UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE FÍSICA DE SÃO CARLOS

Felipe Bessa Coelho

Controlador para bobinas de *shimming* via HTTP

São Carlos

2016

Felipe Bessa Coelho

Controlador para bobinas de *shimming* via HTTP

Dissertação apresentada ao Programa de Pós-Graduação em Física do Instituto de Física de São Carlos da Universidade de São Paulo para obtenção do título de Mestre em Ciências

Área de concentração: Física Aplicada
Opção: Física Computacional
Orientador: Alberto Tannús

Versão Corrigida

(versão original disponível na Unidade que aloja o Programa)

São Carlos

2016

AUTORIZO A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTE TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Ficha catalográfica revisada pelo Serviço de Biblioteca e Informação do IFSC,
com os dados fornecidos pelo(a) autor(a)

Coelho, Felipe Bessa

Controlador para bobinas de shimming via HTTP /
Felipe Bessa Coelho; orientador Alberto Tannús -
versão corrigida -- São Carlos, 2016.
66 p.

Dissertação (Mestrado - Programa de Pós-Graduação em
Física Computacional) -- Instituto de Física de São
Carlos, Universidade de São Paulo, 2016.

1. Ressonância magnética. 2. Shimming. 3. HTTP. I.
Tannús, Alberto, orient. II. Título.

Agradecimentos

Ao Prof. Dr. Alberto Tannús pela paciência e liberdade para condução do projeto. Pela confiança e apoio para resolver problemas tanto dentro como fora do laboratório, sem os quais não teria sido possível concluir este projeto.

Ao Dr. Mateus J. Martins e ao Dr. Edson L. G. Vidoto pelos vários ensinamentos e discussões acerca do projeto.

Aos demais amigos do CIERMag pelo trabalho desenvolvido por cada um e suas contribuições ao projeto. Pelas discussões que moldaram este trabalho à forma que tem hoje, pelos momentos de descontração e por toda a ajuda que deram para que pudesse concluir o projeto em tempo: Pizetta, Danilo, Pedro, Maurício, Douglas, Teixeira, Lucas, Rafael, Ricardo, Carlinhos, Tiago, Gustavo, Mariana, Gélia, Nilzeli e Patrícia.

Aos meus pais pelo apoio e incentivo irrestrito aos estudos na graduação e pós-graduação, sem os quais não teria o conhecimento necessário para chegar até aqui.

À Tamires, por estar ao meu lado em todos os momentos, compartilhando tanto dos progressos quanto das dificuldades.

Às agências de fomento FAPESP:2005/5.6663-1 e CNPQ:137.503/2012-1, FINEP e FNS.

"Do or do not. There is no try."

Yoda

Resumo

COELHO, F. B. **Controlador para bobinas de *shimming* via HTTP**. 2016. 66 p. Dissertação (Mestrado em Ciências) - Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 2016.

A Ressonância Magnética (RM) é uma técnica extremamente promissora para diversas áreas do conhecimento, desde seu uso conhecido em medicina até aplicações na indústria a fim de melhorar a eficiência de diversos processos. Este projeto trata do desenvolvimento de uma das partes do Espectrômetro Digital de RM desenvolvido pelo Centro de Imagens e Espectroscopia *in vivo* por Ressonância Magnética (CIERMag), o controlador de bobinas de *shimming*. O ajuste de *shimming* oferece melhorias significativas no resultado de experimentos e é parte integral de qualquer sistema de RM tanto por imagens como por espectroscopia. Foi idealizado um equipamento para estabelecer as correntes adequadas em bobinas de *shimming* que pode ser acessado tanto local como remotamente, permitindo a integração com outros subsistemas do Espectrômetro Digital. O equipamento consiste de *hardware* específico para estabelecer e analisar as correntes estabelecidas em bobinas de *shimming* e também de *software* adequado. Este projeto estabelece as diretrizes de funcionamento do controlador de bobinas de *shimming* e como ele interage com outros equipamentos.

Palavras-chave: Ressonância magnética. *Shimming*. HTTP.

Abstract

COELHO, F. B. **Shimming coil controller over HTTP**. 2016. 66 p. Dissertação (Mestrado em Ciências) - Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 2016.

Magnetic Resonance is an extremely promising technique to many areas of knowledge, from its use in medicine to industrial applications where it can improve the efficiency of different processes. This project describes the development of one element of the Digital Spectrometer being developed by CIERMag: the shimming coil controller. Shimming adjustments in magnetic resonance improves experiments' results significantly and is an integral part of any magnetic resonance system, from imaging to spectroscopy. A system as developed to establish the proper currents in shimming coils that can be accessed both locally and remotely, easing the integration with other subsystems of the Digital Spectrometer. The controller consists of both custom built hardware and software to establish and analyze the currents in shimming coils. This project also establishes the controller requirements and how it interacts with other systems.

Keywords: Magnetic resonance. Shimming. HTTP.

Lista de ilustrações

Figura 1 – Exemplo do processo de correção de inomogeneidade de campo magnético.	21
Figura 2 – Volume de interesse dentro de um magneto. As linhas de campo na região central, correspondente ao campo B_0 , são paralelas ao eixo Z . Costuma-se falar sobre o volume de interesse como uma esfera, aqui representada com diâmetro d .	23
Figura 3 – Diagrama do controlador.	27
Figura 4 – <i>Kit</i> de desenvolvimento PIC Ethernet Starter Kit.	28
Figura 5 – O computador Raspberry Pi.	29
Figura 6 – Erro médio de cada <i>Digital/Analog Converter</i> (Conversor Digital/Analógico) (DAC) obtido no primeiro protótipo. O código de entrada corresponde ao valor enviado via SPI para cada DAC.	31
Figura 7 – Polaridade e fase do <i>clock</i>	32
Figura 8 – Propagação do sinal em dispositivos SPI em <i>daisy chain</i> . Cada cor representa um comando enviado a um dispositivo SPI, e como ele trafega por todos os dispositivos até chegar em seu destino.	33
Figura 9 – Estrutura interna do controlador.	35
Figura 10 – Dispositivos conectados por SPI.	36
Figura 11 – Exemplo de requisição <i>Hypertext Transfer Protocol</i> (HTTP)	36
Figura 12 – Parte do diagrama esquemático do projeto do controlador, mostrando a ligação entre o Raspberry Pi e os módulos DAC e <i>Analog/Digital Converter</i> (Conversor Analógico/Digital) (ADC).	39
Figura 13 – Exemplo de montagem impossível com uma única camada. Não há como manter as duas trilhas coloridas no mesmo plano da placa de circuito impresso.	40
Figura 14 – Representação de camadas internas e uma via em uma placa de circuito impresso de múltiplas camadas.	41
Figura 15 – Trecho com erro de <i>Design Rules Checking</i> (DRC) exibido pelo OrCAD PCB Editor. A distância entre as duas trilhas apresentadas é menor do que o menor valor permitido.	42
Figura 16 – Duas das camadas usadas para produção do módulo DAC do controlador.	43
Figura 17 – Processo de isolamento de trilhas com a fresadora.	43
Figura 18 – A fresadora ProtoMat S62.	44
Figura 19 – Tanque com solução de cloreto cúprico para corrosão da placa de circuito impresso.	45
Figura 20 – Processo fotográfico para revelação da imagem da camada.	46
Figura 21 – Corrosão do cobre no processo fotográfico.	46
Figura 22 – Três tipos de vias frequentemente usadas. (a) via regular (b) via enterrada (<i>buried via</i>) (c) via cega (<i>blind via</i>).	47

Figura 23 – A estação de metalização Contac RS	47
Figura 24 – Representação do processo de deposição galvânica.	48
Figura 25 – Etapas do processo de metalização de furos.	49
Figura 26 – Placas produzidas externamente para os módulos de DAC e ADC.	50
Figura 27 – Perfil de temperatura para <i>reflow soldering</i>	51
Figura 28 – Gabaritos para aplicação de solda testados.	52
Figura 29 – Exemplo de tarefa idempotente do Ansible.	55
Figura 30 – Diagrama de pacotes do <i>software</i> do controlador.	57
Figura 31 – Processo de substituição dos comandos em cada um dos módulos DAC.	59
Figura 32 – Emissão de múltiplos comandos para alteração de dois canais em um único módulo DAC.	60
Figura 33 – Placas de circuito impresso já populadas.	61
Figura 34 – O controlador de bobinas de <i>shimming</i> com todos os módulos.	62

Lista de tabelas

Tabela 1 – Equações de campo para harmônicos esféricos de primeira e segunda ordem	23
---	----

Lista de Abreviações

A/D Analógico/Digital

ADC *Analog/Digital Converter* (Conversor Analógico/Digital)

API *Application Programming Interface*

CIERMag Centro de Imagens e Espectroscopia *in vivo* por Ressonância Magnética

D/A Digital/Analógico

DAC *Digital/Analog Converter* (Conversor Digital/Analógico)

DHCP *Dynamic Host Configuration Protocol*

DRC *Design Rules Checking*

HTTP *Hypertext Transfer Protocol*

HTTPS *HTTP Secure*

IP *Internet Protocol*

JSON *Javascript Object Notation*

LSB *Least Significant Bit*

MISO *Master Input Slave Output*

MOSI *Master Output Slave Input*

PADCT Programa de Apoio ao Desenvolvimento Científico e Tecnológico

ppm partes por milhão

PXE *Preboot eXecution Environment*

RM Ressonância Magnética

RMN Ressonância Magnética Nuclear

RPP *Reverse Pulse Plating*

SCLK *Serial Clock*

SMD *Surface Mount Devices*

SPI *Serial Peripheral Interface*

SS *Slave Select*

SUS Sistema Único de Saúde

TCP *Transmission Control Protocol*

TLS *Transport Layer Security*

ToRM Tomógrafo de Ressonância Magnética

UDP *User Datagram Protocol*

UV Ultravioleta

WSGI *Web Server Gateway Interface*

YAML *Yet Another Markup Language*

Sumário

1	Introdução	19
1.1	Contextualização	19
1.2	Ressonância magnética	19
1.2.1	Princípios de Ressonância Magnética	20
1.2.2	Limitações	20
1.3	<i>Shimming</i>	20
1.3.1	Ajuste automático	22
1.4	O projeto Tomógrafo de Ressonância Magnética (ToRM)	23
1.5	Objetivo	24
2	Controlador de bobinas de <i>shimming</i>	25
2.1	O controlador	25
2.1.1	Abordagem	26
2.1.1.1	PIC	28
2.1.1.2	Raspberry Pi	29
2.1.2	Conversores A/D e D/A	29
2.1.3	SPI	30
2.1.4	Ethernet e <i>layout</i> da rede	34
2.1.5	HTTP	35
2.1.5.1	JSON	37
2.2	Placa de Circuito Impresso	38
2.2.1	Projeto	38
2.2.2	Confecção	41
2.2.2.1	LPKF	42
2.2.2.2	Processo fotográfico	44
2.2.2.3	Metalização	45
2.2.2.4	Fabricação externa	48
2.2.2.5	Montagem	49
2.2.3	Módulos	52
2.3	Software	53
2.3.1	Interface <i>web</i> e linha de comando	53
2.3.2	Automação	53
3	Resultados	57
3.1	O <i>software</i> de controle	57
3.1.1	Acesso aos dispositivos físicos	57
3.1.2	Deteção dos módulos disponíveis	58
3.1.3	Criação e validação de comandos	59

3.2 O *hardware* do controlador 60

4 **Conclusão** **63**

4.1 Trabalhos futuros 63

Referências **65**

1 Introdução

1.1 Contextualização

A Ressonância Magnética ([RM](#)) é uma técnica de grande importância para diversas áreas do conhecimento, desde a medicina até a agricultura. Por ser uma técnica não invasiva e não destrutiva, seu uso permite diversos tipos de análise que não são possíveis com outras técnicas.

Um sistema de [RM](#), no entanto, é complexo e possui muitas partes que devem interagir entre si para se obter o resultado desejado. Com o avanço da tecnologia, novos equipamentos e mecanismos foram introduzidos, aumentando ainda mais a complexidade do sistema como um todo. Espera-se que novos equipamentos tenham resultados cada vez melhores, mas no âmbito da [RM](#), cada novo equipamento causa alterações na homogeneidade do campo magnético que podem prejudicar os resultados.

Como parte do projeto [ToRM](#), este trabalho tem como foco o desenvolvimento de um dos subsistemas de um espectrômetro digital de ressonância magnética produzido com tecnologia nacional: um controlador para bobinas de *shimming*.

1.2 Ressonância magnética

Quando núcleos de átomos são expostos a campos magnéticos com componentes estáticas e que variam com o tempo, é possível observar que parte da energia usada para produzir a variação do campo magnético é absorvida e re-emitida pelos núcleos dos átomos. A essa re-emissão é dado o nome de Ressonância Magnética Nuclear ([RMN](#)), frequentemente abreviada para apenas Ressonância Magnética.

A capacidade dos núcleos de reemitir a energia transferida a estes faz com que seja possível detectar não só a presença de determinados núcleos em uma substância como também sua densidade em um certo volume. Por sua vez, determinar estas quantidades — ainda que de forma relativa devido à dificuldade de obtenção de valores absolutos — é o que permite, em experimentos de [RM](#):

- Obter imagens do interior de uma amostra ou paciente
- Identificar moléculas presentes em substâncias

Uma propriedade de grande importância da técnica de Ressonância Magnética é que ela é não destrutiva. O objeto em estudo não precisa ser modificado para a análise, e a

radiação presente é não ionizante, portanto não age diretamente em tecidos vivos alterando sua constituição genética.

Essas características tornam a Ressonância Magnética uma técnica consideravelmente versátil, sendo utilizada em várias áreas como física e química, medicina, agricultura, na indústria petrolífera e ciência dos materiais.

1.2.1 Princípios de Ressonância Magnética

A ressonância magnética ocorre devido à interação entre núcleos de átomos e campos magnéticos, interação essa que é fruto de duas propriedades intrínsecas de núcleos de átomos: o *spin* e o *momentum* angular. O *spin* induz no núcleo do átomo um momento magnético, que por sua vez interage com campos magnéticos externos. Quando uma amostra contendo uma quantidade de átomos suficientemente grande é submetida a um campo magnético externo, a magnetização total da amostra possui mesma direção e tem intensidade proporcional à intensidade do campo magnético externo.

Devido à dependência direta entre a magnetização e o campo magnético externo, pequenas variações locais no campo magnético externo são refletidas na magnetização, também localmente. Ao produzir variações conhecidas no campo magnético, é portanto possível recuperar informações de locais específicos da amostra, um processo chamado de *codificação espacial*.

1.2.2 Limitações

A ressonância magnética como um todo, porém, é extremamente suscetível a inhomogeneidades do campo magnético. Experimentos para obtenção de imagens geralmente requerem variações menores que 3 partes por milhão (ppm), enquanto que em espectroscopia é comum depender de homogeneidades menores que 1 ppm. (1)

Campos magnéticos tão homogêneos não são obtidos facilmente, pois praticamente quaisquer objetos presentes no entorno do campo magnético, assim como o próprio equipamento que o produz podem causar variações inesperadas.

1.3 *Shimming*

Para corrigir a inhomogeneidade inerente ao ambiente de realização do experimento, diversas técnicas são empregadas. Cada uma tem uma característica específica, e correções podem ser feitas tanto de forma física — nos equipamentos propriamente ditos — como de forma computacional — alterando como são geradas as formas de onda a ser aplicadas

nos equipamentos. De forma geral, todo procedimento para redução da inhomogeneidade de campo magnético recebe o nome de *Shimming*. (2)

O termo decorre do procedimento de correção da inhomogeneidade de campos magnéticos produzidos por magnetos com núcleos de ferro (*yokes*). As peças polares, responsáveis pela definição da região útil dos magnetos, eram acopladas aos *yokes* através de parafusos, e seu paralelismo era corrigido com finas cunhas de cobre (*shims*) deformadas à sua espessura ideal mediante o aperto daqueles parafusos. (3) Generalizou-se então o uso do termo *shim* como qualquer procedimento para correção de imperfeições de campo magnético.

Apesar de serem empregadas técnicas distintas para a correção de inhomogeneidades de campo magnético, todas se baseiam em produzir uma variação de campo magnético contrária a presente na região de interesse. A figura 1 mostra como é feita a correção de campo. Na figura 1a, tem-se um exemplo de intensidade de campo magnético em determinada direção. Para reduzir a variação no campo magnético, são gerados campos com diferentes intensidades, como na figura 1b. Ao somar os campos da figura 1b obtém-se o campo magnético da figura 1c. Este campo possui intensidade tal que, ao ser somado ao campo magnético presente inicialmente na figura 1a, obtém-se um campo de variação mínima, como mostra a figura 1d.

Figura 1 – Exemplo do processo de correção de inhomogeneidade de campo magnético.

O uso das cunhas de cobre para efetuar a correção de campo é uma forma de *shimming* passivo. Após a instalação de um novo magneto, é feito um mapeamento do campo magnético na região de interesse. (4) Com as medições de campo estabelecidas, cunhas são adicionadas de forma iterativa, geralmente com o auxílio de um *software* específico que determina a posição de cada uma, até que a homogeneidade do campo atinja os níveis desejados. O *shimming* passivo, apesar de relativamente simples de ser realizado, não é preparado a cada experimento por demandar muito tempo e não necessariamente prover os melhores resultados. O *shimming* passivo, além de não se adequar a todo tipo de amostra diretamente, também é sensível a variações de temperatura, causando flutuações indesejadas durante experimentos com gradientes mais intensos.

A outra forma de se realizar ajuste de campo, o *shimming* ativo, é feita com o uso de bobinas extras de formato similar às bobinas de gradiente tradicionais. Cada bobina é projetada para produzir um determinado padrão de campo magnético, e considerando-se a perturbação existente na região de interesse, aplicam-se correntes de diferentes intensidades em cada bobina a fim de minimizar essa perturbação. (5)

Em geral, em experimentos de RM, busca-se limitar a variação do campo magnético no volume de interesse para algum valor conhecido, por exemplo 1 ppm como descrito na seção 1.2.2. O volume de interesse é frequentemente descrito como um volume esférico com origem no centro do objeto em análise, como mostra a figura 2. Convenciona-se que o campo magnético principal, chamado B_0 , encontra-se paralelo ao eixo Z .

Com *shimming* ativo é possível utilizar uma quantidade arbitrária de bobinas de *shimming* para se obter a correção desejada. O problema passa a ser, então, determinar qual o valor ótimo de corrente que deverá ser estabelecida em cada uma das bobinas. Essas bobinas são construídas para produzir gradientes de campo magnético de acordo com o que se conhece por *harmônicos esféricos*. Cada componente da expansão do campo magnético em harmônicos esféricos recebe um nome relacionado à equação da componente em coordenadas cartesianas, de acordo com a tabela 1. Vale notar que as componentes de primeira ordem são equivalentes às bobinas de gradiente convencionais, e por esse motivo, é comum realizar *shimming* de primeira ordem apenas com alterações na corrente do sistema de gradientes já existente. (6)

1.3.1 Ajuste automático

Devido à toda a variação que pode ser obtida com bobinas de *shimming* ativo, pode-se gastar muito tempo realizando um ajuste sem que sejam obtidos resultados de fato satisfatórios. Para esses casos, existem algoritmos que realizam o processo automaticamente, como o FASTMAP (7) e o FAST(EST)MAP. (8) Outros mecanismos mais genéricos podem

Figura 2 – Volume de interesse dentro de um magneto. As linhas de campo na região central, correspondente ao campo B_0 , são paralelas ao eixo Z . Costuma-se falar sobre o volume de interesse como uma esfera, aqui representada com diâmetro d .

Fonte: Elaborada pelo autor.

Tabela 1 – Equações de campo para harmônicos esféricos de primeira e segunda ordem

Nome	Equação de campo
X	x
Y	y
Z	z
Z2	$2z^2 - (x^2 + y^2)$
ZX	zx
ZY	zy
$X^2 - Y^2$	$x^2 - y^2$
2XY	xy

Fonte: Adaptada de MINER; CONOVER. (3).

também ser empregados, como o uso de gradientes descendentes ou mesmo do Simplex (9) para casos apenas com componentes lineares.

1.4 O projeto *ToRM*

O projeto *ToRM* teve início na década de 1980 como um empreendimento para o desenvolvimento de tecnologia voltada para ressonância magnética no país. Em 1983 foi

construído o primeiro protótipo do grupo, o qual era capaz de produzir imagens para pequenos objetos, sendo responsável pelas primeiras imagens por RM no hemisfério sul.

Ainda na década de 1980, foi produzido um segundo equipamento com financiamento proveniente do Programa de Apoio ao Desenvolvimento Científico e Tecnológico (PADCT). O sistema de imagens produzido utilizava um magneto de 2.0 Tesla, e era capaz de produzir imagens para objetos de até 15 cm de diâmetro. (10, 11) Diversas publicações surgiram como resultado de experimentos com esse equipamento, inclusive resultados de aplicação na indústria, o que trouxe o interesse de empresas como a Embrapa para o projeto. Novos equipamentos continuaram a ser desenvolvidos até que, em 1995, o grupo passou a operar um sistema clínico de RM dentro da Santa Casa de São Carlos. Esse sistema operou por 8 anos e gerou mais de 8 mil exames, em grande parte para o Sistema Único de Saúde (SUS).

Atualmente, o grupo se encontra desenvolvendo um espectrômetro de ressonância magnética de concepção totalmente digital. Apesar de o espectrômetro ser o elemento central, ele necessita de outros elementos para ser de fato funcional, como por exemplo a Console (12) e a IDE. (13)

1.5 Objetivo

O controlador de bobinas de *shimming*, apesar de não ser necessário para a obtenção de um espectrômetro funcional, representa um elemento com capacidade para melhorar substancialmente a qualidade dos experimentos. O objetivo deste é projetar e construir o *hardware* e o *software* necessários para que a Console possa definir os coeficientes de *shimming* de um experimento. Não faz parte desse projeto o desenvolvimento de algoritmos de ajuste de *shimming* automático, mas ele inclui as capacidades básicas para que isso seja possível futuramente.

2 Controlador de bobinas de *shimming*

2.1 O controlador

Todo objeto próximo da região onde se localiza o espectrômetro distorce de alguma forma o campo magnético principal do qual dependem os experimentos de RM. Devido a essas distorções, é necessário aplicar correções a fim de tornar o campo magnético mais homogêneo. Com um campo magnético mais homogêneo, experimentos são mais previsíveis e o ruído pode ser significativamente reduzido. Em experimentos de imagem, um campo magnético mais homogêneo permite a utilização de gradientes de campo menos intenso, e conseqüentemente uma menor largura de banda de aquisição e, em decorrência, menor ruído. Em espectroscopia, a homogeneidade de campo é fundamental para que se observe espectros com resolução suficiente para permitir a separação das linhas. Mesmo em campos não tão intensos, onde a dispersão do *chemical shift* — que é proporcional à intensidade do campo — é pequena e a largura das linhas é inversamente proporcional à homogeneidade do campo. Dessa forma, a essência de se observar linhas resolvidas espectralmente é diretamente relacionada à qualidade do procedimento de *shimming*.

O papel do controlador de bobinas de *shimming* é definir a intensidade do campo magnético produzido por cada uma das bobinas. Isto é feito controlando a corrente que fluirá por cada uma delas da forma mais precisa possível, para que a inomogeneidade final do campo magnético principal seja mínima.

É importante ressaltar que o controlador é responsável por gerenciar um *conjunto* de bobinas, cada qual com sua característica própria, como descrito na seção 1.3. Cada instalação tem necessidades diferentes, e por isso usa quantidades de bobinas diferentes para seus propósitos. Em geral, instalações para geração de imagens utilizam menos bobinas de *shimming* do que instalações para experimentos de espectroscopia. Isto se deve ao fato de que experimentos de imagens podem utilizar gradientes de campo magnético intensos, o que produz uma grande dispersão espectral, muito maior que a produzida pela variação espacial do campo gerada pelas diversas fontes de inomogeneidades. Já em espectroscopia, não há como controlar a dispersão, que ocorre naturalmente devida ao *chemical shift*, portanto o campo tem que ser feito homogêneo para que a dispersão pela inomogeneidade seja menor que a do *chemical shift*.

Para acomodar a variação entre diferentes ambientes e aplicações, o controlador foi projetado para suportar até 40 canais — cada canal representa uma bobina de *shimming* — simultaneamente, suficiente para a grande maioria das instalações. A limitação no número de canais se deve, no entanto, ao *hardware*: o *software* é capaz de suportar um número

arbitrariamente grande de canais, e caso seja necessário, o equipamento pode ser alterado para adicionar mais canais. Essa propriedade do controlador segue a política adotada pelos pesquisadores do CIERMag para a definição das especificações gerais do Espectrômetro Digital, especificamente o quesito Escalabilidade. Neste caso, o sistema já se encontra preparado para atender à demanda por novas funcionalidades, previstas para futuros experimentos.

O controlador foi projetado para ser modular. Há um limite máximo de 40 canais, mas cada instalação pode usar uma quantidade reduzida. Blocos de 8 canais cada podem ser instalados ou removidos quando necessário, e o *software* se adequa à quantidade de canais disponíveis no equipamento.

Além de ser capaz de controlar uma quantidade grande de canais, uma característica importante do controlador é que ele deve ser preciso. Quanto mais fino o ajuste do campo magnético, menor será o ruído final obtido após o ajuste do *shimming*.

Apesar de ser possível realizar alterações de *shimming* em cada experimento, muitas vezes uma mesma configuração é mantida por vários dias ou semanas. Para esses casos, a velocidade de alteração da corrente que flui em cada uma das bobinas não é um fator crítico, sendo priorizada a estabilidade. Entretanto, por vezes não é possível obter um ajuste que seja igualmente satisfatório para todo o volume útil do equipamento. Uma determinada região pode ter alta homogeneidade, mas outra adjacente — por exemplo em um experimento de aquisição de imagens funcionais *Echo-planar Multislice* — pode estar aquém do esperado. Para esses casos, o controlador foi projetado para realizar mudanças rápidas da configuração de *shimming*. Será possível realizar experimentos em que a configuração será alterada durante o próprio experimento, o que é chamado de *shimming dinâmico*, com interferência mínima no tempo de execução total.

Para garantir que todo o sistema esteja funcionando adequadamente, um sistema de monitoração foi incluído no projeto. A corrente que passa pelas bobinas será constantemente monitorada para que seja possível determinar se não está sendo estabelecida uma corrente diferente da esperada, ou mesmo se a bobina está presente ou danificada.

2.1.1 Abordagem

O controlador foi projetado para ser modular, de forma a ser possível utilizar diferentes quantidades de canais de *shimming* em cada ambiente. É composto por essencialmente três partes, identificadas na figura 3. A figura 3 também dispõe o controlador no contexto do projeto maior .

Figura 3 – Diagrama do controlador.

Fonte: Elaborada pelo autor.

O controlador, como equipamento, deveria ser posicionado próximo ao espectrômetro por dois motivos: reduzir a latência de comunicação com equipamentos de potência e permitir a comunicação direta com o espectrômetro em algumas situações. A grande maioria das interações com o controlador, no entanto, seria realizada de uma estação de trabalho não necessariamente próxima. Devido à distância, foi decidido que a comunicação entre a estação de trabalho e o controlador seria realizada via Ethernet, como descrito na seção 2.1.4.

Imaginou-se desde o início do projeto que o controlador seria um equipamento independente, não sendo parte de uma estação de trabalho. Por esse motivo, foram buscadas alternativas para embarcar o controlador no espectrômetro de alguma forma. A primeira alternativa foi utilizar um microcontrolador convencional, mas este deveria ser capaz de se comunicar via Ethernet sem muitos periféricos. O microcontrolador PIC32MX795F12L foi escolhido por ser distribuído junto a um *kit* de desenvolvimento que incluía, entre outras coisas, um conector Ethernet e várias linhas SPI. O projeto com o PIC, no entanto, foi abandonado por motivos descritos na seção 2.1.1.1.

Um dos fatores usados para decidir qual microcontrolador utilizar foi a presença ou não de canais SPI no dispositivo. Os conversores analógico/digital e digital/analogico de interesse usavam SPI como forma de comunicação, como descrito na seção 2.1.2. O protocolo em si e o modo como foi utilizado durante o desenvolvimento são apresentados na seção 2.1.3.

Para substituir o PIC foi utilizado um Raspberry Pi, um computador de pequeno porte que além de sanar alguns dos problemas do PIC, também possuía algumas outras

vantagens. As características que tornaram o Raspberry Pi uma alternativa viável são descritas na seção 2.1.1.2.

2.1.1.1 PIC

O controlador, por estar em um local diferente da estação de trabalho, precisaria de um processador próprio que fosse capaz de se conectar a uma rede Ethernet. O primeiro equipamento escolhido para servir de central do espectrômetro foi o *PIC32MX795F12L*, um microcontrolador de 32 bits da Microchip.

A principal motivação para a escolha deste microcontrolador é que ele possui, dentro de seu núcleo, um módulo Ethernet MAC, permitindo o uso de PHYs diversos. Além disso, seu *kit* de desenvolvimento, apresentado na figura 4, já contém todos os componentes necessários para conectar o dispositivo a uma rede Ethernet, sendo bastante simples de implantar o dispositivo em seu destino final.

Figura 4 – *Kit* de desenvolvimento PIC Ethernet Starter Kit.

Fonte: Elaborada pelo autor.

Ele também possui diversos canais SPI e I2C, que seriam utilizados posteriormente para comunicação com conversores analógico/digital e digital/analógico.

O projeto com o microcontrolador PIC, no entanto, apresentou diversas dificuldades e por isso foi abandonado. Apesar de o equipamento ser bastante versátil, sua versatilidade e minimalismo tornaram-se empecilhos quando modificações mais abrangentes foram necessárias.

A maior dificuldade foi encontrada no *software* de base para o microcontrolador. Escrito em C, esse *software* interfaceava com componentes do *kit* de desenvolvimento como o

controlador de rede, de USB entre outros. Por ser necessário lidar com interfaces de muito baixo nível, muito tempo foi gasto resolvendo problemas não diretamente relacionados ao projeto principal.

2.1.1.2 Raspberry Pi

No lugar do projeto com o microcontrolador *PIC32MX795F12L*, foi decidido utilizar o computador Raspberry Pi. O computador completo, apresentado na figura 5, tem tamanho comparável ao kit de desenvolvimento do PIC32 e resolve diversos dos problemas encontrados com o modelo anterior:

- Por executar Linux diretamente, a pilha TCP/IP faz parte do sistema operacional multi tarefa. Não é necessário fazer alterações nos *drivers* para que a interface de rede possa ser utilizada satisfatoriamente
- Também como resultado por executar Linux, é possível compilar e usar diversas ferramentas que não podem ser executadas no microcontrolador

Figura 5 – O computador Raspberry Pi.

Fonte: Elaborada pelo autor.

2.1.2 Conversores A/D e D/A

Desde o começo do projeto, era de interesse que o controlador fosse capaz de realizar ajustes finos na corrente de cada bobina. O modo mais direto de conseguir um ajuste adequado é utilizar um conversor analógico/digital de boa resolução, com pelo menos 12 *bits* de resolução.

Foram escolhidos como conversores os dispositivos *LTC2656* e *LTC1856*, produzidos pela *Linear Technologies*, ambos com 16 bits de resolução.

Ambos os conversores funcionam de forma similar: comunicam-se via SPI e contém, cada um, 8 canais. Valendo-se da intenção de tornar o controlador o mais modular possível, cada módulo foi construído a partir de um conversor analógico/digital e um conversor digital/analógico. Por esse motivo, o controlador permite o acesso a no mínimo 8 canais, e no máximo 40, sempre em múltiplos de 8.

A alta resolução se faz necessária para que seja possível ajustar o *shimming* de forma bastante precisa. Um melhor ajuste de *shimming* pode ocasionar uma melhoria significativa na qualidade do resultado final. A alta resolução, no entanto, também ocasionou alguns contratempos: a mínima variação de tensão na saída do conversor digital/analógico é de cerca de $70\mu V$, e com uma variação tão pequena, o desenho da placa de circuito impresso torna-se essencial para garantir que a saída esperada não seja afetada por fatores externos. Foram seguidas as diretrizes especificadas no *datasheet* de cada componente, mas em testes iniciais ainda sim houve mais ruído do que o esperado.

Como pode ser observado na figura 6, apesar de haver uma variação de até $3,5mV$ no sinal esperado, a saída é monotônica. Isso é importante porque o controlador deve ser capaz de aplicar pequenas variações de corrente de forma consistente. O valor de cada canal não tem significância física direta, e bobinas de mesma função, mas em espectrômetros diferentes, vão utilizar valores diferentes de corrente para obter a mesma correção de campo. Valores absolutos, portanto, não são de grande importância para o controlador, exceto para ajustes de *shimming* iniciais, como bons pontos de partida para o ajuste — Algoritmos de *shimming* automático podem obter resultados mais rapidamente com bons valores padrão. É importante, no entanto, que valores relativos sejam consistentes para garantir que, por exemplo, algoritmos automáticos possam convergir para um mínimo global, e não somente local.

No lado do conversor Analógico/Digital (A/D), a situação é similar: foi observado nos protótipos que os valores lidos apresentavam uma pequena variação em relação a uma fonte de tensão confiável, de magnitude similar à da observada nos DACs e também monotônica.

2.1.3 SPI

Serial Peripheral Interface (SPI) é um protocolo de comunicação bidirecional síncrono que utiliza 4 linhas digitais. É um protocolo utilizado principalmente em sistemas embarcados devido à sua simplicidade. Dependendo da aplicação, é possível reduzir a comunicação para 3 ou até mesmo só 2 canais.

Figura 6 – Erro médio de cada DAC obtido no primeiro protótipo. O código de entrada corresponde ao valor enviado via SPI para cada DAC.

Fonte: Elaborada pelo autor.

Os seguintes sinais são usados para comunicação via SPI:

Master Output Slave Input (MOSI)

Master Input Slave Output (MISO)

Serial Clock (SCLK)

Slave Select (SS)

Como sugerem os nomes dos sinais, o protocolo pressupõe a existência de um dispositivo (mestre) que controla a todos os outros (escravos). A comunicação, apesar de bidirecional, é sempre iniciada pelo mestre e dados são transmitidos simultaneamente nas duas direções. Mesmo quando a intenção é somente enviar dados para algum dispositivo o mestre receberá dados do escravo, ainda que esses dados não contenham nenhuma informação.

Dados são transmitidos de fato durante cada pulso do sinal de *clock*, durante o qual cada dispositivo tem um curto período para realizar a amostragem do sinal em sua linha de dados de entrada. Dois parâmetros controlam o modo como essa amostragem é feita: polaridade e fase do *clock*, identificados, respectivamente, por CPOL e CPHA, sendo atribuídos a cada um os valores 0 ou 1.

A polaridade de *clock* indica qual o nível do sinal de *clock* quando este está em espera, ou seja, quando não há dado sendo transmitido. Quando CPOL = 0, o sinal de *clock* é

mantido em nível lógico baixo, e quando $\text{CPOL} = 1$, em nível lógico alto. A fase do *clock* diz respeito ao momento em que o sinal deve ser amostrado: quando $\text{CPHA} = 0$, o sinal é amostrado na borda de subida, e quando $\text{CPHA} = 1$ o sinal é amostrado na borda de descida. A figura 7 mostra quando o sinal é amostrado em cada um dos casos.

Figura 7 – Polaridade e fase do *clock*

Fonte: Adaptada de *SERIAL...* (14).

Dispositivos SPI podem ser conectados de forma *paralela* ou em *daisy chain*, como mostrado na figura 10. Como usualmente cada dispositivo utiliza o sinal SS para determinar se ele deve ou não processar os sinais em sua entrada, cada dispositivo extra precisa de potencialmente uma linha lógica a mais a partir do mestre para controlar a comunicação, como na figura 10a. No entanto, dependendo dos dispositivos utilizados, é possível organizar os dispositivos em *daisy chain*. Neste modo, a saída de um dispositivo é conectada à entrada do dispositivo seguinte, até que o último dispositivo na cadeia seja ligado ao mestre novamente. Essa forma de comunicação, mostrada na figura 10b, funciona somente quando o dispositivo SPI dá suporte a esse método. Quando possível, no entanto, permite a utilização de somente uma linha SS para todos os dispositivos.

Usualmente, dispositivos em *daisy chain* são dispostos próximos uns aos outros, o que se traduz em menores trilhas na placa de circuito impresso. Menores trilhas tornam o projeto mais simples e possuem menor capacitância parasita, permitindo que o circuito funcione adequadamente em frequências mais altas.

Além dos benefícios de espaço físico, é possível também reduzir consideravelmente o tempo de transmissão de comandos usando *daisy chain*. A figura 8 mostra um exemplo de transmissão em *daisy chain*. Nela, cada linha representa um módulo DAC, e é possível ver, ao longo do tempo, como cada comando é repassado para o módulo DAC seguinte: o primeiro bloco de dados recebidos pelo DAC 1 está, ao fim da transmissão, no DAC 5. Ao se usar *daisy chain*, é eliminado todo o *overhead* que estaria presente caso fosse necessário iniciar uma transmissão para cada DAC separadamente.

Um dos maiores inconvenientes do SPI é que o protocolo não é padronizado. Apesar de amplamente utilizado, não há um padrão descrevendo todas as interações possíveis entre mestre e escravo, fazendo com que diferentes fornecedores implementem o protocolo com

Figura 8 – Propagação do sinal em dispositivos SPI em *daisy chain*. Cada cor representa um comando enviado a um dispositivo SPI, e como ele trafega por todos os dispositivos até chegar em seu destino.

Fonte: Elaborada pelo autor.

pequenas variações relativos um ao outro. No controlador de *shimming*, os dois dispositivos utilizados são produzidos pela mesma empresa, e tem especificações bastante similares.

Neste projeto, foram utilizados dois tipos de dispositivos SPI: os DACs, que podem ser utilizados com *daisy chain*, e os ADCs, que devem ser endereçados de forma paralela.

As linhas SCLK, MOSI e MISO foram compartilhadas com todos os dispositivos. Para reduzir o efeito de capacitâncias parasitas denegrindo o sinal, foi utilizado um *buffer* entre o Raspberry Pi e os demais dispositivos, valendo-se também de que as entradas dos dispositivos inoperantes se encontram em estado de alta impedância.

Como os DACs são acessados em *daisy chain*, somente uma linha extra é utilizada para eles. No caso dos ADCs, toda operação resulta em uma leitura, e como o canal de saída é usado para transmitir os dados dessa leitura, não é possível usar ligação em *daisy chain*. Os ADCs são acessados de forma paralela.

O Raspberry Pi possui dois canais SPI para aplicações externas, sendo que as linhas

SCLK, MOSI e MISO são compartilhadas entre os dois canais, e espera-se que os pinos CE0 e CE1 (*Chip Enable*) sejam usados com dispositivos SPI em paralelo. Como seriam controlados 10 dispositivos SPI com os mesmos sinais lógicos, seriam necessários a princípio 10 sinais do tipo SS. Desses 10, é possível reduzir para 6 valendo-se do *daisy chain* presente nos DACs. Como não há tantos canais disponíveis no Raspberry Pi, o controle deveria ser feito por *software* de alguma forma. Decidiu-se então usar um dos canais disponíveis para controlar os DACs e o outro para controlar os ADCs, usando três pinos extras para ativar um multiplexador que iria de fato alimentar os sinais SS dos ADCs. Não há uma redução significativa na quantidade de sinais necessários — 5 linhas saindo do Raspberry Pi contra 4 — mas usar o *chip enable* como faz com que, após decidir qual ADC receberá os comandos, a entrada e saída de dados seja feita diretamente pelo *driver*, sem necessidade de controlar o *slave select* diretamente.

A estrutura geral do controlador, apresentada na figura 9, mostra a separação entre cada módulo e como são interligados com componentes externos. Vale ressaltar também a presença de um botão de “pânico”: caso haja uma falha considerável no controlador ou no equipamento como um todo, a saída pode ser cortada imediatamente por ação externa.

2.1.4 Ethernet e *layout* da rede

O projeto ToRM desde muito cedo utiliza Ethernet na camada de enlace para transporte de dados entre nós adjacentes. As altas taxas de transmissão possíveis junto à confiabilidade e isolamento do par trançado fazem com que o seu uso seja bem-vindo em um ambiente com quantidade significativa de interferência eletromagnética externa como o que se vê nos laboratórios de ressonância magnética.

Uma das vantagens imediatas em se usar Ethernet é a ubiquidade. Praticamente qualquer computador possui uma entrada Ethernet à qual o controlador pode ser conectado ou, em uma rede mais complexa, é possível utilizar roteadores e *switches* para conectar o controlador a equipamentos mais distantes. É um protocolo estável, em uso há décadas e que pode ser usado com meios físicos mais rápidos, como fibras ópticas, caso necessário com pouca ou nenhuma alteração no *software* do controlador.

Apesar de não fazer parte do escopo do projeto, o *layout* da rede é de grande importância para o funcionamento do controlador. Cada dispositivo na rede deve ser capaz de se conectar a outros dispositivos do projeto ToRM, sendo utilizada a pilha TCP/IP ou UDP/IP dependendo da aplicação. A alocação de endereços IP para cada dispositivo é função de um servidor DHCP externo. Este servidor deve distribuir endereços específicos para cada aplicação, de forma que, dentro da mesma rede, cada dispositivo possa se conectar a outros de forma transparente e consistente.

Figura 9 – Estrutura interna do controlador.

Fonte: Elaborada pelo autor.

2.1.5 HTTP

Depois de admitir um endereço [IP](#), cada serviço precisa estar pronto para receber ou efetuar requisições. Há muitos protocolos disponíveis para transmitir dados entre dispositivos, com diversas vantagens e desvantagens entre eles. Um dos protocolos mais comuns para troca de mensagens é o [HTTP](#), usado em praticamente qualquer dispositivo conectado à internet.

[HTTP](#) é um protocolo de aplicação usado principalmente para transmitir hipertexto em uso desde 1991, sendo que a versão 1.1, de 1999, [\(15\)](#) ainda correspondia a cerca de 70% de todo o tráfego HTTP em outubro de 2015. [\(16\)](#) Em sua forma mais comum, é um protocolo textual simples de ser manipulado e que oferece facilidade para ser depurado se necessário. É extensível e pode ser usado independente do tipo de conteúdo que se deseja transmitir. Uma requisição [HTTP](#) no formato texto tem a estrutura da figura [11](#), mas permite agregar diversas outras informações além das apresentadas.

Figura 10 – Dispositivos conectados por SPI.

Fonte: Elaborada pelo autor.

```
GET / HTTP/1.1
Host: exemplo.com.br
Content-Type: text-plain
Content-Length: 8
```

```
conteudo
```

Figura 11 – Exemplo de requisição HTTP

Fonte: Elaborada pelo autor.

Uma outra característica de grande interesse do protocolo é a ausência de armazenamento de estado. Isso faz com que cada requisição seja independente, e possa ser processada sem conhecimento de qualquer outra requisição que seja feita ao sistema. Cada requisição deve conter toda a informação necessária para produzir os resultados esperados. Em grande escala, essa propriedade promove um melhor uso de caches e permite que uma mesma requisição seja processada por diferentes servidores sem comprometer o seu resultado.

No projeto [ToRM](#), tem-se a intenção de fazer com que quase todos os serviços de sistemas possam ser acessados via [HTTP](#). Isso torna o acesso a diferentes partes do sistema mais uniforme, e devido à grande variedade de bibliotecas disponíveis, é possível acessar serviços sobre [HTTP](#) em diferentes linguagens sem grande dificuldade.

Para tornar a interação com outros serviços ainda mais simples, os diferentes projetos do [ToRM](#) podem se valer do PyMR: uma biblioteca desenvolvida pelos pesquisadores do [CIERMag](#) que estabelece uma interface de programação para equipamentos de ressonância magnética. (12) Com essa interface, será possível obter informações de outros sistemas de forma mais consistente. Com todos os dispositivos utilizando o mesmo protocolo e usando

interfaces similares para comunicação, a resolução de eventuais problemas ou mesmo a coleta de informações sobre esses sistemas se torna muito mais simples.

Uma consideração importante no acesso a serviços remotos diz respeito à autenticação e autorização. Especialmente no contexto de equipamentos médicos é essencial que todos os acessos sejam autenticados para evitar danos ao paciente tanto quanto ao equipamento. Existem algumas formas de se autenticar requisições [HTTP](#) do lado do servidor, como por exemplo o uso do cabeçalho *Authorization*, *cookies* ou OAuth. O processo de autenticação, no entanto, é realizado valendo-se dos próprios mecanismos do HTTP, em geral cabeçalhos especiais. Isso significa que se um terceiro na rede puder ler o conteúdo da mensagem, ele também pode criar novas mensagens e causar distúrbios nos equipamentos. Para evitar essa situação, é necessário que além de conferir autenticação, as mensagens possam ser transmitidas de forma confidencial. Dentro do [HTTP](#), isso geralmente é resolvido com o uso de uma abstração extra, chamada de *Transport Layer Security* ([TLS](#)), o que dá origem ao que se conhece como *HTTP Secure* ([HTTPS](#)). Os mecanismos para se oferecer um serviço sobre [HTTPS](#) são bem conhecidos e o uso adequado da tecnologia provê confidencialidade e integridade de forma praticamente transparente aos clientes.

O controlador permite que seus clientes realizem principalmente as seguintes operações:

- Obtenham o número de canais de *shimming* disponíveis
- Obtenham a intensidade de corrente atualmente estabelecida em cada bobina de *shimming*
- Definam novos valores para a corrente em cada bobina de *shimming*

Essas operações permitem que o controlador cumpra sua finalidade — a de efetivamente estabelecer a corrente esperada em cada bobina de *shimming* — mas para que se integre de forma mais completa ao projeto [ToRM](#), outras operações são necessárias. Por exemplo, notificar outros sistemas se algum canal não apresenta o resultado esperado. Condições anômalas podem indicar falhas mais profundas no equipamento e divulgar esse tipo de informação antes de realizar outras operações pode ser essencial para a manutenção dos equipamentos.

2.1.5.1 [JSON](#)

Javascript Object Notation ([JSON](#)) é um formato para representação de dados relativamente compacto e de boa legibilidade para humanos. [JSON](#) é usado pela maior parte dos projetos relacionados ao espectrômetro devido à sua simplicidade. É usado tanto para

arquivos de configuração de serviços como para transporte de dados entre equipamentos diferentes.

No controlador, **JSON** é usado principalmente para o envio de comandos ao servidor. A Console — ou outro cliente — envia para o controlador uma requisição **HTTP** com um caminho especificando o tipo de operação a ser realizada, por exemplo definir valores de corrente em diversos canais, e no corpo da mensagem um objeto **JSON** com os valores a serem aplicados em cada canal.

O uso do mesmo formato em vários dispositivos permite também que alguns tipos de mensagens possam ser padronizadas — por exemplo, incluindo uma chave comum indicando que houve um erro em uma operação de forma que qualquer outro dispositivo pode prontamente verificar se algo não está funcionando como o esperado.

2.2 Placa de Circuito Impresso

O controlador de bobinas de *shimming* é um projeto que envolve tanto elementos de *software* como de *hardware*. Nas seções anteriores foram discutidos principalmente aspectos de *software* relativos ao controlador, como os protocolos de comunicação em uso e o ambiente de execução. Mas o controlador em si é definido pelo seu *hardware*. Cada módulo pode ser substituído independentemente, sem afetar o funcionamento dos demais.

A confecção de placas de circuito impresso é um processo que passa por várias etapas, as quais vão ser descritas nas próximas seções.

2.2.1 Projeto

Um projeto de *hardware* passa por várias etapas, sendo que em cada etapa uma série de ferramentas diferentes são necessárias. De forma simplificada, um projeto de *hardware* é desenvolvido na seguinte sequência:

1. Desenho do diagrama esquemático
2. Organização dos componentes na placa de circuito impresso
3. Confecção da placa de circuito impresso
4. Montagem dos componentes
5. Teste e distribuição

A primeira etapa é o desenho do circuito propriamente dito. O produto desta etapa é o diagrama esquemático, que representa as conexões lógicas entre os diversos elementos do circuito. Essa etapa representa cada elemento de forma abstrata — ainda não é necessário conhecer precisamente qual o formato físico de cada componente. São traçadas as interconexões entre os pinos dos elementos e geralmente realizadas simulações para determinar se o circuito apresenta o comportamento desejado. No projeto do controlador, foi utilizado o OrCAD Capture (17) para o projeto do diagrama esquemático, o qual pode ser visto na figura 12.

Figura 12 – Parte do diagrama esquemático do projeto do controlador, mostrando a ligação entre o Raspberry Pi e os módulos DAC e ADC.

Fonte: Elaborada pelo autor.

Após o diagrama esquemático estar pronto, a próxima etapa é transformar o diagrama lógico em sua representação física, descrevendo o formato de cada ligação entre componentes. Nessa etapa, cada ligação lógica passa a ser descrita por uma ou mais *trilhas*, as quais são construídas como trechos de cobre contínuo. Cada trilha deve obedecer a uma série de parâmetros, tais como:

- Larguras mínima e máxima
- Comprimento máximo ou vinculado ao comprimento de outras trilhas

- Distância mínima de outras trilhas
- Uso de cantos vivos ou arredondados
- Quantidade máxima de trocas de camadas através de vias
- Sobreposição máxima com outras trilhas em camadas diferentes

Devido à necessidade de se conectar fisicamente diferentes componentes, a disposição dos componentes tem um papel extremamente importante na placa final, sendo possível inclusive não haver solução para o problema de interconexão dos componentes com uma trilha contínua. A figura 13 mostra um exemplo desse tipo de situação quando há apenas um plano de roteamento. Para reduzir essa limitação, é extremamente comum que placas possuam diversas camadas extras, muitas delas internas. *Vias* então são usadas para interligar trilhas em camadas diferentes, como mostra a figura 14.

Figura 13 – Exemplo de montagem impossível com uma única camada. Não há como manter as duas trilhas coloridas no mesmo plano da placa de circuito impresso.

Fonte: Elaborada pelo autor.

O roteamento de uma placa é feito, geralmente, com uma mistura de métodos manuais e automáticos. Existem diversos tipos de roteadores e algoritmos para esses roteadores, sendo que cada um apresenta melhores resultados com situações diferentes. Em circuitos menores, grande parte das trilhas pode ser feita manualmente para garantir que terão o formato desejado. Em circuitos maiores, no entanto, o número de trilhas pode ser grande o suficiente para tornar o processo manual muito cansativo ou mesmo propenso a falhas não facilmente detectáveis automaticamente. Nesses casos, o projetista estabelece em linhas gerais por onde um determinado grupo de trilhas deve passar, e o roteador então cria as trilhas de forma automática. Após um roteamento bem-sucedido, o projetista pode ajustar trilhas individualmente conforme o necessário.

Durante o roteamento, uma série de regras devem ser seguidas para reduzir falhas durante a fabricação e para assegurar ao projetista algumas garantias, tais como o não aquecimento de trilhas de maior corrente ou a redução de capacitâncias parasitas. Todas essas regras são verificadas de forma automática, em um processo conhecido como [DRC](#).

Figura 14 – Representação de camadas internas e uma via em uma placa de circuito impresso de múltiplas camadas.

Fonte: Elaborada pelo autor.

Fabricantes de placas de circuito impresso muitas vezes disponibilizam conjuntos de regras para **DRC** para os *softwares* de projeto mais comuns, as quais podem ser integradas ao conjunto de regras já existentes sem necessidade de intervenção extra. No projeto do controlador, foi utilizado o OrCAD PCB Editor, o qual também conta com **DRC** como parte do fluxo de trabalho convencional. A figura 15 mostra uma seção de circuito em que um erro de **DRC** foi detectado.

2.2.2 Confecção

Com o circuito roteado, pode-se iniciar o processo de confecção das placas. O circuito projetado dentro do OrCAD PCB Editor é exportado para arquivos *Gerber*. Gerber é um formato para representação de imagens vetoriais em duas dimensões, usado como o formato padrão para representação das camadas usadas na produção de placas de circuito impresso. O formato representa cada camada somente como um conjunto de formas geométricas, sem associar a cada uma delas qualquer significado além do nome.

Cada camada do projeto é exportada separadamente, e constitui um arquivo independente. Cada camada no contexto do projeto não se refere somente a camadas de cobre, mas inclui também:

Figura 15 – Trecho com erro de DRC exibido pelo OrCAD PCB Editor. A distância entre as duas trilhas apresentadas é menor do que o menor valor permitido.

Fonte: Elaborada pelo autor.

- Máscaras de solda
- Máscaras para pasta de solda
- *Silkscreen*
- Outras camadas de caráter apenas informativo.

A figura 16 mostra exemplos de duas camadas do projeto, ambas do módulo DAC do controlador.

Após a obtenção de todas as camadas como arquivos Gerber, o processo pode seguir de duas formas: pode-se produzir as placas localmente ou enviar para um fabricante externo.

2.2.2.1 LPKF

O CIERMag conta com equipamentos para manufatura de placas de circuito impresso localmente. Os equipamentos e software, todos produzidos pela LPKF, consistem de:

(a) Camada TOP do DAC. Representa as regiões onde há cobre na camada

(b) Camada SILKTOP do DAC. Representa a camada de *silk screen*, com informações úteis para montagem e inspeção da placa

Figura 16 – Duas das camadas usadas para produção do módulo DAC do controlador.

Fonte: Elaborada pelo autor.

- CircuitCam para processar os arquivos Gerber
- Fresadora modelo ProtoMat® S62
- Estação de deposição galvânica Contac RS
- Expositor

O processo é iniciado no CircuitCam, um *software* que é usado para gerar contornos para as trilhas descritas nas camadas de cobre. A fresadora então segue estes contornos e isola cada trilha removendo com uma fresa cônica ou de topo o material ao seu redor, como é mostrado na figura 17. A figura 18 mostra a fresadora em funcionamento.

(a) Trilha dentro do CircuitCAM. Em verde é exibido o caminho da fresa para isolar a trilha.

(b) Placa produzida

Figura 17 – Processo de isolamento de trilhas com a fresadora.

Fonte: Elaborada pelo autor.

(a) A fresadora utilizada para confecção dos protótipos (b) Detalhe da fresadora em funcionamento

Figura 18 – A fresadora ProtoMat S62.

Fonte: Elaborada pelo autor.

2.2.2.2 Processo fotográfico

A fresa remove fisicamente o cobre de algumas regiões a fim de isolar trilhas. Esse processo, apesar de rápido, tem baixa resolução, com distância mínima entre trilhas de aproximadamente $200\mu m$. Além disso, devido a possíveis variações da espessura do cobre na placa, é desencorajado o uso de trilhas muito finas pois pode ocorrer ruptura devido ao esforço mecânico desenvolvido pela fresa.

O método fotográfico é o mais empregado em fábricas de placa de circuito impresso por sua eficiência quando aplicado a um grande número de placas. O mesmo processo, com pequenas variações, pode ser empregado para o recorte das áreas de cobre assim como para a criação das máscaras de solda e *silkscreen*. Em linhas gerais, o processo funciona da seguinte forma:

1. A placa é coberta com um material fotossensível
2. Uma máscara monocromática opaca é usada para ocultar parte da placa. O conjunto placa e máscara é então exposto à uma luz especial, geralmente ultravioleta, que faz com que o material fotossensível cure e enrijeça-se nas áreas a que foi exposto
3. A placa é lavada em solução apropriada, removendo o material que não foi exposto

Para camadas como *silkscreen* e máscara de solda, o processo termina nessa etapa. Mas para camadas de cobre, ainda é necessário remover o cobre indesejado da placa. A placa é então colocada em uma solução corrosiva, que remove o cobre da superfície. Ao fim do processo, o material fotossensível pode ser também removido com a solução apropriada.

Duas soluções diferentes foram usadas durante o desenvolvimento do projeto: uma a base de cloreto férrico (FeCl_3) e outra produzida com cloreto cúprico (CuCl_2). A solução de cloreto cúprico é mantida em um tanque especial, mostrado na figura 19, o qual possui um borbulhador de oxigênio e um aquecedor. O aquecimento melhora a eficiência do processo, e o borbulhamento promove tanto a uniformização da solução na superfície da placa como também sua autorregeneração — a solução nunca se exaure.

O processo completo é ilustrado nas figuras 20 e 21.

Figura 19 – Tanque com solução de cloreto cúprico para corrosão da placa de circuito impresso.

Fonte: Elaborada pelo autor.

2.2.2.3 Metalização

O projeto do controlador utilizou uma placa de duas camadas, com cobre exposto nos dois lados da placa. Para conectar trilhas em camadas diferentes, são usadas *vias*, furos posteriormente metalizados. Vias podem interligar diferentes camadas, e recebem nomes diferentes dependendo da sua disposição. A figura 22 mostra os tipos de vias mais comuns.

A metalização dos furos no processo galvânico da Contac RS é feita em quatro etapas. A estação de metalização da figura 23 possui seis tanques, cada um para um tipo de

Figura 20 – Processo fotográfico para revelação da imagem da camada.

Fonte: Elaborada pelo autor.

Figura 21 – Corrosão do cobre no processo fotográfico.

Fonte: Elaborada pelo autor.

solução.

A primeira etapa consiste em condicionar a superfície dielétrica da placa, normalmente produzida com fibra de vidro (FR4). Inicialmente lava-se a placa em uma solução alcalina, responsável por remover graxas e óleos presentes em sua superfície e no substrato pelo qual foi feita a furação. Imediatamente após, é feito um condicionamento da superfície para que se torne receptiva às partículas de carbono do banho subsequente.

Com a superfície devidamente limpa e ativada, a placa é então mergulhada em uma solução contendo nanopartículas de carbono condicionadas para que tenham afinidade com a superfície ativada, a qual é parcialmente adsorvida pelo substrato da placa durante a imersão. A solução de carbono é usada como transformadora da propriedade de condução da superfície dielétrica, essencial para o início do processo de deposição.

Figura 22 – Três tipos de vias frequentemente usadas. (a) via regular (b) via enterrada (*buried via*) (c) via cega (*blind via*).

Fonte: Elaborada pelo autor.

Figura 23 – A estação de metalização Contac RS

Fonte: Elaborada pelo autor.

A placa é então submetida ao processo de deposição galvânica. Nesta etapa, a placa é imersa em uma solução de sulfato cúprico (CuSO_4) e frações niveladoras e abrillantadoras. A deposição é iniciada com a aplicação de uma corrente entre o anodo — duas chapas de cobre fosforoso — e o catodo — a placa de circuito impresso — como mostra a figura 24. É também aplicado o método de *Reverse Pulse Plating* (RPP), que facilita a deposição em furos e reentrâncias e normaliza a deposição em substratos com distribuição de campo

elétrico altamente não uniforme.

Figura 24 – Representação do processo de deposição galvânica.

Fonte: Elaborada pelo autor.

Por fim, pode-se depositar uma fina camada de estanho sobre o cobre, usando também deposição galvânica. Essa camada protege o cobre, que é mais susceptível a oxidação que o estanho. A camada extra de estanho, no entanto, não é aplicada quando é usado o método fotográfico. Quando o método fotográfico é usado nesse caso, devido à presença de dois materiais distintos, a velocidade de corrosão de cada material é diferente e pode ocasionar falhas nas áreas expostas (*undercutting*).

A figura 25 mostra um pedaço de placa de circuito impresso em cada uma das etapas da metalização dos furos.

2.2.2.4 Fabricação externa

Para os módulos do DAC e do ADC, foram produzidas placas para testes iniciais, mas por serem necessárias cinco placas de cada módulo, optou-se por produzir as placas em uma fabricante especializada. Produtoras precisam, em geral, apenas dos arquivos Gerber de cada camada e da furação. As fotos da figura 26 mostram os dois módulos produzidos. Esta abordagem foi adotada mesmo tendo disponibilidade dos equipamentos de prototipagem no CIERMag, que foram, contudo, danificados e estiveram inoperantes por um tempo considerável das nossas atividades de pesquisa.

(a) Este é o furo no início do processo, antes mesmo da higienização

(b) Após a imersão em solução com nanopartículas de carbono, as paredes do furo ficam cobertas com a solução

(c) A eletrodeposição cobre a parede do furo condicionada com a solução anterior com cobre, conectando de fato os dois lados da placa

(d) É feita a deposição galvânica de uma fina camada de estanho para proteção do cobre

Figura 25 – Etapas do processo de metalização de furos.

Fonte: Elaborada pelo autor.

2.2.2.5 Montagem

A maior parte dos componentes da placa são do tipo *Surface Mount Devices* (SMD), quando cada componente é soldado diretamente sobre a placa, e não através dela. Componentes SMD são em geral menores que equivalentes *through-hole*, com pinos consideravelmente menores e menos espaço entre pinos. Isso dificulta a soldagem por métodos tradicionais, o que deu origem ao que se conhece por *reflow soldering*.

No processo de *reflow*, ao invés de se aquecer cada pino individualmente a fim de fundir

(a) Placa para o módulo ADC

(b) Placa para o módulo DAC

Figura 26 – Placas produzidas externamente para os módulos de DAC e ADC.

Fonte: Elaborada pelo autor.

a mistura de solda e fundente (pasta de solda), cada pino é inicialmente coberto com uma pequena quantidade de pasta de solda. Sobre essa pasta são então colocados todos os componentes SMD, posicionando-os o mais próximo possível da disposição final. A placa inteira é então colocada em um forno com programação de temperatura controlada. Dentro do forno a solda é fundida e a junta entre o pino e o *pad* é formada. A tensão superficial da solda fundida é responsável por corrigir pequenos detalhes de posicionamento e alinhamento dos componentes. Após esse processo, a placa pode ser resfriada e caso haja algum componente do tipo *through-hole* esses serão soldados usando *wave soldering* ou processos manuais.

O controle de temperatura durante o processo de *reflow* é extremamente importante. Variações muito bruscas de temperatura podem ocasionar quebras na placa, e um período muito longo de aquecimento pode trazer prejuízos aos componentes. O processo de *reflow* é dividido em quatro etapas:

1. Na primeira etapa, a placa é aquecida em rampa até aproximadamente 150°C. O objetivo desta fase é aquecer de forma uniforme toda a placa, de forma gradual, a fim de não ocasionar variações bruscas em componentes mais sensíveis que possam danificá-los.
2. Após a temperatura alvo ter sido atingida, ela é mantida por até dois minutos. Durante esta etapa, a temperatura da placa é uniformizada e componentes mais voláteis da pasta de solda tem a chance de evaporar, além de ser iniciada a ativação do fundente contido na pasta de solda.

3. Passado o tempo necessário para a uniformização da temperatura, o processo inicia a fase de *reflow* propriamente dita, quando a solda se liquefaz e une-se aos pinos e *pads*. A solda funde completamente a 217°C, e o forno é aquecido até uma temperatura pouco acima desta. É necessário manter o forno a uma temperatura acima de 217°C por tempo suficiente para que a solda se espalhe na região de interesse, mas não tempo o suficiente para que danifique os componentes eletrônicos. Em geral, a temperatura máxima não é mantida por mais que um minuto, sendo reduzido o tempo caso haja componentes muito sensíveis no projeto.
4. A última etapa é apenas para o resfriamento da placa, em rampa descendente de temperatura. A temperatura deve ser reduzida de forma controlada a fim de evitar rupturas na estrutura da solda ou mesmo dos componentes da placa (*annealing*).

O perfil de temperatura usado para cada placa pode variar consideravelmente dependendo dos componentes presentes, mas um perfil generalizado, o qual foi empregado para os componentes do controlador, é apresentado na figura 27.

Figura 27 – Perfil de temperatura para *reflow soldering*.

Fonte: Elaborada pelo autor.

A aplicação da pasta de solda na placa deve ser feita de maneira também controlada. Com pouca pasta de solda, pode não haver material o suficiente para formar uma junta adequada, e pasta de solda em excesso pode ocasionar curtos, principalmente em componentes menores. A aplicação é geralmente feita por serigrafia (*silk-screen*), usando-se uma matriz rígida de metal (alumínio ou aço inoxidável) ou poliamida. Devido à consistência

da pasta de solda, a espessura da matriz de serigrafia é de extrema importância para garantir bons resultados.

Para o projeto do controlador, foram desenvolvidas máscaras usando chapas de diferentes metais disponíveis no centro de pesquisa. Foram testadas chapas com espessura aproximada de 0,08mm, 0,12mm e 0,40mm. A melhor distribuição de solda foi obtida com a chapa de 0,12mm, com a qual não foram observados curtos ou juntas mal formadas. Os gabaritos para aplicação de pasta testados são mostrados na figura 28.

Figura 28 – Gabaritos para aplicação de solda testados.

Fonte: Elaborada pelo autor.

2.2.3 Módulos

O controlador foi desenvolvido para ser modular desde sua concepção. Devido à versatilidade do Espectrômetro Digital, era de interesse ser possível adequar o controlador de *shimming* também ao ambiente em que foi instalado. Para isso, o projeto foi dividido em três módulos: uma “placa mãe”, um módulo contendo o [ADC](#) e um módulo contendo o [DAC](#). Idealmente, cada módulo [DAC](#) deve ser acompanhado de um módulo [ADC](#), mas optou-se por fazê-los separados por dois motivos:

- Apesar de não ser indicado, ainda é possível utilizar o controlador sem monitoração de retorno
- Cada módulo se comunica com a central de forma ligeiramente diferente, sendo mais prático controlar dois tipos de módulos diferentes a controlar dispositivos diferentes dentro de um mesmo módulo
- Módulos podem ser trocados individualmente caso haja algum defeito

2.3 Software

2.3.1 Interface *web* e linha de comando

O *software* do controlador em si é dividido em três partes:

- Uma biblioteca que oferece serviços de baixo nível, como a alteração e leitura dos valores de corrente estabelecidos em cada canal
- Um *software* de linha de comando que apenas aciona funções distintas da biblioteca descrita no item anterior
- Uma interface *web* que aciona as mesmas funções da aplicação de linha de comando, mas sendo acessada sobre [HTTP](#) e com autenticação

Todo o conjunto é desenvolvido em Python, (18) sendo que a interface *web* é construída usando o *framework* Django. (19) Deve-se notar que a interface *web* expõe, principalmente, apenas uma *Application Programming Interface* ([API](#)) para acesso por outros serviços. Os métodos disponíveis não foram projetados para visualização direta em um navegador.

2.3.2 Automação

Uma parte fundamental de todo *software* diz respeito a quão fácil é realizar sua instalação no sistema alvo. Quanto mais etapas a instalação do *software* exige, maior é a chance de que ocorram erros durante o processo, especialmente se é feito de forma manual. Um modo simples e muito eficiente de reduzir tanto falhas no processo quanto o tempo necessário para sua realização é automatizar o processo.

A instalação do controlador envolve, pelo menos, as seguintes etapas:

1. Instalar o sistema operacional para o controlador
2. Configurar o sistema operacional, como interfaces de rede, usuários, diretórios para armazenar dados etc.
3. Instalar dependências do *software* do controlador
4. Instalar o *software* do controlador
5. Configurar o *software* do controlador

A primeira etapa, a instalação do sistema operacional base, infelizmente não pode ser automatizada pois a versão do Raspberry Pi utilizada nesse projeto não suporta recursos de inicialização pela rede como o *Preboot eXecution Environment* (PXE). Tal técnica seria viável, no entanto, com o Raspberry Pi 3, que permite a inicialização do sistema operacional pela interface de rede. Todas as etapas subsequentes podem ser automatizadas, dependendo apenas de o sistema operacional base estar instalado.

Há várias alternativas de *software* para automatizar a instalação, desde *scripts* escritos unicamente para o projeto, passando por instaladores executáveis até ferramentas capazes de manter informações sobre um grande número de equipamentos a fim de facilitar a configuração de, por exemplo, *clusters*.

Para o controlador, decidiu-se utilizar uma ferramenta específica para automação de processos, o Ansible. (20) Entre os motivos para escolha dessa ferramenta estão:

- Não tem dependências externas, pode ser executado sem nenhum tipo de servidor externo necessário
- Pode ser executado diretamente de dentro do sistema a ser configurado ou remotamente
- Escrito em Python, necessita de poucos módulos fora da biblioteca padrão e em muitos sistemas não exige nenhum componente além dos já em uso
- Usa *Yet Another Markup Language* (YAML) para descrição das operações. A forma declarativa exigida pelo formato é de fácil entendimento

Um ponto muito importante da ferramenta é que, assim como outras ferramentas de gerenciamento de configuração (*configuration management*), é muito simples especificar tarefas de forma que sejam idempotentes, i.e., a execução de uma determinada tarefa repetidas vezes deixará o sistema sempre no mesmo estado. Um exemplo de tarefa idempotente do Ansible é mostrado na figura 29.

Em linhas gerais, o Ansible executa as seguintes tarefas:

1. Habilita o módulo de SPI do *kernel* do Raspberry Pi e instala bibliotecas para acesso aos canais SPI
2. Instala as dependências do *software* do controlador, como servidores *web* e de aplicação


```
---
- name: Cria diretorio
  file:
    path: /var/lib/ciermag/controller
    state: directory
    mode: 0700
    owner: root
    group: root
- name: Inicia e habilita nginx
  service:
    name: nginx
    state: started
    enabled: yes
```

Figura 29 – Exemplo de tarefa idempotente do Ansible.

Fonte: Elaborada pelo autor.

3. Copia o código do controlador para dentro do dispositivo
4. Aplica as configurações para todas as aplicações do dispositivo, incluindo o próprio *software* do controlador

Apesar de não fazer parte diretamente desse projeto, os mesmos princípios de automação usados para o controlador poderão ser aplicados a outros equipamentos, permitindo por exemplo que o endereço de alguns dispositivos seja entregue para as aplicações que os usam sem necessidade de intervenção manual.

3 Resultados

O objeto principal do projeto é o desenvolvimento do equipamento capaz de controlar as bobinas de *shimming*, mas como foi discutido nas seções do capítulo 2, diversos outros itens fazem-se necessários para o funcionamento pleno do projeto.

3.1 O *software* de controle

Ainda que seja possível a um operador interfacear com o controlador acessando o equipamento diretamente, todos os subsistemas do projeto ToRM se comunicam com o controlador através da rede IP.

O *software* de controle do equipamento é dividido principalmente em três partes, como mostrado na figura 30: uma biblioteca, uma interface de linha de comando e uma interface *web*. A biblioteca, chamada de **shimcontrol**, é responsável pelo acesso direto ao *hardware*. A ela cabe:

- Controlar o uso dos dispositivos **SPI** (acesso aos dispositivos de baixo nível `/dev/spidevX.X`)
- Detectar o número de módulos **DAC** e **DAC** disponíveis
- Construir e validar comandos para os módulos

Figura 30 – Diagrama de pacotes do *software* do controlador.

Fonte: Elaborada pelo autor.

3.1.1 Acesso aos dispositivos físicos

Sistemas operacionais tipo Unix, como o Linux em execução no Raspberry Pi, possuem arquivos especiais chamados *arquivos de dispositivos*. Esses arquivos representam

dispositivos físicos ou lógicos disponíveis ao sistema, como por exemplo discos rígidos ou, o que é mais relevante para este projeto, dispositivos [SPI](#).

O *kernel* do Linux possui um *driver* para a utilização de dispositivos [SPI](#) em modo usuário — ou seja, sem necessidade de código executando dentro do *kernel* — chamado *spidev*, o qual gerencia os dispositivos `/dev/spidevX.X`. Devido à forma como funciona o [SPI](#), não é possível que mais de um processo utilize o dispositivo simultaneamente. A biblioteca de acesso ao *hardware* foi projetada para acessar o arquivo de dispositivo somente durante o tempo necessário, para que operações externas, como por exemplo acesso direto por um operador, não sejam afetadas.

3.1.2 Detecção dos módulos disponíveis

A característica modular do controlador faz com que seja possível utilizar de um a cinco módulos de cada tipo dependendo das necessidades da instalação local. No entanto, é necessário que o controlador saiba quantos módulos estão disponíveis para corretamente validar operações.

Dados de entrada precisam ser validados para que clientes sejam avisados caso tentem, por exemplo, estabelecer corrente ou ler informações de um canal inexistente. Dados de saída precisam ser validados para que o controlador possa corretamente construir a cadeia de operações para os [DACs](#) — que utilizam *daisy chain*, conforme a figura [10b](#) — e para que possam endereçar corretamente os [ADCs](#).

O controlador conta com um pequeno conector para ser usado quando não há um módulo instalado. Essa peça apenas faz a conexão direta dos sinais [MISO](#) e [MOSI](#), de forma que nos [DACs](#), o sinal possa ser propagado mesmo sem a presença de um módulo instalado, e nos [ADCs](#), o sinal retorne à origem sem alteração.

Assumindo então que todos os módulos faltantes estão preenchidos com este conector, o procedimento para detecção dos módulos presentes é feito da seguinte forma:

DACs Cada [DAC](#) possui um registrador de deslocamento de 32 *bits*. Esses registradores são inicialmente limpos executando comandos do tipo *no-op* (*no operation*) em cada um dos dispositivos. Em seguida, um comando longo, de 20 *bytes* (5 comandos de 32 *bits*), é enviado aos [DACs](#). Cada comando individual possui, mesmo no caso de *no-ops*, um espaço para indicar o valor de tensão de saída do canal desejado e esse valor é armazenado no registrador de deslocamento interno. Pode-se então enviar valores conhecidos nesses campos e comparar aos valores lidos como resposta ao comando. Por exemplo, caso haja 3 módulos instalados, os dois primeiros comandos

enviados serão retornados e portanto o controlador pode determinar que dois módulos não estão presentes.

ADCs Durante uma troca de dados entre o controlador e cada **ADC**, os primeiros 8 *bits* representam o canal a ser lido. Durante os próximos 16 *bits*, os dados do dispositivo SPI mestre são descartados, enquanto o **ADC** retorna uma leitura. Caso o caminho seja curto-circuitado pelo conector (quanto o módulo do ADC não está presente), o controlador receberá exatamente os mesmos dados que enviou ao invés de uma leitura válida, o que permite determinar a presença do módulo.

3.1.3 Criação e validação de comandos

Esse é um aspecto mais voltado principalmente aos comandos endereçados aos **DACs**.

Comandos enviados aos **DACs** devem ser enviados em blocos de tamanho igual à quantidade de módulos disponíveis, já que deve-se substituir o comando atual presente em cada módulo, como indica a figura 31. A alteração do valor de um único canal deve incluir também *no-ops* para todos os outros **DACs**, de forma que apenas a operação necessária seja realizada.

Figura 31 – Processo de substituição dos comandos em cada um dos módulos **DAC**.

Fonte: Elaborada pelo autor.

Outro aspecto importante é a execução de mais de um comando quando necessário. Caso seja desejado alterar o valor de mais de um canal parte do mesmo dispositivo, cada alteração deve ser feita independentemente, pois cada comando pode executar apenas uma ação. Um exemplo deste tipo de operação é mostrado na figura 32. Uma consequência importante a ser ressaltada é que o tempo de execução do comando não depende então somente do número de alterações, mas também de quais alterações devem ser feitas. Cada comando pode ser executado em um ciclo ou, caso sejam necessárias alterações em todos os 8 canais de um único módulo **DAC**, 8 ciclos.

Para os **ADCs**, a única validação necessária é saber se o canal de interesse existe, o que é detectado com os procedimentos da seção 3.1.2.

Figura 32 – Emissão de múltiplos comandos para alteração de dois canais em um único módulo DAC.

Fonte: Elaborada pelo autor.

3.2 O *hardware* do controlador

O controlador em si foi projetado para ser modular. Isso se traduziu no projeto em separar o equipamento em três partes: um módulo para os ADCs, um módulo para os DACs e uma “placa mãe”. Cada módulo passou por algumas revisões até chegar à forma da figura 33, com o *layout* finalizado.

A placa mãe, inicialmente, incluía também uma fonte de alimentação, mas esta foi removida para reduzir o espaço utilizado pelo controlador diretamente. A alimentação externa é, no entanto, isolada dentro do circuito para reduzir o efeito de possíveis interferências externas e internas, entre os diversos componentes do controlador. A figura 34 mostra o controlador completo, com todos os módulos.

(a) A Placa mãe

(b) O módulo ADC

(c) O módulo DAC

Figura 33 – Placas de circuito impresso já populadas.

Fonte: Elaborada pelo autor.

Figura 34 – O controlador de bobinas de *shimming* com todos os módulos.

Fonte: Elaborada pelo autor.

4 Conclusão

A ressonância magnética é uma técnica bastante versátil, com uma grande variedade de aplicações. Para muitas delas, no entanto, o equipamento base existente não é o suficiente para se extrair resultados satisfatórios ou mesmo resultados que permitam um estudo mais detalhado do que está em análise. O *shimming* é uma das ferramentas existentes para melhorar a qualidade dos experimentos, e foi o objeto de estudo deste trabalho.

Desde o começo do projeto estabeleceu-se que o controlador deveria ter uma característica modular, o que foi possível, por exemplo, com os módulos independentes para DACs e ADCs. Por poderem ser trocados a qualquer hora, admite-se o uso do controlador em diferentes instalações sem prejudicar seu funcionamento.

Em particular o uso do Raspberry Pi se mostrou bastante adequado frente às expectativas de versatilidade e robustez do sistema. A princípio a falta de canais SPI o suficientes poderia ser um problema, mas essa restrição foi, ao fim, benéfica: apesar de o *hardware* ter se tornado mais complexo para suportar os dois modos de comunicação — paralelo e em *daisy chain* — a definição de novos valores de corrente para os DACs se tornou bem mais rápida, pois passou a ser obrigatório o uso de *daisy chain*, fazendo uma atualização completa de todos os canais ser até cinco vezes mais rápida.

O *software* desenvolvido se mostrou também bastante versátil. Alterações são simples de ser realizadas e a disponibilidade do controlador na rede permite que vários outros dispositivos possam no futuro se aproveitar de recursos que ele disponibilize. Caso novas funcionalidades sejam necessárias, bastará uma atualização de *software* para implementar os novos recursos, sem necessidade de modificações do *hardware*.

O controlador depende de equipamentos extras — a ligação com pré-amplificadores e outros elementos de potência — para executar suas funções adequadamente. Mas ao mesmo tempo que ele não opera sozinho, é necessário que as ações que ele pode realizar estejam bem definidas e a interface de controle com outros subsistemas também. De modo geral, o controlador cumpriu o propósito de estabelecer as capacidades desejadas para o subsistema de *shimming* e poderá em breve ser usado em sistemas reais.

4.1 Trabalhos futuros

As características base do controlador foram definidas, mas há diversas oportunidades para expansão tanto do controlador como do *software* cliente que podem ser trabalhadas.

Um item muito importante mas que exigirá tempo considerável para a implementação

são os algoritmos de *shimming* automático. O ajuste de *shimming* em algumas aplicações pode durar várias horas, impossibilitando o uso em experimentos que exigem um ajuste preciso com maior frequência, por exemplo para diferentes experimentos.

O controlador de *shimming* é dinâmico, mas requer que os parâmetros de *shimming* sejam definidos diretamente na Console. No futuro, espera-se adicionar pontos de acesso para que o espectrômetro possa definir valores de *shimming* diretamente: uma série de valores pode ser computada previamente, e o espectrômetro poderá realizar alterações durante a execução de uma sequência, por exemplo em experimento do tipo *multi slice*, para que cada parte do objeto de estudo tenha um ajuste ideal.

A interface *web* para o controlador atualmente funciona só para clientes que podem processar as respostas [JSON](#) diretamente. Planeja-se adicionar uma aplicação *web* simples para que um operador que não pode acessar a Console no momento possa ao menos consultar o estado do controlador.

Referências

- 1 CALLAGHAN, P. T. *Principles of nuclear magnetic resonance microscopy*. Oxford: Oxford University Press, 1991.
- 2 POOLE, M. *Improved equipment and techniques for dynamic shimming in high field MRI*. 2007. Ph. D. Thesis - University of Nottingham, 2007.
- 3 MINER, V. W.; CONOVER, W. W. The shimming of high resolution NMR magnets. 1997. Disponível em: <<http://www.acornnmr.com/content/downloads/shimming.pdf>>. Acesso em: 27 jul. 2016.
- 4 JEZZARD, P. *Shim coil design, limitations and implications*. 2006. Disponível em: <http://mri-q.com/uploads/3/4/5/7/34572113/shim_coil_design_jezzard.pdf>. Acesso em: 27 jul. 2016.
- 5 WENDT, M. *Second order shimming of high field magnets*. Iselin, New Jersey, 2002. Disponível em: <https://static.healthcare.siemens.com/siemens_hwem-hwem_sxxa_websites-context-root/wcm/idc/groups/public/@global/@imaging/@mri/documents/download/mdaw/mtqy/~edisp/second_hot_topic_brochure-00017029.pdf>. Acesso em: 27 jul. 2016.
- 6 BLASCHE, M. *Magnet homogeneity and shimming*. 2015. Disponível em: <https://static.healthcare.siemens.com/siemens_hwem-hwem_sxxa_websites-context-root/wcm/idc/groups/public/@global/@imaging/@mri/documents/download/mda1/ntmx/~edisp/wp_magnethomogeneity_final_hood05162002277576-02503650.pdf>. Acesso em: 27 jul. 2016.
- 7 GRUETTER, R. Automatic, localized in vivo adjustment of all first-and second-order shim coils. *Magnetic Resonance in Medicine*, v. 29, n. 6, p. 804–811, 1993.
- 8 GRUETTER, R.; TKÁČ, I. Field mapping without reference scan using asymmetric echo-planar techniques. *Magnetic Resonance in Medicine*, v. 43, n. 2, p. 319–323, 2000.
- 9 NELDER, J. A.; MEAD, R. A simplex method for function minimization. *Computer Journal*, v. 7, n. 4, p. 308–313, 1965.
- 10 TANNÚS, A. *Desenvolvimento da tecnologia de tomografia por ressonância magnética nuclear*. 1987. 227 p. Tese (Doutorado) — Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 1987.
- 11 MARTINS, M. J. *Desenvolvimento de um tomógrafo de ressonância magnética: integração e otimização*. 1995. 78 p. Tese (Doutorado) — Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 1995.
- 12 PIZETTA, D. C. *Biblioteca, API e IDE para o desenvolvimento de projetos de metodologias de Ressonância Magnética*. 93 p. Dissertação (Mestrado em Ciências) — Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 2014.
- 13 SILVA, D. M. D. D. d. *Desenvolvimento de console multiplataforma para aquisição, organização e visualização de dados do espectrômetro digital de RM do CIERMag: ToRM Console*. 145 p. Dissertação (Mestrado em Ciências) — Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 2014.

- 14 SERIAL peripheral interface bus. 2016. Disponível em: <https://en.wikipedia.org/w/index.php?title=Serial_Peripheral_Interface_Bus&oldid=730163286>. Acesso em: 27 jul. 2016.
- 15 FIELDING, R. et al. *Hypertext Transfer Protocol – HTTP/1.1*. 1999. Disponível em: <<http://dx.doi.org/10.17487/RFC2616>>. Acesso em: 27 jul. 2016.
- 16 VARVELLO, M. et al. *Is the web HTTP/2 yet?* 2015. Disponível em: <<http://isthewebhttp2yet.com/files/http2-pam16.pdf>>. Acesso em: 27 jul. 2016.
- 17 MITZNER, K. *Complete PCB design using OrCAD Capture and PCB Editor*. 2009. Disponível em: <https://books.google.com.br/books?id=z7RCh8bK6_8C>. Acesso em: 27 jul. 2016.
- 18 PYTHON. *Python language reference*. 2016. Disponível em: <<https://www.python.org/>>. Acesso em: 27 jul. 2016.
- 19 DJANGO SOFTWARE FOUNDATION. *Django*. 2016. Disponível em: <<https://www.djangoproject.com/>>. Acesso em: 27 jul. 2016.
- 20 RED HAT INC. *Ansible documentation*. 2016. Disponível em: <<http://docs.ansible.com/ansible/>>. Acesso em: 27 jul. 2016.