APLICAÇÕES PRÁTICAS DO PADRÃO IEEE 1149.1 PARA PORTA DE ACESSO DE TESTE E ARQUITETURA DE VARREDURA DE LIMITE

Wellington da Silva Gaedke[1] e Renato Kazuo Miyamoto[2]

RESUMO

Este artigo aborda as aplicações do padrão IEEE 1149.1 (JTAG) para teste e reparo de placas de circuito impresso complexas, destacando os desafios impostos pela miniaturização e pela falta de acesso a diagramas esquemáticos. Apresenta a arquitetura de varredura de limites (boundary scan), descrevendo os componentes essenciais, como células de teste, controlador TAP e registradores de instrução, além da linguagem BSDL, essencial para se utilizar com eficácia o padrão. Utilizando ferramentas open-source (OpenOCD) e um depurador J-Link, demonstra-se como realizar o acesso e controle de registradores em um FPGA Lattice, detalhando a identificação de pinos, configuração de arquivos e execução de instruções. Os resultados comprovam a eficácia do método para controle de pinos, diagnóstico de interconexões e simulação de sinais, mesmo sem acesso à programação interna do circuito. O estudo evidencia a viabilidade de técnicas baseadas em JTAG para reparos em sistemas embarcados, utilizando ferramentas de baixo custo e contribuindo na manutenção em equipamentos críticos.

Palavras-chave: IEEE 1149.1; JTAG; boundary scan; OpenOCD; reparo de placas.

Practical Applications of the IEEE 1149.1 Standard for Test Access Port (TAP) and Boundary-Scan Architecture

ABSTRACT

This paper examines applications of the IEEE 1149.1 standard (JTAG) for testing and repairing complex printed circuit boards, focusing on challenges arising from miniaturization and limited access to schematic diagrams. It details the boundary scan architecture, describing key components including test cells, TAP controller, instruction registers, and the essential BSDL language for effective standard implementation. Using open-source tools (OpenOCD) and a J-Link debugger, the study demonstrates register access and control in a Lattice FPGA, covering pin identification, file configuration, and execution of instructions. Results validate the method's effectiveness for pin control, interconnection diagnostics, and signal simulation, even without access to the circuit's internal programming. The research confirms the viability of JTAG-based techniques for embedded system repairs using low-cost tools and contributing to the maintenance of critical equipment.

Keywords: IEEE 1149.1; JTAG; boundary scan; OpenOCD; PCB repair.

[1] Graduando em Engenharia Elétrica. E-mail: wellington.gaedke@outlook.com
[2] Doutorado em Engenharia Elétrica. E-mail: renato.miyamoto@sistemafiep.org.br

1 INTRODUÇÃO

O aumento da complexidade e miniaturização de sistemas embarcados e placas de circuito impresso, tem possibilitado grandes avanços e melhorias em sistemas de infraestrutura crítica, e consequentemente a popularização de tecnologias avançadas, em questão de velocidade e processamento, para o meio comum (Parker, 2003).

Essas questões geraram preocupações, já antes do início do século XXI, sobre como testar placas de circuito impresso (PCB, na sigla em inglês para *Printed Circuit Board*) de múltiplas camadas, cujos circuitos estavam sendo cada vez menores e localizados em *pads*, e cuja interferência dos testes nos sinais da placa precisava ser mínima. Para os fabricantes de PCB, o foco estava nesses desafios, enquanto para o usuário final, a preocupação era com a possibilidade de realizar reparos em placas sem ter o diagrama do circuito ou a programação dos dispositivos (Bennetts; Ley; Bales, 2022).

Em 1990 foi criado o padrão IEEE 1149.1, cuja última revisão foi feita em 2013. Esse padrão, denominado Porta de Acesso de Teste (TAP, na sigla em inglês para *Test Access Port*) e varredura de borda, descreve a estrutura de células de varredura, suas funcionalidades, a máquina de estado responsável pelo controle dessas células (TAP Controller) e a interface de comunicação (Protocolo JTAG), além da estrutura da documentação para se fazer uso dessas arquiteturas. Esse padrão é implementado pelos fabricantes dos Circuitos Integrados (CIs), e atualmente todos os circuitos FPGA, CPLD e circuitos com núcleo ARM e RISC-V possuem suporte a este padrão (Maunder; Tullos, 1990).

Dentre as funcionalidades que esse padrão fornece, está a possibilidade de controlar individualmente pinos de entrada/saída dos circuitos, podendo enviar padrões para simular sinais de controle ou comunicação. Também é possível capturar o registro das células de varredura através do qual é possível identificar possíveis padrões nos pinos, de protocolos de comunicação, por exemplo, sem a necessidade de ter conhecimento de como foi programado aquele CI. Além disso também é possível congelar o estado dos pinos do CI, fazendo com que se tornem

inalterados para sinais externos. Dentre outras instruções que são descritas nos arquivos BSDL específicos de cada CI (ARM Limited, 2022).

Existem diversas empresas que fornecem dispositivos de depuração com suporte ao JTAG. No entanto, esses dispositivos costumam estar atrelados aos *softwares* e/ou CI dos próprios fabricantes, além do *firmware* e, em casos como da empresa XJtag, aos arquivos Gerber da placa. Alguns casos, como por exemplo da Intel, são fornecidos apenas a funcionários.

Esses dispositivos costumam ser extremamente caros, e dado as limitações descritas anteriormente, são difíceis de torná-los viáveis em serviços de reparo. Porém existem alternativas baseadas em Chips de comunicação com suporte JTAG como o FT2232H da fabricante FTDI, que podem ser utilizados com *software* opensource como OpenOCD ou urJtag, ou alguns modelos básicos da Xilinx e SEGGER, que possuem suporte nós *software* citados. Apesar desses modelos muitas vezes não possuírem suporte à depuração de núcleo, o *software* OpenOCD possibilita o acesso distinto aos TAPs do núcleo e do Boundary Scan (FitzPatrick, 2019).

A grande dificuldade em utilizar modelos de depuradores genéricos e *software* opensource, é a complexidade da configuração, e a necessidade de conhecer em profundidade os aspectos relacionados a depuração do CI em questão e do *software*, pois essas ferramentas não costumam ser automatizadas ou possuírem interface gráfica (Rempel, 2019).

A dificuldade de se fazer reparos em placas complexas, ou mesmo diagnosticar sistemas com múltiplos módulos (sem realizar troca de peças), que em alguns casos podem custar centenas de milhares (como casos presenciados pelo próprio autor), alinhado a dificuldade em se obter ferramentas de diagnóstico avançadas no Brasil, justifica a necessidade de conhecer e divulgar a utilidade e usabilidade desse padrão com ferramentas opensource, o que será tema deste artigo (Santos, 2023).

Nesse sentido, a motivação desse trabalho consiste em apresentar um método eficaz e de relativo baixo custo para acessar e utilizar o padrão 1149.1 implementado em circuitos, destacando seus benefícios e possibilidades para o

reparo de placas de circuito. Para isso, foi utilizado o servidor de depuração on-chip OpenOCD e o depurador SEGGER J-link Base.

Uma breve introdução sobre as contribuições e as características do tema proposto nesse trabalho são apresentadas nesta seção. A sequência deste trabalho está organizada em 4 seções, a saber: a Seção 2 contempla a fundamentação teórica, abordando os principais conceitos e estudos relacionados ao tema; a Seção 3 descreve a metodologia adotada para o desenvolvimento da pesquisa; a Seção 4 apresenta e discute os resultados obtidos; por fim, a Seção 5 reúne as conclusões do estudo, destacando as contribuições e possíveis direções para trabalhos futuros.

2 ARQUITETURA DE VARREDURA DE LIMITES PELO PADRÃO IEEE 1149.1

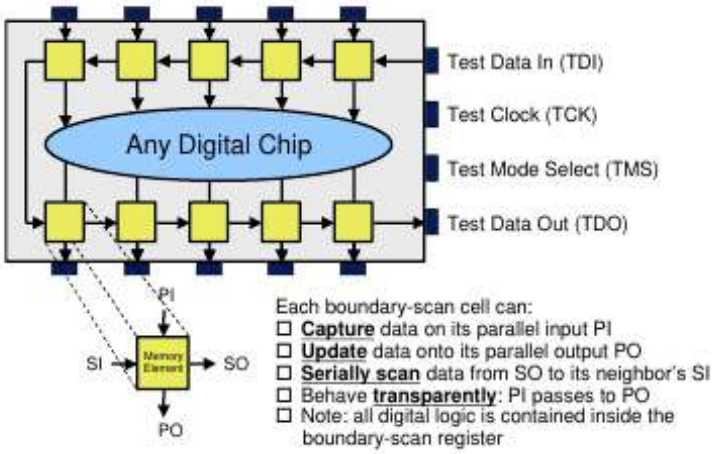
O padrão IEEE 1149.1, também conhecido como JTAG (*Joint Test Action Group*), foi desenvolvido com o objetivo de viabilizar o teste de circuitos integrados e placas de circuito impresso em contextos em que os métodos tradicionais de acesso físico se tornam limitados ou inviáveis. Esse padrão define uma Porta de Acesso de Teste (TAP – *Test Access Port*), composta por sinais dedicados (TDI, TDO, TCK, TMS e, opcionalmente, TRST), que permite o controle e a observação de dispositivos digitais de forma padronizada. Através dessa interface, é possível aplicar sequências de teste diretamente aos registradores internos dos componentes, sem a necessidade de sondas externas ou pontos de teste físicos, o que reduz a complexidade e o custo das etapas de verificação e depuração (IEEE, 2013).

A arquitetura de varredura de limite (*boundary scan*), especificada pelo mesmo padrão, introduz registradores de varredura ao longo das entradas e saídas dos circuitos integrados, possibilitando o controle e a leitura do estado lógico de cada pino de forma serial. Isso permite a realização de testes de interconexão entre dispositivos montados em uma placa, bem como a detecção de falhas como curtos-circuitos, pinos soltos ou conexões incorretas. Além dos testes estruturais, a arquitetura também é amplamente utilizada em aplicações de depuração de

sistemas embarcados e na programação de dispositivos como FPGAs e microcontroladores (Mazidi; Mazidi; Mckinlay, 2007).

Cada célula I/O (*Input/Output*) do dispositivo é suplementada com um elemento de memória chamado célula de varredura de limite, que são configuradas como registradores de deslocamento paralelos. Assim, cada célula pode realizar a captura de dados que são carregados pelo núcleo do dispositivo para suas saídas, e o carregamento de dados já existentes na célula de varredura diretamente para a respectiva saída do dispositivo, conforme ilustra a Figura 1 (Bennetts; Ley; Bales, 2022).

Figura 1 – Princípio da arquitetura de varredura de limites.



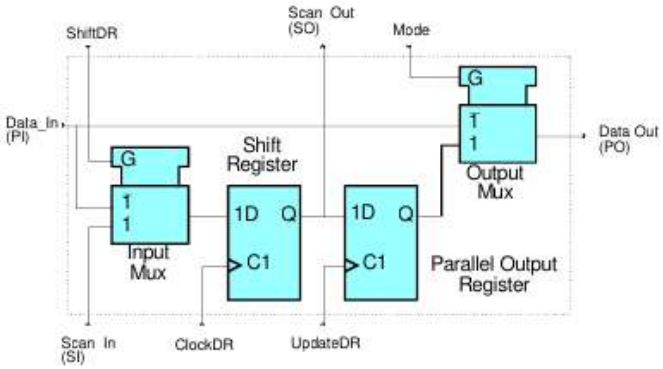
Fonte: Bennetts (2022).

Os dados podem ser enviados para as células através do registrador de deslocamento de maneira serial, iniciando por um pino dedicado de entrada do dispositivo, chamado “Test Data In” (TDI) e terminando e terminando em um pino dedicado de saída, chamado “Test Data Out” (TDO). Ainda há dois outros pinos dedicados, chamado “Test ClocK” (TCK) e “Test Mode Select” (TMS), que controla o modo de operação dos registradores (Bennetts; Ley; Bales, 2022).

A Figura 2 ilustra uma célula de varredura básica universal. A célula possui quatro modos de operação, a saber: normal, update, capture, e serial shift. O elemento de memória é mostrado como um flip-flop tipo-D com multiplexação de dados nas etapas inicial e final do processo. Durante o modo normal, a célula

transmite o dado de entrada diretamente para a saída. No modo capture, o valor presente no pino de entrada paralela (PI) é armazenado no flip-flop. No modo shift, os dados são deslocados em série pelo registrador de deslocamento. Por fim, no modo update, o valor armazenado é transferido para o registrador de saída, atualizando o pino de saída (PO). A multiplexação nas entradas e saídas possibilita essa operação flexível em diferentes etapas do processo.

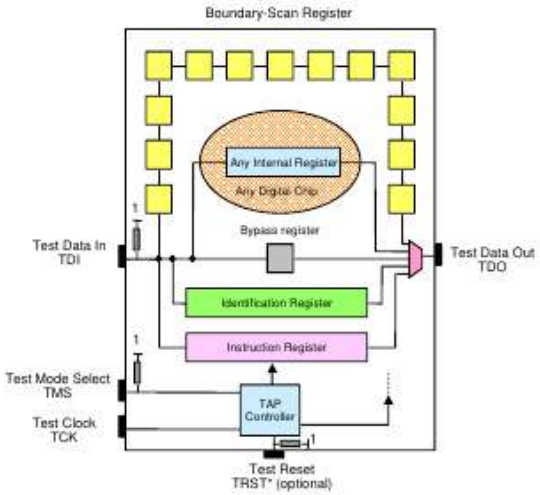
Figura 2 – Célula básica de varredura de limites.



Fonte: Bennetts (2022).

A Figura 3 mostra a arquitetura de chip da IEEE (*Institute of Electrical and Electronics Engineers*) 1149.1, proposta pela organização JTAG (Joint Test Action Group) após nove anos de discussão.

Figura 3 – IEEE 1149.1 Arquitetura de chip



Fonte: Bennetts (2022).

A Figura 3 apresenta os seguintes elementos:

- Um conjunto de quatro pinos dedicados – Test Data In (TDI), Test Mode Select (TMS), Test Clock (TCK), Test Data Out (TDO) e um pino opcional, Test Reset (TRST). Estes pinos são coletivamente chamados de Test Access Port (TAP);
- Uma célula de varredura de limites em cada pino de entrada e saída primários do dispositivo, conectados internamente para formar um registrador serial de varredura de limites;
- Uma máquina de estados finitos denominada Controlador TAP, com entradas TCK, TMS e TRST;
- Um Registrador de Instruções (IR) de n-bit (n >= 2);
- Um registrador de By-pass de 1-bit;
- Um registrador opcional de 32-bit, chamado Registrador de Identificação (Ident), que carrega um código de identificação permanente do dispositivo.

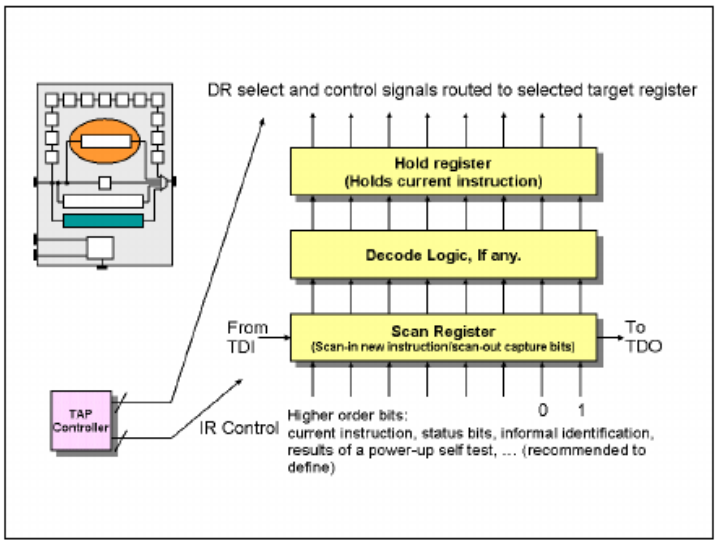
Em qualquer momento, apenas um registrador pode estar conectado de TDI para TDO. O registrador selecionado é detectado pela decodificação da saída do Registrador de Instrução (IR).

2.1 O REGISTRADOR DE INSTRUÇÕES

O Registrador de Instrução tem uma seção de deslocamento conectada ao TDI e ao TDO, e uma seção de retenção, que mantém a instrução atual, conforme ilustra a Figura 4.

Os sinais de controle do IR se originam do controlador TAP e podem causar um deslocamento para dentro (*shift-in*), um deslocamento para fora (*shift-out*) através da seção de mudança do IR, ou fazer com que o conteúdo da seção de mudança seja passado para a seção de espera (operação update). O IR deve ter pelo menos 2-bits de comprimento, para permitir a codificação das quatro instruções obrigatórias – By-pass, Sample, Preload, Extest – mas não tem um comprimento máximo definido.

Figura 4 – O Registrador de instruções



Fonte: Bennetts (2022).

2.1.2 OPERAÇÕES DO REGISTRADOR DE INSTRUÇÕES

As instruções são comandos padronizados que são interpretados pelos registradores e pelo TAP Controller. Há comandos obrigatorios, os quais o CI deve dar suporte para se adequar ao padrão IEEE 1149.1, e instruções opcionais, as quais o fabricante pode ou não implementar, mas cuja funcionalidade também é padronizada.

Instruções obrigatórias:

- BYPASS: Deve receber um código all-1s. Faz com que o registrador de Bypass seja colocado entre os pinos TDI e TDO.
- SAMPLE e PRELOAD: Selecionam o registrador Boundary-scan. A instrução SAMPLE configura as células para amostrar (capture) valores que se movem para o dispositivo. A instrução PRELOAD é usada para carregar valores conhecidos nas células de varredura. Os códigos para essas instruções não são definidos.
- EXTEST: Seleciona o registrador Boundary-scan preparatório para teste de interconexão. A partir da revisão de 2001 do padrão, não possui mais código definido.

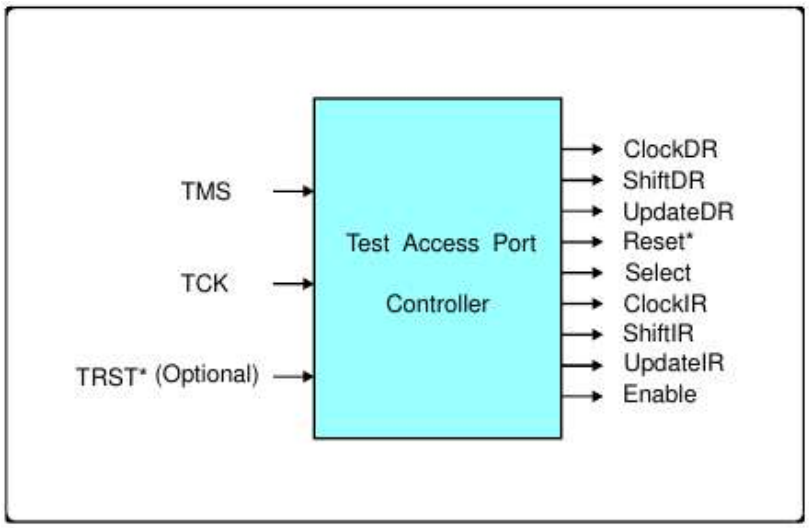
Instruções opcionais:

- INTEST: Testa a lógica interna do circuito.
- IDCODE: Realiza a leitura do código de identificação do dispositivo.
- RUNBIST: Inicia uma rotina interna de autoteste e coloca o registrador pass/fail result entre TDI e TDO.
- CLAMP: Direciona valores das células de varredura para as saídas do dispositivo, e então seleciona o registrador Bypass entre TDI e TDO.
- HIGHZ: Coloca todas as saídas em modo de alta impedância (high-Z), mas deixa o Bypass como registrador selecionado.

2.2 FUNCIONAMENTO DO CONTROLADOR TAP

O controlador TAP é uma máquina de estado finito implementada no CI, que possibilita o envio de sinais de controle para os registradores, atravé da interface JTAG. Os comandos enviados, alteram os estados do controlador, que seleciona o registrador que irá atuar. A Figura 5 mostra os sinais de entrada e os estados do controlador TAP.

Figura 5 – Visão Geral do Controlador TAP



Fonte: Bennetts (2022).

Os terminais obrigatórios são:

- Test Data In (TDI): Entrada serial de dados de teste, com valor padrão 1.

- Test Data Out (TDO): Saída serial de dados de teste, com valor padrão de Z e ativo somente durante uma operação de deslocamento.
- Test Mode Select (TMS): Sinal de controle de entrada serial, com valor padrão 1.
- Test Clock (TCK): Relógio de teste dedicado, qualquer valor conveniente.

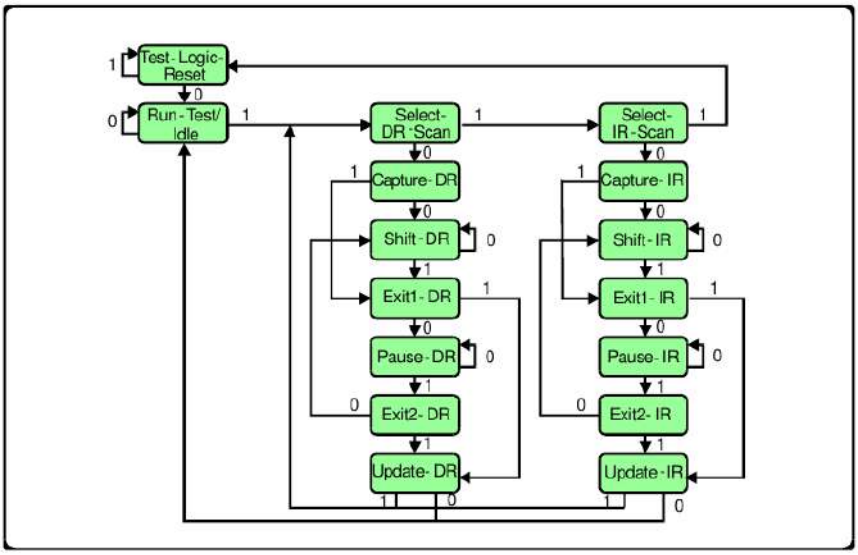
Terminal opcional:

- Test Reset (TRST): Reinicialização assíncrona do controlador TAP com valor padrão 1 e ativo em low.

TMS e TCK (e TRST) vão para o controlador de uma máquina de estados finitos, que produz os vários sinais de controle. Existem sinais dedicados ao IR (ClockIR, ShiftIR, UpdateIR) e sinais genéricos para todos os registradores de dados (ClockDR, ShiftDR, UpdateDR), e sinais genéricos Select, Reset e Enable. O registrador de dados que responde é aquele habilitado pelos sinais de controle condicional gerados nas saídas paralelas do IR, de acordo com a instrução específica.

A Figura 6 mostra a tabela de estados para o controlador TAP. O valor nos arcos de transição é o valor de TMS. Uma transição de estado ocorre na borda positiva de TCK e os valores mudam na borda negativa de TCK.

Figura 6 – Diagrama da tabela de estados do controlador TAP



Fonte: Bennetts (2022).

O controlador TAP inicializa no estado Test-Logic-Reset. Enquanto TMS permanece em 1, o estado permanece inalterado. Por o sinal de TMS em nível baixo causa uma transição para o estado Tun-Test/Idle.

Com uma sequência um-um o controlador passa para o estado Select-IR-Scan, podendo passar pelos demais estados conforme necessários. A última operação é Update-IR, onde a instrução carregada no Shift-IR é transferida para a seção hold do Registrador, e se torna a instrução atual. Com isso o registrador de dados identificado pela instrução atual substitui o IR como registrador de destino entre TDI e TDO. E com isso é possível manipular o registrador de dados de destino com as instruções genéricas.

Em geral, um controlador TAP requer quatro flip-flops para manter os valores de certos sinais de saída. O decodificador de próximo estado adicional e a lógica do decodificador de saída adicionam outros 20 a 40 gates.

2.3 FORMATOS DE DADOS RELACIONADOS

2.3.1 BOUNDARY-SCAN DESCRIPTION LANGUAGE (BSDL)

O Boundary Scan Description Language (BSDL) é um subconjunto do VHDL, que descreve como o IEEE 1149.1 é implementado no dispositivo e como ele opera. Um dos maiores usos do BSDL é a capacidade de desenvolvimento de ferramentas para automatizar os processos de testes baseados na IEEE 1149.1. Teradyne estima que criar testes padrões para micrprocessadores normalmente requer aproximadamente sete semanas. Enquanto que criar testes para microprocessadores com suporte a IEEE 1149.1 leva aproximadamente duas horas (Bennetts; Ley; Bales, 2022).

Os arquivos BSDL para dispositivos com varredura de limites são fornecidos pelos próprios distribuidores desses dispositivos. Uma descrição BSDL para um dispositivo consiste dos seguintes elementos:

- Declarações de entidades: essa seção inicia com uma “entity statement” e termina com um “end statement”.

- Parâmetros genéricos: É um parâetro que pode vir de fora da entidade ou se padrão na mesma, como um tipo de pacote de encapsulamento ou a descrição física de um pino.
- Descrição de portas lógicas: Dá nomes para os pinos I/O (pinos de sistema e TAP) e denota sua natureza, como input, output, bidirecional.
- Declarações de Uso: Se refere a definições externas encontradas nos pacotes.
- Mapeamento de Pinos: Fornece um mapeamento dos sinais lógicos para os pinos físicos de determinado dispositivo.
- Identificação da porta de comunicação: Define os estados dos pinos do TAP dos dispositivos.
- Descrição do Registrador de Instruções: Identifica as característica dependentes do dispositivo do Registrador de Instruções.
- Descrição do registrador de acesso: Define qual registrador será posto entre TDI e TDO para cada instrução.
- Descrição do Registrador de varredura: Contém uma lista das células de varredura de limites, junto com informações a respeito do tipo de célula e controle associado.

A seguir, são detalhados os aspectos metodológicos deste trabalho, que segue um processo sistemático para a configuração e utilização do OpenOCD em operações de depuração via JTAG. O fluxo inicia com a configuração do OpenOCD, criação dos arquivos de configuração e identificação dos pinos JTAG. Após inicializar o OpenOCD e estabelecer a comunicação Telnet, verifica-se a cadeia TAP com *scan_chain* e o arquivo BSDL. Por fim, executam-se comandos *irscan* e *drscan* para controle e teste da interface JTAG.

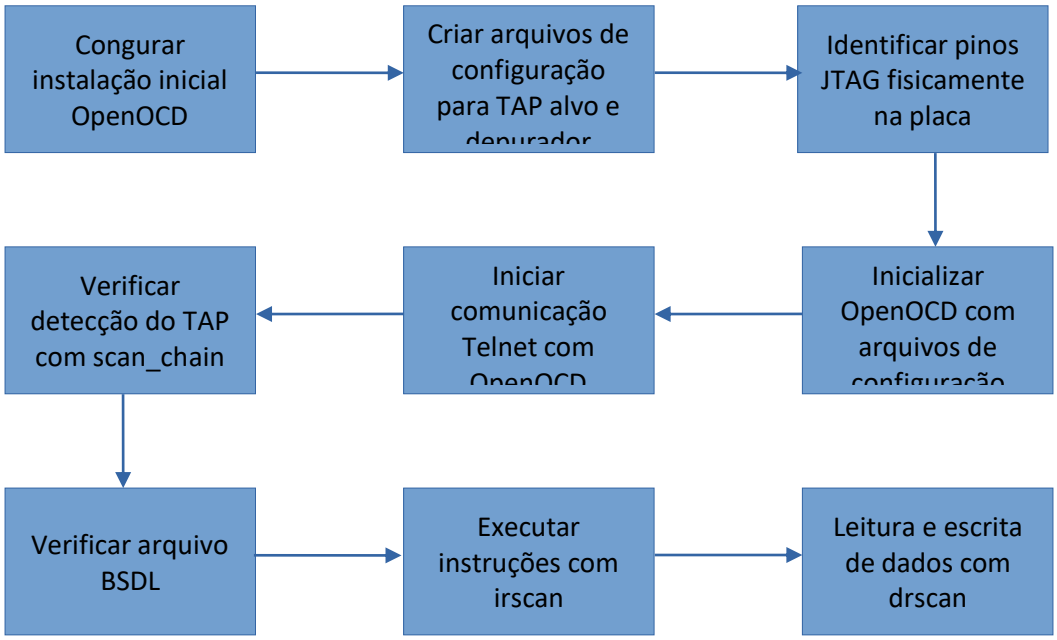
3 METODOLOGIA

Nesta seção será apresentada os aspectos metodológicos utilizados neste trabalho, com a finalidade de demonstrar o método de configuração do *software* OpenOCD para utilizar um depurador JTAG genérico e acessar o TAP de um CI,

possibilitando o acesso aos registradores e aos dados do Boundary Scan, e a utilização do arquivo BSDL do dispositivo para realizar o correto controle e interpretação do Boundary Scan. O fluxograma da Figura 7 ilustra os principais passos para se ter acesso a execução das instruções.

Para isso, foi utilizado uma placa responsável pela comunicação entre um equipamento ECG e um Tomógrafo, que possui um FPGA da Lattice Semi., com número de série LCMXO2-7000HC-5TG144C. Não existe nenhum propósito relacional na utilização específica da placa em questão, serve apenas a título de demonstração de que o padrão é implementado em diversos circuitos. A Figura 8 mostra a placa utilizada, com o depurador J-Link conectado aos terminais JTAG. Para comunicação com o TAP foi utilizado o depurador SEGGER J-link, conectado aos pinos TMS, TCK, TDO, TDI e GND do conector. A ordem dos pinos não costuma ser padrão em nenhuma aplicação, variando de acordo com o projeto da placa.

Figura 7 – Principais passos acessar os registradores.



Fonte: Elaborado pelo autor (2025)

Como o projeto da placa (Figura 8) não é fornecido aos usuários, e apesar do conector estar identificado como JTAG, foi necessário utilizar um multímetro em

modo de continuidade para identificar os pinos e a ordem correta, com o auxílio do datasheet do FPGA.

Figura 8 – Depurador j-Link e PCB utilizados.

Fonte: Elaborado pelo autor (2025).

Para que o OpenOCD consiga se comunicar tanto com o depurador quanto com o FPGA em questão, é necessário informar arquivos de configuração (.cfg) para eles. Existem arquivos de configuração para os depuradores com suporte, como é o caso do J-link, bastando alterar determinadas características.

Já para os CIs, existem arquivos genéricos para modelos mais comuns como o stm32, para o FPGA utilizado foi necessário criar um arquivo de configuração próprio para o TAP. Durante a instalação do OpenOCD, é necessário personalizar a instalação, para que ele forneça suporte ao adaptador J-Link através de comandos passados ao *script* de instalação.

Após iniciar o OpenOCD com as configurações corretas, é possível se comunicar com o servidor RPC através de uma porta TCP, permitindo executar comandos Tcl e receber os resultados, além de possibilitar a criação de scripts. A Figura 9 mostra o OpenOCD após inicializado e conectado ao servidor telnet.

No arquivo de configuração do depurador é necessário informar o protocolo de comunicação que será utilizado. Já no arquivo de configuração é necessário informar as características do TAP. Vale ressaltar que existem CI que possuem TAP individuais compartilhando o mesmo barramento, que precisam ser configurados separadamente e acessados por bits diferentes do registrador. Diferentes TAPs possuem diferentes terminações nos nomes, são elas: bs, cpu, etb, flash, jrc e tap.

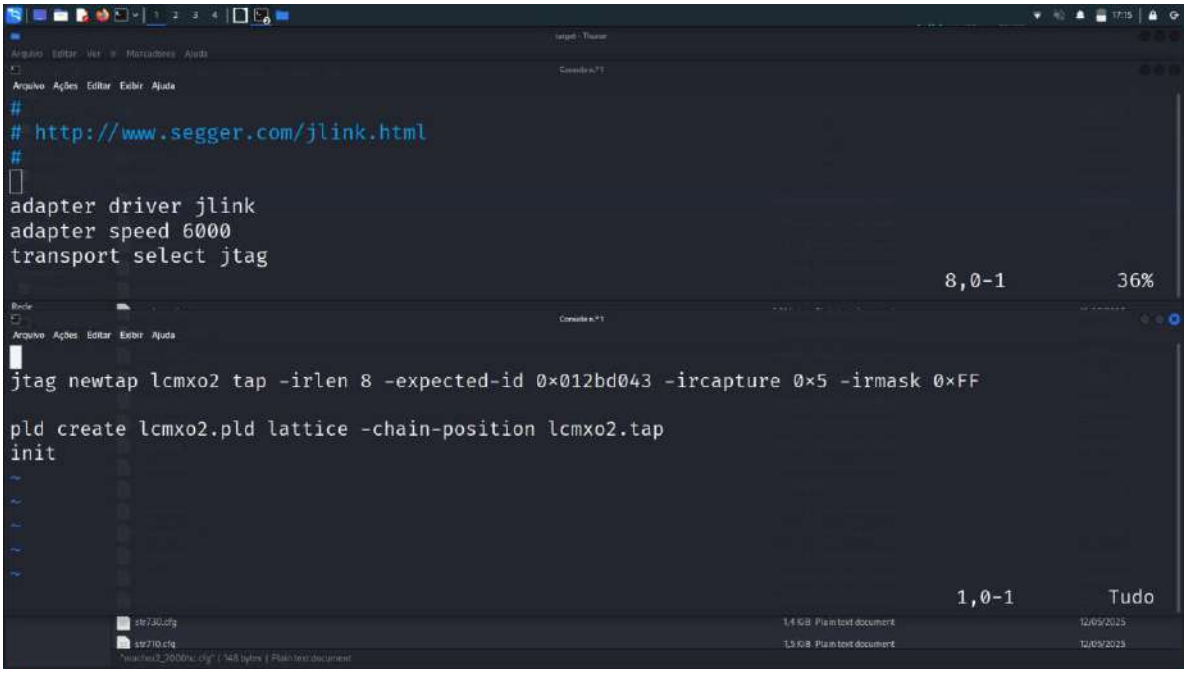
Assim, torna-se necessário verificar o datasheet para verificar a estrutura e quantidade do tap e o arquivo BSDL para verificar os bits relacionados a cada TAP.

Figura 9 – Estado do terminal após inicialização correta.

Fonte: Elaborado pelo autor (2025).

No caso de FPGA ou CPLD a terminação é tapname.tap, para este CI existe apenas um tap. A Figura 10 mostra a configuração do depurador e do CI.

Figura 10 – Configuração do depurador e TAP do FPGA.



Fonte: Elaborado pelo autor (2025).

Através do arquivo BSDL do FPGA, foi possível obter aos valores referentes as instruções suportadas, as funcionalidades das células Boundary Scan, e realizar a interpretação dos dados recebidos em resposta aos comandos. A placa em questão também, também possui LED's de status, os quais foi possível controlar através do Boundary Scan, verificando fisicamente o funcionamento. O Quadro 1 apresenta uma relação com as instruções executadas, os respectivos registradores acessados, e a funcionalidade.

Quadro 1 – Instruções executadas com respectivos registradores de acesso.

Instrução	Funcionalidade	Registro acessado
IDECODE	Leitura Idcode	Idecode
BYPASS	Bypass registro Boundary Scan. I/O com funções normais	Bypass
CLAMP	Trava outputs em valores preload. Inputs normais.	Bypass
HIGHZ	Impede sinais inputs e outputs.	Bypass
SAMPLE/PRELOAD	Sample, leitura Boundary Scan. Preload carrega dados nos bits Boundary Scan.	Boundary Scan
EXTEST	Envia os dados Preload para as células Boundary Scan Dinamicamente.	Boundary Scan

Fonte: Elaborado pelo autor (2025).

4 APRESENTAÇÃO E DISCUSSÃO DOS RESULTADOS

De acordo com o arquivo BSDL há suporte para as seguintes instruções, com seus respectivos bits de acionamento: EXTEST(0x15), SAMPLE/PRELOAD(0x1C), BYPASS(0xFF), IDCODE(0xE0), HIGHZ(0x18), CLAMP(0x78). Ainda, é importante salientar que tanto o BSDL quanto as respostas do Boundary Scan fornecem os

valores das instruções e dados como binário, mas para ser fornecido como válido ao OpenOCD é necessário que estejam em Hexadecimal. As instruções são passadas para o registrador de instruções da seguinte forma:

irscan <tap> <instruction>

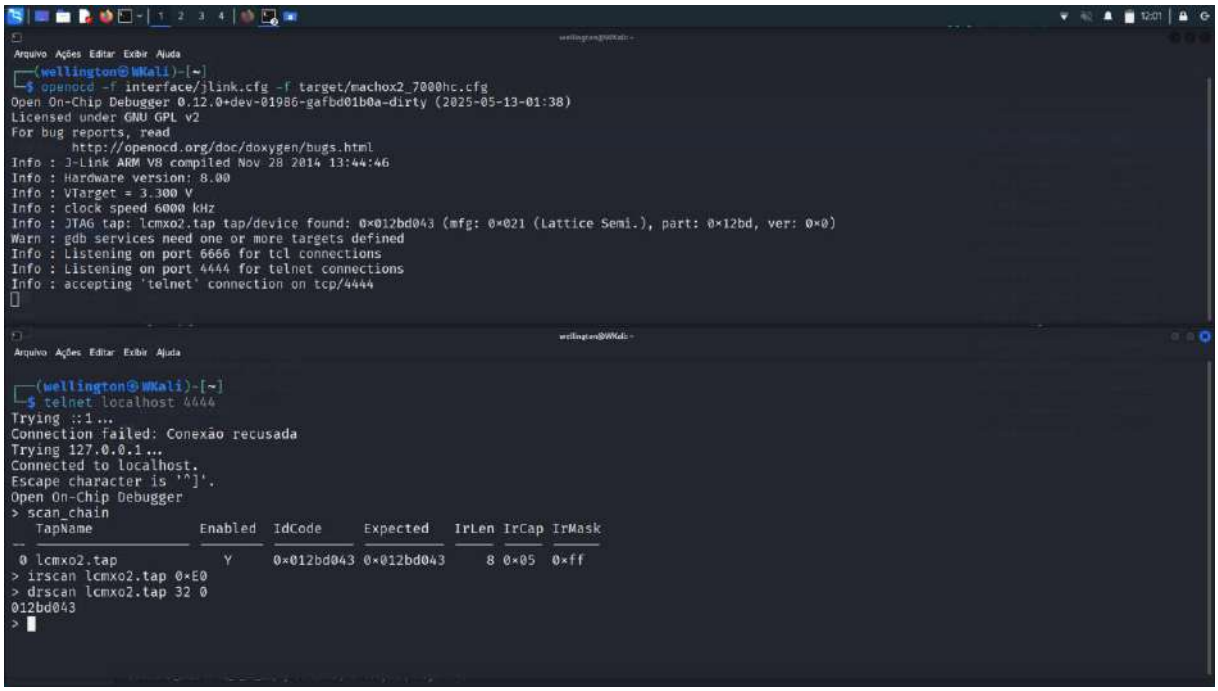
Para ler ou escrever dados no registrador de dados, é utilizado o comando:

drscan <tap> <length DR> <value>

No campo length DR é colocado o tamanho do registro Boundary Scan, nesse caso 664 bits, já o campo value é colocado o valor que se deseja enviar ao Boundary Scan, e dever ser do mesmo tamanho do registro, é aceitavel valores hexadecimal(0x) e binário(0b), para realizar a leitura é enviado apenas o valor 0.

A instrução IDCODE retorna o ID do fabricante implementado no chip, que também pode ser encontrado no BSDL, além de ser útil para identificar possíveis falsificações também serve como um teste primário para a comunicação com o TAP. A Figura 11 mostra a implementação do mesmo.

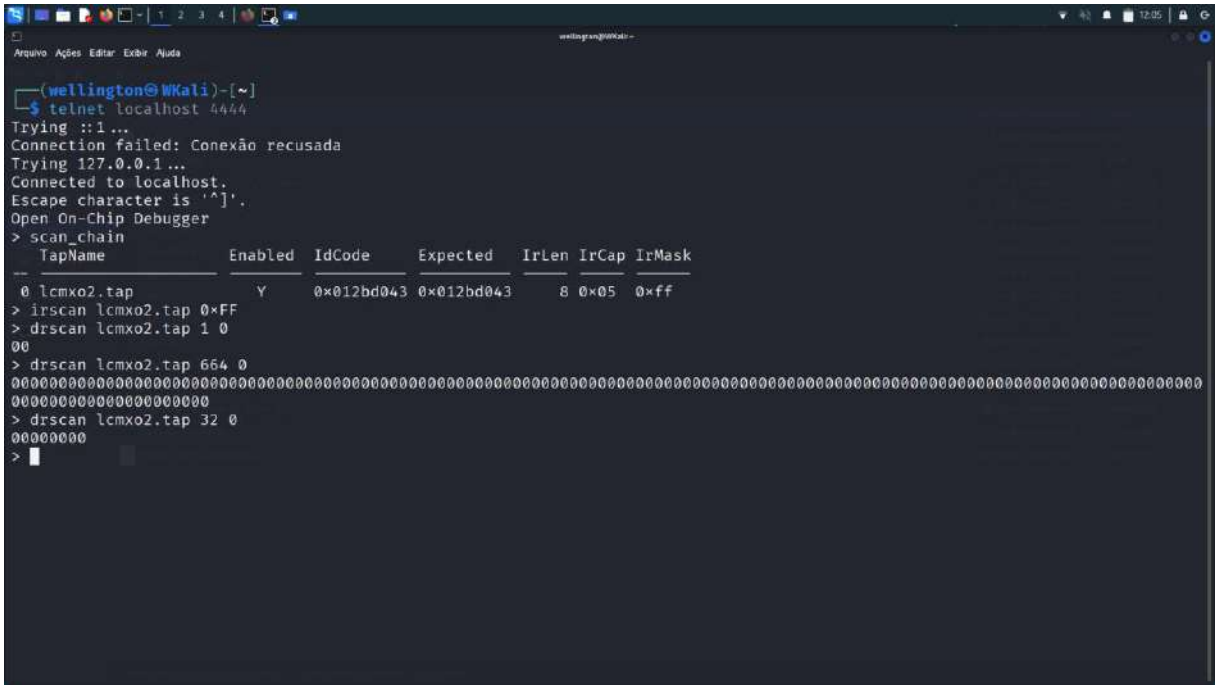
Figura 11 – Execução e leitura da instrução IDCODE.



Fonte: Elaborado pelo autor (2025).

A instrução BYPASS seleciona o registrador de bypass, que possui 1 bit. Essa instrução é útil quando se testa cadeias JTAG com mais de um CI, limitando o número de bits a ser lido o registrador de dados, caso não se deseje ler os dados de um ou outro CI, além de aumentar o tempo de resposta já que o registro é serial. No caso da placa em questão, não teve muita utilidade funcional. A Figura 12 mostra a execução dessa instrução. Tentativas de ler o registro Boundary Scan, retornam um padrão de 0x0. Após realizar o reset da interface, o registro Boundary Scan não está mais disponivel para leitura.

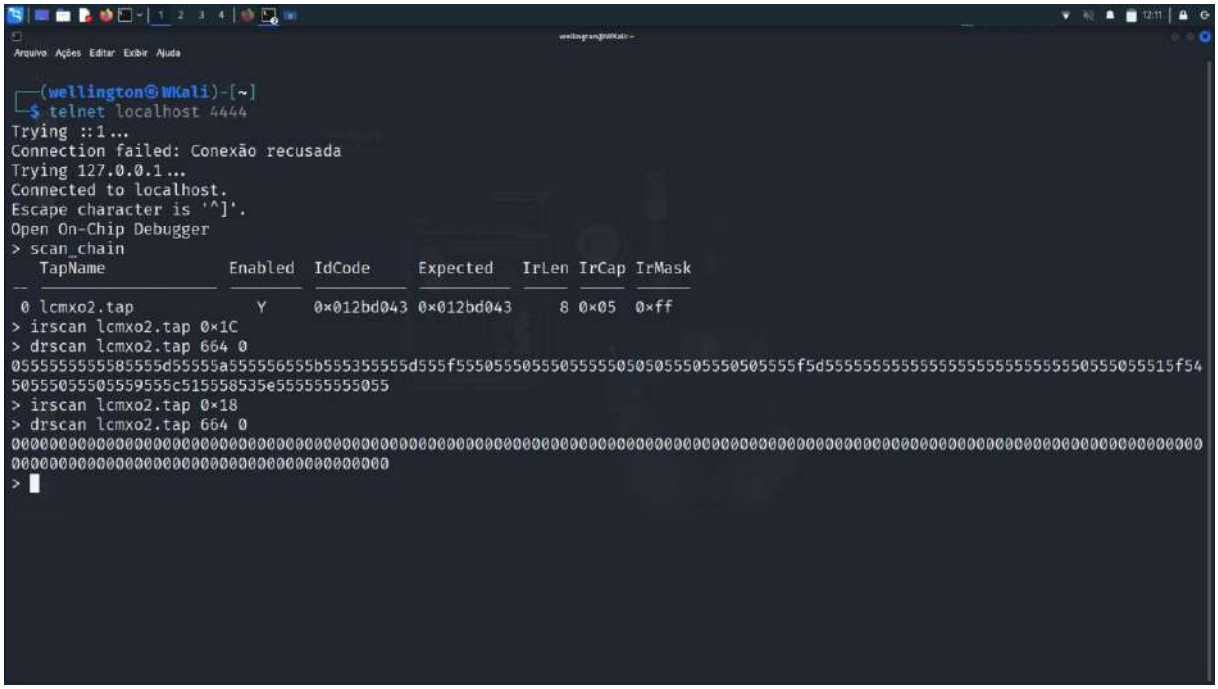
Figura 12 – Execução da instrução bypass.



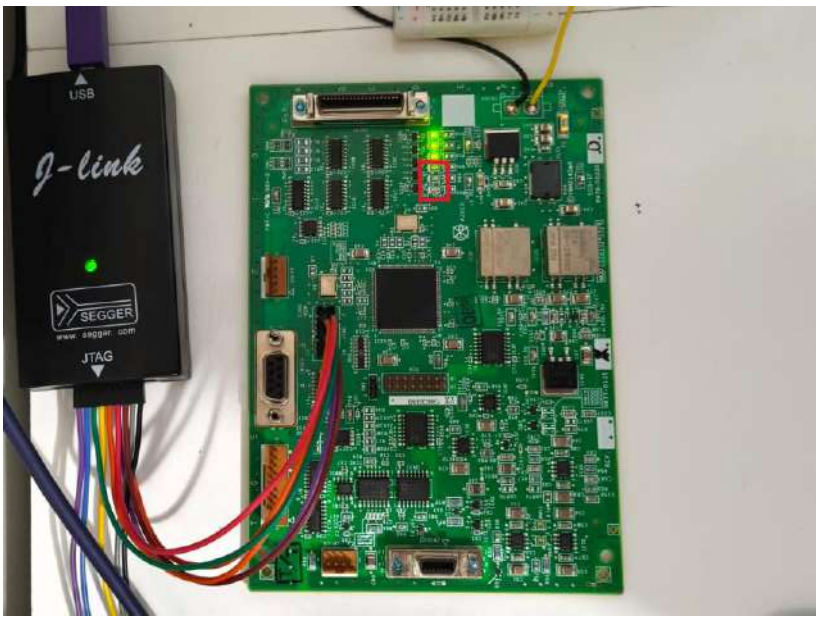
Fonte: Elaborado pelo autor (2025).

A instrução HIGHZ é um método de segurança em testes com múltiplos CIs, onde os pinos de um CI podem ser colocados em estado de alta impedância, para que o mesmo não seja afetado por outros CI da cadeia, ou mesmo dispositivos não-JTAG em teste. A Figura 13(a) ilustra a execução da instrução, em que é possível verificar que a instrução causa alterações físicas na placa, pela alteração do LEDs de status da mesma, conforme observado na Figura 13(b).

Figura 13 – Instrução HIGHZ: (a) Execução da instrução highz; (b) Alteração dos LEDs de status após execução de highz.



(a)

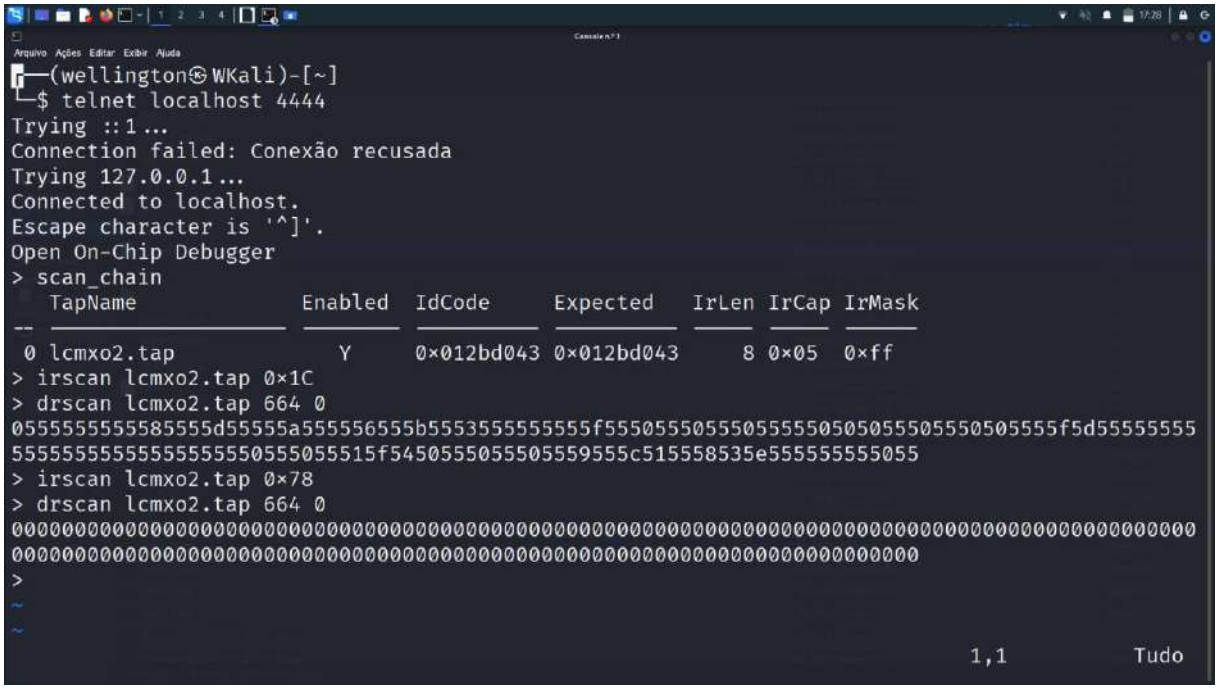


(b)

Fonte: Elaborado pelo autor (2025).

A instrução CLAMP captura o estado do registro Boundary Scan e faz com que os pinos de saída sejam controlados pelo registro, isolando os pinos de saída da lógica interna e os mantendo com os valores capturados, enquanto os pinos de entrada continuam controlados pela lógica interna e passíveis de ser alterados por sinais externos. Útil para testar o comportamento do CI em relação a sinais externos e como os sinais de saída do CI afetam demais periféricos conectados a placa.

Figura 14 – Execução da instrução CLAMP.



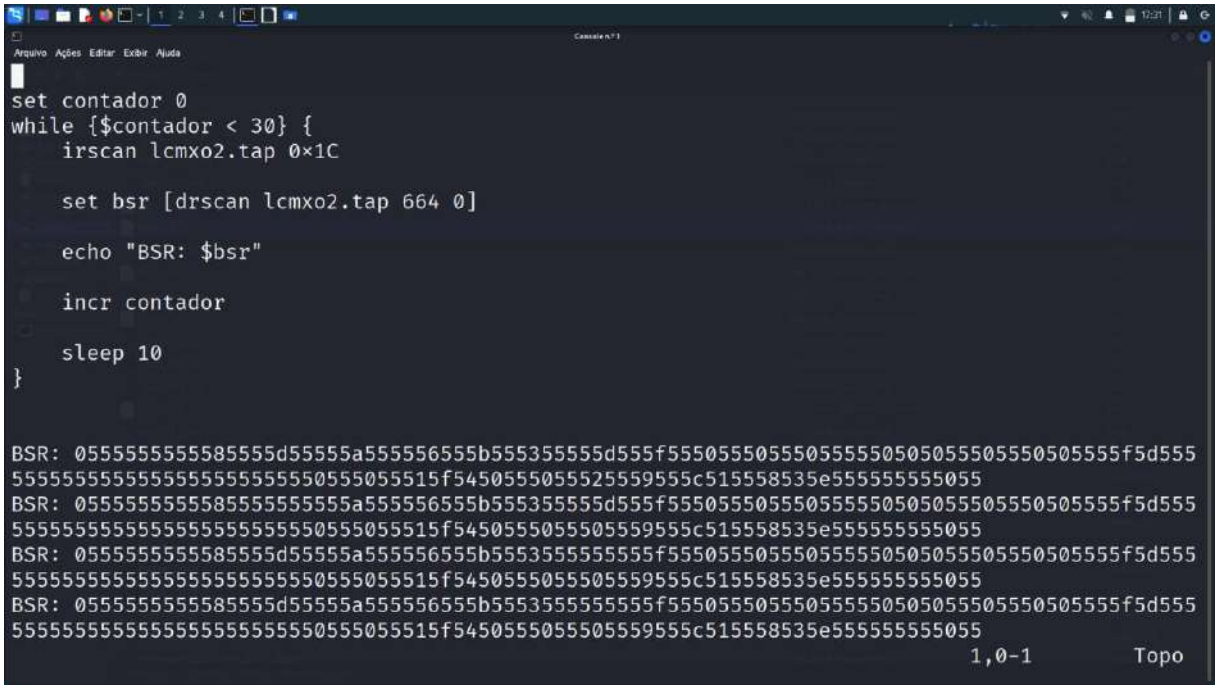
Fonte: Elaborado pelo autor (2025).

As instruções CLAMP, HIGHZ e BYPASS, selecionam o registrador de bypass, ou seja, não é possível ler ou escrever no registro boundary scan ao executar essas instruções.

A instrução SAMPLE/PRELOAD possui duas funções, SAMPLE captura os dados atuais do Boundary Scan, e PRELOAD pré-carrega dados no Boundary Scan, esses dados são aplicados ao Boundary Scan após executar a função CLAMP ou EXTEST. Após carregar essa instrução no registrador, as funções de carregamento e captura pode ser controlados com “drscan”. A Figura 15 mostra a execução dessa instrução imda em um loop while, o que permite capturar uma sequência contínua de

dados de cada pino. É possível comparar as sequências de bits com assinaturas de protocolos de comunicação, sinais de clock, etc.

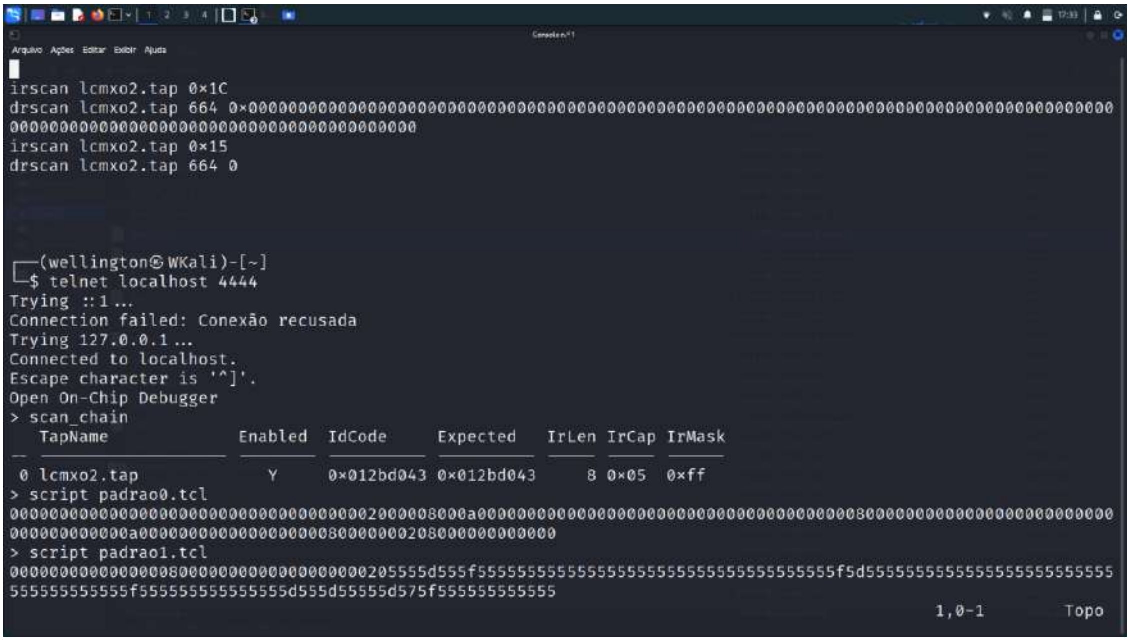
Figura 15 – Captura de dados do BSR utilizando a instrução SAMPLE com laço while.



Fonte: Elaborado pelo autor (2025).

A instrução EXTEST é utilizada em conjunto com a função SAMPLE/PRELOAD, assim como CLAMP, mas com a diferença de que além de carregar os pinos de saída com valores especificos, faz o monitoramento dos pinos de entrada, e diferente da instrução CLAMP que congela as saídas em estados especificos, com EXTEST é possível realizar a atualização continua do registro Boundary Scan sem ter de retornar para a instrução preload. Essa instrução é mais utilizada para verificar curto-circuto entre pinos do CI, integridade de sinais entre CIs, ou mesmo acionar circuitos passivos. A Figura 16(a) mostra a execução da instrução EXTEST precedida da instrução PRELOAD. A Figura 15(b) mostra alterações nos LEDs de status da placa.

Figura 16 – Instrução EXTEST: (a) Utilização da instrução PRELOAD e extest; (b) Alteração nos LEDs após execução das instruções.



(a)

(b)

Fonte: Elaborado pelo autor (2025).

A metodologia adotada neste artigo demonstrou a viabilidade de utilizar ferramentas open-source, como o OpenOCD, em conjunto com um depurador JTAG genérico (J-Link), para acessar e controlar o TAP de um FPGA. A configuração personalizada dos arquivos .cfg, a identificação correta dos pinos JTAG, o estudo do arquivo BSDL e da arquitetura do CI alvo, são essenciais para se ter sucesso com o uso do OpenOCD.

Além disso, a execução prática de instruções como SAMPLE, HIGHZ e EXTEST, aliada à verificação física dos resultados nos LEDs da placa e as saídas de dados no terminal, validou a eficácia do método proposto. Essa estratégia não apenas comprova as possibilidades do padrão IEEE 1149.1 em cenários de reparo e diagnóstico em serviços não-OEM (Original Equipament Manufacturer), mas também destacou a importância de soluções acessíveis e flexíveis para a engenharia de sistemas embarcados.

5 CONSIDERAÇÕES FINAIS

Este estudo demonstrou a viabilidade e eficácia do padrão IEEE 1149.1 (JTAG) como uma ferramenta essencial para o teste, diagnóstico e reparo de placas de circuito impresso complexas, especialmente em cenários onde o acesso a diagramas esquemáticos ou programação interna é limitado. A utilização de ferramentas open-source, como o OpenOCD, em conjunto com depuradores JTAG genéricos, como o J-Link, mostrou-se uma alternativa acessível e poderosa para acessar e controlar os registradores de Boundary Scan em dispositivos que oferecem suporte ao padrão.

Os resultados obtidos comprovaram a capacidade do padrão JTAG para realizar tarefas como controle de pinos, simulação de sinais, captura e análise de dados de barramentos. A execução prática de instruções suportadas pelo CI, aliada à análise do arquivo BSDL, destacou a importância de uma configuração precisa e do entendimento da arquitetura do dispositivo para o sucesso da metodologia.

Além disso, o estudo evidenciou o potencial dessa abordagem para ampliar as possibilidades de manutenção em sistemas embarcados críticos, reduzindo

custos e dependência de ferramentas proprietárias. A flexibilidade e o baixo custo das soluções apresentadas tornam-nas particularmente relevantes para cenários onde recursos são limitados ou para serviços de reparo não-OEM.

Embora o método proposto demonstre potencial no reparo de placas de circuito impresso, é importante ressaltar suas limitações: i) escopo restrito: os testes foram realizados em um único modelo de FPGA, não abrangendo microcontroladores, CPLDs ou sistemas com múltiplos CIs, onde a implementação do padrão e as funcionalidades das intruções variam; ii) Foco específico: o trabalho priorizou o acesso a registradores do padrão IEEE 1149.1, sem explorar técnicas complementares (ex.: uso de osciloscópios e analisadores lógicos), que aumentariam a eficácia do diagnóstico.

Essas limitações, no entanto, abrem oportunidades para pesquisas futuras. Como sugestões para trabalhos futuros, recomenda-se a exploração de técnicas automatizadas para interpretação e captura de dados do Boundary Scan, o desenvolvimento de interfaces mais amigáveis para ferramentas open-source e a aplicação do método em outros dispositivos, como microcontroladores e CPLDs.

Por fim, este trabalho reforça a importância do padrão IEEE 1149.1 como um aliado indispensável para a engenharia de manutenção e o desenvolvimento de sistemas embarcados, destacando seu papel na promoção de soluções sustentáveis e eficientes para os desafios tecnológicos atuais.

REFERÊNCIAS

ARM Limited. Arm Debug Interface: Architecture Specification. 2022. Ebook. Disponível em: https://developer.arm.com. Acesso em: 04 Mai. 2025.

BENNETTS, Ben; LEY, Adam; BALES, Ben. IEEE 1149.1 JTAG and Boundary Scan Tutorial.2ª Edição. Ebook. Disponível em: https://www.asset-intertech.com. Acesso em: 15 Mar. 2025.

BENNETT, Steve. jim Tcl reference manual. V0.83. Disponível em: https://jim.tcl-lang.org.

LATTICE Semi. LCMXO2_7000HC BSDL Model. V1.04. 2022. Disponível em: BSDL_Model.

FITZPATRICK, Joe. Real Hardware Hacking for $30 or Less. Infosec In the City. 2019. Disponivel em: https://infocon.org. Acesso em: 12 Mar. 2025.

INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. *IEEE Standard Test Access Port and Boundary-Scan Architecture.* IEEE Std 1149.1-2013 (Revision of IEEE Std 1149.1-2001). New York: IEEE, 2013. Disponível em: https://standards.ieee.org. Acesso em: 15 abr. 2025.

LATTICE Semi. MachXO2 Family Data Sheet 2025. 126p. Disponível em:https://www.latticesemi.com.

MAZIDI, Muhammad Ali; MAZIDI, Janice Gillispie; MCKINLAY, Rolin D. *Programação em linguagem C: para microcontroladores da família 8051.* São Paulo: Pearson, 2007.

MAUNDER, Colin M.; TULLOSS, Rodham E. The Test Access Port And Boundary-Scan Architecture. 1ª Edição. Los Alamitos, California: IEEE Computer Society Press, 1990.

PARKER, Kenneth P. The Boundary-Scan Handbook. 3ª Edição. Boston, Massachusetts: Springer Press, 2003.

REMPEL, Oleksij. OpenOCD – Beyond Simple Software Debugging. Embedded Linux Conference + OpenIOTSummit. 2018. Disponivel em: https://osseu18.sched.com. Acesso em: 10 Mar. 2025.

RATH, Dominic. Design and Implementation of an On-Chip Debug Solution for Embedded Target Systems based on the ARM7 and ARM9 Family. 133f. Tese. University of Applied Sciences Augsburg, Augsburg, 2005. Disponivel em: http://openocd.org/files/thesis.pdf. Acesso em: 30 Mar. 2025.

SANTOS, Willian. Repair, Don’t Wast: The Solution to Our E-Waste Problem. Disponível em: https://maintenanceworld.com. Acesso em: 31 Mai. 2025.

The OpenOCD Project. Open On-Chip Debugger: OpenOCD User’s Guide. V0.10.0. 2018. 173p. Disponível em: https://openocd.org/pages/documentation.html.