa 2483, quando seu nível lógico encontra-se em alto (5V). Dessa forma, a bobina do relé AG5013 é energizada, e o contato B4-B5 fechado. Isso possibilita o acionamento da gaveta de potência, caso as demais condições sejam satisfeitas (KA1 energizada, por exemplo). Com o nível lógico baixo (0V), o transistor corta e a bobina do relé é desenergizada.

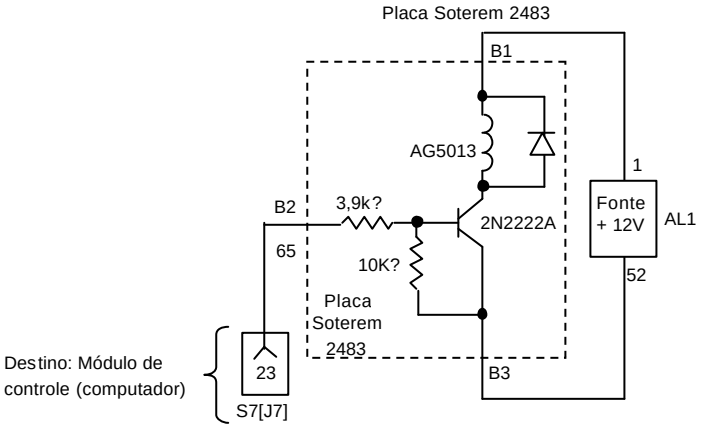

Figura 18: Circuito da placa SOTEREM 2483 dentro do módulo de comando.

As tabelas 6 e 7 apresentam a pinagem dos conectores S7 e S8 da gaveta de *pilotage*, com os respectivos canais digitais da placa AD/DA (PCI 6025E) que estão ligados aos pinos. A tabela 8 mostra a descrição de alguns fios do módulo de comando.

Conector S7		Placa PCI 6025E		Descrição
Pino	Fio	ID	Pino	
1	Preto	PC6	53	LED Manuel
2	Rosa	DIO0	25	LED Butées
3	Branco	DIO2	29	LED Poursuite
4	Laranja-Branco	DIO3	31	LED Arrêt d'urgence
5	Preto-Branco	PC1	63	FIO 102 (KA1)
6	Vermelho	PC3	59	FIO 103 (KA4)
7	--	--	--	--
8	--	--	--	--
9	Laranja	GND	70	Terra
10	Marron-Branco	DIO7	32	FIO 53 - Defeito ELE
11	--	--	--	--
12	--	--	--	--
13	--	--	--	--
14	Verde-Branco	PC7	51	LED Servei
15	Verde	DIO4	26	LED Puissance
16	Vermelho-Branco	DIO5	28	LED Défaut
17	Marrom	PC0	65	FIO 104 (KA2)
18	Azul-Branco	PC2	61	Chave Manuel
19	--	--	--	--
20	--	--	--	--
21	--	--	--	--
22	Amarelo	DIO6	30	FIO 47 - Defeito AZ
23	Azul	DIO1	27	FIO 6 (B2) - Base transistor
24	--	--	--	--
25	--	--	--	--

Tabela 6: Pinagem do conector S7 da gaveta de *pilotage*.

Conector S8		Placa PCI 6025E		Descrição
Pino	Fio	ID	Pino	
1	--	--	--	--
2	Verde-Cinza	PC4	57	Libera leitura Azimute (19B)
3	Laranja-Cinza	PB7	67	A15 (MSB)
4	Azul-Cinza	PB5	71	A13
5	Roxo-Branco	PB3	75	A11
6	Marrom-Branco	PB1	79	A09
7	Amarelo-Cinza	PA7	83	A07
8	Verde	PA5	87	A05
9	Vermelho-Cinza	PA3	91	A03
10	Branco	PA1	95	A01
11	--	--	--	--
12	--	--	--	--
13	--	--	--	--
14	Amarelo por fora	GND	94	Terra
15	Rosa	PC5	55	Libera leitura Elevação (25A)
16	Laranja	PB6	69	A14
17	Azul	PB4	73	A12

18	Roxo	PB2	77	A10
19	Marrom	PB0	81	A08
20	Preto	PA6	85	A06
21	Vermelho	PA4	89	A04
22	Amarelo	PA2	93	A02
23	Branco-Cinza	PA0	97	A00 (LSB)
24	--	--	--	--
25	--	--	--	--

Tabela 7: Pinagem do conector S8 da gaveta de *pilotage*.

Fio	Descrição (Rótulo)
L7,61,49,64 e 86	230V (TC1)
L6, 62, 45, 67e 7	Neutro (TC1)
06 e 69	48V AC (TC1)
52	GND (AL1)
1	+12V (AL1)
60	-12V (AL1)
54	+5V (AL1)
65	+24V (AL2)
17	GND (AL2)

Tabela 8: Descrição de alguns fios da gaveta de *pilotage*.

6 Módulo de controle

Este módulo é composto pelo computador e pela placa conversora AD/DA (PCI 6025E) da *National Instruments*, que dispõe de 32 canais digitais para leitura (*input*) ou escrita (*output*), além de duas saídas analógicas. Dentre os canais digitais, 24 são fornecidos através do CI 82C55 (*Programmable Peripheral Interface - PPI*), dispostos em 3 portas (PA, PB e PC) de 8 canais cada. A configuração destes canais como entrada ou saída não pode ser realizada individualmente, como no caso dos canais DIO (*Digital Input Output*). Se um dos canais da porta A for configurado como entrada, todos os demais desta porta também serão. O mesmo vale para a porta B.

A porta C difere um pouco em relação às outras, pois o seu sistema agrupa a configuração dos canais de 4 em 4, ou seja, caso o canal PC0 seja escolhido como saída, os canais PC1, PC2 e PC3 também terão a mesma configuração. Com isso, os canais PC4, PC5, PC6 e PC7 podem ser configurados de uma forma diferente (como entrada, por exemplo). A tabela 9 apresenta todos os canais digitais da placa AD/DA e as suas respectivas configurações (*Input* ou *Output*).

Placa PCI 6025E			
ID	**Pino**	**Descrição**	**Configuração**
PA0	97	A00 (LSB)	INPUT
PA1	95	A01	INPUT
PA2	93	A02	INPUT
PA3	91	A03	INPUT
PA4	89	A04	INPUT
PA5	87	A05	INPUT
PA6	85	A06	INPUT
PA7	83	A07	INPUT
PB0	81	A08	INPUT

PB1	79	A09	INPUT
PB2	77	A10	INPUT
PB3	75	A11	INPUT
PB4	73	A12	INPUT
PB5	71	A13	INPUT
PB6	69	A14	INPUT
PB7	67	A15 (MSB)	INPUT
PC0	65	FIO 104 (KA2)	INPUT
PC1	63	FIO 102 (KA1)	INPUT
PC2	61	Chave Manuel	INPUT
PC3	59	FIO 103 (KA4)	INPUT
PC4	57	Libera leitura Azimute (19B)	OUTPUT
PC5	55	Libera leitura Elevação (25A)	OUTPUT
PC6	53	LED Manuel	OUTPUT
PC7	51	LED Survie	OUTPUT
DIO0	25	LED Butées	OUTPUT
DIO1	27	FIO 6 (B2) - Base transistor	OUTPUT
DIO2	29	LED Poursuite	OUTPUT
DIO3	31	LED Arrêt d'urgence	OUTPUT
DIO4	26	LED Puissance	OUTPUT
DIO5	28	LED Defáut	OUTPUT
DIO6	30	FIO 47 - Defeito AZ	INPUT
DIO7	32	FIO 53 - Defeito ELE	INPUT

Tabela 9: Configuração dos canais digitais da placa AD/DA (PCI 6025E).

7 Software de Controle e Rastreio

A versão atual do software de controle e rastreio da estação EMMN é a 0.2 (*CITO0v2)*, que prevê o uso de arquivos no formato texto contendo as efemérides da rota desejada (passagem do satélite). O objetivo principal do programa é movimentar a antena adequadamente, através das técnicas de controle (algoritmo PI), dentro das posições e horários fornecidos pelo arquivo fonte. Para tal é necessário um ajuste das rotas fornecidas em relação ao sistema de referência da antena, pois normalmente existe um *offset* entre ambos. Dessa forma o *software* de controle e rastreio é dividido em duas partes: uma que trata das técnicas de controle, e outro que realiza o ajuste da rota fornecida. A seção 12 apresenta a problemática do sistema de referência da antena, e as soluções encontradas para a realização do rastreio. Atualmente as efemérides são geradas manualmente com uso de um programa no *Windows.*

Essa versão engloba uma série de subsistemas responsáveis por atividades isoladas, como por exemplo, a monitoração do sistema de acionamento (gaveta de *puissance*). Outra funcionalidade desta versão corresponde à geração de arquivos de log, o que permite o registro de eventos inesperados no decorrer da operação da estação. Os tópicos abordados nas próximas seções serão:

- Requisitos do sistema
- Características da versão 0.2
- Estrutura do programa
- Funções criadas
- Implementações futuras

8. Requisitos do sistema

O *software* de controle foi desenvolvido em C/C++, com uso das bibliotecas do projeto GNU e *Comedi*. A tabela 10 apresenta a listagem dos requisitos básicos do sistema e suas respectivas versões para compilação do *CITO0v2*.

	Versão	Descrição
S.O Linux	3.0r0	Distribuição Linux Debian
Comedi	0.7.63-2	Drives para a placa AD/DA
Comedilib	0.7.18-1	Bibliotecas de programação em C
G++	2.95	Compilador C++
Kernel	2.4.31	Kernel mais novo da versão 2.4

Tabela 10: requisitos do sistema.

Com os requisitos mínimos do sistema devidamente instalados, o *software* de rastreio poderá ser compilado com o seguinte comando:

[neo@zion]$ g++ -lcomedi -lpthread CITO0v2.c efem.cpp -o CITO0v2

8.1 Integração da Placa PCI 6025E

Alguns problemas sugiram na integração da placa AD/DA PCI 6025E. Apesar da instalação adequada dos *drives* do *Comedi*, o *subdevice* 3 responsável por 24 canais digitais, não foi reconhecido pelo computador. Após uma consulta ao manual do fabricante, descobriu-se que parte de seus canais digitais (exatamente 24) era fornecida através do CI 82C55A (*Programmable Peripheral Interface* - PPI). A medida tomada foi consultar a documentação do *Comedi* na *internet*, o que resultou na seguinte recomendação: para placas com presença desse CI, seria necessária uma nova compilação do *Comedi* incluindo dois *drives*, um para a placa PCI 6025E e outra para o 82C55A. Quando o módulo do primeiro fosse carregado, este trataria de carregar o segundo automaticamente.

Apesar de todos os procedimentos adotados corretamente, o computador não conseguiu carregar os dois módulos de uma vez. Sempre o *8255.o* conduzia a um erro (ver figura 19). Depois de horas de muita pesquisa na *net*, descobriu-se que o problema estava justamente no *Comedi*. Em uma das listas do projeto, um de seus desenvolvedores reconheceu o erro e forneceu uma solução para o impasse. O procedimento adotado para instalação dos *drives* supracitados será descrito no próximo item.

```
MITE:0xdf001000 mapped to d0903000 DAQ:0xd002000 mapped to d0905000
( irq = 5 ) 8255 support not configured -- disabling subdevice
```

Figura 19: Erro apresentado pelo computador.

8.1.1 Procedimento para instalação do *Comedi*

O *Comedi* é um conjunto de *drives* para dispositivos de aquisição de dados com suporte a cerca de 200 placas de diferentes fabricantes. É necessário adicionar o(s) modulo(s) desejados ao *kernel* do Linux, pois somente este pode tratar diretamente com o *hardware* do PC. Alguns passos devem ser adotados para o processo de inclusão da placa ao computador:

1. Saber se o computador reconheceu a placa no *slot* PCI.

2. Recompilar o *kernel* para obter um fonte completo. Isso é primordial no processo de instalação, uma vez que muitas distribuições quando instaladas apresentam um *kernel* pré-compilado que nem sempre é completo. O *Comedi* necessita dessa condição, e sem ela nada poderá ser realizado.
3. Compilar os módulos do *Comedi*.
4. Solucionar o problema da placa PCI 6025E devido ao uso do CI 82C55.
5. Instalar o *Comedilib* para dar suporte a programação em C.

8.1.2 Checar o barramento PCI

No *prompt* de comando digite:

[neo@zion]$ /sbin/lspci.

A saída deve ser semelhante à figura 20, com a presença da placa da *National Instruments* na listagem dos itens reconhecidos pelo barramento PCI. Neste momento o computador sabe apenas o nome do dispositivo, porém não como operá-lo.

```
00:00.0 Host bridge: Intel Corp. 440FX - 82441FX PMC [Natoma] (rev 02)
00:07.0 ISA bridge: Intel Corp. 82371SB PIIX3 ISA [Natoma/Triton II]
(rev 01)
00:07.1 IDE interface: Intel Corp. 82371SB PIIX3 IDE [Natoma/Triton
II]
00:0b.0 VGA compatible controller: Matrox Graphics, Inc. MGA 2064W
[Millennium] (rev 01)
00:0f.0 Class ff00: National Instruments: Unknown device 2a70
00:11.0 Ethernet controller: 3Com Corporation 3c900 Combo [Boomerang]
```

Figura 20: Saída do comando /sbin/lspci

8.1.3 Fonte do *kernel*

O *kernel* deve ser recompilado para tornar possível o uso do *Comedi*, tendo em vista a necessidade do seu fonte estar completo. Este documento não tem como objetivo a descrição dos passos necessários para esta tarefa, uma vez que a literatura vigente aborda bem este tema.

8.1.4 Módulos *Comedi*

O próximo passo corresponde a compilação dos módulos do *Comedi*. O fonte do *Comedi* pode ser adquirido no *site* oficial do projeto ou utilizando o aplicativo *dselect* (gerenciador de pacotes) da distribuição Debian 3.0r1. No último caso, o Debian disponibiliza um arquivo compactado no diretório */usr/src*. Deve-se também instalar o pacote referente às bibliotecas de programação em C (*Comedilib*). Descompacte o arquivo fonte utilizando o comando **tar -xzvf comedi.tar.gz**. Em seguida, no interior do novo diretório criado, os passos abaixo devem ser executados:

1. Rodar o programa de configuração com o seguinte comando: **./configure**
2. Responder as questões solicitadas. Quando perguntado sobre usar o modo *Verbose Debugging*, responda **n** como indicado na figura 21. Se ele for ativado, mensagens indesejáveis poderão aparecer na tela quando os programas forem executados.

Utiliza-se esta opção quando é desejada a depuração de algum *software* criado pelo usuário que não está funcionando adequadamente. Dê suporte ao fabricante *national* e a placa PCI 6025E. Não esquecer de incluir o *drive* da placa 8255.

```
Enter location of Linux source tree /usr/src/linux-2.4.18-27.7.x
Verbose Debugging (CONFIG_COMEDI_DEBUG) [Y/n/?] n
Kernel Comedilib (CONFIG_COMEDI_KLIB) [N/m/?] n
*
* Hardware device drivers
*

Data Translation boards (CONFIG_COMEDI_DT) [N/y/?] n
National Instruments boards (CONFIG_COMEDI_NI)[Y/n/?] y

      AT-MIO E series (CONFIG_COMEDI_NI_ATMIO) [M/n/?] n
      PCMCIA MIO E series (CONFIG_COMEDI_NI_MIO_CS) [M/n/?] n
      NI-DAQ-DIO24 (CONFIG_COMEDI_NI_DAQ_DIO24) [M/n/?] n
      PCI-MIO E series (CONFIG_COMEDI_NI_PCIMIO) [M/n/?] m
      NI PCI-DIO series (CONFIG_COMEDI_NI_PCIDIO) [M/n/?] n
      NI 670x series (CONFIG_COMEDI_NI_670X) [M/n/?] n
      AT-MIO-16D (CONFIG_COMEDI_NI_ATMIO16D) [M/n/?] n
      AT-A2150 (CONFIG_COMEDI_NI_AT_A2150) [M/n/?] n
      LAB-PC and compatibles (CONFIG_COMEDI_NI_LABPC) [M/n/?] n
      AT-AO-6/10 (CONFIG_COMEDI_NI_AT_AO) [M/n/?] n
      DAS08 compatible driver (CONFIG_COMEDI_DAS08) [N/m/?] n
      DAS16 compatible driver (CONFIG_COMEDI_DAS16) [N/m/?] n

      ...

      Generic 8255 support (CONFIG_COMEDI_8255) [M/n/?] m

      ...
```

Figura 21: Saída do comando ./configure.

3. Executar os seguintes comandos:
   make – compilar os módulos selecionados
   make install – instalar o *Comedi.*
   make dev – criar os *devices* (/dev/comedi0)
4. Retirar o *bug* do *Comedi* através dos comandos:
   make clean
   make CFLAGS=-DCONFIG_COMEDI_8255=1

9. Características da versão 0.2

A versão 0.2 foi criada para dar suporte a nova configuração de *hardware* da estação, uma vez que o protótipo montado em bancada foi reestruturado e alocado numa gaveta industrial da antiga estação SACI. O novo *hardware* dispõe de LEDs para indicar a ocorrência de algum evento (fim de curso acionado, inversor com defeito,

etc.), além de botões e chaves para interação com o usuário. Em relação a versão anterior, o *software* atual apresenta as seguintes atualizações:

- Mudança na configuração dos canais
- Mudança na estrutura das *threads*
- Exclusão do código referente ao freio do motor elevação
- Acionamento da gaveta de potência (*puissance*) através do computador

Para que o *software CITO0v2* funcione adequadamente é necessário que a rotina *rc.local2* seja executada primeiro, pois ela é responsável por configurar e calibrar a placa, carregar os módulos do *Comedi* e inicializar o sistema. A figura 22 apresenta a listagem deste *Shell script*.

```
#! /bin/bash

echo "Carregando modulos comedi"
/sbin/modprobe ni_pcimio
/sbin/modprobe 8255
echo "OK"

echo "Configurando Placa"
/usr/sbin/comedi_config /dev/comedi0 ni_pcimio
echo "OK"

echo "Calibrando Placa"
/usr/bin/comedi_calibrate -f /dev/comedi0
echo "OK"

echo "Inicializando o sistema"
/home/neo/citosina/startup
echo "OK"
```

Figura 22: Listagem da rotina *rc.local2*.

A versão anterior desta rotina (*rc.local1*), não executava a etapa de inicialização do sistema, uma vez que o *hardware* acoplado a placa era mais simples do que o atual. Em virtude das alterações já mencionadas, antes da gaveta de *pilotage* ser ligada, os canais digitais da placa devem ser configurados. Os canais destinados a acionar algum dispositivo (LEDs, ligar gaveta de potência, etc.) devem manter um nível lógico adequado, evitando problemas ao ligar o sistema pela chave principal da gaveta de *pilotage*.

Quando a placa AD/DA passa pelas etapas de carregar módulos (etapa 1), configurar (etapa 2) e calibrar (etapa 3), seus canais digitais permanecem em nível lógico alto[6] (5V). Nesse momento, caso a chave principal seja acionada, a gaveta de potência ligará e possivelmente causará danos a estrutura física da estação. Outros transtornos podem ser gerados, como acendimento inadequado de LEDs, indicando eventos que não ocorreram.

Em virtude da necessidade de uma prévia configuração dos canais digitais da placa, um novo *software* (*startup*) foi desenvolvido para dar suporte a versão 0.2 do *software* de rastreio. Seu objetivo é preparar a placa a AD/DA para trabalhar em harmonia com o *hardware*, mesmo que o programa de rastreio não esteja sendo

[6] Nem sempre isto ocorre. Alguns canais digitais podem iniciar em nível lógico baixo (0V), ao invés de alto (5V).

executado. Dessa forma é possível que o usuário do sistema mantenha a gaveta de *pilotage* e o computador, ambos ligados, sem a necessidade do *CITO0v2* ser executado.

Com esta nova versão do *software* de rastreio (0.2), além do programa *startup*, outros dois *softwares* de menor porte foram desenvolvidos, com o objetivo de ligar e desligar a gaveta de potência, pois tal tarefa não é mais realizada através de botoeiras. Os programas *ligar* e *desligar* são utilizados somente para situações de teste ou emergência, pois o *CITO0v2* proporciona o controle automático do sistema de potência. O apêndice B apresenta a listagem dos programas *startup*, *ligar* e *desligar*.

O *hardware* atual da estação EMMN não necessita mais que o *software* de rastreio controle o freio do motor elevação[7]. Esta modificação foi realizada na função *escritaELE(double volts)*, responsável por escrever o valor presente em *volts* na saída da placa AD/DA. Quando o valor passado em seu argumento era "0", o freio do motor era acionado, evitando que o próprio peso da antena modificasse sua posição atual. Caso qualquer outro valor (entre -10V a 10V) fosse passado, o freio era retirado, e a escrita realizada. Na versão 0.2 do *software* de rastreio, a função *escritaELE(double volts)* não mais atua no acionamento do freio.

10. Estrutura do programa

O *software* de rastreio *CITO0v2* foi desenvolvido utilizando-se *threads* para a execução das diversas tarefas propostas. Basicamente são cinco as atividades realizadas pelo sistema: **monitoração e registro**, **impressão**, **apontamento**, **rastreio e interação com o usuário**. A figura 23 apresenta o esquema de criação das *threads* responsáveis por estas tarefas.

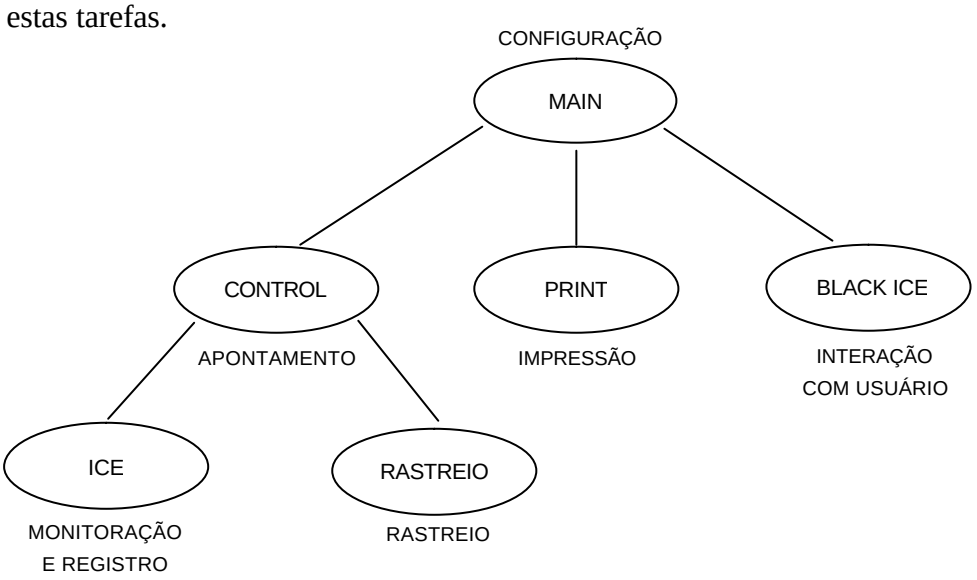

Figura 23: Esquema geral das *threads*

Inicialmente a função *main* abre o *device*, faz as devidas configurações dos canais digitais da placa (escrita ou leitura), carrega o arquivo com as efemérides, e em seguida cria as *threads control*, *print* e *black ICE*. Apesar da *thread black ICE* ter sido criada, está encontra-se inativa no sistema neste momento, aguardando um sinal[8] para iniciar suas atividades.

[7] A retirada do freio no motor elevação é realizada agora pelo módulo de comando.
[8] Este sinal é fornecido através de *Condition Variables*.

Após o estágio de configuração realizado pela função *main*, a *thread* de impressão passa a mostrar na tela a contagem regressiva para o início da passagem do satélite. Quando restam 60 segundos, a contagem é parada (não é mais exibida na tela) e a gaveta de potência é acionada, além do respectivo LED que informa essa situação. Na seqüência, a *thread* de apontamento (*control*) é criada, e passa a atuar, posicionando a antena no primeiro ponto desejado, de acordo com o arquivo de efemérides. A partir daí é apresentado ao usuário um conjunto de informações importantes, como a posição atual da antena, o valor dos sinais de controle, os erros para cada posição e as respectivas referências. Também é incluso o valor das partes integrativas de cada sinal de controle, para análise e aperfeiçoamento da técnica de *anti-reset windup*. A figura 24 apresenta um exemplo dessas informações na tela.

```
ELE=29.904364   AZ=100.069277   outvELE=0.044073   outvAZ=0.065797
IELE=-0.013309   IAZ=-0.024230   erroELE=0.095636   erroAZ=-0.069277
refELE=30   refAZ=100
```

Figura 24: Saída de dados do algoritmo de rastreio.

Quando restam 10 segundos para a passagem do satélite, a *thread* de apontamento cria a *thread* de rastreio[9], e em seguida se finaliza. As informações da figura 24 continuam a serem exibidas na tela, porém a uma taxa menor do que antes, em virtude do período de amostragem ser maior (1s) nesta fase.

A monitoração da estação EMMN em relação a eventos inesperados é de responsabilidade da *thread ICE* (*Invasion Counter Electronic*s). Os eventos previstos por este subsistema são: fim de curso acionado, botão de emergência acionado, teste de *LEDs* solicitado pelo usuário, e inversor ELEVACAO e/ou AZIMUTE com defeito ou desligado. Quando esta *thread* detecta que um desses eventos ocorreu, ela cancela a *thread* responsável pela impressão (*print*), a *thread* de apontamento (*control*), a *thread* de rastreio (*rastreio*), e em seguida aciona o(s) *LED*(*s*) da gaveta de *pilotage* referente ao evento ocorrido. Com isso, a rotina ICE ativa a *thread blackICE*, que será encarregada de interagir com o usuário[10], informando a situação atual do sistema. Em seguida a *ICE* é finalizada. A figura 25 apresenta esta situação.

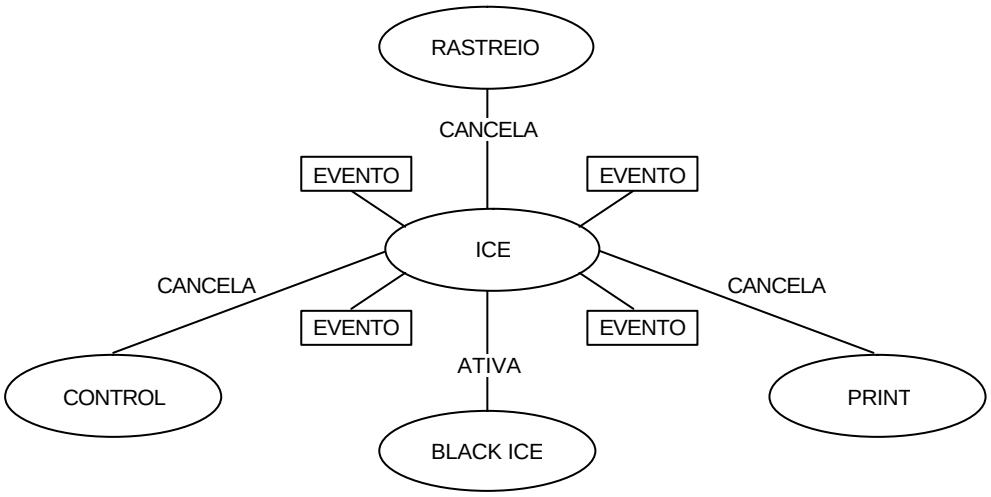

Figura 25: Esquema de ativação da *thread* ICE.

[9] Quando criada, a *thread rastreio* aciona o LED *Poursuite*.
[10] Caso o fim de curso seja acionado, a rotina *blackICE* tem condições de retirar a antena desta condição. Tal procedimento é descrito no apêndice A.

11 Funções do programa

No desenvolvimento do *CITO0v2* foram criadas 3 funções responsáveis pela escrita dos sinais de controle, e pela leitura da posição da antena, tanto no azimute quanto na elevação. A tabela 11 apresenta tais funções e seus respectivos argumentos, além de uma descrição básica do funcionamento de cada uma delas.

Nome	Argumento	Descrição
pos()	--	Atualiza os valores das variáveis globais AZ e ELE reponsáveis pelas posições *azimute* e *elevação* da antena.
escritaELE(double volts)	Valor da tensão a ser aplicada	Escreve o valor presente em *volts* na saída da placa AD/DA responsável pelo acionamento da elevação.
escritaAZ(double volts)	Valor da tensão a ser aplicada	Escreve o valor presente em volts na saída da placa AD/DA responsável pelo acionamento do azimute.

Tabela 11: Funções criadas para o *software CITO0v2*.

12 Sistema de Posição da Antena

Um fator importante para o *software* de controle e rastreio corresponde à escolha adequada do sistema de posição para a antena. Fisicamente, ela pode girar de 0 a 360 graus no eixo azimute, e de -3 a 182 graus na elevação. Essa folga de alguns graus para cima e para baixo na elevação é utilizada pelos sensores de fim de curso, para acionar o sistema de proteção e desligar o módulo de potência, caso a antena não pare.

Quando os *resolvers* foram instalados no eixo da antena, o seu zero não foi colocado no zero desejado para o sistema, o que requer uma posterior adaptação na leitura, pelo *CITO0v2.* A partir daí, uma equivalência entre escalas é realizada, levando-se em consideração os valores extremos para cada uma. As equações 1 e 2 foram obtidas por uma simples regra de três composta, de acordo com a figura 26. Deve-se notar que um giro na antena, de um fim de curso a outro, tanto no azimute, quanto na elevação, corresponde a menos que um giro completo do respectivo *resolver*. Desta forma evita-se ambigüidade na leitura da posição da antena.

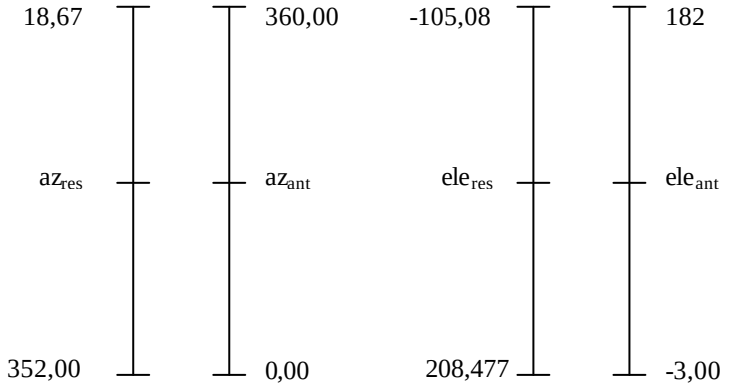

Figura 26: Equivalência entre escalas para os eixos azimute e elevação.

$$az_{ant} = -1.08 \cdot az_{res} + 380.16 \qquad (01)$$

$$ele_{ant} = -0.59 \cdot ele_{res} + 120 \quad (02)$$

A posição angular da antena no eixo azimute, logo após a conversão entre escalas, é representada por az_{ant}, e no eixo elevação por ele_{ant}. As equações 3 e 4 são utilizadas na conversão dos valores fornecidos pela placa SOTEREM 2266-1 (res_{az} e res_{ele}), em valores angulares (az_{res} e ele_{res}) com unidade em graus.

$$az_{res} = res_{az} \cdot \frac{360}{65535} \quad (03)$$

$$ele_{res} = res_{ele} \cdot \frac{360}{65535} \quad (04)$$

12.1 Mudança de Referência das Coordenadas Fornecidas ao Sistema

As posições 0º e 360º azimute da antena foram definidas como sendo o seu fim de curso. Esta referência foi adotada por facilitar a lógica de controle, uma vez que não haverá passagem de 359º para 0º sem passar por 180º. No entanto, isto nos obriga a converter as efemérides fornecidas para o rastreio, que utilizam o norte geográfico como referência, para o sistema de referência adotado (0º e 360º no fim de curso da antena). A Figura 27 ilustra esse problema.

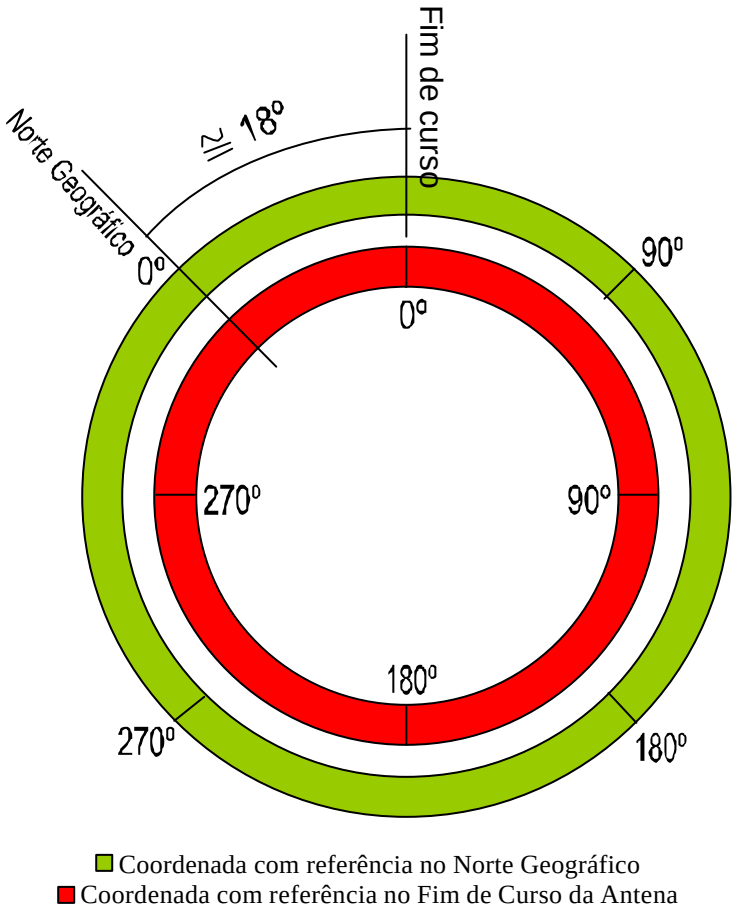

Figura 27: Relação entre as coordenadas com referência no norte geográfico e no fim de curso da antena

Para se obter a equação de conversão entre esses sistemas de referência, foi necessário obter a diferença angular entre os dois. O procedimento adotado foi:

1. Calcular a passagem do sol ao longo do dia;
2. Ajustar a posição da antena usando o programa de posicionamento para que a sombra do receptor coincida com centro da antena;
3. No instante que o passo 2 for alcançado, anotar o horário e as coordenadas fornecidas pelo programa;
4. Calcular a diferença entre a coordenada indicada pelo programa de posicionamento (coordenada com referência no fim de curso), com a coordenada do sol naquele instante (com referência geográfica).

Outra opção para esse experimento é gerar um sinal de rádio em ponto geográfico conhecido, e ajustar a posição da antena até que se obtenha uma potência máxima na recepção deste sinal. De posse das posições geográficas (latitude, longitude e altura) da antena e da fonte do sinal, que podem ser obtidas com um GPS, realiza-se uma triangularização para se obter o azimute e elevação real. Por último, calcula-se a diferença entre a coordenada indicada pelo programa de apontamento, e a coordenada com referência no norte geográfico. O valor obtido (*offset*) para a situação atual da EMMN foi de 18º.

$$az_{ant} = az_{geo} + 18 \tag{05}$$

Dessa forma, a cada interação no cálculo do sinal de controle, a coordenada geográfica do arquivo de efemérides (az_{geo}) é ajustada (equação 05), e posteriormente utilizada como referência no sistema. Para que seu valor ficasse no intervalo entre 0 e 360, foi utilizado o algoritmo da figura 28 na conversão de az_{geo} em az_{ant}.

```
se az_ant > 360
   az_ant = az_ant - 360
fim se
```

Figura 28: Algoritmo para ajuste do ângulo azimute.

O procedimento para obtenção do valor de *offset* foi adotado também para a elevação, a fim de realizar pequenas correções sobre os valores de fim de curso adotados para o sistema de coordenadas da antena.

12.2 Solução do problema de rastreio pelo fim de curso

Não raramente temos passagens de satélites pelo fim de curso azimute da antena. Para evitar uma interrupção no rastreio devido ao fim de curso, realiza-se uma conversão nas coordenadas da passagem para que se trabalhe no segundo quadrante da elevação (entre 90 e 180°), durante todo o rastreio, evitando assim o fim de curso.

Uma vez que a elevação possui 180 graus de liberdade, existem sempre duas posições em que a antena pode apontar para uma mesma efeméride. Por exemplo, uma efeméride 30,0° az e 45° ele equivale a 210° az e 135° ele, e 350º az e 5° ele pode ser atingido com 170° az e 175º ele. O programa que calcula as efemérides a serem apontadas pela antena (*software* no *windowns*), na realização do rastreio, sempre utiliza elevação entre 0° e 90°. Isto, no entanto, não nos impede de utilizar o segundo quadrante da elevação para atingir essa efeméride. A figura 29 apresenta o algoritmo de conversão das efemérides, para trabalhar-se com o segundo quadrante da elevação.

```
el = 180 – el
az = az + 180
se  az >= 360
     az = az – 360
fim_se
```

Figura 29: Algoritmo para conversão das efemérides no segundo quadrante da elevação.

Após carregar uma passagem no programa de rastreio (*CITO0v2*) e converter as coordenadas para as coordenadas da antena (ver item 12.1), verifica-se se haverá passagem pelo fim de curso. Caso a passagem pelo fim de curso seja confirmada, todas as efemérides são convertidas para que seja utilizado o segundo quadrante da elevação. Assim um rastreio que começaria em 340º az e 0º ele, e terminaria em 133º az 0º ele (obviamente passando pelo 0º az), seria convertido em sua equivalente no segundo quadrante da elevação. Com isso este rastreio começará em 160º az 180º ele, e terminará em 313º az 180º ele, evitando-se assim passar pelo fim de curso azimute.

O algoritmo para verificar a passagem pelo 0º é bem simples. Sabe-se que a região a ser varrida no eixo azimute, durante a passagem de um satélite, é menor ou igual a 180º. Portanto, a única forma do módulo da diferença entre as coordenadas AZ do início, e do final ser maior que 180º, é no caso de uma passagem pelo 0º. Assim temos o algoritmo da figura 30.

```
se abs( az_inicio - az_fim ) > 180
     Mudar as coordenadas para
     trabalhar com 2ºquadrante
     da elevação.
fim se
```

Figura 30: Algoritmo de identificação para uso das coordenadas no segundo quadrante.

12.3 Proteção de Fim de Curso

Existem três proteções para evitar que a antena seja danificada devido a um impacto num fim de curso:

- A proteção via software (módulo de controle) que impede que seja dado um comando para mover a antena para uma posição que ultrapassem um fim de curso;
- A proteção elétrica (módulo de comando), acionada por sensores de fim de curso, que através de contactores corta a alimentação dos dois motores que movimentam a antena;
- A proteção mecânica (módulo posicionador), implementada através de borrachas que visam amortecer o impacto.

A proteção elétrica realizada pelo módulo de comando (*status* dos relés KA1, KA2 e KA3) se comunica com o *software* de rastreio, através das entradas digitais da placa de aquisição (ver tabela 9). Desta forma o programa toma conhecimento do acionamento da proteção e utiliza uma saída gráfica para notificar o operador.

12.4 Seqüência do Software de Controle e Rastreio

O programa de controle e rastreio (*CITO0v2*) realiza os seguintes passos durante a sua execução:

1) Carrega de um arquivo texto, a tabela contendo a seqüência de efemérides e respectivos horários do rastreio a ser realizado. Nesta etapa executa-se a mudança da referência (ver item 12.1), e caso necessário, a conversão para se trabalhar no segundo quadrante da elevação (ver item 12.2).
2) Posiciona a antena no ponto de surgimento (primeira efeméride da tabela) quando restam 60 segundos para o início do rastreio.
3) Começa a realizar o laço de controle a partir do instante de início, até que o último elemento da tabela tenha sido alcançado.
4) Finaliza o programa.

Em paralelo a essa seqüência, o programa monitora se houve acionamento da proteção elétrica (*thread ICE*). Caso algo ocorra, a seqüência de rastreio pula para a etapa 4, e uma mensagem indicando que houve acionamento da proteção elétrica é impressa na tela. As funções relativas a mudança da referência e quadrante de operação foram desenvolvidas em C++, com definições e descrições presentes nos arquivos efem.h e efem.cpp, respectivamente. O Apêndice D apresenta a classe Efem criada para dar suporte as operações de mudança no sistema de coordenadas da antena.

13 Implementações futuras de *Software*

É necessário ainda a implementação de algumas atividades e tarefas adicionais, como por exemplo, o controle manual do posicionamento da antena através do teclado ou mouse. Uma interface gráfica em modo texto também será incorporada ao *software* com uso da biblioteca *Ncurses.*

A idéia final do *software* de rastreio para a estação EMMN é torná-la totalmente independente da ação humana (exceto em caso de manutenção e escolha das efemérides). Sua versão atual (0.2), já proporciona o acionamento do sistema de potência no horário previsto, bem como o seu desligamento ao término da tarefa. Porém, ainda são necessários alguns ajustes finos, que tornarão o *software* mais operacional, principalmente com a inclusão da interface gráfica.

14 Bibliografia

[1] Tocci, Ronald J., Widmer, Neal S., *Sistemas Digitais Princípios e Aplicações*, sétima edição, LTC, 2000.

[2] Dorf, Richard C. , Bishop , Robert H., *Sistemas de controle modernos*, oitava edição, LTC, 2001.

[3] Ogata, K., *Engenharia de controle moderno*, terceira edição, Prentice Hall, 1997.

[4] Nise, Norman S., *Engenharia de Sistemas de Controle*, terceira edição, LTC, 2002.

[5] Satry, S., *Adaptive control: stability, convergence and robustness*, Prentice Hall, 1989.

[6] Ioannou, Petros A., Sun, J., *Robust adaptive control*, 1996, Prentice Hall.

[7] Leroy Somer SMV UM Moteur autosynchrones – catalogue technique;

[8] Leroy Somer UMV 4301 – Variateur de vitesse pour moteur asynchrones avec et sans retour et pour moteurs autosynchrones – installation et maintenance;

[9] Leroy Somer UMV 4301 – Variateur de vitesse pour moteur asynchrones avec et sans retour et pour moteurs autosynchrones – Paramétrage et synoptiques;

[10] GESPAC - Dual Parallel Interface Modulo (2x16 I/O) : GESPIA 2A & 2AW;

[11] RDC-19220 & RD-19230 -Series Converters - Applications Manual MN-19220XX;

[12] SYNCHRO/RESOLVER CONVERSION HANDBOOK;

[13] S-band small satellite ground station description for SACI, 29/08/1997;

[14] Nomenclature de maintenance de la station SOL SACI;

[15] Notice technique positionneur SACI – SREM – 2639 – TN-0001, 10/09/1997;

[16] Notice technique abaisseur 2255.2 MHz/70 MHz, 16/07/1997;

[17] Amplificateur lineaire 2 GHz/50W, Réference: M20.40.70BR

A Interação com o usuário

Este apêndice trata da interação do sistema com o usuário para os seguintes eventos: fim de curso acionado, botão de emergência acionado e inversor ELEVACAO e/ou AZIMUTE com defeito ou desligado. Quando um deles ocorre, a mensagem da figura 31 é apresentada ao usuário. O único evento passível de solução corresponde ao fim de curso acionado. As demais situações não podem ser resolvidas dessa forma, pois necessitam de uma manutenção mais elaborada.

```
O sistema foi desligado (favor consultar arquivo de log).
Selecione uma das opções:
[1]-Continuar a monitoração
[2]-Tentar solucionar o problema
```

Figura 31: Mensagem apresentada ao usuário quando o sistema é desligado inesperadamente.

Ao invés do usuário selecionar a opção 2 (ver figura 31), este poderá continuar a monitoração, sendo exibido na tela as mesmas informações presentes na figura 24. Os sinais de controle são zerados e somente as informações relativas ao posicionamento da antena são atualizadas. Caso a segunda opção tenha sido escolhida e o fim de curso acionado, a tela da figura 27 será exibida. Caso contrário, uma mensagem solicitando ao usuário para consultar o arquivo de *log* do sistema será mostrada. Logo em seguida o procedimento para retirada da antena do fim de curso poderá ser executada (figura 32).

```
O sistema foi desligado por que um dos fins de cursos foi acionado.
Deseja iniciar o processo de retirada agora (s/n)?
```

Figura 32: Mensagem apresentada ao usuário para retirar a antena do fim de curso.

B Listagem dos programas auxiliares da versão 0.2

```
#include <stdio.h>   /* para o printf */
#include <comedilib.h>
#include <sys/time.h>
#include <math.h>
#include <stdlib.h>
//atualizado 20/01/2006
//Objetivos:
//Este programa tem por finalidade inicializar o sistema EMMN

int subdevA = 1; // Saida analogica
int subdevD = 2;//Saidas e entradas digitais DIO
int subdevPABC=3; //Saidas e entradas digitas PA, PB e PC
int chanL; //Canal de leitura da posicao PA0 a PA7 e PB0 a PB7
int chanA = 0; // Saida analogica DAC0OUT 20 AOGND 23
int chanE = 1; // Saida analogica DAC1OUT 21 A0GND 23
//canais DIO
int chanP=1; // Canal para ligar a gaveta de potencia e liberar o
freio DIO1 27
int range = 0;
int aref = AREF_GROUND;
void escritaELE(double volts);
void escritaAZ(double volts);
comedi_t *it; //ponteiro para o device da placa

main()
{
  int w;

  //abertura do device
  it=comedi_open("/dev/comedi0");

  //Configuracao dos canais de PC4(20) a PC7(23)
  for(w=20;w<24;w++)
  {
     chanL=w;
     comedi_dio_config(it,subdevPABC,chanL,COMEDI_OUTPUT);
     comedi_dio_write(it,subdevPABC,chanL,1);
  }
  //Configuracao dos DIO
  //DIO0 - LED BUTEES - Fim de curso acionado
  //DIO1 - Liga gaveta de potencia
  //DIO2 - LED POURSUITE - Rastreiando
  //DIO3 - LED ARRET D'URGENCE - Parada de emergencia
  //DIO4 - LED PUISSANCE - Gaveta de potencia ligada
  //DIO5 - LED DEFAUT - Defeito
  //DIO6 - AZIMUTE/Fio 47 - Informa se o inversor azimute esta ligado
  //DIO7 - ELEVACAO/Fio 53 - Informa se o inversor elevacao esta
ligado
  for(w=0;w<6;w++)
  {
     chanL=w;
     comedi_dio_config(it,subdevD,chanL,COMEDI_OUTPUT);
     comedi_dio_write(it,subdevD,chanL,1);
  }
  comedi_dio_write(it,subdevD,chanP,0);

  escritaELE(0);
  escritaAZ(0);
```

```
  return 0;
}


void escritaELE(double volts)
{
  lsampl_t data, maxdata;
  comedi_range *range1;

  maxdata=comedi_get_maxdata(it, subdevA, chanE);
  range1=comedi_get_range(it,subdevA,chanE,range);
  data=comedi_from_phys(volts, range1, maxdata);
  comedi_data_write(it,subdevA,chanE,range,aref,data);
}

void escritaAZ(double volts)
{
  lsampl_t data, maxdata;
  comedi_range *range1;

  maxdata=comedi_get_maxdata(it, subdevA, chanA);
  range1=comedi_get_range(it,subdevA,chanA,range);
  data=comedi_from_phys(volts, range1, maxdata);
  comedi_data_write(it,subdevA,chanA,range,aref,data);
}
```

Figura 33: Listagem do programa *startup*.

```
#include <stdio.h>   /* para o printf */
#include <comedilib.h>
//Este programa tem como finalidade desligar a gaveta de potencia
int subdev = 2;//Saidas/entradas digitais  DIO
int chanP = 1;//Canal DIO1 27
int chanPUIS=4;

int main(int argc,char *argv[])
{
  comedi_t *it;
  int bit;

  it=comedi_open("/dev/comedi0");

  //Escrita nos canais
  comedi_dio_config(it,subdev,chanP,COMEDI_OUTPUT);
  comedi_dio_write(it,subdev,chanP,0); //Manda nivel logico 0
  comedi_dio_write(it,subdev,chanPUIS,1);
}
```

Figura 34: Listagem do programa *desligar*.

```
#include <stdio.h>   /* para o printf */
#include <comedilib.h>
//Este programa tem o objetivo de ligar a gaveta de potencia
int subdev = 2;//Saidas/entradas digitais  DIO
int chanP = 1;//Canal DIO1 27

int main(int argc,char *argv[])
{
  comedi_t *it;
  int bit;

  it=comedi_open("/dev/comedi0");
```

```
  //Escrita nos canais
  comedi_dio_config(it,subdev,chanP,COMEDI_OUTPUT);
  comedi_dio_write(it,subdev,chanP,1); //Manda nivel logico 1
}
```

Figura 35: Listagem do programa *ligar*.

C Placa INPE-01

A placa INPE-01 foi confeccionada para servir de interface entre a placa SOTEREM 2266-1, o módulo de comando, e o módulo de controle (computador). Seu desenho foi desenvolvido com uso do *software Eagle*, versão 4.15 *light*, disponibilizado na *internet* pelo fabricante. O conector DB20 utilizado no desenho da placa (conexão superior), foi diferente do utilizado na montagem da placa (conexão lateral). Isso provocou alguns problemas, pois suas pinagens não são compatíveis entre si. A tabela 12 apresenta a pinagem teórica para este conector, e a sua versão final para o conector adotado. Pela tabela, observa-se que o pino 1 do conector ideal corresponde ao pino 20 do conector adotado, e vice e versa. Dessa forma, caso o conector ideal seja utilizado, as ligações com o módulo de controle (placa PCI 6025E) deverão ser refeitas[11]. Caso contrário, o *software* de rastreio não funcionará adequadamente.

Conector BD20 (Placa INPE-01)				**Conector S8 (Gaveta de *pilotage*)**		**Placa PCI 6025E (conector final)**	
Teórico (conector ideal)		**Adotado (conector final)**					
Pino	**Descrição**	**Pino**	**Descrição**	**Pino**	**fio**	**ID**	**Pino**
20	--	20	A00	23	Branco-Cinza	97	PA0
19	GND	19	A01	10	Branco	95	PA1
18	19B	18	A02	22	Amarelo	93	PA2
17	25A	17	A03	9	Vermelho-Cinza	91	PA3
16	A15	16	A04	21	Vermelho	89	PA4
15	A14	15	A05	8	Verde	87	PA5
14	A13	14	A06	20	Preto	85	PA6
13	A12	13	A07	7	Amarelo-Cinza	83	PA7
12	A11	12	A08	19	Marrom	81	PB0
11	A10	11	A09	6	Marrom-Branco	79	PB1
10	A09	10	A10	18	Roxo	77	PB2
9	A08	9	A11	5	Roxo-Branco	75	PB3
8	A07	8	A12	17	Azul	73	PB4
7	A06	7	A13	4	Azul-Cinza	71	PB5
6	A05	6	A14	16	Laranja	69	PB6
5	A04	5	A15	3	Laranja-Cinza	67	PB7
4	A03	4	25A	15	Rosa	55	PC5
3	A02	3	19B	2	Verde-Cinza	57	PC4
2	A01	2	GND	14	Amarelo por fora	94	GND
1	A00	1	--	1	--	--	--

Tabela 12: Pinagem do conector DB20 para suas duas versões: conexão superior (teórica) e conexão lateral (prática).

As figuras 36 e 37 apresentam respectivamente, o desenho da placa (*layout*) e o esquemático, ambos gerados através do *Eagle*.

[11] Com uso do conector ideal, sem a mudança das ligações com a placa PCI 6025E, o bit menos significativo A00 não estará ligado com a placa AD/DA. Já o pino A15, por exemplo, estará ligado ao pino PA4.

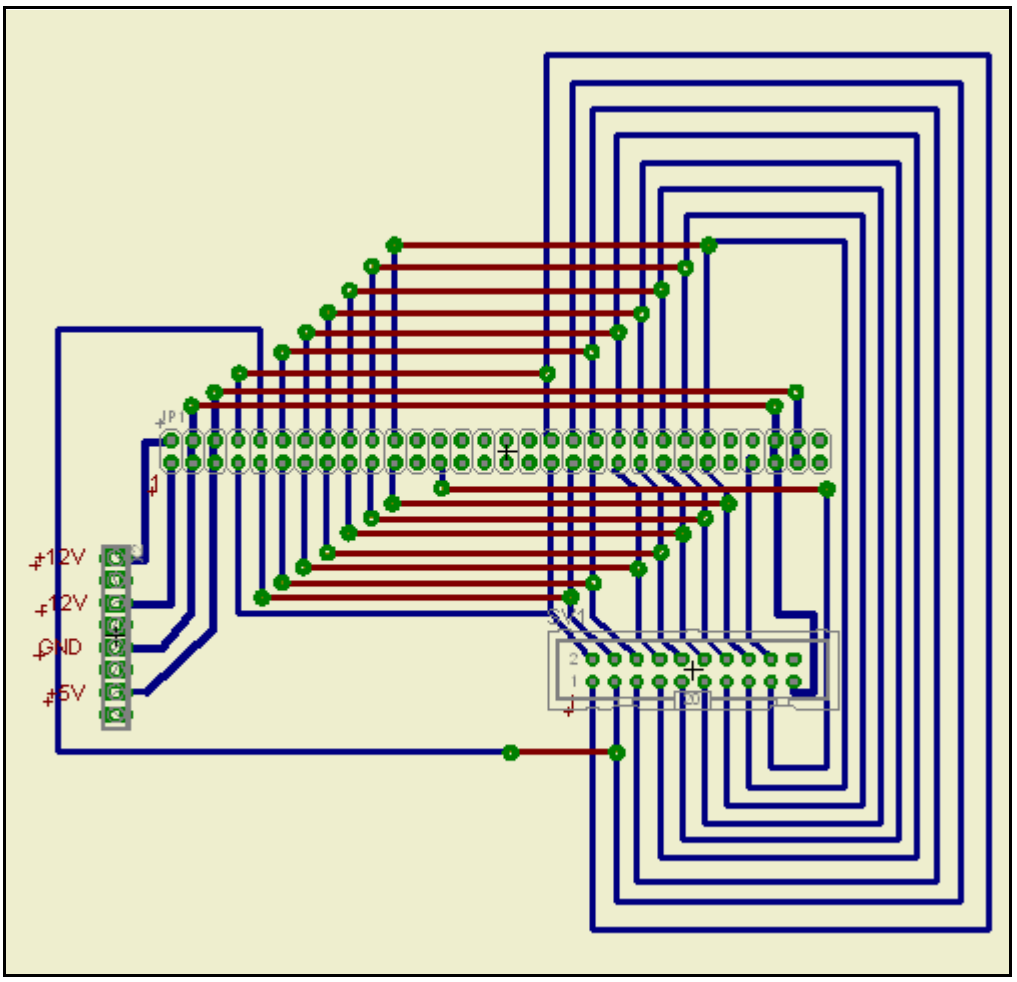

Figura 36: *Layout* da placa INPE-01 desenvolvido com uso do *software Eagle*.

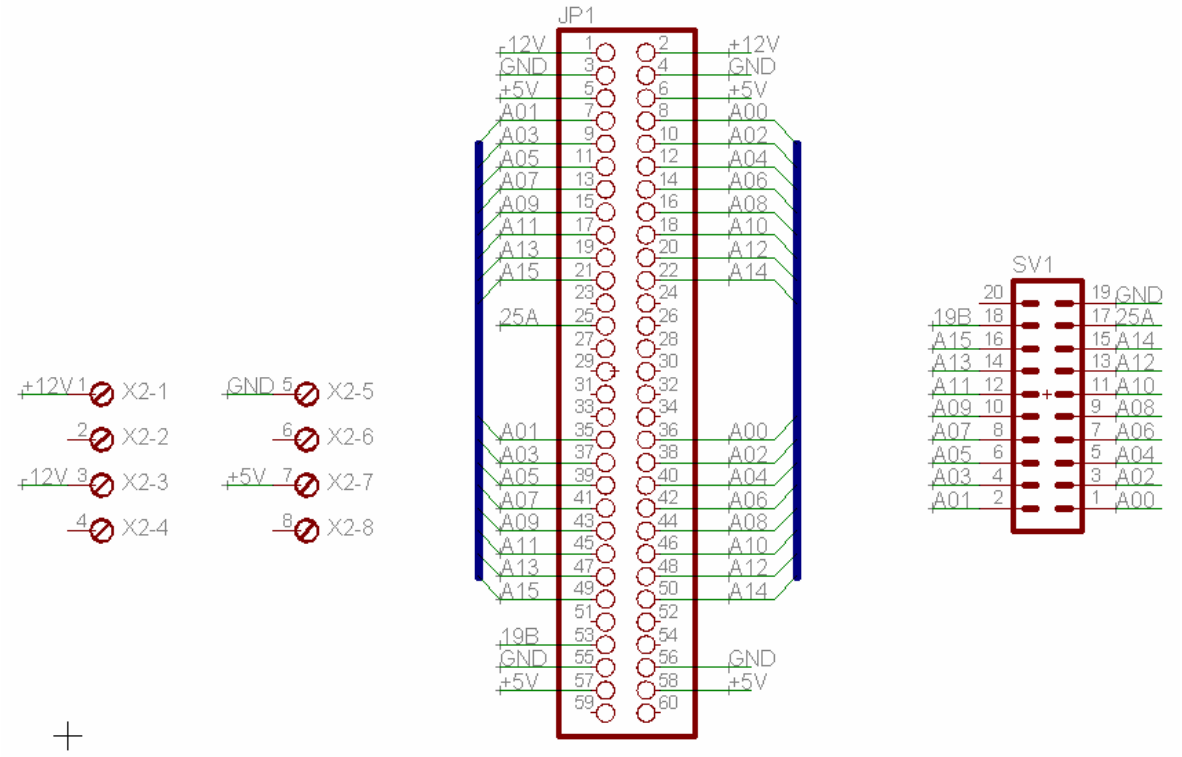

Figura 37: Esquemático da placa INPE-01 desenvolvido com uso do *software Eagle*.

D Classe Efem

Esta classe fornece os métodos para extrair as efemérides e respectivos horário, referentes à uma passagem de um arquivo texto. O arquivo deve respeitar o formato descrito no anexo E.

D.1 Funções Membro Públicas:

Efem ();
Construtor. Inicia os dados membro da função com valores consistentes. Isso garante que caso seja executado o comando Efem::get() antes de qualquer arquivo ter sido carregado pela função Efem::load_f(), os valores retornados estejam dentro do padrão.

int load_f(const char* nome_arq);
Carrega um arquivo com a tabela das efemérides.
Retorna zero em caso de sucesso e o código do erro em caso de falha.
O arquivo deve estar no formato correto (exemplo no Anexo E).
Esta função translada a referência das coordenadas do arquivo do norte geográfico para fim de curso da antena (sistema adotado pela estação), e realiza a conversão para que se trabalhe no segundo quadrante da elevação caso necessário.

void get(time_t *ef_time, float *azi, float *ele) const;
Passa o horário, azimute e a elevação da posição atual da tabela das efemérides. O horário é dado no formato time_t.

int next();
Avança uma linha na tabela e retorna 0.
Retorna 1 se não for possível por ultrapassar o fim da tabela;

E Formato do arquivo de efemérides

13 07 2005 12:09:59.635	22.912	0.000	3232.078289
13 07 2005 12:10:09.000	23.168	0.563	3170.275440
13 07 2005 12:10:19.000	23.451	1.177	3104.345221
13 07 2005 12:10:29.000	23.746	1.803	3038.487776
13 07 2005 12:10:39.000	24.053	2.444	2972.715745
13 07 2005 12:10:49.000	24.375	3.100	2907.042828
13 07 2005 12:10:59.000	24.711	3.771	2841.483798
13 07 2005 12:11:09.000	25.062	4.460	2776.054615
13 07 2005 12:11:19.000	25.431	5.166	2710.772573
13 07 2005 12:11:29.000	25.818	5.892	2645.656444
13 07 2005 12:11:39.000	26.226	6.638	2580.726658
13 07 2005 12:11:49.000	26.655	7.407	2516.005502
13 07 2005 12:11:59.000	27.108	8.199	2451.517343
13 07 2005 12:12:09.000	27.587	9.017	2387.288890
13 07 2005 12:12:19.000	28.094	9.861	2323.349491
13 07 2005 12:12:29.000	28.631	10.735	2259.731469
13 07 2005 12:12:39.000	29.203	11.640	2196.470506
13 07 2005 12:12:49.000	29.812	12.578	2133.606092
13 07 2005 12:12:59.000	30.463	13.553	2071.182028
13 07 2005 12:13:09.000	31.159	14.567	2009.247009
13 07 2005 12:13:19.000	31.906	15.622	1947.855294
13 07 2005 12:13:29.000	32.709	16.722	1887.067468
13 07 2005 12:13:39.000	33.575	17.870	1826.951305
13 07 2005 12:13:49.000	34.512	19.069	1767.582770
13 07 2005 12:13:59.000	35.527	20.323	1709.047130
13 07 2005 12:14:09.000	36.633	21.636	1651.440220
13 07 2005 12:14:19.000	37.839	23.011	1594.869851
13 07 2005 12:14:29.000	39.160	24.451	1539.457355
13 07 2005 12:14:39.000	40.613	25.959	1485.339254
13 07 2005 12:14:49.000	42.214	27.538	1432.669029
13 07 2005 12:14:59.000	43.988	29.188	1381.618899
13 07 2005 12:15:09.000	45.959	30.908	1332.381535
13 07 2005 12:15:19.000	48.157	32.696	1285.171532
13 07 2005 12:15:29.000	50.616	34.544	1240.226419
13 07 2005 12:15:39.000	53.378	36.442	1197.806899
13 07 2005 12:15:49.000	56.484	38.372	1158.195925
13 07 2005 12:15:59.000	59.985	40.309	1121.696154
13 07 2005 12:16:09.000	63.928	42.218	1088.625272
13 07 2005 12:16:19.000	68.360	44.054	1059.308760
13 07 2005 12:16:29.000	73.315	45.761	1034.069816
13 07 2005 12:16:39.000	78.803	47.275	1013.216539
13 07 2005 12:16:49.000	84.802	48.525	997.026967
13 07 2005 12:16:59.000	91.238	49.442	985.733172
13 07 2005 12:17:09.000	97.985	49.969	979.506195
13 07 2005 12:17:19.000	104.871	50.070	978.443870
13 07 2005 12:17:29.000	111.703	49.738	982.563442
13 07 2005 12:17:39.000	118.296	48.998	991.800247
13 07 2005 12:17:49.000	124.501	47.897	1006.012667
13 07 2005 12:17:59.000	130.221	46.500	1024.992518
13 07 2005 12:18:09.000	135.412	44.879	1048.479170
13 07 2005 12:18:19.000	140.071	43.100	1076.175338
13 07 2005 12:18:29.000	144.224	41.226	1107.762614
13 07 2005 12:18:39.000	147.912	39.304	1142.915282
13 07 2005 12:18:49.000	151.184	37.375	1181.311562
13 07 2005 12:18:59.000	154.088	35.468	1222.642014
13 07 2005 12:19:09.000	156.671	33.602	1266.615225
13 07 2005 12:19:19.000	158.975	31.792	1312.961190
13 07 2005 12:19:29.000	161.036	30.047	1361.432841