UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO DE CIÊNCIAS, TECNOLOGIAS E SAÚDE DO CAMPUS ARARANGUÁ
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

Leomar Marcelo Marschalk

Desenvolvimento da camada zero do padrão IEEE 1876-2019 para um laboratório remoto de FPGA

Araranguá
2024

Leomar Marcelo Marschalk

Desenvolvimento da camada zero do padrão IEEE 1876-2019 para um laboratório remoto de FPGA

Trabalho de Conclusão de Curso do Curso de Graduação em Engenharia de Computação submetido ao Centro de Ciências, Tecnologias e Saúde do Campus Araranguá da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Engenharia de Computação.
Orientador: Prof. Jim Lau, Dr.

Araranguá
2024

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Marschalk, Leomar Marcelo
Desenvolvimento da camada zero do padrão IEEE 1876-2019 para um laboratório remoto de FPGA. / Leomar Marcelo Marschalk ; orientador, Jim Lau, 2024.
27 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Araranguá, Graduação em Engenharia de Computação, Araranguá, 2024.

Inclui referências.

1. Engenharia de Computação. 2. laboratório remoto. 3. FPGA. 4. aprendizado a distância. I. Lau, Jim. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Computação. III. Título.

Leomar Marcelo Marschalk

Desenvolvimento da camada zero do padrão IEEE 1876-2019 para um laboratório remoto de FPGA

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de Bacharel em Engenharia de Computação e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Computação.

Araranguá, 29 de Novembro de 2024.

Documento assinado digitalmente
Jim Lau
Data: 10/12/2024 14:50:28-0300
CPF: ***.464.702-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Jim Lau, Dr.
Coordenador do Curso

Banca Examinadora:

Documento assinado digitalmente
Jim Lau
Data: 10/12/2024 14:50:54-0300
CPF: ***.464.702-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Jim Lau, Dr.
Orientador

Documento assinado digitalmente
ANALUCIA SCHIAFFINO MORALES
Data: 10/12/2024 14:58:27-0300
CPF: ***.256.420-**
Verifique as assinaturas em https://v.ufsc.br

Prof[a]. Analúcia Schiaffino Morales, Dr[a].
Avaliadora
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Fabricio de Oliveira Ourique
Data: 11/12/2024 06:42:05-0300
CPF: ***.167.860-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Fabrício de Oliveira Ourique, Dr.
Avaliador
Universidade Federal de Santa Catarina

Desenvolvimento da camada zero do padrão IEEE 1876-2019 para um laboratório remoto de FPGA

Leomar Marcelo Marschalk* Jim Lau†

2024, Novembro

Resumo

A programação de hardware digital é essencial para o desenvolvimento de circuitos complexos, possibilitando a criação e simulação de sistemas com alta precisão e confiabilidade. As Linguagens de Descrição de Hardware (LDHs) desempenham um papel fundamental nesse processo, permitindo a modelagem, simulação e otimização de circuitos antes de sua implementação física. No entanto, as restrições orçamentárias em universidades brasileiras dificultam o acesso a kits de desenvolvimento de FPGA, essenciais para o aprendizado de LDHs, comprometendo o aprendizado prático. Para solucionar essa questão, foi proposto o desenvolvimento de um laboratório remoto de FPGA, em colaboração entre a Universidade Federal de Santa Catarina (UFSC) e o Laboratory at Distance (L@D) da Université TÉLUQ, no Canadá. Este trabalho desenvolve uma camada de hardware deste laboratório remoto, composta pelo kit de desenvolvimento FPGA e um computador servidor. O desenvolvimento da camada de hardware consiste na implementação de um meio de comunicação entre o kit de desenvolvimento FPGA e o servidor, permitindo a interação remota com o FPGA por meio de comandos enviados ao servidor. A comunicação entre o kit de desenvolvimento FPGA e o servidor ocorre por meio de um conversor USB para UART. Para viabilizar a comunicação, foi desenvolvido um circuito digital capaz de se comunicar por UART, recebendo sinais de botões e chaves seletoras virtuais do servidor e enviando ao servidor sinais de LEDs e displays de sete segmentos. Para auxiliar os testes, também foi criado um laboratório remoto minimalista, utilizando a camada de hardware desenvolvida. Os resultados mostraram uma boa usabilidade e a capacidade da camada de hardware de se recuperar rapidamente de erros de transmissão. Assim, o trabalho implementou com sucesso uma camada de hardware, tornando funcional uma parte crucial de qualquer laboratório remoto para FPGA.

Palavras-chave: laboratório remoto, FPGA, aprendizado à distância

*leomarmm1@gmail.com
†jim.lau@ufsc.br

Layer zero development of the IEEE 1876-2019 standard for a FPGA remote laboratory

Leomar Marcelo Marschalk* Jim Lau†

2024, Novembro

Abstract

Digital hardware programming is essential for the development of complex circuits, enabling the creation and simulation of systems with high precision and reliability. Hardware Description Languages (HDLs) play a fundamental role in this process, allowing the modeling, simulation, and optimization of circuits before their physical implementation. However, budget constraints in Brazilian universities hinder access to FPGA development kits, which are essential for learning HDLs, thereby compromising practical learning. To address this issue, the development of a remote FPGA laboratory was proposed, in collaboration between the Federal University of Santa Catarina (UFSC) and the Laboratory at Distance (L@D) of Université TÉLUQ in Canada. This work develops a hardware layer for this remote laboratory, consisting of the FPGA development kit and a server computer. The development of the hardware layer involves implementing a communication medium between the FPGA development kit and the server, allowing remote interaction with the FPGA through commands sent to the server. Communication between the FPGA development kit and the server is established via a USB-to-UART converter. To enable communication, a digital circuit was developed capable of communicating via UART, receiving signals from virtual buttons and switches from the server and sending signals to the server from LEDs and seven-segment displays. To support testing, a minimalist remote laboratory was also created using the developed hardware layer. The results demonstrated good usability and the hardware layer's ability to quickly recover from transmission errors. Thus, this work successfully implemented a hardware layer, making a crucial part of any remote FPGA laboratory functional.

Keywords: Remote laboratory, FPGA, e-learning

*leomarmm1@gmail.com
†jim.lau@ufsc.br

Sumário

1 Introdução

A programação aplicada a componentes de hardware digital é uma abordagem essencial para o desenvolvimento de circuitos complexos. Essa técnica possibilita a criação e simulação de sistemas com elevada precisão, permitindo projetar soluções inovadoras e eficientes. Por meio dessa metodologia, possibilita-se realizar testes e otimizações antes da implementação física, assegurando maior confiabilidade, flexibilidade e precisão no design de hardware, alinhando-o às demandas tecnológicas contemporâneas.

As Linguagens de Descrição de Hardware (LDHs) desempenham um papel importante nesse processo de modelagem e desenvolvimento de circuitos eletrônicos, especialmente na área de lógica digital. Essas linguagens permitem que os engenheiros detalhem o funcionamento, o projeto e a interconexões de sistemas eletrônicos de maneira precisa e organizada. Além disso, LDHs oferecem suporte para simulações robustas, que permitem avaliar o comportamento funcional do hardware em ambientes virtuais antes da etapa de fabricação. Essa capacidade de simulação reduz significativamente os riscos e os custos associados ao desenvolvimento de hardware digital.

No entanto, a aquisição de kits de desenvolvimento didáticos de FPGA, essenciais para aulas práticas, enfrenta alguns desafios nas universidades brasileiras devido a restrições orçamentárias. Conforme apontam (REIS; MACARIO, 2020), os recursos de custeio das universidades federais diminuíram significativamente de 27,97% entre 2013 a 2019. Isso teve um impacto direto na capacidade das instituições de manter e comprar novos equipamentos. Além disso, os investimentos diminuíram ainda mais acentuadamente, de 95,29% entre 2011 a 2019 devido à diminuição da verba destinada a melhorias e inovações tecnológicas nas universidades.

Nesse contexto, surge uma solução inovadora: os laboratórios remotos de hardware. Esses ambientes permitem a realização de experimentos didáticos por meio da internet (ORDUÑA et al., 2015). Embora sejam laboratórios físicos, podem ser controlado e monitorado remotamente (GARCIA-LORO; CRISTOBAL; CASTRO, 2019), proporcionando uma experiência de aprendizado prática e flexível. Integrados a sistemas de gestão de aprendizagem, como o Moodle, esses laboratórios facilitam o acesso e a interação dos alunos, promovendo uma educação de qualidade a distância, que complementa e potencializa a formação teórica oferecida pelas LDHs.

Por meio da colaboração entre o professor Marcelo Berejuck, da Universidade Federal de Santa Catarina (UFSC/Brasil), e o Laboratory at Distance (L@D) da Université TÉLUQ (Canadá), foi proposto o desenvolvimento de um laboratório remoto de FPGA conforme (BEREJUCK et al., 2022). Este trabalho desenvolve a camada de hardware deste laboratório remoto, composta pelo kit de desenvolvimento FPGA e um computador servidor. O desenvolvimento da camada de hardware consiste na implementação de um meio de comunicação entre o kit de desenvolvimento FPGA e o servidor, permitindo a interação remota com o FPGA por meio de comandos enviados ao servidor.

Este trabalho está organizado em sete seções. A Seção 2 apresenta uma revisão da literatura sobre laboratórios remotos de FPGA, oferecendo uma visão geral das pesquisas anteriores na área. A Seção 3 apresenta alguns conceitos fundamentais utilizados na elaboração deste trabalho. A Seção 4 detalha a abordagem metodológica utilizada. A Seção 5 descreve a implementação prática da abordagem proposta. A Seção 6 apresenta e discute os resultados obtidos. Por fim, a Seção 7 apresenta as conclusões.

2 Trabalhos Relacionados

Na literatura científica foram encontrados poucos trabalhos dedicados exclusivamente a laboratórios remotos utilizando FPGAs. As buscas foram realizadas nas plataformas *IEEE Xplore*, CAPES e *Google Scholar*, utilizando a combinação de palavras-chave "*remote lab*, "FPGA"e *"e-learning"*, filtrando pelo período de 2018 até 2022. Nos poucos artigos encontrados sobre laboratórios remotos de FPGA, priorizou-se aqueles mais alinhados com a proposta de um laboratório remoto de FPGA didático.

Em (OBALLE-PEINADO et al., 2020), apresentam um laboratório remoto de FPGA voltado para o ensino à distância, utilizando a placa Digilent Inc. Nexys 3. O sistema permite que os usuários configurem e interajam com o FPGA por meio de uma aplicação web, sendo necessário cadastro para acesso, o que permite o levantamento de estatísticas de uso e a limitação do tempo de acesso, garantindo o uso compartilhado. O laboratório é versátil, permitindo a escolha de periféricos emulados, como teclados hexadecimais, para fornecer sinais de entrada ao FPGA. A arquitetura adota um modelo cliente-servidor, com comunicação via HTTP, e utiliza um Arduino Uno no servidor para simular os periféricos acessados pelos alunos, reduzindo a complexidade do código no FPGA, mas gerando custos adicionais de hardware. Diferentemente, o presente trabalho realiza a simulação dos periféricos diretamente no FPGA, conforme detalhado na Seção 4.2.2, o que reduz os custos e mantém a eficiência do sistema.

De acordo com (MOHSEN; GADALRAB et al., 2019), o estudo se concentra na implementação das regras de negócios do laboratório remoto de FPGA no lado servidor, sem detalhar a implementação no lado cliente. O servidor utiliza a distribuição Linux "Debian 8 Jessie" e implementa duas funcionalidades principais: o controle de acesso de usuários e comunicação com o FPGA. O controle de acesso é baseado no sistema de contas do Linux, onde cada usuário registrado possui uma conta específica que é bloqueada para prevenir acessos não autorizados. A comunicação entre o servidor e o FPGA é realizada por meio de mecanismos de depuração própria da Xilinx, utilizando uma conexão USB JTAG. O laboratório requer reserva de horário, liberando a conta do usuário e o acesso à porta USB do FPGA no momento agendado, gerenciado pelo sistema de permissões do Linux. No entanto, a utilização de recursos proprietários da Xilinx limita a compatibilidade do projeto com FPGAs de outros fornecedores. Em nosso projeto, buscamos uma alternativa de comunicação servidor-FPGA compatível com diversos FPGAs de diferentes fabricantes.

Em (DE MEDEIROS, 2018), foi implementado um laboratório remoto didático para FPGAs, voltado para as disciplinas de Prática de Eletrônica Digital na Universidade de Brasília, utilizando o kit de desenvolvimento Basys 3. O sistema é baseado no modelo cliente-servidor, permitindo o acesso ao laboratório por meio de um website, onde o usuário pode enviar seu projeto e interagir com o FPGA remotamente, utilizando chaves seletoras e botões virtuais, enquanto acompanha o kit de desenvolvimento por meio de uma transmissão de vídeo ao vivo. No lado servidor, a comunicação com o FPGA é realizada via UART, através da qual os estados dos periféricos (botões e chaves) são enviados. No entanto, foi identificada uma limitação no trabalho relacionada à ausência de mecanismos de detecção e correção de erros de transmissão. Para superar essa limitação, nosso trabalho introduziu mecanismos de controle por CRC e temporizadores watchdog, assegurando a robustez e confiabilidade do sistema. Os resultados obtidos demonstram a eficácia da implementação, tornando o laboratório remoto mais eficiente e eficaz para os usuários.

Diante das limitações observadas nos trabalhos existentes, este trabalho propõe uma solução que melhora a experiência em laboratórios remotos para FPGAs, priorizando

acessibilidade, flexibilidade e custo reduzido na implementação. A próxima seção irá detalhar nossa proposta, que propõe uma solução compatível com diversos fabricantes e famílias de FPGAs, além de incluir mecanismos de controle e confiabilidade, como CRC e temporizadores watchdog, para proporcionar maior confiabilidade no uso à distância.

3 Fundamentação Teórica

3.1 FPGA

Um *Field Programmable Gate Array* (FPGA) é um dispositivo eletrônico reconfigurável e versátil, amplamente utilizado em eletrônica digital e sistemas embarcados. Diferente dos circuitos integrados tradicionais, um FPGA consiste em uma matriz de blocos lógicos programáveis que podem ser configurados para executar diversas funções específicas, permitindo operações lógicas complexas e comunicação eficiente entre os blocos. Sua flexibilidade torna-o ideal para prototipagem rápida e desenvolvimento de sistemas complexos, com a capacidade de reconfiguração, permitindo atualizações e modificações sem a necessidade de trocar o hardware. FPGAs são empregados em áreas como processamento de sinais, telecomunicações, aeroespacial e automação industrial, oferecendo alto desempenho e baixa latência, e possibilitam a integração de sistemas completos em um único chip, combinando processadores, memória e periféricos.

Segundo (CHU, 2008), existem várias maneiras de implementar um bloco lógico, sendo mais comum a utilização de *Look-up Tables (LUTs)*. LUTs são pequenos blocos de memória que geralmente quatro ou cinco entradas, que endereçam 16 ou 32 células de armazenamento, cada uma contendo um valor lógico, zero ou um. Cada bloco lógico em um FPGA pode ser configurado para realizar funções simples, enquanto as chaves programáveis permitem personalizar as interconexões entre as células, possibilitando a criação de circuitos digitais personalizados e adaptados às necessidades específicas do projeto. Além disso, muitos FPGAs possuem células de propósito específico, como blocos de memória e multiplicadores, que ampliam ainda mais suas capacidades, tornando-os adequados para uma vasta gama de aplicações.

3.2 VHDL

Segundo (D'AMORE, 2015), a linguagem VHDL (VHSIC Hardware Description Language) foi desenvolvida nos anos 80 por IBM, *Texas Instruments* e *Intermetrics* para atender à necessidade de projetar e documentar de forma padronizada os circuitos do projeto VHSIC (Very High Speed Integrated Circuit), liderado pela DARPA (Departamento de Defesa dos Estados Unidos). Em 1987, a IEEE ratificou a primeira versão da linguagem, tornando-a um padrão.

A linguagem VHDL permite a criação de projetos hierárquicos, o que significa que circuitos descritos em VHDL podem ser reutilizados em circuitos maiores, facilitando a modularidade. A descrição dos circuitos pode ser feita de duas formas: estrutural, que representa a interconexão entre diferentes componentes, e comportamental, que descreve o comportamento do circuito por meio de código.

Um arquivo VHDL possui uma estrutura mínima composta por uma *entity* (entidade) e uma *architecture* (arquitetura). Na entidade, são definidos apenas os sinais de entrada e saída do circuito, especificando o tipo desses sinais, como valor lógico binário (bit), inteiro, número real, entre outros, além da possibilidade de vetores desses tipos.

Na arquitetura, é descrito o funcionamento do circuito, que pode ser detalhado de forma estrutural (por meio da interconexão entre componentes), comportamental (descrevendo o comportamento lógico) ou uma combinação de ambas.

3.3 UART

O protocolo *Universal Asynchronous Receiver Transmitter* (UART) é um método de comunicação de hardware full-duplex, conhecido por sua simplicidade, pois requer apenas dois fios para a comunicação bidirecional: um para transmissão (TX) e outro para recepção (RX). Essa característica facilita a integração e reduz os custos do sistema de comunicação, tornando-o ideal para diversas aplicações. Sendo um protocolo assíncrono, o UART não necessita de um sinal de clock para sincronizar a comunicação, permitindo a transmissão de dados entre dispositivos sem a exigência de uma referência de tempo comum (PEÑA; LEGASPI, 2020).

Para a comunicação UART funcionar adequadamente, o hardware deve possuir ao menos duas portas, RX (recepção) e TX (transmissão), ambas operando com a mesma velocidade e configurações, garantindo a integridade dos dados. A Figura 1 ilustra a interação eficiente entre dois dispositivos UART, destacando a importância das portas RX e TX na transmissão de dados em sistemas embarcados e outras aplicações de comunicação serial.

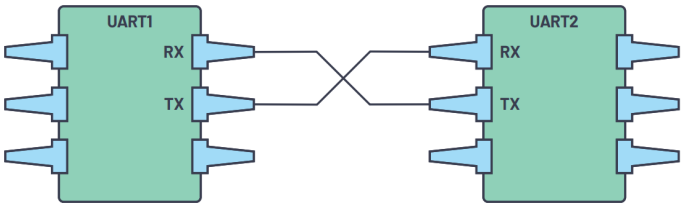


Figura 1 – Comunicação direta entre dois dispositivos UART. **Fonte:** (PEÑA; LEGASPI, 2020).

4 Desenvolvimento

Esta seção descreve os materiais e métodos utilizados, assim como a arquitetura do laboratório remoto. A abordagem descrita apresenta o desenvolvimento de um sistema de transmissão e recepção de estados lógicos de chaves, botões, LEDs e displays de sete segmentos, permitindo a comunicação entre um servidor e o kit de desenvolvimento DE1-SoC (referido simplesmente como placa FPGA). A implementação visa garantir uma interface eficiente e responsiva entre o servidor e a FPGA, permitindo o controle remoto dos dispositivos e facilitando a interação entre o usuário e os elementos do hardware.

4.1 Materiais e Métodos

O desenvolvimento deste trabalho seguiu a metodologia bottom-up proposta por (WOLF, 2012), na qual cada parte do projeto foi abordada de maneira incremental. Inicialmente, foram desenvolvidos componentes individuais, que foram posteriormente integrados para formar o sistema completo. O fluxo do projeto é ilustrado na Figura 2.

O desenvolvimento começou com a especificação de um laboratório remoto baseado no modelo cliente-servidor. Este laboratório remoto deve oferecer duas funcionalidades

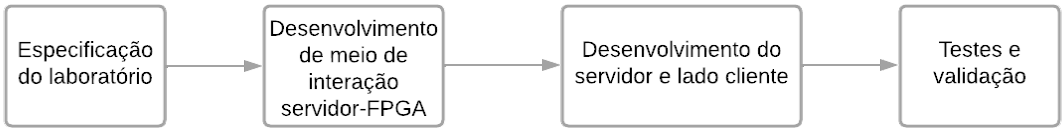


Figura 2 – Fluxo do projeto. **Fonte:** Do autor.

principais: a configuração da placa FPGA, que consiste na gravação de um projeto de circuito digital na placa para sua execução — este projeto, por exemplo, envolve o uso de estados lógicos de chaves, botões, LEDs e displays de sete segmentos — e a interação remota do usuário com a placa FPGA após a configuração.

A configuração da placa FPGA foi integrada ao laboratório sem dificuldades, uma vez que o fabricante já disponibiliza ferramentas adequadas para essa tarefa (ver seção 4.2.2). No entanto, a interação remota com a placa FPGA não é suportada nativamente pelo fabricante. Portanto, o próximo passo do projeto foi desenvolver um meio de comunicação entre a placa FPGA e o servidor, utilizando transmissão de dados via UART. Nesta transmissão, os estados são codificados em sinais binários lógicos de 0 ou 1, representando a ativação ou desativação dos periféricos virtuais da placa FPGA.

Os periféricos virtuais representados na transmissão incluem 10 LEDs, 6 displays de sete segmentos, 10 chaves seletoras e 4 botões. Embora esses periféricos estejam presentes na placa FPGA, eles são simulados pelo laboratório remoto, uma vez que a interação física direta com a placa não é possível. Neste trabalho, os LEDs e displays de sete segmentos são denominados periféricos de saída, pois representam dados que podem ser observados, mas não alterados diretamente pelo usuário. Por outro lado, os botões e chaves são classificados como periféricos de entrada, pois permitem ao usuário fornecer dados e interagir com o laboratório remoto.

Nesta fase, foi desenvolvido um componente auxiliar em VHDL — um circuito digital composto por uma interconexão de portas lógicas e flip-flops que processam os sinais de entrada para produzir sinais de saída — com a função de estabelecer a comunicação entre a FPGA e o servidor, possibilitando a interação com os periféricos da placa FPGA. Após a finalização desse componente, foi desenvolvido o servidor, assegurando a integração completa do sistema.

Os materiais e softwares utilizados neste projeto foram:

- 1x kit de desenvolvimento FPGA DE1-SoC;
- 1x módulo conversor USB-UART CP2102;
- Quartus Prime Lite 22.1;
- Node.js, com as bibliotecas express e serialport.

4.1.1 Kit de desenvolvimento Terasic DE1-SoC

O Kit de desenvolvimento DE1-SoC, ilustrado na Figura 3, é equipado com um circuito integrado FPGA Intel Cyclone SoC V 5CSEMA5F31C6N. Para este trabalho, os seguintes recursos foram essenciais:

- Periféricos de entrada e saída:
  - 10x chaves seletoras (referidas como Switch x10 na Figura 3)
  - 4x botões (referidas como Button x4 na Figura 3)
  - 10x LEDs (referidas como LED x10 na Figura 3)
  - 6x displays de sete segmentos (referidas como 7-Segment Display na Figura 3)
- Entrada de alimentação (Power DC Jack na Figura 3)
- Entrada USB para configuração do FPGA (USB-Blaster II na Figura 3)
- Pinos de entrada/saída de uso geral (2x20 GPIO x2 na Figura 3)

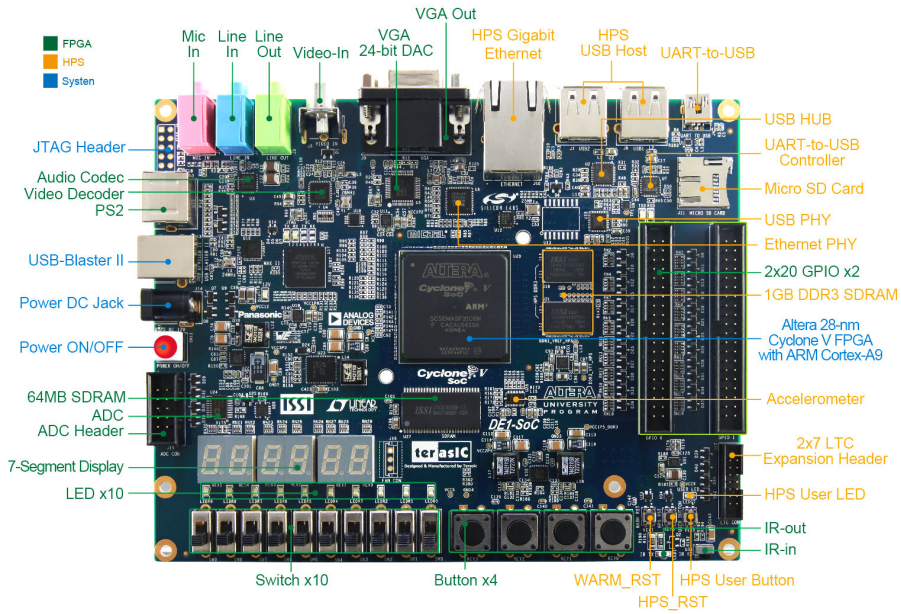


Figura 3 – Kit de desenvolvimento Terasic DE1-SoC. **Fonte:** (TERASIC, 2024).

4.2 Arquitetura do Laboratório Remoto

Como descrito anteriormente, este trabalho foi uma colaboração entre o professor Marcelo Berejuck, da Universidade Federal de Santa Catarina (UFSC/Brasil), e o Laboratory at Distance (L@D) da Université TÉLUQ (Canadá), onde foi implementado a camada zero do laboratório remoto como ilustrado na Figura 4, integrando a placa FPGA a um computador servidor para permitir o acesso remoto à placa, e propondo uma arquitetura mínima de laboratório que viabilize a interação remota.

O laboratório remoto foi implementado utilizando o modelo cliente-servidor, onde o servidor atua como intermediário na comunicação entre o usuário e o kit de desenvolvimento DE1-SoC, denominado aqui como a placa FPGA. No lado do cliente, foi desenvolvida uma interface de usuário que simula a interação física com a placa FPGA. Essa interface opera em conjunto com a placa, o qual é configurada com um projeto compatível com a arquitetura descrita ao final da Seção 4.2.2. A seguir serão detalhados as diferenças entre os processos de execução de um projeto em dois ambientes: o convencional, no qual o usuário interage fisicamente com a placa FPGA, e o remoto, sendo a proposta deste

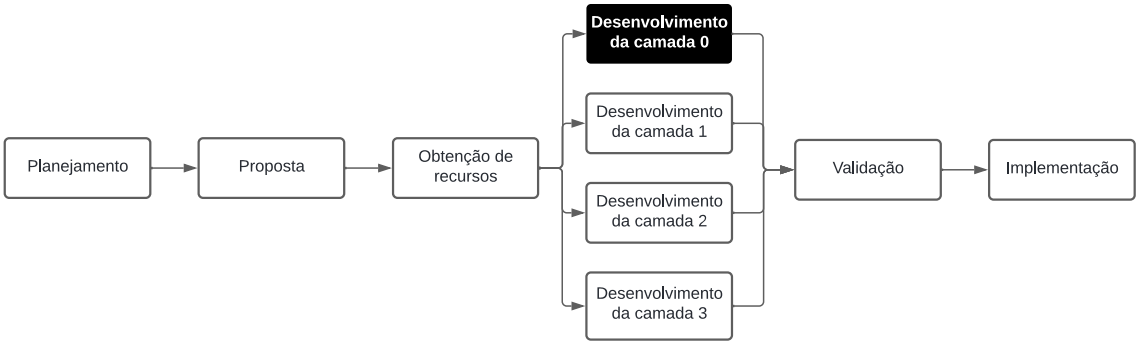


Figura 4 – Fluxo do projeto proposto por (BEREJUCK et al., 2022). **Fonte:** adaptado de autor (PEREIRA, 2022).

trabalho, onde o laboratório é acessado à distância, permitindo a interação com o placa física remotamente.

4.2.1 Ambiente convencional

Em um ambiente convencional, onde o usuário precisa interagir fisicamente com a placa FPGA, o desenvolvimento do código é realizado utilizando o pacote de softwares Quartus Prime para criar o projeto. Neste projeto, o usuário desenvolve um circuito digital baseado em portas lógicas. Para implementar esse circuito, é necessário escrever um código em VHDL, que define a estrutura de entradas e saídas, bem como a lógica de execução do circuito digital.

Após o desenvolvimento do circuito digital no Quartus Prime, o usuário deve seguir as etapas abaixo na interface do ambiente de design da ferramenta para configurar o FPGA:

- Mapeamento dos pinos: O usuário utiliza a ferramenta “Pin Mapper” do pacote Quartus para associar as entradas e saídas do circuito digital aos pinos físicos do FPGA. Isso permite que o FPGA saiba como conectar os sinais internos aos pinos externos.
- Síntese do circuito: em seguida, o usuário inicia o processo de síntese do circuito digital. Nesse estágio, o código-fonte é compilado, gerando um arquivo de configuração chamado “bitstream”. Esse arquivo contém informações de configuração específicas para a implementação do circuito no FPGA.
- Configuração do FPGA: Com o arquivo bitstream em mãos, o usuário utiliza a ferramenta “Programmer” do Quartus para configurar o FPGA. A placa FPGA está conectada ao computador via porta USB durante esse processo.

Após seguir essas etapas, o usuário estará pronto para interagir com o circuito digital implementado no FPGA. Por exemplo:

- Utilizando um microcontrolador: O usuário pode conectar um microcontrolador aos pinos de entrada mapeados previamente. Isso permite a inserção dos sinais de entrada no circuito digital e ler os sinais de saída gerados pelo FPGA.
- Kits de desenvolvimento: Em kits de desenvolvimento como o DE1-SoC, alguns pinos de entrada e saída já estão permanentemente conectados a botões, LEDs e

outros periféricos. O usuário pode interagir com o circuito pressionando os botões do kit e observando as saídas nos LEDs e displays. A Figura 5 ilustra este tipo de interação, onde é possível observar o tempo decorrido, expresso em minutos, segundos e centésimos de segundo nos displays de sete segmentos.

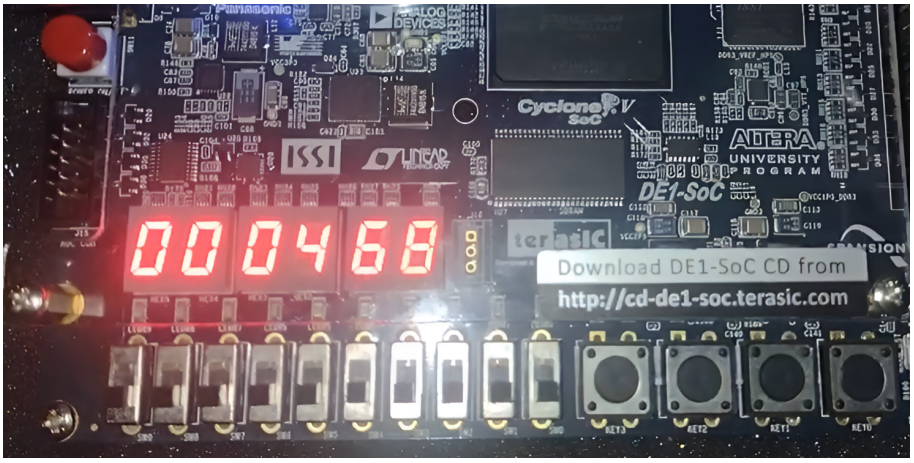


Figura 5 – Ilustração de um DE1-SoC, operando um circuíto digital que implementa um cronômetro. **Fonte:** Do autor.

4.2.2 Ambiente Remoto

Por outro lado, o acesso remoto a placa FPGA, que é a proposta deste trabalho, requer que o laboratório remoto disponibilize meios de interação com o placa. Nesse sentido, o laboratório contará com uma interface gráfica no lado cliente, a partir de agora denominada interface de usuário, que permitirá que o usuário interaja com a placa FPGA remotamente, por meio das ações seguintes:

- **Configuração da placa FPGA:** A interface de usuário permite que o usuário selecione e envie arquivos de configuração ao FPGA por meio da interface. Após o envio do arquivo, a interface sinaliza ao usuário se a configuração foi bem-sucedida ou se houve erros.
- **Alteração e observação de estados de periféricos:** A interface de usuário simula os periféricos de entrada e saída presentes na placa FPGA, permitindo que o usuário interaja com estes periféricos simulados, alterando os estados dos periféricos de entrada, e observando os estados dos periféricos de saída. Isto fornece ao usuário um análogo remoto as ações que seriam realizadas durante o teste de um circuito digital em um ambiente presencial.

Para garantir o funcionamento adequado da interface do usuário, que envolve a transmissão de estados de botões e chaves seletoras em formato binário (zeros e uns), simulando entradas digitais em um kit de desenvolvimento e, consequentemente, recebendo respostas visualizadas por meio de LEDs e displays de sete segmentos, é importante estabelecer uma conexão com um servidor que acomode os serviços oferecidos pela interface de usuário:

- **Configuração da placa FPGA:** O servidor transmite o arquivo de configuração, pré-definido ou enviado pelo usuário à placa FPGA e retorna ao usuário se a configuração foi bem-sucedida ou não.
- **Alteração e observação de estados de periféricos:** O servidor transmite à placa FPGA os estados dos periféricos de entrada conforme as entradas atribuídas pelo usuário na interface de usuário, e o resultado executado pela placa FPGA é enviada para a interface de usuário onde os estados de saída são atualizados conforme o circuito digital proposto pelo usuário.

4.2.3 Comunicação entre o Servidor e a Placa FPGA

A placa FPGA, componente essencial do sistema, não possui nativamente um meio de interação virtual que possa ser explorado pelo servidor para alterar e monitorar sinais digitais. A interação está restrita a botões, chaves e pinos físicos, além do monitoramento visual por meio de LEDs e displays de sete segmentos, que ocorrer presencialmente.

Para viabilizar a interação virtual e permitir que o servidor se comunique com o circuito digitai, sendo o projeto desenvolvido pelo usurário baseado em portas lógicas, por meio de um barramento, como o USB, utiliza-se um adaptador CP2102 USB para UART. Esse adaptador, quando conectado à placa FPGA, oferece sinais de recepção e transmissão UART, possibilitando a comunicação entre qualquer circuito digital implementado na placa FPGA e o servidor via porta USB. Contudo, isso exigiria que cada usuário desenvolvesse um circuito digital capaz de interpretar sinais UART, realizar a lógica interna e fornecer saídas em sinais UART, o que poderia comprometer a usabilidade do laboratório remoto.

Para resolver o problema de conexão com o servidor via UART, foi desenvolvido um circuito digital denominado "monitor". Este circuito, composto por uma interconexão de portas lógicas e flip-flops, estabelece a comunicação externa com o servidor através de um barramento UART. Internamente, o monitor fornece à FPGA sinais de entrada que representam os estados lógicos de botões e chaves seletoras, além de sinais de saída correspondentes aos estados de LEDs e displays de sete segmentos. Esses sinais são essenciais para o funcionamento remoto do experimento, um circuito digital desenvolvido pelo usuário, que também é composto por uma interconexão de portas lógicas e flip-flops. O "experimento" opera com sinais de entrada representando periféricos da placa FPGA, como chaves seletoras e botões, e gera sinais de saída correspondentes a periféricos de saída, como LEDs e displays de sete segmentos.

Para possibilitar a interação do usuário com o experimento, é necessário conectar os pinos dos periféricos do circuito digital monitor aos pinos correspondentes do projeto de interação da FPGA. Os pinos UART devem ser conectados aos pinos físicos da FPGA, permitindo a interação com o adaptador CP2102. Essa arquitetura resulta em um projeto VHDL válido para o laboratório remoto, conforme ilustrado na Figura 6. Ao compilar este projeto, composto pela interconexão entre o monitor e o experimento, gera-se um arquivo de configuração compatível com o laboratório remoto proposto.

Dessa forma, o laboratório remoto para a placa FPGA é composto por três partes integradas: a interface do usuário no lado do cliente, o computador servidor do laboratório e o FPGA DE1-SoC, que juntos formam o lado servidor do laboratório. Essas três partes trabalham em conjunto para criar um ambiente interativo de laboratório remoto. A arquitetura do laboratório remoto está ilustrada na Figura 7. Cada uma dessas partes será detalhada nas seções subsequentes.

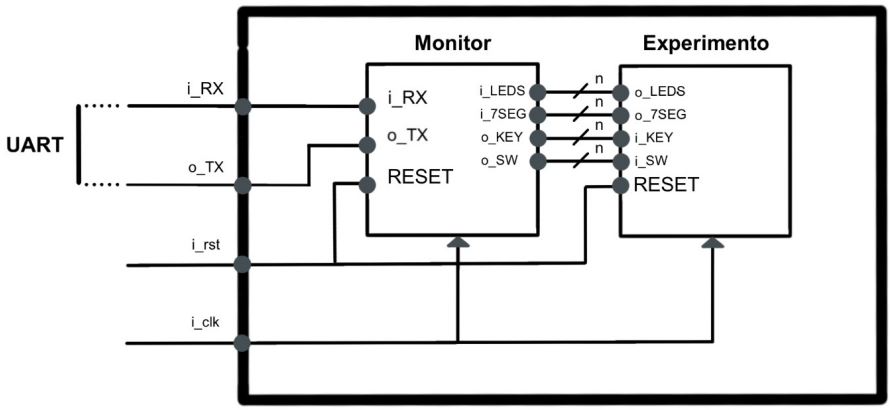


Figura 6 – Arquitetura mínima para funcionamento de um projeto no laboratório. **Fonte:** Do autor.

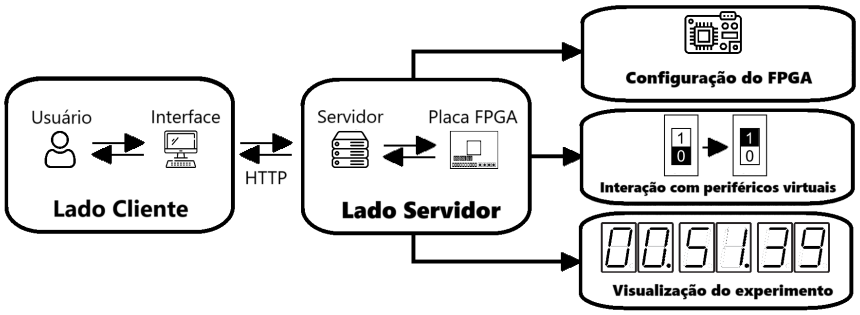


Figura 7 – Arquitetura proposta para o laboratório remoto. **Fonte:** Do autor.

5 Implementação

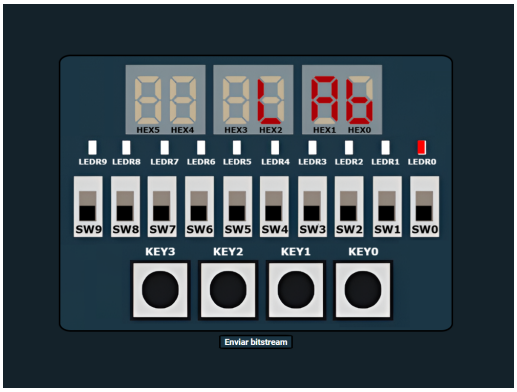
A implementação do laboratório remoto foi estruturada para proporcionar uma interação intuitiva e eficiente com a placa FPGA, facilitando tanto a configuração quanto a observação dos periféricos simulados. Esta seção descreve a construção e funcionamento dos dois principais componentes do sistema: a interface de usuário e o lado servidor.

5.1 Interface de Usuário

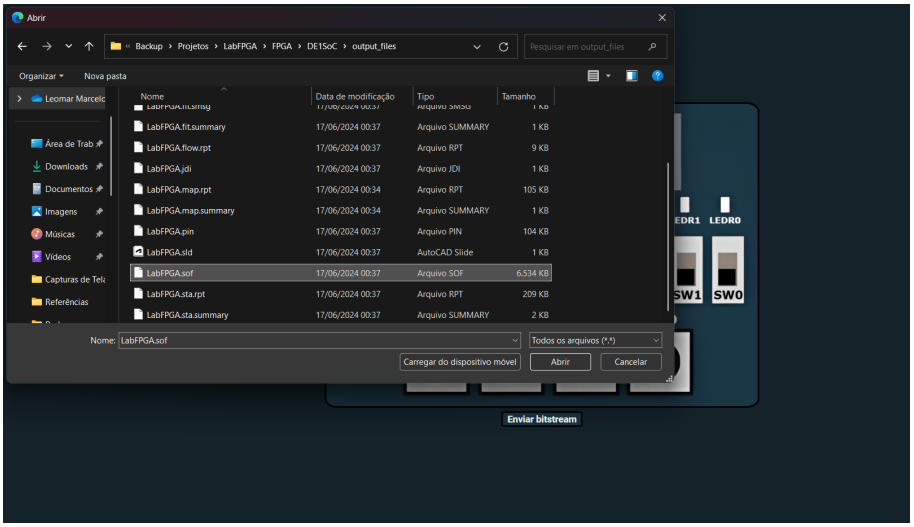
A interface de usuário foi projetada para permitir a interação direta com a placa FPGA simulada. Como ilustrado na Figura 8a, a interface é composta por elementos visuais que replicam os periféricos da placa física, incluindo displays de sete segmentos, LEDs, chaves seletoras e botões. A simulação desses periféricos é essencial para a visualização e operação da configuração do FPGA. Os principais componentes da interface incluem:

- **Placa FPGA simulada:** Representa a placa física e contém elementos como 6 displays de sete segmentos, 10 LEDs, 10 chaves seletoras e 4 botões. As chaves seletoras e botões imitam as interações físicas, com as chaves ativadas por cliques e os botões acionados enquanto pressionados. Os LEDs e displays são atualizados pelo servidor e não podem ser alterados pelo usuário.

- **Botão de configuração:** Permite ao usuário carregar e aplicar um circuito digital no FPGA. O botão aciona uma janela de seleção de arquivos, ilustrada pela Figura 8b, para escolher um arquivo de configuração compatível com a arquitetura descrita na Seção 4.2.2, que é então enviado ao servidor. Após o envio, o servidor confirma a aplicação da configuração e reinicia o FPGA e os periféricos simulados na interface.



(a) Interface de Usuário



(b) Interface e Seletor de Arquivos

Figura 8 – Ilustração da Interface de Usuário. Em (a) temos uma visão geral da interface durante um experimento ativo; em (b) temos o seletor de arquivos onde o usuário pode selecionar o bitstream que configurará a placa FPGA. **Fonte:** Do autor.

5.2 Lado Servidor

O lado servidor foi desenvolvido para suportar as funcionalidades oferecidas pela interface de usuário, sendo dividido em três partes principais:

- **Comunicação com a Interface de Usuário:** Utiliza o protocolo WebSockets para comunicação bidirecional e velocidade na transmissão de dados. A interface de usuário envia mensagens ao servidor para relatar alterações nos estados dos periféricos de entrada, como a detecção de novos estados.Por sua vez, o servidor informa a interface sobre qualquer mudança nos estados dos periféricos de saída.

- **Configuração da Placa FPGA:** Ao receber um arquivo de configuração via HTTP, o servidor grava o arquivo em disco e executa o processo de programação utilizando a ferramenta **quartus_pgm** da Intel. O servidor monitora o progresso da programação para garantir sua conclusão bem-sucedida.
- **Interação com a Placa FPGA:** Dada a falta de um meio nativo para interação com o servidor, foi implementado um circuito digital monitor na placa FPGA, conforme especificado na Seção 4.2.2. Esse circuito possibilita a comunicação entre o FPGA e o servidor, descrita abaixo, permitindo a alteração dos sinais e a interação com a placa.

A Figura 9 ilustra o processo de comunicação implementado entre o servidor e o circuito digital monitor, que chamaremos de FPGA para simplificação.

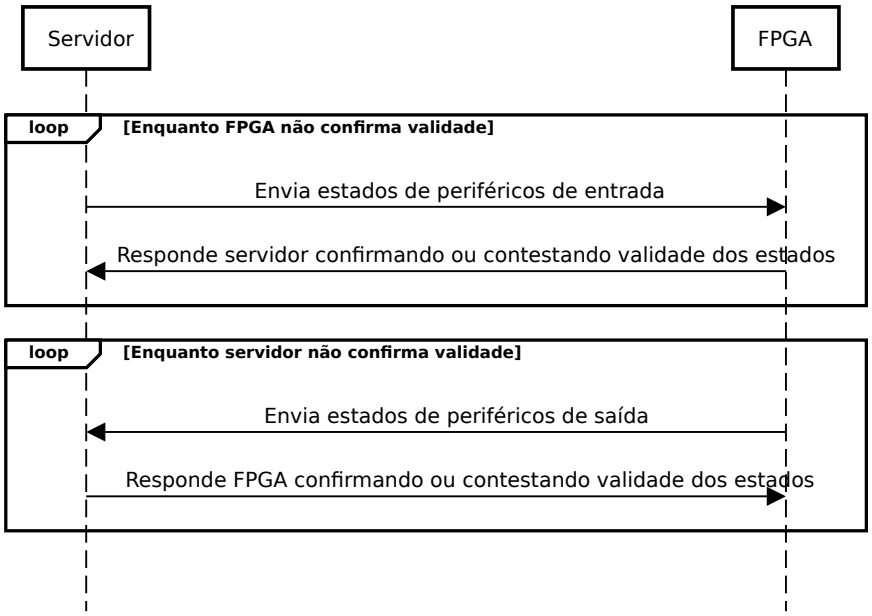


Figura 9 – Diagrama de interação descrevendo interação entre servidor e FPGA. **Fonte:** Do autor

A comunicação começa com o servidor enviando os estados dos periféricos de entrada para o FPGA. Juntamente com os dados, um código de verificação cíclica de redundância (CRC-8) é incluído. Esse código é calculado com base nos estados dos periféricos e serve para detectar possíveis alterações na mensagem devido a interferências durante a transmissão. Após receber esses dados, o FPGA verifica a integridade da mensagem utilizando o código CRC-8 e responde ao servidor indicando se a transmissão foi bem-sucedida. Se houver falha na verificação, o FPGA reenvia os estados dos periféricos de entrada.

Em seguida, o FPGA envia ao servidor os estados dos periféricos de saída, também acompanhados de um código CRC-8. O servidor então valida a mensagem e responde ao FPGA confirmando a integridade da transmissão. Se nenhum erro for detectado, a comunicação é reiniciada. Caso contrário, o FPGA reenviará os estados dos periféricos de saída até que a transmissão seja confirmada como bem-sucedida.

Durante a comunicação, caso ocorra algum travamento, toda a comunicação é reiniciada. Para que isto ocorra, o servidor e o monitor implementam um temporizador watchdog, um componente temporizador que exige reinicialização periódica de sua contagem,

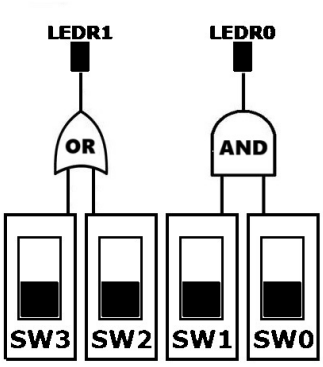
pois caso o temporizador atinja seu limite programado, a rotina de comunicação é reiniciada forçadamente. Este mecanismo impede que o laboratório pare de funcionar devido a erros de comunicação que possam dessincronizar o servidor com FPGA durante a transmissão de dados.

6 Resultados e Discussões

Nesta seção, será apresentado um estudo de caso que demonstra o funcionamento do sistema desenvolvido ao executar um circuito digital. Além disso, será realizada uma análise qualitativa do sistema por meio da Escala de Usabilidade do Sistema (SUS - System Usability Scale), conforme descrito por (BROOKE, 1995). A SUS consiste em uma avaliação na qual os usuários atribuem notas a dez afirmações sobre o sistema, utilizando uma escala de 1 (discordo totalmente) a 5 (concordo totalmente).

6.1 Estudo de Caso

Para a demonstração, o laboratório remoto foi configurado com um circuito digital básico, conforme ilustrado na Figura 10a. Este circuito consiste em uma porta lógica AND e uma porta lógica OR, utilizando chaves seletoras como sinais de entrada e LEDs para exibir as saídas. Na Figura 10a, observa-se o diagrama do circuito, que mostra como as entradas das chaves seletoras são processadas para controlar os LEDs. A implementação do circuito, descrita em VHDL, é apresentada na Figura 10b, onde se encontra o código que define o comportamento lógico das portas AND e OR utilizadas.



(a) Circuíto digital implementado.

```vhdl
architecture rtl of USER_VHDL is
begin

   -- Apagando os displays de sete segmentos.
   o_HEX5 <= "1111111";
   o_HEX4 <= "1111111";
   o_HEX3 <= "1111111";
   o_HEX2 <= "1111111";
   o_HEX1 <= "1111111";
   o_HEX0 <= "1111111";

   -- Apagando os LEDs não utilizados
   o_LEDR(9 downto 2) <= "00000000";

   -- Atribuindo lógica aos LEDs 0 e 1
   o_LEDR(0) <= i_SW(0) AND i_SW(1);
   o_LEDR(1) <= i_SW(2) OR i_SW(3);

end rtl;
```

(b) Código em VHDL do circuíto digital.

Figura 10 – Representações do circuíto digital implementado. **Fonte:** Do autor.

O circuito consiste em duas operações lógicas principais, que utilizam diferentes combinações de chaves seletoras para controlar os LEDs:

- Primeira Operação Lógica (AND): As chaves seletoras SW0 e SW1 estão conectadas a uma porta AND. Essa configuração faz com que o LED 0 acenda somente quando ambas as chaves (SW0 e SW1) estão na posição "ligada", indicando um nível lógico alto em ambas as entradas. Caso uma das chaves esteja "desligada", o LED 0 permanece apagado.

- Segunda Operação Lógica (OR): As chaves seletoras SW2 e SW3 estão conectadas a uma porta OR. Nesta configuração, o LED 1 acenderá se pelo menos uma das chaves (SW2 ou SW3) estiver na posição "ligada"(nível lógico alto). Isso significa que o LED 1 será apagado apenas quando ambas as chaves SW2 e SW3 estiverem na posição "desligada".

Na prática, as chaves seletoras no kit FPGA funcionam como entradas de nível lógico: na posição "desligada"(abaixada), correspondem a um nível lógico baixo; na posição "ligada"(levantada), correspondem a um nível lógico alto. Os LEDs, por sua vez, refletem o resultado das operações lógicas: ficam apagados para um nível lógico baixo e acesos para um nível lógico alto. A Figura 11 exibe a simulação do circuito digital, realizada no software ModelSim.

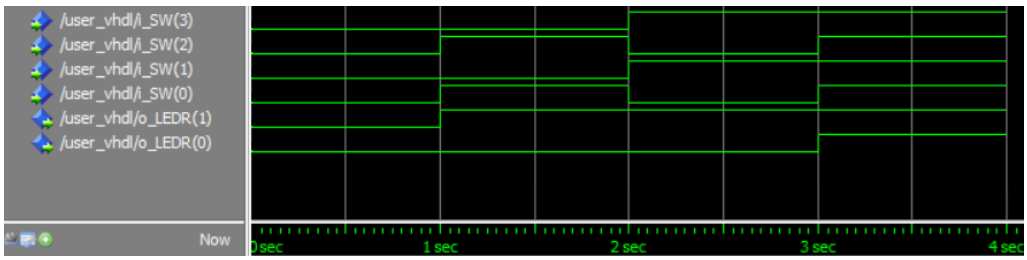


Figura 11 – Simulação do circuíto digital. **Fonte:** Do autor.

Na simulação, as quatro primeiras ondas representam os sinais de entrada provenientes das chaves seletoras, dispostas em ordem decrescente: SW3 a SW0. As duas últimas ondas indicam os sinais de saída dos LEDs, com LEDR1 acima de LEDR0. Inicialmente, todas as ondas estão em nível lógico baixo, e, quando elevadas, passam a representar nível lógico alto.

A simulação foi realizada para testar os quatro estados possíveis das portas lógicas. No caso da porta lógica AND, o LED 0 acende somente quando ambas as chaves seletoras SW0 e SW1 estão em nível lógico alto (ligadas), atendendo à condição AND. Se uma ou ambas as chaves estejam desligadas (nível lógico baixo), o LED permanece apagado. Os casos testados na simulação, separados em passos de tempo de um segundo, foram:

a. **Todas as chaves seletoras desligadas:** Ambos os LEDs desligados

b. **Apenas SW0 e SW2 acionadas:** LEDR0 desligado, LEDR1 ligado.

c. **Apenas SW1 e SW3 acionadas:** LEDR0 desligado, LEDR1 ligado.

d. **Todas as chaves seletoras acionadas:** Ambos os LEDs ligados.

No laboratório remoto, os testes realizados na simulação foram replicados com precisão. A Figura 12 mostra a interface de usuário do laboratório, juntamente com o FPGA utilizado, executando o terceiro (**c**) caso de teste, com as chaves SW1 e SW3 ligadas. A Figura 13 apresenta exclusivamente a interface de usuário para cada cenário testado na simulação, na ordem descrita nos casos de testes acima. Observa-se que os LEDs na interface respondem de forma idêntica para cada caso de teste simulado.

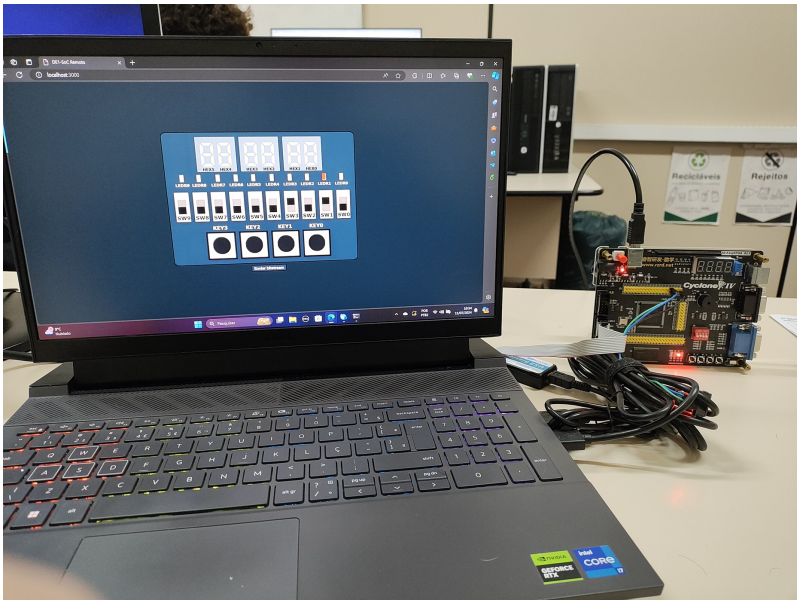
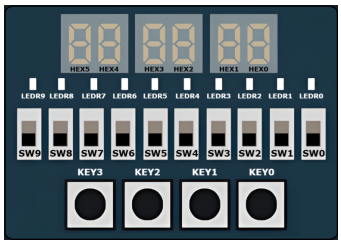
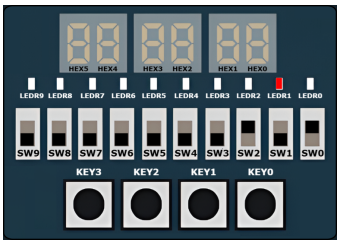


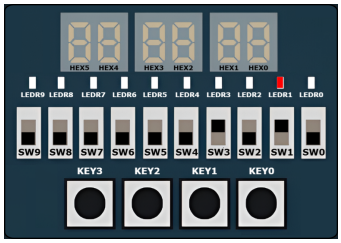
Figura 12 – Laboratório remoto em execução. **Fonte:** Do autor.



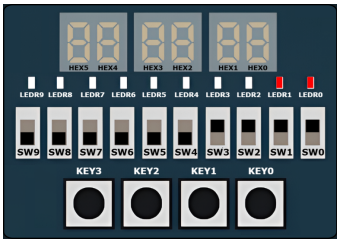
(a) Chaves e LEDs desligados.



(b) SW0 e SW2 acionadas, LEDR1 ligado.



(c) SW1 e SW3 acionados, LEDR1 ligado.



(d) Todas chaves ligadas, todos LEDS ligados.

Figura 13 – Laboratório Remoto, apenas ambiente virtual. **Fonte:** Do autor.

6.2 Validação Qualitativa com Usuários

Os testes qualitativos foram realizados com a participação dos alunos da disciplina de Linguagens de Descrição de Hardware, que utilizaram o laboratório remoto entre 03/06/2023 e 21/06/2023. Nesse período, onze alunos, de um total de 17 distribuídos entre as turmas 04655A (matutino) e 04655B (noturno), desenvolveram e testaram circuitos digitais por meio do laboratório remoto. O servidor do laboratório foi hospedado em um laptop Asus F570ZD, conectado à rede interna da UFSC Araranguá, permitindo o acesso ao sistema tanto no campus quanto remotamente, via VPN da UFSC.

Após a utilização, os alunos preencheram um formulário baseado na escala de usabilidade SUS (System Usability Scale) descrito por (BROOKE, 1995), adaptada para

o contexto do laboratório. Além disso, foi disponibilizado um campo aberto para que os usuários justificassem suas respostas e sugerissem melhorias para o sistema. As afirmações adaptadas abordaram os seguintes pontos:

1. Gostaria de usar esse sistema com frequência.
2. O sistema desnecessariamente complexo.
3. O sistema é fácil de usar.
4. É necessário ajuda de uma pessoa com conhecimentos técnicos para usar o sistema.
5. As funções do sistema estão bem integradas.
6. O sistema apresenta muita inconsistência.
7. As pessoas aprenderão a usar este sistema rapidamente.
8. O sistema é confuso de usar.
9. Sinto-me confiante ao utilizar o sistema.
10. Foi preciso aprender várias coisas novas antes de utilizar o sistema.

No total, onze alunos participaram dos testes. A Figura 14 apresenta a pontuação média para as afirmações obtidas ao final do período de testes.

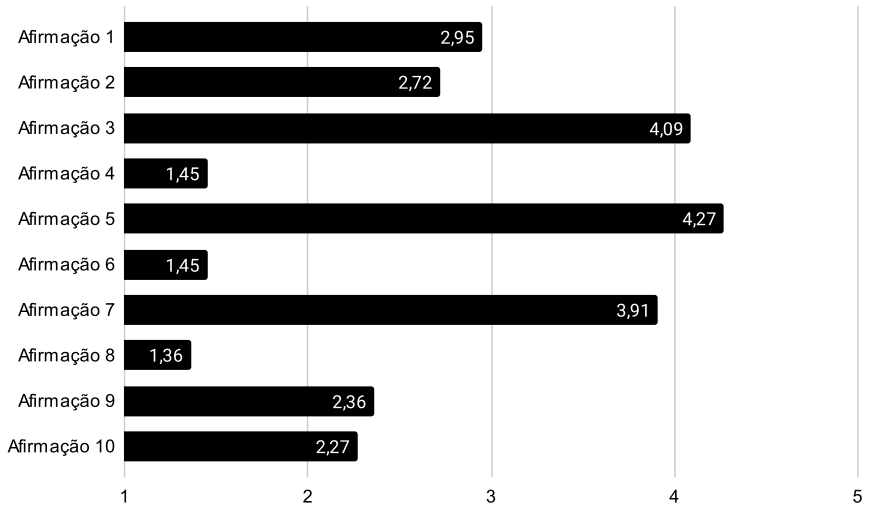


Figura 14 – Pontuação média para as afirmações. **Fonte:** Do autor

Conforme descrito por (BROOKE, 1995), o cálculo da pontuação final do sistema é feito somando as contribuições de cada afirmação, seguindo os critérios abaixo:

- Para afirmações ímpares, a contribuição é a pontuação obtida menos 1;
- Para afirmações pares, a contribuição é de 5 menos a pontuação obtida.

Após somar as contribuições, multiplica-se o total por um fator de 2,5 para obter a pontuação final. Usando este método, o sistema alcançou uma pontuação de 70,82. De acordo com (BANGOR et al., 2009 apud BROOKE, 2013), conforme ilustrado pela Figura 15, essa nota é classificada como aceitável.

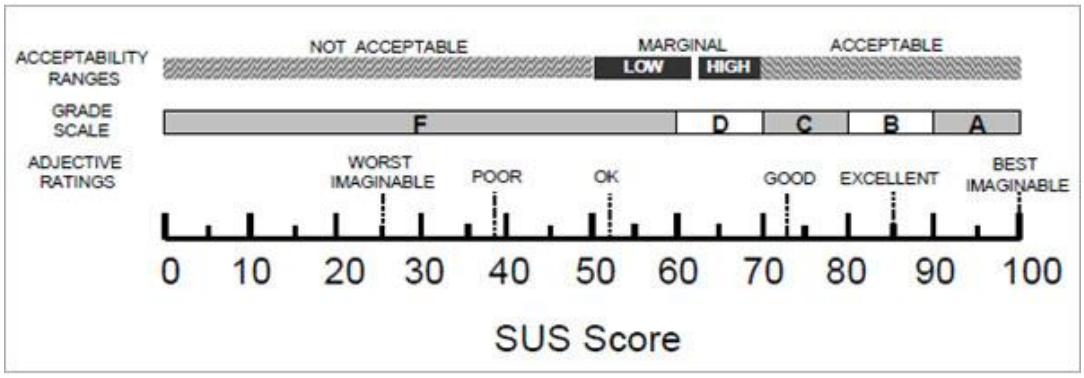


Figura 15 – Classificação das pontuações SUS. **Fonte:** (BANGOR et al., 2009 apud BROOKE, 2013)

7 Conclusão

O trabalho desenvolvido implementou um laboratório remoto para FPGA, permitindo aos usuários configurar remotamente o FPGA com um circuito digital próprio e interagir com ele por meio de uma interface gráfica. Essa interface simula os periféricos existentes na placa FPGA, como botões, chaves seletoras, LEDs e displays de sete segmentos. Além disso, foram desenvolvidos mecanismos de detecção e correção de falhas, permitindo que o laboratório se recupere de erros sem a necessidade de reiniciar ambos servidor e placa FPGA.

Nos testes realizados, o laboratório remoto funcionou conforme esperado, sem apresentar problemas de latência ao acionar os periféricos na interface de usuário. Os sinais gerados pela placa FPGA, conforme os estados dos periféricos de entrada definidos na interface, foram compatíveis aos sinais observados durante simulações, conforme demonstrado na Seção 6.1. Estes testes foram conduzidos em uma rede local e circuitos digitais simples, o que contribuiu para o desempenho estável do sistema. Os testes também incluíram a introdução de falhas na comunicação - como a corrupção de pacotes, ou a desconexão deliberada dos fios UART conectados na placa FPGA - para testar a capacidade do sistema de se recuperar rapidamente e de maneira imperceptível ao usuário. O laboratório remoto foi capaz de lidar tanto com pacotes corrompidos e também com travamentos oriundos da perda de sinal ou sincronia na comunicação entre o FPGA e o servidor.

Além disso, foi possível avaliar a usabilidade do sistema por meio da escala SUS, obtendo uma pontuação de 70,82, o que indica um nível de usabilidade aceitável. No entanto, alguns pontos de melhoria foram identificados, tais como:

- **Latência:** alguns usuários sofreram com conexões de alta latência, fazendo com que suas ações demorassem muito para se concretizarem, dando a impressão que o sistema não estava reconhecendo suas ações.

- **Complexidade:** embora o sistema seja fácil de usar, muitos usuários tiveram dificuldades em usar o sistema sem um manual explicativo de como desenvolver e integrar seus circuítos digitais com o laboratório. Alguns alunos apontaram que uma aula dedicada a apresentação do laboratório teria sido de grande ajuda.

Após avaliação das respostas dos alunos em relação ao laboratório remoto, foram identificadas algumas áreas que precisam ser melhoradas em trabalho futuros:

- Oferecer uma documentação mais clara aos usuários do laboratório;
- Permitir a utilização do laboratório por vários usuários simultaneamente;
- Desenvolver uma maneira de integração do circuito digital desenvolvido pelo usuário com o laboratório remoto que seja transparente ao usuário.
- Ajustar a interface de usuário para suavizar a experiência de usuários sob uma conexão de alta latência.

A implementação dessas melhorias pode eliminar grande parte dos problemas relatados pelos usuários durante o teste qualitativo.

Referências

BANGOR, A. et al. Determining what individual sus scores mean: Adding an adjective rating scale. *J. Usability Stud.*, v. 4, p. 114–123, 04 2009. Citado na página 24.

BEREJUCK, M. D. et al. Work-in progress: Development of an e-learning fpga platform following the ieee sa std. 1876 - 2019 standard for networked smart learning objects for online laboratories. In: AUER, M. E.; BHIMAVARAM, K. R.; YUE, X.-G. (Ed.). *Online Engineering and Society 4.0*. Cham: Springer International Publishing, 2022. p. 3–9. ISBN 978-3-030-82529-4. Citado 2 vezes nas páginas 8 e 14.

BROOKE, J. Sus: A quick and dirty usability scale. *Usability Eval. Ind.*, v. 189, 11 1995. Citado 3 vezes nas páginas 20, 22 e 23.

BROOKE, J. Sus: a retrospective. *Journal of Usability Studies*, v. 8, p. 29–40, 01 2013. Citado na página 24.

CHU, P. P. *FPGA Prototyping by VHDL Examples*. [S.l.]: John Wiley & Sons, Inc., 2008. 11 p. Citado na página 10.

D'AMORE, R. *VHDL - Descrição e Síntese de Circuitos Digitais*. [S.l.]: LTC, 2015. 2–3 p. Citado na página 10.

DE MEDEIROS, R. B. Laboratório remoto baseado em fpga aplicado nas disciplinas de prática de eletrônica digital 1 e 2 da faculdade unb gama. *Universidade de Brasília*, p. 1–56, 2018. Citado na página 9.

GARCIA-LORO, F.; CRISTOBAL, E. S.; CASTRO, M. Pilar: A federation of visir systems for analog electronics. In: IEEE. *2019 5th Experiment International Conference (exp. at'19)*. [S.l.], 2019. p. 260–261. Citado na página 8.

MOHSEN, A. E.-R.; GADALRAB, M. Y. et al. *Remote FPGA Lab For ZYNQ and Virtex-7 Kits*. [S.l.]: IEEE, 2019. Citado na página 9.

OBALLE-PEINADO, O. et al. *FPGA-Based Remote Laboratory for Digital Electronics*. 2020. Citado na página 9.

ORDUÑA, P. et al. An extensible architecture for the integration of remote and virtual laboratories in public learning tools. *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, IEEE, v. 10, n. 4, p. 223–233, 2015. Citado na página 8.

PEÑA, E.; LEGASPI, M. G. Uart: A hardware communication protocol understanding universal asynchronous receiver/transmitter. *Analog Dialogue*, Analog Devices, v. 54, 2020. Citado na página 11.

PEREIRA, R. S. Desenvolvimento da camada três do padrão ieee 1876-2019 para um laboratório remoto de ensino em fpga com suporte à aprendizagem pelo moodle, utilizando learning tools interoperability. *TCC(graduação) - Universidade Federal de Santa Catarina. Campus Araranguá. Engenharia da Computação.*, p. 1–35, 2022. Citado na página 14.

REIS, L. F.; MACARIO, E. Dívida pública e financiamento das universidades federais e da ciência e tecnologia no brasil (2003-2020). *Práxis Educacional*, v. 16, n. 41, p. 20–46, 2020. Citado na página 8.

TERASIC, I. *Terasic - SoC Platform - Cyclone - DE1-SoC Board*. [S.l.], 2024. Disponível em: <https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=205&No=836&PartNo=3#contents>. Citado na página 13.

WOLF, M. *Computers as components: principles of embedded computing system design*. [S.l.]: Elsevier, 2012. Citado na página 11.