INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA - CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO DE GRADUAÇÂO EM ENGENHARIA ELETRÔNICA

JOANA PAULA ABREU WASSERBERG

DESENVOLVIMENTO DE UMA JIGA DE TESTE PARA VALIDAÇÃO FUNCIONAL DE PLACAS CONTROLADORAS DE ARMÁRIOS INTELIGENTES

FLORIANÓPOLIS, 2026

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA - CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO DE GRADUAÇÂO EM ENGENHARIA ELETRÔNICA

JOANA PAULA ABREU WASSERBERG

DESENVOLVIMENTO DE UMA JIGA DE TESTE PARA VALIDAÇÃO FUNCIONAL DE PLACAS CONTROLADORAS DE ARMÁRIOS INTELIGENTES

Trabalho de Conclusão de Curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina como parte dos requisitos para obtenção do título de Bacharel em Engenharia Eletrônica

Orientador:
Prof. Joabel Moia, Dr. Eng.

FLORIANÓPOLIS, 2026

Ficha de identificação da obra elaborada pelo autor.

Wasserberg, Joana Paula Abreu
Desenvolvimento de uma jiga de teste para validação **funcional de placas controladoras de armários** inteligentes **/ Joana Paula Abreu Wasserberg; orientação de** Joabel **Moia. - Florianópolis, SC,** 2026.
85 p.

Trabalho de Conclusão de Curso (TCC) - Instituto Federal **de Santa Catarina, Câmpus** Florianópolis. Bacharelado **em Engenharia Eletrônica. Departamento** Acadêmico **de** Eletrônica.
Inclui Referências.

1. Jiga de testes. 2. Automação industrial. 3. Teste **funcional. 4. Sistemas embarcados. 5.** Armários inteligentes. **I. Moia, Joabel. II. Instituto Federal** de Santa **Catarina. III. Desenvolvimento de uma** jiga de **teste para validação funcional de placas** controladoras **de armários** inteligentes.

DESENVOLVIMENTO DE UMA JIGA DE TESTE PARA VALIDAÇÃO FUNCIONAL DE PLACAS CONTROLADORAS DE ARMÁRIOS INTELIGENTES

JOANA PAULA ABREU WASSERBERG

Este trabalho foi julgado adequado para obtenção do título de Engenheiro em Eletrônica e aprovado na sua forma final pela banca examinadora do Curso Engenharia Eletrônica do Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina.

Florianópolis, 19 de fevereiro de 2026.

Banca examinadora:

gov.br
Documento assinado digitalmente
JOABEL MOIA
Data: 19/02/2026 09:01:06-0300
Verifique em https://validar.iti.gov.br

Joabel Moia, Dr.
Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina

gov.br
Documento assinado digitalmente
RENAN AUGUSTO STARKE
Data: 19/02/2026 16:03:44-0300
Verifique em https://validar.iti.gov.br

Renan Augusto Starke, Dr.
Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina

ICP-Edu
Documento assinado digitalmente
Mateus Nava Mezaroba
Data: 19/02/2026 12:26:04-0300
CPF: ***.100.130-**
Verifique as assinaturas em https://v.ufsc.br

Mateus Nava Mezaroba, Me.
Universidade Federal de Santa Catarina

Em memória de Luciano Brame Ferreira,
sua dedicação à indústria e inovação em
Santa Catarina deixa um legado que será
sempre honrado.

AGRADECIMENTOS

Agradeço, primeiramente, às pessoas que foram minha base. À minha mãe, Sandra, cuja determinação e persistência sempre foram meus maiores exemplos e contribuíram muito para meu processo de aprendizado durante a graduação. Ao meu pai, Gilberto, pelo exemplo e incentivo incansável ao estudo e ao conhecimento. Ao meu irmão, Davi, por abrir minha mente para o mundo da tecnologia e sempre acreditar no meu potencial.

À empresa Brametec, agradeço pela oportunidade de crescimento profissional e pelo apoio para que eu pudesse alinhar o desenvolvimento deste trabalho às demandas reais da indústria, contribuindo com meus conhecimentos para o avanço tecnológico da organização. Agradeço em especial aos meus colegas: ao Renato, pelas ideias inspiradoras, e ao Leonardo, pelo companheirismo e por suas contribuições durante o desenvolvimento deste projeto.

Aos colegas e amigos que conheci durante essa jornada acadêmica, principalmente àqueles com quem mais compartilhei conhecimento. A amizade e o apoio de cada um de vocês fizeram com que esta caminhada fosse mais leve e prazerosa.

Ao Instituto Federal de Santa Catarina, pelo ensino público de qualidade. Aos meus professores, por todo ensinamento, em especial ao meu orientador, Joabel, por sua orientação fundamental ao longo deste trabalho. Sua dedicação, paciência, conhecimento e bom humor contribuíram imensamente para o meu desenvolvimento pessoal e acadêmico.

Por fim e não menos importante, agradeço ao meu companheiro de vida, Matheus, pelo amor, dedicação e incentivo incondicional. Obrigado pelas palavras de carinho, pela compreensão e por ser meu ponto de apoio. Foram tantos os momentos de dúvidas que a sua confiança em mim foi o que me fez seguir em frente.

Os cientistas estudam o mundo como ele é;
os engenheiros criam o mundo que
nunca existiu.
Theodore von Kármán

RESUMO

Com a crescente complexidade dos sistemas eletrônicos e o aumento da demanda produtiva na indústria, a automação dos processos de validação torna-se fundamental para garantir confiabilidade, repetibilidade e rastreabilidade. Este trabalho apresenta o projeto e desenvolvimento de uma jiga de testes automatizada para validação funcional de placas controladoras de armários inteligentes (*Smart lockers*) da empresa Brametec, cujo processo de testes era anteriormente realizado de forma manual, configurando um gargalo produtivo suscetível a falhas humanas e com rastreabilidade limitada. A solução proposta integra *hardware*, *firmware* e *software* em uma arquitetura híbrida, utilizando um computador de placa única Raspberry Pi 4 para a orquestração do processo, interface homem-máquina e gerenciamento de dados, e um microcontrolador ATmega328PB para o controle dos sinais e execução dos testes de baixo nível. A interface física da jiga baseia-se na técnica de cama de pregos (*Bed of Nails*), empregando agulhas de teste retráteis de ponta serrilhada, garantindo contato elétrico confiável com a placa sob teste. A metodologia adotada concentra-se no teste funcional de circuito (FCT), abrangendo a gravação automática de *firmware*, a verificação dos níveis de tensão, a validação da comunicação serial e a análise lógica de 32 canais de acionamento de solenoides e respectivos sensores, por meio de simulação eletrônica de cargas. Os resultados obtidos em bancada demonstraram a eficácia da jiga na detecção de falhas de manufatura, a eliminação da subjetividade dos testes manuais e a redução do tempo de validação. Além disso, a geração automática de registros de teste assegura a rastreabilidade do processo, fornecendo subsídios para a melhoria contínua da qualidade e da eficiência produtiva.

Palavras-chave: Jiga de testes. Automação industrial. Teste funcional. Sistemas embarcados. Armários inteligentes.

ABSTRACT

With the increasing complexity of electronic systems and the growing production demand in the industry, the automation of validation processes has become essential to ensure reliability, repeatability, and traceability. This work presents the design and development of an automated test jig for the functional validation of controller boards used in smart lockers developed by the company Brametec, whose testing process was previously performed manually, constituting a production bottleneck susceptible to human error and limited traceability. The proposed solution integrates hardware, firmware, and software within a hybrid architecture, employing a Raspberry Pi 4 single-board computer for process orchestration, human–machine interface, and data management, and an ATmega328PB microcontroller for signal control and execution of low-level tests. The physical interface of the test jig is based on the Bed of Nails technique, using spring-loaded test probes with serrated tips to ensure reliable electrical contact with the device under test. The adopted methodology focuses on functional circuit testing (FCT), including automatic firmware programming, voltage level verification, serial communication validation, and logical analysis of 32 solenoid actuation channels and their respective sensors through electronic load simulation. The results obtained from bench tests demonstrated the effectiveness of the test jig in detecting manufacturing defects, eliminating the subjectivity associated with manual testing, and reducing overall validation time. Furthermore, the automatic generation of test records ensures process traceability, providing valuable support for continuous improvement in product quality and production efficiency.

Keywords: Test jig. Industrial automation. Functional circuit test. Embedded systems. Smart lockers.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ASCII	*American Standard Code for Information Interchange*
ATE	*Automated Test Equipment*
BOD	*Brown-out Detection*
CPU	*Central Processing Unit*
DUT	*Device Under Test*
EEPROM	*Electrically Erasable Programmable Read-Only Memory*
FCT	*Functional Circuit Test*
FSM	*Finite State Machine*
GND	*Ground*
GPIO	*General Purpose Input/Output*
GUI	*Graphical User Interface*
I/O	*Input/Output*
ICSP	*In-Circuit Serial Programming*
ICT	*In-Circuit Test*
IHM	Interface Homem-Máquina
MCU	*Microcontroller Unit*
MISO	*Master In Slave Out*
MOSFET	*Metal-Oxide-Semiconductor Field-Effect Transistor*
MOSI	*Master Out Slave In*
PCB	*Printed Circuit Board*
PCI	Placa de Circuito Impresso
PRU	*Programmable Real-Time Unit*
PTH	*Pin Through Hole*
RAM	*Random Access Memory*
RST	*Reset*

RTOS	*Real-Time Operating System*
SBC	*Single Board Computer*
SCK	*Serial Clock*
SPI	*Serial Peripheral Interface*
USB	*Universal Serial Bus*

SUMÁRIO

1 INTRODUÇÃO

Com a crescente necessidade de automação nos processos industriais, apoiados em sistemas eletrônicos cada vez mais complexos e confiáveis, surge a demanda por métodos que validem efetivamente o funcionamento dos produtos antes de sua comercialização. Nesse cenário, as jigas de testes apresentam-se como uma solução inteligente, permitindo a execução de testes automatizados em circuitos eletrônicos por meio de um sistema integrado. Essa abordagem viabiliza a análise comportamental do produto para fins estatísticos e de controle de qualidade, reduzindo atividades manuais repetitivas, mitigando falhas humanas e otimizando o custo-benefício das linhas de produção.

Segundo o ISTQB (2016), a automação de testes tem como principais objetivos o ganho de eficiência, o aumento da cobertura dos testes e a geração automática de documentação. Equipamentos de teste automatizados aprimoram significativamente os processos industriais ao fornecerem resultados padronizados e rápidos, além de permitirem a verificação de um maior número de parâmetros e a criação de documentação para rastreabilidade de testes executados. Portanto, a otimização da fase de testes não apenas eleva a qualidade e a confiabilidade dos produtos, mas também impulsiona um crescimento corporativo mais assertivo e sustentável.

A automação deste processo visa mitigar o que Rodrigues (2009) identifica como a principal vulnerabilidade dos ensaios manuais: a suscetibilidade a falhas humanas. A conexão repetitiva de instrumentos e a interpretação visual de resultados, típicas do método não automático, configuram-se como fontes de erro que comprometem a confiabilidade final do produto.

A Brametec é uma empresa de tecnologia especializada no desenvolvimento e fabricação de armários inteligentes (*Smart Lockers*), integrando *hardware* e *software* para soluções de armazenamento. Com a recente expansão de mercado e o consequente aumento na demanda produtiva, o processo fabril da empresa deparou-se com novos desafios, especialmente na etapa de controle de qualidade.

Atualmente, a validação do *hardware* caracteriza-se como um gargalo produtivo: é um processo manual, moroso e que exige operadores com conhecimento técnico específico para garantir que atenda às especificações de testes do produto. A escassez de tempo para testes manuais minuciosos e a suscetibilidade à falha humana aumentam o risco de defeitos passarem despercebidos, gerando custos elevados com retrabalho de componentes que não foram testados até a fase final de montagem.

Visando solucionar esses problemas e garantir a escalabilidade da produção, este trabalho propõe o projeto e desenvolvimento de uma jiga de teste automati-

zada. O equipamento será capaz de gravar o *firmware*, validar a comunicação e testar a integridade dos pinos de entrada e saída de forma autônoma. A implementação deste sistema visa, não apenas agilizar o processo, mas assegurar a confiabilidade das placas eletrônicas, eliminar a subjetividade dos testes manuais e prover rastreabilidade das placas eletrônicas através do registro automático das operações.

Para atender a essa demanda, propõe-se o desenvolvimento de um sistema composto por uma estrutura mecânica dedicada, projetada para garantir um encaixe preciso e o contato elétrico confiável com o dispositivo sob teste (DUT - *Device Under Test*) nos pontos de validação. A arquitetura de *hardware* deverá integrar uma unidade de processamento central capaz de gerenciar a instrumentação e a aquisição de dados. Em relação ao *software*, o foco será prover uma interface gráfica intuitiva para o operador e uma estrutura de código modular, assegurando um sistema confiável e facilitando futuras manutenções e atualizações

1.1 Justificativa

Considerando o atual cenário competitivo entre empresas de tecnologia, em que existe uma grande busca para realizar entregas mais ágeis e com alta confiabilidade, é essencial encontrar soluções que possam reduzir os custos operacionais, sendo que uma das etapas mais críticas do processo de produção de um equipamento eletrônico é a realização de um teste funcional do circuito (FCT - *Functional Circuit Test*).

Nesse sentido, desenvolver uma jiga de testes automatizada capaz de validar o funcionamento das placas e gravar o *firmware* pode reduzir significativamente os custos de produção e acelerar o *time-to-market* dos armários inteligentes. Essa abordagem também permite executar cenários de testes exaustivos repetidamente, de forma rápida e eficiente, o que contribui para garantir a qualidade final do produto entregue ao cliente.

Além disso, a implementação desta jiga é estratégica no contexto produtivo da empresa Brametec. Visto que a validação atual das placas é feita de forma manual, o processo torna-se suscetível à falha humana e dificulta a rastreabilidade de defeitos recorrentes ou de lotes problemáticos de placas eletrônicas. A ausência de um sistema padronizado torna complexa a identificação de gargalos na montagem ou falhas de componentes específicos, demandando muito tempo da equipe de engenharia para diagnósticos que poderiam ser mais rápidos.

A utilização de conceitos de instrumentação virtual e o *design* de interfaces gráficas amigáveis em ambientes de teste são fundamentais para abstrair a complexidade do *hardware*. Essa abordagem facilita a operação por técnicos da produção e assegura a integridade dos dados coletados. Nesse contexto, a implementação de

uma camada de *software* de alto nível, composta de recursos visuais claros, torna-se essencial para que o operador compreenda o status de cada teste em tempo instantâneo, tornando o processo de validação intuitivo e garantindo rastreabilidade por meio da geração automática de documentação através de arquivos de registros de operações.

Sendo assim, a criação de um ambiente de testes automatizado torna-se relevante para solucionar os gargalos de produção identificados. A integração entre *hardware* e *software* proposta preenche a lacuna existente na rastreabilidade de problemas funcionais e na velocidade de gravação, consolidando a qualidade e a padronização das entregas da empresa.

1.2 Definição do Problema

De acordo com Rodrigues (2009), o desenvolvimento de soluções de engenharia no ambiente fabril concentra-se, sobretudo, na automação de processos. Esta estratégia é apontada pelo autor como o meio mais eficaz para elevar os índices de produção enquanto, simultaneamente, reduz os custos operacionais de fabricação. No entanto, contrastando com essa necessidade de otimização, o atual processo de validação das placas eletrônicas utilizadas nos armários inteligentes da Brametec ainda é realizado de forma manual, dependendo exclusivamente da intervenção humana.

Essa verificação manual apresenta-se intrinsecamente lenta e sujeita à subjetividade e à fadiga do operador, o que aumenta a probabilidade de falhas não detectadas passarem para as etapas finais de montagem. Além disso, a ausência de um sistema automatizado impede a geração padronizada de registros de testes, dificultando a rastreabilidade de defeitos e a análise estatística da qualidade dos lotes produzidos.

O problema central, portanto, reside na incapacidade do atual método de testes manuais em acompanhar a demanda crescente de produção com a velocidade, confiabilidade e rastreabilidade exigidas, gerando custos de retrabalho e riscos à qualidade do produto final.

Diante da necessidade de diminuir falhas e otimizar o tempo de produção, este trabalho visa responder à seguinte questão: Como projetar e implementar uma jiga de testes automatizada, integrando *hardware* e *software*, para realizar a validação funcional e garantir a rastreabilidade das placas eletrônicas aplicadas na produção de armários inteligentes da Brametec?

1.3 Objetivo Geral

Projetar e desenvolver uma jiga de testes automatizada composta por *hardware*, *firmware* e *software*, capaz de realizar a gravação, validação funcional e registro de dados das placas eletrônicas que compõem os armários inteligentes da empresa Brametec.

1.4 Objetivos Específicos

Para alcançar o objetivo geral, foram definidos os seguintes objetivos específicos:

a) Levantar os requisitos para realizar o teste da placa sob teste, identificando os pontos de testes, tensões de operação e forma de comunicação a serem validados.

b) Otimizar o custo de desenvolvimento do projeto, alinhando estratégias de engenharia à utilização de recursos e componentes já disponíveis na empresa para garantir uma implementação de baixo custo.

c) Projetar a estrutura mecânica da jiga para acomodação da placa sob teste, garantindo o alinhamento preciso das agulhas de teste (*pogo pins*) com os pontos de contato da PCI (Placa de Circuito Impresso).

d) Desenvolver o *hardware* de interface, responsável pela aquisição de dados, incluindo os circuitos de condicionamento de sinal e a interconexão entre a placa microcontrolada de instrumentação e a unidade de processamento central.

e) Desenvolver o *software* de gerenciamento e controle da jiga, implementando uma Interface Homem-Máquina (IHM) que assegure a usabilidade e a operação intuitiva do sistema.

f) Desenvolver rotinas de automação para a gravação do *firmware* na placa de teste e para a execução sequencial dos testes funcionais (comunicação, pinos de entrada e saída, sensores).

1.5 Estrutura do Trabalho

Este trabalho está organizado em seis capítulos. O Capítulo 1 apresenta a introdução, abordando o contexto da empresa Brametec e da produção de armários inteligentes, a justificativa para a automação dos testes, a definição do problema, os objetivos e a estrutura geral do documento.

O Capítulo 2 apresenta a fundamentação teórica necessária para o embasamento do projeto, iniciando pela definição de jigas de teste e sua relevância na

garantia da qualidade industrial. Discutem-se as metodologias de teste estrutural e teste funcional, seguidas pelo detalhamento de componentes físicos como as agulhas de teste e suas geometrias de contato. O capítulo também explora as arquiteturas de *hardware* e os conceitos de automação por *software* que sustentam a solução desenvolvida. Por fim, são introduzidos os princípios de Interface Homem-Máquina (IHM), focando em usabilidade e design de interação para garantir uma operação intuitiva, segura e eficiente na linha de produção.

O Capítulo 3 descreve a metodologia adotada, explicitando as etapas seguidas para o desenvolvimento do trabalho, desde a revisão bibliográfica e levantamento de requisitos até a definição da estratégia de validação do equipamento.

O Capítulo 4 apresenta o projeto e o desenvolvimento da jiga de testes, abrangendo o projeto do *hardware* - o estudo de circuitos eletrônicos, esquemático e *layout* da placa de interface -, a construção da estrutura mecânica de acoplamento, a implementação do *firmware* de baixo nível e a programação do *software* de controle e interface com o usuário.

No Capítulo 5 são apresentados os resultados experimentais obtidos e a respectiva discussão. A análise foca na validação funcional do equipamento utilizando placas padrão e unidades com falhas induzidas, verificando a eficácia na detecção de defeitos, a integridade dos registros de teste e o ganho de eficiência em comparação ao processo manual.

Por fim, o Capítulo 6 reúne as considerações finais, destacando o atendimento aos objetivos propostos, as limitações encontradas e sugerindo possibilidades de melhorias para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

O presente capítulo apresenta os conceitos fundamentais que sustentam o desenvolvimento deste trabalho. Inicialmente, define-se o conceito de jiga de testes e sua importância estratégica na garantia da qualidade e eficiência no cenário produtivo. Em seguida, são abordadas as metodologias de teste estrutural (ICT - *In-Circuit Test*) e funcional (FCT - *Functional Circuit Test*), detalhando as estratégias de sinergia industrial para mitigação da fuga de defeitos no processo de fabricação de placas.

Posteriormente, o capítulo explora os conceitos teóricos de automação de testes por *software* e analisa os elementos de interface física, com foco nas agulhas de teste (*pogo pins*) e no estudo comparativo de suas geometrias para contato elétrico. São introduzidas as arquiteturas de processamento e controle, comparando o uso de microcontroladores (MCUs) e computadores de placa única (SBCs) na orquestração de sistemas de teste automatizados.

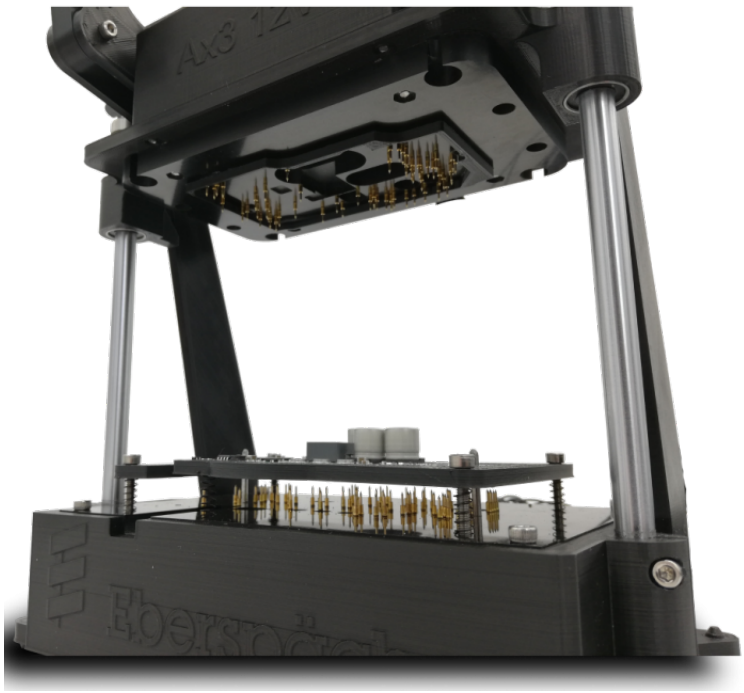
Por fim, o capítulo aborda os princípios de Interface Homem-Máquina (IHM), fundamentais para a abstração da complexidade do *hardware*. Discutem-se os pilares de usabilidade, visibilidade do sistema e *feedback* em tempo instantâneo, elementos essenciais para garantir uma operação intuitiva, segura e eficiente na linha de produção.

2.1 Jiga de testes

Uma jiga de testes é essencialmente a interface física e lógica projetada para conectar um dispositivo sob teste (DUT - *Device Under Test*) ao sistema de medição e controle. De acordo com Farias (2023), uma jiga de testes automatizada visa facilitar a conexão e a execução de sequências de verificação em componentes ou placas eletrônicas, reduzindo a necessidade de intervenção manual e minimizando erros humanos.

Estruturalmente, a jiga de testes atua como uma solução eletromecânica de precisão. O subsistema mecânico garante o alinhamento micrométrico entre o dispositivo sob teste e a matriz de sondas, enquanto o subsistema elétrico utiliza agulhas retráteis (*pogo pins*) para formar o que a indústria denomina “Cama de Pregos” (*Bed of Nails*). A Figura 1 apresenta um mecanismo para automação de testes desenvolvido pela empresa Elo Print (2026). Esta configuração permite que sinais de estímulo sejam injetados e respostas sejam coletadas simultaneamente em múltiplos nós da PCI, funcionando como uma ponte de comunicação de alta fidelidade entre o controlador e o dispositivo em análise.

Figura 1 – Jiga de teste Eloprint



Fonte: Elo Print (2026).

Conforme aponta Floriani (2023), a automação de testes é essencial para mitigar a subjetividade inerente aos processos manuais e assegurar a aplicação uniforme de critérios de qualidade. Nesse contexto, a necessidade de implementação de jigas de testes na indústria tem como objetivo resolver gargalos de qualidade e eficiência, visando especificamente:

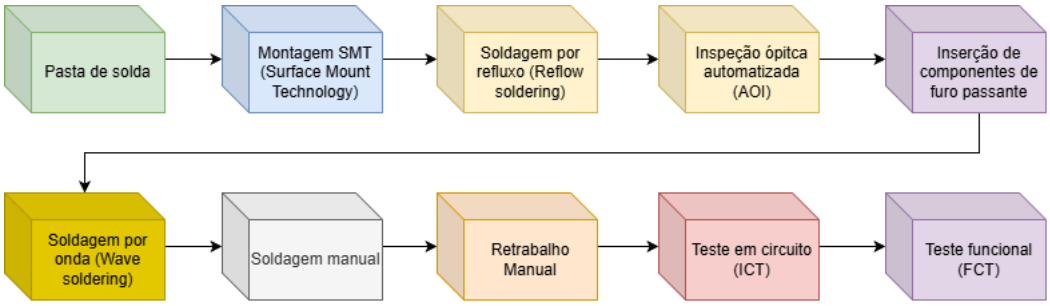
- Garantir repetibilidade para ambiente de teste ser idêntico para cada unidade, eliminando erros subjetivos de testes manuais.
- Acelerar o processo produtivo, reduzindo o tempo de produção através de execução de testes rápidos.
- Permitir a identificação de falhas latentes que não seriam detectadas por simples inspeção visual, assegurando que o produto final atenda os critérios de projeto.

2.1.1 Metodologias de Teste: ICT e FCT

A estratégia de verificação pode seguir diferentes metodologias de acordo com o objetivo industrial. O controle de qualidade na manufatura eletrônica moderna fundamenta-se na detecção prévia de falhas para minimizar o custo de retrabalho seguindo em dois principais testes: o teste estrutural (ICT - *In-Circuit Test*) e o teste funcional (FCT - *Functional Circuit Test)*.

O processo de fabricação de PCI montadas é uma operação complexa, composta por múltiplas etapas semiautomatizadas que operam sob alto volume e velocidade. Devido a essa natureza, falhas de manufatura são inerentes ao processo, exigindo estratégias de inspeção rigorosas. A Figura 2 ilustra o fluxo típico de produção, destacando que os testes de controle de qualidade, como o ICT e o FCT, concentram-se no estágio final da linha produtiva.

Figura 2 – Processo de fabricação de PCI montada



Fonte: Adaptado de Keysight Technologies (2023).

O ICT concentra-se na integridade física e estrutural da placa. Segundo a Keysight Technologies (2023), o objetivo é isolar e verificar cada componente individualmente através da técnica conhecida como análise nodal, garantindo que a malha elétrica foi montada conforme o esquema original. Por meio de uma matriz densa de agulhas, medem-se valores de impedância e continuidade sem necessariamente energizar a placa ou executar seu *firmware*.

Esta metodologia é altamente eficaz na detecção de defeitos de manufatura, como: curtos-circuitos entre trilhas e pinos, circuitos abertos (falta de solda ou solda fria) e componentes ausentes, valores incorretos ou polaridade invertida.

O FCT avalia a PCI montada como um sistema integrado, emulando seu cenário de uso definitivo. Conforme destaca a Magellan Circuits (2026), essa abordagem simula o ambiente operacional real para validar o comportamento sistêmico do dispositivo sob condições efetivas de trabalho. Para tanto, a National Instruments (2024) ressalta que o FCT exige que o dispositivo sob teste esteja energizado e com seu *firmware* ativo, utilizando instrumentação específica para emular as cargas e interface de comunicação — como o RS-485 e SPI — que o *hardware* encontrará em campo.

Enquanto o ICT se restringe à detecção de erros de montagem e defeitos estruturais, o FCT é vital para identificar falhas dinâmicas que só se manifestam durante a plena operação do *hardware*. De acordo com Magellan Circuits (2026), essa modalidade de teste permite:

- Identificação de interferências eletromagnéticas (EMI - *Electromagnetic Interfe-*

rence) ou flutuações de tensão que testes estáticos não captam.

- Validação da troca de pacotes e da conformidade temporal em barramentos seriais (RS-485, SPI, I2C) sob condições reais de tráfego.
- Validação da interação lógica entre os comandos do microcontrolador, o acionamento de atuadores e a resposta de leitura.

A aplicação conjunta das metodologias ICT e FCT visa minimizar a denominada fuga de defeitos (*Defect Escape*), um dos indicadores mais críticos na gestão da qualidade industrial. Este fenômeno é definido como a ocorrência de falhas que, ao não serem detectadas durante as etapas de verificação interna, são identificadas apenas pelo cliente final ou em fases subsequentes da cadeia de valor. A necessidade de conter esse escape justifica-se pelo incremento progressivo e severo dos custos de reparo; quanto mais tardia é a identificação de uma não-conformidade no fluxo produtivo, maiores são os encargos operacionais e financeiros acumulados para a organização.

Nesse cenário, as metodologias operam como filtros complementares em um sistema de cascata. O ICT proporciona um diagnóstico rápido e preciso para o reparo, identificando pontualmente componentes defeituosos ou falhas de montagem física. Complementarmente, o FCT atua como a barreira final de segurança, garantindo que unidades aprovadas estruturalmente não apresentem instabilidades de desempenho ou incompatibilidades de *software* em condições dinâmicas. Segundo a National Instruments (2024), um sistema de teste funcional robusto constitui a última oportunidade técnica de capturar falhas sistêmicas antes que estas resultem em prejuízos externos ou danos à reputação da marca.

A eficácia dessa estratégia de filtragem mista é corroborada por dados do setor. Estudos de caso conduzidos pela Keysight Technologies (2023) demonstram que a implementação de sistemas automatizados de teste resultou em uma redução de 20% nas tarefas manuais e um incremento de 10% na produtividade de placas para sistemas avançados de assistência ao condutor (ADAS). Adicionalmente, observou-se que a capacidade produtiva em fabricantes de autopeças registrou aumentos de até seis vezes com a transição para essas metodologias de verificação.

2.1.2 Automação de testes por *software*

A inteligência de uma jiga de testes reside na sua camada de *software*. De acordo com Bernardo e Kon (2008), testes automatizados são *scripts* ou programas desenvolvidos para exercitar as funcionalidades de um sistema e realizar verificações automáticas sobre os resultados obtidos. A principal vantagem dessa abordagem é a

capacidade de repetir casos de teste de forma rápida, precisa e com mínimo esforço humano.

Conforme aponta o *ISTQB (2014)*, o objetivo central da automação é confirmar o funcionamento do projeto conforme os requisitos, ajudando a reduzir falhas através da detecção antecipada de erros em execuções repetíveis.

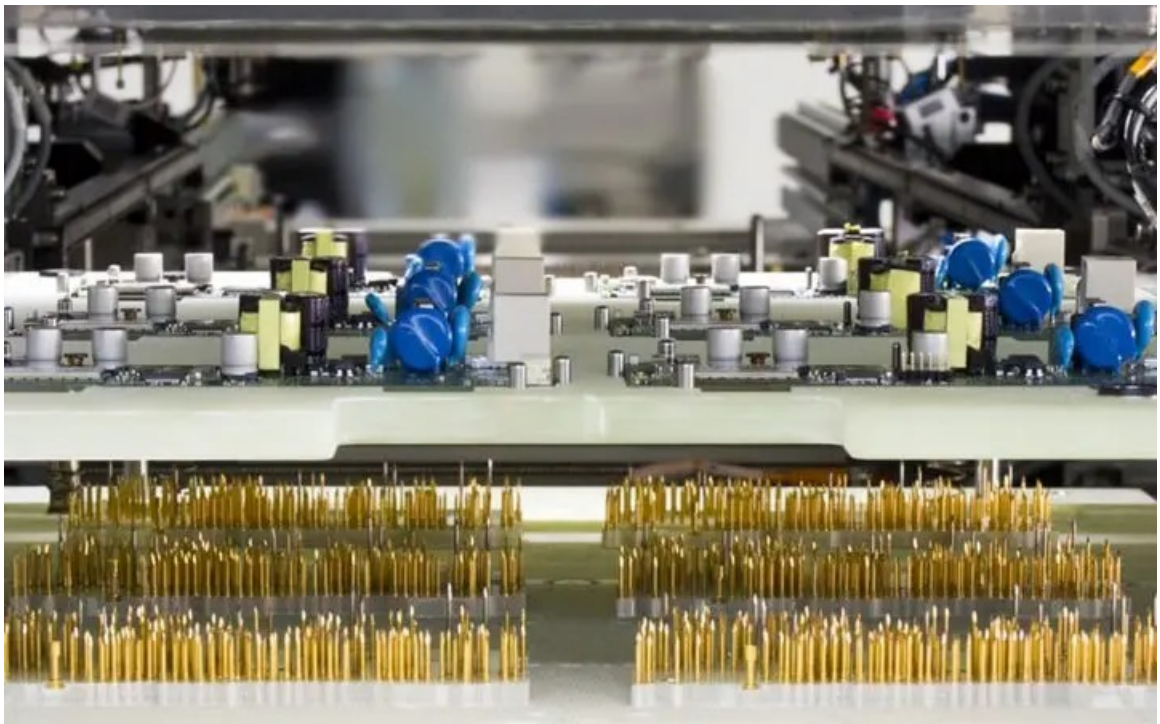
Os testes funcionais, conhecidos como testes de “caixa preta”, focam na finalidade do sistema baseando-se em seus requisitos funcionais, sem a necessidade de conhecimento sobre a implementação interna do código.

Segundo Freeman (2002), a dinâmica consiste em aplicar um conjunto de dados de entrada conhecidos e comparar os dados de saída resultantes com os resultados esperados. Qualquer discrepância entre a saída obtida e a esperada permite identificar falhas de processamento ou comportamentos inesperados no *hardware* que está em teste.

2.2 Agulhas de Teste

A interface física entre o equipamento de teste e o DUT é um dos elementos mais críticos no projeto de uma jiga. A topologia adotada neste trabalho baseia-se no conceito industrial conhecido como “Cama de Pregos” (*Bed of Nails*), conforme apresentado na Figura 3.

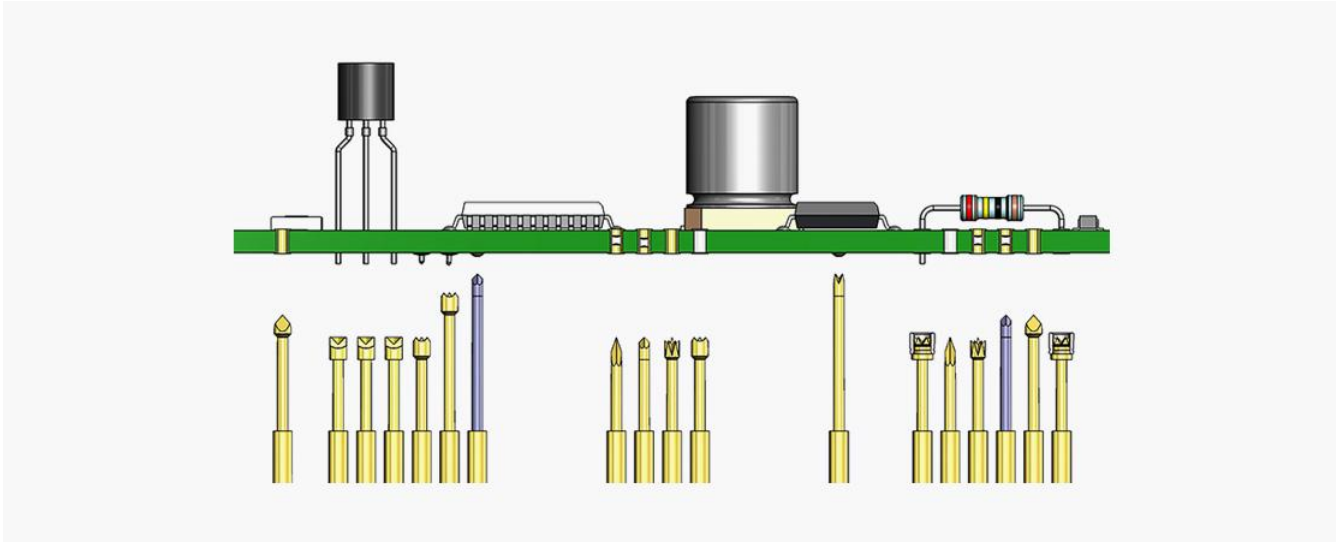
Figura 3 – Exemplo de cama de pregos



Fonte: NextPCB (2024)

Neste sistema, uma matriz de sondas retráteis carregadas por mola (*spring-loaded test probes*, popularmente conhecidas como *pogo pins*) é posicionada de forma a coincidir com os pontos de teste (TPs) da placa de circuito impresso. A confiabilidade deste contato depende de alguns fatores, como a força exercida pela mola, a capacidade de romper camadas de óxido ou resíduos de fluxo de solda, e a geometria da ponta da agulha em relação ao alvo (*pad* plano ou terminal de componente). A Figura 4 ilustra diferentes tipos de pontas de agulhas para suas diferentes aplicações.

Figura 4 – Agulhas de contato para testes PCI



Fonte: Brazil Connex (2025)

2.2.1 Análise das Pontas de Prova

Durante a fase de pesquisa e contato com fornecedores especializados, identificou-se que a escolha da ponta da agulha deve considerar o tipo de componente presente na DUT. No caso específico deste projeto, a placa sob teste utiliza componentes com tecnologia PTH (*Pin Through Hole*). Isso implica que, na face inferior da placa (lado da solda), existem terminais físicos que se projetam além da superfície da PCI.

Essa característica torna o uso de pontas planas ou pontiagudas simples (tipo lança) inadequado, pois há risco de deslizamento mecânico quando a agulha encontra a superfície irregular do terminal soldado. Com base nisso, foi realizado um estudo comparativo entre três geometrias de pontas disponíveis no mercado para determinar a mais adequada ao cenário de terminais PTH.

2.2.1.1 Agulha de Ponta Côncava

A ponta côncava, ilustrada na Figura 5, possui uma depressão em formato de taça em sua extremidade. De acordo com o guia de seleção da Feinmetall (Feinme-

tall GmbH, 2021), esta geometria é especificamente projetada para o teste de pinos longos ou terminais de *wire-wrap*. A principal característica é o auto-centramento. Ao ser pressionada contra o terminal PTH, a "taça" captura o pino do componente, impedindo que a agulha escorregue para o lado, garantindo um contato elétrico estável mesmo se houver pequenas vibrações ou desalinhamentos. No entanto, em ambientes com alta contaminação, o formato de "copo" pode acumular resíduos de sujeira ou fluxo, exigindo manutenção mais frequente.

Figura 5 – Agulha com ponta côncava

Fonte: Feinmetall GmbH (2025).

2.2.1.2 Agulha de Ponta Serrilhada

A ponta serrilhada, como mostra a Figura 6, apresenta uma cabeça com múltiplas arestas vivas em sua extremidade. Essa geometria é projetada para maximizar os pontos de contato simultâneos. Fabricantes como a QA Technology Company, Inc. (2017) asseguram este modelo como uma opção para terminais e pinos, oferecendo excelente estabilidade contra o deslizamento lateral. As múltiplas pontas são capazes de romper camadas de óxidos e contaminantes com eficiência. Além disso, sua área de contato ampliada oferece segurança em situações onde há pequenas variações na altura ou geometria do alvo, sendo versátil para diferentes tipos de terminais.

Figura 6 – Agulha com ponta serrilhada

Fonte: Feinmetall GmbH (2025).

2.2.1.3 Agulha de Ponta Coroa

A agulha de ponta tipo Coroa, apresentada na Figura 7, caracteriza-se por múltiplas arestas afiadas — tipicamente 4 ou 8 — com extremidades proeminentes. Tais arestas são extremamente eficazes para transpassar camadas de fluxo de solda e verniz, assegurando baixa resistência de contato. Para componentes PTH, essa geometria pode ser eficiente caso o pino do componente se encaixe no centro da coroa; entretanto, seu uso é predominantemente recomendado para o contato em *pads* planos ou vias preenchidas. Devido à multiplicidade de suas pontas, ela garante que, mesmo diante de pequenos desalinhamentos mecânicos, o contato elétrico com a superfície plana do *pad* ou da via seja estabelecido por ao menos alguns de seus pontos de contato

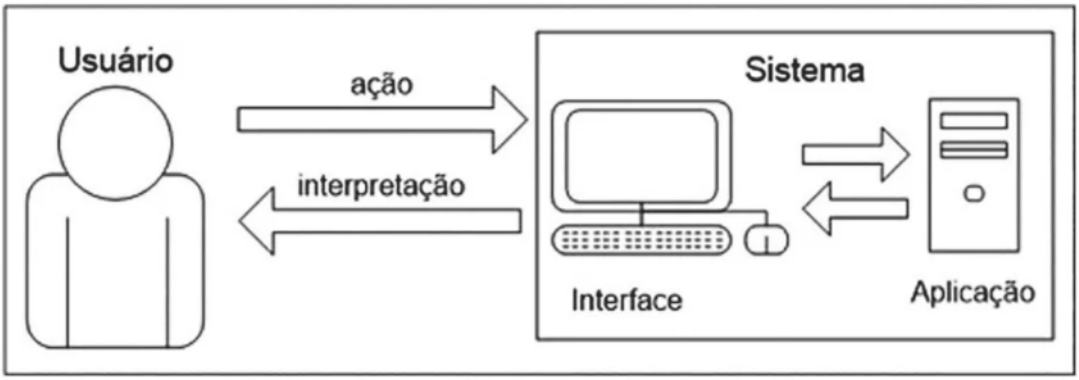
Figura 7 – Agulha com ponta coroa

Fonte: Feinmetall GmbH (2025).

2.3 Interface Homem-Máquina

A Interface Homem-Máquina (IHM) define-se como o conjunto de componentes de *hardware* e *software* que viabiliza a interação entre o usuário e um sistema computacional. Segundo Barreto *et al.* (2018), a interface atua como um tradutor que converte as intenções do operador em comandos executáveis pela máquina e, inversamente, apresenta dados processados pelo *hardware* de forma compreensível ao ser humano, a Figura 8 exemplifica essa interação entre homem-máquina.

Figura 8 – Interação entre usuário e sistema



Fonte: Prates e Barbosa (2007).

Sob uma perspectiva industrial, o fabricante Victor Vision (2024) ressalta que a IHM funciona como uma “ponte” de comunicação, permitindo que o operador monitore e controle processos em tempo instantâneo através de recursos visuais e táteis, o que é fundamental para a agilidade na tomada de decisão.

No contexto de sistemas de teste automatizados (ATE - *Automated Test Equipment*), a IHM desempenha um papel estratégico ao abstrair a complexidade dos sinais elétricos e protocolos de baixo nível. Para que essa interação seja eficaz, de acordo com Barreto *et al.* (2018), o design da interface deve buscar a “transparência tecnológica”, permitindo que o usuário se concentre na tarefa e não no funcionamento interno do *software*.

Dessa forma, a implementação de uma IHM caracteriza-e por:

- Fornecer indicadores visuais instantaneamente sobre o progresso e o resultado de cada etapa do teste, eliminando ambiguidades na interpretação dos dados.
- Adotar um design que guie o operador por um roteiro pré-definido, bloqueando ações inconsistentes que possam comprometer a integridade da DUT.
- Manter a interface responsiva, especialmente em comunicações seriais de longa duração, utilizando técnicas de multiprocessamento ou *multithreading* para evitar o congelamento da tela.

Complementarmente, os princípios de usabilidade de Nielsen (1994) reforçam que uma interface eficiente deve garantir a visibilidade do estado do sistema de forma constante. A escolha do *framework* para o desenvolvimento dessa interface gráfica (GUI - *Graphical User Interface*) impacta diretamente na modularidade e na portabilidade da solução. Conforme destacado por Barreto *et al.* (2018), a utilização de ferramentas que suportam interfaces ricas e intuitivas contribui diretamente para a satisfação e a produtividade do usuário final.

Para exibir essa interface e apresentá-la ao usuário, nas aplicações industriais modernas, tem-se utilizado telas *touchscreen*, como pode ser visto na Figura 9. Essa tecnologia elimina a dependência de periféricos externos, como teclado e mouse, o que otimiza o espaço físico na bancada de testes e torna a operação mais direta e assertiva para o ambiente de fábrica.

Figura 9 – Operador IHM industrial

Fonte: Victor Vision (2024)

2.4 Unidades de processamento em sistemas de testes automatizados

O núcleo de controle de uma jiga de testes funcional é responsável por orquestrar a lógica de verificação, garantindo que os estímulos sejam aplicados e as respostas coletadas com precisão temporal. Segundo Farias (2023), a implementação de um sistema automatizado exige uma unidade de processamento capaz de integrar *hardware* e *software* de forma coesa, permitindo a repetibilidade dos testes. A escolha desta unidade é pautada pelo compromisso entre o determinismo necessário para o controle de sinais, flexibilidade exigida pela interface com o usuário e a complexidade do dispositivo sob teste.

Para o desenvolvimento de jigas, a unidade de processamento pode atender determinados requisitos, por exemplo, para garantir a capacidade que um sinal de teste dure exatamente o tempo estipulado, realizar a abstração de *hardware*, facili-

tando a manipulação de portas lógicas e interfaces de comunicação e fazer o gerenciamento de dados.

2.4.1 Microcontroladores

Microcontroladores (MCU - *Microcontroller Unit*) são circuitos integrados que reúnem, em um único chip, uma unidade central de processamento (CPU - *Central Processing Unit*), memórias (*Flash*, RAM) e, em alguns casos, EEPROM (*Electrically Erasable Programmable Read-Only Memory*) e periféricos de entrada e/ou saída. Essa arquitetura integrada os torna especialmente adequados para tarefas de controle de baixo nível, aplicações embarcadas dedicadas e sistemas que exigem resposta determinística e em tempo real, características que podem ser fundamental em ambientes industriais e em jigas de testes automatizadas.

Em aplicações industriais, os MCUs são empregados no controle de atuadores, aquisição de sinais analógicos e digitais, comunicação com sensores e execução de rotinas de teste repetitivas, garantindo confiabilidade, previsibilidade temporal e baixo consumo de energia.

No cenário atual, duas arquiteturas se destacam para esta aplicação:

- Arquitetura AVR: possui uma gama alta de documentação, exemplos e casos de uso. Segundo a Microchip Technology Inc. (2021), sua arquitetura de 8-*bits* é eficiente para manipulação direta de I/O, embora possua menor poder de processamento, sendo utilizada para jigas que demandam alto número de acionamentos digitais simples.
- Arquitetura ARM Cortex-M: segundo a STMicroelectronics (2024), sua arquitetura de 32-*bits* representa o padrão moderno de sistemas. É utilizada quando a jiga precisa realizar processamento de sinal localmente ou gerenciar protocolos de comunicação de alta velocidade diretamente no nível de controle.

2.4.2 Computadores de placa única (SBC)

Quando a jiga de testes exige uma IHM mais elaborada, maior capacidade de processamento de dados ou integração com sistemas de rede complexos, utilizam-se computadores de placa única (SBC - *Single Board Computers*). Diferentemente dos microcontroladores, os SBC são capazes de executar sistemas operacionais completos, oferecendo maior flexibilidade de *software*, trazendo a capacidade computacional de um computador para o ambiente de teste compacto.

Esses dispositivos são especialmente adequados para aplicações que envolvem utilização de periféricos, como telas, teclados ou *mouses*, registro e análise de

dados, comunicação com servidores, execução de algoritmos avançados e interfaces gráficas ricas.

Um sistema baseado em SBC, rodando um sistema operacional completo, geralmente em Linux, permite abstrair a complexidade do teste para o operador através de uma IHM gráfica e intuitiva. Além disso, o SBC como unidade de processamento dos dados, é o responsável por armazenar registros de testes detalhados, gerenciar a comunicação em rede para integração com outros sistemas e executar *scripts* de teste dinâmicos escritos em linguagens de alto nível, como o *python*, permitindo alterações rápidas na lógica de teste.

O exemplo mais conhecido nesta categoria é a Raspberry Pi, que se consolidou na indústria devido ao seu vasto ecossistema de *software* e gerenciamento de pinos de entradas e saídas acessível, como definido em Raspberry Pi Ltd (2024), permitindo que ela atue tanto como computador de gestão quanto como interface direta de controle em aplicações menos críticas temporalmente.

Outra alternativa relevante no cenário industrial é a família BeagleBone, notável pela presença de unidades de tempo real programáveis (PRUs - *Programmable Real-Time Units*). Essas unidades permitem que o SBC execute tarefas determinísticas de alta velocidade paralelamente ao sistema operacional. No entanto, para o escopo deste projeto, essa opção não se torna necessária e o ecossistema de suporte gráfico é menos abrangente se comparado ao da Raspberry Pi, adicionando uma complexidade de desenvolvimento sem trazer benefícios práticos à camada de gerenciamento e IHM.

2.5 Resumo das tecnologias adotadas

Com base na fundamentação teórica apresentada, este projeto adota uma abordagem híbrida para atender aos requisitos de projeto definidos. Nesta configuração, o SBC gerencia a interface com o usuário, o banco de dados e o fluxo lógico de alto nível, enquanto envia comandos simplificados para o MCU, que executa as ações físicas de acionamento e medição. Essa abordagem desacopla a interface gráfica do controle crítico: se a interface gráfica travar ou demorar para atualizar, o microcontrolador continua monitorando as tensões e correntes, garantindo a segurança do dispositivo sob teste.

O Quadro 1 resume as tecnologias selecionadas e suas respectivas justificativas técnicas para o desenvolvimento da jiga.

Quadro 1 – Tecnologias selecionadas para o projeto

Tecnologia selecionada	**Justificativa**
Raspberry Pi 4 (SBC)	Necessidade de SO Linux para rodar interface gráfica, integração nativa com periféricos e desenvolvimento do *software*.
ATmega328PB (MCU)	Necessidade de controle de múltiplos pontos de I/O, interface serial e monitoramento de tensões e correntes.
Agulhas serrilhadas	Geometria escolhida para a interface de contato mecânico por maior aderência dos terminais e garantir estabilidade mecânica em terminais irregulares.
Python PySide6	Linguagem de alto nível para desenvolvimento da IHM e permite *multithreading* para não travar a interface durante os testes.

Fonte: Autor.

3 METODOLOGIA

Este capítulo apresenta a metodologia e as etapas de desenvolvimento de uma jiga de testes destinada à placa eletrônica de um armário inteligente da empresa Brametec. O projeto, caracterizado como uma pesquisa de natureza aplicada e experimental, tem como objetivo a construção de um equipamento que integre *hardware* e *software* para automatizar o controle de qualidade em ambiente de produção, abrangendo desde a gravação de microcontroladores até a validação funcional e o teste de componentes.

A abordagem metodológica estrutura-se de forma sequencial, iniciando-se pelo estudo das especificações técnicas do dispositivo sob teste e pela definição da mecânica de contato. Na sequência, detalha-se o projeto eletrônico, composto pela placa de controle e pela placa de interface, que acomoda as agulhas de teste. O capítulo descreve ainda o desenvolvimento do *firmware* e do *software* de gerenciamento, concluindo com a modelagem mecânica da estrutura e a validação experimental do protótipo em bancada.

3.1 Métodos Aplicados

A etapa inicial consistiu em um estudo sobre o funcionamento e a implementação de uma jiga de teste, com foco em testes funcionais de circuito. A pesquisa abordou as estruturas mecânicas, os componentes eletrônicos necessários e as estratégias de *software* utilizadas na indústria. Esse levantamento serviu como base técnica para definir a arquitetura e os componentes que seriam utilizados no projeto.

Na sequência, para a execução do projeto, os procedimentos foram divididos em etapas lógicas de desenvolvimento. Inicialmente, foram analisados o esquemático e o *layout* da placa sob teste. Esta etapa teve como objetivo mapear os pontos de teste (*test points*) críticos e compreender os requisitos de operação, identificando a tensão de alimentação e o modo de comunicação.

O desenvolvimento lógico foi conduzido em duas frentes complementares: uma dedicada ao *firmware* do microcontrolador da jiga, responsável pelo controle de baixo nível e aquisição de sinais, e outra voltada para o *software* de interface. Esta segunda parte foi estruturada para organizar a sequência automática de testes, gerenciar o processo de gravação do arquivo binário na placa sob teste e processar os dados coletados para fornecer uma resposta imediata ao operador ("Sucesso/Falha").

Na etapa final, referente à validação dos resultados, foram realizados testes práticos submetendo à jiga tanto placas em pleno funcionamento quanto unidades com falhas pré-identificadas. A análise priorizou a validação funcional dos estágios críticos, verificando a conformidade dos níveis de tensão, a integridade da troca de

dados via comunicação serial e as principais falhas de eletrônica. Além disso, avaliou-se a eficácia da jiga em simular o ambiente real de operação, assegurando a correta simulação de acionamento das solenoides e a leitura dos seus respectivos sensores de estado. Dessa forma, confirmou-se a capacidade do equipamento em segregar as unidades aprovadas das reprovadas conforme os critérios estabelecidos.

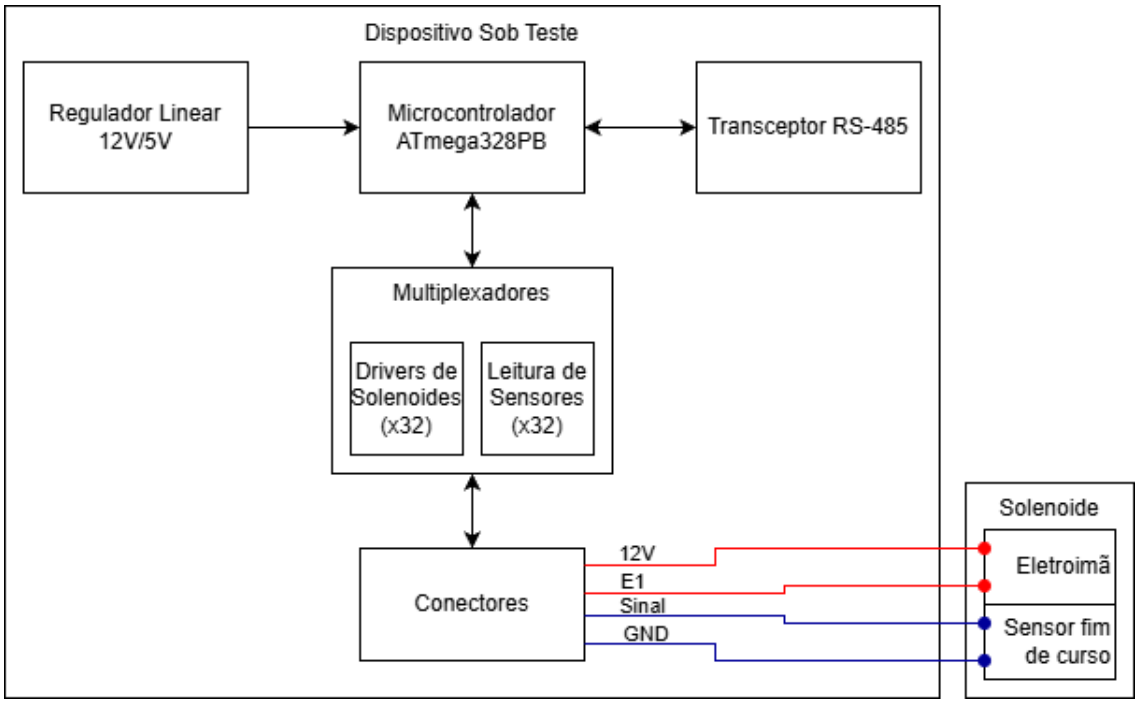
3.2 Caracterização do Dispositivo Sob Teste (DUT)

Para a concepção da jiga de testes, a etapa fundamental consistiu na análise técnica detalhada do dispositivo eletrônico a ser validado, adiante denominado dispositivo sob teste ou também referido como placa sob teste.

O produto em questão consiste em uma placa controladora para armários inteligentes (*smart lockers*). Sua função principal é gerenciar portas de armários através do acionamento de solenoides, bem como monitorar o estado dessas portas (aberta/fechada) através de sensores. A placa opera como um dispositivo escravo em uma interface de comunicação RS-485, recebendo comandos de uma central para liberar o acesso aos compartimentos.

Devido a propriedade intelectual com a empresa Brametec, o esquema elétrico detalhado e fotografias do *layout* da PCI completa não serão reproduzidos neste documento. A análise a seguir baseia-se na arquitetura funcional do sistema, conforme representado no diagrama de blocos da Figura 10.

Figura 10 – Diagrama de blocos funcional da DUT



Fonte: Autor.

A análise do esquema elétrico e a análise funcional da DUT revelaram cinco subsistemas críticos que definem os requisitos de *hardware* e *software* da jiga:

- A placa é alimentada com tensão nominal de 12 V. Internamente, um regulador linear converte este potencial elétrico para 5 V, nível responsável pela alimentação do microcontrolador e dos circuitos responsáveis pela lógica digital. Dada a criticidade para a estabilidade do sistema, a saída deste regulador (barramento de 5 V) foi definida como um TP obrigatório. A Jiga deverá validar se a tensão encontra-se dentro da tolerância aceitável antes de prosseguir com os testes lógicos.
- A troca de dados da placa ocorre através de um barramento serial RS-485. Esta interface utiliza um par diferencial de sinais, cujos pontos de conexão na PCI devem ser acessados pela jiga para verificar não apenas a continuidade elétrica, mas a integridade funcional do transceptor e a qualidade da troca de pacotes.
- A gravação do *firmware* no microcontrolador é realizada via ICSP (*In-Circuit Serial Programming*), utilizando o protocolo SPI. Consequentemente, a Jiga deve garantir o contato físico com os pinos MISO, MOSI, SCK e RST. Isso permite que a jiga realize o processo de gravação do código de produção e verifique a assinatura do microcontrolador.
- A placa possui capacidade para gerenciar até 32 canais de fechaduras. Cada canal é composto por uma saída para solenoide (acionamento indutivo) e uma entrada para sensor de fim de curso (retorno de estado). Isso totaliza 64 pontos de I/O (*Input/Output*) que demandam validação individualizada.
- Para otimizar o uso de portas do microcontrolador, o sistema utiliza lógica de multiplexação tanto para o acionamento dos transistores do tipo MOSFETs quanto para a leitura dos sensores. Portanto, a Jiga não deve apenas testar a continuidade elétrica ("teste de agulhas"), mas sim validar a lógica de endereçamento, garantindo que o comando para uma determinada porta acione o transistor correto e leia o sensor correspondente, sem interferência em canais adjacentes.

Com base na caracterização destes subsistemas, definiu-se a topologia da interface eletromecânica de contato — conhecida como *bed of nails* ou cama de pregos. O mapeamento final resultou na alocação estratégica de agulhas de teste distribuídas da seguinte forma: 64 pontos dedicados às entradas e saídas (32 para solenoides e 32 para sensores), 6 pontos para comunicação e programação (4 para ICSP e 2 para RS-485) e 5 pontos para as malhas de energia. Dentre estes últimos, destacam-se a entrada de alimentação (12 V), o monitoramento da tensão lógica (5 V) e três

conexões de referência (GND), essenciais para assegurar a equipotencialidade e o retorno de corrente adequado entre a jiga e a DUT.

3.2.1 Interface de comunicação

Para além das verificações elétricas, a jiga interage com o DUT em nível lógico. A placa sob teste utiliza um protocolo de comunicação serial proprietário sobre a interface física RS-485. Nesta topologia, a jiga atua como mestre (*master*), enviando comandos de requisição, enquanto o DUT opera como escravo (*slave*), respondendo com a confirmação da ação ou o estado atual dos periféricos.

A validação do *firmware* e do *hardware* de comunicação baseia-se na transmissão de cadeias de caracteres (*strings*) pré-definidas e na subsequente análise da resposta recebida. Para cada etapa do teste funcional — como o acionamento de uma fechadura ou a leitura de sensores — a jiga envia o comando correspondente e compara o retorno da placa com o padrão esperado para aquele cenário.

O Quadro 2 apresenta a estrutura dos principais comandos disponíveis na DUT para a rotina de testes, detalhando os comandos de envio e as respostas esperadas para cenários de sucesso e exceção.

Quadro 2 – Fluxo de comandos e respostas do protocolo de comunicação

Função	**Descrição do Comando**	**Comportamento da Resposta**
Abrir uma porta	Envia solicitação para acionar a fechadura de uma solenoide específica.	Retorna confirmação de sucesso, aviso informativo (porta já aberta) ou mensagem de erro.
Status de uma porta	Solicita a leitura do sensor de uma solenoide específica.	Retorna o estado atual da porta (ex: aberta ou fechada).
Status de todas as portas	Solicita a leitura simultânea dos sensores de todas as portas.	Retorna uma sequência de dados representando o estado de cada porta.
Versão do *firmware*	Solicita informações sobre a gravação da placa.	Retorna a identificação da versão do *firmware* e data.

Fonte: Autor.

A lógica de verificação implementada na jiga valida não apenas a integridade do pacote recebido, mas também interpreta o conteúdo semântico da mensagem. Por exemplo, ao enviar o comando para abrir uma porta, o sistema aguarda uma resposta específica dependendo do estado inicial do dispositivo. O detalhamento do ciclo completo de testes e os critérios de aceitação serão descritos nas seções subsequentes.

3.3 Definição do Circuito de Testes

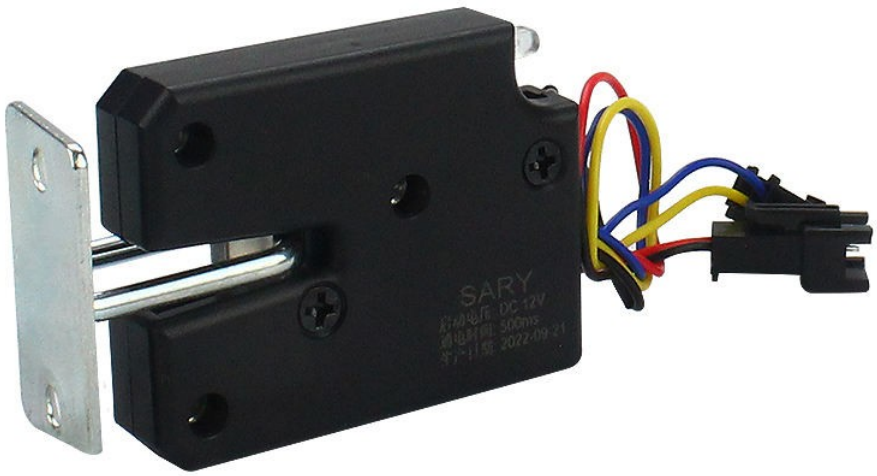
Um aspecto fundamental do projeto da jiga é a capacidade de realizar a validação do funcionamento de forma puramente elétrica, simulando os estados da solenoide sem a necessidade de conectar a carga física.

Para a validação dos acionamentos, optou-se pela utilização de uma simulação de carga ativa em detrimento de solenoides físicas acionáveis. Conforme destaca Rodrigues (2009), a utilização de cargas eletrônicas variáveis permite uma maior linearidade na simulação, além de simplificar a topologia do *hardware* ao eliminar a necessidade de comutação mecânica das 32 solenoides necessárias para testar o funcionamento da placa.

A estratégia consiste em monitorar se o circuito de acionamento fornece a tensão correta (simulando a ativação da bobina) e, em resposta, manipular eletricamente os sinais de retorno do sensor. Dessa forma, a jiga valida o sistema sob teste, testando a lógica de controle sem a complexidade e o desgaste de acionar 32 mecanismos físicos sucessivamente.

Inicialmente, é necessário caracterizar o funcionamento dos componentes sob teste: a solenoide de acionamento da porta e seu respectivo sensor. A solenoide, apresentada na Figura 11, opera como uma trava eletromecânica dotada de uma lingueta; em seu estado de repouso (“fechada”), ela mantém a porta travada, enquanto o acionamento para a posição “aberta” libera a lingueta, abrindo a porta. Complementarmente, o sensor integrado atua como uma chave fim de curso, operando nos estados normalmente aberto (NA), quando a trava está liberada, e normalmente fechado (NF), quando a trava está recolhida.

Figura 11 – Solenoide

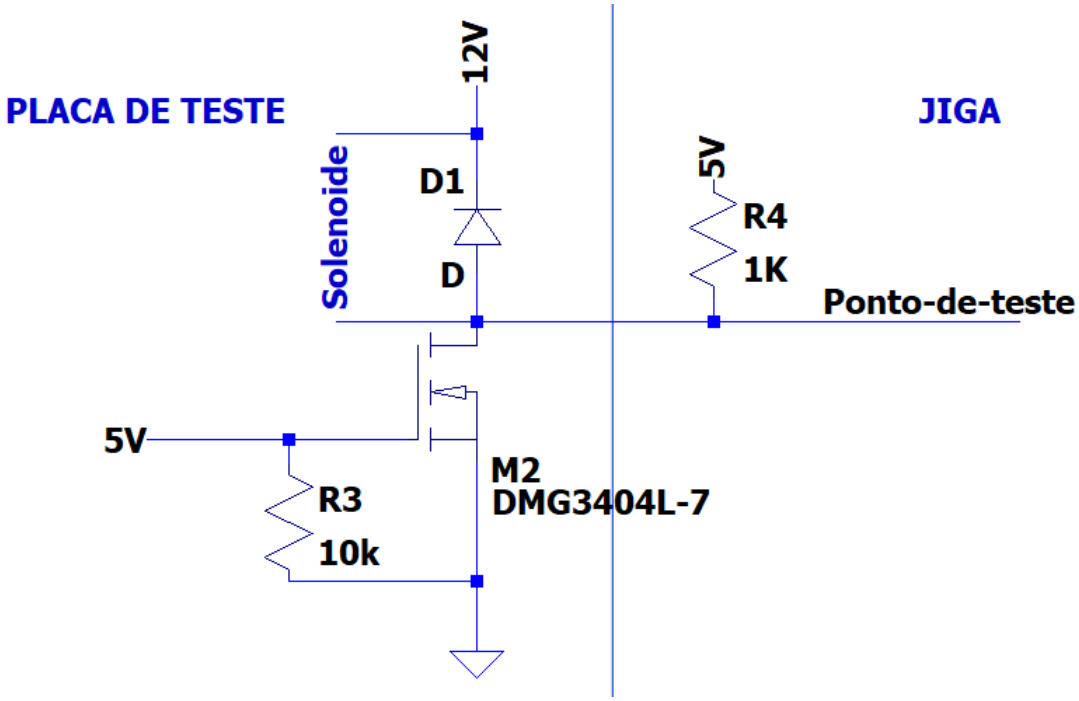


Fonte: Sary (2021)

Segundo as especificações do fabricante, a solenoide opera com tensão nominal de 12 V e corrente de 2 A a 20 ºC. O seu regime de trabalho é intermitente, com um ciclo de trabalho (*duty cycle*) de 10% (1 segundo ligada e 9 segundos desligada). O conjunto inclui um sensor do tipo chave fim de curso, que opera em circuito aberto quando a solenoide está aberta e em curto-circuito quando está fechada.

Para validar a topologia do circuito de verificação, realizaram-se ensaios preliminares nas placas disponíveis utilizando um *firmware* de depuração para operações de leitura e escrita. O objetivo foi mapear os níveis lógicos correspondentes aos estados mecânicos da trava, validando o funcionamento físico da solenoide. A Figura 12 ilustra o arranjo experimental, uma linha vertical separa o circuito do dispositivo sob teste da eletrônica de monitoramento da jiga. Na placa de teste, o acionamento é executado por um transistor MOSFET que chaveia o terminal de terra (GND) da solenoide. O circuito da jiga, por sua vez, monitora esse chaveamento, verificando se o ponto de teste foi levado ao potencial elétrico de referência ou se permanece em estado de *pull-up*.

Figura 12 – Circuito de validação da solenoide



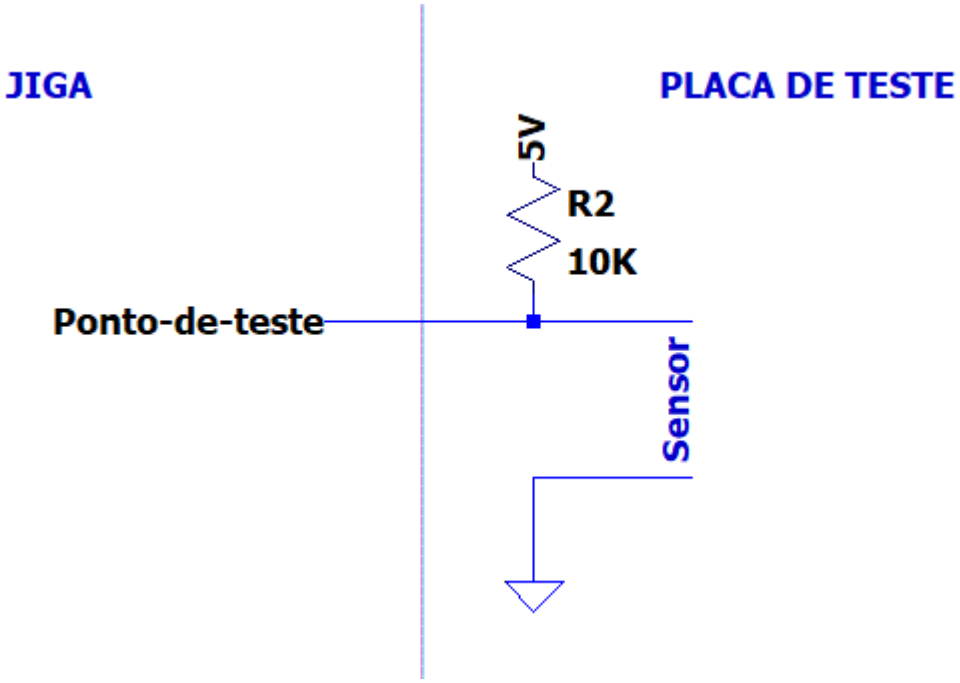
Fonte: Autor.

A lógica do Ponto-de-teste é validada da seguinte forma:

- Nível Lógico Alto: Corresponde à solenoide fechada (lingueta em posição de bloqueio).
- Nível Lógico Baixo: Corresponde à solenoide aberta (lingueta acionada para liberação).

Para a leitura do retorno do sensor da solenoide, a placa de teste utiliza um resistor de *pull-up* conectado a 5 V, conforme detalhado na Figura 13, uma linha vertical delimita a interface entre o circuito do dispositivo sob teste e o sistema de acionamento da jiga. O procedimento de validação consiste em alternar o estado lógico no ponto de teste, simulando o fechamento do contato ao levar o sinal de nível alto (5 V) para nível baixo (GND). A jiga monitora o retorno da placa para confirmar se o sistema reconhece corretamente o sensor como “aberto” (nível lógico alto) ou “fechado” (nível lógico baixo).

Figura 13 – Circuito de validação do sensor



Fonte: Autor.

Para escalar a estratégia de simulação para a totalidade dos pontos de teste, o circuito de verificação foi replicado e estruturado através do multiplexador CD74HC4067SM96. Este circuito integrado atua como uma interface bidirecional, permitindo tanto a leitura de níveis lógicos quanto a imposição de tensão (5 V ou GND) no canal selecionado, o qual é definido pela combinação binária dos pinos de endereço S0, S1, S2 e S3.

Esta configuração viabiliza a execução da seguinte lógica de teste na jiga:

- Teste da solenoide: o sistema opera em modo de leitura para validar o comando de acionamento. Ao monitorar o canal selecionado, se for identificado nível lógico baixo, confirma-se que o circuito da DUT atuou para abrir a porta; caso o sinal permaneça em nível alto, indica que a solenoide permaneceria fechada (não acionada).

- Teste do sensor: o sistema opera em modo de escrita. O multiplexador projeta no canal da respectiva solenoide o estado que se deseja simular. A jiga força o sinal para nível baixo (GND) para simular o fechamento da chave do sensor, validando se a placa principal é capaz de ler e interpretar corretamente esse retorno e para validar o sensor aberto, aplica um *reset* a saída do multiplexador.

A Tabela 1 apresenta a lógica de endereçamento necessária para a seleção dos canais do multiplexador. Cada CI possui 16 canais de entrada ou saída. O multiplexador é ativo apenas com o *enable* em nível lógico baixo.

Tabela 1 – Tabela da verdade do multiplexador

S0	S1	S2	S3	$\overline{E}$	Canal Selecionado
X	X	X	X	1	Nenhum
0	0	0	0	0	0
1	0	0	0	0	1
0	1	0	0	0	2
1	1	0	0	0	3
0	0	1	0	0	4
1	0	1	0	0	5
0	1	1	0	0	6
1	1	1	0	0	7
0	0	0	1	0	8
1	0	0	1	0	9
0	1	0	1	0	10
1	1	0	1	0	11
0	0	1	1	0	12
1	0	1	1	0	13
0	1	1	1	0	14
1	1	1	1	0	15

Fonte: Adaptado de Texas Instruments (2024).

Considerando a demanda de 32 pontos de teste para as solenoides e 32 pontos para os sensores, totalizando 64 canais de I/O, foram utilizados quatro multiplexadores no projeto, 2 conjuntos para leitura do acionamento da solenoide e 2 conjuntos para escrita do sensor. Estes componentes estão interconectados a um microcontrolador ATMEGA328PB, que é responsável por gerenciar o endereçamento dos multiplexadores e realizar o monitoramento dos pontos de teste da DUT. Os pinos de endereçamento do multiplexador (S0, S1, S2 e S3) são compartilhados nos quatro componentes utilizados.

3.4 Sistema de Controle e Gravação

A escolha da Raspberry Pi 4 Modelo B como unidade central de processamento deve-se à sua credibilidade, disponibilidade na empresa e facilidade de integra-

ção com periféricos. O sistema operacional baseado em linux permite acesso remoto via SSH e oferece suporte nativo a displays *touchscreen*, essenciais para a operação da IHM da jiga.

O *software* de gerenciamento foi desenvolvido em linguagem *Python*, utilizando o *framework* PySide6, criado pelo The Qt Company (2025). Esta ferramenta foi selecionada por permitir o desenvolvimento ágil de uma interface gráfica modular e intuitiva, garantindo que as ferramentas de diagnóstico sejam de fácil compreensão para o operador final.

Além do gerenciamento da interface, a Raspberry Pi atua como *hardware* de gravação via ICSP. O sistema utiliza o barramento SPI nativo da Raspberry para realizar a configuração dos *fuses* e *bits* de configuração e a gravação do *firmware* no microcontrolador da placa sob teste. Considerando que a Raspberry Pi opera com lógica de 3,3 V e o microcontrolador da DUT opera em 5 V, foi implementado um conversor de nível lógico bidirecional nas linhas de comunicação para garantir a compatibilidade e a integridade elétrica dos sinais.

A gravação da placa é realizada utilizando o programa AVRDUDE (AVR Downloader Uploader) (DEAN *et al.*, 2025). Trata-se de uma ferramenta de linha de comando voltada para ler, escrever e manipular memórias internas dos microcontroladores AVR da Microchip Technology. Ele suporta a programação de memória *flash* e EEPROM, além da configuração dos *bits* de fusível (*fuses*) e de bloqueio.

Para integrar o AVRDUDE à lógica da jiga de testes e garantir a repetibilidade do processo, foi desenvolvido um *shell script* executado no ambiente linux da Raspberry Pi. Este *script* atua como uma camada de abstração, organizando as chamadas ao programador e tratando os códigos de retorno para validar cada etapa.

As principais funcionalidades implementadas no algoritmo de gravação são:

1. O *script* configura o AVRDUDE para utilizar o *driver linuxspi*, permitindo que a Raspberry Pi programe o microcontrolador da placa sob teste diretamente através de seu barramento SPI nativo, sem a necessidade de programadores externos USB.

2. Antes de iniciar a gravação, realiza um “teste de *handshake*”. Ele tenta ler a assinatura do dispositivo (*Device Signature*). Caso a comunicação falhe na velocidade padrão, o algoritmo automaticamente reduz a velocidade do *clock* SPI e tenta novamente.

3. O *script* impõe a configuração dos *bits* de fusível antes do *firmware*. Isso é crítico para garantir que o microcontrolador opere com a fonte de *clock* correta e os níveis de *Brown-out Detection* (BOD) adequados.

4. O processo segue o fluxo de:
    a) Apagar o chip.
    b) Gravar a memória *flash*.
    c) Ler a memória gravada e comparar com o arquivo original.
5. Ao finalizar a gravação, o AVRDUDE libera o barramento SPI, mas nem sempre garante o estado lógico da linha de *reset*. Para contornar isso, o *script* utiliza a ferramenta *gpioset* (da biblioteca *libgpiod*) para forçar o pino de *reset* (GPIO 25) para nível lógico alto, assegurando que a placa sob teste reinicie e execute o *firmware* recém-gravado imediatamente.

Se qualquer etapa falhar, o *script* aborta a operação e retorna um código de erro para a interface de controle.

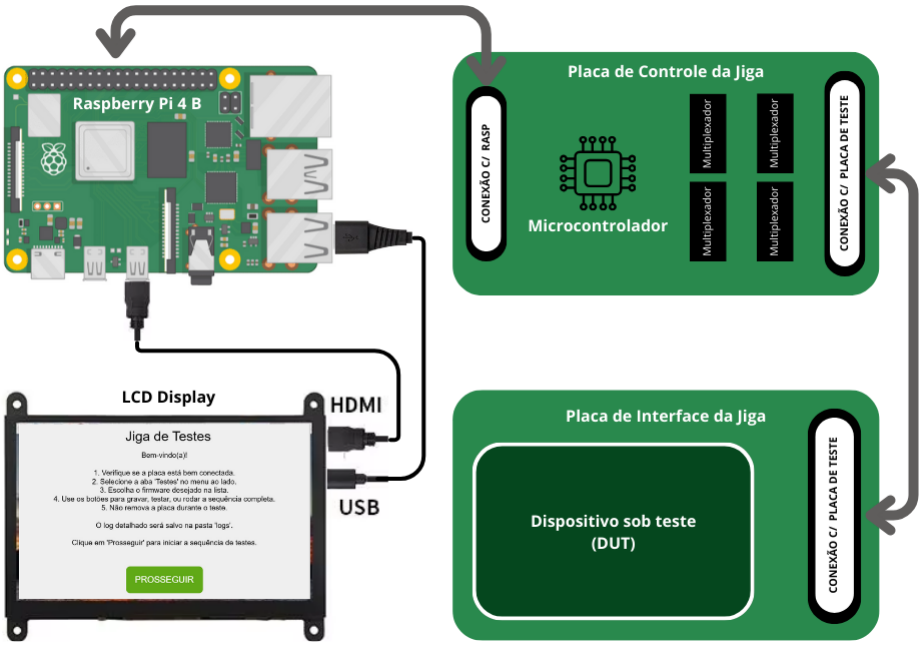
3.5 Projeto da placa de circuito impresso para a jiga de testes

O desenvolvimento das Placas de Circuito Impresso (PCI) aplicadas para a implementação da jiga de teste priorizou a integridade de sinal e a modularidade do sistema, requisitos essenciais para suportar a alta densidade de conexões dos múltiplos pontos de teste. Visando a viabilidade econômica e a facilidade de manutenção, optou-se por uma arquitetura segregada em dois módulos de *hardware* distintos: a placa de controle e a placa de interface (responsável pela fixação das agulhas de teste).

Essa estratégia de desacoplamento isola a lógica de controle e processamento da estrutura mecânica de contato. Dessa forma, caso o *design* do dispositivo sob testes sofra alterações futuras, apenas a Placa de Interface necessitará de revisão, preservando-se o investimento e a engenharia aplicados na placa de controle. Tal abordagem simplifica a manutenção mecânica e reduz significativamente os custos e o tempo de atualização da jiga.

A Figura 14 apresenta o diagrama de interconexões do sistema. A placa de controle atua como o núcleo de instrumentação, alojando o microcontrolador auxiliar responsável pela aquisição de dados (leitura dos sensores, simulação de carga das solenoides e monitoramento da tensão de 5 V).

Figura 14 – Diagrama de blocos das conexões eletrônicas



Fonte: Autor.

3.5.1 Projeto da Placa de Controle

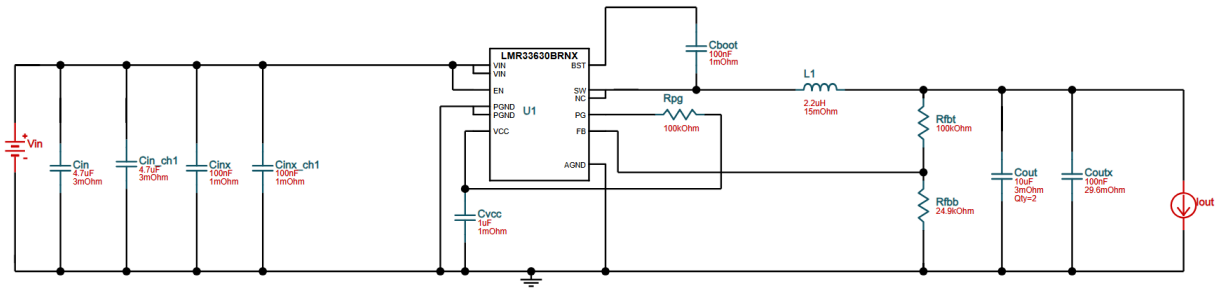
A placa de controle foi projetada para operar com uma tensão de entrada nominal de 12 V. Um subsistema crítico deste projeto é a fonte de alimentação regulada, responsável por converter a tensão de entrada para 5 V estáveis, com capacidade de corrente suficiente para alimentar os componentes da placa de controle e a Raspberry Pi.

Para garantir a eficiência energética, essa fonte foi implementada utilizando uma topologia de conversor CC-CC do tipo *step-down* (Buck). O componente selecionado para essa função foi o LMR33630 da Texas Instruments. O dimensionamento do circuito foi realizado por meio da ferramenta Texas Instruments Webench® Power Designer, adotando-se os seguintes parâmetros:

- Tensão de Entrada (V_{IN}): 12 V a 13,5 V;
- Tensão de Saída (V_{OUT}): 5 V;
- Corrente de Saída (I_{OUT}): 3 A;
- Temperatura Ambiente (T_A): 30 °C.

A ferramenta de simulação gerou a topologia de referência apresentada na Figura 15, indicando os valores ideais de indutância e capacitância para a estabilidade da malha de controle.

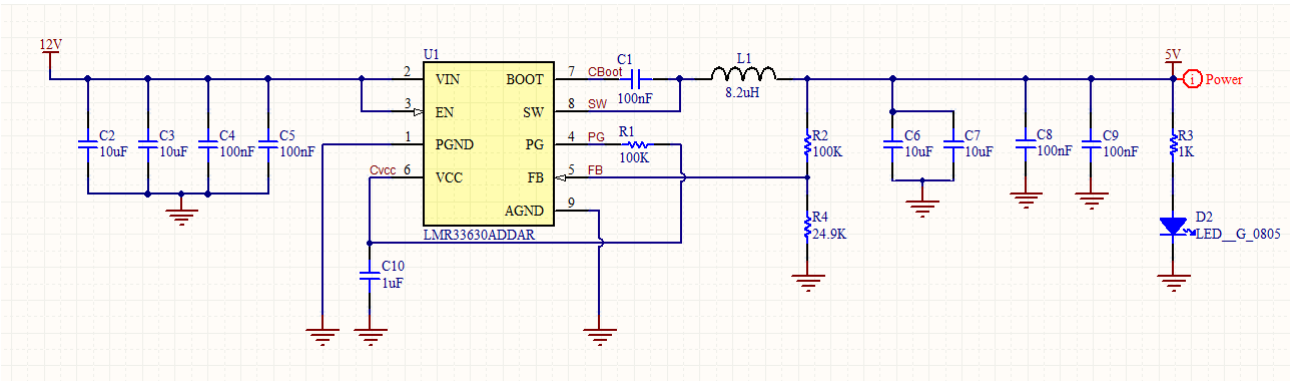
Figura 15 – Circuito de referência para o conversor buck LMR33630



Fonte: Texas Instruments (2025).

Com base na sugestão do fabricante, o circuito final foi adaptado para atender às disponibilidades de componentes comerciais e restrições de *layout* da placa. O esquemático implementado na jiga é apresentado na Figura 16.

Figura 16 – Esquemático final do conversor buck implementado



Fonte: Autor.

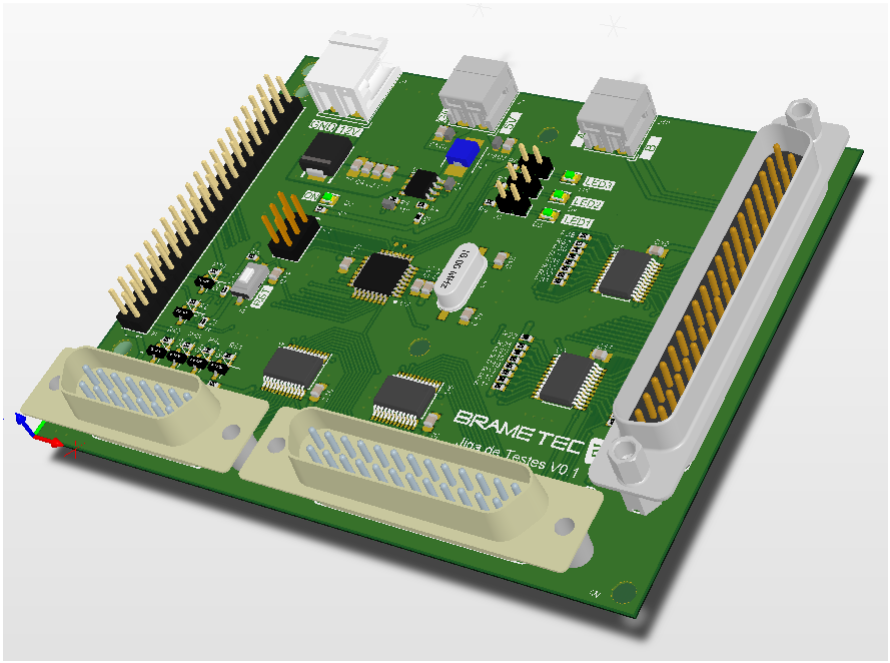
O projeto da PCI integra os conectores de alta densidade para acoplamento com a placa de interface, o barramento de conexão com a Raspberry Pi, os multiplexadores analógicos para expansão das portas, o microcontrolador ATmega328PB e os estágios de conversão de nível lógico bidirecional.

Devido a restrições de confidencialidade industrial, o esquemático detalhado contendo as interconexões entre os blocos multiplexadores, o microcontrolador e os demais subcircuitos não serão apresentados integralmente. Contudo, o Apêndice A fornece uma representação funcional do sistema por meio de um diagrama de blocos, detalhando os principais sinais e a arquitetura das conexões implementadas.

A visualização tridimensional da placa finalizada pode ser observada na Figura 17. As vistas detalhadas das camadas superior (*top*) e inferior (*bottom*) da Placa de Controle encontram-se disponíveis no Apêndice B.

Além disso, este módulo contém os circuitos conversores de nível lógico, essenciais para a interface de comunicação segura entre a Raspberry Pi e a DUT durante o processo de gravação e testes.

Figura 17 – Renderização 3D da placa de controle



Fonte: Autor.

3.5.2 Projeto da Placa de Interface

O desenvolvimento da placa de interface demandou um rigoroso alinhamento mecânico, visto que este componente é responsável pelo acoplamento físico com os *test points* da DUT. A modelagem foi executada em *software* CAD eletrônico, utilizando como referência as coordenadas originais dos arquivos de fabricação da placa sob teste.

O processo consistiu na importação das coordenadas X e Y dos furos e ilhas de solda da placa original, assegurando a concentricidade entre as agulhas de teste e os pontos de contato. Dada a topologia repetitiva do circuito na placa sob teste — composto por múltiplos canais idênticos de controle —, utilizou-se a ferramenta de replicação hierárquica de *layout*. Essa técnica garantiu que o padrão de espaçamento fosse replicado com exatidão para todos os canais, mitigando riscos de erro humano no posicionamento manual e assegurando a confiabilidade da interface mecânica.

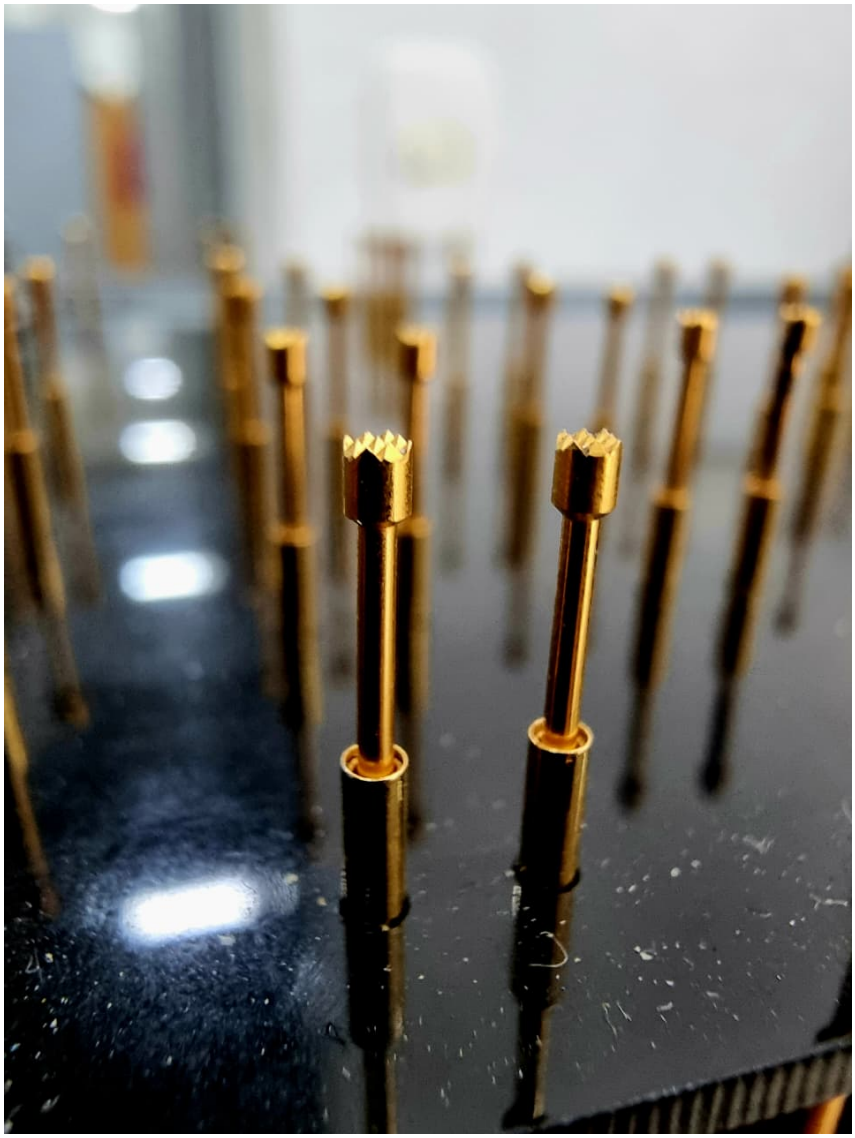
Para a definição da interface de contato, foram realizados testes práticos comparativos entre as geometrias apresentadas no capítulo 2 de fundamentação teórica.

Embora a literatura sugira o uso de pontas côncavas para terminais PTH, nos ensaios realizados neste projeto, a ponta do tipo serrilhada apresentou desempe-

nho superior, conforme pode ser visualizado na Figura 18. A escolha justifica-se pela sua configuração com múltiplas arestas vivas, que proporcionou uma penetração mais eficaz em superfícies oxidadas, garantindo baixa resistência de contato.

Adicionalmente, observou-se empiricamente que a geometria serrilhada favoreceu o autoalinhamento com os terminais dos componentes, compensando as tolerâncias mecânicas inerentes ao processo de fabricação da PCI e evitando o deslizamento da agulha, falha que foi observada nos testes com a ponta tipo coroa.

Figura 18 – Detalhamento da agulha de teste tipo serrilhada

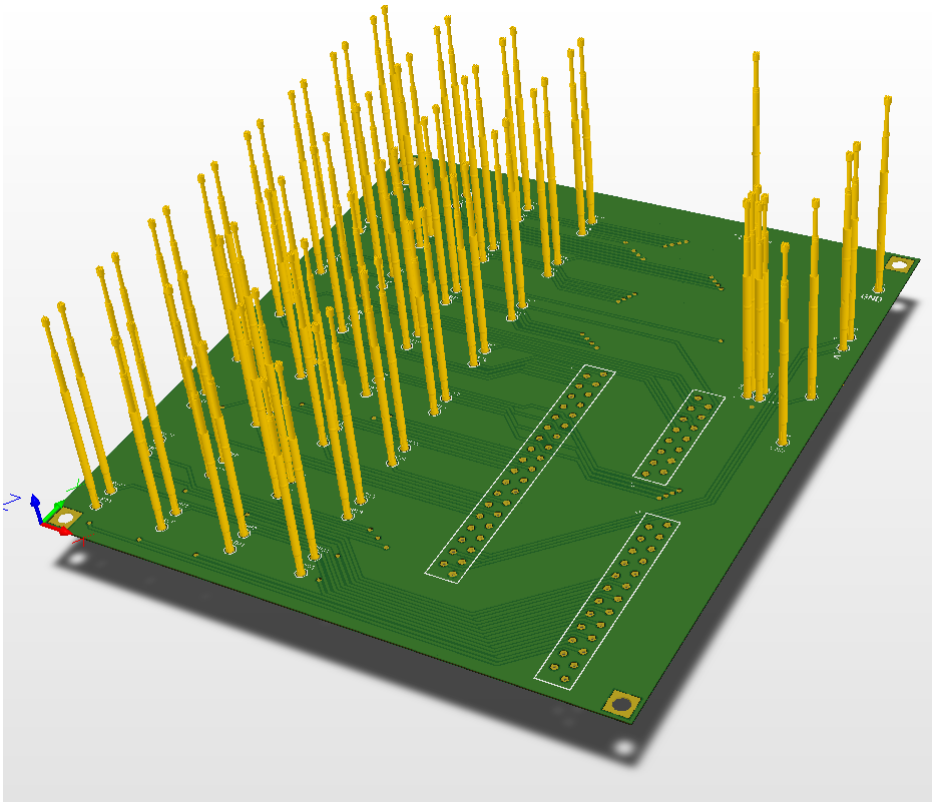


Fonte: Autor.

O esquemático da placa de interface, com as ligações feitas das agulhas para os conectores encontra-se no Apêndice C.

A visualização tridimensional da placa finalizada pode ser observada na Figura 19. As vistas detalhadas das camadas superior (*top*) e inferior (*bottom*) da Placa de Controle encontram-se disponíveis no Apêndice D.

Figura 19 – Renderização 3D da placa de interface



Fonte: Autor.

3.6 Desenvolvimento do *firmware* da jiga

A arquitetura de *software* da jiga de testes é estruturada em dois níveis distintos de abstração. A camada de baixo nível, executada pelo microcontrolador da placa de controle, transforma o dispositivo em um simulador de *hardware* para a automação dos testes. Esta etapa é responsável pela aquisição de dados, realizando a simulação física do acionamento de solenoides e a leitura de sensores. O *firmware* opera como uma interface direta de *hardware*, comunicando-se via protocolo serial com a camada de aplicação superior, hospedada na Raspberry Pi.

A divisão de tarefas adotada neste projeto corrobora a arquitetura proposta por Rodrigues (2009), que sugere a implementação de uma camada de acesso ao *hardware* distinta da camada de aplicação. Essa separação assegura que o gerenciamento crítico dos periféricos (realizado pelo microcontrolador) permaneça isolado das rotinas de alto nível e interface com o usuário (executadas na Raspberry Pi), garantindo maior confiabilidade e modularidade ao sistema.

O desenvolvimento do *firmware* para o microcontrolador ATmega328PB foi pautado na necessidade de criar uma interface simples e previsível entre o ambiente de controle digital e os atuadores físicos. A arquitetura de *software* organiza-se em duas frentes principais: o protocolo de comunicação externa e a lógica interna de

execução.

3.6.1 Interface de comunicação da jiga

A camada de comunicação transforma o dispositivo em um simulador de *hardware* dedicado. O *firmware* opera como uma interface direta, recebendo requisições da camada de aplicação superior via interface serial.

A comunicação foi implementada sob um protocolo baseado em caracteres ASCII, decisão que prioriza a legibilidade e facilita a depuração em campo. O delimitador de instruções é o caractere de nova linha, o que permite ao *firmware* isolar e interpretar comandos de maneira sequencial.

O Quadro 3 detalha a estrutura do protocolo, apresentando a sintaxe dos comandos e as respostas esperadas.

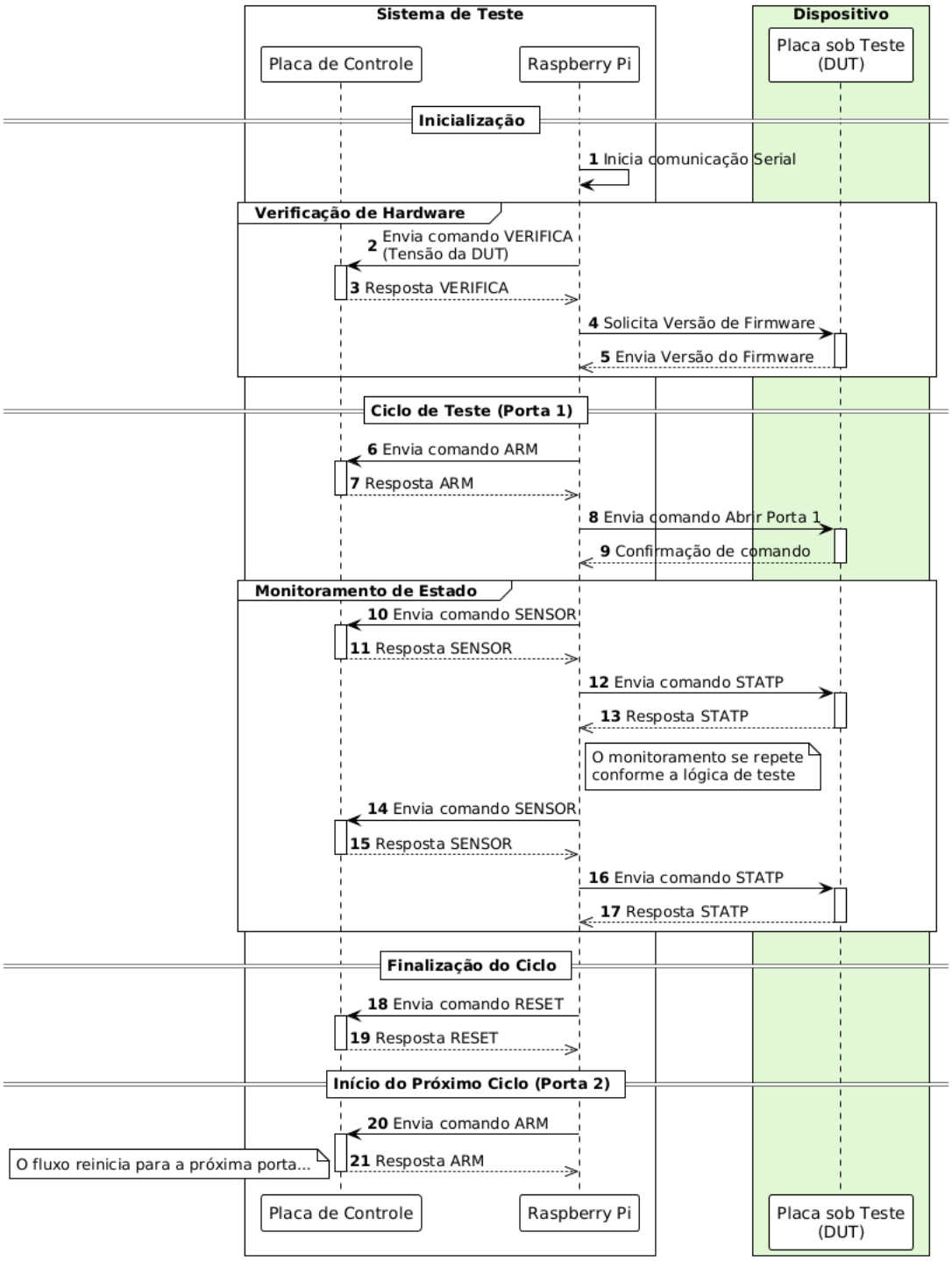
Quadro 3 – Protocolo de comunicação do *firmware* da jiga

Comando	**Descrição Funcional**	**Respostas Possíveis**
VERIFICA	Lê a tensão no pino de verificação para validar se a placa sob teste está acoplada e se a alimentação (5 V) está estável.	VERIFICA=OK: Tensão na faixa (> 4.5 V). VERIFICA=LOW: Tensão insuficiente.
ARM<N>	Prepara (“arma”) o simulador para a porta <N> (ex: ARM12). Define o sensor com nível baixo e ativa a *flag* de monitoramento.	ARM<N>=OK: Porta armada com sucesso. ARM<N>=INVALID_PORT: Porta fora da faixa (1-32).
SENSOR	Alterna (*toggle*) o estado do sensor da porta armada.	SENSOR=0: Novo estado *LOW*. SENSOR=1: Novo estado *HIGH*. SENSOR=INVALID_PORT: Porta fora da faixa (1-32).
RESET	Comando global. Reinicializa o monitoramento e zera a seleção de portas para evitar interferência entre testes.	RESET=OK: Estado reinicializado com sucesso.

Fonte: Autor.

Este protocolo provê o ferramental necessário para coordenar os testes. A Figura 20 ilustra o diagrama de sequência UML, onde a Raspberry Pi atua como dispositivo mestre, requisitando o monitoramento de pontos específicos enquanto interage simultaneamente com o dispositivo sob teste.

Figura 20 – Diagrama de sequência UML



Fonte: Autor.

3.6.2 Arquitetura de *Software* e Fluxo de Execução

Internamente, o *firmware* foi desenvolvido sob uma arquitetura de laço de controle contínuo (*super-loop*). Priorizando a simplicidade, o código opera em uma estrutura sequencial de verificação por *polling*.

Conforme ilustrado no fluxograma da Figura 21, o ciclo de execução inicia-

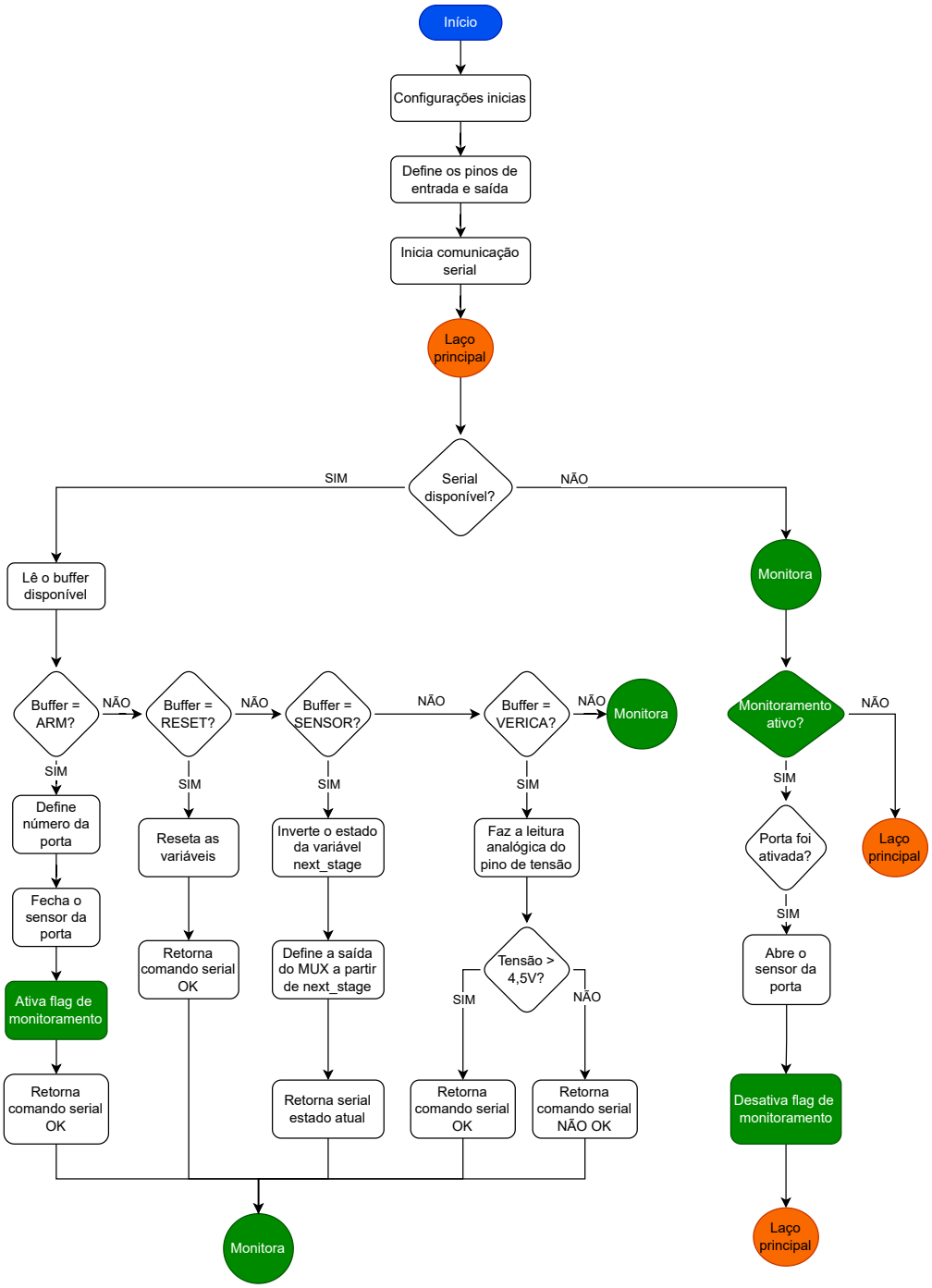
se com a configuração dos periféricos de entrada e saída. Após esta etapa, o sistema entra no laço principal, dividindo-se em duas rotinas críticas:

1. Tratamento de comunicação: O laço verifica a disponibilidade de dados no *buffer* serial. Caso haja uma requisição, o sistema identifica comandos como ARM (para armar o teste), SENSOR (teste de atuadores) ou *RESET*, retornando uma resposta imediata ao controlador mestre.
2. Rotina de monitoramento: Independentemente da comunicação, o fluxo converge para a verificação física. Se uma porta estiver "armada", o *firmware* lê o estado da entrada digital correspondente. Ao detectar o acionamento, executa a abertura do sensor e desativa a *flag* de monitoramento, completando o ciclo de teste.

A função primordial deste módulo do *firmware* é validar a simulação de carga da solenoide com fidelidade. Após o recebimento do comando `ARM`, o microcontrolador entra em estado de monitoramento contínuo no ponto de teste específico, aguardando o sinal de atuação.

Ao detectar a energização da solenoide por parte da DUT, o *firmware* comuta imediatamente o sensor, emulando o comportamento mecânico real de abertura da fechadura. Nesta etapa, devido ao assincronismo natural entre o processamento da jiga e da DUT, podem ocorrer divergências na mensagem de retorno enviada à Raspberry Pi (podendo ser "*OK! - Abrindo*" ou "*OK! - Porta já está aberta*"). Independentemente da variação na resposta, o teste é considerado aprovado, pois ambas as mensagens confirmam que o *firmware* detectou corretamente a transição do nível de tensão, validando o acionamento eletrônico do circuito.

Figura 21 – Fluxograma *firmware* da jiga



Fonte: Autor.

3.7 *Software* de Gerenciamento

O *software* de alto nível, hospedado na Raspberry Pi desenvolvido em linguagem *Python*, utiliza o *framework* PySide6 (Qt for *Python*) para a construção da Interface Homem-Máquina (IHM). A escolha desta arquitetura justifica-se pela agilidade de desenvolvimento, pelo suporte nativo a comunicações assíncronas e pela facilidade de integração com o sistema operacional linux embarcado.

Este componente assume a função de gerenciamento global, sendo responsável não apenas pela interação com o usuário, mas também pela coordenação sequencial dos testes, pela interpretação dos dados recebidos do microcontrolador e pela geração dos relatórios de validação.

3.7.1 Lógica e Fluxo de Teste

Para assegurar a fluidez da interface gráfica e prevenir o congelamento da tela durante a comunicação serial, o *software* foi estruturado utilizando o conceito de *multithreading*. Uma *thread* define-se como o menor fluxo de processamento programado que pode ser gerenciado independentemente por um escalonador do sistema operacional, permitindo a execução concorrente de tarefas. A arquitetura foi dividida da seguinte forma:

- *Main GUI thread*: responsável exclusivamente pela renderização dos elementos gráficos e pela captura de eventos do *touchscreen*, garantindo a responsividade da interface.
- *Worker thread*: encarregada da lógica de processamento pesado, gerenciando a lógica do teste e a troca de mensagens pela serial com a placa de controle.

Essa separação de processos assegura que os comandos seriais permaneçam responsivos mesmo durante rotinas de espera (*polling*) ou aguardo de resposta da DUT.

A lógica de teste foi implementada seguindo um roteiro sequencial, projetado para validar as funcionalidades da DUT respeitando sua ordem de dependência lógica.

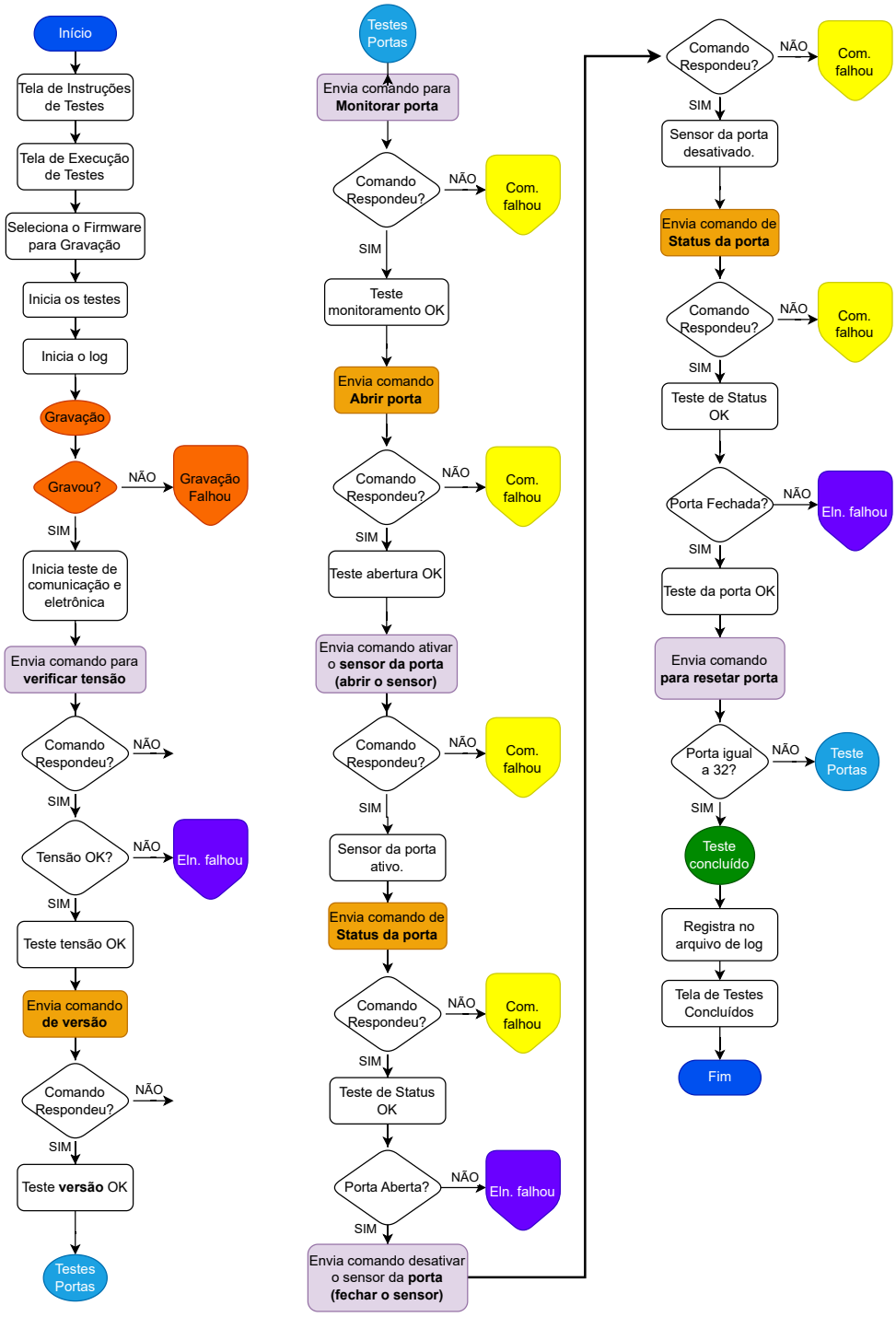
Devido à complexidade do processo e à necessidade de garantir que o sistema jamais entre em um estado indefinido, o algoritmo principal foi estruturado baseando-se no conceito de máquina de estados finitos (FSM - *Finite-State Machine*). Conforme detalhado no fluxograma da Figura 22, o processo inicia-se com a identificação da placa e evolui para a execução dos blocos de teste:

1. Inicialização: abertura da porta serial e realização do *handshake* para sincronizar a Raspberry Pi com o microcontrolador da jiga.
2. Testes estáticos: verificação preliminar dos níveis de tensão da DUT para garantir a integridade elétrica antes dos testes de carga.
3. Testes dinâmicos: comando de acionamento de cargas (solenoide) e leitura de retorno dos sensores.
4. Retorno: processamento dos dados coletados nas etapas anteriores e exibição do resultado final (sucesso ou falha) ao operador.

Para mitigar falsos negativos causados por ruídos de comunicação ou instabilidades momentâneas, o sistema implementa uma rotina de tratamento de erros.

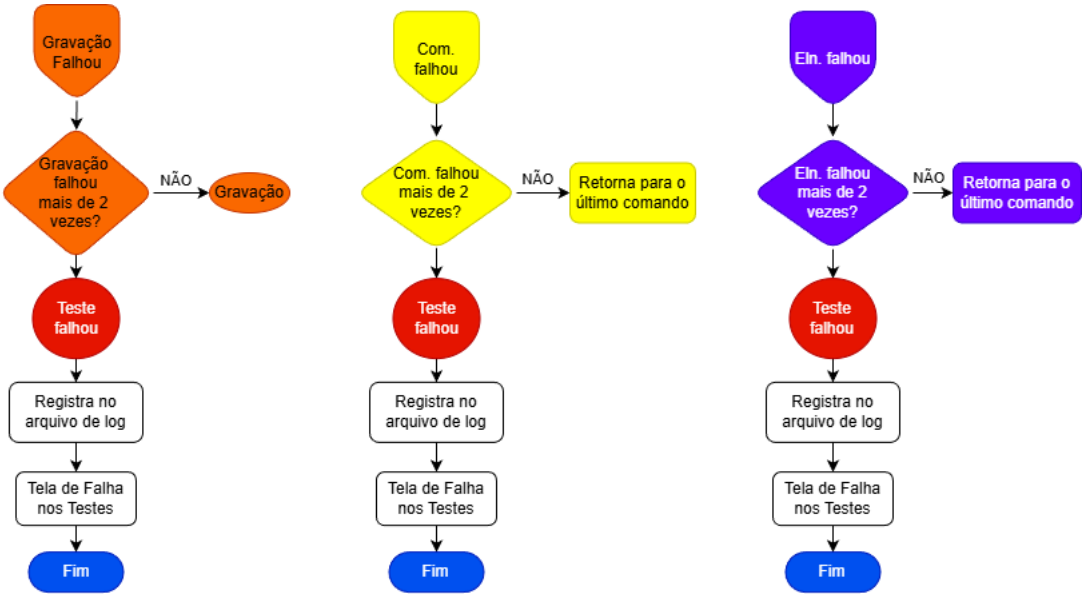
O algoritmo não reprova a placa imediatamente após uma falha de comunicação, em vez disso, ele utiliza um contador de tentativas. O código possui configurações de *timeout* (tempo limite de espera) para a resposta serial e um número máximo de retentativas para cada etapa. Conforme ilustra o fluxograma da Figura 23, caso ocorra um erro, o sistema incrementa o contador e repete o teste até que o limite seja atingido, momento em que a falha é definitivamente consolidada e reportada ao usuário.

Figura 22 – Fluxograma do processo principal de teste



Fonte: Autor.

Figura 23 – Lógica de retentativas e tratamento de falhas



Fonte: Autor.

3.7.2 Interface de *feedback* visual

A IHM foi projetada priorizando a objetividade e a clareza na comunicação com o operador da linha de produção. Em consonância com a necessidade de agilidade no processo industrial, a interface utiliza indicadores textuais explícitos para apresentar o veredito dos testes.

Para orientar a codificação da interface gráfica, foram elaborados protótipos de tela (*mockups*) na fase de projeto, definindo a ergonomia e a disposição dos elementos de controle. A estrutura de navegação foi consolidada em quatro abas principais, conforme detalhado a seguir:

1. Instruções (tela inicial): apresenta o roteiro de preparação e segurança, como mostra a Figura 24.

Figura 24 – Protótipo da tela inicial de instruções

Fonte: Autor.

2. Testes completos: centraliza a seleção de *firmware* e os comandos de gravação e teste automatizado, exibido na Figura 25.

Figura 25 – Protótipo da tela de execução dos testes

Fonte: Autor.

3. Testes unitários: permite o acionamento individual de canais para diagnósticos pontuais, como mostra a Figura 26.

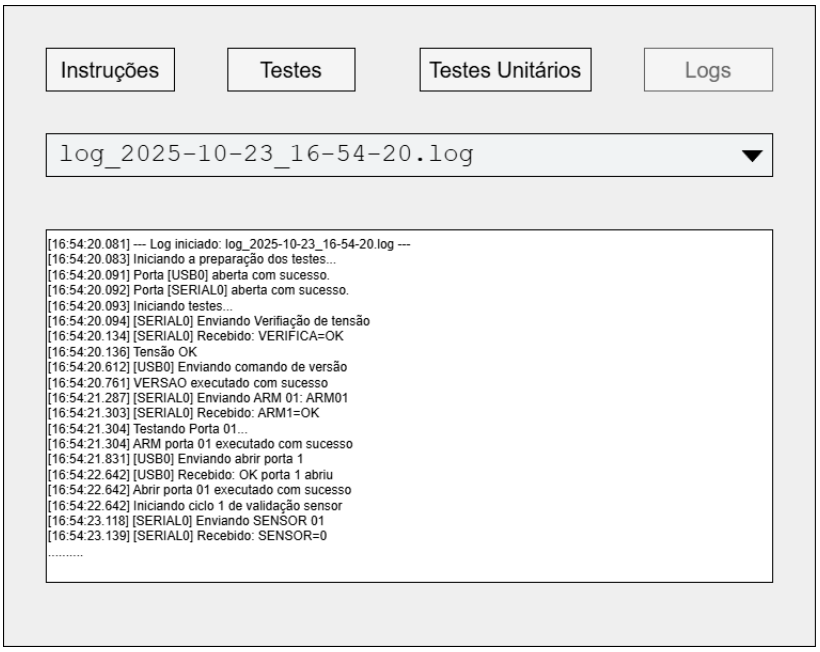
Figura 26 – Protótipo do menu de testes unitários

Fonte: Autor.

4. Logs: interface para gestão e visualização dos históricos de teste, exibido na Figura 27.

As Figuras de 24 a 27 ilustram a concepção visual proposta para estas etapas.

Figura 27 – Protótipo da tela de visualização de *logs*



Fonte: Autor.

Durante a execução, o resultado das operações é exibido instantâneo, informando de maneira direta o *status* final como “Sucesso” ou, em situações adversas, discriminando a falha e indicando a etapa específica onde o erro ocorreu. Essa abordagem textual elimina a ambiguidade na interpretação dos resultados, permitindo que o operador identifique imediatamente se a placa foi aprovada ou qual subsistema (gravação, eletrônica ou comunicação) necessita de revisão.

Além da visualização instantânea na tela, o *software* garante a rastreabilidade dos dados. Atualmente, os arquivos de *logs* armazenam data, hora e a descrição técnica das falhas, o que viabiliza análises estatísticas sobre a qualidade da produção e a recorrência de defeitos. Embora a identificação específica do operador não seja processada nesta versão inicial do sistema, a arquitetura de dados foi projetada para permitir a futura implementação de módulos de *login*. Tal funcionalidade agregaria valor estratégico à empresa, possibilitando o monitoramento de produtividade por turno e a identificação de eventuais necessidades de treinamento operacional.

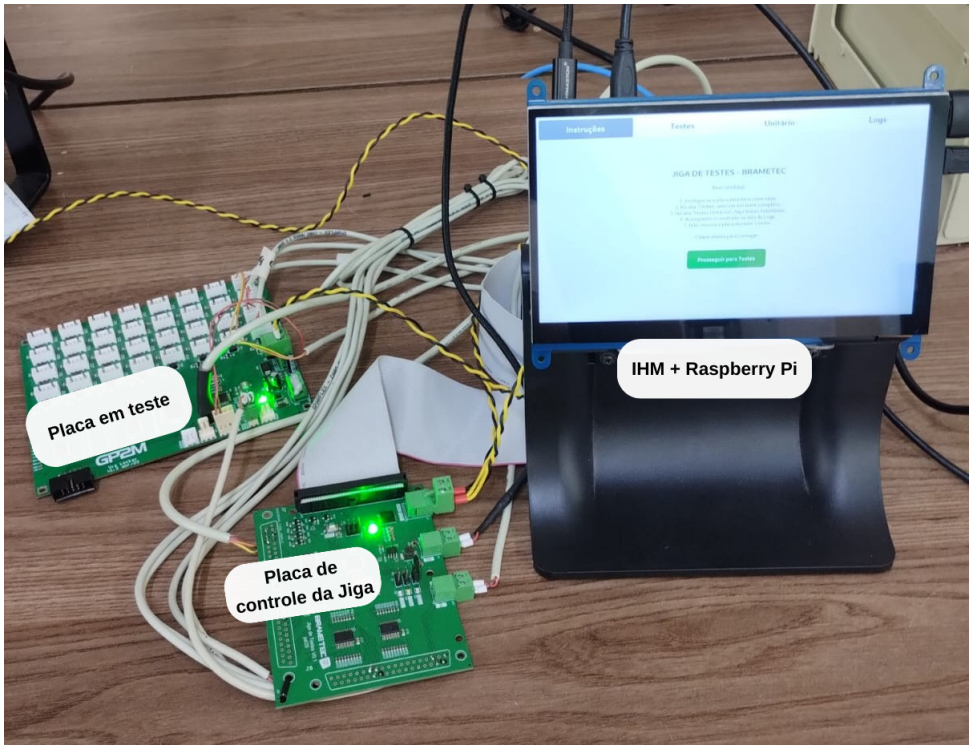
4 APRESENTAÇÃO DOS RESULTADOS

Nesta etapa, os resultados obtidos concentram-se na validação lógica e elétrica do sistema desenvolvido. Como o objetivo principal deste trabalho foi o desenvolvimento da arquitetura eletrônica e de *software* da jiga de teste, a validação funcional foi priorizada. Embora o projeto mecânico tenha sido elaborado para o acoplamento das agulhas de teste à placa sob teste, o protótipo físico do mecanismo de prensagem não foi manufaturado nesta fase.

Consequentemente, a apresentação dos resultados foi feita através de uma abordagem de validação por bancada. As conexões entre a placa de controle da jiga, o computador central (Raspberry Pi) e a DUT foram estabelecidas através de chicotes de cabos conectados diretamente aos pontos de teste. Esta metodologia permitiu isolar as variáveis do sistema, garantindo que o *firmware* e o *software* de controle estivessem plenamente funcionais e depurados, eliminando a incerteza de possíveis falhas de mau contato mecânico ou desalinhamento físico durante o desenvolvimento do código.

A validação do sistema foi realizada conforme o *setup* de testes ilustrado na Figura 28.

Figura 28 – *Setup* de testes

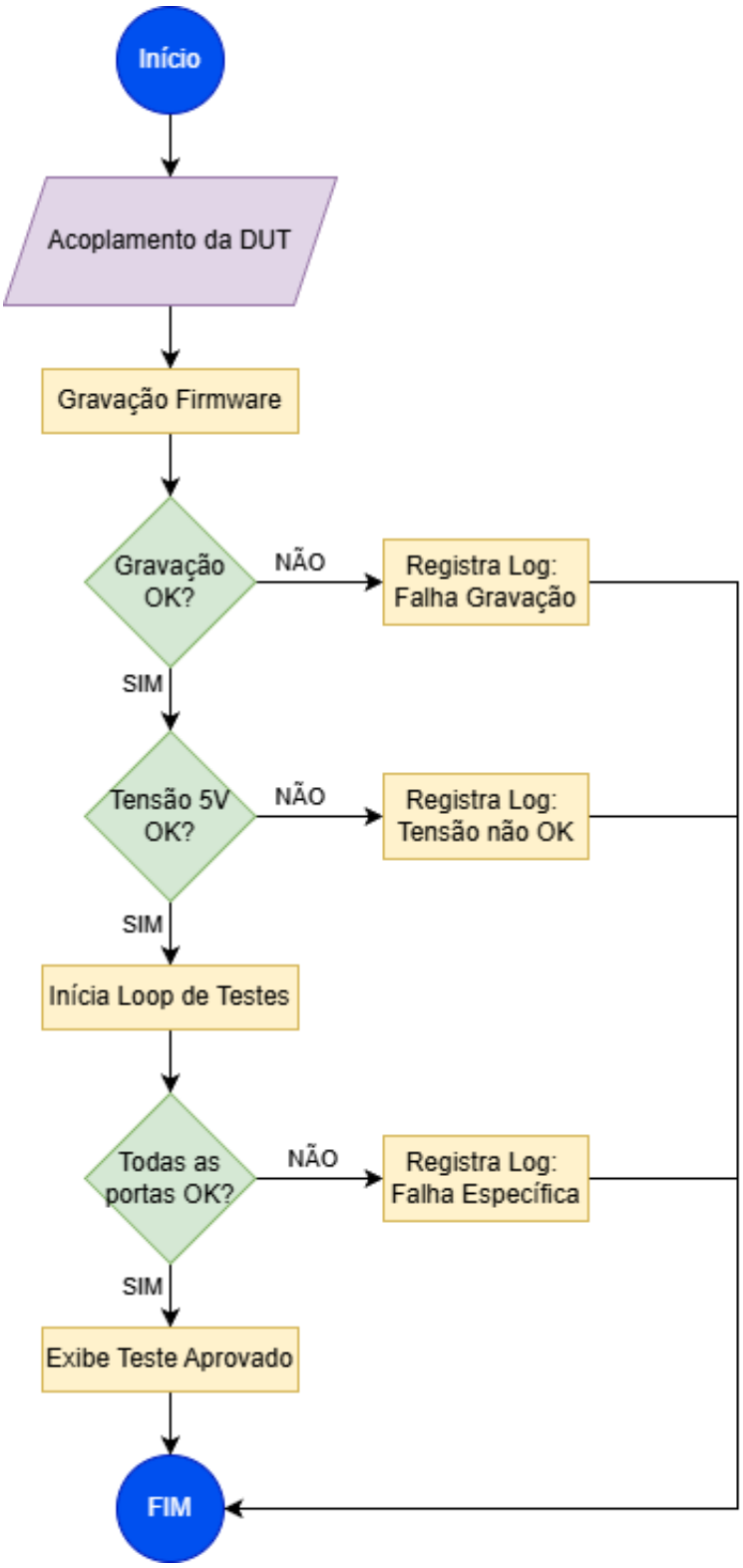


Fonte: Autor.

Os resultados apresentados a seguir demonstram a integridade funcional do sistema nessas condições controladas. O Fluxograma 29 detalha a sequência

lógica de etapas executadas automaticamente pelo *software* durante o processo de teste da jiga.

Figura 29 – Fluxograma do processo de teste automatizado

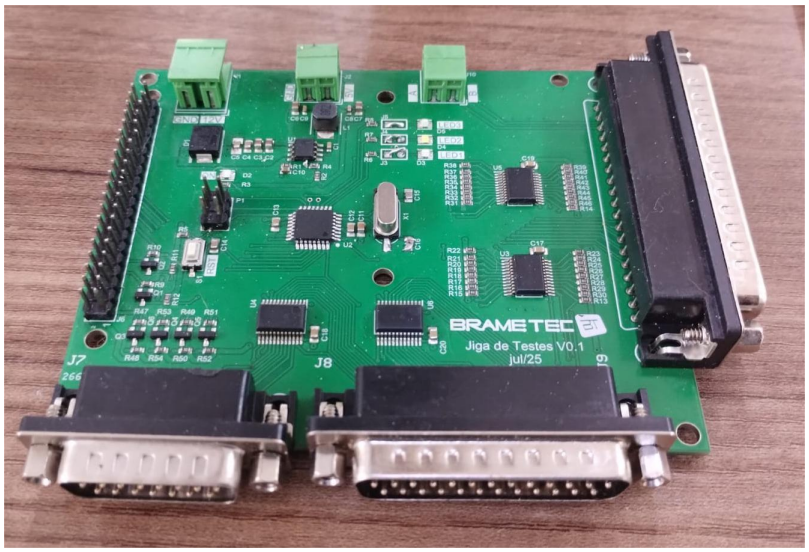


Fonte: Autor.

A placa de controle, responsável pelo chaveamento de sinais e intermédio

entre o processamento e a instrumentação, foi validada eletricamente. O resultado final da montagem da PCI é apresentado na Figura 30.

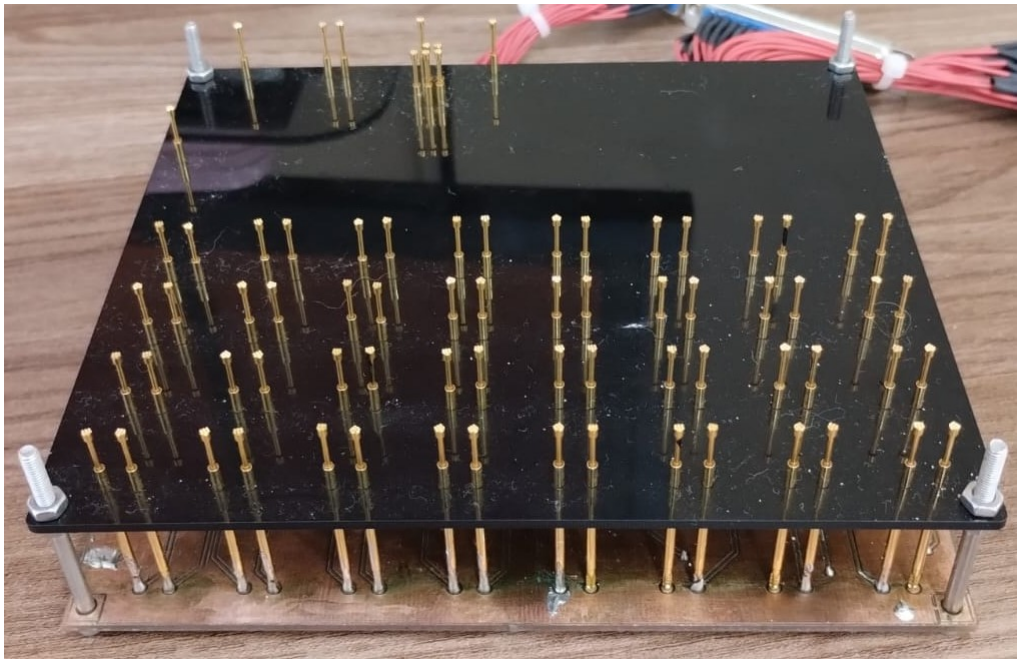
Figura 30 – Placa de controle



Fonte: Autor.

Paralelamente, a placa de interface (cama de agulhas) foi produzida para verificar a precisão dimensional do projeto. Na figura 31 é possível observar que além da PCI fabricada (parte inferior), foi adicionado um suporte de acrílico (parte superior preta) para dar mais estabilidade e precisão para as agulhas.

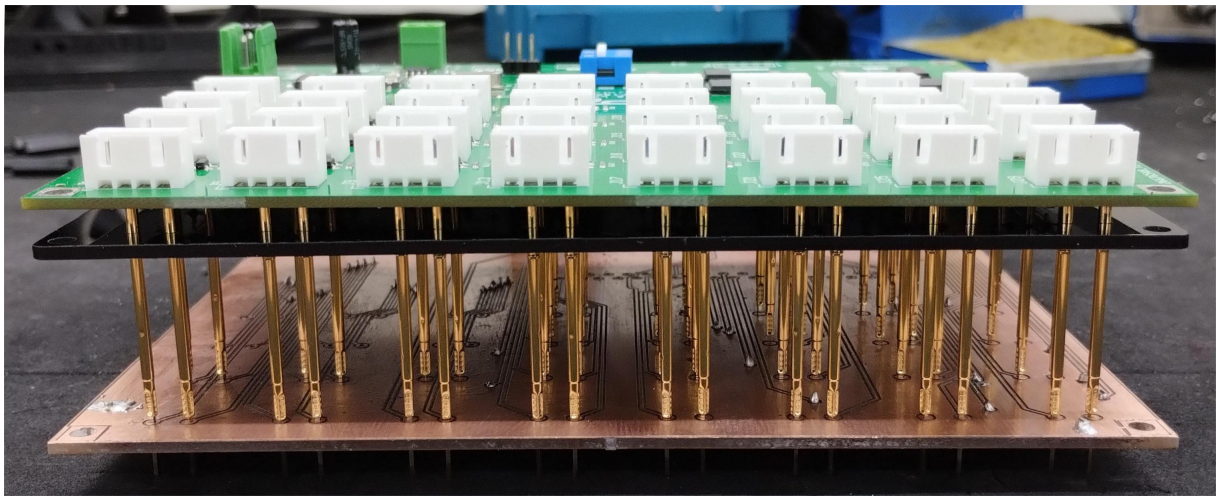
Figura 31 – Placa de interface



Fonte: Autor.

A Figura 32 mostra a placa de interface (parte superior) e a placa sob teste (parte superior), onde foi realizada uma inspeção visual e física, confirmando o alinhamento correto das agulhas com os respectivos *test points* da placa sob teste, validando o *layout* proposto.

Figura 32 – Placa de interface e DUT



Fonte: Autor.

Em relação à IHM, o *software* desenvolvido conta com uma estrutura de navegação composta por quatro abas. A operação inicia-se pela tela de introdução, que apresenta instruções de segurança e preparação para o operador. Ao acionar o botão “Prosseguir para Testes”, ilustrado na Figura 33, o sistema direciona o usuário para o ambiente de execução de testes.

Figura 33 – Tela inicial da aplicação

Fonte: Autor.

A tela de execução, exibida na Figura 34, centraliza as operações disponí-

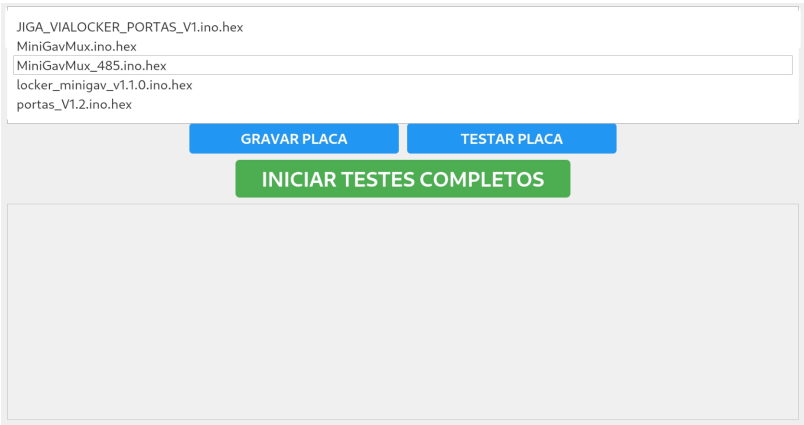
veis. O sistema permite flexibilidade operacional, oferecendo modos distintos: apenas gravação de *firmware*, apenas testes funcionais, ou o ciclo completo de testes, que otimiza o tempo de produção ao agrupar as duas etapas em um único comando.

Figura 34 – Tela de execução de testes

Fonte: Autor.

Para a etapa de gravação, o sistema exige a definição do arquivo de gravação a ser utilizado. A Figura 35 demonstra a interface de seleção, onde uma lista dinâmica de *firmwares* é apresentada. Esta funcionalidade foi projetada prevendo a escalabilidade do projeto, permitindo que a mesma jiga seja adaptada para testar diferentes versões de produto ou placas distintas no futuro, bastando atualizar o banco de arquivos.

Figura 35 – Tela de seleção de *firmware*



Fonte: Autor.

Visando facilitar a manutenção e a depuração de falhas, a terceira tela oferece um menu de testes unitários (Figura 36). Este modo permite que o setor da engenharia da empresa ou o operador técnico acione individualmente portas específicas

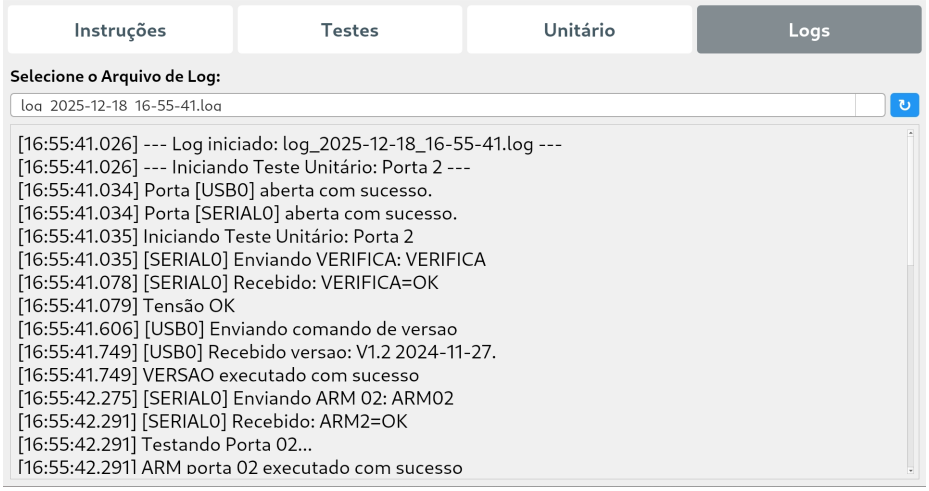
ou periféricos da placa, isolando problemas sem a necessidade de executar o ciclo completo de validação.

Figura 36 – Menu de teste unitário

Fonte: Autor.

A quarta tela é dedicada à exibição dos resultados e histórico. A Figura 37 exemplifica a saída de um *log* de operação, detalhando o status de cada etapa executada. O sistema classifica os resultados em sucesso ou falha, categorizando os erros em: falha de gravação, falha eletrônica ou falha de comunicação.

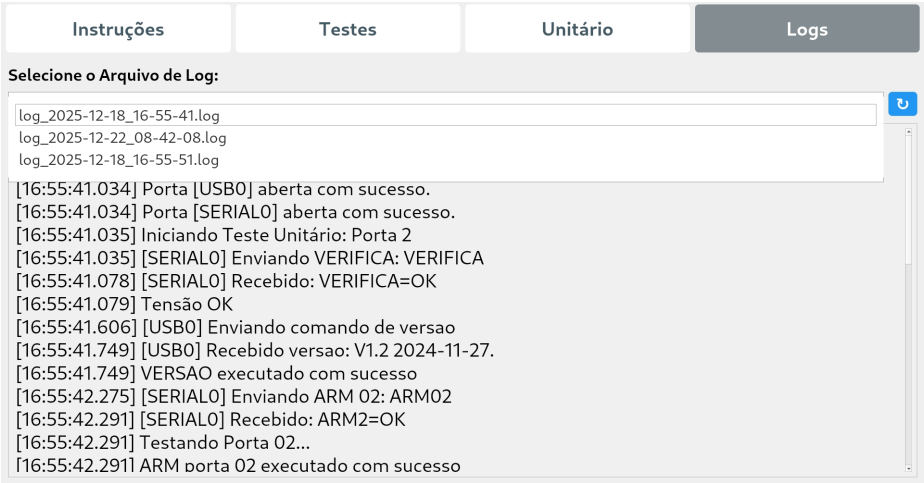
Figura 37 – Menu principal de *logs* e resultados



Fonte: Autor.

Além da visualização *online*, o sistema garante a rastreabilidade do processo através do registro dos dados. A Figura 38 exibe a interface de consulta a arquivos de *log* passados, permitindo auditoria e controle de qualidade sobre os lotes testados.

Figura 38 – Tela de visualização do histórico de *logs*



Fonte: Autor.

Por fim, visando a integração futura de todos os componentes em um ambiente de teste robusto e ergonômico, foi desenvolvido o projeto mecânico da estrutura. A Figura 39 ilustra a modelagem CAD do mecanismo. Embora não manufaturado nesta etapa, este projeto valida a viabilidade espacial e mecânica do sistema, garantindo que o acoplamento entre a eletrônica desenvolvida e a estrutura física seja compatível para implementações futuras.

Figura 39 – Proposta do projeto mecânico 3D renderizado da jiga de teste

(a) **Vista frontal** **(b)** **Vista isométrica**

Fonte: Autor.

4.1 Análise e Discussão dos Resultados

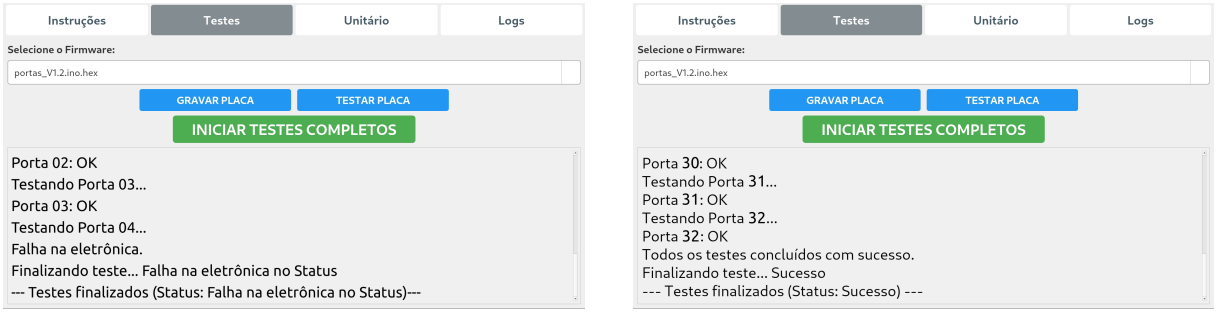
A validação da jiga de testes foi conduzida através da execução de ciclos completos de verificação, utilizando tanto placas controladoras padrão (conhecidas como *golden samples*) quanto unidades com falhas induzidas propositalmente. Os resultados obtidos demonstram a eficácia da integração entre o *hardware* de instru-

mentação e o *software* de controle, atendendo plenamente aos requisitos funcionais estabelecidos para o projeto.

4.1.1 Validação da Automação do Processo

A principal contribuição do sistema desenvolvido reside na padronização e repetibilidade do roteiro de testes. Conforme evidenciado na Figura 40, o sistema executa sequencialmente a verificação de grandezas elétricas, a gravação de *firmware* e os testes funcionais de periféricos sem qualquer intervenção humana entre as etapas.

Figura 40 – Comparativo do teste completo



(a) Resultado de falha **(b) Resultado de sucesso**

Fonte: Autor.

Diferente do método manual anterior, onde o operador necessitava validar individualmente cada componente sujeito a erros de interpretação, a jiga centralizou essas operações. A resposta visual binária e imediata (“Sucesso“ ou “Falha“) elimina a subjetividade na análise, garantindo que apenas as placas que cumpram todos os requisitos técnicos avancem para a etapa de montagem final do produto.

4.1.2 Processo de Detecção de Falhas

A capacidade de diagnóstico do equipamento foi validada através da simulação de erros comuns ao processo de manufatura eletrônica.

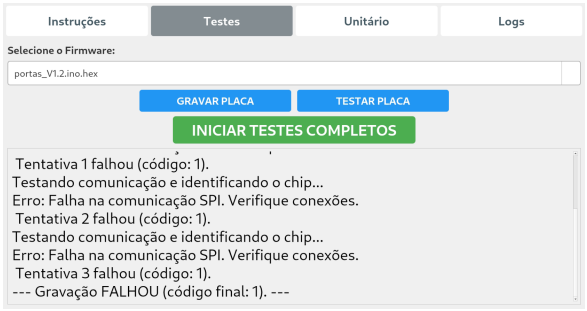
Inicialmente, foram avaliadas as falhas de comunicação. Conforme observado na Figura 41, ao interromper a linha de dados RS-485, o *software* identificou corretamente a ausência de resposta (*time-out*) após o número programado de tentativas. Isso confirma a validação da lógica de tratamento de erros, impedindo a aprovação de placas com transceptores defeituosos ou problemas no barramento de comunicação.

Figura 41 – Exemplo de detecção de falha de comunicação

Fonte: Autor.

Nos ensaios de gravação de *firmware* (Figura 42), o sistema demonstrou capacidade de validar a escrita na memória *flash* e também a correta configuração dos *fuses* do microcontrolador. O bloqueio do processo diante de uma falha de verificação assegura a integridade do *software* embarcado no produto final.

Figura 42 – Comparativo da tela de gravação



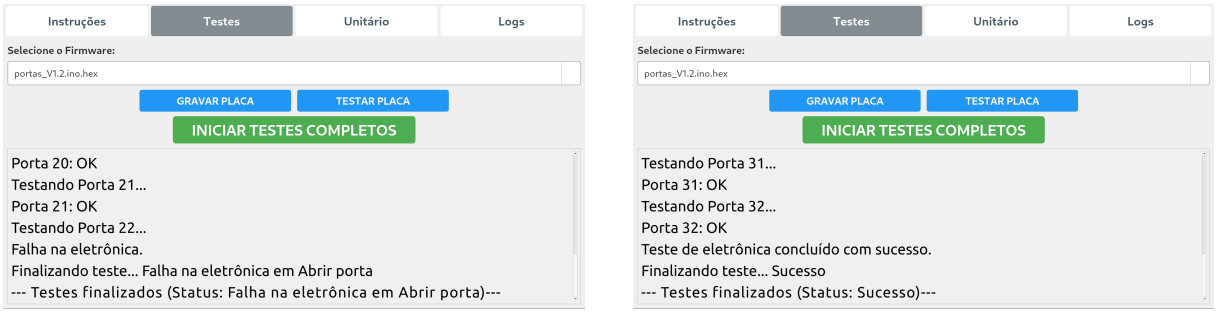
(a) Cenário de falha

(b) Cenário de sucesso

Fonte: Autor.

No que tange à validação do *hardware*, os testes eletrônicos apresentados na Figura 43 comprovaram a eficiência da jiga em interagir fisicamente com a placa sob teste. O algoritmo desenvolvido varre as portas de entrada e saída (I/O), verificando se os níveis lógicos correspondem aos estímulos enviados. No cenário de falha ilustrado, o sistema foi capaz de detectar um periférico que não respondeu ao comando de acionamento, simulando um componente danificado ou uma trilha rompida. Já no cenário de sucesso, confirmou-se a integridade elétrica de todos os circuitos testados, validando a montagem dos componentes discretos e a continuidade das conexões.

Figura 43 – Comparativo do teste de eletrônica



(a) Indicação de falha na eletrônica **(b) Indicação de sucesso na eletrônica**

Fonte: Autor.

A validação do *hardware* pode ser testada também na parte de Testes Unitários. O acionamento individual de uma porta, sem fazer a execução completa de todos os testes faz o teste ser mais rápido e objetivo, para validar apenas um ponto do circuito, para validar, caso haja, alguma exceção, conforme detalhado na Figura 44. Na indicação de falha, a aplicação testou para uma conexão elétrica que estava parcialmente incorreta. Em contrapartida, o cenário de sucesso demonstra a execução fluida do código quando todas as pré-condições lógicas são atendidas, assegurando a confiabilidade do veredito final emitido pela jiga.

Figura 44 – Comparativo do teste unitário

(a) Indicação de falha no teste **(b) Indicação de sucesso no teste**

Fonte: Autor.

4.1.3 Rastreabilidade e Análise de Dados

Um avanço significativo em relação ao processo manual é a implementação da geração automática de *logs*. O registro detalhado de cada etapa — incluindo *timestamps*, comandos enviados e respostas recebidas — constitui uma ferramenta vital para a gestão da qualidade. A análise desses dados permitiu verificar a estabilidade da comunicação serial entre a Raspberry Pi e a Placa de Controle, validando a arquitetura mestre-escravo proposta. Além disso, a persistência dessas informações possibilita ao setor de engenharia identificar falhas recorrentes em lotes específicos,

transformando o teste final em uma fonte de dados estatísticos para a melhoria contínua do processo produtivo.

4.1.4 Resumo da Cobertura de Testes

O Quadro 4 consolida o estado atual de implementação das funcionalidades de teste da Jiga, diferenciando o que foi validado em bancada do que permanece como proposta para etapas futuras.

Quadro 4 – Resumo dos testes e status de implementação

Funcionalidade/Teste	**Status**	**Observação**
Alimentação (12 V e 5 V)	Realizado	Validado via *hardware* e *firmware*.
Gravação de *Firmware* (ICSP)	Realizado	Integração com avrdude funcional.
Comunicação Serial (RS-485)	Realizado	Validação de integridade de dados.
Teste de Acionamento de Solenoides	Realizado	Simulação de carga ok.
Teste de Leitura de Sensores (*Feedback*)	Realizado	Lógica de *pull-up* validada.
Interface Homem-Máquina (IHM)	Realizado	Navegação e *logs* funcionais.
Acionamento Mecânico	Pendente	Projeto 3D finalizado; manufatura futura.
Integração com Banco de Dados Nuvem	Sugestão	Previsto para expansão futura.

Fonte: Autor.

4.2 Dificuldades Encontradas e Soluções Implementadas

Durante o desenvolvimento da jiga, surgiram desafios técnicos que exigiram ajustes estratégicos na arquitetura original. A principal dificuldade residiu na elaboração de um processo de verificação rigoroso que mitigasse a ocorrência de “falsos sucessos” — situações em que a placa sob teste poderia ser aprovada mesmo com falhas latentes. Para solucionar esse problema, optou-se pela segmentação detalhada dos ensaios. O primeiro estágio valida se o comando de abertura da porta foi efetivamente executado pela DUT, enquanto o segundo estágio foca exclusivamente na alternância de estados do sensor (aberto e fechado). Essa decomposição lógica não apenas ampliou a cobertura dos testes, mas também resolveu problemas de sincronização entre a placa sob teste e a placa de controle da jiga.

Somado a isso, outro desafio relevante envolveu o sincronismo entre a IHM e o *firmware*. Inicialmente, o *software* desenvolvido em *python* apresentava travamentos na interface ao aguardar o retorno de comandos de longa duração, como o processo de gravação de memória. Para contornar essa limitação, implementou-se uma

arquitetura baseada em *multithreading* do *framework* PySide6. Essa abordagem permitiu isolar o processamento da comunicação em uma linha de execução secundária, garantindo que a interface principal permanecesse responsiva e fornecesse *feedback* instantâneo ao operador durante toda a execução dos testes.

4.3 Considerações Finais do Capítulo

Os resultados apresentados confirmam que a arquitetura proposta para a jiga de testes automatizada é funcional. A validação em bancada demonstrou que tanto o *hardware* de controle quanto o *software* de interface cumprem os requisitos de velocidade, precisão e rastreabilidade exigidos pela Brametec.

Embora a estrutura mecânica de prensagem não tenha sido integrada fisicamente nesta fase, o alinhamento dimensional das agulhas e a modelagem 3D asseguram a viabilidade do sistema completo. A transição do método de teste manual para esta solução automatizada representa um ganho significativo na confiabilidade do processo produtivo, eliminando a subjetividade dos testes manuais e fornecendo dados valiosos para a melhoria contínua da qualidade das placas fabricadas.

5 CONSIDERAÇÕES FINAIS

O presente trabalho atingiu seu objetivo geral ao projetar e desenvolver uma jiga de testes automatizada funcional para as placas controladoras de armários inteligentes da Brametec. A integração proposta entre o *hardware* de instrumentação e o *software* de controle resultou em um equipamento capaz de realizar, de forma autônoma, as etapas críticas de gravação de *firmware* e validação funcional de periféricos.

A análise dos resultados confirma o cumprimento de todos os objetivos específicos estabelecidos. O levantamento dos requisitos técnicos permitiu o mapeamento preciso dos pontos de teste e tensões de operação da placa sob teste. Alinhado à estratégia de otimização de custos da empresa, o projeto utilizou componentes de baixo custo e recursos já disponíveis, garantindo a viabilidade econômica da implementação. No âmbito mecânico, a estrutura projetada assegurou o alinhamento dimensional necessário para o contato das agulhas de teste com a PCI, validado por meio da inspeção física dos protótipos de interface.

Quanto ao desenvolvimento técnico, o *hardware* de interface e condicionamento de sinais mostrou-se eficaz na interconexão entre a placa microcontrolada e a unidade de processamento central. O *software* de gerenciamento cumpriu seu papel ao oferecer uma IHM intuitiva, que abstrai a complexidade do sistema para o operador. Por fim, as rotinas de automação implementadas integraram com sucesso a gravação de *firmware* e a execução sequencial dos testes funcionais, eliminando a subjetividade do processo manual anteriormente utilizado.

5.1 Análise de Limitações e Abrangência

Embora os resultados confirmem a eficácia do sistema, é fundamental delinear as fronteiras de atuação do equipamento desenvolvido. A confiabilidade das validações realizadas pela jiga está intrinsecamente ligada à precisão do contato mecânico entre a interface de agulhas e a placa sob teste. Eventuais falhas de alinhamento ou desgaste dos *pogo pins* podem induzir a ocorrência de falsos negativos — caracterizados pela reprovação de uma placa funcional devido à ausência de continuidade elétrica momentânea nos pontos de teste.

Ademais, convém ressaltar que o escopo desta solução é predominantemente funcional e lógico. Portanto, anomalias paramétricas sutis, como variações de impedância em linhas de alta frequência ou desvios térmicos e de consumo de corrente que não comprometam o funcionamento imediato do circuito, podem não ser detectadas nesta versão. O sistema foca na garantia da integridade operacional e na correta gravação do *firmware*, delegando análises de estresse elétrico a etapas de testes mais específicas.

5.2 Perspectivas de Impacto Produtivo e Operacional

Sob uma análise qualitativa, a transição do método de teste manual para o sistema automatizado possui o potencial de representar uma mudança de paradigma para a linha de produção da Brametec. Embora a validação quantitativa (*time-study*) dependa da futura implementação do equipamento no ambiente fabril para a coleta de dados reais, as características do projeto permitem projetar ganhos significativos em eficiência e confiabilidade.

Enquanto o procedimento manual é inerentemente suscetível à fadiga do operador e à variabilidade de interpretação — fatores que elevam a probabilidade de erros e o risco de aprovação de itens defeituosos —, a jiga desenvolvida oferece um comportamento padrão. A execução padronizada do roteiro de testes assegura que cada unidade seja submetida a estímulos idênticos, garantindo uma repetibilidade que o processo manual não consegue atingir. Do ponto de vista de gestão, espera-se que essa padronização resulte na redução de custos com retrabalho e na mitigação de falhas em campo, impactando positivamente a garantia pós-venda.

Conclui-se que a automação proposta fornece a infraestrutura necessária para a maturidade do controle de qualidade da empresa. Ao eliminar a subjetividade humana e instituir a rastreabilidade automática via geração de *logs*, o sistema deixa de ser apenas uma ferramenta de verificação e torna-se um gerador de dados para o setor de engenharia, fundamentando futuras tomadas de decisão baseadas em evidências técnicas.

5.3 Sugestões para trabalhos futuros

Visando a evolução deste equipamento e sua plena integração aos conceitos da Indústria 4.0, sugerem-se estudos complementares:

- Propõe-se a elaboração de um documento técnico completo, contendo esquemáticos atualizados, listas de materiais e roteiros de montagem para permitir a replicação da jiga. Adicionalmente, recomenda-se a redação de um manual de operação e a estruturação de um programa de treinamento para os operadores, garantindo o uso correto do equipamento e a segurança durante os ensaios.
- Sugere-se a implementação de um módulo de autenticação na IHM. Isso permitiria associar cada ciclo de teste a um operador específico, possibilitando o monitoramento de produtividade por turno e facilitando auditorias em casos de falhas detectadas em campo.
- Recomenda-se a realização de uma análise na linha de produção para quantificar o ganho de eficiência operacional (placas por hora) e validar a eficácia na

detecção de falhas através de ensaios com um "lote de ouro"contendo defeitos conhecidos e mapeados.

- Propõe-se o aprimoramento do *firmware* para incluir testes de impedância entre as saídas dos multiplexadores antes da energização total da placa sob teste. Dado o espaçamento reduzido dos componentes SMD (*Surface Mount Device*) (aprox. 0,305 mm), essa medida é vital para detectar curtos-circuitos de solda, protegendo tanto a jiga quanto a placa sob teste de danos irreversíveis.
- Sugere-se a implementação de um sistema de etiquetagem automática. Acoplado a uma impressora térmica na Raspberry Pi, o sistema geraria etiquetas com número de série e versão de *firmware* para unidades aprovadas.
- Recomenda-se a migração do armazenamento dos *logs* (atualmente em texto) para um banco de dados relacional ou em nuvem. Isso permitiria à engenharia monitorar falhas recorrentes, gerando indicadores de qualidade para retroalimentar o processo de *design* do produto.

REFERÊNCIAS

BARRETO, J. dos S. *et al. Interface humano-computador*: [recurso eletrônico]. Porto Alegre: SAGAH, 2018. E-book. ISBN 9788595027374. 30, 31

BERNARDO, P. C.; KON, F. A importância dos testes automatizados. *Engenharia de Software Magazine*, Rio de Janeiro, v. 1, n. 3, 2008. 25

Brazil Connex. *Agulhas de contato para testes PCB: agulhas para ICT e FCT*. [S.l.], 2025. Disponível em: https://brazilconnex.com.br/assets/pdf/completo-agulhas-de-teste-ICT-e-PCB.pdf. Acesso em: 16 fev. 2026. 27

DEAN, B. S. *et al. AVRDUDE: AVR Downloader/UploaDEr*. 2025. Disponível em: https://github.com/avrdudes/avrdude. Acesso em: 10 nov. 2025. 43

Elo Print. *POL-2*. 2026. Catálogo de Produtos. Disponível em: https://www.eloprint.com/pol-2/. Acesso em: 5 jul. 2025. 22, 23

FARIAS, C. de. *Desenvolvimento de uma jiga de testes automatizados para componentes eletrônicos digitais utilizando o Robot Framework*. 87 p. Monografia (Trabalho de Conclusão de Curso (Graduação em Engenharia Eletrônica)) — Instituto Federal de Santa Catarina, Florianópolis, SC, 2023. 22, 31

Feinmetall GmbH. *Contact Probes for ICT and FCT*: Technical catalogue. Herrenberg, Alemanha, 2021. 28

Feinmetall GmbH. *Contact Probes: Products and Solutions*. Herrenberg, Alemanha: [s.n.], 2025. Disponível em: https://www.feinmetall.com/en/products/contact-probes. Acesso em: 10 dez. 2025. 28, 29

FLORIANI, H. C. *Desenvolvimento de uma jiga de teste automatizada para teste de fontes chaveadas*. Dissertação (Trabalho de Conclusão de Curso) — Universidade Federal de Santa Catarina, 2023. Acesso em: 16 fev. 2026. Disponível em: https://repositorio.ufsc.br/handle/123456789/255074. 23

FREEMAN, H. Software testing. *IEEE Electronic Library (IEL) Journals*, v. 5, n. 3, 2002. 26

ISTQB. *Syllabus Agile Tester*. [S.l.], 2014. 26

ISTQB. *Advanced Level Syllabus*: Test automation engineer. [S.l.], 2016. Versão 2016. 17

Keysight Technologies. *Choosing the Right Path for PCB Test*: White paper. Santa Rosa, CA, 2023. 24, 25

Magellan Circuits. *What Problems Identified by Functional Testing of PCBs?* 2026. Disponível em: https://pt.magellancircuits.com/what-problems-identified-by-functional-testing-of-pcbs/. Acesso em: 7 dez. 2025. 24

Microchip Technology Inc. *ATmega328P Datasheet*. Chandler, AZ, 2021. 32

National Instruments. *Fundamentals of Building a Test System*: A guide to automated test design. Austin, TX, 2024. Technical White Paper. 24, 25

NextPCB. *ICT vs FCT: PCBA Test Strategy*. Shenzhen, China: [s.n.], 2024. Disponível em: https://www.nextpcb.com/blog/ict-vs-fct-pcba-test-strategy. Acesso em: 5 jul. 2025. 26

NIELSEN, J. *Usability Engineering*. San Francisco, CA: Morgan Kaufmann, 1994. ISBN 9780125184069. 31

PRATES, R. O.; BARBOSA, S. D. J. Introdução à teoria e prática da interação humano-computador fundamentada na engenharia semiótica. *In*: KOWALTOWSKI, T.; BREITMAN, K. (Ed.). Rio de Janeiro: SBC, 2007. p. 263–326. 30

QA Technology Company, Inc. *Tip Style Selection Guide*: Application note anq121-a. Hampton, NH, 2017. 28

Raspberry Pi Ltd. *Raspberry Pi 4 Model B Datasheet*. Cambridge, Reino Unido, 2024. 33

RODRIGUES, C. E. B. *Jiga Automática para Testes de Fontes Chaveadas*. Monografia (Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica)) — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009. Orientador: Prof. Dr. Yeddo Braga Blauth. 17, 19, 39, 49

Sary. *XG07E SARY Brand Low Power Consumption Solenoid*. Zhejiang, China: [s.n.], 2021. Disponível em: https://www.alibaba.com/product-detail/XG07E-SARY-Brand-Low-Power-Consumption_1600214700213.html. Acesso em: 20 jul. 2025. 39

STMicroelectronics. *STM32 32-bit ARM Cortex MCUs Documentation*. Genebra, Suíça, 2024. 32

Texas Instruments. *LMR33630 SIMPLE SWITCHER® 3.8-V to 36-V, 3-A Synchronous Step-Down Voltage Converter*: Datasheet. Dallas, TX, 2020. SNVSB09C. Disponível em: https://www.ti.com/lit/ds/symlink/lmr33630.pdf. Acesso em: 18 jul. 2025. 45

Texas Instruments. *CD74HCx4067 High-Speed CMOS Logic 16-Channel Analog Multiplexer and Demultiplexer*. Dallas, TX, 2024. Rev. D. SCHS209D. Disponível em: https://www.ti.com/lit/ds/symlink/cd74hc4067.pdf. Acesso em: 5 ago. 2025. 42

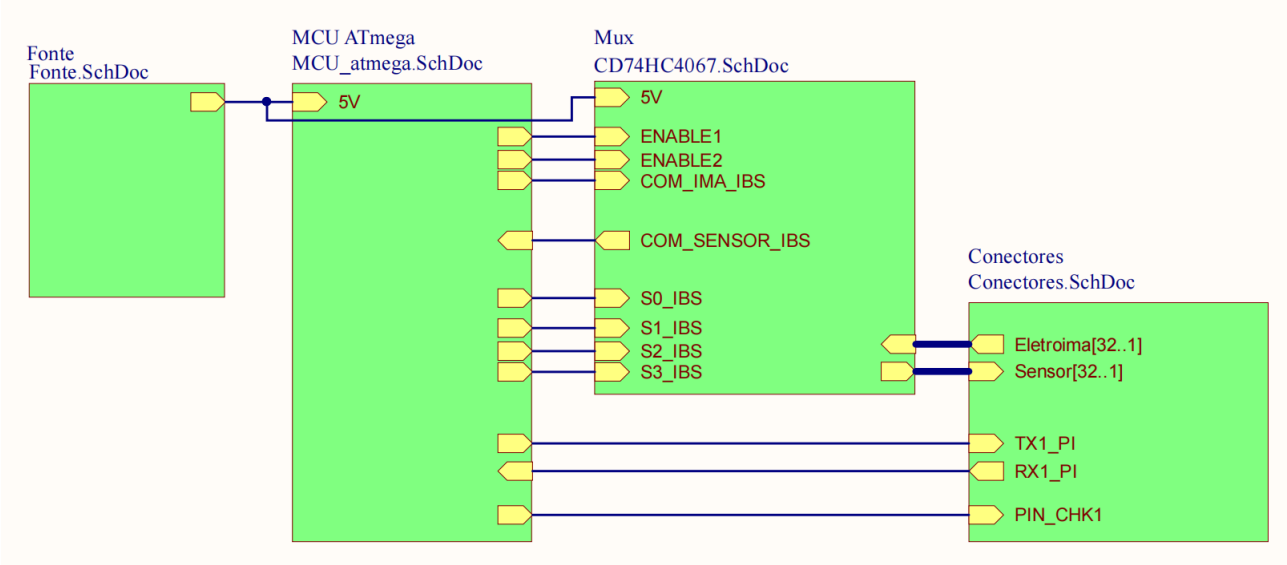
Texas Instruments. *WEBENCH® Power Designer*. Dallas, TX: [s.n.], 2025. Disponível em: https://webench.ti.com/power-designer/switching-regulator. Acesso em: 18 jul. 2025. 45, 46

The Qt Company. *Qt for Python (PySide6)*. Espoo, Finlândia: [s.n.], 2025. Disponível em: https://doc.qt.io/qtforpython-6/. Acesso em: 12 nov. 2025. 43

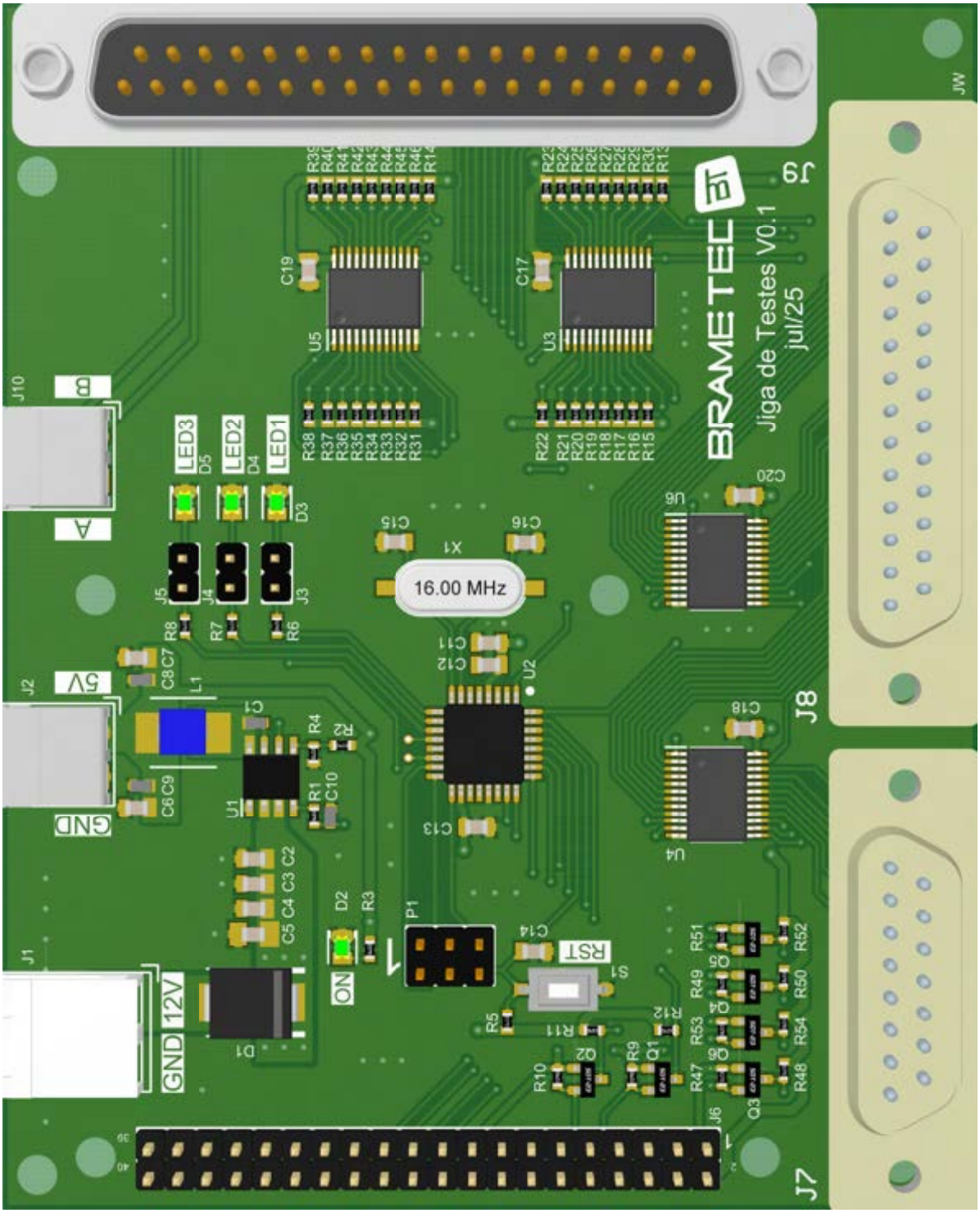
Victor Vision. *O que é IHM? Entenda para que serve e como funciona*. Brasil: [s.n.], 2024. Disponível em: https://victorvision.com.br/blog/ihm/. Acesso em: 15 dez. 2025. 30, 31

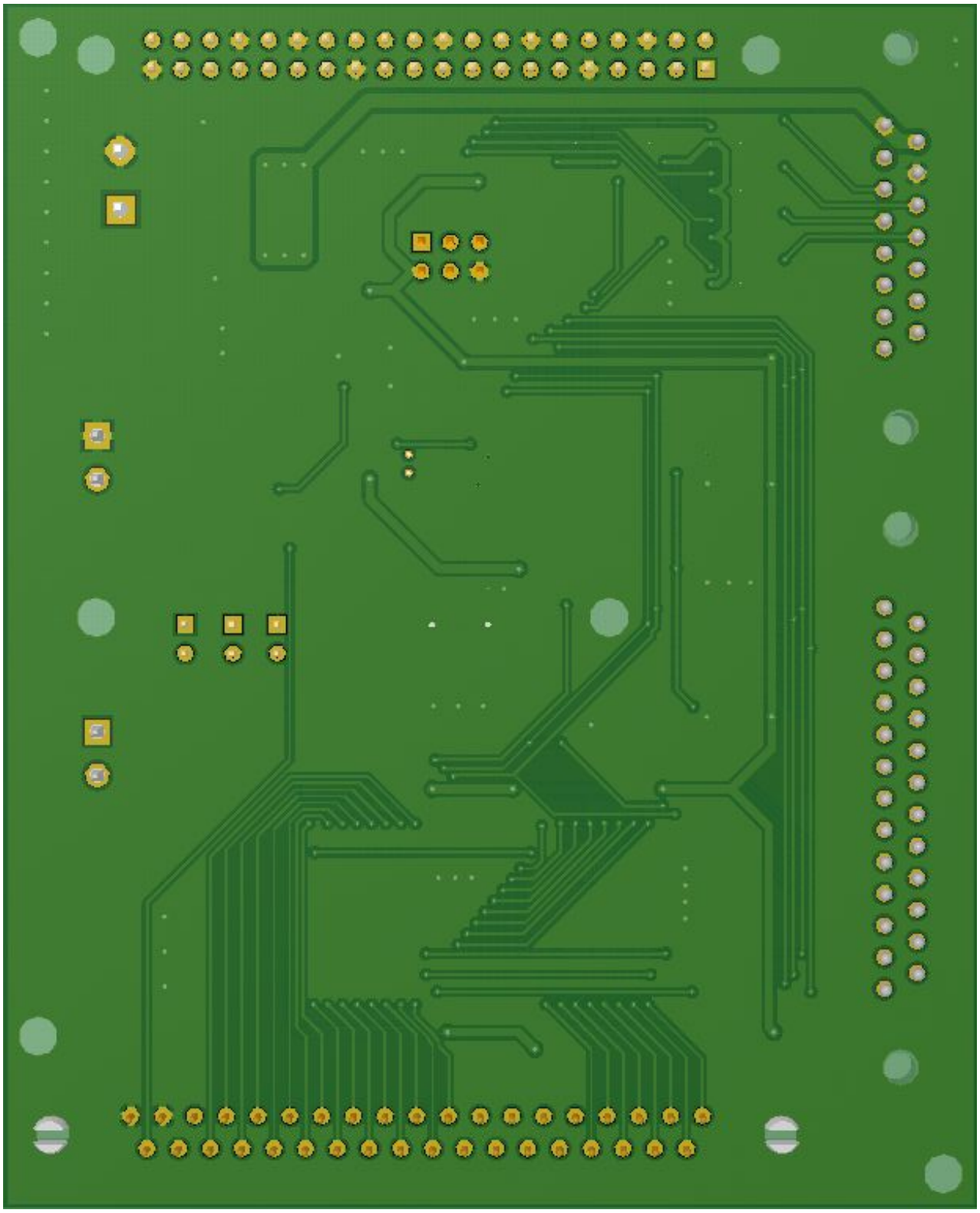
APÊNDICES

APÊNDICE A – DIAGRAMA DE BLOCOS PLACA DE CONTROLE DA JIGA

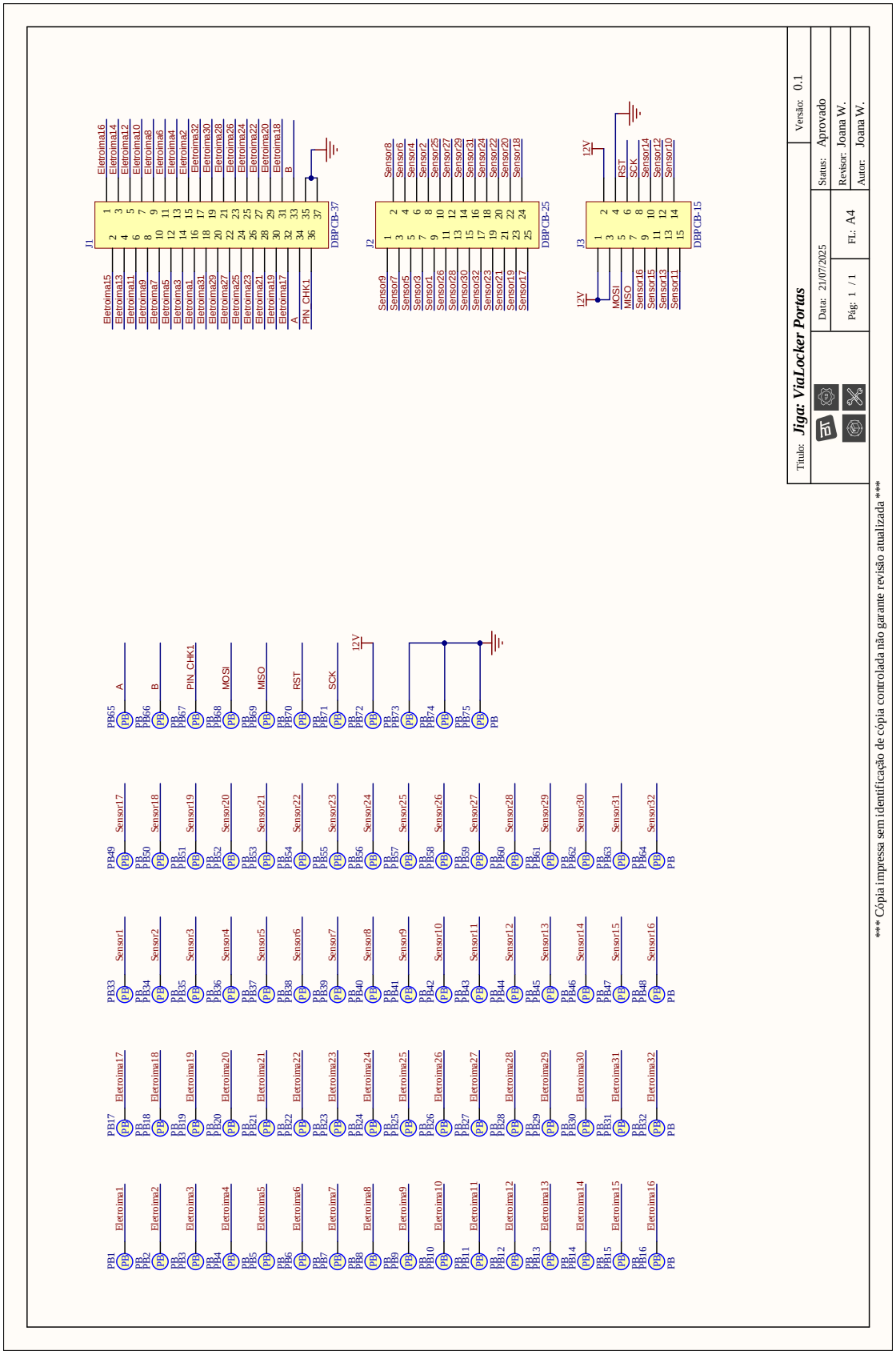


APÊNDICE B – LAYOUT PLACA DE CONTROLE DA JIGA





APÊNDICE C – ESQUEMÁTICO PLACA DE INTERFACE DA JIGA



APÊNDICE D – LAYOUT PLACA DE INTERFACE DA JIGA