

Escola Politécnica da Universidade de São Paulo

Departamento de Engenharia de Energia e Automação Elétricas



Automação de um aerogerador simulado

Pedro Ivo Anastácio Pellegrini

Vinícius Fernandes Lourenzen de Oliveira

Projeto de Formatura/2014

Pedro Ivo Anastácio Pellegrini
Vinícius Fernandes Lourenzen de Oliveira

Automação de um aerogerador simulado

Área de Concentração: Engenharia
Elétrica - Ênfase em Energia e
Automação Elétricas

Orientador: Prof. Dr. Eduardo Lorenzetti
Pellini

São Paulo
2014

Escola Politécnica da Universidade de São Paulo

Departamento de Engenharia de Energia e Automação Elétricas



Automação de um aerogerador simulado

Alunos: Pedro Ivo Anastácio Pellegrini
Vinícius Fernandes Lourenzen
de Oliveira

✓ Orientador: Eduardo Lorenzetti Pellini

✓ Coordenador: Giovanni Manassero

Agradecimentos

Somos gratos ao Professor Eduardo Lorenzetti Pellini por todo incentivo e apoio prestado, além da confiança depositada nos orientados quanto à disponibilização da estrutura do laboratório. Também agradecemos ao Engenheiro do Laboratório, Leonardo Pereira, o qual sempre se disponibilizou para nos auxiliar, desde o início deste projeto.

Resumo

Este trabalho apresenta a automação de um aerogerador com máquina de indução duplamente alimentada de 2 MW embarcada em um CompactRIO da National Instruments. Uma vez que um aerogerador de tal magnitude não é acessível na universidade, o processo de automação se baseou em um modelo já existente, simulado em tempo real através do Real Time Digital Simulator (RTDS). Os objetivos principais consistem em realizar o sequenciamento de partida, parada e parada de emergência do aerogerador e avaliação da capacidade do sistema da National Instruments em operar o sistema, já que todo esse comando será realizado pelo CompactRIO. Para realização dessas tarefas, foi necessário implementar controles de apontamento da nacela e ângulo de ataque das pás. O CompactRIO utilizado é o 9024, acoplado ao chassis 9111. O CompactRIO conseguiu, de forma geral, atender às expectativas do controle, realizando um ciclo completo de processamento em 1 ms em qualquer uma das etapas de sequenciamento. Contudo, ainda existem pontos-chaves em que alguns ciclos excedem o tempo esperado. Devido às constantes de tempo mecânicas serem muito maiores que 1 ms, isso é imperceptível para o aerogerador.

Palavras-chave: Automação, Aerogerador, Processamento em tempo real, CompactRIO.

Abstract

This project presents an automation strategy embedded in a National Instruments CompactRIO system for a 2MW doubly fed induction generator wind turbine. In general, a powerful wind turbine is not available on universities for studies. Therefore, the automation was based on an existing model from the Real Time Digital Simulator (RTDS). The main objectives are: create the start-up, shutdown and emergency stop process of the wind turbine, it also includes the evaluation of the CompactRIO acting like a wind turbine controller. The development of this process requested pitch and yaw control algorithms, thus they were also developed. The CompactRIO is the 9024 model embedded in a 9111 chassis. In general, the National Instruments Real Time system was able to achieve a 1 ms scan time in every states of the wind turbine operation. Nonetheless, some specifics scans cycles were lost because they exceeded the 1 ms threshold. Even so, this lack of calculation is imperceptible by the wind turbine due to mechanical time constants are too large when compared to electrical ones.

Keywords: Wind turbine automation, Wind turbine, Real time systems, CompactRIO.

Sumário

1. INTRODUÇÃO	10
2. OBJETIVOS	12
2.1 OBJETIVOS SECUNDÁRIOS	12
3. METODOLOGIA	13
4. CONCEITOS BÁSICOS DE AEROGERADORES	16
4.1 TORRE	16
4.2 TURBINA	16
4.3 CAIXA DE VELOCIDADES	16
4.4 GERADOR	17
4.5 MODELO AERODINÂMICO	18
5. DESCRIÇÃO DO SISTEMA DE CONTROLE DO AEROGERADOR	20
5.1 CONTROLE DE ÂNGULO DAS PÁS	21
5.2 CONTROLE DE DIRECIONAMENTO DA NACELE	22
5.2.1 <i>Descrição Física do Sistema de Direcionamento</i>	22
5.2.2 <i>Bloco de Controle do Sistema de direção da Nacele</i>	23
6. CONTROLADOR LÓGICO PROGRAMÁVEL	25
6.1 ASPECTOS FÍSICOS	25
6.2 ARQUITETURA	28
7. DESCRIÇÃO DO MODELO DE AEROGERADOR DO RTDS	30
7.1 REDE DA SIMULAÇÃO	30
7.2 BLOCO WIND TURBINE & METERS	33
7.3 BLOCO START UP TIMER & CROWBAR CONTROL	39
7.4 INTERFACE RUNTIME	40
8. INTRODUÇÃO AO LABVIEW	41
8.1 ARQUITETURA DE PROGRAMAÇÃO	43
8.1.1 <i>Variáveis</i>	43
8.1.2 <i>Laços, Testes condicionais, Shift Registers, Enums, Clusters</i>	44
8.1.3 <i>Blocos de programação</i>	46
9. INTEGRAÇÃO DO RTDS COM O CRIO9024	48
9.1 CONTROLE DE APONTAMENTO DA NACELE NO CRIO	48
9.2 CONTROLE DE ÂNGULO DAS PÁS	51
9.2.1 <i>Modificação do sistema de pitch no RTDS</i>	52
9.2.2 <i>Implementação do controle de Pitch no cRIO</i>	53
9.3 MODIFICAÇÃO DO MODELO DO RTDS	57
10. PROGRAMAÇÃO DO CRIO	62
10.1 PROGRAMAÇÃO DA MÁQUINA DE ESTADOS	62
10.2 PROGRAMAÇÃO DO OSCILÓGRAFO	68
10.3 INTERFACE COM O USUÁRIO	71
10.4 CONEXÕES ELÉTRICAS	72
11. CONCLUSÃO	75

12. BIBLIOGRAFIA	77
------------------------	----

1. Introdução

Segundo dados da EPE (Empresa de Pesquisa Energética), espera-se que a demanda de energia elétrica no Brasil cresça de um patamar de aproximadamente 458 TWh/ano em 2013 para um patamar de 687 TWh/ano em 2023[1]. Observa-se um surpreendente aumento de 50% no consumo de energia nos próximos 10 anos. Além disso, a capacidade de geração elétrica instalada no país é de 121GW [2], ou seja, se o sistema nacional funcionasse no limite de sua capacidade, produziria aproximadamente 1060 TWh/ano, de forma que, mantendo a taxa atual do crescimento do consumo de energia, em menos de 30 anos a capacidade atual de geração seria superada. Levando em conta o fato de que o sistema não pode operar constantemente gerando sua máxima potência, observa-se a importância de aumentar a potência instalada para suprir essa crescente demanda por energia elétrica.

Outro grande interesse da expansão do sistema é tornar a matriz energética brasileira mais renovável e menos danosa ao ambiente, gerando assim créditos carbono e conservando recursos não renováveis para outras aplicações. Para tanto, é necessário buscar uma fonte com grande capacidade de geração e custos competitivos. Nesse cenário, a energia eólica surge como uma excelente alternativa na geração de energia elétrica, convertendo a energia dos ventos, recurso absolutamente renovável, em energia elétrica. Para se ter uma ideia do potencial de geração dessa fonte, segundo a ANEEL – Agência Nacional de Energia Elétrica, a capacidade de geração no mundo de energia eólica é de 53000TWh, o que representa quatro vezes o consumo de energia mundial [3], e com o desenvolvimento dessa tecnologia, os custos da produção dessa energia estão cada vez menores.

No entanto, do ponto de vista técnico, não é simples converter a energia dos ventos em energia elétrica. Um dos grandes problemas desse tipo de geração é conciliar a geração com velocidade variável com a frequência da rede, potência gerada, potência reativa gerada e tensão. Para tanto, vários tipos de máquinas podem ser utilizadas com diversos tipos de controle, entre elas o DFIG – Doubly Fed Induction Generator. Segundo [4], o DFIG apresenta a melhor relação de custo por MWh produzido, sendo essa uma das razões tipo da escolha desse gerador como objeto de estudo do trabalho. Sabendo que a máquina de indução é própria para se gerar energia na frequência da rede, é necessário fazer o controle da potência ativa e reativa que será fornecida.

O controle da potência ativa e da velocidade, a grosso modo, é feito através do controle de duas variáveis: o ângulo de ataque das pás do gerador e o ângulo de orientação do eixo da turbina em relação à direção do vento. Já o controle de tensão e potência reativa fornecida é feito pelo conversor *back to back*, também denominado AC/DC/AC¹. Outra incumbência do conversor é definir a frequência de alimentação do rotor para obter uma amplitude de variação da velocidade em torno de 30% da nominal [5], definindo se o fluxo de potência ocorrerá do gerador à rede ou da rede ao gerador. O foco desse trabalho é o controle do ângulo das pás e da orientação do eixo, além da automação completa do sistema, como um acréscimo à proposta inicial, estudar o controle de um conversor. Serão feitos estudos da partida do gerador, parada, parada de emergência, curto-circuito na rede e rejeição de carga. Dessa forma, por meio de simulações, busca-se avaliar a viabilidade técnica da geração de energia com esse tipo de equipamento.

Outro objetivo importante do projeto é testar a integração do controlador lógico programável, com um simulador digital em tempo real (RTDS²), testando, por meio de simulações, as respostas de um sistema real de controle e automação de um gerador eólico de energia. Para tanto, a parte mecânica e de potência do gerador serão simuladas no RTDS e os sistemas de automação e controles propostos estarão no controlador, permitindo avaliar o funcionamento do conjunto.

O grande diferencial da proposta é abrir um precedente de estudo e montagem de um sistema de controle e automação de uma turbina de geração eólica para o departamento de Energia e Automação da Engenharia Elétrica da Escola para que futuramente mais estudos sobre o tema sejam abordados a partir de um ponto mais avançado.

¹ Sigla para *Alternate Current/Direct Current/Alternate Current*. Representando a ordem de conversão das formas de onda que o conversor eletrônico realiza.

² Sigla para *Real Time Digital Simulator*

2. Objetivos

Um primeiro objetivo desse trabalho é abordar o cenário de automação de um gerador eólico moderno. Entre muitos geradores possíveis, é abordado o uso do gerador de indução duplamente alimentado (DFIG), munido de conversor eletrônico de topologia *back-to-back*. Também estuda-se o controle do ângulo de ataque das pás e controle da orientação da turbina eólica.

Outro propósito do projeto é a implementação da lógica de controle, sequenciamento e supervisão desse tipo de gerador em um controlador lógico programável real. Para tanto, também será desenvolvida uma tela interface de operação e controle da planta de usuário (SCADA).

Usar o controlador em um ambiente computacional de simulação que represente a turbina e seus acessórios em funcionamento em tempo real por meio de um simulador de tempo real, permitindo operar e monitorar o sistema como em um gerador real é uma das metas do projeto.

Por fim o teste e comissionamento do sistema para análise dos resultados e comparação com simulações esperadas tanto em condições normais de funcionamento, como partida, parada e regime, quanto em situações de emergência, como por exemplo: disparo da turbina, curto-circuito na rede e rejeição de carga é outra meta.

2.1 Objetivos Secundários

Pode-se citar a interdisciplinaridade como um dos escopos deste trabalho. São abordados, por exemplo, os seguintes temas: computação, comunicação, controle, automação, supervisão, máquinas elétricas, eletrônica de potência e sistemas de potência, abrangendo, portanto, boa parte dos campos de estudo da engenharia elétrica.

Contribuir com o departamento desenvolvendo uma ferramenta didática na área de geração eólica, envolvendo suas tecnologias relacionadas, é outro mérito que podemos destacar no projeto. Dessa forma, o projeto pode servir como base para trabalhos futuros.

Por fim, é importante destacar que o trabalho permite aplicar métodos de engenharia no desenvolvimento de projetos. Integrar diferentes equipamentos, avaliar simulações computacionais, desenvolver os diversos sistemas para possibilitar o funcionamento de um gerador, por exemplo, são atribuições de um engenheiro elétrico. Portanto, esse projeto permite avaliar a capacidade de exercício da engenharia, um dos propósitos do projeto de formatura.

3. Metodologia

Uma vez que o projeto trata do funcionamento de um aerogerador de grandes proporções, não há um protótipo disponível para uso didático. Sabendo dessa limitação, a automação, como já dito anteriormente, será realizada a partir da simulação do aerogerador.

Feitas tais considerações, o projeto começa com a pesquisa bibliográfica das partes integrantes do sistema de automação, denominada **atividade a**, aprofundando o conhecimento do assunto e fornecendo fundamentos para o desenvolvimento do projeto. A Figura 1 apresenta os principais componentes a serem estudados no aerogerador.

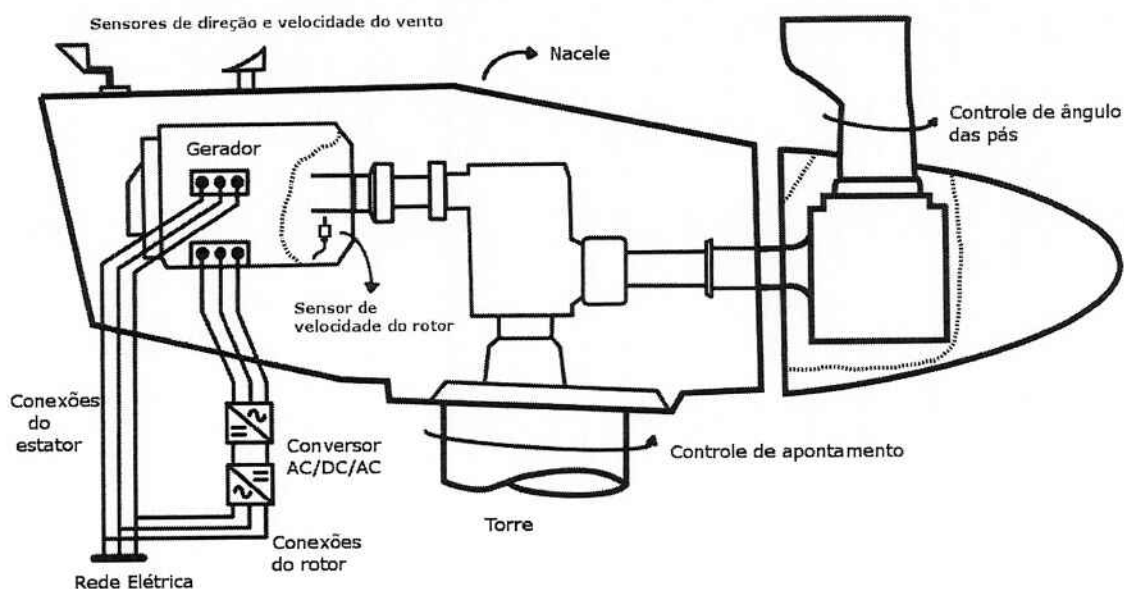


Figura 1: Aerogerador em corte

Na Figura 1, identifica-se, basicamente, a torre de sustentação, o rotor das pás e a nacelle, abrigo dos componentes do aerogerador. Acima da nacelle, estão os sensores de velocidade e direção do vento, ambos auxiliam o controle de ângulo das pás (*pitch angle control*) e o controle de apontamento (*yaw control*). Pela velocidade do vento sabe-se se a máquina deve entrar em operação ou sair de operação, condição de vento excessivo ou insuficiente. O controle de apontamento direciona o aerogerador para receber o máximo fluxo de ar; já o controle de ângulo altera o ângulo de ataque das pás com o vento incidente, determinando a quantidade de energia do vento que será convertida em torque mecânico. Dessa maneira, obtém-se torque máximo ou torque zero para mesma direção do aerogerador em relação ao fluxo de ar. Outro componente importante é o conversor eletrônico, responsável por controlar a amplitude e fase da tensão rotórica, ajustando o ponto de operação da máquina a partir da alimentação da rede elétrica. Controlar o conversor exige a leitura de

todas as tensões e correntes do mesmo, estabilização da corrente de entrada e manutenção da tensão DC em níveis aceitáveis.

A automação será previamente feita no RSCAD para depois ser repassada ao controlador externo. Será utilizado um controlador da *National Instruments*, CompactRIO-9024, também disponível no laboratório. Já que o equipamento é da *National Instruments*, a linguagem de programação é o LabVIEW, assim sendo, todo o controle implementado no RSCAD deve ser repassado para o LabVIEW. Esse software é fornecido pela Escola através de uma licença online. Aproveitando a plataforma de programação do LabVIEW será feito um sistema SCADA (sistema supervisor e de controle). Determina-se, para continuidade do projeto, a **atividade b**: integração e teste do controlador cRIO9024 com RTDS. Concluindo a integração do controlador ao RTDS, inicia-se o processo de migração da automação presente no RTDS ao cRIO. Para tanto, haverá uma etapa de definição dos blocos de automação que devem ser migrados, bem como as respectivas entradas e saídas, e as características dessas entradas e saídas. Essa atividade será determinada como **atividade c**.

Em seguida, será executada a etapa de programação do controlador e substituição da automação no RTDS pela automação externa, no cRIO. Sendo necessário testá-las para certificar-se do correto funcionamento.. Simultaneamente, o desenvolvimento do sistema supervisor ocorrerá, sendo que será feito na própria interface dos instrumentos virtuais do LabVIEW. Essa etapa será chamada de **atividade d**.

Finalizada a transferência da automação para o controlador será necessária uma etapa de comparação e avaliação do desempenho e possibilidade de automação do aerogerador com o controlador cRIO, bem como as diferenças desse controlador para um CLP convencional (isto é, que segue a **IEC 61131**). Chamaremos essa etapa de **atividade e**.

Após a comparação, a última etapa do projeto será a avaliação da necessidade e impacto de cada bloco de automação no funcionamento do aerogerador, fazendo estudos de diversos regimes de ventos e as diversas situações que podem ocorrer, colocando em risco os equipamentos ou mesmo prejudicando o desempenho em termos de geração. Essa etapa será denominada **atividade f**.

Por fim, a **atividade g**, que ocorrerá ao longo do projeto, paralelamente às outras atividades, será a documentação do projeto e seus resultados.



Figura 2: Exemplo de SCADA de um gerador eólico. Fonte: [9]

4. Conceitos básicos de aerogeradores

Os primeiros geradores eólicos de velocidade fixa com controle de *pitch* foram concebidos por meados da década de 1970 e, quanto ao tipo de gerador elétrico, utilizavam a máquina de indução com rotor gaiola de esquilo. Quando uma máquina desse tipo é conectada à rede elétrica de frequência fixa, ela desenvolve rotações quase à velocidade constante independente do fluxo de ar incidente, logo, eram aerogeradores de velocidade fixa e dependiam do controle de *pitch* para regular o quanto de energia extraíam. Conforme a evolução da tecnologia, houve aumento do tamanho dos aerogeradores, pois o vento é um recurso de baixa densidade energética, de forma que, é preciso grandes áreas para extrair grandes quantidades de energia. Tomando como exemplo a maior turbina da fabricante Vestas, com 164 metros de diâmetro das pás e 8 MW de potência, vê-se que ela gera pouco mais de 1% da potência de uma turbina da usina de Itaipu, a qual desenvolve 750 MW de potência.

Assim sendo, para aerogeradores de tais proporções atingiu-se um design representado pela Figura 3, a qual demonstra componentes básicos de um aerogerador moderno.

4.1 Torre

Atualmente, constroem-se torres com variadas alturas, desde 40 até 80 metros por exemplo. Como mostra a Figura 3, as torres abrigam os transformadores, que elevam a tensão para conectar com a rede elétrica, as escadas, para acesso à nacelle, e os cabos de energia. Nas torres mais largas já são instalados elevadores para auxiliar a subida. O cabeamento fica preso à torre, no entanto, nos últimos 10 metros, o cabo fica solto, a fim de permitir a torção do mesmo, consequência do giro de apontamento da turbina.

4.2 Turbina

Quando em rotação, as pás das turbinas cobrem uma grande área, as mesmas podem medir desde 20 metros, aerogeradores de 600 kW, até 80 metros, aerogeradores de 8 MW de potência. Devido ao tamanho do conjunto, a velocidade do rotor é baixa para grandes turbinas, aproximadamente 0,5 Hz. Turbinas de pás fixas podem apresentar design que as levam a uma condição de *stall* assim que a velocidade do vento ultrapassa o limite estabelecido, por volta de 25 m/s, por isso, portadoras do controle passivo de *stall*. Detalhes do controle ativo do ângulo da turbina serão abordados adiante. O rotor das pás é dotado de freio para mantê-lo parado durante manutenções.

4.3 Caixa de Velocidades

Esse elemento multiplica a velocidade do rotor das pás para atingir velocidades suficientes para o rotor do gerador. As caixas de velocidades mais comuns tem até 3 reduções. Mas, em alguns aerogeradores, não há aumento

da velocidade, o acoplamento é direto, requisitando geradores com maior número de polos e, conseqüentemente, um conjunto final mais pesado.

4.4 Gerador

Os geradores mais difundidos são máquina de indução duplamente alimentada (DFIG), máquina de indução com rotor gaiola de esquilo, gerador síncrono com excitação elétrica e gerador síncrono excitado por ímãs permanentes. Dependendo das condições, o gerador pode necessitar de refrigeração forçada. Os geradores, a não ser no caso do DFIG, têm os estatores conectados à rede pelo conversor eletrônico. Os conversores são de 690V nas potências mais baixas e de 3300V nas mais altas.

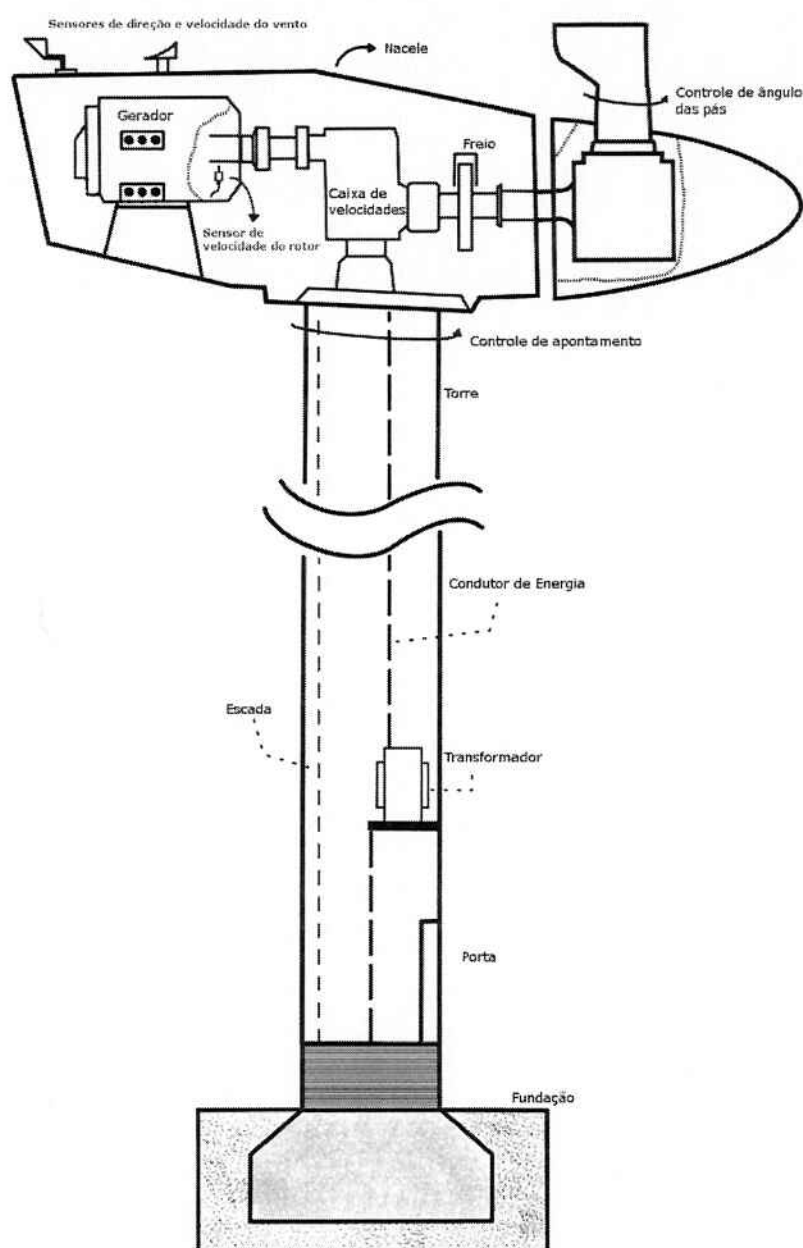


Figura 3: Componentes de um aerogerador

4.5 Modelo Aerodinâmico

A potência (P_v) do fluxo de ar de velocidade V_v em uma superfície de área A_1 é definida da seguinte maneira:

$$P_v = \frac{1}{2} \rho A_1 V_v^3,$$

na qual ρ é a densidade do ar.

Sabendo que o aerogerador não é ideal, ele converte uma fração de P_v para a turbina. Assim, a potência da turbina (P_t) é expressa por:

$$P_t = \frac{1}{2} \rho \pi R^2 V_v^3 C_p,$$

na qual R é o raio da turbina e C_p é o coeficiente aerodinâmico de potência.

O C_p representa a eficiência da turbina em converter energia do vento em energia mecânica, é um termo adimensional e é função da velocidade do vento, da velocidade de rotação da turbina e do ângulo de *pitch*.

Pelo Teorema de Betz, o máximo C_p que pode ser atingido, portanto, a máxima eficiência da turbina, é de 0,593. Esse coeficiente é não linear e descrito em função de λ (velocidade de ponta de pá) e ângulo de *pitch* (β), como pode-se observar abaixo:

$$C_p(\lambda, \beta) = k_1 \left(\frac{k_2}{\lambda_i} - k_3 \beta - k_4 \beta^{k_5} - k_6 \right) e^{\frac{k_7}{\lambda_i}},$$

$$\lambda_i = \frac{1}{\lambda + k_8} \text{ e } \lambda = \frac{R \omega_t}{V_v},$$

na qual, ω_t é a velocidade de rotação do rotor das pás em rad/s e os coeficientes k_1 a k_8 são fatores constantes em cada turbina.

Conclui-se que para realizar o rastreamento de máxima eficiência do sistema, deve-se utilizar técnicas de controle não linear para percorrer os pontos de máximo C_p para cada velocidade de ponta de pá.

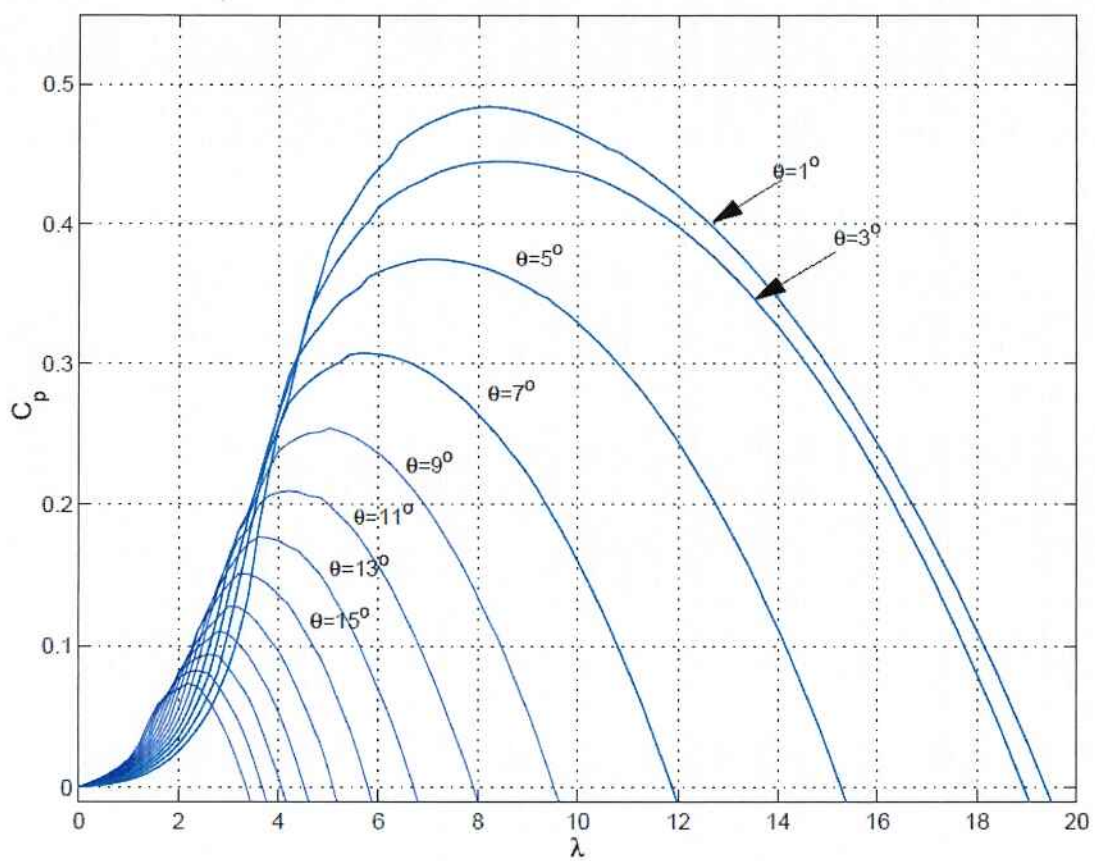


Figura 4: Gráfico de C_p em função de λ . Retirado de [10]

5. Descrição do sistema de controle do aerogerador

Simplificadamente, pode-se afirmar que o objetivo do controle de um aerogerador é extrair a máxima potência do vento ao mesmo tempo em que gera energia elétrica na frequência da rede e não sobrecarrega o equipamento. Uma visão geral do controle pode ser obtida a partir do diagrama de blocos do modelo de simulação do aerogerador.

De acordo com [10], um diagrama de blocos similar a muitos geradores eólicos é apresentado na figura 5.

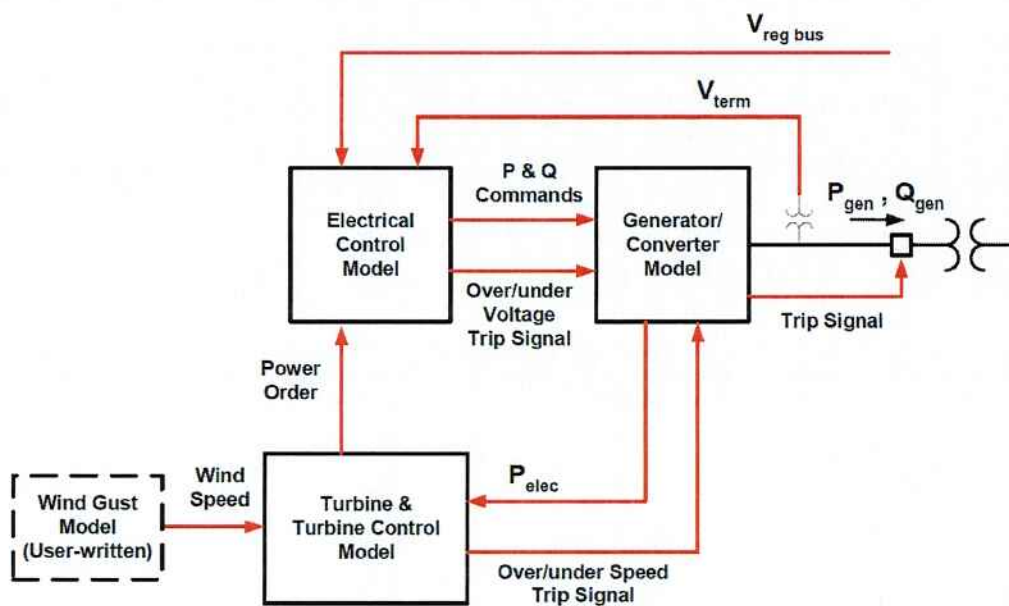


Figura 5: Diagrama de Blocos de um aerogerador, retirado de [10]

Através da Figura 5, tem-se o controle dividido em 2 áreas: controle da parte mecânica do aerogerador e controle elétrico.

O controle elétrico analisa diversos sinais e condições para tomada de decisão. Para determinar o quanto de potência ativa e reativa deve ser injetada na rede, o controle elétrico leva em conta qual a potência mecânica que está sendo entregue pela turbina. Contudo, outra maneira, mais comum para parques eólicos, é através do sistema supervisor do parque, o qual monitora a tensão na barra de interconexão dos aerogeradores e gera tal informação para os geradores (Representado na figura 5 como $V_{reg\ bus}$). A leitura da potência reativa do gerador também auxilia a manutenção da tensão de saída, V_{term} . Cita-se ainda o controle elétrico da tensão de saída. O mesmo desconecta o gerador da rede se a tensão permanecer abaixo ou acima da tensão nominal (1 pu) por um especificado período de tempo.

O controle mecânico baseia-se nos controles de apontamento e ângulo das pás, não permitindo que o rotor ultrapasse a velocidade nominal. No momento, é esse ponto que será aprofundado.

5.1 Controle de ângulo das pás

Denominado, em inglês, como *pitch control*, o mesmo é um dos dois mecanismos de atuação mecânica no aerogerador. A Figura 6 demonstra que o conceito desse controle é simples, consiste na alteração do ângulo das pás com o vento incidente. Quanto maior o vento incidente, maior a energia transmitida do vento para as pás.

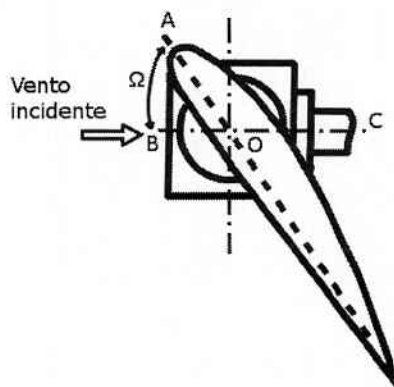


Figura 6: Exemplo do ângulo de ataque das pás

Pela figura 6, identifica-se o ângulo de ataque das pás como AÔB. Quanto ao ângulo de *pitch*, geralmente representado pela letra β , é determinado a partir da posição das pás que extrai o máximo torque do vento. Sendo assim, se considerarmos que isso ocorre para a posição atual das pás na figura 6, a linha AO representaria $\beta=0^\circ$ e seria a referência. Dessa maneira, conforme Ω diminui, β aumenta e a turbina tende à posição de *stall*, ou seja, para ângulos grandes de *pitch*, menor é a potência mecânica fornecida.

O correto ajuste desse controle está intimamente ligado à eficiência da geração de energia. Esse fato pode ser observado na Figura 4: há grande variação de eficiência conforme o ângulo de *pitch* (Θ , no caso da figura) e velocidade de ponta de pá. Ainda nota-se que ângulo de *pitch* mínimo não garante altas eficiências em velocidades de ponta de pá baixas.

Atualmente, para aerogeradores de megawatts de potência, esse controle é individual para cada pá visando otimização da distribuição de esforços mecânicos no conjunto do rotor com as pás. No desenvolvimento desse trabalho, todas as pás apresentarão o mesmo ângulo de *pitch*.

5.2 Controle de Direcionamento da Nacele

O controle de direção do aerogerador consiste no controle de apontamento da nacele. Esse apontamento está diretamente relacionado com a potência mecânica aplicada ao eixo do gerador. Sabe-se que a perda de potência mecânica entregue ao eixo segue uma relação senoidal com o eixo do gerador e direção do vento, apesar de algumas distorções causadas pela incidência lateral do vento nas pás do gerador. Portanto, para obter o máximo torque mecânico no eixo do gerador e evitar esforços laterais nas pás do aerogerador, é necessário alinhar a nacele com o vento.

Em aerogeradores de pequeno porte, em geral com potência nominal abaixo de 1kW, esse controle pode ser passivo, isto é, um leme posicionado atrás da nacele faz com que o próprio vento modifique a direção do eixo do gerador, sem ser necessário o dispêndio de energia para tal mudança. No entanto, para aerogeradores de maior porte, esse controle é ativo.

5.2.1 Descrição Física do Sistema de Direcionamento

A configuração mais comum desse sistema consiste em um apoio em forma de anel na parte superior da torre de sustentação do gerador, sobre o qual rolamentos sustentam a nacele, permitindo-a girar livremente sobre a torre.

Uma saliência interna à torre, dentada, é conectada à nacele por meio de engrenagens ligadas à motores em corrente alternada, responsáveis pelo movimento giratório da nacele. Esses motores são acionados por um sinal correspondente ao ângulo do eixo do gerador e a direção do vento. Além disso, uma caixa de engrenagens é necessária para aumentar o torque aplicado.

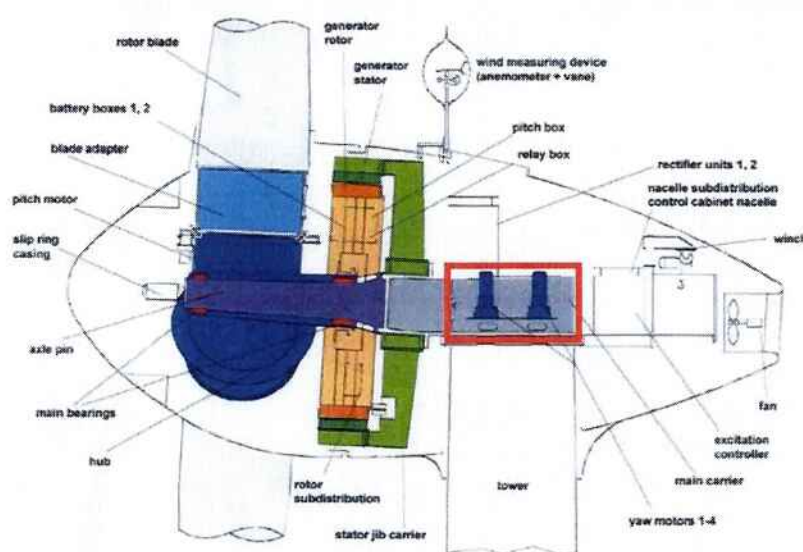


Figura 7: Motores de atuação do controle de yaw realçados em vermelho.
Retirado de [11]

Além disso, para não causar o desgaste das engrenagens e aumentar a precisão do posicionamento é necessário um sistema de freios que pode ser hidráulico ou elétrico. Para tanto há um disco interno de freio da nacele, onde o sistema de freio, por meio do atrito, atua.

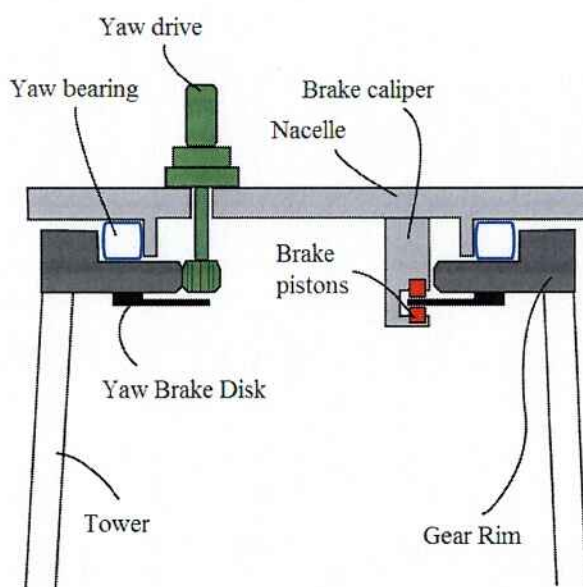


Figura 8: Estrutura física do mecanismo de yaw. Retirado de [12]

5.2.2 Bloco de Controle do Sistema de direção da Nacele

Para fazer o controle do ângulo da nacele em relação à direção do vento é necessário obter informações de um anemoscópio, que indica a direção do vento, geralmente posicionado acima da nacele, atrás das pás. Sabe-se que essa medida de direção é imprecisa devido à turbulência no vento causada pelas pás, no entanto a direção real do vento pode ser aproximada pela direção média do vento medida nesse aparelho.

A partir dessa medição, para que o posicionamento da nacele não seja constantemente ativado, submetendo a turbina à esforços, é definido um limiar. A partir desse limiar, o controle da direção da nacele passa a atuar, girando a nacele sobre a torre até que a diferença entre a direção média do vento e a direção da nacele seja nula.

Outro inconveniente do controle ativo da nacele é que ao girar sobre o eixo da torre, se mantido sempre o mesmo sentido de rotação da nacele, o cabeamento interno à torre é torcido, assim, deve existir um limite máximo de rotações para que não ocorra o rompimento desse cabeamento. O limite de rotações da turbina em uma mesma direção é de cerca de 5 voltas completas em torno da torre segundo [11]. Além disso, é importante evitar que a turbina seja direcionada na posição oposta ao vento para evitar esforços excessivos, nas pás e na torre. Um exemplo de estratégia de controle adotada para o direcionamento da nacele pode ser visto na figura 9:

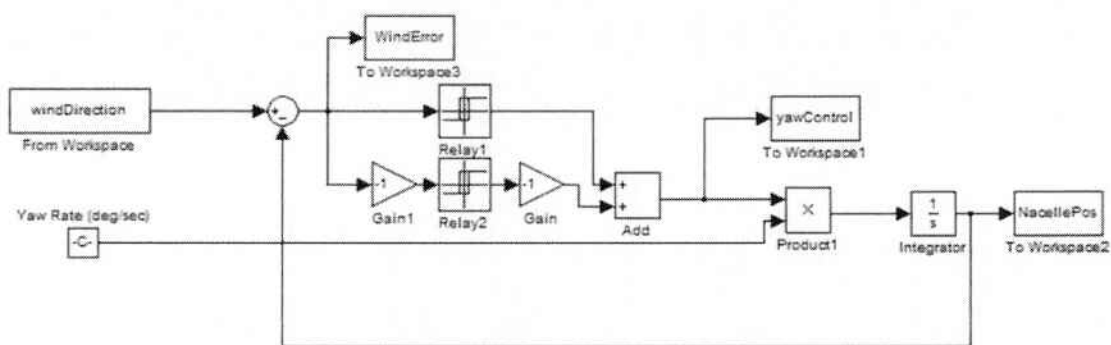


Figura 9: Sistema de controle do yaw. Retirado de [13]

Como entrada do bloco, a direção medida do vento em relação ao eixo de gerador e taxa de variação da direção do vento em relação posição da nacelle. O sistema possui um limiar em que, quando a taxa de variação do vento ultrapassa um determinado valor, o sistema começa a atuar, até que o erro da mudança de direção do vento se torna zero.

6. Controlador Lógico Programável

6.1 Aspectos Físicos

Como já mencionado na metodologia, este projeto contempla o uso de um sistema de processamento em tempo real da *National Instruments*, compactRIO 9024. Denominados 'cRIO' pela empresa, essa linha de controladores é composta por um chassis, o qual pode trazer o controlador embutido ou não, e cartões modulares que aumentam as funções que o conjunto pode realizar. A figura 10 mostra, na parte de cima, apenas o chassis e, abaixo, o chassis com um controlador embutido.



Figura 10: Exemplos de chassis. Retirado de [14]

Os cartões modulares apresentam-se em diversos modelos e, portanto, cada modelo pode realizar funções específicas. Por exemplo, habilitar uma comunicação wireless, realizar uma interface analógica digital, desempenhar comunicação por portas serial, entre outras funções. Uma imagem do modelo que será utilizado está na figura 11, nota-se que o controlador é separado do chassis, logo, pode haver a transferência para chassis maiores caso haja necessidade. O modo de conexão está na figura 12.



Figura 11: cRIO-9024. Retirado de [14]

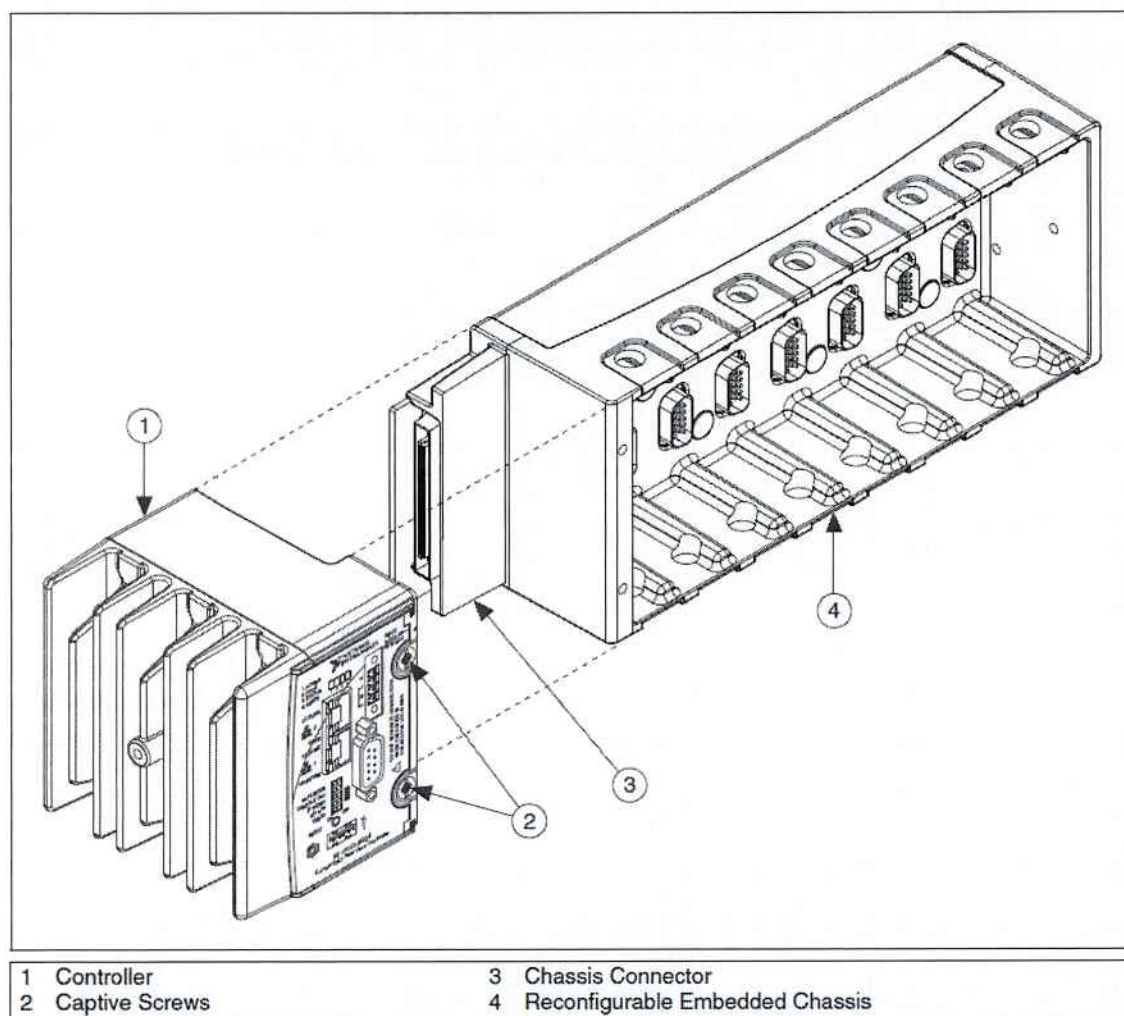


Figura 12: Conexão do controlador com o chassis

Outras características do modelo podem ser encontradas a seguir. Na

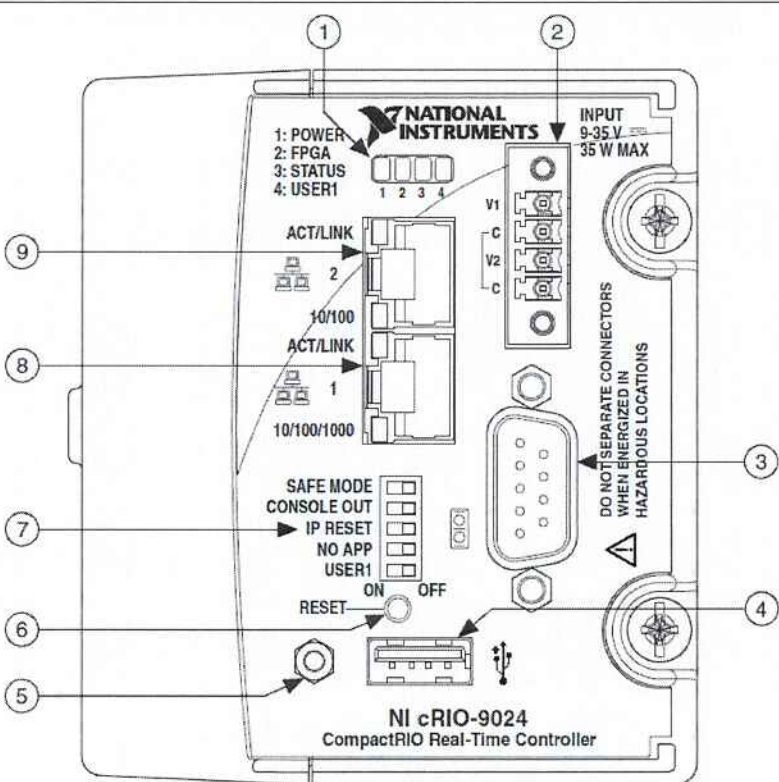


Figura 13: Vista frontal do controlador cRIO 9024.

O cRIO 9024 dispõe de um processador de 800 MHz da *Freescale* para operações em tempo real, utiliza memória de 512 MB de RAM e 4 GB para armazenamento de dados. Apontados na figura 13, tem-se:

1 – *Leds* de indicação. O *led* 1 indica se há alimentação e de qual fonte ela é proveniente já que é possível fazer uma ligação redundante. O *led* 2 ajuda a encontrar erros na programação da FPGA (descrita mais adiante). O *led* 3 não acende durante operação normal, assim, apenas acusa erros, os quais podem ser determinados pela quantidade de vezes que ele pisca. Finalmente, o *led* 4 pode ser programado pelo usuário.

2 – Alimentação do dispositivo. A entrada de alimentação aceita tensões de 9 a 35 Volts de corrente contínua, pode-se realizá-la por qualquer par V – C visto na figura, no entanto, as fontes devem compartilhar o mesmo negativo e devem diferir em seus potenciais de pelo menos 0,5V. O controlador utilizará a fonte

de maior tensão. De acordo com [15], com 8 cartões instalados recomenda-se uma fonte de 55W.

3 – Porta serial que pode servir como input ou output dependendo da programação desejada.

4 – Entrada USB para conexão com memórias externas. Reconhece mais de um dispositivo ao mesmo tempo se for utilizado um extensor de saídas.

5 – Fixação da USB

6 – *Reset*

7 – Chaves de controle. A chave *Safe Mode* determina se o *software* do controlador irá inicializar assim que o controlador for ligado. A chave *Console Out*, quando ligada, emite informações sobre o *firmware* e *IP* para a porta serial. Já a chave *IP reset* troca a configuração da rede pela configuração de fábrica. A chave *NO APP*, quando ligada, não permite a execução do código gravado no controlador assim que o mesmo é ligado. Por fim, *User 1* é uma chave para atender às necessidades do usuário.

8 e 9 – São conexões de rede Ethernet. Toda a necessidade de configuração do cRIO-9024 deve ser feita exclusivamente pela porta 1, indicada, nesse caso, como 8.

6.2 Arquitetura

Quanto à arquitetura, os sistemas da *National Instruments* contêm além do processador, um *chip* FPGA (*Field Programmable Gate Array*), por isso foi citada sua presença nos *leds* da figura 13. A FPGA administra as entradas e saídas enquanto que o processador, associado a um sistema operacional, realizada a tomada de decisões e comunicação com outros dispositivos. Esse tipo de circuito híbrido apresenta maior confiabilidade e velocidade de operação.

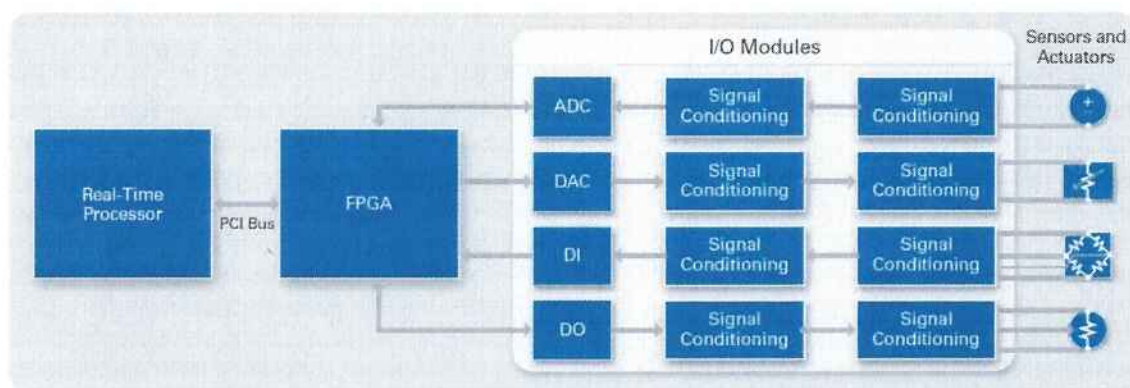


Figura 14: Arquitetura simplificada do controlador. Retirado de [17]

O processador realiza cálculos determinísticos em ponto flutuante com variáveis de até 256 *bits*. Na interface de programação há opções para conferir a qual velocidade o processador está executando o *loop* e, se em algum momento, o código não foi executado dentro do tempo esperado. Outra função presente é a de *watchdog*, forçando reinicialização do processo caso encontre

algum erro grave. A FPGA faz comunicação direta com os cartões de entrada e saída e pode ser reprogramada para realizar algumas tarefas de processamento, tornando a resposta do sistema mais rápida.

Apesar de todas essas funcionalidades, a programação da FPGA para máxima velocidade de comando dos *inputs* e *outputs* não é uma tarefa simples, além de que ela deve ser sincronizada com a velocidade do processador com a finalidade de não perder valores adquiridos. Essa comunicação pode ser realizada por *buffers* ou pontualmente. O modo mais fácil é utilizar uma programação de fábrica para a FPGA e realizar o tratamento das entradas e saídas pela *NI Scan Engine*. Nesse modo, a máxima velocidade de aquisição de dados é de 1 kHz, impondo um ciclo máximo de processamento de 1ms, velocidade significativa quando comparada à velocidades de controladores lógicos programáveis comuns.

O último modo de operação consiste em usar a *NI Scan Engine* para alguns cartões e programar a FPGA para controlar os outros cartões do chassis. Esse método desperdiça algumas potencialidades do sistema já que vai limitar a memória disponível de tratamento de dados da FPGA uma vez que a prioridade fica com a estrutura de *Scan Engine* nesse caso.

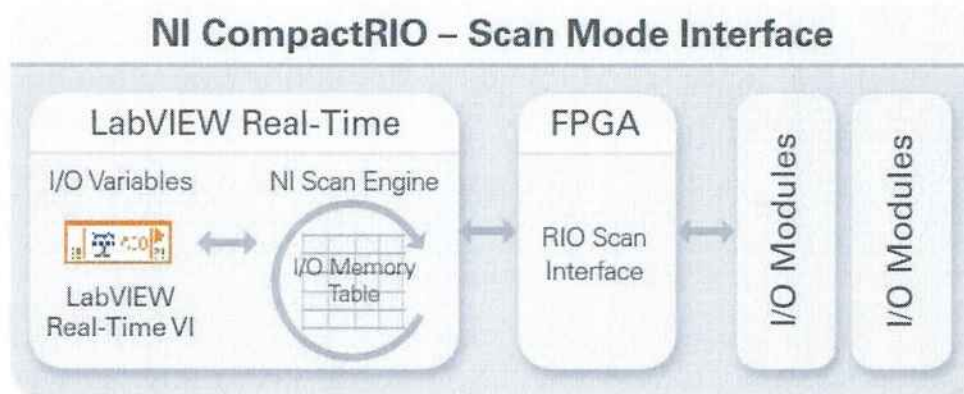


Figura 15: Estrutura do modo *Scan Engine*

Nesse projeto, optou-se pela configuração de *Scan Engine* na máxima velocidade disponível para programação das entradas e saídas, dessa maneira, o código foi programado para ciclo de *scan* de 1 ms. Quanto aos módulos de entrada e saída, foram utilizados:

- Entrada Analógica: NI 9205 +/- 10V
- Saída Analógica: NI 9264 +/- 10V
- Entrada Digital: NI 9421 24V
- Saída Digital: NI 9472 24V

7. Descrição do modelo de aerogerador do RTDS

Para execução das simulações de operação do aerogerador será necessária a utilização de um modelo pronto do mesmo. Para tanto, o próprio RSCAD, software de interface com o RTDS, possui um caso exemplo de aerogerador com gerador de indução duplamente alimentado. Logo, esse foi o modelo escolhido para fazer o controle e a automação. A descrição mais detalhada dessa simulação pode ser encontrada em [16].

7.1 Rede da simulação

No software RSCAD, os modelos de circuito e controle são construídos em uma tela própria para essa construção chamada *Draft*. O arquivo *Draft* da simulação encontra-se na figura 16:

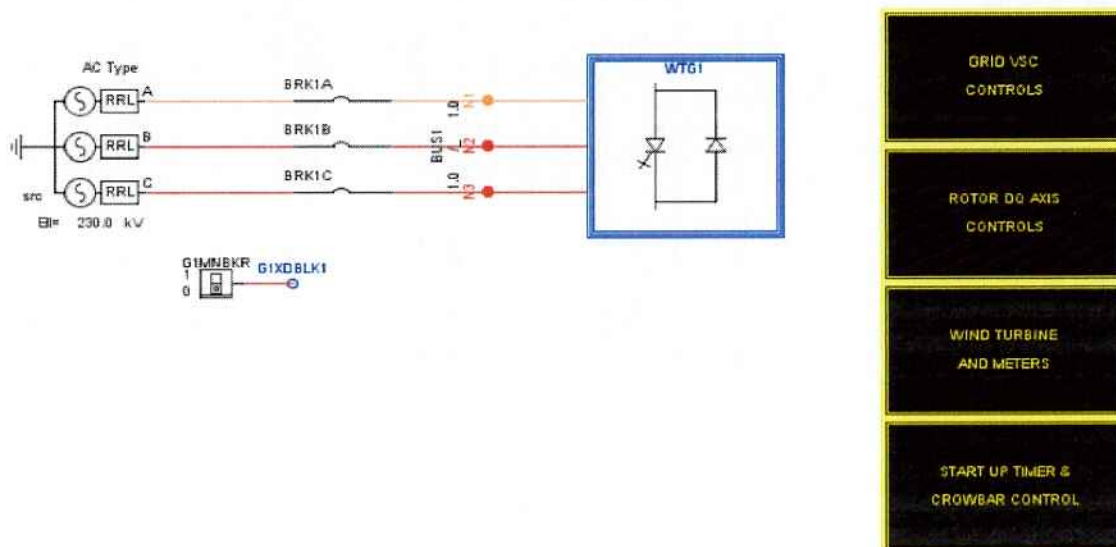


Figura 16: Ambiente de construção da simulação

Nessa tela pode ser observado o modelo utilizado para a rede do circuito. Trata-se de um modelo de um gerador representando um barramento infinito de 230kV que será alimentado pelo aerogerador. O botão denominado G1MNBKR comanda o *breaker* 1 (BRK1A/B/C), quando acionado fecha esse contator, ligando a rede ao aerogerador.

O bloco chamado WTG1 é chamado *Small time step*. É assim denominado por tratar de variáveis que necessitam de um tempo de *scan*, ou seja, um tempo de atualização de variáveis, menor, devido ao fato de uma mudança muito frequente de variáveis. Nesse caso, trata-se do conversor que tem uma alta frequência de chaveamento, segundo [7], de até 5 kHz, ou seja, a cada 200 μ s. No entanto, segundo [16], é necessário para tal simulação um passo de tempo de 3 microssegundos.

Dentro desse bloco encontram-se o conversor, a máquina de indução, possíveis pontos de falta no gerador e na rede, transformadores de conexão com a rede e de interligação do *large time step* (passo de simulação normal) com *small time step* e, ainda, o sistema de *crowbar*.

Ainda nessa tela, podemos observar 4 blocos de controle distintos, sendo que dois deles representam controles relacionados ao conversor *back-to-back*, são eles o *grid VSC controls* e o *rotor dq controls*, não explorados de forma detalhada.

Outro bloco de controle é o *Wind turbine meters*, onde a parte mecânica do gerador está representada, junto com o seu controle de ângulo de ataque das pás, medidas de potência gerada e consumida em cada parte do gerador e cálculo de tensão em cada fase da rede. Como pôde ser observado, nesse bloco de controle, poderia estar implementada a resposta do aerogerador ao controle de apontamento, no entanto, esse controle não é próprio do modelo. Dentro desse bloco, portanto, foi necessária a implementação dessa atitude.

Por fim, no bloco *start up timer and crowbar controls*, podemos encontrar o bloco de controle responsável por controlar a simulação de faltas na rede e nos circuitos internos do gerador, o controle de desbloqueio do eixo rotor com a partida da máquina de indução e o controle da *crowbar*. Sendo esse último um controle responsável pela proteção do conversor em caso de alteração da tensão do link DC.

Bloco de controle WTG1

Na figura 17 podemos observar o circuito interno do bloco WTG1:

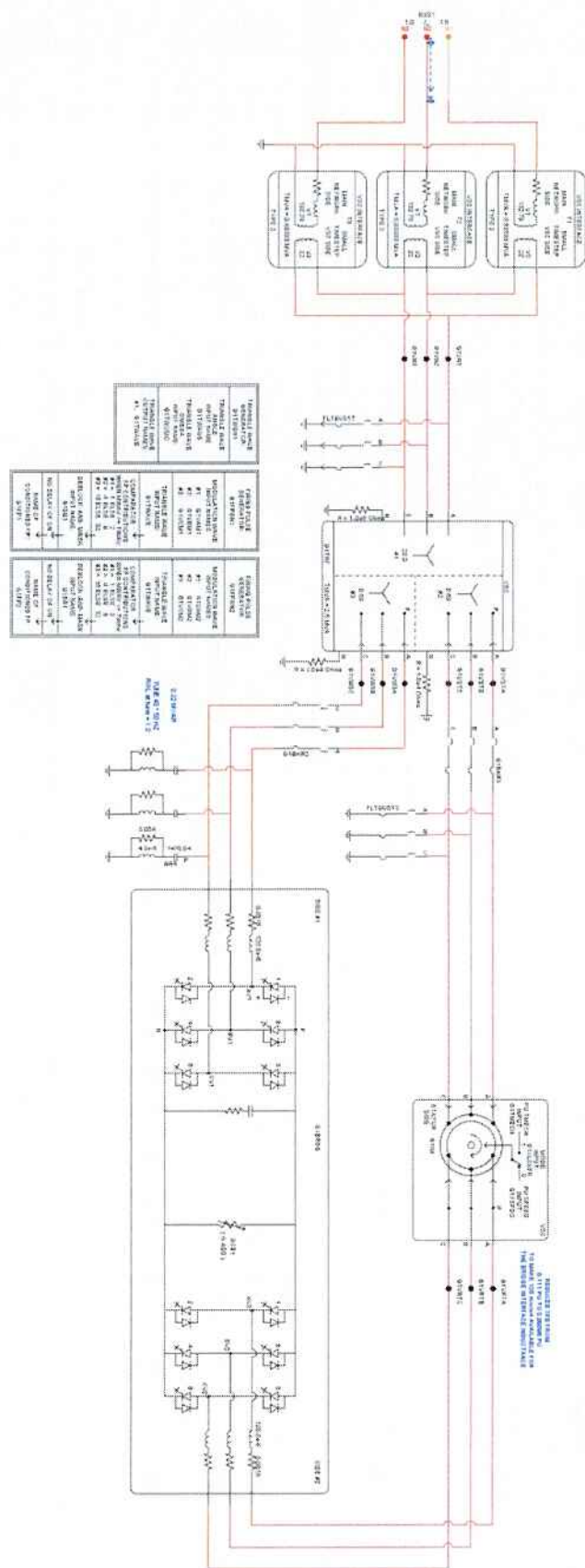


Figura 17: Conteúdo do bloco WTG1 (*Small time step*)

Logo na entrada do bloco há 3 transformadores chamados *VSC interface*, que são transformadores próprios do *Draft* do RSCAD, próprios para converter as grandezas de uma linha que é monitorada fora do *small time step* para grandezas monitoradas dentro do *small time step*. Em seguida, em derivação para terra, encontra-se um *breaker* chamado FLTBUS1T, o qual simula faltas para terra na saída do gerador.

Em seguida, encontra-se um transformador trifásico com 3 terminais que conecta a rede ao rotor e à entrada do conversor *back-to-back*, ajustando a tensão de 22kV para 690V, tensão típica do gerador eólico. A ligação do transformador ao estator, terminal #2 do trafo, possui ainda um terminal em derivação para terra, chamado FLTBUS1S que simula faltas na saída do estator do DFIG. Já na saída #3 do transformador, saída que vai para o conversor encontramos um filtro RLC, para eliminar as distorções de corrente geradas pelo conversor. Ainda nos terminais #2 e #3, há *breakers*, G1BRK1 e G1BRK2, que conectam o gerador a rede.

Ainda nesse bloco encontramos um modelo de máquina de indução duplamente alimentada, controlada pelo modo de velocidade, isto é, o sinal G1XSPDO, que representa a velocidade do rotor que é dada como entrada desse modelo. Por fim, temos o modelo do conversor *back-to-back* que conecta a rede ao rotor da máquina de indução.

7.2 Bloco Wind Turbine & meters

Nesse bloco é possível identificar ainda alguns sub blocos. O primeiro, figura 18, é responsável pelo controle mecânico da turbina.

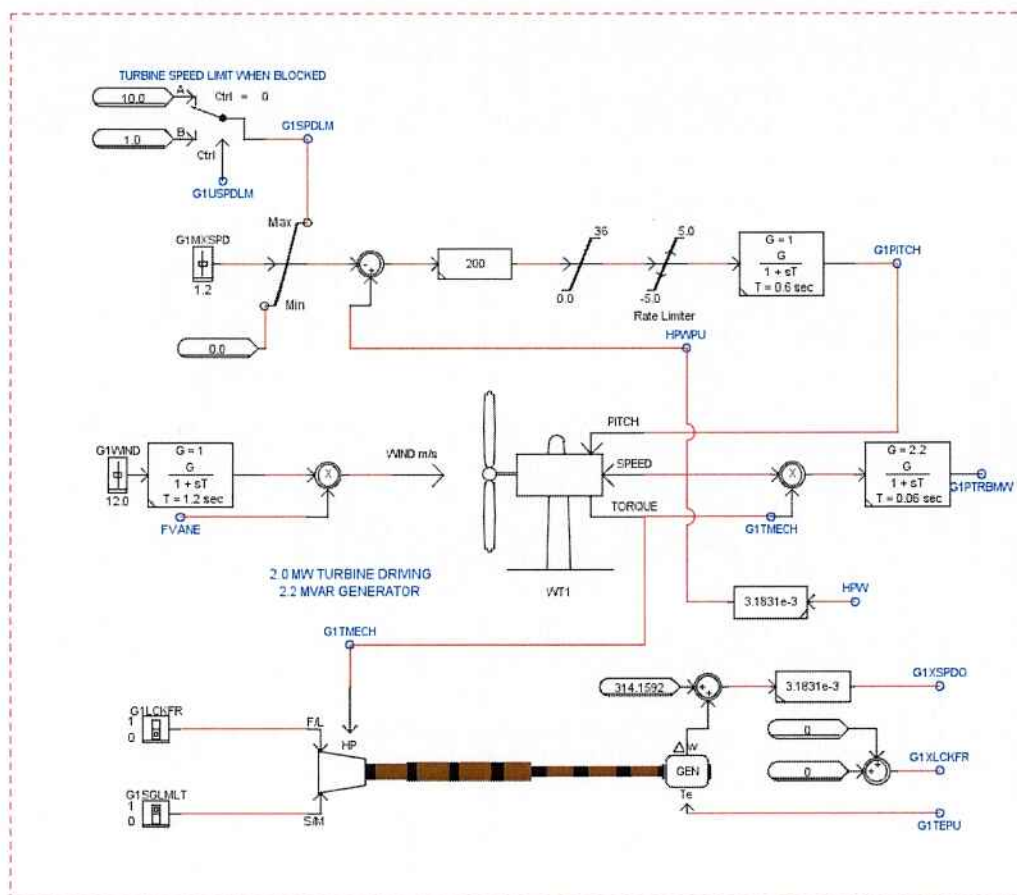


Figura 18: Sub bloco mecânico do gerador eólico

Nesse bloco, visualiza-se na parte superior um limitador de velocidade do eixo do gerador, controlado pelo sinal G1USPDLIM, que quando acionado, define a velocidade máxima do gerador como 1 p.u., velocidade que deverá ser mantida pela variação do ângulo de ataque das pás. O *slider* G1MXSPD define a velocidade máxima que o eixo do gerador pode chegar, sendo que se o limitador G1SPDLIM for menor em magnitude que o valor de entrada do *slider*, então ele passará a ser considerado o valor limite de velocidade. A diferença desse valor com o sinal HPWPU, sinal que representa a diferença entre a velocidade atual do eixo gerador e a velocidade nominal desse mesmo eixo, passa por um multiplicador, um limitador, um limitador de taxa de crescimento e um atraso, gerando um sinal G1PITCH, que é o ângulo de *pitch*.

Os sinais G1PITCH e G1WIND (após passar por um atraso) entram no bloco WT1, modelo mecânico da turbina, e WT1 fornece o valor de torque aplicado nas pás (G1TMECH). Esse valor é o sinal de entrada do bloco *MultiMass*, o qual calcula a velocidade do gerador a partir das inércias do eixo da máquina e do rotor das pás. O bloco *MultiMass* gera o sinal HPW, velocidade do gerador, esse mesmo sinal é dividido por um valor de base e realimenta o bloco WT1, concluindo a malha do sistema. Outra entrada do

bloco é chamada G1TEPU, que é o torque elétrico da máquina, representando o efeito da carga elétrica no aerogerador. A saída do sistema é um sinal Δw , sendo, esse, um sinal que somado com a velocidade nominal do eixo gera o sinal real de velocidade do eixo rotor da máquina, chamado G1XSPDO. O sinal que representa a potência mecânica coletada pelas pás do aerogerador é G1PTRBMW.

O sinal G1XLCFKR é sempre nulo pois ele define que o modelo da máquina de indução (figura 17) será comandado por sinais de velocidade.

Na figura 18, ainda constata-se que esse programa foi criado para simular uma turbina de 2MW impulsionando um gerador de 2,2 MVA. Outro dado importante que o programa fornece, é a curva de eficiência do aerogerador, figura 19. Na mesma, observa-se que a turbina é bastante idealizada, já que para ângulo de *pitch* nulo, o sistema está em eficiência máxima para qualquer velocidade de ponta de pá.

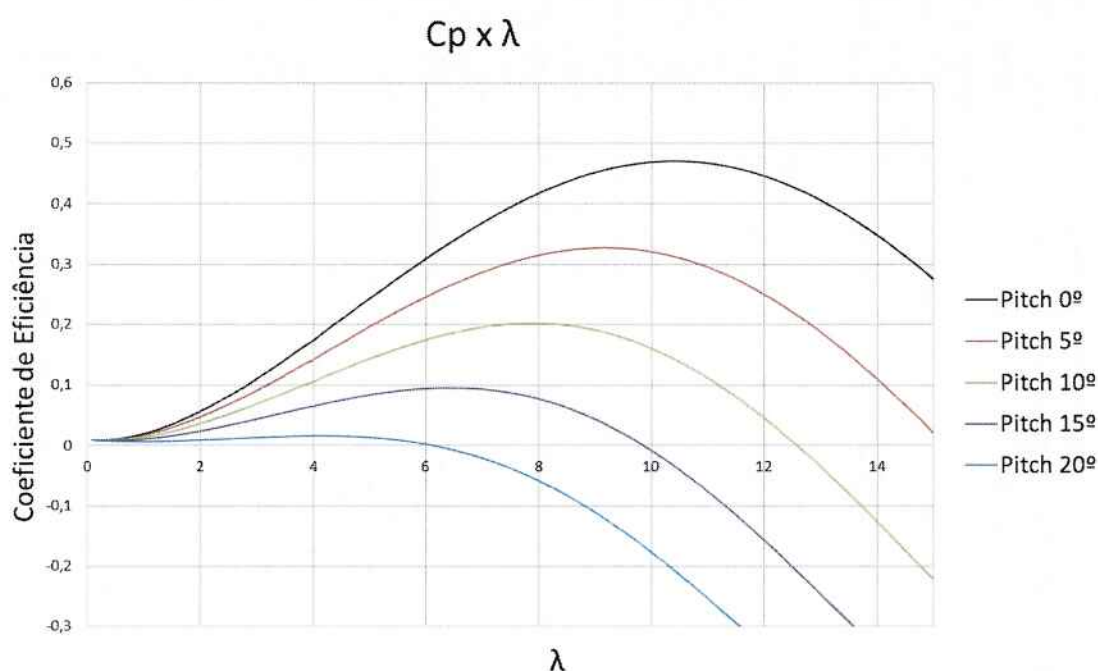


Figura 19: Curva de Eficiência por λ do modelo

Voltando ao bloco da figura 18, encontramos o seguinte sub bloco (figura 20):

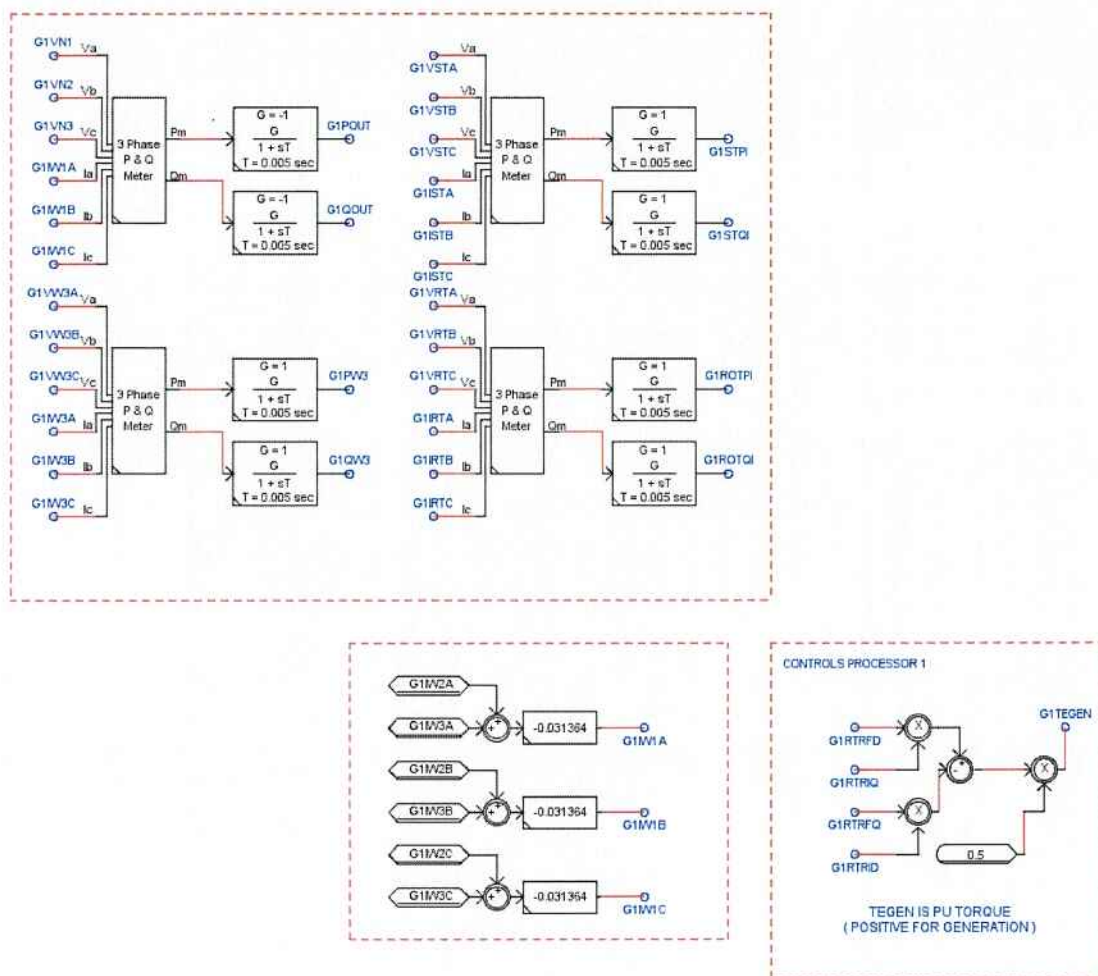


Figura 20: Sub bloco de medição

Trata-se de um bloco de medição que calcula: G1POUT, G1QOUT, potências ativas e reativas na rede geradas pelo aerogerador, G1PW3, G1QW3, potências ativas e reativas que fluem do conversor para a rede, G1STPI, G1STQI, que são as potências ativas e reativas que fluem do estator para a rede e as potências G1ROTPI, G1ROTQI, que são as potências ativas e reativas que fluem do conversor a para o rotor. Também há o cálculo, por fase, das correntes que vão para a rede (G1IW1A/B/C), por meio da soma das correntes do estator para a rede e do conversor para a rede. Há um pequeno cálculo de referência de magnitude e fase das tensões da rede:

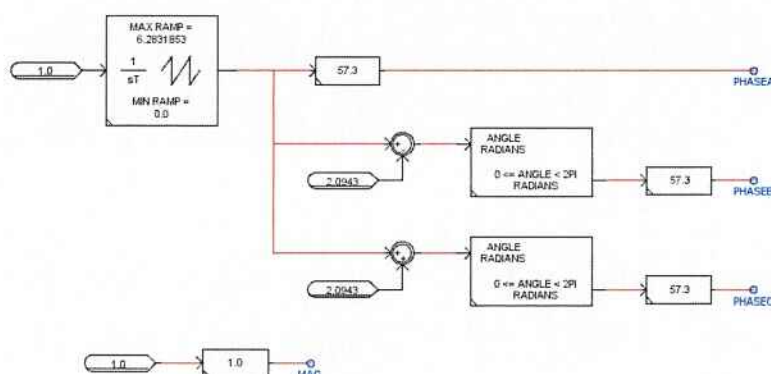


Figura 21: Magnitude de fase das tensões de referência da rede

Por fim, a figura 22 mostra um bloco de controle de apontamento do aerogerador confeccionado para simular esse sistema.

A entrada do sistema é a direção do vento dada por um *slider* e é chamada de VANEABS por ser um valor em radiano de referência do sistema. Esse valor é o ângulo de referência para o qual a nacele deve estar apontada. Esse valor passa por um atraso para que a mudança de direção do vento não seja instantânea, simulando uma mudança mais natural da direção do vento. Logo em seguida observa-se que VANEABS1, que representa a direção referência do vento, ao ser subtraída do ângulo chamado YAW, ângulo de apontamento da nacele, resulta VANEREL1, que é o ângulo diferença entre os ângulos referência do vento e do apontamento, gerando uma diferença entre esses ângulos que é o ângulo entre a direção do vento e a direção da nacele. Após aplicar um fator cosseno, cria-se um fator chamado FVANE, que multiplicará a velocidade do vento na entrada do gerador, representando a componente de velocidade que incide diretamente na área de captação do fluxo de ar do aerogerador.

O sinal VANEREL1 primeiramente passa por um comparador que verifica se esse ângulo é superior a um ângulo mínimo de ativação, definido como 0,35 rad. Em seguida, se esse ângulo for superior ao ângulo de ativação será iniciada uma temporização que define que o vento deve estar com um ângulo de diferença acima do ângulo mínimo de ativação durante um intervalo de tempo pré-definido, que nesse caso está definido como 2 segundos. Esses controles tornam-se necessários dado que é necessária energia para rotacionar a nacele com seus componentes além de submeter a estrutura a esforços, de forma que essa atitude não deve ser acionada a todo momento, mas apenas em casos de real necessidade.

The diagram illustrates the control system for an inverted pendulum. It starts with a reference input $VANEABS$ which passes through a transfer function block $\frac{G}{1+sT}$ with $G=1$ and $T=1.2 \text{ sec}$ to produce $VANEABS1$. This signal is then processed by a $\cos(\text{rad})$ block to generate $FVANE$. The signal $FVANE$ is summed with $VANEABS1$ at a summing junction to produce $VANEREL1$. This signal is then multiplied by a gain of $|X|$ and fed into a conditional block. The conditional block has two inputs: A (the signal from the gain block) and B (a constant 0.35). The output of the conditional block is Y , which is inverted and used as the reset (RST) signal for a timer. The timer has a preset value of 2 and its output Y is fed into another conditional block. This second conditional block also has inputs A (the timer output) and B (a constant 2), and its output Y is $VANEPUSE$. The signal $VANEPUSE$ is then fed into a T-Type Flip-Flop along with a constant 1 and a signal from the $INVERSAO$ block. The flip-flop has two outputs, Q and \bar{Q} , which are used to generate $CTLVANE$ and NEG . The signal $CTLVANE$ is used as the reset (RST) signal for a timer. The timer has a preset value of 4.011001604 and its output Y is fed into a conditional block. This conditional block has inputs A (the timer output) and B (a constant 4.011001604), and its output Y is $INVERSAO$. The signal NEG is used as the control ($Ctrl$) signal for a switch. The switch has two inputs: A (a constant -0.08726) and B (a constant 0.08726). The output of the switch is $Ctrl = 1$, which is used as the control ($Ctrl$) signal for another switch. This second switch has inputs A (a constant 0.0) and B (the signal from the first switch). The output of the second switch is $Ctrl = 1$, which is used as the control ($Ctrl$) signal for an integrator. The integrator has a gain of $\frac{1}{T}$ and a time constant $T=1.0$, and its output is YAW .

A seguir, figura 23, pode-se observar um exemplo simulado de funcionamento do yaw control. Observa-se que em termos de torque, o fator FVANE é cúbico.

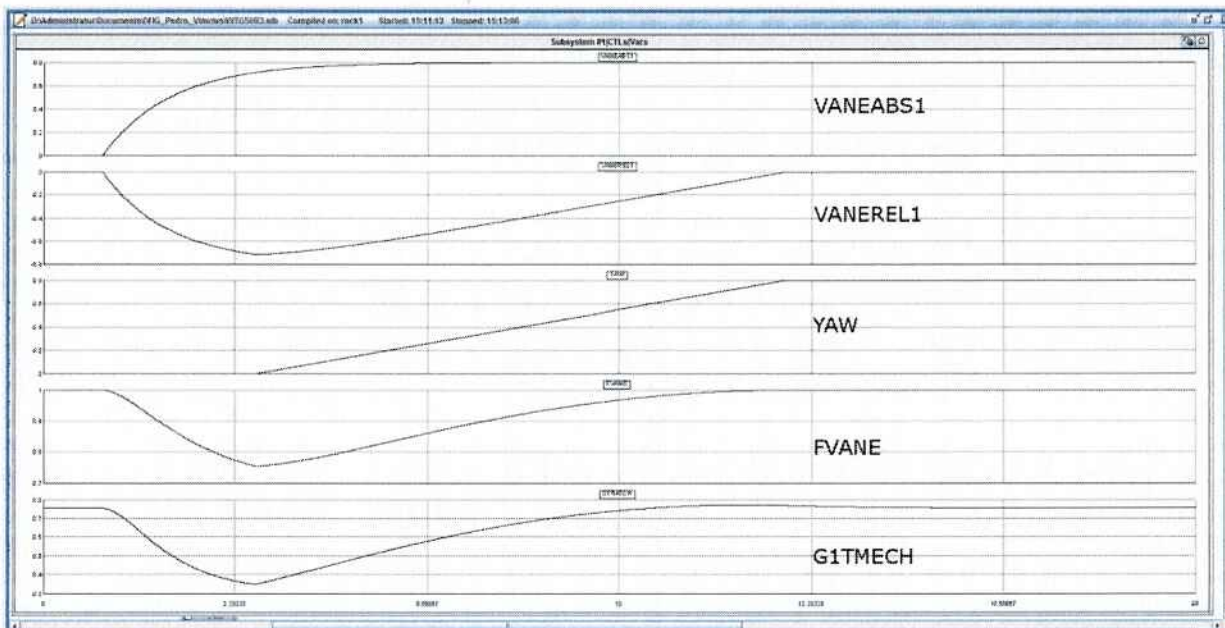


Figura 23: Sinais simulados do modelo

7.3 Bloco *Start up timer & Crowbar control*

Nesse bloco encontramos as seguintes estruturas de controle:

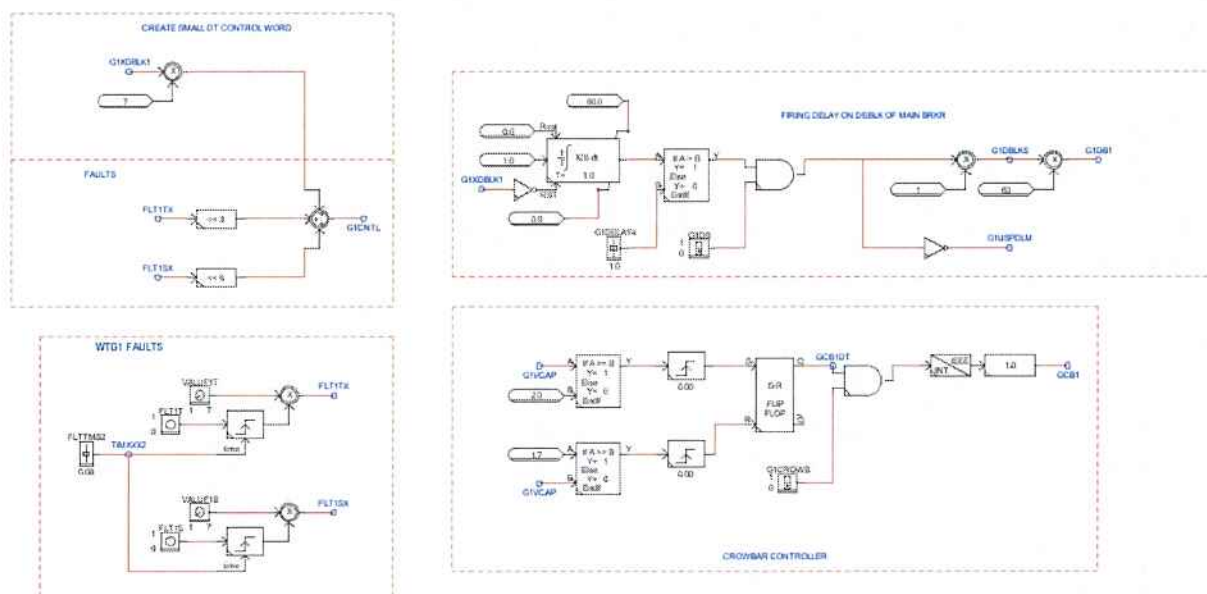


Figura 24: Conteúdo do bloco *Start up timer & Crowbar control*

No sub bloco *WTG1 FAULTS*, há uma estrutura que define alguns parâmetros de faltas na rede que devem ser simuladas. O *slider* FLT1TMS2 define qual o tempo da duração da falta no sistema, VALUF1T e VALUF1S definem o tipo de falta que ocorrerá respectivamente na rede e na entrada do

estator, representados por FLTBUS1T e FLTBUS1S no bloco de *small time step*. Os botões FL1T e FL1S acionam essas faltas respectivamente na rede e no estator. Já o sub bloco *Create Small dt Control Word* gera uma sequência de 9 bits que representa as fases que serão fechadas nos *breakers* do bloco *small time step*. Os 3 primeiros bits (menos significativos) conectam a geração à rede pelo fechamento dos *breakers* G1BRK1, G1BRK2 e G1BRK3. Os três bits seguintes controlam FLTBUS1T, e os três últimos o *breaker* FLTBUS1S.

Já o sub bloco *firing delay on DEBLK main BREAKER* define um tempo em que o eixo rotor fica bloqueado em 1 p.u. de velocidade, após a partida do gerador. Para tanto, G1DB deve estar acionado e o *slider* G1DELAY4, definirá o tempo que o rotor deve permanecer bloqueado após comando de partida.

Por fim, o sub bloco *crowbar controller*, define o acionamento da *crowbar* pelo sinal GCB1. Caso a tensão no capacitor do conversor, chamada G1VCAP, ultrapasse o valor de 2 p.u. a *crowbar* é acionada para proteção do conversor, e só desligada do sistema quando essa tensão volta a ficar abaixo de 1,7 pu.

7.4 Interface RunTime

A interface onde as variáveis podem ser observadas em tempo real encontra-se na parte chamada de *RunTime* do RSCAD, nessa interface ocorre todo o processo de simulação e as variáveis podem ser controladas e monitoradas. O *RunTime* permite a inclusão de gráficos que vão se atualizando continuamente, ou gráficos que só são atualizados após um período de tempo determinado (1s, 40s, 120s, etc). É possível criar botões e sinalização de comando, criar desenhos, enfim, é o ambiente de interação com a simulação. A seguir (figura 25) tem-se essa interface em funcionamento.

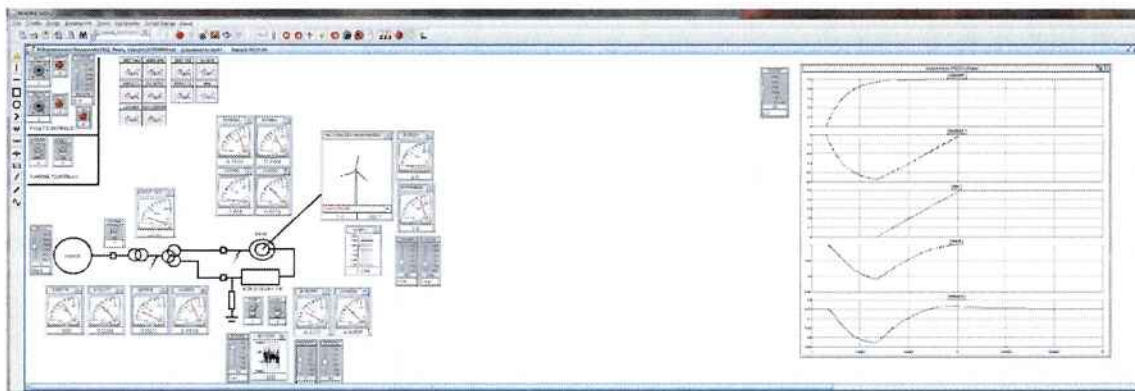


Figura 25: Interface *RunTime*

8. Introdução ao LabVIEW

Com a finalidade de facilitar a compreensão do projeto, explicita-se nesse capítulo o funcionamento básico do LabVIEW bem como a estrutura de programação.

Os programas em LabVIEW são chamados *Virtual Instruments* (VI) e são compostos por duas janelas: o *front panel* e o *block diagram* (figura 26). De acordo com o fabricante, as VIs tem esse nome porque imitam os instrumentos físicos como osciloscópios, botões, leds, switches, etc.

Por essas razões, o *front panel* representa a interface com o usuário, enquanto o *block diagram* contém a programação em blocos. Assim, um programa em LabVIEW é composto necessariamente por essas duas janelas. Ainda na figura 26, demonstra-se que os indicadores ou controles criados no *front panel* aparecem no *block diagram* para serem conectados à programação. Percebe-se que uma parte do código vem dos elementos desejados no *front panel*.

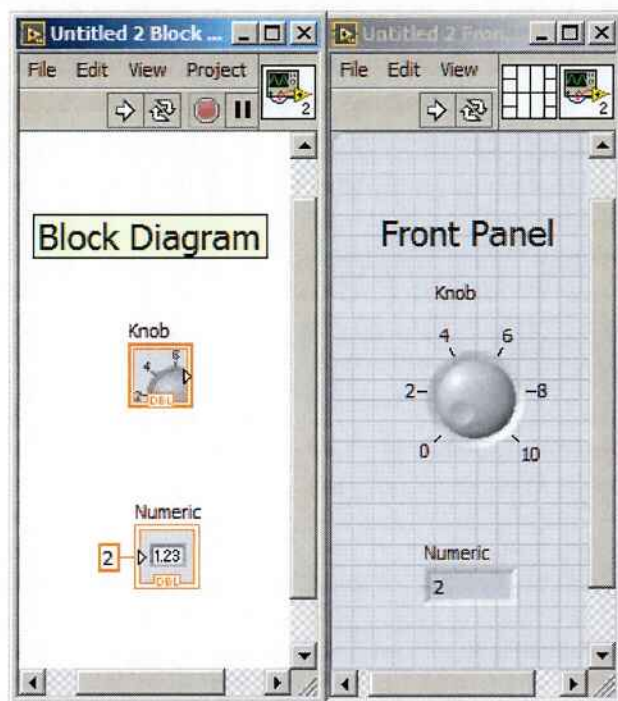


Figura 26: Relação entre os componentes do *front panel* e *block diagram*

Destaca-se, ainda, que dificilmente é possível organizar toda a programação em apenas uma VI, logo, o código desse projeto vai ao sentido de separar as VIs por funções, ou seja, existe uma VI para cada controle e uma VI principal.

Esta última seria como a função *main* de um programa em C, chamando todas as outras funções na ordem desejada. Logicamente, para criar toda essa interação, o LabVIEW precisa que todas as VIs estejam armazenadas em um mesmo projeto. A figura 27 mostra a janela do *Project Explorer*.

No *Project Explorer* administram-se diversas funções, entre elas, pode-se citar o ato de adicionar novas VIs, adicionar pastas, adicionar os *targets* (loais aonde as VIs serão executadas – próprio computador, cRIO 9024, etc), entre outras.

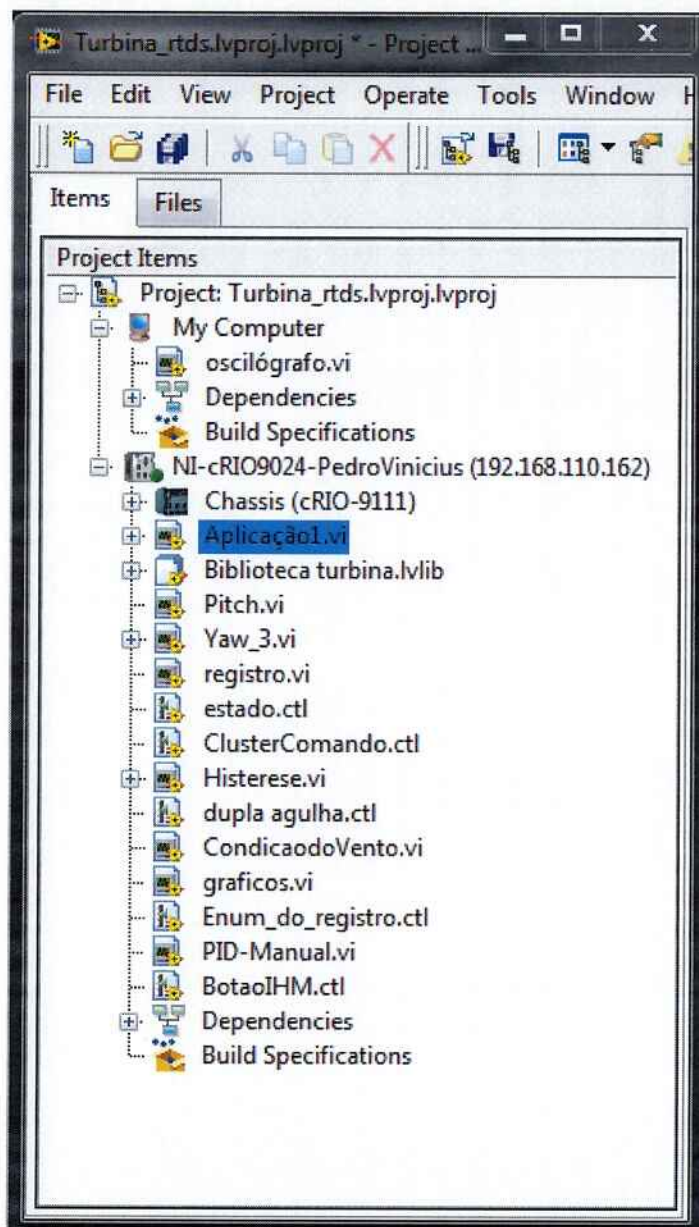


Figura 27: Janela do *Project Explorer*

8.1 Arquitetura de programação

O LabVIEW desenvolve a programação em blocos e apresenta tanto aspectos mais comuns (uso de *arrays*, vetores, funções lógicas), quanto algumas estruturas específicas (*clusters*, *shared variables*). Outro ponto interessante é que o LabVIEW faz execução paralela de todo o código presente no *block diagram*, ou seja, a disposição do mesmo não determina sua ordem de execução, sendo assim, existem outras maneiras para se construir um código sequencial, até mesmo usando o bloco que cria uma sequência de programação. O código criado pode ser executado em qualquer dispositivo que suporte a linguagem sem a necessidade declarar novas estruturas. Nesse trabalho queremos gravar nosso programa em um controlador externo e isso é determinado através do *Project Explorer*.

No *Project Explorer* determina-se em qual dispositivo a VI será executada, então, deve-se criá-la embaixo da árvore do controlador, no exemplo da figura 27 há VIs embaixo de 'My Computer' (as quais serão executadas no computador) e embaixo de 'NI-cRIO9024-PedroVinicius' (as quais serão executadas no cRIO). Dentro desse arranjo, muitas vezes é necessário trocar dados entre VIs que estão em dispositivos diferentes, para isso, utilizam-se recursos como a *shared variable*.

8.1.1 Variáveis

As variáveis podem ser locais, globais ou *shared variables*. Essas últimas permitem que a informação passe de um dispositivo ao outro através da rede, possibilitando fazer a aquisição de dados do controlador e mostrar em um gráfico no computador local por exemplo. As variáveis locais ficam sempre na mesma VI e são úteis para melhor organização do código. As variáveis globais não foram exploradas, então, não serão explicadas. Dependendo do tipo de informação, as cores das variáveis, dos blocos e dos fios de conexão mudam, relacionando as cores com o tipo de dado da seguinte maneira:

- Azul: número inteiro
- Laranja: número em ponto flutuante
- Binário: verde
- Rosa: *strings*

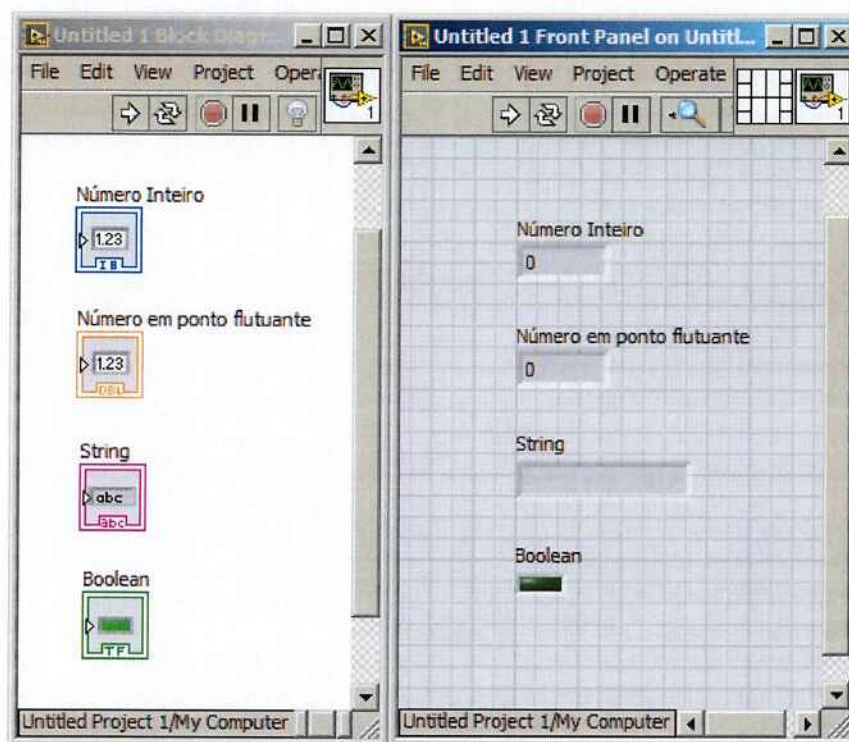


Figura 28: Relação de cores com o tipo de dado

8.1.2 Laços, Testes condicionais, *Shift Registers*, *Enums*, *Clusters*

Com o objetivo de facilitar a descrição do código implementado, são explicadas a seguir algumas estruturas básicas do LabView.

Em LabView os laços são semelhantes *Timed Loop* apontado na figura 29, uma espécie de caixa que cerca o código a ser executado. Todo o código que vem dentro do laço ficará sendo executado até que alguma variável modifique a condição de parada. Na figura 29, observa-se que há uma constante de valor falso conectada à condição de parada, portanto, o laço nunca deixará de ser executado a não ser que o botão de *stop* (abaixo de 'Project' na barra superior) seja pressionado, contudo, isso seria uma ação externa ao programa. Ainda dentro dos laços, pode-se obter informações sobre quantas iterações já se passaram, através da conexão de um indicador ('Indicator' na figura) conectado a 'Iterações', tal informação aparecerá no *front panel*.

Esse tipo de laço (*Timed Loop*) é especial para uso de tarefas determinísticas já que o mesmo tem prioridade sobre qualquer outra execução em paralelo e diferentes formas de se regular o tempo da iteração. Essa estrutura ainda dispõe de variáveis específicas que informam quanto tempo o processador está tomando para executar o código dentro do laço. Se a execução do código ocorre abaixo do tempo determinado nas configurações do laço, o processamento fica ocioso até começar o próximo *scan*, abrindo espaço

para execução de tarefas paralelas. Caso ocorra o contrário, há variáveis que avisam que o tempo está sendo excedido e configurações prévias que decidem por continuar a tarefa ou descartar aquela iteração.

Outra estrutura presente na figura é o *shift register* – SR, o mesmo permite que o resultado do processamento da iteração anterior, esteja na próxima iteração, ou seja, na figura 29, o resultado armazenado no SR da direita vai para o SR da esquerda, sendo utilizado já na próxima iteração. Todo SR deve ser inicializado com algum valor, assim, considerando apenas a operação de subtração, ocorre a seguinte sequência no programa:

- $i=0$ – programa subtrai 5 de 8 e armazena o valor 3 no SR à direita;
- $i=1$ – programa toma o valor do SR à esquerda, 3, subtrai 3 de 8 e armazena 5 no SR à direita.
- $i=2$ - programa toma o valor do SR à esquerda, 5, subtrai 5 de 8 e armazena 3 no SR à direita.
- O ciclo recomeça para $i=1$ e assim permanece.

Percebe-se que o valor do armazenado no *shift register* alterna entre 5 e 3.

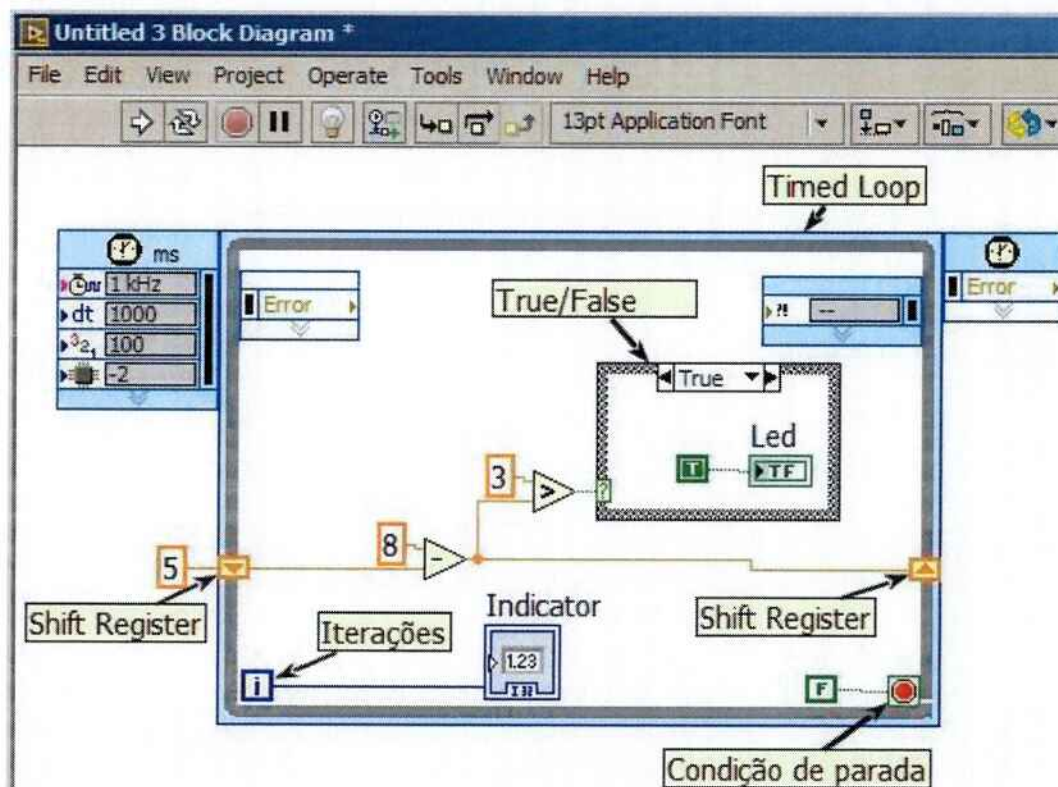


Figura 29: Laço, Teste condicional e *Shift Register* em uma VI

A próxima estrutura presente na figura 29 é o teste condicional, o mesmo também engloba o código a ser executado, mais do que isso, ele o divide entre as condições de verdadeiro ou falso. A decisão de verdadeiro ou falso é o sinal digital que entra no ponto de interrogação na borda da estrutura.

Dentro das condições de verdadeiro e falso, devem ser colocados os códigos correspondentes. No código apresentado, toda vez que o resultado da conta for igual a 3, há a condição verdadeira, sinalizada por um led aceso, na condição falsa, apesar de não explícita na figura, a sinalização é inversa.

Na figura 30, há exemplos da estrutura Enum e do cluster. O Enum possibilita criar uma lista de casos com o nome que o usuário desejar. Adicionalmente, pode-se conectar o Enum a uma estrutura de teste condicional, assim, essa última estrutura dispõe de todos os casos que existirem no Enum, podem ser criados códigos para cada uma das condições do Enum dessa maneira. Na figura 30, criou-se os casos 'Condicao1', 'Condicao2', 'Condicao3' e pode-se observá-los na caixa de seleção da estrutura condicional. Essa é a base para uma criação de uma máquina de estados em LabView.

Nota-se ainda o cluster nessa figura. O cluster é criado no *front panel* e pode-se colocar qualquer elemento no mesmo, tem-se, como exemplo, um 'led', 'Botao1' e o 'Texto'. Dentro do *block diagram* pode-se ler o estado das variáveis do cluster ou alterar esses estados com o uso dos blocos adequados. Uma vez que o cluster está presente no *front panel*, ele se torna uma interface com o código que está sendo executado e, ao mesmo tempo, consegue organizá-lo, pois pode armazenar qualquer tipo de variável, diminuindo o número de blocos no código.

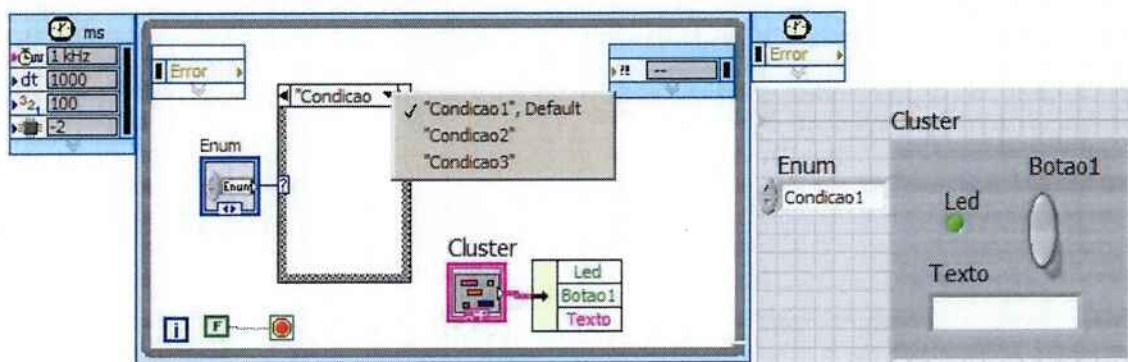


Figura 30: Exemplo de uso de cluster e Enum

8.1.3 Blocos de programação

Os blocos disponíveis são diversos e o objetivo dessa seção é apresentá-los sem aprofundamento.

Os principais blocos são os de funções básicas de programação (laços, condições, variáveis, funções lógicas, funções matemáticas), contudo, há uma enormidade de outros cujas funções podem abranger:

- abrir/fechar um arquivo para gravação;

- criar/alterar um arquivo zip;
- funções matemáticas complexas (probabilidade, integrais, derivadas, operações geométricas, decomposição de matrizes);
- geração de sinais com ou sem ruído;
- geração de imagem de disco de um dispositivo externo;

Enfim, as possibilidades são inúmeras, mas o adensamento de blocos torna o código difícil de depurar. Para contornar essa situação é possível criar subVIs, transformando um código extenso em um novo bloco com entradas e saídas definidas pelo usuário. Tal artifício pode ser visto na figura 31.

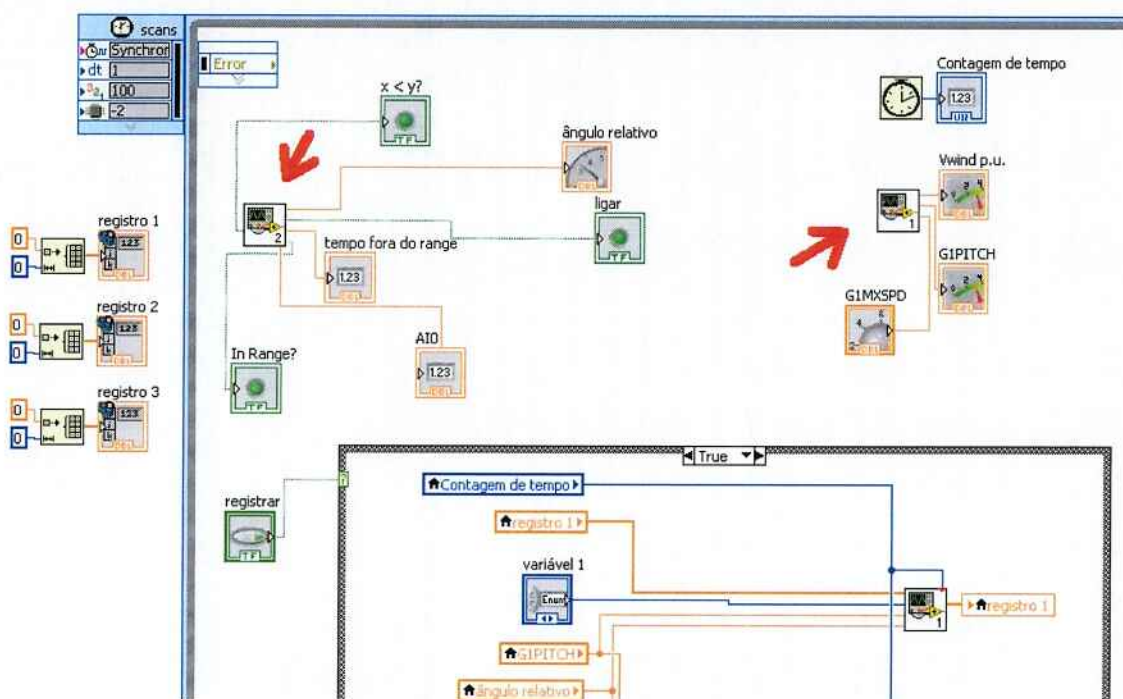


Figura 31: Detalhe das subVIs apontadas em vermelho

9. Integração do RTDS com o cRIO9024

Feita a conexão inicial do RTDS com o compactRIO, seguiu-se o teste da comunicação entre as partes utilizando a simulação da turbina eólica. Para tanto, se executou uma primeira etapa de externalização da automação do aerogerador, inserindo no cRIO a automação de controle do apontamento da nacele. Seguida da externalização do controle do ângulo das pás.

9.1 Controle de apontamento da nacele no cRIO

O controle de apontamento da nacele foi modificado para que sinais de saída e de entrada dessa simulação fossem trocados entre RTDS e cRIO. O bloco de simulação desse controle foi transferido ao cRIO. Nesse novo bloco modificado, foi definido como saída do RTDS, a analógica Y_YAWNORMAL, que representa o apontamento da nacele em relação a uma referência angular 0°. Outra grandeza de saída da do RTDS é Y_VANEABS1, que representa a posição angular do vento com relação à mesma referência angular 0° considerada para o sinal Y_YAWNORMAL. Além disso, como entradas, foram definidos os sinais Y_LIGA e Y_INVERTER, sendo que eles representam, respectivamente, o acionamento do motor de rotação da nacele sobre a torre e o controle de inversão de fases de alimentação desse motor, definindo um sentido de rotação. Chamaremos essa diferença, quando positiva, de rotação no sentido horário e, quando negativa, de rotação no sentido anti-horário. O novo bloco de YAW está apresentado na figura 32.

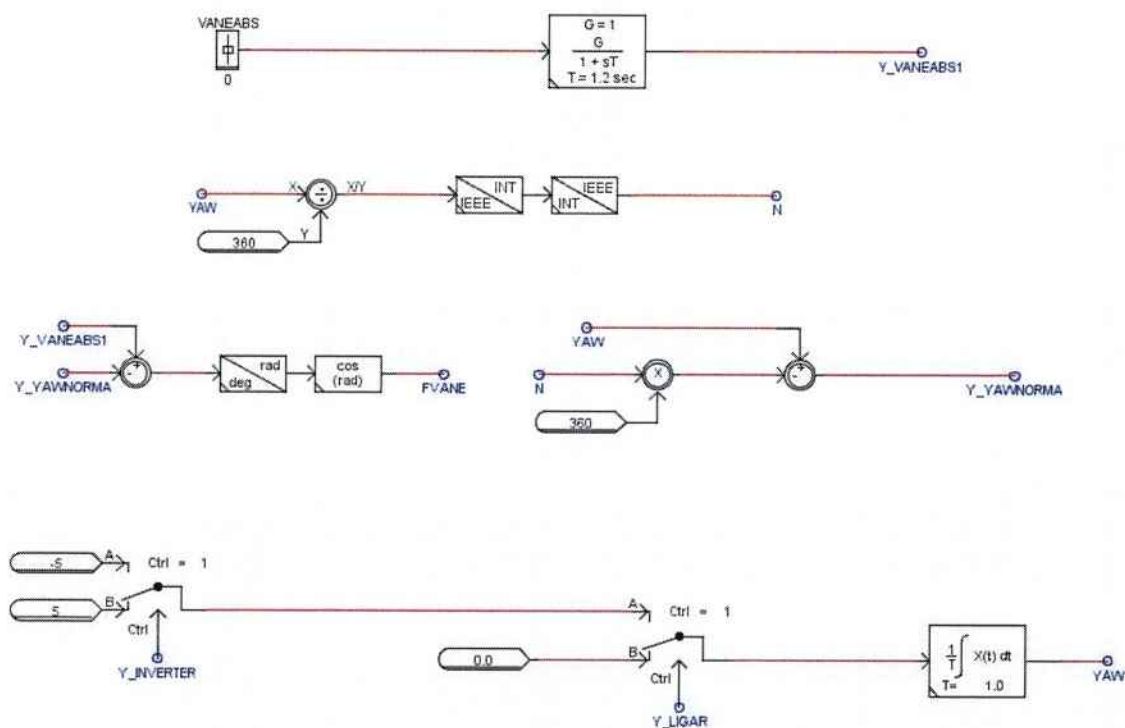


Figura 32: Controle de apontamento da nacele no RTDS

Já no LabVIEW, foram definidos 3 blocos de controle, o bloco de tratamento de entrada, bloco de decisão liga/desliga e bloco de decisão do sentido de rotação. No primeiro, trata-se o ângulo de entrada para definição do menor caminho de rotação para alinhamento com o vento, no segundo bloco, é feito o controle de definição de acionamento do motor até a correção de posição ser finalizada, e o terceiro, define em qual sentido será realizada a rotação. Portanto, as entradas analógicas Y_YAWNORMAL e Y_VANEABS1 traz o ângulo de diferença a ser corrigido e as saídas digitais Y_Ligar e Y_Inverter definem as saídas que, respectivamente, aciona o motor e define o sentido de rotação. A programação em LabVIEW pode ser observada abaixo:

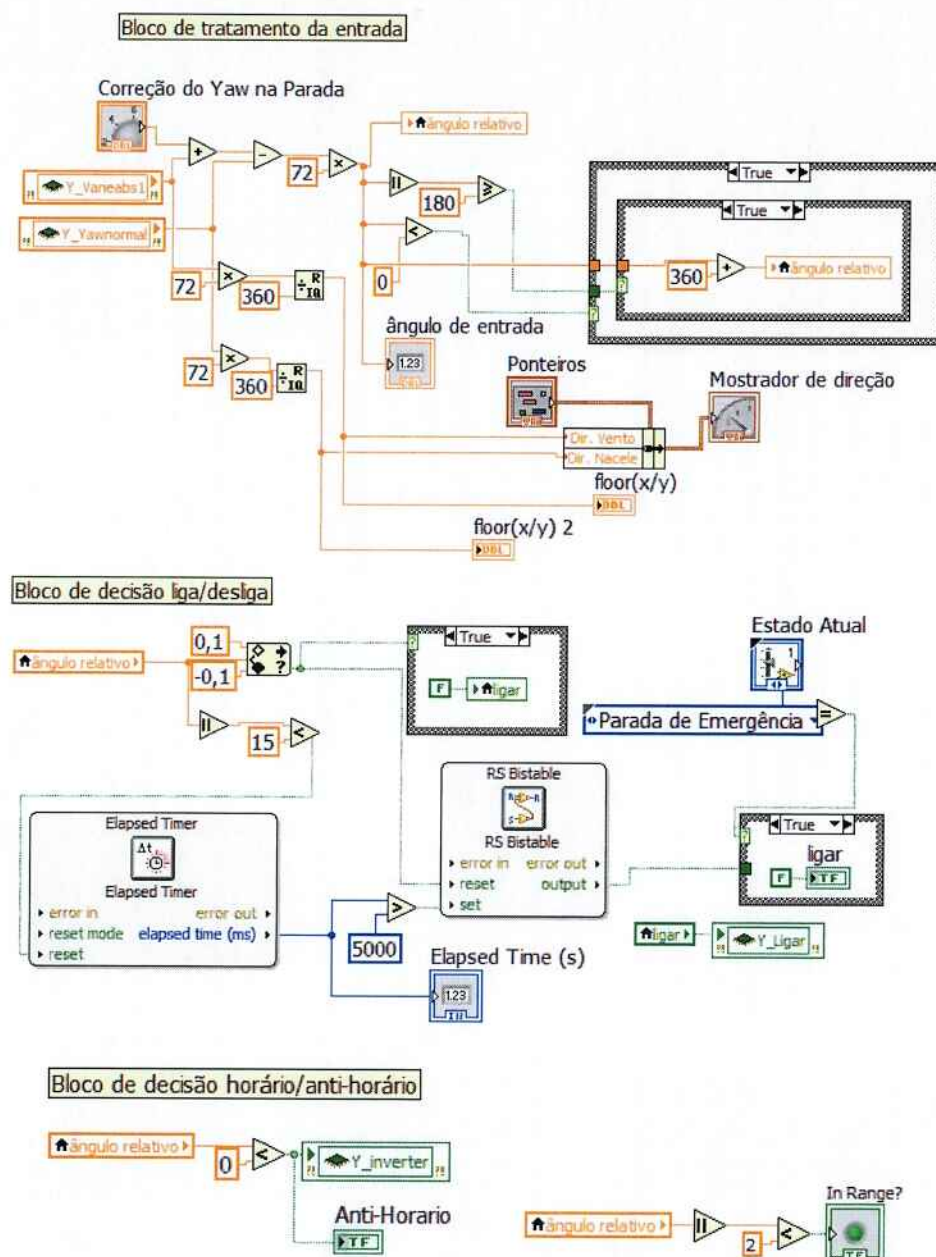


Figura 33: Programação do controle de apontamento da nacele no LabVIEW

As entradas analógicas tem um intervalo de 0V a 5V, que representam 360°. Essas entradas são traduzidas para graus por meio da multiplicação desses valores por 72. Em seguida, subtraindo a posição da nacele da posição do vento, é obtida a diferença angular que deve ser corrigida, chamada no código de ângulo relativo. A partir disso decide-se o melhor caminho a ser seguido pela nacele, de forma a minimizar o percurso e evitar que a nacele fique em direção oposta ao vento, forçando as pás no sentido de rotação oposto ao convencional. Por fim a variável ângulo relativo armazenada é o ângulo relativo que apresenta o sentido de menor caminho a ser percorrido.

Após realização do tratamento da entrada define-se no código de decisão liga/desliga. Através do bloco *Elapsed timer* um tempo mínimo de deslocamento do vento para atuação do controle de apontamento. Nesse caso, foi definido 5s de tempo mínimo. Também é definido um ângulo mínimo de deslocamento para atuação do controle de apontamento: 15°. O bloco *Elapsed timer*, uma vez alcançado o tempo mínimo com o vento acima do ângulo mínimo de atuação, vai dar o sinal de set no *flip-flop* SR, definindo o fechamento da chave de acionamento do motor. A chave só é desligada quando é dado *reset* no *flip-flop*, condição atingida quando a nacele fica alinhada com o vento dentro de uma precisão de 0,1°.

Por fim, o código que define o sentido de rotação avalia se a variável ângulo relativo é positiva ou negativa, de forma que o sinal dessa variável define se a rotação será no sentido horário ou anti-horário. O sinal digital realiza a inversão do sentido convencional de rotação nos casos em que é conveniente adotar esse sentido.

Por fim, dessa automação, realizaram-se simulações para avaliar seu correto funcionamento e foi obtido o resultado da figuras 34, a seguir:

Controle do apontamento da nacele

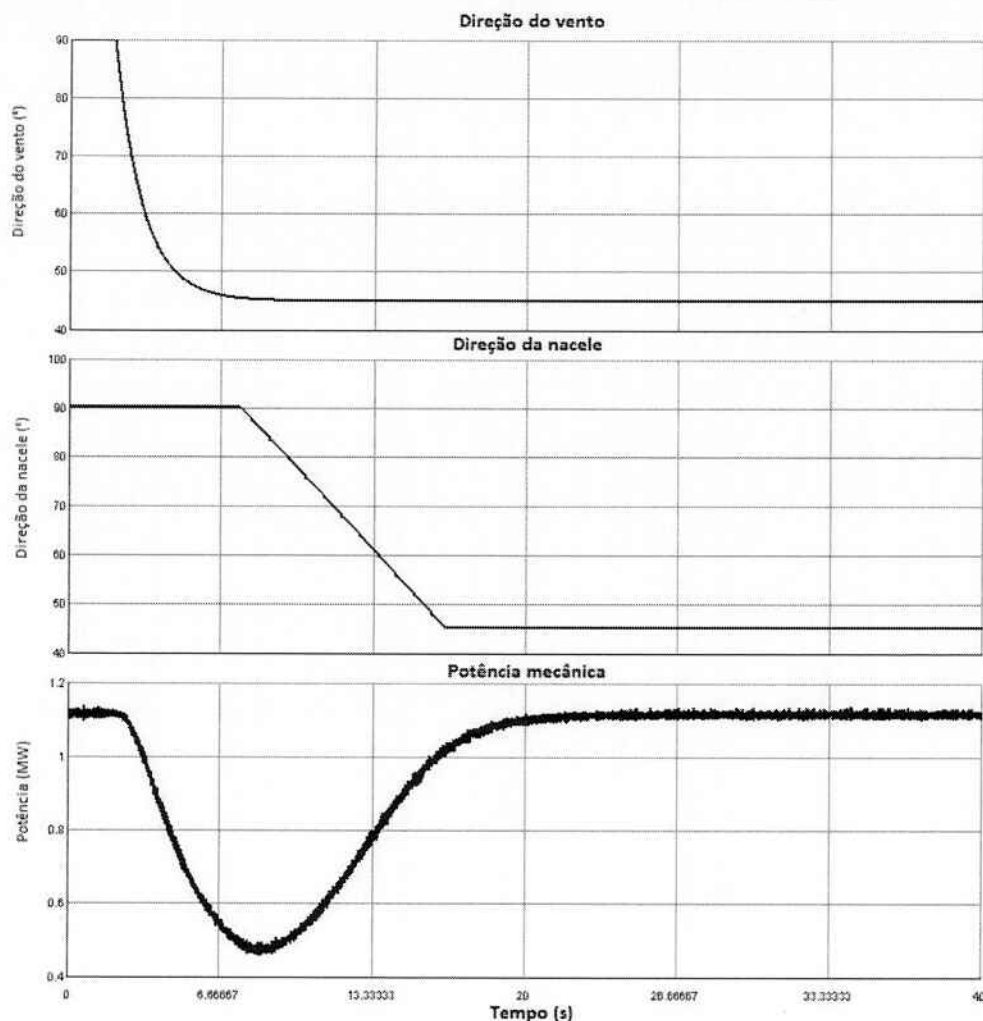


Figura 34: Simulação do controle de posição, com rotação no sentido horário

É possível observar no teste realizado a atuação conforme programado do apontamento da nacele. Após mudança de direção do vento, o controle aguarda 5 segundos para verificação da permanência da mudança de direção do vento e em seguida começa a corrigir a direção da nacele. Dado que a potência mecânica do vento é cúbica com sua velocidade, e ainda, que uma mudança de direção altera a velocidade de um fator cossenoidal do ângulo de diferença entre nacele e vento, observa uma queda de potência que é cúbica com o cosseno da diferença angular dessas grandezas. Assim sendo, verifica-se a importância desse controle para otimizar a geração de energia.

9.2 Controle de ângulo das pás

Na figura 18, disponibilizou-se o modelo de controle de ângulo de pás utilizado pelo simulador. Nota-se que o controle é feito a partir de uma

velocidade de referência, G1MXSPD, determinada manualmente, e de um limitador de velocidade relacionado às condições de operação da turbina. A partir dessa referência, define-se o ângulo de *pitch*. Para tanto, é feita uma relação proporcional entre a velocidade excedente à referência, e o ângulo das pás. Esse valor proporcional é dado pelo bloco multiplicador por 200 e define o ângulo referência de ataque das pás, ou seja, ângulo necessário para que a velocidade do eixo da máquina volte ao valor de referência. Há também, logo em seguida, limitadores de taxa de variação desse ângulo e um limitador de ângulo máximo em 36°.

O sistema do RTDS foi modificado de modo a contemplar um atraso e um integrador, simulando um servo motor. Quanto ao cRIO, desenvolveu-se um controlador proporcional-integral para gerar o valor de ângulo de *pitch* de referência. Mas, o valor enviado para o simulador é a diferença entre o valor de *pitch* de referência e o *pitch* atual do aerogerador. Um esquema em blocos está na figura 35.

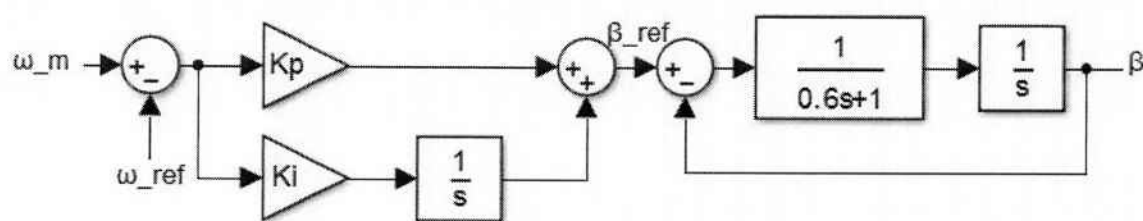


Figura 35: Controle de ângulo de apontamento das pás

9.2.1 Modificação do sistema de *pitch* no RTDS

No RTDS, optou-se por colocar um sistema mais próximo da realidade, no qual simula-se a atuação do *pitch* a partir de um integrador e de uma função de atraso. Também há uma saturação no valor de 0° e 36°, representando um fim de curso. Na figura 36, destaca-se apenas essa modificação, excluindo o resto da simulação da imagem. O sinal que chega para o RTDS é P_PITCH ($P_PITCH = \beta_{ref} - \beta$ pela figura 36), ou seja, é o sinal de erro que irá corrigir a posição do ângulo de ataque das pás. Percebe-se também a presença de um seletor antes do bloco de atraso. Esses blocos de decisão, antes da função de atraso, verificam se P_PITCH está dentro do limiar de [0,065;-0,065], em caso positivo, a entrada para as funções de atraso e integração é nula. Caso contrário, o sinal de entrada é P_PITCH. Essa modificação impede que ruídos provenientes da ligação entre o RTDS e o cRIO permaneçam aumentando ou diminuindo o resultado da integração ($\frac{1}{s}$), prejudicando a resposta do sistema.

Para essa modificação, foi necessário enviar o sinal G1PITCH e o sinal P_Veixo (velocidade do eixo em p.u) do simulador para o cRIO.

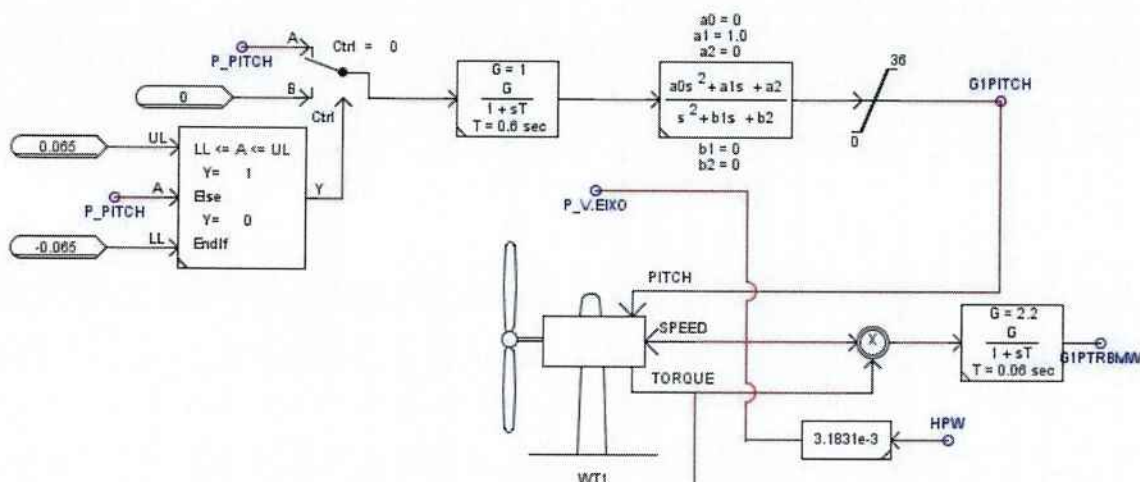


Figura 36: Modificação do sistema de *pitch* no RSCAD

9.2.2 Implementação do controle de *Pitch* no cRIO

No cRIO foram utilizadas duas subVIs. A primeira subVI, figura 37, fornece todos os parâmetros necessários para o controlador PI gerar uma resposta para o sistema. É importante destacar aqui que a atuação do *pitch* em condições de emergência deve ser um pouco mais veloz, portanto, a primeira subVI apresenta uma lógica que chaveia para outro código durante a emergência, desabilitando o controle PI nesse caso especial.

Na figura 37, observa-se que, conectado à subVI PI, tem-se:

- *it_time* (tempo de execução da iteração),
- *vref* (velocidade de referência do eixo),
- *Veixo* (velocidade real do eixo),
- *Kp* e *Ki* (constantes proporcional e integral),
- Constante *False* (impede o reset da integral),
- Indicador 'error?' (indica se houve erro no processo de integração),
- Resposta do controlador multiplicada por uma constante para ser enviada ao simulador através da saída analógica, denominada 'P_PitchError'.

Uma vez que *P_PitchError* é uma variável representativa da tensão de saída analógica e o intervalo utilizado é $[-5V; 5V]$, o seu valor máximo é de 5V. Assim, $36^\circ \times 0,13889 = 5$. As duas estruturas condicionais são lógicas para determinar a máxima velocidade de referência.

A primeira, comandada por *FD_Block* (sinal do simulador que limita a velocidade de referência do eixo se a turbina estiver desconectada da rede), limita a velocidade de referência em 1 p.u ou 10 p.u. Já a segunda, determina a velocidade de referência de acordo com *G1MXSPD*: se *G1MXSPD* é menor que o valor de bloqueio, ele se torna velocidade de referência, caso contrário, o

valor de bloqueio é a velocidade de referência. Já adianta-se aqui que G1MXSPD é um *setpoint* definido ao longo do programa de acordo com os estados em que o aerogerador opera, portanto, nenhum cálculo o define. Como já dito, observa-se que o código observável na figura 37 é válido enquanto o sistema não entrar no estado de parada de emergência.

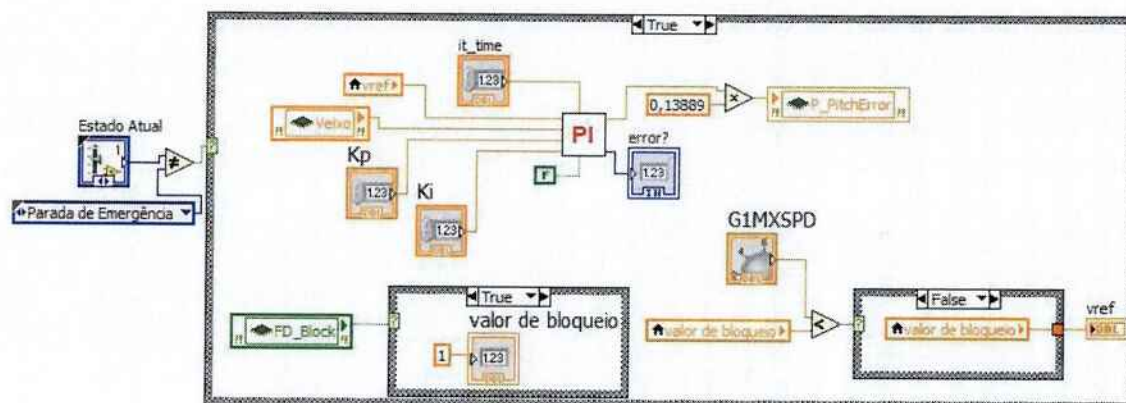


Figura 37: VI que realiza o controle de pitch

A condição de parada de emergência está representada na figura 38. Essa lógica também precisa do tempo de integração já que a ideia é que seja somado o valor de 0,00278V a cada iteração ao valor de P_PitchError até que G1PITCH (valor do ângulo atual do RTDS) seja igual a 5V. No entanto, como o valor de cada integração varia, se esse valor for menor que o estipulado (1ms), deve-se multiplicar 2,78 por 0,001, mas se for maior, deve-se usar o *it_time*. A soma de 2,78V a cada segundo imprime uma rampa de 20°/s até que o *pitch* atinja o valor máximo. Quando G1PITCH for maior que 5V, o sinal de erro é zerado, desabilitando os integradores do RTDS.

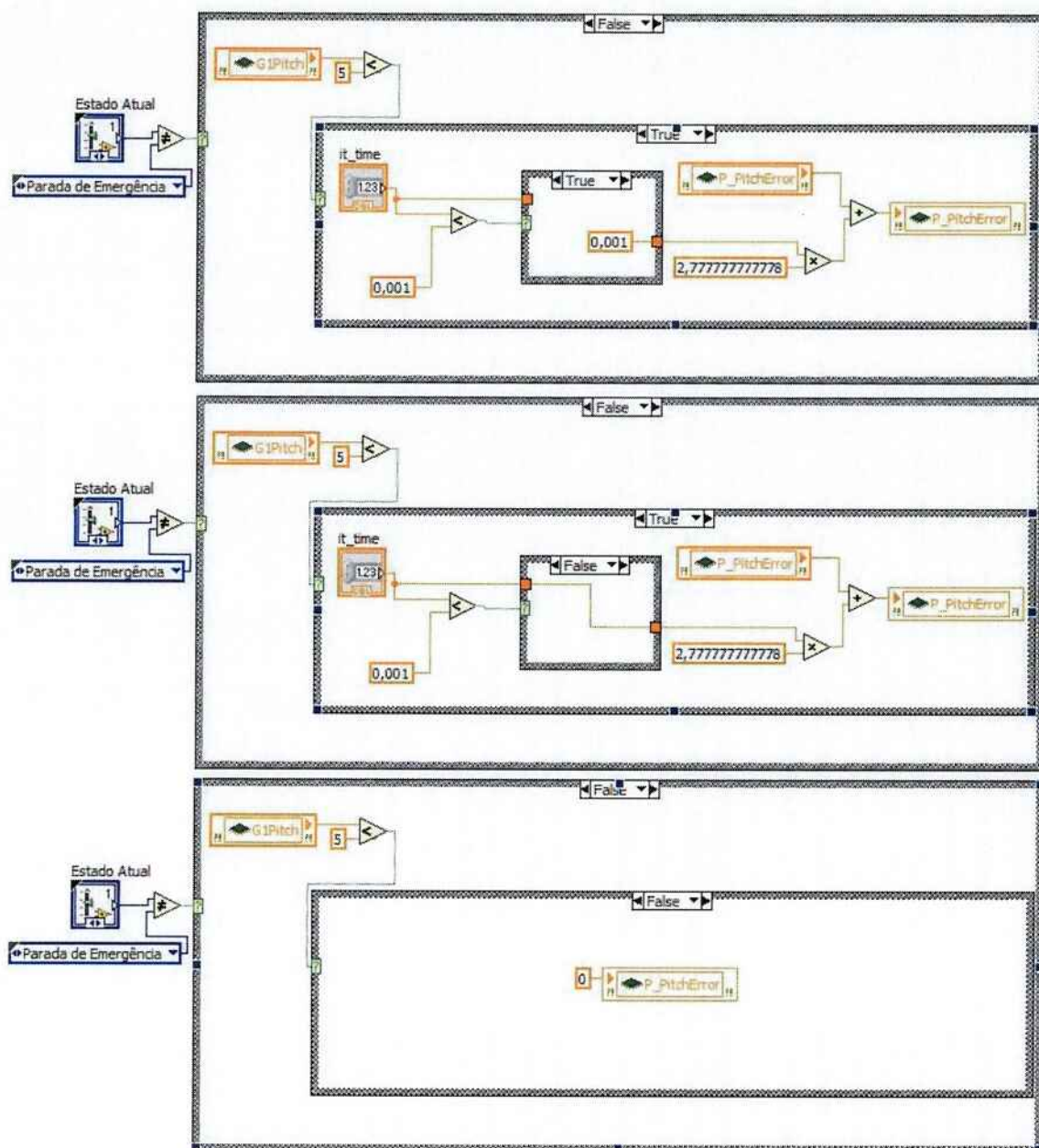


Figura 38: Atuação do *pitch* para parada de emergência

Na próxima figura, 39, demonstra-se o bloco PI, presente na figura 37. Dentro desse bloco há o código do controle proporcional-integral com alguns artifícios de saturação do valor de saída. Identifica-se na figura 39:

- V_m : Velocidade do eixo do rotor (Veixo na figura 37);
- *Setpoint*: V_{ref} (presente na figura 37);
- K_p e K_i : constantes proporcional e integral;
- *it_time*: tempo de cada iteração;
- $G1PITCH$: Valor atual de β fornecido pelo aerogerador
- Output: Valor de erro, em graus, entre β_{ref} menos β (na figura 37, esse valor é convertido para $P_PitchError$).

O 'it_time' passa por um teste condicional para definir o valor do passo de integração, já que o sistema é discreto. Se o tempo de integração for menor que 1ms, o passo de integração será de 1ms. Se o tempo de integração for maior que 1ms, esse valor superior será o passo de integração. O resultado do controlador PI passa pela 'Saturação1', impedindo que valores acima de 36 ou menores que zero passem para a próxima etapa.

Outra característica importante desse código é a saturação da integral. Toda vez que 'Valor da Integral' for maior de 36 e 'Vm' menos 'Setpoint' for positivo ou 'Valor da Integral' menor que 0 e 'Vm' menos 'Setpoint' menor que 0, defini-se um passo de integração nulo, não permitindo que a integral continue calculando após atingir o ângulo máximo ou mínimo. Já a 'Saturação2' é apenas um artifício para zerar o 'Output' assim que o valor de erro for muito baixo.

Os valores de Kp e Ki foram obtidos empiricamente, valem 7 e 3 respectivamente, de forma que a atitude do sistema não é mais veloz do que 10°/s. Para sintonizar tecnicamente o controlador, seria necessário descobrir a relação entre o ângulo de *pitch* e a velocidade de referência no eixo, mas essa relação é não linear e ainda depende de dados mecânicos não claramente fornecidos pelo modelo, como raio das pás, velocidade do rotor em rad/s, entre outros dados. Houve tentativas de se ajustar os ganhos de acordo com [18] e [19], inferindo alguns tamanhos de pá e velocidades do modelo, mas elas foram infrutíferas. Pode-se concluir também que, sendo o modelo de eficiência ideal (de acordo com a figura 19), basta manter o ângulo das pás nulo, que a operação do aerogerador ocorre na máxima eficiência. Esse fato dispensa grandes técnicas de sintonização do controle proporcional-integral.

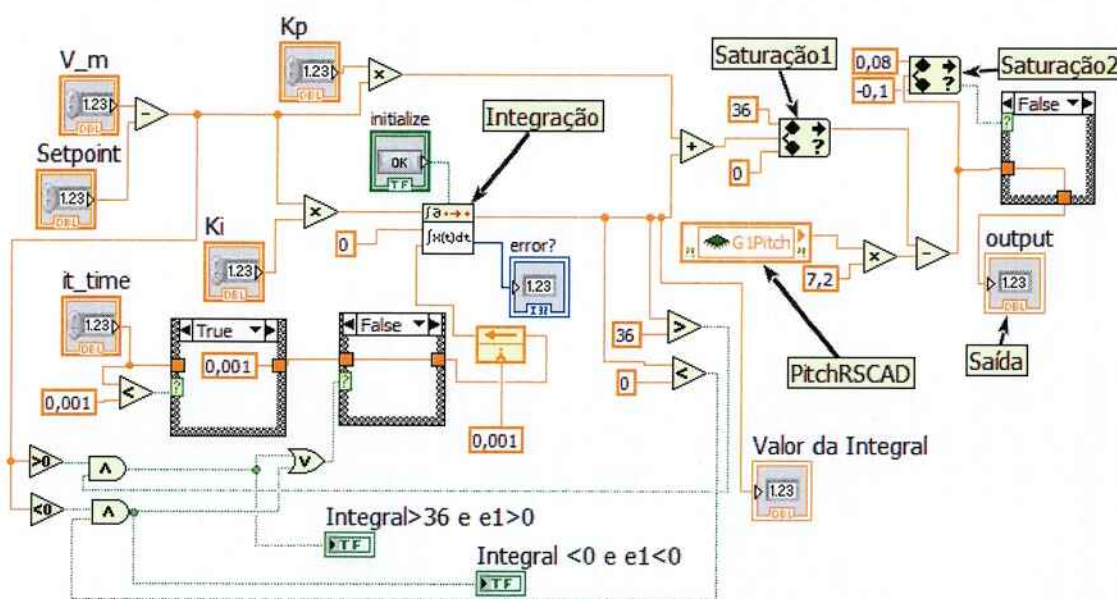


Figura 39: Controlador Proporcional Integral

9.3 Modificação do modelo do RTDS

As modificações para programação do sistema completo abrangeram também o modelo do RTDS, e é isso que será explicado a seguir.

A figura 40, abaixo, resultou de modificações no sistema original. Em comparação com a figura 16, foram adicionados os seguintes blocos: bloco para reorganizar a divisão de processamento no simulador (1), um botão de reconhecimento de falha (2), entrada proveniente do cRIO para fechar os disjuntores da rede (3) e os sinais de *input* e *output* (4).

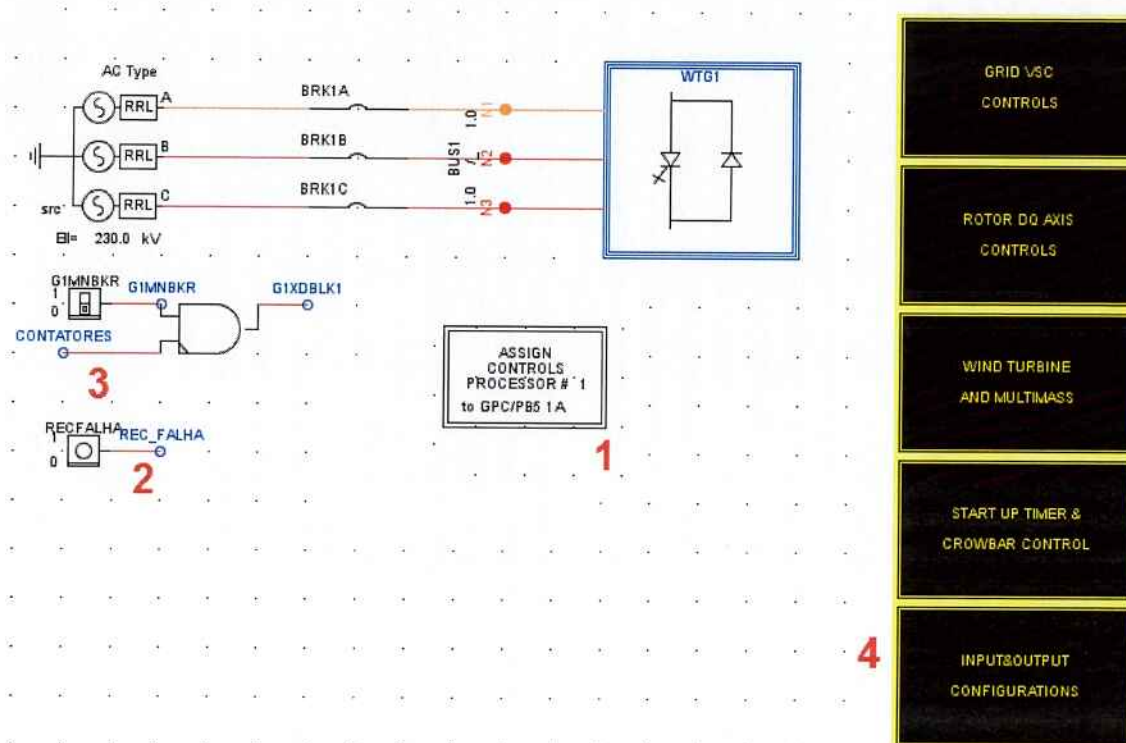


Figura 40: Modificações realizadas no modelo simulado

Devido à adição de lógicas, o RTDS exigiu que houvesse realocação manual de processamento nos cartões apesar de não utilizar nenhum processador a mais.

A necessidade do botão de reconhecimento de falha está presente na explicação da máquina de estados. Esse botão simula a situação em que um operador necessita ir até o parque eólico durante uma parada de emergência da turbina e verificar o que ocorreu. O comando de 'contadores' serve apenas para o controlador controlar os disjuntores do aerogerador, mas, ele só pode fazê-lo se G1MNBKR estiver em 1. G1MNBKR simula o sinal de *trip* da proteção elétrica, assim, o cRIO não consegue comandar o disjuntor da rede e sempre leva o aerogerador a um estado de emergência se G1MNBKR for acionado. O sinal G1XDBLK1 representa o estado do disjuntor do sistema. Para melhor visualização, os *inputs* e *outputs* criados foram separados em uma pasta própria.

Dentro do bloco *Wind Turbine and Multimass* foram realizadas algumas modificações para criar um freio para o sistema. A figura 41 apresenta algumas diferenças em relação à figura 18. Uma vez que o sistema de ângulo das pás já foi explicado, esse texto se voltará para o restante dos elementos.

O sinal de vento recebeu um sinal de multiplicação denominado FVANE (pode ser visto na figura 32). FVANE é o cosseno da diferença entre o ângulo do aerogerador e do vento, ele simula a diminuição do vento incidente conforme o vento muda de direção. No sinal de vento ainda foi incorporado um ruído que é gerado de acordo com um curva Gaussiana, dessa maneira, a simulação fica um pouco mais próxima da realidade pois o padrão do vento é inconstante.

O bloco *MultiMass*, no fim da figura, recebeu um *flip flop* que comanda o sinal lógico de freio desse bloco. As entradas do *flip flop* (Freio e Freio_RST) são geradas pelo cRIO. Toda vez que a saída do *flip flop* for 1, a saída de velocidade será nula, pois será considerado caso de freio aplicado. O sinal G1TMECH foi saturado em um valor máximo devido à indefinição causada ao bloco WT1 quando HPW (velocidade do eixo) vale 0 (condição de freio aplicado). Nessa situação, G1TMECH assume valores não aceitáveis pois a nacele está apontada para o vento mas o rotor não ganha velocidade.

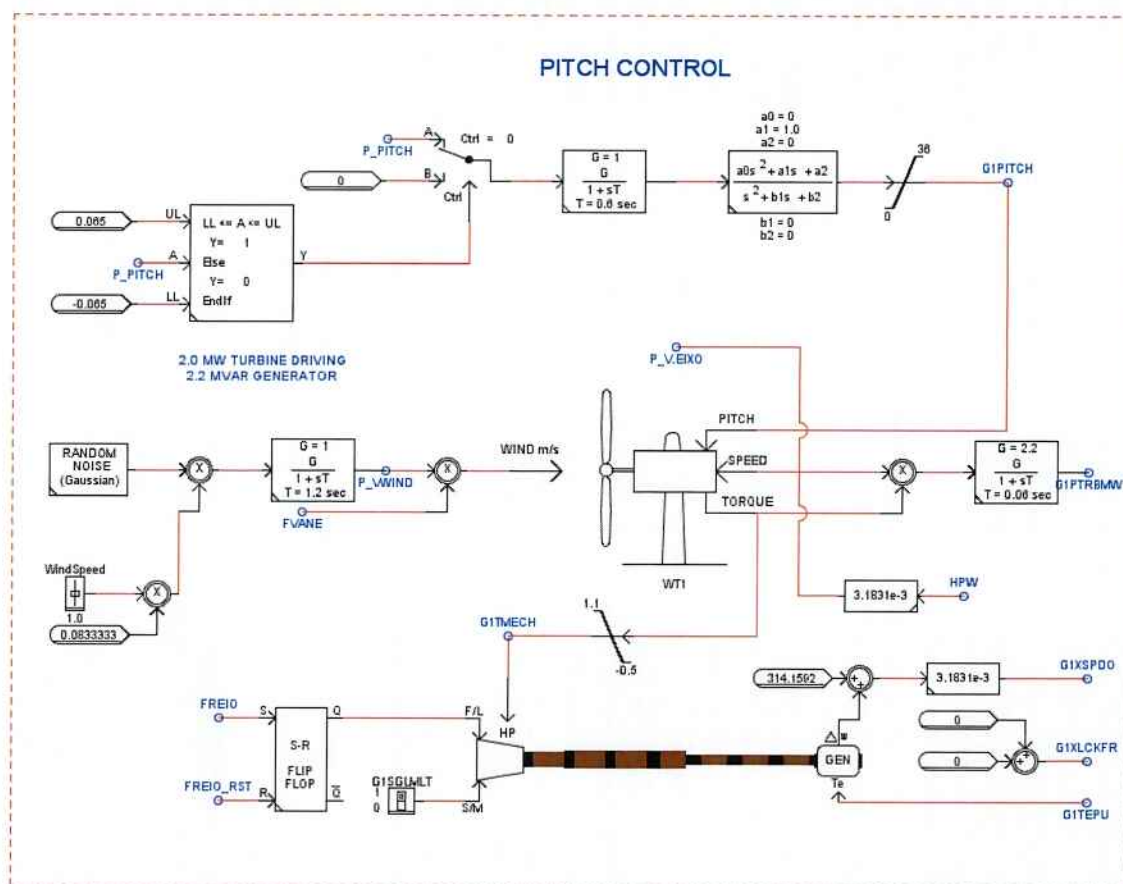


Figura 41: Alterações no modelo de simulação

Ao fim, nas figuras 42 e 43, pode-se observar as entradas e saídas digitais e analógicas do RTDS.

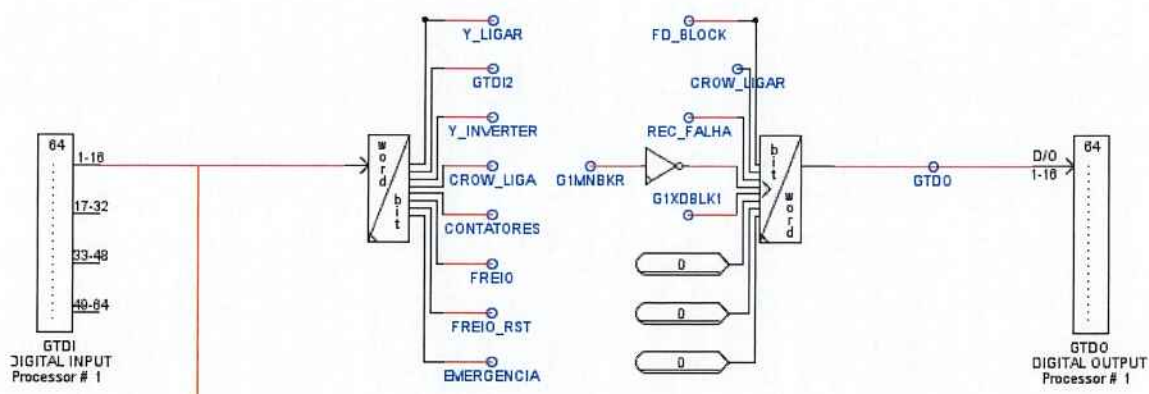


Figura 42: Da esquerda para direita, entradas e saídas digitais do RTDS

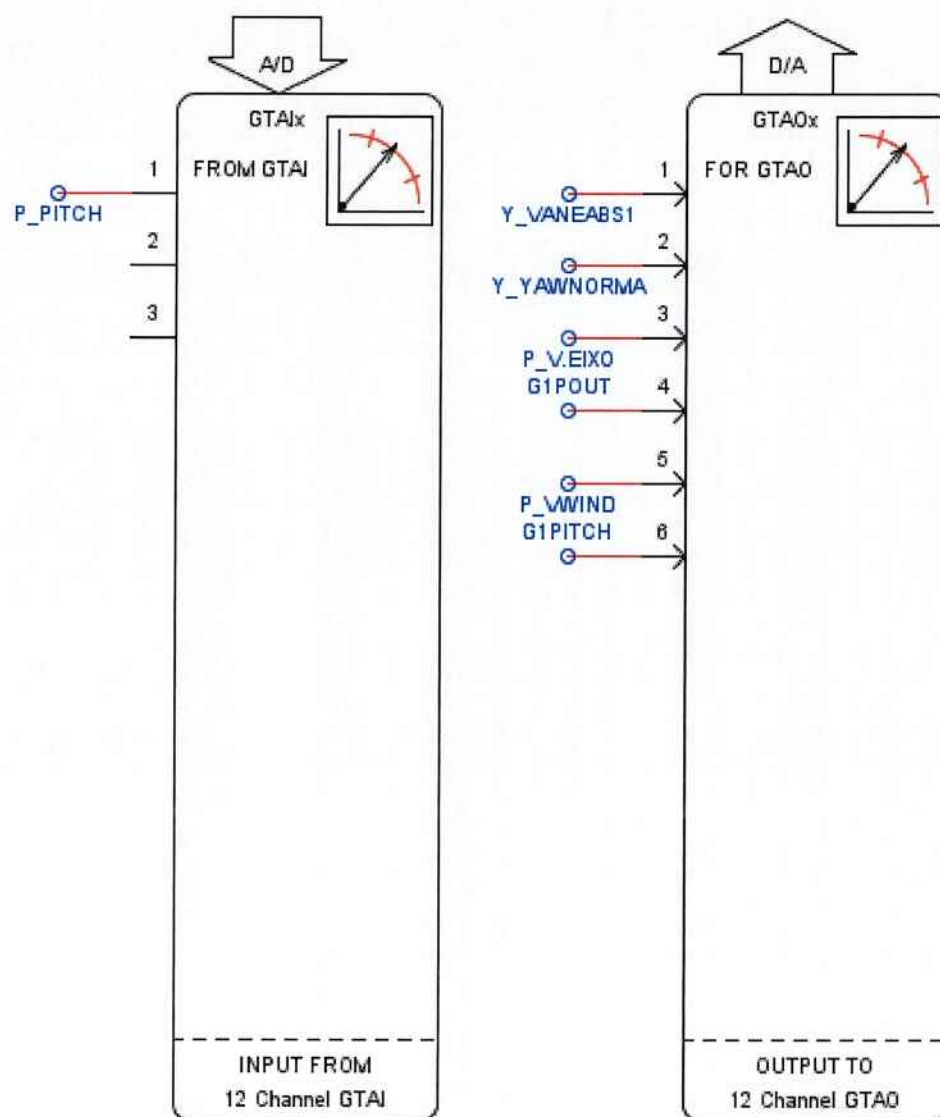


Figura 43: Da esquerda para direita, entradas e saídas analógicas do RTDS

As variáveis expostas nas figuras 42 e 43 estão na Tabela 1. A Tabela 1 foi criada de forma que os sinais estejam referidos ao cRIO, assim, uma saída/entrada é uma saída/entrada do cRIO.

Tabela 1: Lista de entradas e saídas para o controlador

Sinal	Entrada/Saída	Tipo	Descrição
Y_VANEABS1	Entrada	Analógica	Posição do Vento
Y_YAWNORMAL	Entrada	Analógica	Posição da Nacele
P_VEIXO	Entrada	Analógica	Velocidade do Eixo do Gerador
G1POUT	Entrada	Analógica	Potência gerada para rele elétrica
P_VWIND	Entrada	Analógica	Velocidade do vento
G1PITCH	Entrada	Analógica	Ângulo de <i>pitch</i> das pás
P_PITCH	Saída	Analógica	Diferença entre ângulo de <i>pitch</i> de referência e o real
FD_BLOCK	Entrada	Digital	Lógica do <i>setpoint</i> de velocidade do eixo
RECFALHA	Entrada	Digital	Botão para liberar falha
G1MNBKR	Entrada	Digital	Sinal de trip da proteção elétrica
G1XDBLK1	Entrada	Digital	Estado do disjuntor da rede elétrica
Y_LIGAR	Saída	Digital	Comando do motor que rotaciona a nacele
Y_INVERTER	Saída	Digital	Lógica de inversão da rotação da nacele
CONTADORES	Saída	Digital	Comando de fechamento do disjuntor da rede
FREIO	Saída	Digital	Acionamento do freio mecânico
FREIO_RST	Saída	Digital	Desarme do freio mecânico
EMERGENCIA	Saída	Digital	Lógica que aciona sirene de emergência

10. Programação do cRIO

Após apresentar os controles de ângulo das pás e apontamento do gerador no capítulo acima, descreve-se nessa seção toda a programação de sequenciamento de partida, parada e parada de emergência, além de alguns outros detalhes. O código para realizar esse sequenciamento é uma máquina de estados construída com os princípios apresentados no capítulo 8.

10.1 Programação da Máquina de Estados

Os estados e as transições criados no programa estão na figura 44.

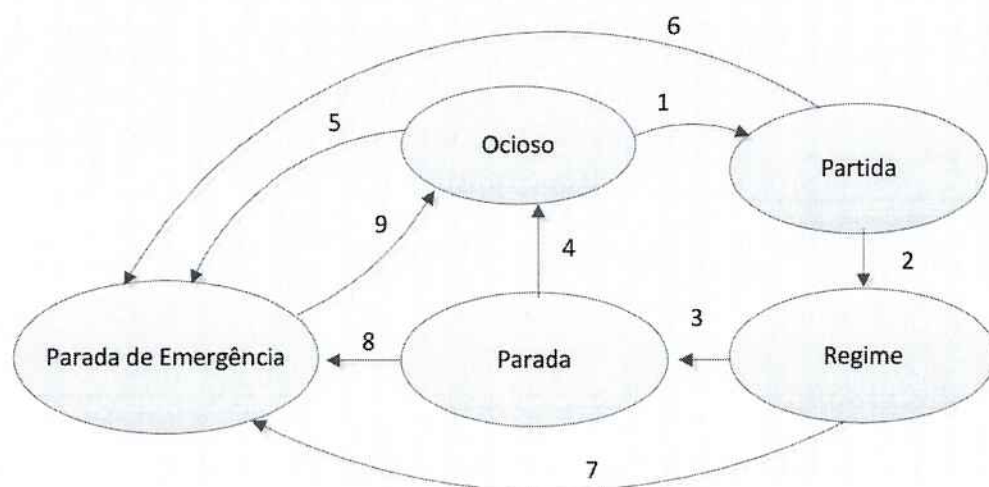


Figura 44: Máquina de Estados para comando do aerogerador

O estado Ocioso é o inicial, ele é melhor representado pela tabela 2.

Tabela 2: Estado Ocioso

Estado	Características do Estado	Condições de Transição
Ocioso	G1MXSPD = -1,2 Correção do yaw na parada = 1,25 SetFreio = True ResetFreio = False ResetContatores = True Sirene Emergência = False	Transição 1
		Botão de Partida = True
		Transição 5
		G1MNBKR = True Botão de Emergência = True

Nesse estado o aerogerador está parado com o freio aplicado (SetFreio = True), colocado a uma direção de 90° em relação ao vento (Correção do yaw na parada = 1,25), com o disjuntor da rede aberto (ResetContatores = True). Dessa forma, nada ocorre até haja o comando de partida pelo usuário. Outra forma de sair do estado Ocioso é a proteção elétrica da rede alertar o cRIO que houve pedido de abertura de disjuntor.

O código desenvolvido foi isolado em outra VI para facilitar a visualização do mesmo, essa VI está na figura 45. O código utiliza a estrutura de *shift register* para informar a iteração seguinte qual estado deve-se transitar. O cluster dos botões de entrada passa por uma VI de identificação de qual botão foi apertado, então, ela gera um número de acordo, esse número sinaliza um caso dentro do Enum, indicando o próximo estado. Foi necessário criar um biestável para acionar os contatores em qualquer estado que o sistema esteja, já que o *flip flop* se encontra fora da estrutura condicional. Caso há proteção elétrica atue no sistema, o código ignora a entrada dos botões e escreve no *shift register* à direita o valor 'Parada de Emergência'.

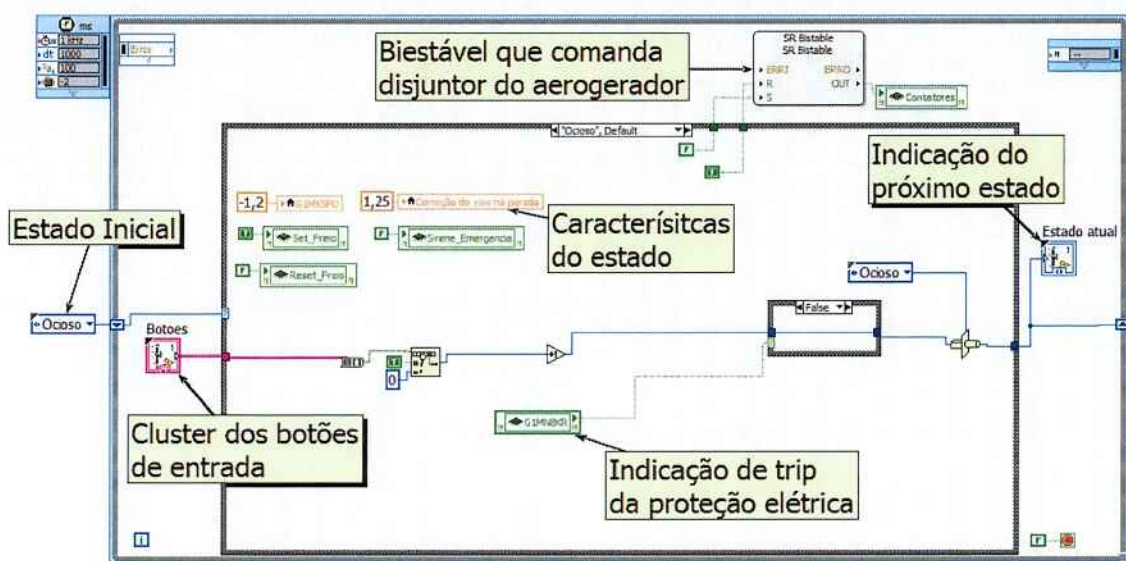


Figura 45: Estado Ocioso

Assim que o botão de Partida é apertado, a máquina realiza a Transição1 da figura 44.

O estado de Partida está representado na tabela 3.

Tabela 3: Estado Partida

Estado	Características do Estado	Condições de Transição
Partida	G1MXSPD = 1,2 Correção do yaw na parada = 0 SetFreio = False ResetFreio = True ResetContatores = False SetContatores = True Sirene Emergência = False	Transição2
		SetContatores = True
		Transição6
		G1MNBKR = True Botão de Emergência = True

A tabela 3 representa as características do estado, no entanto, existe uma lógica para que as mesmas aconteçam, não é como no estado Ocioso. Isso é

mais evidente ao constatar que $\text{SetContatores} = \text{True}$ é uma característica de transição, mas está colocada em característica do estado também. O código está na figura 46.

A subVI de análise do vento, gera um sinal digital alto se o vento estiver entre 6 e 24 m/s. Esse sinal aciona um timer que após 5 segundos também gera um sinal alto, indicando que o vento deve se manter e, logo, o sistema pode iniciar. Se o sistema pode iniciar, G1MXSPD assume o valor de 1,2 p.u para que o *pitch* possa atuar, solta-se o freio e espera-se o rotor acelerar. Em momento oportuno, decidido pela velocidade atingida, fecha-se o disjuntor (contator) da rede, criando torque contrário ao movimento do rotor. Contudo, a aplicação na região exata de velocidade ainda permite a partida do aerogerador. Testes empíricos mostraram que isso ocorre de acordo com a velocidade do vento incidente e é proporcional a ela no modelo. A velocidade aceitável em p.u para fechar o disjuntor está em torno de 8,5% do valor do vento até o valor de 12 m/s, depois desse valor, mantém-se o fechamento para velocidade de 1 p.u.

O sistema transita de estado assim que o disjuntor é fechado, indo para Regime. Abaixo, resume-se o sequenciamento apresentado.

- 'Correção do yaw na parada' assume valor nulo: nacele aponta para direção do vento;
- Nacele alinhada com o vento libera o freio;
- Vento entre 6 e 25 m/s por mais de 5 segundos junto à condição anterior libera o freio e G1MXSPD assume 1,2;
- Após a velocidade do eixo chegar ao esperado fecha-se o disjuntor e o aerogerador vai para condição de regime.

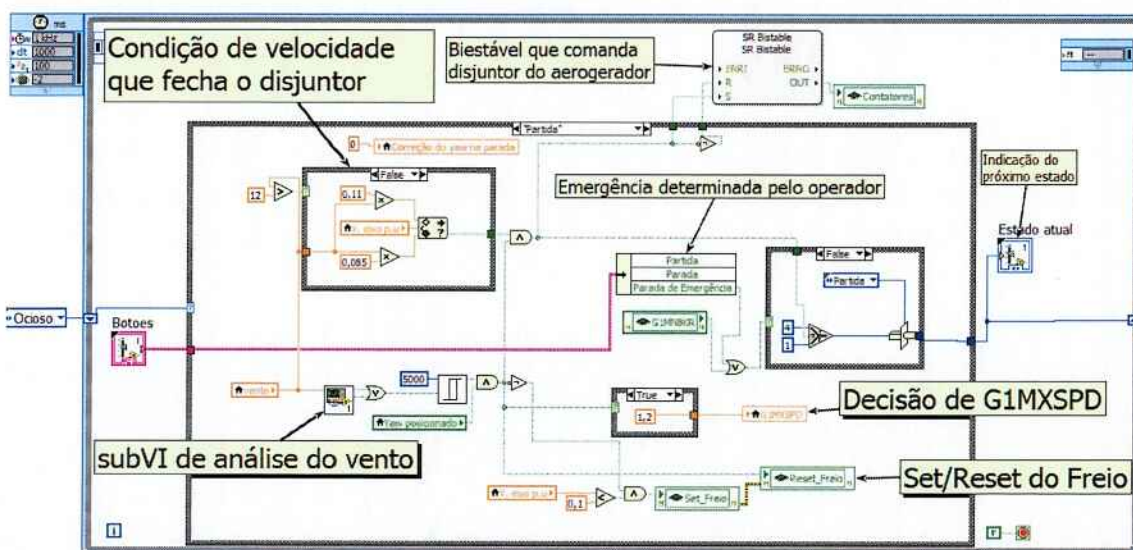


Figura 46: Código da partida

O estado Regime está apresentado na tabela 4:

Tabela 4: Estado Regime

Estado	Características do Estado	Condições de Transição
Regime	G1MXSPD = 1,2 Correção do yaw na parada = 0 SetFreio = False ResetFreio = True ResetContatores = False SetContatores = True Sirene Emergência = False	<u>Transição3</u>
		Vento fora do intervalo = True Botão de Parada = True
		<u>Transição7</u>
		G1MNBKR = True Botão de Emergência = True

O estado Regime apenas fica conferindo se o vento está acima do valor de *cut-off* ou abaixo do valor de *cut-in*. Caso isso ocorra ou por determinação do operador, o sistema vai para a condição de parada. Obviamente, esse estado fica atento às condições que podem levar a uma parada de emergência: sinalização da proteção elétrica ou determinação do operador.

O código representativo desse estado está na figura 43. Na figura 48, está o código real. Como as subVIs que comandam o ângulo das pás e o apontamento da nacele estão fora do teste condicional, elas sempre estão sendo executadas devido à estrutura de paralelismo do LabView.

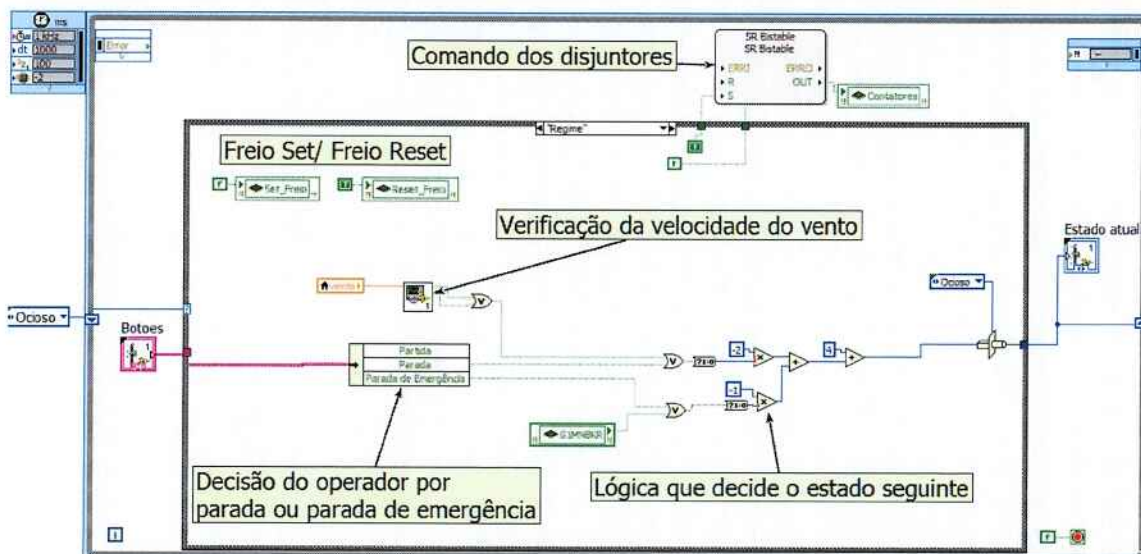


Figura 47: Código do estado Regime

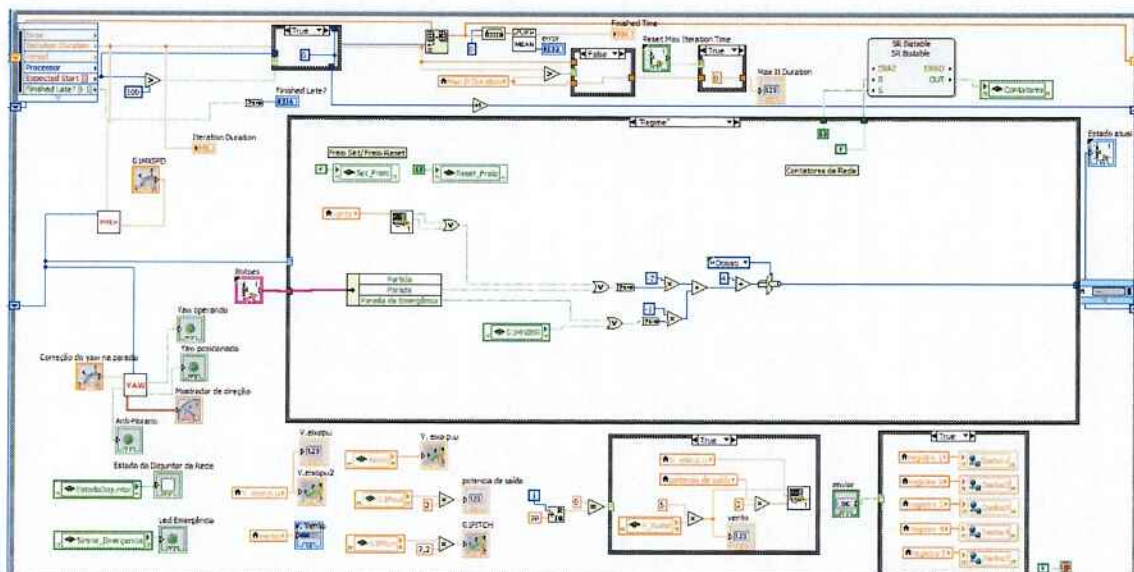


Figura 48: Estado Regime no código completo

As características do Estado Parada estão representadas na tabela 5.

Tabela 5: Estado Parada

Estado	Características do Estado	Condições de Transição
Parada	G1MXSPD = 0 Correção do yaw na parada = 1,25 SetFreio = True ResetFreio = False ResetContatores = True SetContatores = False Sirene Emergência = False	Transição4
		Yaw Posicionado = True
		Transição8
		G1MNBKR = True Botão de Emergência = True

Da mesma forma que o estado Partida, esse estado necessita de um sequenciamento já que, por exemplo, não se aplica freio ao rotor da turbina enquanto o ângulo de *pitch* e a velocidade não atingirem valores seguros, ou seja, a turbina não pode estar extraindo energia do vento e a velocidade deve ser baixa. O código desse estado está na figura 49.

Nessa condição, G1MXSPD assume valor nulo, como o rotor tem velocidade (Veixo), há um erro positivo no controlador e, portanto, o ângulo de *pitch* começa a aumentar. A eficiência do aerogerador diminui rapidamente com a atuação do *pitch*, assim, assim que a velocidade do aerogerador for menor que 0,1 p.u ou o ângulo de *pitch* atingir valor máximo, 'Correção do yaw na parada' assume 1,25, levando o aerogerador a ficar de lado para o vento incidente. Logo que isso ocorrer, o sistema volta ao estado ocioso, aplica o freio e fica estático. Durante a parada, o contator é aberto assim que o sistema registrar que a potência entregue para rede for negativa.

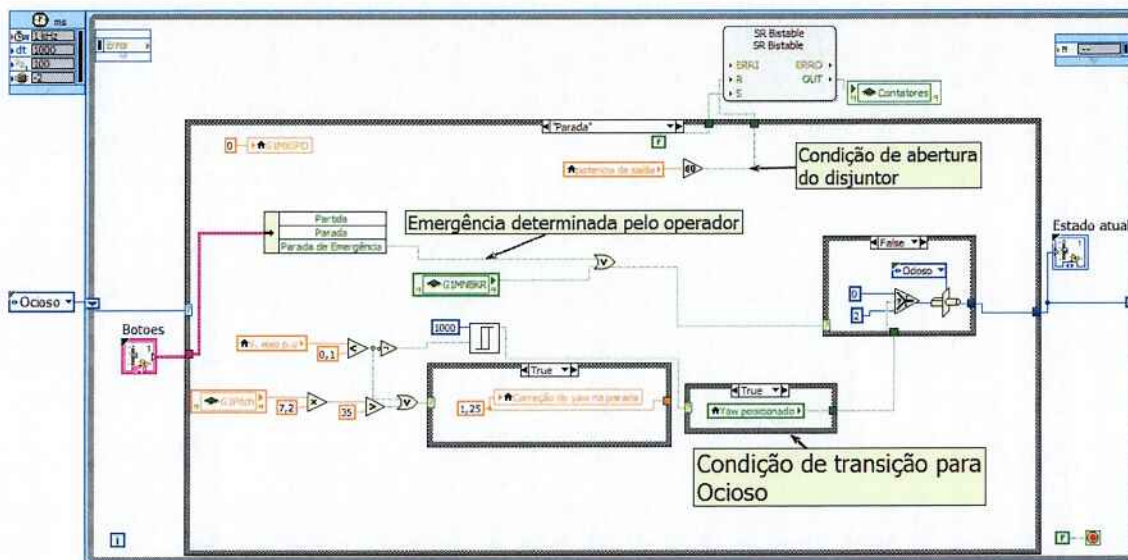


Figura 49: Código do estado Parada

Por último, apresenta-se a tabela do estado Parada de Emergência.

Tabela 5: Estado Parada de Emergência

Estado	Características do Estado	Condições de Transição
Parada de Emergência	G1MXSPD = não importa Correção do yaw na parada = não importa SetFreio = True ResetFreio = False ResetContatores = True SetContatores = False Sirene Emergência = True	Transição9
		Reconhecimento de Falha

Nesse estado, A ideia é que o sistema pare o mais rápido possível e permaneça imóvel. Sendo assim, o controle proporcional-integral deixa de atuar (como já explicado no capítulo 9), então G1MXSPD fica com o valor anterior, mas não importa. A 'Correção do yaw na parada' também não importa já que o controle de apontamento é bloqueado nessa situação. Assim que o sistema passa a consumir energia da rede, o disjuntor é aberto. Também ocorre uma aplicação de um sinal digital de emergência para acionamento de uma sirene no parque eólico. O código está na figura 50.

Por fim, só é possível sair desse estado se for feito reconhecimento local da falha através de um botão e se não houver sinalização de emergência da proteção elétrica.

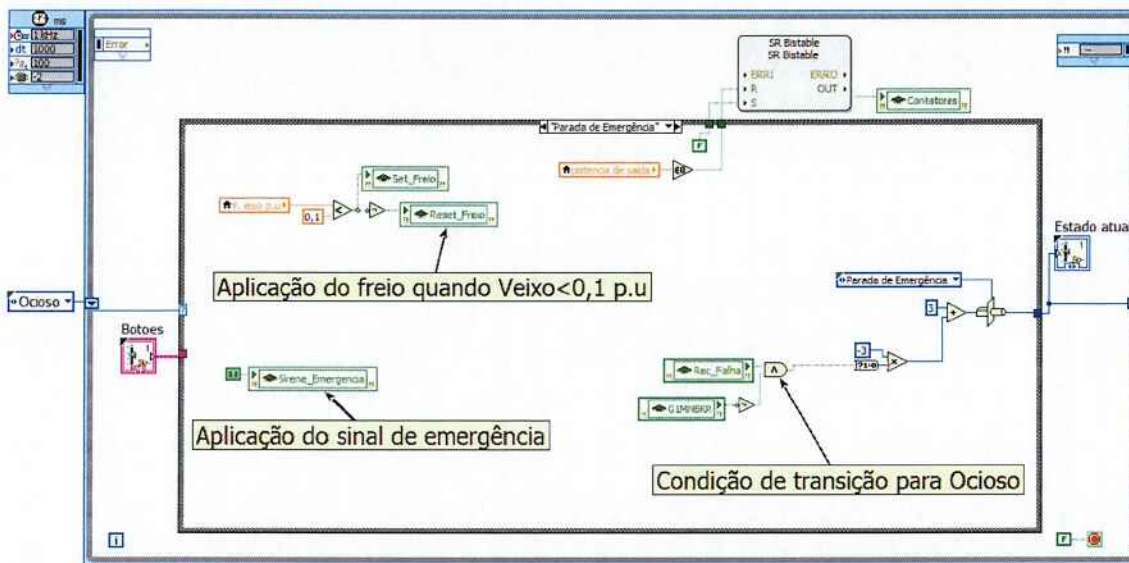


Figura 50: Código do estado Parada de Emergência

10.2 Programação do Oscilógrafo

Visando a aquisição de dados do controlador para eventuais análises de como o processo está ocorrendo, foram criadas *shared variables*, variáveis que podem ser acessadas por todas as Vis do projeto, para armazenar vetores de registros de dados. Antes de analisar o código que realiza essa função, a figura 51 ilustra como a mesma está presente no *front panel*. Na seção nomeada "Registro" há os botões "registrar" e "enviar". Para cada uma das variáveis 1a 5 escolhe-se entre direção do vento, direção da nacele, ângulo de ataque das pás, velocidade do eixo da máquina e potência fornecida à rede, sendo a variável escolhida a que será registrada.



Figura 51: *Front Panel* da VI principal do programa

Na figura 52 aparece o código do botão registrar do *front panel* (registrar e enviar) e ele se conecta à estrutura condicional (função if em C). Dentro da condição comandada pelo botão “registrar” há as variáveis registro 1 a 5 e as variáveis que podem ser registradas (direção do vento, direção da nacela, ângulo de ataque das pás, velocidade do eixo da máquina e potência fornecida à rede). Esses blocos são conectados a uma subVI que vai adicionar ao *array* das variáveis registro .

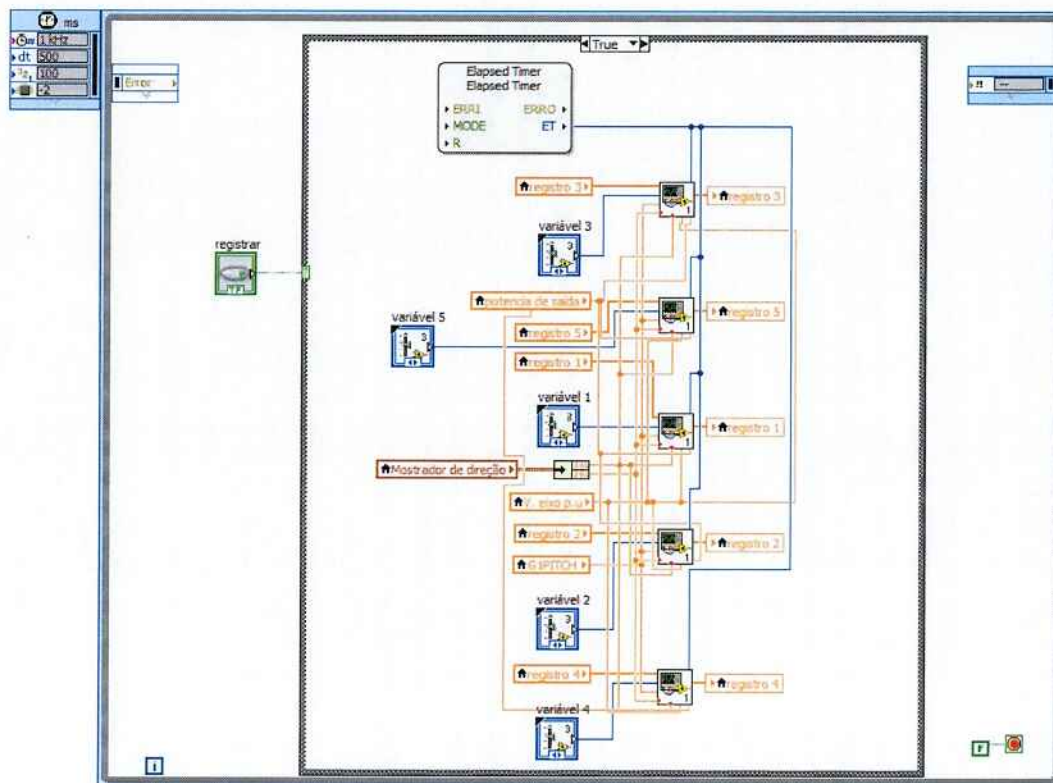


Figura 52: Variáveis a serem registradas entrando na VI do oscilógrafo

Na figura 53 está o código comandado pelo botão enviar. Ao ser pressionado o botão registra os vetores registro1 a registro 5, respectivamente nas variáveis Dados 1 a Dados 5. Dessa forma os dados poderão ser armazenados nas *Shared Variables*.

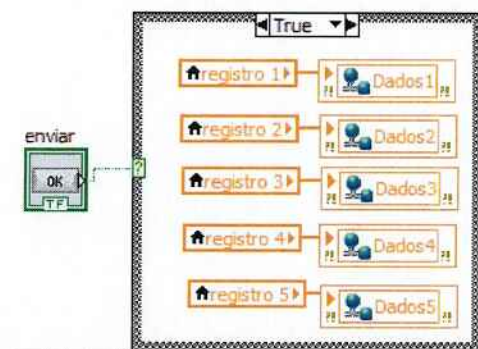


Figura 53: Detalhe do código de envio de dados

Na figura 54 está a Vi que registra cada dado medido ao vetor de registro.

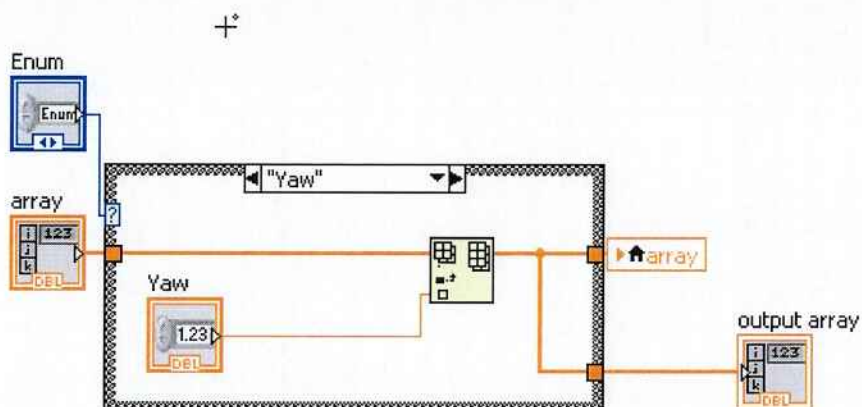


Figura 54: Código da subVI que faz o registro

Assim fica definido que o botão registrar armazena as variáveis em um vetor interno à VI e o botão enviar, transfere as informações desse vetor para uma *Shared Variable*.

Um novo VI deve ser executado para que o registro das shared variables vá para o computador. Esse VI, chamado oscilógrafo, está, no projeto, alocado no computador, como pode ser visto na figura 27. A única função dessa último VI é salvar os valores das variáveis dados 1 a 5 em um arquivo texto. A figura 55 mostra apenas o código que recebe a variável dados2, mas a VI recebe também os valores de dados1, dados3, dados4, dados5.

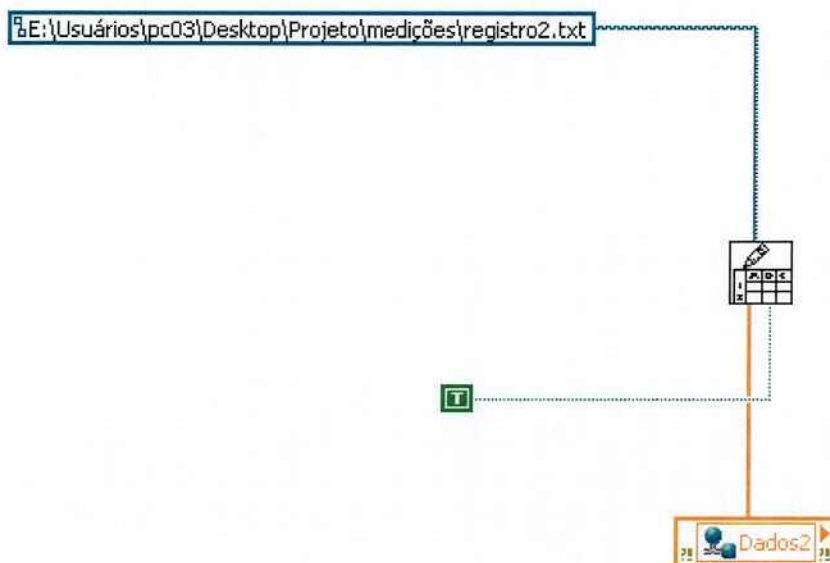


Figura 55: Código da VI que salva os dados recebidos no computador

10.3 Interface com o usuário

Para que um operador possa visualizar as principais variáveis do aerogerador e poder controlar os principais parâmetros da turbina, é necessária a criação de uma interface com o usuário. No caso da programação em LabVIEW, a interface é gerada simultaneamente com o desenvolvimento das VIs. No entanto, em cada tipo de controle exercido por cada VI surgem variáveis que não apenas não são de utilidade alguma para o operador, como também não devem ser alteradas pelo mesmo. Portanto, é necessário definir uma interface contendo apenas as variáveis de interesse para o operador, onde ele tenha o acesso às principais informações do aerogerador. Nesse contexto, conforme são criados os controles da turbina, está sendo criada uma interface de operação.

A interface está na figura 56 e foi dividida em 3 blocos: o bloco do ângulo de ataque das pás (*pitch*), o bloco do apontamento da nacele (*yaw*), um bloco de registro de dados e um bloco do estado de operação. No primeiro é possível definir a velocidade referência do eixo em regime, dado por G1MXSPD e visualizar a velocidade do eixo em pu (Vwindpu) e o ângulo de ataque (G1PITCH). No segundo bloco é possível visualizar o ângulo relativo entre o vento e o apontamento da nacele, além de ver se o motor de correção do ângulo da nacele está atuando ou não (ligar) e se o ângulo relativo está fora do range de atuação da correção (In Range?). O terceiro traz opções de variáveis a serem registradas e opção de registrá-las ou não. O último, por fim, traz comandos de partida, parada e parada de emergência do aerogerador e a informação sobre o estado atual da turbina.

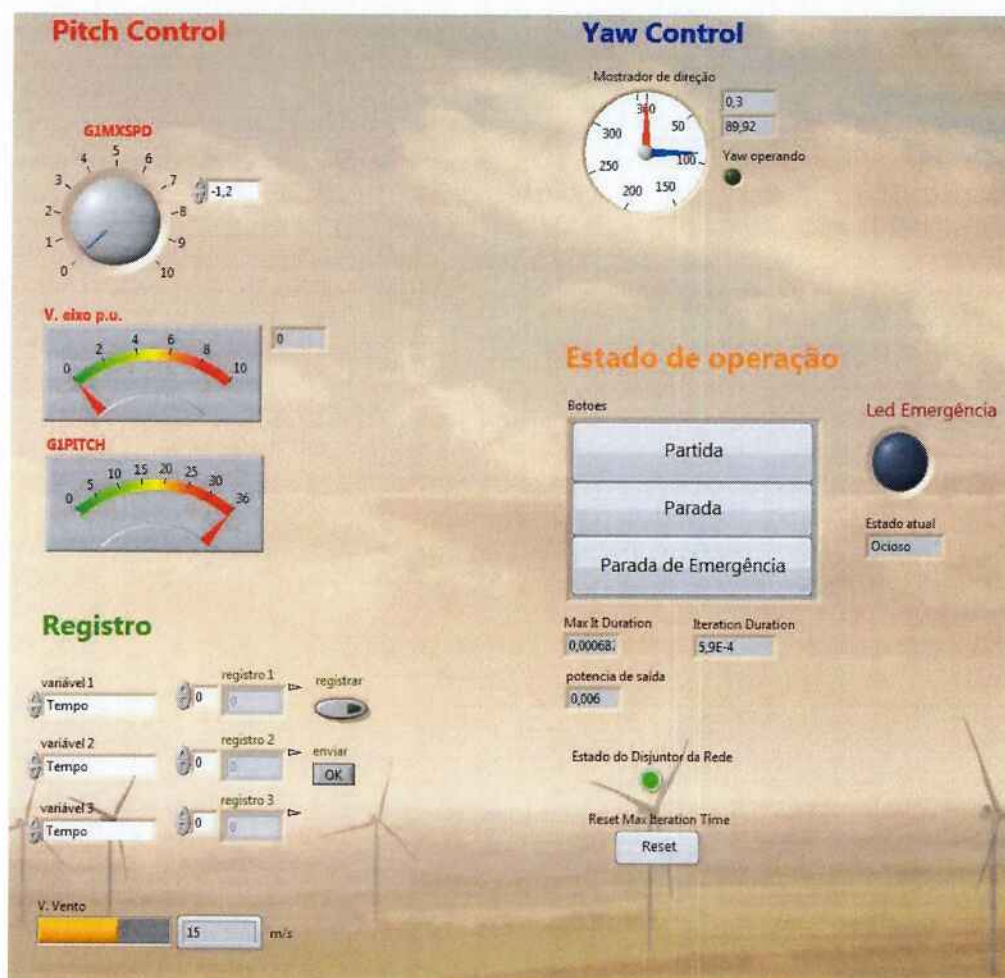


Figura 56: Interface Homem-Máquina

10.4 Conexões Elétricas

As conexões elétricas digitais e analógicas estão representadas nas figuras 57 a 60. No caso, foram utilizados chicotes com 10 cabos por chicote, assim, só essas conexões foram representadas. Tanto os cartões digitais e analógicos do controlador e do simulador possibilitam mais conexões.

Os cartões digitais de saída dos dois dispositivos requerem uma fonte externa para gerar tensão. A fonte de alimentação utilizada é a mesma do cRIO 9024, portanto, o nível de tensão é de 24V. Já os cartões analógicos tem outra particularidade.

O cartão analógico de entrada do RTDS aceita apenas entrada diferencial, obrigando o uso de um par de fios para transmissão de sinais. Mas o cartão analógico de saída do RTDS apresenta negativo comum para todas as saídas. Os níveis de tensão estão entre 10V e -10V.

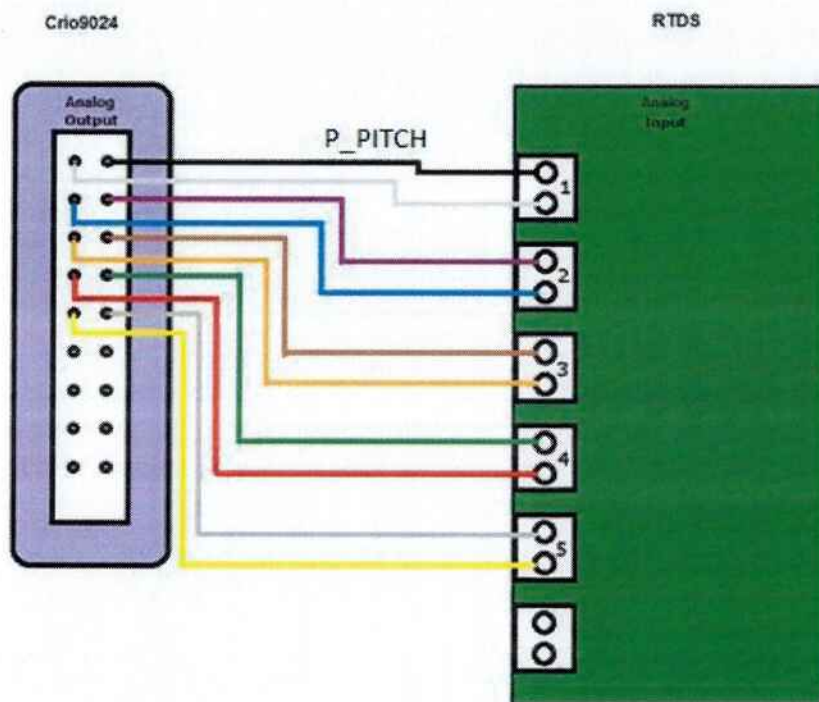


Figura 57: Saída analógica do controlador realizada por canais diferenciais

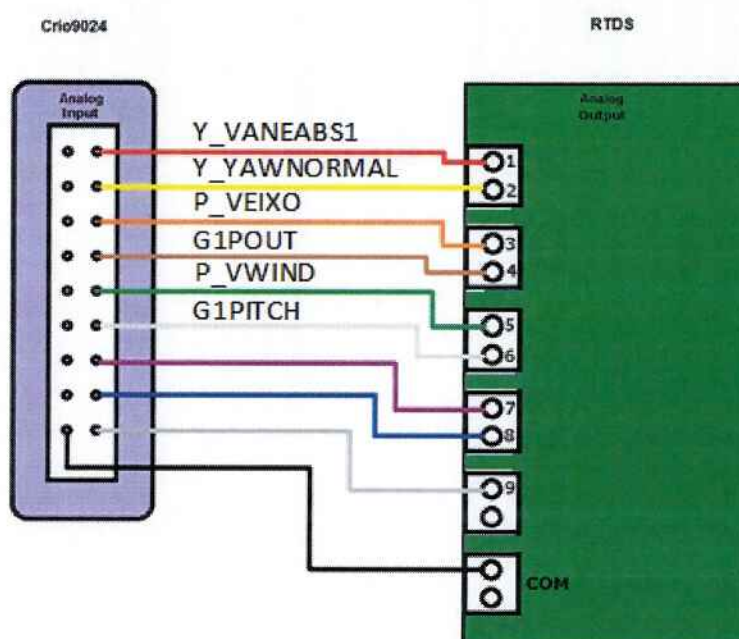


Figura 58: Entrada analógica do controlador realizada por canais com terra comum

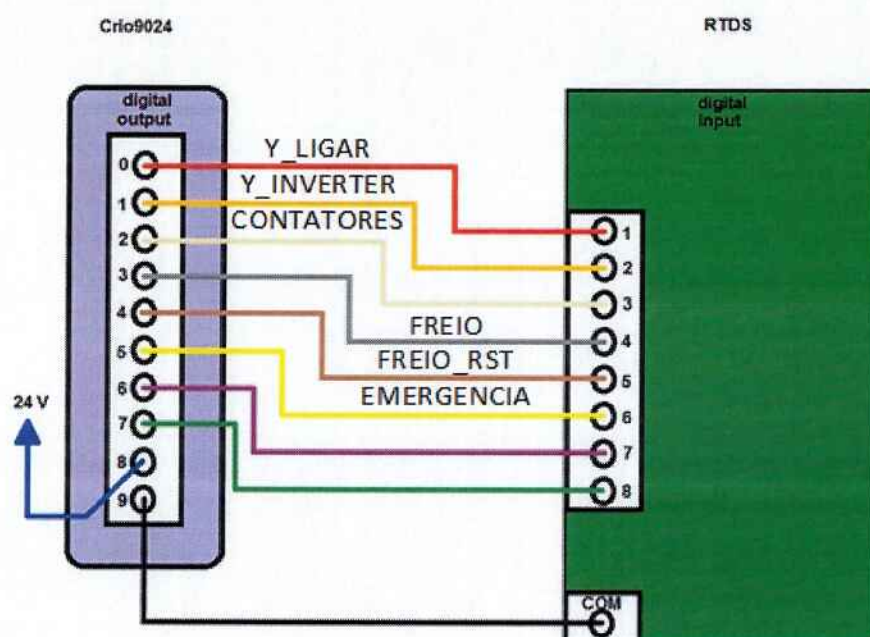


Figura 59: Saída digital do controlador

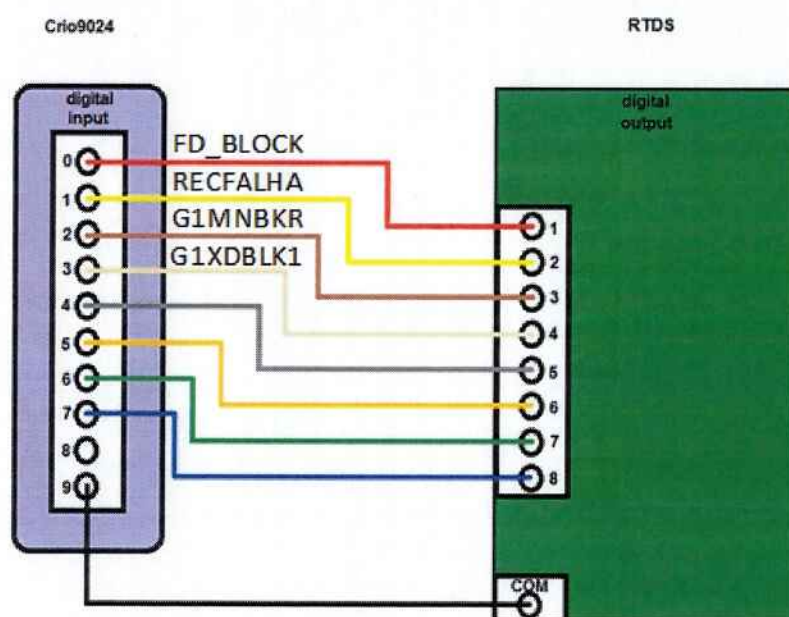


Figura 60: Saída digital do RTDS na qual não foi representada a alimentação externa do cartão

11. Conclusão

Foi avaliado no início do projeto que a geração de energia por meio da fonte eólica apresenta uma grande variedade de dificuldades técnicas, causadas principalmente pelo desafio da geração com uma velocidade variada do vento. Dada a importância dessa fonte de energia, especialmente pela preocupação ambiental e sua crescente utilização, é necessário buscar soluções eficazes para os problemas apresentados na sua utilização. A ideia principal desse projeto de formatura é tentar resolver parte desses empecilhos usando ferramentas de automação e controle para tornar viável e eficaz o uso do vento para geração de energia. Sendo o objetivo, não apenas implementar as soluções de automação, mas poder testar sua atuação da forma mais verossímil possível, por meio de um simulador em tempo real, dada a impossibilidade de testá-la com um protótipo físico. Foi também utilizado um controlador lógico programável como ferramenta para automatização do sistema.

Na busca dessas soluções, foi possível identificar e implementar parte das automações características de uma turbina eólica. A primeira delas foi o controle do ângulo de ataque das pás do aerogerador. Nessa automação foi verificada sua importância para manutenção da estabilidade da velocidade do eixo da máquina. É um mecanismo importante para evitar grandes distúrbios na geração e danos estruturais, devidos à alterações bruscas da velocidade do vento. Também é notável sua utilidade na prevenção de acidentes usando o ataque das pás para realizar paradas de emergência do aerogerador.

Dentre as automações implementadas, o controle de apontamento da nacele é responsável pelo correto posicionamento da área de captação do vento da turbina. Por meio do alinhamento da nacele com a velocidade do vento paralela ao solo, esse controle permite maximizar a geração de energia, tornando a turbina mais eficaz.

A criação de um oscilógrafo para recolhimento de dados da turbina também é importante para o bom funcionamento da geração. Para tanto foi criado, dentro do código do controlador, um oscilógrafo, capaz de recolher dados selecionados da turbina e enviá-lo a dispositivos externos ao controlador. Dessa forma, uma unidade remota pode fazer análise dos dados relevantes do aerogerador.

O sequenciamento de operação desse aerogerador é também parte do trabalho realizado nesse projeto. Para tanto, foi usada a estrutura de máquinas de estados, em que cada estado representa uma etapa da operação do aerogerador. Assim foram definidos os estados: ocioso, partida, regime, parada e parada de emergência. Com essa implementação foi possível verificar o correto funcionamento e identificação das formas de funcionamento da geração.

Nesse projeto também foi criada uma interface com o usuário. O controlador utilizado é fabricado pela National Instruments e por isso sua plataforma de programação é o LabView. Dessa forma, foram criados o que chamado de *Virtual Instruments*, ou Vis. São chamados assim por mimetizarem instrumentos reais de atuação, medição e sinalização, como botões, *displays* e leds. Assim sendo, os VIs permitem que a interface com o usuário seja criada automaticamente. Por meio desses Vis foi criada uma interface amigável, permitindo que o controle e a supervisão do aerogerador sejam feitos remotamente.

O sistema completo implementado se mostra uma ferramenta útil para avaliação do funcionamento de um aerogerador, desenvolver e verificar o impacto do sistema de automação numa turbina desse porte. Foi possível, sem uma turbina real, buscar soluções em automação para melhorar a segurança, a eficiência, e permitir o correto funcionamento do aerogerador.

O projeto ainda permitiu ampliar o conhecimento em diversas áreas do conhecimento da engenharia elétrica, como a geração eólica, máquinas elétricas, automação, sistemas de potência entre outras. Por meio de uma aplicação prática dos conhecimentos adquiridos foram observadas e superadas diversas dificuldades no desenvolvimento de projeto, dando uma noção mais realistas das dificuldades com que um engenheiro se depara na implementação de soluções.

Foi possível exercer as diversas responsabilidades da engenharia, especialmente no que diz respeito a gestão de projetos. O atendimento do cronograma, avaliação das disponibilidades de recursos, definição de objetivos realizáveis, sequenciamento do projeto, divisão de tarefas, documentação do trabalho e pesquisa de soluções na bibliografia existente sobre o assunto, são apenas alguns dos principais aspectos da gestão que foram desenvolvidos ao longo da disciplina.

A disciplina de Projeto de Formatura foi de grande ajuda para o desenvolvimento profissional dos alunos, futuros engenheiros, na medida em que permitiu o desenvolvimento de um projeto real, dentro da própria Universidade, de autoria dos alunos, amparados pelos professores. Foi testada a capacidade de solucionar problemas por meio de conhecimentos que a própria Escola forneceu aos alunos ao longo da graduação. Permitiu também aplicação prática dos conceitos da engenharia, sintetizando na disciplina os pré-requisitos para exercer a profissão.

12. Bibliografia

- [1] **Projeção da demanda de energia elétrica para os próximos 10 anos (2014-2023)**. Rio de Janeiro, Dezembro 2013. Retirado de www.epe.gov.br, acessado dia 27/02.
- [2] **Balanço Energético Nacional 2013**. Ano de base 2012. Retirado de www.ben.epe.gov.br, acessado dia 27/02.
- [3] **Energia Eólica 6**. Retirado de www.aneel.gov.br, acessado dia 27/02.
- [4] Gao, Q; Liu, C; Xie, B; Cai, X. **Evaluation of the mainstream wind turbine concepts considering their reliabilities**. Publicado na IET *Renewable Power Generation*, Junho 2012.
- [5] F. Runcos; R. Carlson; P. Kuo-Peng; H. Voltolini¹; N.J. Batistela. **Geração De Energia Eólica – Tecnologias Atuais E Futuras**. Retirado de www.ecatalog.weg.net, acessado dia 01/03.
- [6] Kang, Shang; Feng, Xue; Yongjie, Fang; Yuehai, Yu. **Comparative Simulation of Dynamic Characteristics of Wind Turbine Doubly-Fed Induction Generator Based on RTDS and MATLAB**. 2010 *International Conference on Power System Technology*.
- [7] Abad, G. **Doubly Fed induction machine: modeling and control for wind energy generation applications**. Hoboken, NJ: IEEE Press, c2011. ISBN 9780470768655.
- [8] RSCAD Technologies. **Real Time Digital Simulator Tutorial Manual (RSCAD Version)**. Março 2012. Disponível no software do simulador.
- [9] Imagem retirada de [http://www.behance.net/gallery/Acciona-Energy-\(SCADA\)/560816](http://www.behance.net/gallery/Acciona-Energy-(SCADA)/560816)
- [10] Miller, Nicholas W.; Price, William W.; Sanchez-Gasca, Juan J. **Dynamic Modeling of GE 1.5 and 3.6 Wind Turbine Generators**. 27 de Outubro de 2003. Versão 3.0
- [11] Ragheb, Magdi; **Control of Wind Turbines**. Apostila do curso Wind Power System. 06/05/2009. <http://mragheb.com/>, acessado dia 30/03/2014
- [12] http://en.wikipedia.org/wiki/Yaw_system, acessado dia 30/03/2014
- [13] Gaiser, Brad D; Hopkins, Andrew; Belen, Fred; Major, Jo S. **Yaw Control Optimization**. <http://windsystemsmag.com>, acessado dia 30/03/2014
- [14] www.ni.com, acessado dia 01/05/2014.

[15] **Operating instructions and specifications CompactRIO NI cRIO-9024.** Retirado de www.ni.com, acessado dia 01/05/2014.

[16] **Wind Turbine Driven Doubly Fed Induction Generator.** Descrição do modelo de simulação de DFIG do RSCAD. Disponível na seção *Help* do programa.

[17] **NI LabVIEW for CompactRIO Developer's Guide - Recommended LabVIEW Architectures and Development Practices for Control and Monitoring Applications.** Disponível em www.ni.com, acessado 20/10/2014.

[18] Hwas, Abdulhamed; Katebi, Reza. **Wind Turbine Control Using PI Pitch Angle Controller.** Março 2012. IFAC Conference on Advances in PID Control.

[19] E. Johnson, Kathryn; Y. Pao, Lucy; J. BALAS, Mark; J. Fingersh, Lee. **Control of Variable-Speed Wind Turbines. Standard and adaptive techniques for maximizing energy capture.** Junho 2006. IEEE Control System Magazine.