Fernando Abrantes Vita

Relatório de Estágio Supervisionado

Campina Grande, PB

2018

Fernando Abrantes Vita

Relatório de Estágio Supervisionado

Relatório de Estágio Supervisionado submetido à Unidade Acadêmica de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Universidade Federal de Campina Grande - UFCG

Centro de Engenharia Elétrica e Informática

Departamento de Engenharia Elétrica

Orientador: Dr. Alexandre Cunha Oliveira

Campina Grande, PB

2018

Fernando Abrantes Vita

Relatório de Estágio Supervisionado

Relatório de Estágio Supervisionado submetido à Unidade Acadêmica de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Aprovado em 10 de agosto de 2018

Dr. Alexandre Cunha Oliveira
Orientador

Dr. Jaidilson Jó da Silva
Convidado

Campina Grande, PB
2018

Dedico este trabalho a minha família, sem eles eu não estaria aqui.

"O homem é o único animal
que sabe que é irracional."

Lista de ilustrações

Lista de abreviaturas e siglas

API	- Interface de Programação de Aplicação (*Application Programming Interface*)
FFT	- Transformada Rápida de Fourier (*Fast Fourier Transform*)
GPIB	- *General Purpose Interface Bus*
HAL	- Camada de Abstração de *Hardware* (*Hardware Abstraction Layer*)
LAN	- Rede Local (*Local Area Network*)
LEIAM	- Laboratório de Eletrônica Industrial e Acionamento de Máquinas
RMS	- Valor Quadrático Médio (*Root Mean Square*)
SCPI	- *Standard Commands for Programmable Instruments*
THD	- Distorção Hamônica Total (*Total Harmonic Distortion*)
UFCG	- Universidade Federal de Campina Grande
USB	- *Universal Serial Bus*
VISA	- Arquitetura de *Software* de Instrumento Virtual (*Virtual Instrument Software Architecture*)

Sumário

1 Introdução

Este relatório refere-se ao estágio supervisionado desenvolvido pelo aluno da graduação em engenharia elétrica Fernando Abrantes Vita no Laboratório de Eletrônica Industrial e Acionamento de Máquinas (LEIAM), situado na Universidade Federal de Campina Grande (UFCG). O estágio aconteceu de 15/06/2018 até 27/07/2018, totalizando 184 horas de trabalho (6 créditos).

O estágio visa o cumprimento das exigências da disciplina Estágio Curricular, que faz parte da grade curricular do curso de engenharia elétrica da UFCG. O objetivo da disciplina é unir o conteúdo aprendido ao longo do curso com a prática profissional.

1.1 Motivação

Foi desenvolvido, por um servidor técnico da UFCG, dois *drivers*: um para controlar remotamente um osciloscópio e outro para controlar um gerador de sinais. Estes serão incorporados a um projeto já em andamento no LEIAM. Diante disso, antes de incorporá-los ao projeto, é importante que todas as funcionalidades estejam presentes e possam ser devidamente testadas.

Deste modo, é de interesse do laboratório o desenvolvimento de uma interface onde os *drivers* possam ser testados de forma simples e que as funcionalidades necessárias, porém ausentes neles, possam ser implementadas.

1.2 Objetivos

O objetivo deste estágio foi, a partir dos *drivers* existentes, desenvolver um programa de computador com interface gráfica em que remotamente fosse possível testar funcionalidades específicas do osciloscópio e do gerador de sinais, como também corrigir e implementar funcionalidades necessárias e ausentes nos *drivers*. Foi requisitado que a comunicação entre o computador e os instrumentos fosse feita utilizando-se da API do VISA e usando a LAN como meio de comunicação.

1.3 O LEIAM

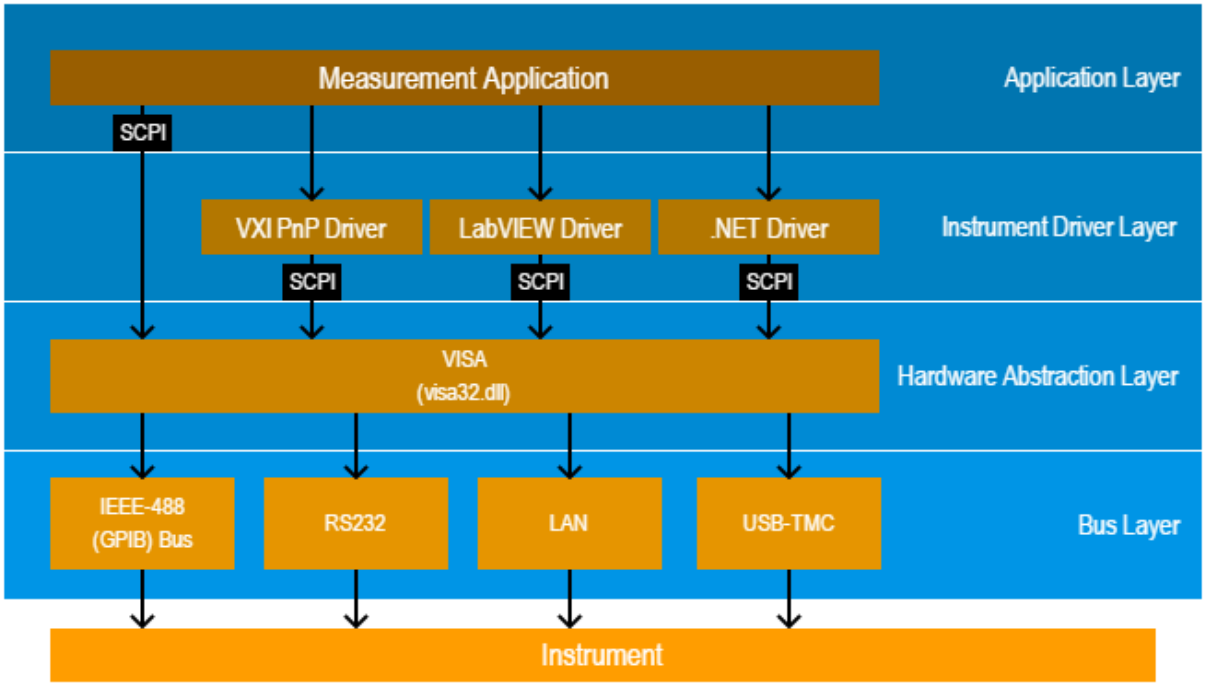
Criado em 1976, o Laboratório de Eletrônica Industrial e Acionamento de Máquinas (LEIAM), local onde o estágio foi realizado, está localizado no bloco CH da Universidade Federal de Campina Grande (UFCG) e pertence ao Departamento de Engenharia Elétrica

(DEE). O LEIAM atua com pesquisa e desenvolvimento, e entre as suas principais áreas de atuação estão: qualidade de energia, fontes alternativas de energia, acionamento e controle de máquinas, e sistemas embarcados.

2 Ferramentas Utilizadas

Para desenvolver uma interface de leitura de dados e controle para o gerador de sinais e osciloscópio, primeiramente é necessário conhecer como a aplicação se comunica e controla os instrumentos utilizados. Na Figura 1, seguindo as setas, é possível ver diferentes possibilidades de como essa comunicação pode ocorrer. Nos tópicos a seguir, serão abordados o SCPI e o VISA, que foram ferramentas utilizadas neste trabalho.

Figura 1 – Modos de comunicação entre a camada de aplicação e o instrumento.



Fonte: (Rohde & Schwarz, 2018).

2.1 SCPI

Observando a Figura 1 é possível ver que todas as formas de comunicação com o instrumento utilizam comandos SCPI. É por meio da interpretação destes comandos, recebidos de forma remota, que o instrumento é configurado e responde as requisições solicitadas.

O SCPI se propõe a ser uma linguagem em comum para servir de interface entre computadores e instrumentos de teste. A ideia é que diferentes instrumentos, independentemente de fabricante, que possuem funções iguais, sejam controlados usando os mesmos comandos. Tais comandos são fáceis de aprender e podem ser utilizados em qualquer linguagem de programação (CONSORTIUM et al., 1999).

Os comandos SCPI apresentam uma estrutura de comando organizada em forma de árvore. Um exemplo de comando SCPI é “SOURce1:VOLTage:UNIT VPP”. Este comando faz com que o valor de tensão mostrado na tela do gerador de sinais seja de pico a pico. Por se tratar de uma estrutura em árvore o UNIT é filho do nó VOLTage, e este último é filho do SOURce. O número 1 ao lado do SOURce representa o número do canal cuja unidade está sendo alterada. Em vez de “VPP”, outras unidades configuráveis são o “VRMS” e o “DBM”, para tensão RMS e decibel-miliwatt respectivamente. Todas as letras que estão em minúsculas são opcionais e podem ser removidas do comando.

Se o objetivo for saber qual a unidade de tensão que está configurada na saída do mesmo canal, basta remover o parâmetro VPP e colocar uma interrogação no final do comando. Assim, a unidade de medição será obtida depois da execução do seguinte comando “SOURce1:VOLTage:UNIT?”.

Detalhes sobre o funcionamento e a sintaxe do SCPI podem ser encontrados no manual publicado (CONSORTIUM et al., 1999) no site da IVI Foundation, ou no manual do programador do instrumento utilizado.

2.2 VISA

Na Figura 1 também é possível ver que a comunicação com o instrumento pode ocorrer através de diferentes barramentos. No passado, por não haver padronização, a comunicação utilizando-se de diferentes barramentos exigiam diferentes implementações. Para resolver esse problema foi desenvolvido o VISA ou o *Virtual Instrument Software Architecture*. Esta arquitetura fornece uma API de comunicação que garante interoperabilidade entre diferentes vendedores, como também que o *software* desenvolvido continue funcionando mesmo que a forma de comunicação com o instrumento mude de LAN para GPIB, RS-232 ou USB (CHEIJ, 2001).

A seguir serão mostradas duas opções que possibilitam a comunicação com a API VISA utilizando a linguagem de programação Python.

2.2.1 PyVISA

PyVISA é um pacote Python, instalável via pip, que executa tanto em Linux, quanto em *Windows* e Mac, tratando-se de um *frontend* para biblioteca VISA. Para o PyVISA funcionar corretamente, é necessário um *backend*. Por padrão, o PyVISA usa como *backend* a biblioteca VISA da *National Instruments* (NI-VISA). Embora não seja recomendado, por não haver testes suficientes, também é possível utilizar uma implementação pura em Python, a “pyvisa-py” (AUTHORS, 2016).

2.2.2 Lantz

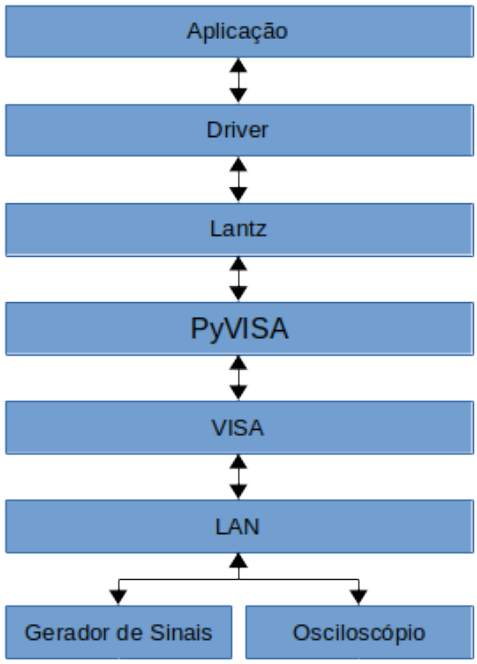
O Lantz é um pacote Python que incorpora o PyVISA e adiciona novos recursos. Dentre estes recursos estão: a criação de logs; o uso de cache, para evitar que um valor configurado seja reescrito desnecessariamente; e acompanhamento de tempos de execução. Além disso, o Lantz traz facilidades para a construção de interfaces gráficas e mantém uma variedade de *drivers* para instrumentos (GRECCO, 2016).

3 Atividades Desenvolvidas

Coube ao estagiário desenvolver um programa de computador que incorporasse as funcionalidades desejadas em uma interface gráfica. Como os *drivers* obtidos estavam escritos em Python 3, esta foi a linguagem escolhida para o desenvolvimento de todo o programa. Para a criação da interface gráfica, foi utilizado o PyQt. Inicialmente foi utilizada a versão 5, porém, por problema de incompatibilidade encontrado durante o desenvolvimento, foi necessário migrar para a versão 4.

Na aplicação, na maior parte dos casos, a origem dos comandos SCPI que chegam até os instrumentos partem dos *drivers*. Porém, em situações específicas, a própria aplicação gera os comandos SCPI e os *drivers* apenas os repassam para o VISA. Não foi necessário se preocupar em como as informações trafegam pela LAN, pois o VISA criou uma camada de abstração entre a aplicação e o instrumento. A arquitetura do *software* desenvolvido pode ser vista na Figura 2 e uma imagem da aplicação desenvolvida pode ser vista na Figura 3.

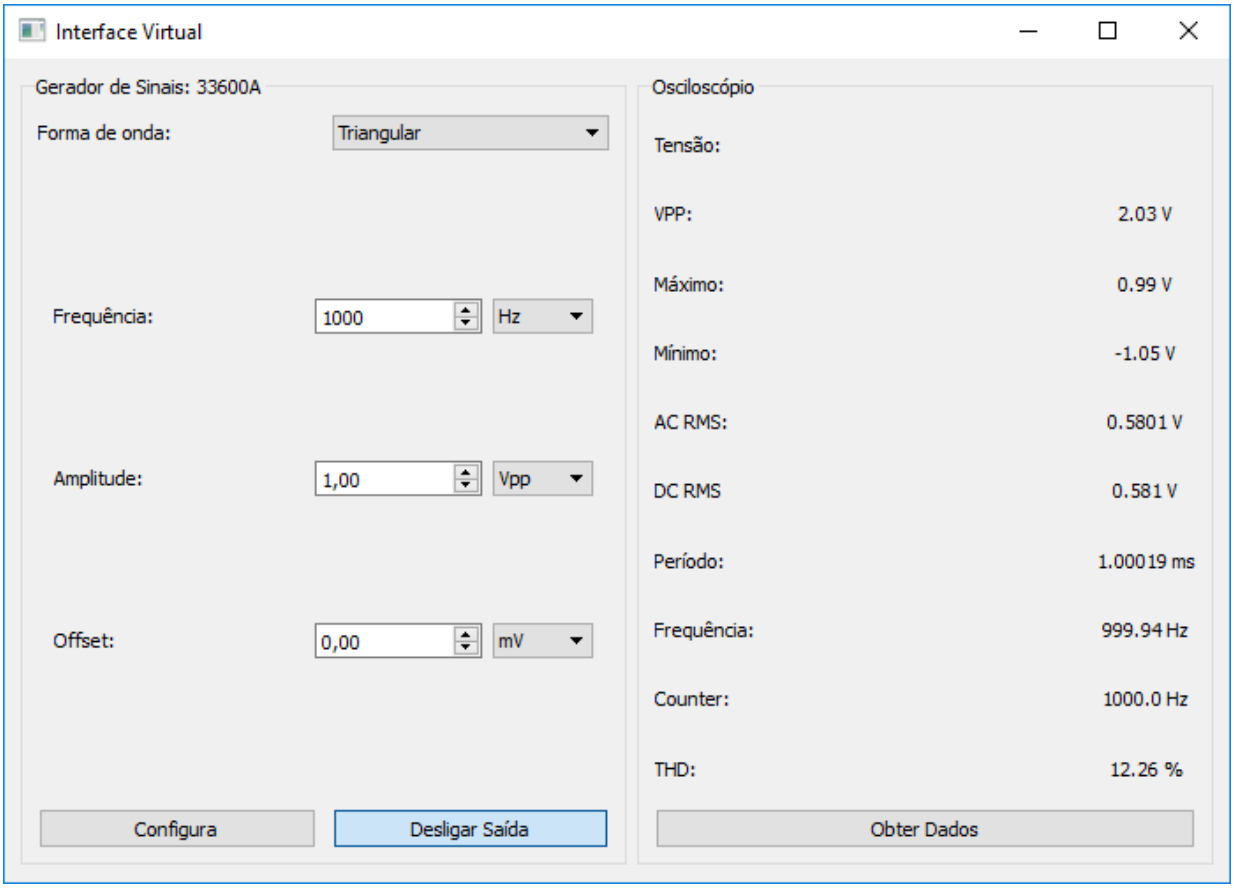
Figura 2 – Arquitetura do *software* desenvolvido.



Fonte: o autor.

Mais detalhes sobre o desenvolvimento de cada módulo podem ser vistos nos tópicos seguintes.

Figura 3 – Interface virtual desenvolvida.



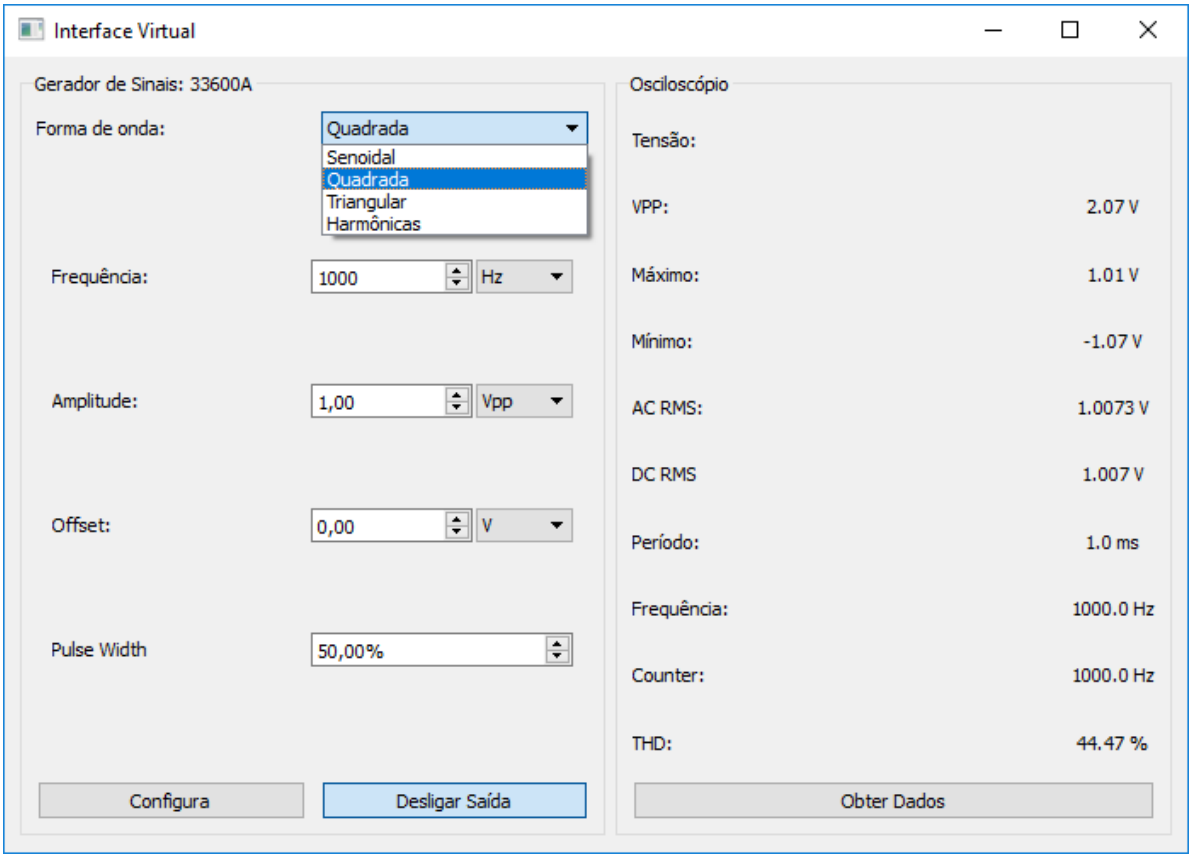
Fonte: o autor.

3.1 Gerador de Sinais

O gerador de sinais utilizado foi o Agilent 33611A(Keysight Technologies, 2014). As funcionalidades requeridas para o gerador de sinais foram a implementação das ondas senoidal, quadrada e triangular para diferentes valores de amplitude, frequência, $offset$, como também diferentes valores de *duty cycles*, para a onda quadrada. Na interface mostrada na Figura 3, por exemplo, uma onda triangular está configurada.

O funcionamento da interface do gerador de sinais é simples, basta inicialmente clicar no menu forma de onda e escolher a forma de onda desejada. Feito isso, devem ser preenchidos os valores desejados de frequência, amplitude e $offset$, e no caso da onda quadrada, a largura de pulso (*Pulse Width*). No momento de inserir os parâmetros desejados, diferentes unidades podem ser escolhidas a partir de uma caixa de seleção. As unidades de frequência disponíveis para escolha são μHz, mHz, Hz, kHz e MHz. Para amplitude as unidades permitidas são mV_{pp}, V_{pp}, mV_{RMS}, V_{RMS} e dBm. Já para o $offset$, as unidades de tensão possíveis são o mV e o V. Por fim, para a onda quadrada, a largura de pulso é sempre um valor percentual. A interface de configuração da onda quadrada pode ser vista na Figura 4.

Figura 4 – Configuração de onda quadrada e resultados obtidos remotamente por meio do osciloscópio.



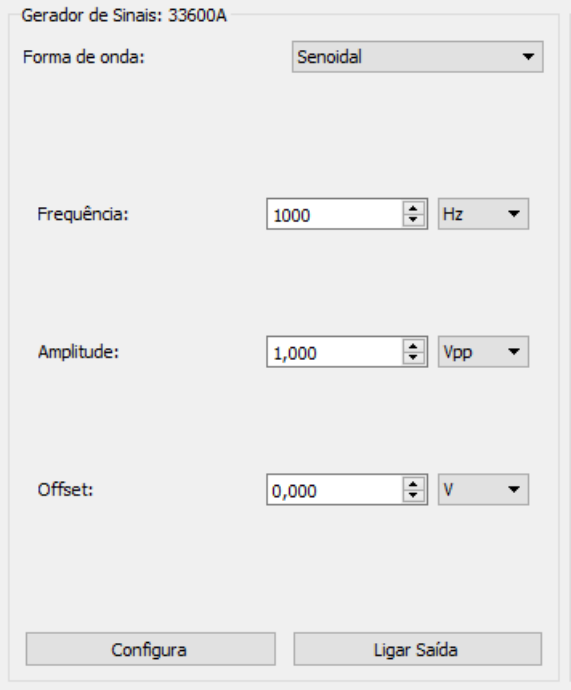
Fonte: o autor.

Depois de configurados os parâmetros desejados é necessário clicar no botão configura. Apenas após o clique do botão é que o computador envia as configurações realizadas para o gerador de sinais. O segundo botão na interface do gerador é o Ligar/Desligar Saída. Ao ser clicado, o estado da saída do gerador é invertido, liga caso esteja desligado e desliga caso esteja ligado. Sendo assim, caso a saída do gerador esteja desligada, são necessários 4 passos para configurar a forma de onda desejada: 1- seleção da forma de onda; 2- escolha dos parâmetros desejados; 3- clique no botão "configura" para que a configuração da onda e dos parâmetros sejam enviados ao gerador; e 4- clique no botão ligar saída para que a onda apareça na saída do gerador. A configuração de uma forma de onda senoidal com a saída desligada pode ser vista da Figura 5.

Com o objetivo de usar o gerador de sinais para testar funcionalidades do osciloscópio foi implementado um quarto modo de configuração de onda, que foi chamado de "harmônicas". Este modo permite que o usuário configure uma onda senoidal (fundamental) e adicione a ela até sua décima segunda harmônica. A interface de configuração das "harmônicas" e o resultado obtido no osciloscópio podem ser vistos nas Figuras 6 e 7.

Os três primeiros parâmetros de configuração da onda com harmônicas são iguais

Figura 5 – Interface de configuração de onda senoidal.



Fonte: o autor.

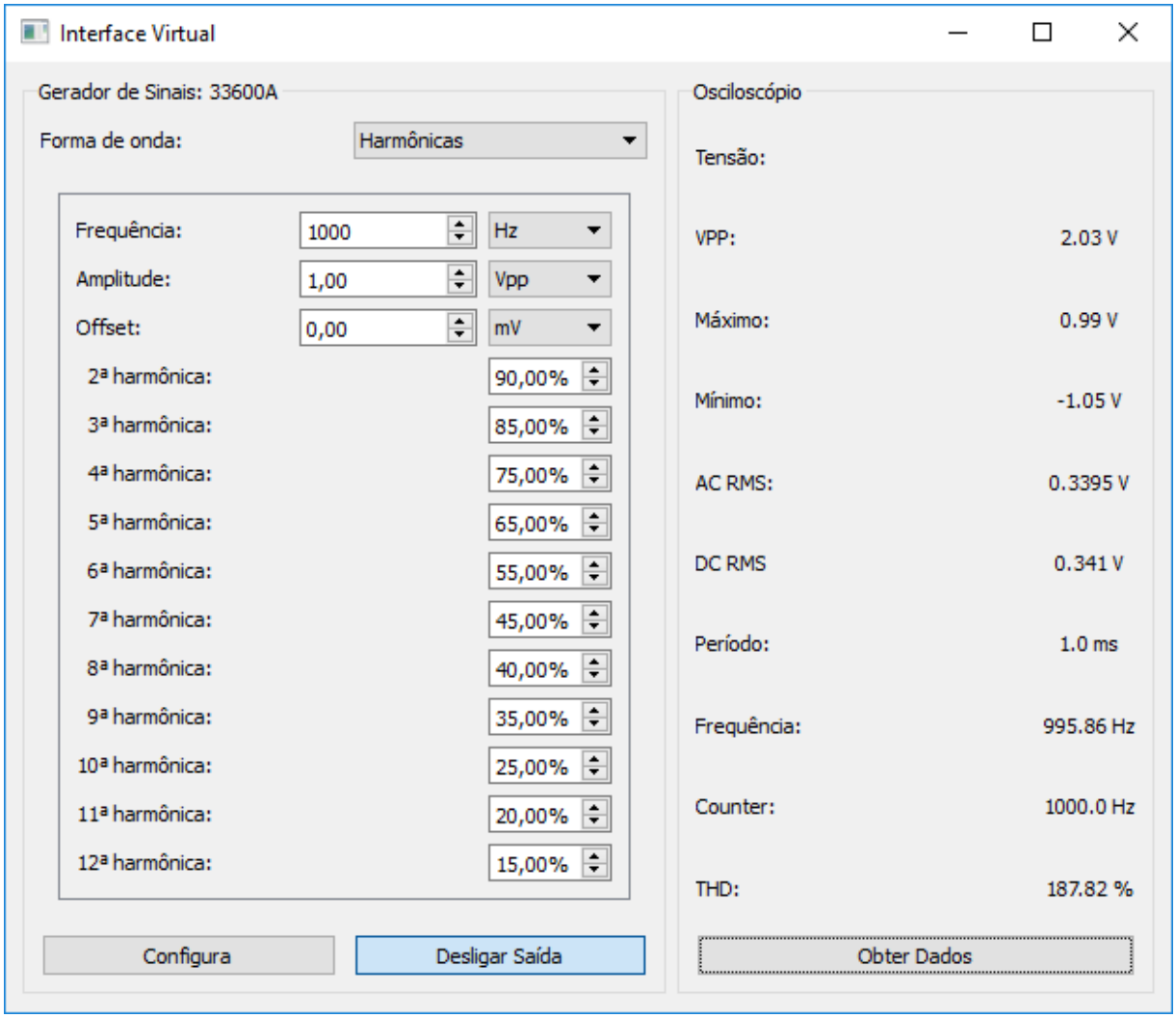
ao da onda senoidal, exceto por uma restrição, a única unidade de configuração possível para a amplitude do sinal é V_{pp}. O valor percentual de cada harmônica está referenciado ao valor de pico da onda fundamental. Suponhamos que a tensão da onda fundamental seja 1 V_{pp} e a terceira harmônica esteja marcada como 20%, isto significa que valor de tensão da terceira harmônica é de 0,2 V_{pp}. É relevante mencionar que o parâmetro configurado de amplitude não representa a amplitude da onda fundamental, e sim, a amplitude total da onda final, ou seja, depois da soma de todas as harmônicas. Uma outra funcionalidade implementada é o cálculo da THD da onda harmônica gerada. Sempre que uma onda harmônica é configurada no osciloscópio, o software faz o cálculo da THD esperada e imprime esse valor no console do Python. Esta funcionalidade é útil para conferir se a medição do osciloscópio está ocorrendo da forma esperada. Um membro do LEIAM ao testar a interface de configuração de onda com harmônicas demonstrou interesse em utilizar a ferramenta em outros projetos do laboratório.

As configurações das ondas senoidal, quadrada e triangular são feitas utilizando serviços do *driver*, já os comandos necessários para a geração de onda com "harmônicas", são enviados pela camada de aplicação. Os comandos que são enviados para configurar uma onda do tipo harmônica são:

1. gerador.function = 'ARB'

2. gerador.write('SOUR1:DATA:VOL:CLE')

Figura 6 – Configuração de onda com harmônicas e resultados obtidos remotamente por meio do osciloscópio.



Fonte: o autor.

3. gerador.write('DATA:ARB harm, ' + str(onda.tolist())[1:-1])

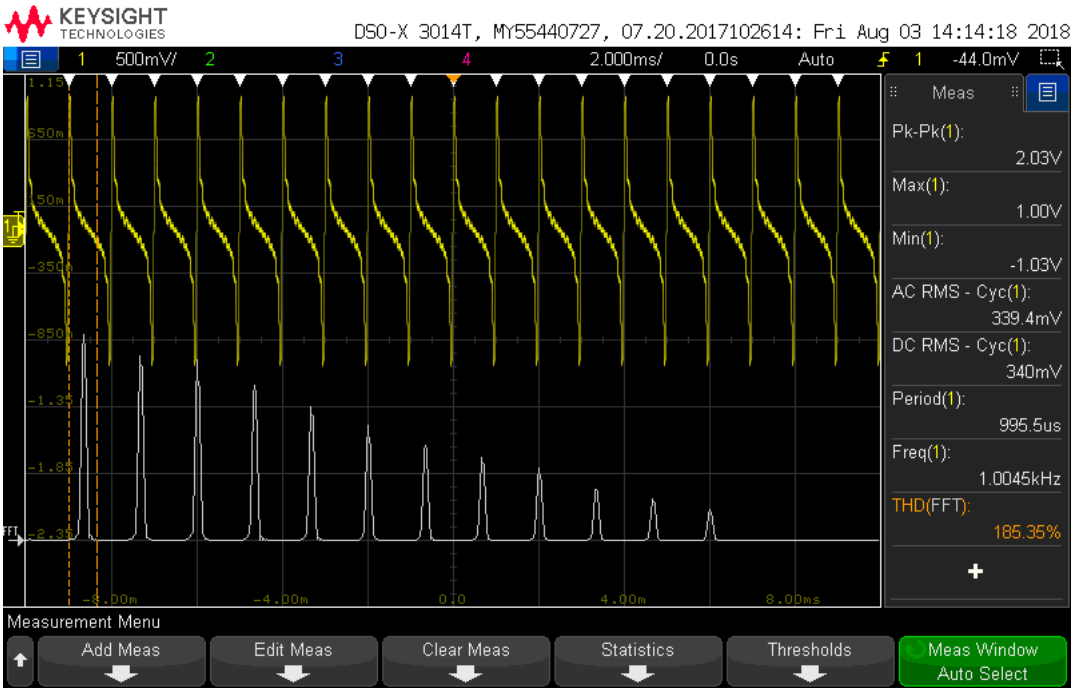
4. gerador.write(':SOUR1:FUNC:ARB harm')

5. gerador.write('DISP:UNIT:ARBR FREQ')

6. gerador.write('FUNC:ARB:FREQ '.format(frequency))

Quem executa o comando 1 é o *driver*. Neste caso, o objetivo é configurar o gerador de sinais para gerar uma forma de onda do tipo arbitrária. O comando 2 limpa as informações de ondas anteriores da memória. O comando 3 envia para memória volátil uma lista de pontos que representam a onda que será gerada, a função str está retornando uma lista de valores separados por vírgulas. Esta lista representa a onda configurada pelo menu harmônicas. O comando 4 seleciona a série de pontos que foi escrita na memória

Figura 7 – FFT e forma de onda gerada a partir da configuração no menu harmônicas vistas na Figura 6 (um offset foi dado para facilitar a visualização das duas curvas).



Fonte: o autor.

e os direciona para a saída do dispositivo. É possível configurar a frequência com que a onda é gerada de três formas: 1- alterando a taxa de amostragem (parâmetro SRATE); 2- escolhendo a frequência com que toda a nuvem de pontos inteira é lida (parâmetro FREQ); ou configurando o período de tempo que levará para percorrer toda a nuvem de pontos (parâmetro PER). Neste sentido, o comando 5 mostra na tela do gerador a frequência com que a onda se repete, já o comando 6 configura a frequência desejada.

Inicialmente a interface do computador com o gerador de sinais foi feita utilizando um *driver* que incorporava a biblioteca PyVISA. Entretanto, durante a realização do estágio, uma segunda versão do *driver* do gerador de sinais foi feita pelo servidor técnico da UFCG, Lúcio Fernando, e este *driver* passou a utilizar a biblioteca Lantz em vez do PyVISA, conforme a Figura 2. Deste modo, todo o trabalho realizado precisou ser migrado para o novo *driver* que incorpora o Lantz.

Todas as funcionalidades pedidas foram implementadas com sucesso. O maior desafio enfrentado nesta etapa foi obter uma instalação funcional da biblioteca Lantz. Mais de 35 horas de trabalho foram utilizadas nesta atividade. Por esse motivo, todo o passo a passo para a criação de um ambiente virtual Python com Lantz foi documentado para que esta dificuldade não exista mais. Os comandos utilizados para a instalação do Lantz podem ser vistos do apêndice A. Por questões de compatibilidade, o uso da biblioteca Lantz foi o motivo pelo qual a interface gráfica teve que ser migrada de PyQt5 para PyQt4.

3.2 Osciloscópio

O osciloscópio utilizado foi o DSOX3014T (Keysight Technologies, 2017). As funcionalidades requeridas para implementação da interface remota no osciloscópio foram a medição de tensão de pico a pico, a tensão RMS, a frequência do sinal e a Distorção Harmônica Total (THD). A interface desenvolvida para a obtenção dos dados do osciloscópio pode ser vista na Figura 3.

O botão “Obter Dados”, ao ser clicado, faz com que uma série de comandos sejam executados. De forma resumida, os passos ocorridos após o clique do botão são: o software se conecta ao osciloscópio; restaura uma configuração de setup salvo na memória conforme solução 1 que será abordado a seguir; inicia o módulo counter; executa o *auto-scale*; espera 2 segundos para evitar que o próximo comando seja executado com o *auto-scale* ainda em execução; e zera todas as estatísticas das medições executando o comando “":MEASure:STATistics:RESet”. Após esses passos, a leitura dos parâmetros são iniciadas obedecendo a seguinte sequência: tensão de pico a pico, tensão máxima, tensão mínima, os valores AC RMS e DC RMS, período, frequência, o Counter e a THD. Por fim, a conexão com o osciloscópio é desfeita.

Para que os valores apresentados na tela sejam atualizados é necessário um novo clique por parte do usuário.

Antes da medição da THD ocorrer, a escala te tempo é aumentada em 3 posições para que a onda completa se repita algumas vezes na tela do osciloscópio e melhore a precisão da FFT. Após a medição, a posição da escala é restaurada.

Nas Figuras 6 e 7 é possível observar as configurações feitas no gerador e a onda gerada, como também os resultados obtidos remotamente e a medição dos mesmos parâmetros sendo mostrados na tela do osciloscópio.

A configuração do osciloscópio demonstrou ser mais complexa do que a do gerador de sinais, em alguns casos demandando vários comandos para fazer o que era desejado. Assim como no gerador de sinais, o *driver* do osciloscópio incorpora a biblioteca Lantz. A interface desenvolvida para o gerador de sinais demonstrou ser de grande auxílio para testar o desenvolvimento da interface do osciloscópio.

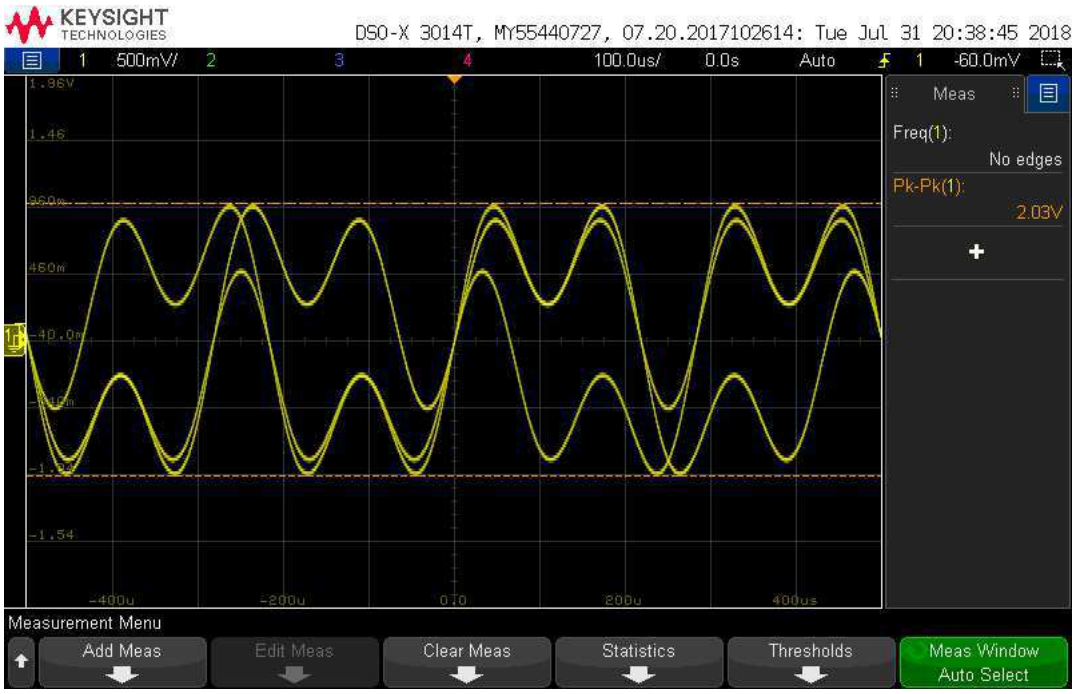
Novamente, todas as funcionalidades desejadas foram implementadas com sucesso, porém nem tudo da forma ideal. O primeiro problema está relacionado ao *auto-scale* do osciloscópio e o segundo se trata de uma dificuldade na leitura remota do valor da THD. Estas duas dificuldades serão abordadas individualmente nos tópicos a seguir.

3.2.1 *Auto-Scale*

O segundo comando enviado, quando o usuário pede para obter os dados do osciloscópio, é o *auto-scale*. Ocorre que nem sempre o *auto-scale* por si só é capaz de estabilizar de forma correta a onda na tela. Quando este problema acontece, a medição de alguns valores pode não ocorrer ou poderá acontecer de forma incorreta. Por exemplo, utilizando a interface "harmônicas" do gerador, criando-se uma onda cuja a fundamental, a terceira e a sétima harmônica possuem o mesmo nível de energia, a onda gerada não fica estabilizada na tela do osciloscópio e, neste caso, a medição da frequência não ocorre, como pode ser visto na Figura 8. Na Figura 9 é possível ver o formato real da onda gerada.

Para corrigir este problema é necessário conhecer a natureza do sinal que está sendo medido ou fazer uso de técnicas mais avançadas e realizar várias tentativas até conseguir identificar que o sinal está sendo capturado de forma correta.

Figura 8 – Resultado do *auto-scale* do osciloscópio depois de aplicada uma onda com fundamental, terceira e sétima harmônicas com o mesmo nível de energia.

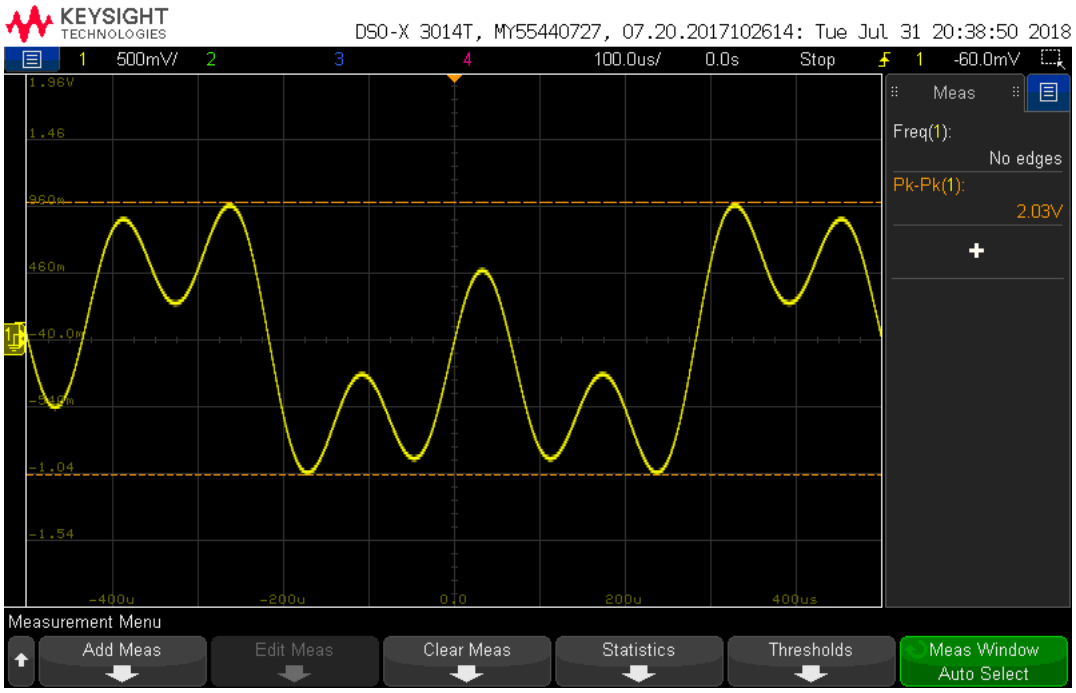


Fonte: o autor.

3.2.2 Leitura da THD

A segunda dificuldade ocorreu no momento da medição da THD. Ao enviar os comandos conforme documentados no manual do programador (Keysight Technologies, 2018) para configurar e obter a THD, o seguinte erro aparece na tela do osciloscópio: "*The selection is not available in this instrument*". O erro persiste mesmo ao utilizar a própria ferramenta do fabricante, o *Command Expert*. Na Figura 11 é possível ver os

Figura 9 – Forma de onda real da Figura 8.



Fonte: o autor.

comandos testados e parte dos erros gerados. A dificuldade acontece apenas na tentativa de configurar e ler a THD utilizando os comandos SCPI.

Figura 10 – Alguns dos comandos testados no *Command Expert* na tentativa de obter a THD de forma remota.

Status	Instrument	Code	Results
✓	DSO-X	(Connect "DSO-X", "TCPIP0::172.22....	
✓	DSO-X	:MEASure:FREQuency? CHANnel1	1000.6
✓	DSO-X	:FFT:SOURce1 CHANnel1	
✓	DSO-X	:FFT:DISPlay 1	
✗	DSO-X	:MEASure:FFT:THD AUTO	SCPI Error: -221,Settings conflict | SCPI Error: -221,Settings conflict
✗	DSO-X	:MEASure:FFT:THD AUTO,FFT	SCPI Error: -220,Parameter error | SCPI Error: -221,Settings conflict
✗	DSO-X	:MEASure:FFT:THD? AUTO	COMException:VI_ERROR_TMO: A timeout occurred | CommunicationTimeoutException:SCPI connection attempt timed out | SCPI Error: -220,I
✗	DSO-X	:MEASure:FFT:THD? AUTO,FFT	COMException:VI_ERROR_TMO: A timeout occurred | CommunicationTimeoutException:SCPI connection attempt timed out | SCPI Error: -220,I
✗	DSO-X	:MEASure:FFT:THD MANual,1000,FFT	SCPI Error: -220,Parameter error | SCPI Error: -221,Settings conflict
✗	DSO-X	:MEASure:FFT:THD? MANual,1000,FFT	COMException:VI_ERROR_TMO: A timeout occurred | CommunicationTimeoutException:SCPI connection attempt timed out | SCPI Error: -220,I

Fonte: o autor.

Uma vez que a THD é um requisito obrigatório no projeto, esta limitação precisava ser solucionada. Para isso desenvolveu-se duas soluções, uma primeira mais complexa e não ideal, que é a que foi implementada neste projeto. E a segunda que foi uma sugestão do próprio suporte técnico do fabricante do equipamento. Nos tópicos a seguir serão abordadas cada solução de forma individual.

Solução 1

Uma vez que a o problema ocorria apenas ao tentar fazer a medição e configuração de forma remota, então, manualmente, a medição da THD foi ativada na tela de medições

do osciloscópio. Depois, todo o estado de configuração do osciloscópio com a medição da THD ativada, foi salvo na memória do próprio instrumento. Restaurando essa configuração é possível voltar para um estado inicial onde se sabe que a medição da THD está visível no menu *measurements*. A visibilidade no menu *measurements* é útil, mesmo de forma remota, pois o comando ":MEASure:RESults?" retorna uma lista de valores separados por vírgula contendo as estatísticas de todas as medições aparentes neste menu. Deste modo, ao receber esta lista de valores, basta usar um filtro para reconhecer qual deles é a da THD e capturar o valor da leitura. Esta foi a implementação utilizada neste projeto. Ela não é ideal pois além de ser mais complexa do que deveria, ela depende de uma arquivo gravado na memória do osciloscópio que pode ser apagado acidentalmente. Insatisfeito com a solução, resolveu-se fazer contato com o suporte do fabricante.

Solução 2

Após o contato com o suporte técnico, este sugeriu que em vez de utilizar a FFT como fonte para a THD, fosse utilizado uma FFT calculada a partir do módulo MATH. A execução dos comandos sugeridos pelo fabricante pode ser vista na Figura 11. A solução proposta funcionou bem e é bem mais simples que a solução anterior. Entretanto, apesar de testada, tal solução não foi integrada a aplicação desenvolvida devido ao término da vigência do estágio. De toda forma, provavelmente ela será incorporada ao *driver* testado.

Figura 11 – Sugestão de solução do suporte do fabricante, executado no *Command Expert*, para obter a THD de forma remota.

Status	Instrument	Code	
✓	DSO-X	(Connect "DSO-X", "TCPIP0::172.22.22.4::hislip0::INSTR", "InfiniiVision 3...	
✓	DSO-X	*IDN?	KEYSIGHT TECHNOLOGIES,DSO-X 3014T,MY55440727,07.20.2017102614
✓	DSO-X	:MEASure:FREQuency? CHANnel1	999.9
✓	DSO-X	:FUNCtion1:OPERation FFT	
✓	DSO-X	:MEASure:FFT:THD? MANual,1000,MATH1	47.781

Fonte: o autor.

4 Considerações Finais

O objetivo deste documento foi apresentar as atividades realizadas no estágio supervisionado, requisito fundamental para a conclusão do curso de engenharia elétrica da Universidade Federal de Campina Grande.

As contribuições do estagiário foram: 1- o desenvolvimento de uma interface virtual para controlar e obter dados do gerador de sinais e do osciloscópio; 2- testar, corrigir problemas e adicionar novas funcionalidades aos *drivers* utilizados.

O estágio cumpriu com o seu objetivo de estimular o desenvolvimento de competências próprias da atividade profissional. Isso foi feito inserindo o aluno dentro de um projeto real, tendo este que cumprir metas, obedecer prazos e enfrentar dificuldades similares àquelas encontradas no dia a dia do engenheiro no mercado de trabalho.

O estagiário não apenas reconhece que desenvolveu novas habilidades e obteve novos conhecimentos, como também julga ser gratificante fazer uso do conhecimento adquirido com empenho na academia para desenvolver soluções que trazem algum benefício prático real.

Referências

AUTHORS, P. *PyVISA: Control your instruments with Python*. 2016. Citado na página 4.

CHEIJ, D. A software architecture for building interchangeable test systems. In: *2001 IEEE Autotestcon Proceedings. IEEE Systems Readiness Technology Conference. (Cat. No.01CH37237)*. [S.l.: s.n.], 2001. p. 16–22. ISSN 1080-7225. Citado na página 4.

CONSORTIUM, S. et al. Standard commands for programmable instruments (scpi), volume 1: Syntax and style. *USA, May*, 1999. Citado 2 vezes nas páginas 3 e 4.

GRECCO, H. *Lantz: Simple yet powerful instrumentation in Python*. 2016. Disponível em: <https://lantz.readthedocs.io/en/0.3/>. Acesso em: 01 ago. 2018. Citado na página 5.

Keysight Technologies. *33600A Series Trueform Waveform Generators - Data Sheet*. [S.l.], 2014. Citado na página 8.

Keysight Technologies. *Osciloscópios InfiniiVision 3000T da Série X - Folha de Dados*. [S.l.], 2017. Citado na página 13.

Keysight Technologies. *Keysight InfiniiVision 3000T X-Series Oscilloscopes - Programmer's Guide*. [S.l.], 2018. Citado na página 14.

Rohde & Schwarz. *Remote Control and Instrument Drivers*. 2018. Disponível em: <https://www.rohde-schwarz.com/driver-pages/remote-control/drivers-remote-control_110753.html>. Acesso em: 01 ago. 2018. Citado na página 3.

Apêndices

APÊNDICE A – Instalação do Lantz

Os comandos abaixo foram realizados em uma máquina com sistema operacional Windows em um prompt Anaconda. As instruções a seguir não abordam a instalação do anaconda.

```
    # Testado com o anaconda 1.6.3 e com o conda 4.5.4

> set CONDA_FORCE_32BIT=1
> conda create -n nomedoambientevirtual python=3.4 pip pyqt=4.11
> conda activate nomedoambientevirtual
> pip install lantz
```