



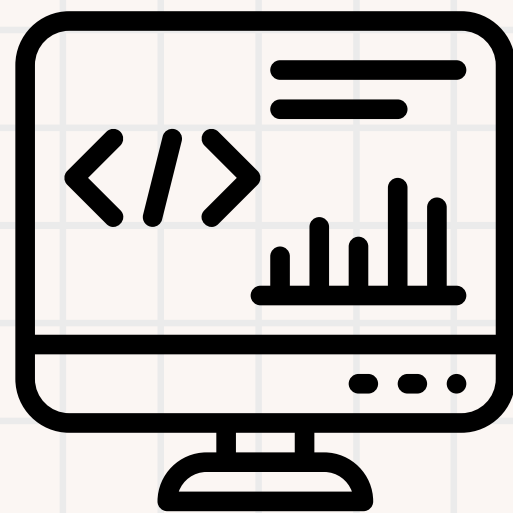
Tutorial

Modulação OFDM no Simulink

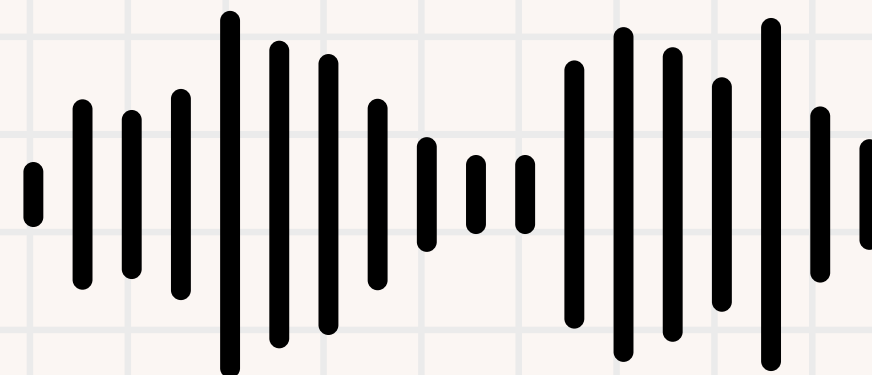
Laboratórios Didáticos para Ensino de Sistemas de
Comunicação

Materiais

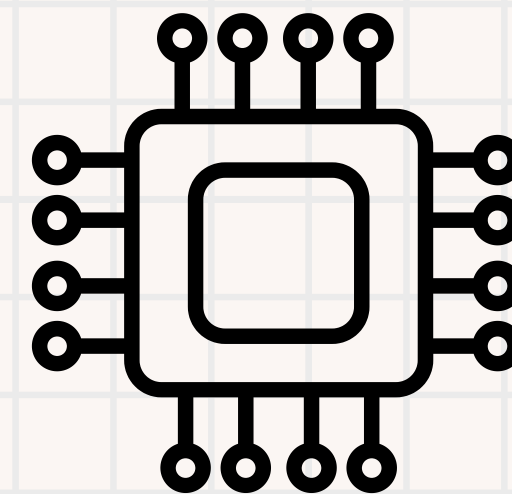
Pré-requisitos




QUARTUS instalado



**Conceito rápido de
modulação OFDM**



FPGA DE2-115



Introdução

Quartus

A implementação em VHDL deste sistema OFDM modula um fluxo de dados de 2 bits (via `qpsk_mapper`) em símbolos transmitidos por subportadoras ortogonais, processados pelas entidades `ifft_core` e `fft_core` (configuradas com $N=64$). O projeto combate a interferência inter-símbolos (ISI) através da inserção de um intervalo de guarda de 16 amostras, gerido pelas entidades `add_cp` (que insere o Prefixo Cíclico) e `remove_cp` (que o remove). Na arquitetura estrutural da entidade `ofdm_system`, o modelo conecta diretamente o transmissor (`ofdm_tx`) ao receptor (`ofdm_rx`), criando um canal ideal para validar a integridade dos sinais I/Q de 16 bits e a recuperação correta dos dados em `bits_out`.

Desenvolvimento

Passo a Passo





Desenvolvimento

Simulink

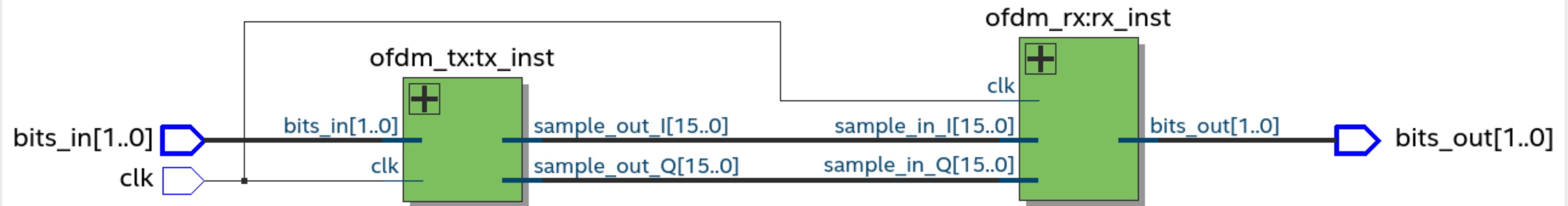
Arquivo do projeto

Para facilitar, é disponibilizado arquivo do modelo em Simulink no Github:



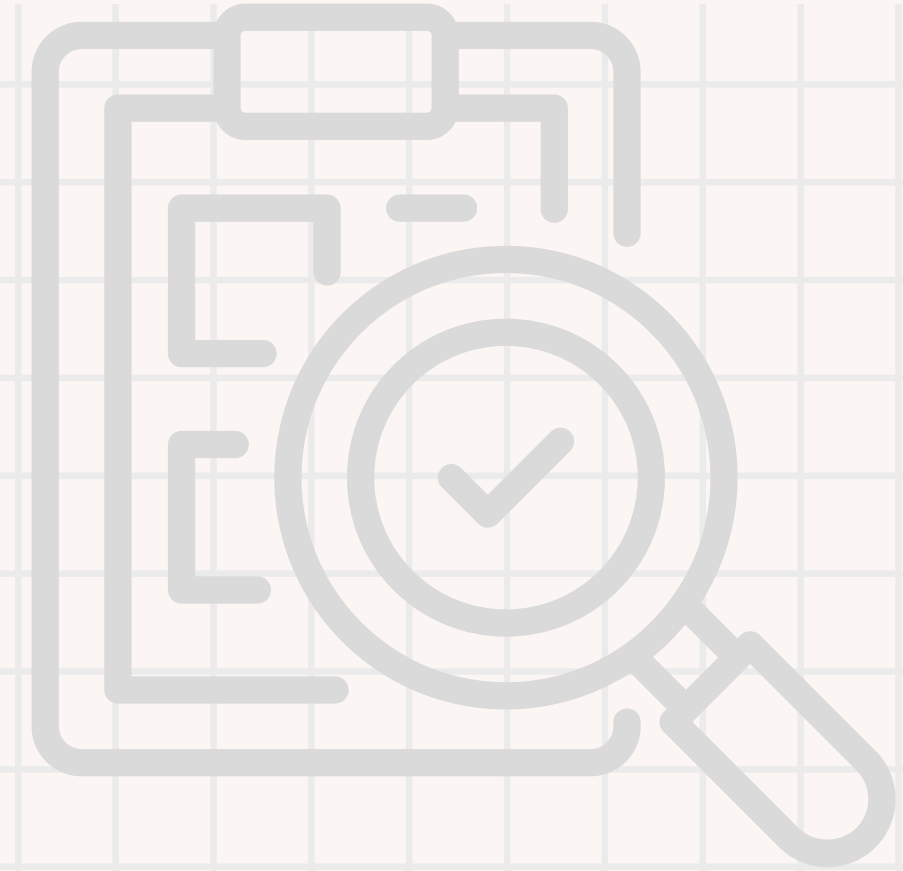
[Link Download](#)

Modulação OFDM - Sistema



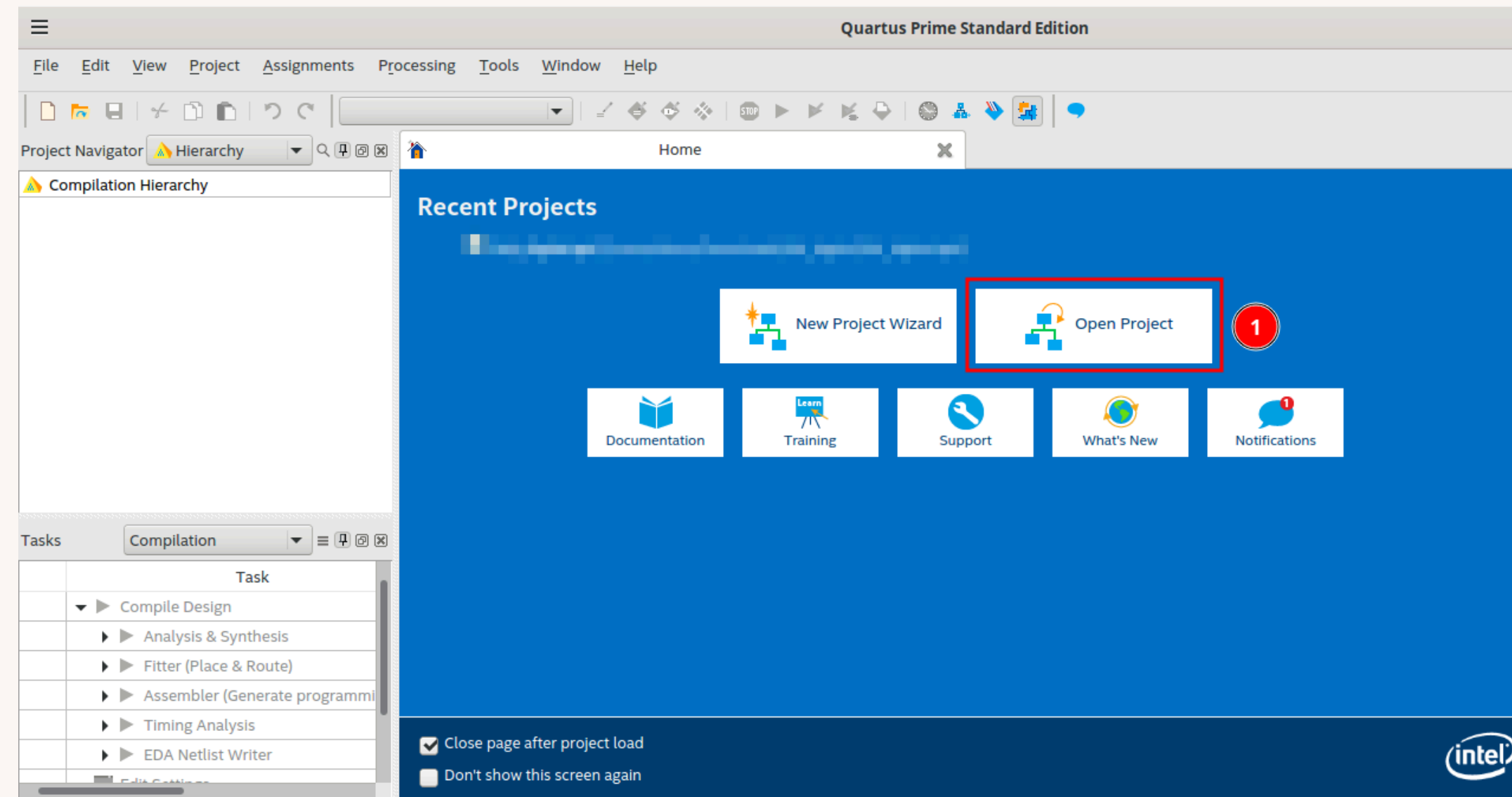
- **Entrada de Dados:** O sistema recebe um fluxo de 2 bits (`bits_in[1..0]`) sincronizados por um clock.
- **Transmissor:** O bloco do transmissor processa esses bits (mapeamento, IFFT, inserção de prefixo cíclico) e gera dois sinais de saída de 16 bits, que representam o sinal modulado no domínio do tempo.
- **Receptor** O bloco do receptor recebe as amostras de 16 bits, realiza o processo inverso (remoção de CP, FFT, desmapeamento) e recupera os dados originais em `bits_out[1..0]`.

Desenvolvimento



A seguir a demonstração de como usar o QUARTUS...

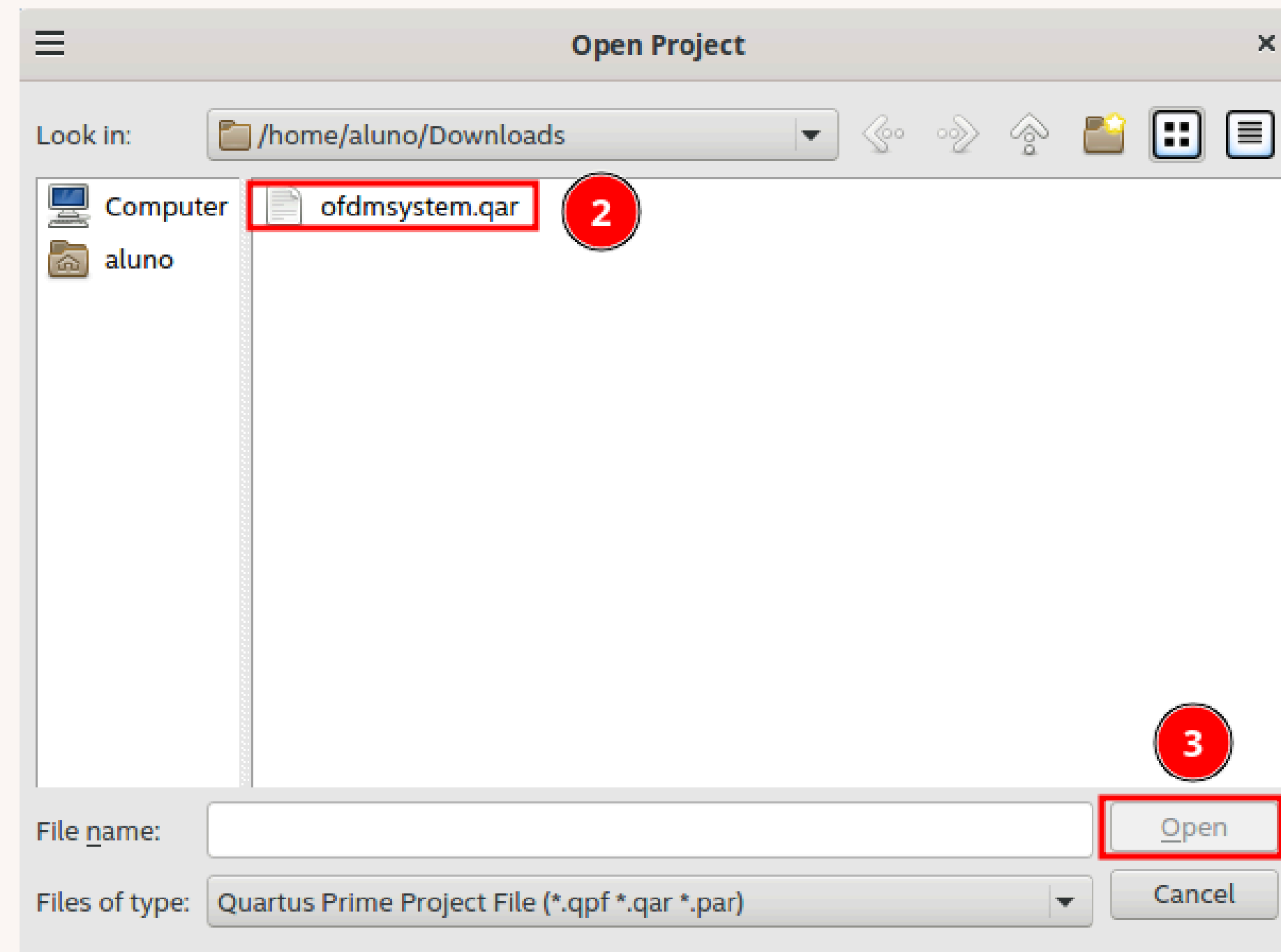
Desenvolvimento



Abra o Quartus

1 - Clique em **Open Project**

Desenvolvimento

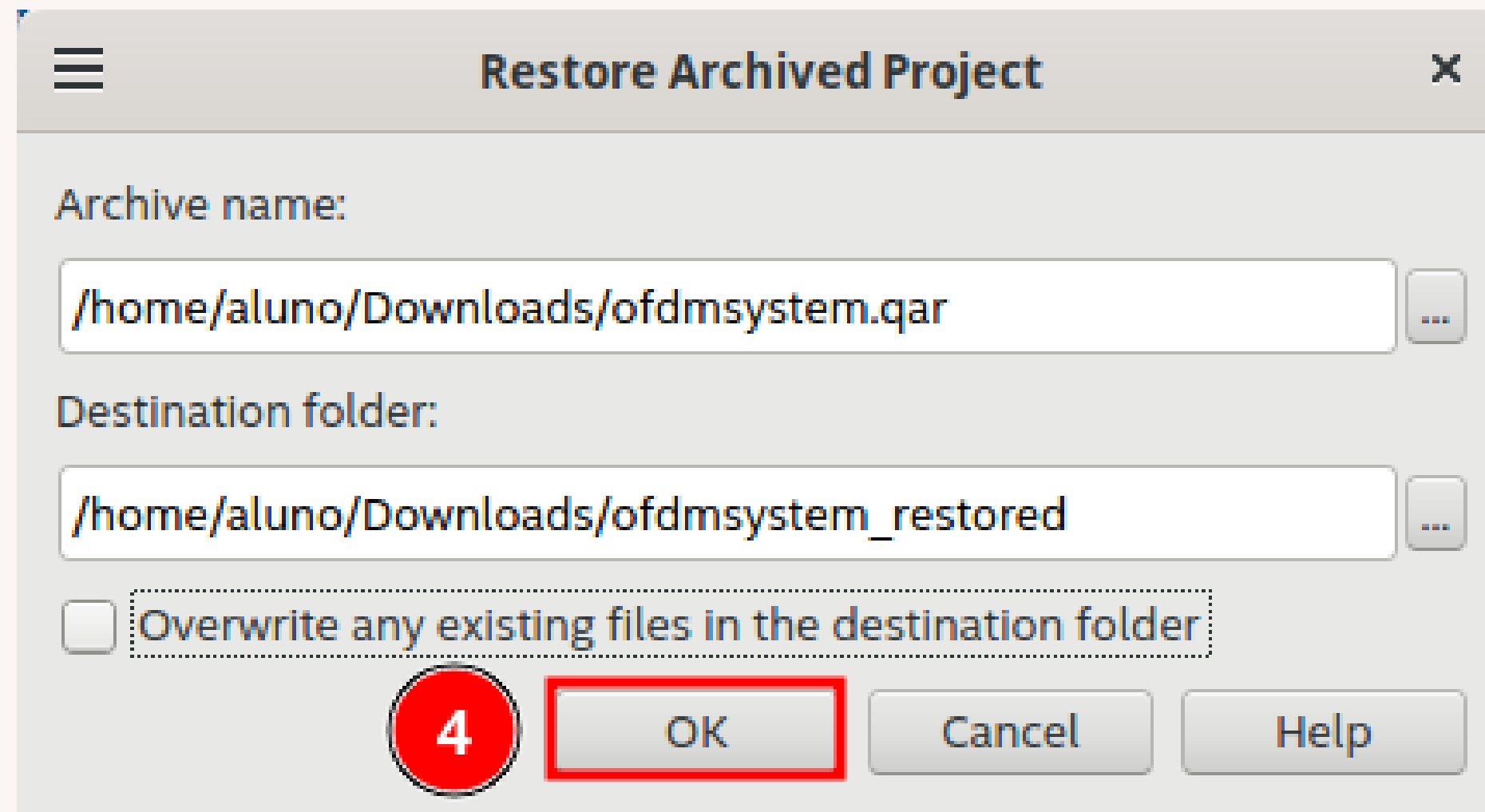


Vá até ao diretório onde foi salvo o arquivo **ofdmsystem.qar**

2 - Clique no arquivo

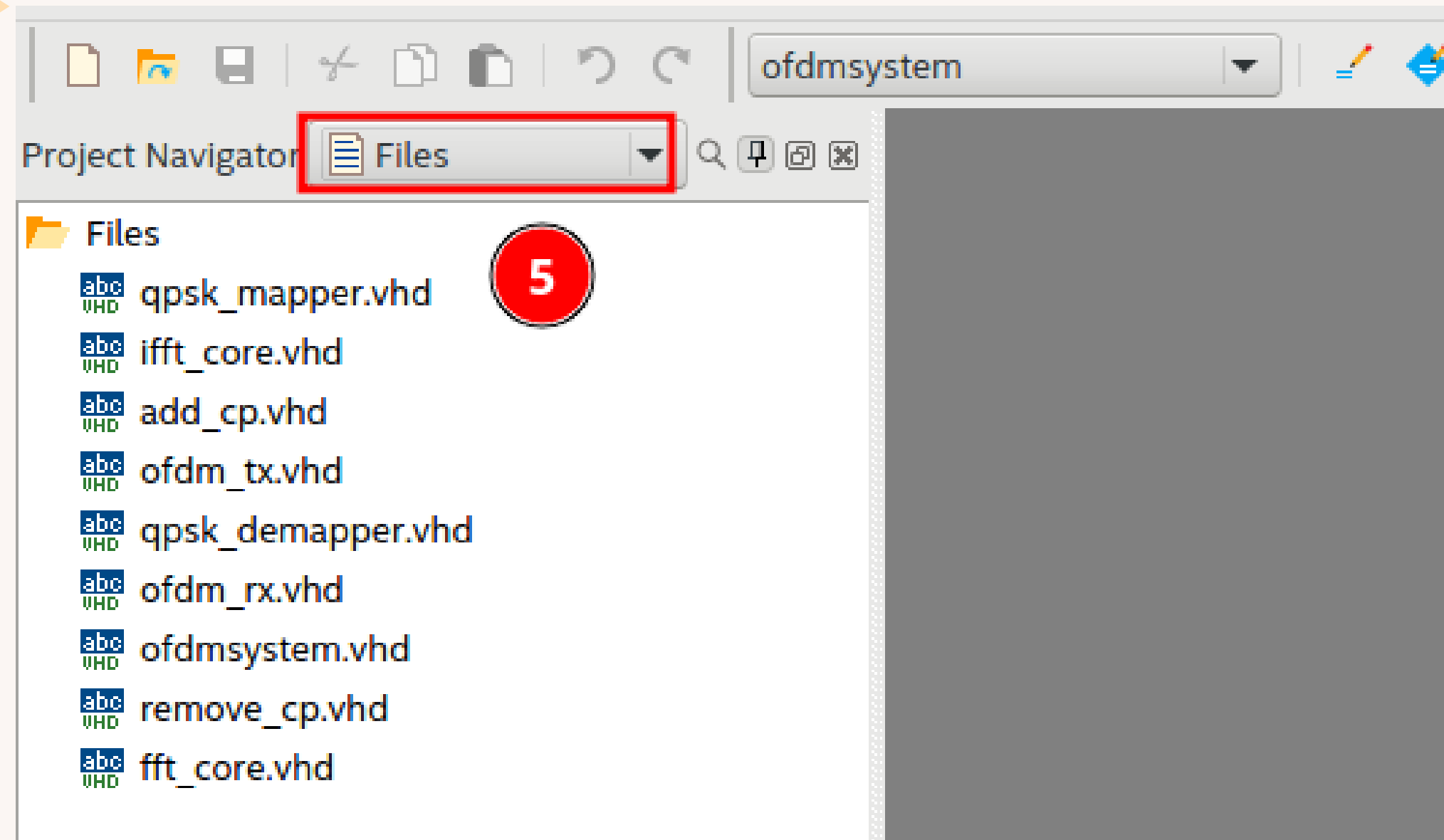
3 - Clique em Open para abri-lo

Desenvolvimento



4 - Clique em Ok
E arquivo da modulação OFDM será aberto.

Desenvolvimento



Vá em Project Navigator

5 - Selecione **Files** Com isso, você consegue visualizar todos os códigos



Entidade ofdmsystem

Desenvolvimento

A entidade **ofdmsystem** atua como o módulo principal (Top-Level) do projeto. Sua função primordial é integrar as instâncias do transmissor e do receptor em um único ambiente de teste, estabelecendo um ciclo fechado de comunicação para validar a integridade funcional do sistema OFDM.

Interface (Portas) O módulo apresenta uma interface externa simplificada, abstraindo a complexidade interna:

- **clk**: O sinal de relógio que sincroniza todo o sistema.
- **bits_in** (2 bits): Entrada de dados digitais para o transmissor. A largura de 2 bits é dimensionada para atender à modulação QPSK utilizada.
- **bits_out** (2 bits): A saída de dados recuperados pelo receptor.

A entidade conecta a saída do transmissor diretamente à entrada do receptor através de barramentos de 16 bits (ponto fixo). Esta configuração cria um ambiente de loopback digital (canal ideal), permitindo verificar se os bits inseridos na entrada são recuperados com fidelidade na saída, validando assim a lógica dos algoritmos de processamento de sinal implementados (FFT/IFFT e Prefixo Cíclico).



Desenvolvimento

Entidade Transmissora ofdm_tx

O módulo **ofdm_tx** é o responsável pela geração do sinal no domínio do tempo. Sua arquitetura processa o fluxo de dados em três etapas sequenciais:

1. **Mapeamento QPSK:** Converte a entrada de 2 bits em símbolos complexos (coordenadas I e Q).
2. **Modulação (IFFT):** Aplica a Transformada Rápida de Fourier Inversa de 64 pontos para levar os símbolos da frequência para o tempo.
3. **Prefixo Cíclico:** Insere uma cópia das últimas 16 amostras no início do símbolo, criando um intervalo de guarda contra interferência inter-símbolos (ISI).

A saída do bloco fornece as amostras digitais I e Q (16 bits) prontas para serem enviadas ao canal.



Desenvolvimento

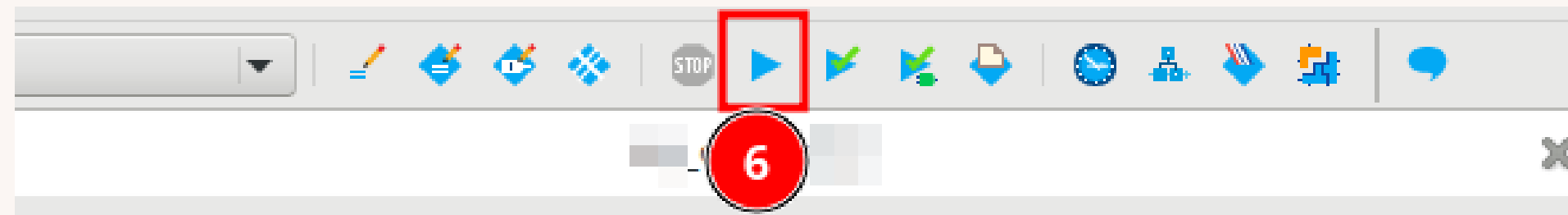
Entidade Receptora ofdm_rx

O módulo **ofdm_rx** realiza o processo inverso ao do transmissor, recuperando os dados originais a partir das amostras recebidas. O fluxo de processamento segue três etapas:

1. **Remoção do Prefixo Cíclico:** O bloco descarta as primeiras 16 amostras de cada símbolo (o intervalo de guarda), isolando apenas a parte útil do sinal para processamento.
2. **Demodulação (FFT):** Aplica a Transformada Rápida de Fourier (FFT) para converter o sinal do domínio do tempo de volta para o domínio da frequência, recuperando as subportadoras.
3. **Desmapeamento QPSK:** Analisa os sinais complexos resultantes (I e Q). Com base no sinal (positivo ou negativo) de cada componente, o sistema decide quais foram os bits transmitidos, reconstruindo o vetor de saída de 2 bits.

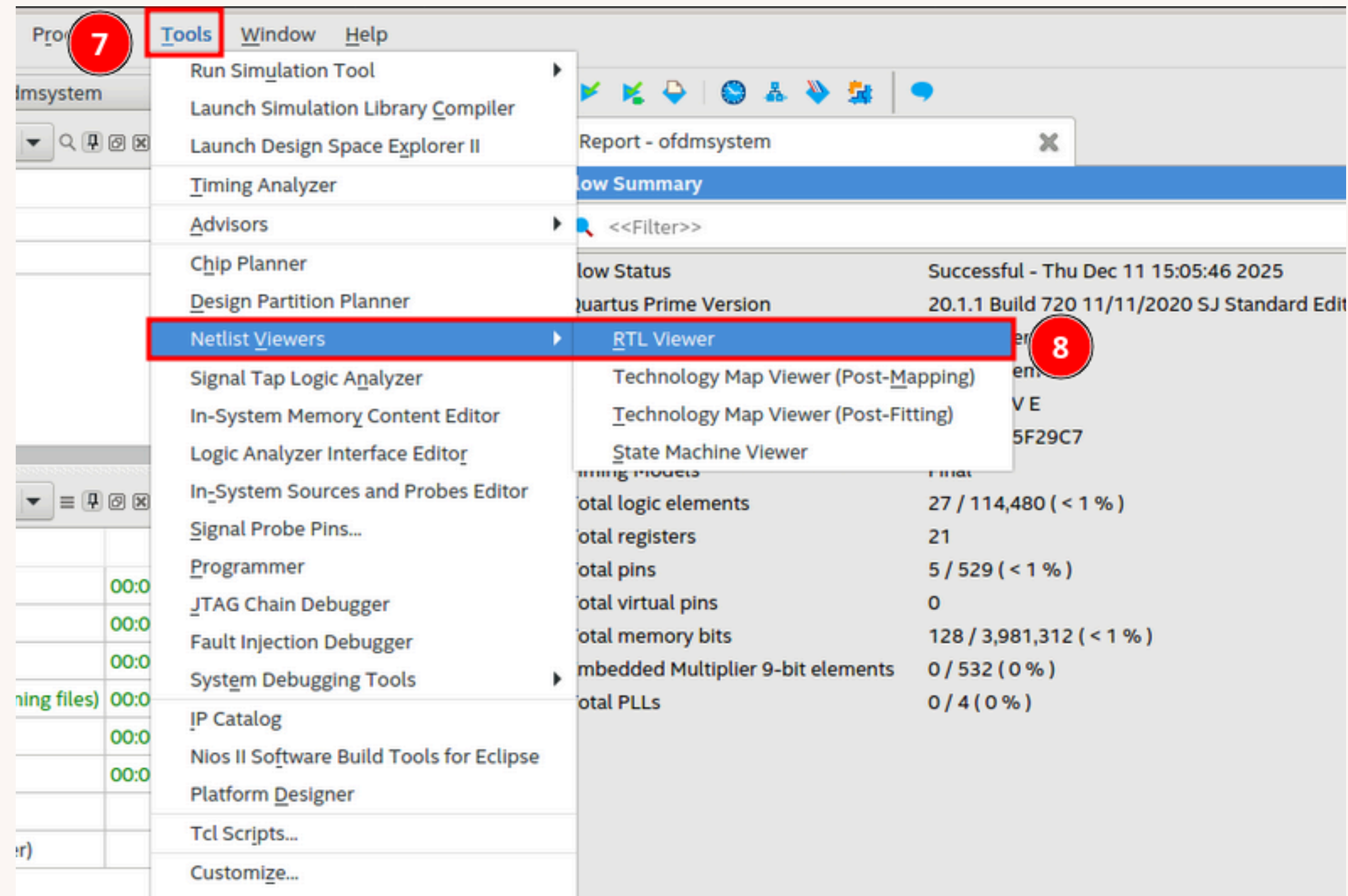
A saída final deste bloco (bits_out) deve ser idêntica à entrada original do transmissor, confirmando o sucesso da comunicação.

Desenvolvimento



6 - Clique no botão RUN para realizar a compilação completa do projeto

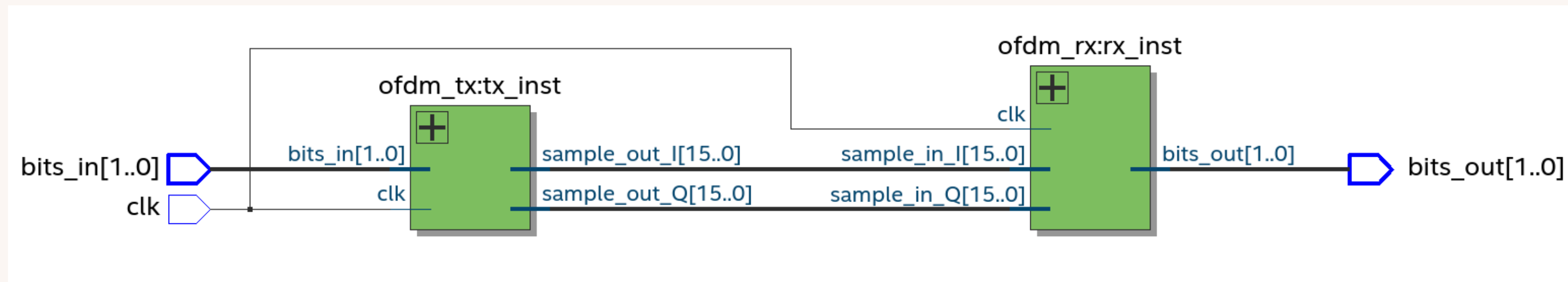
Desenvolvimento



7 - Vá em Tools

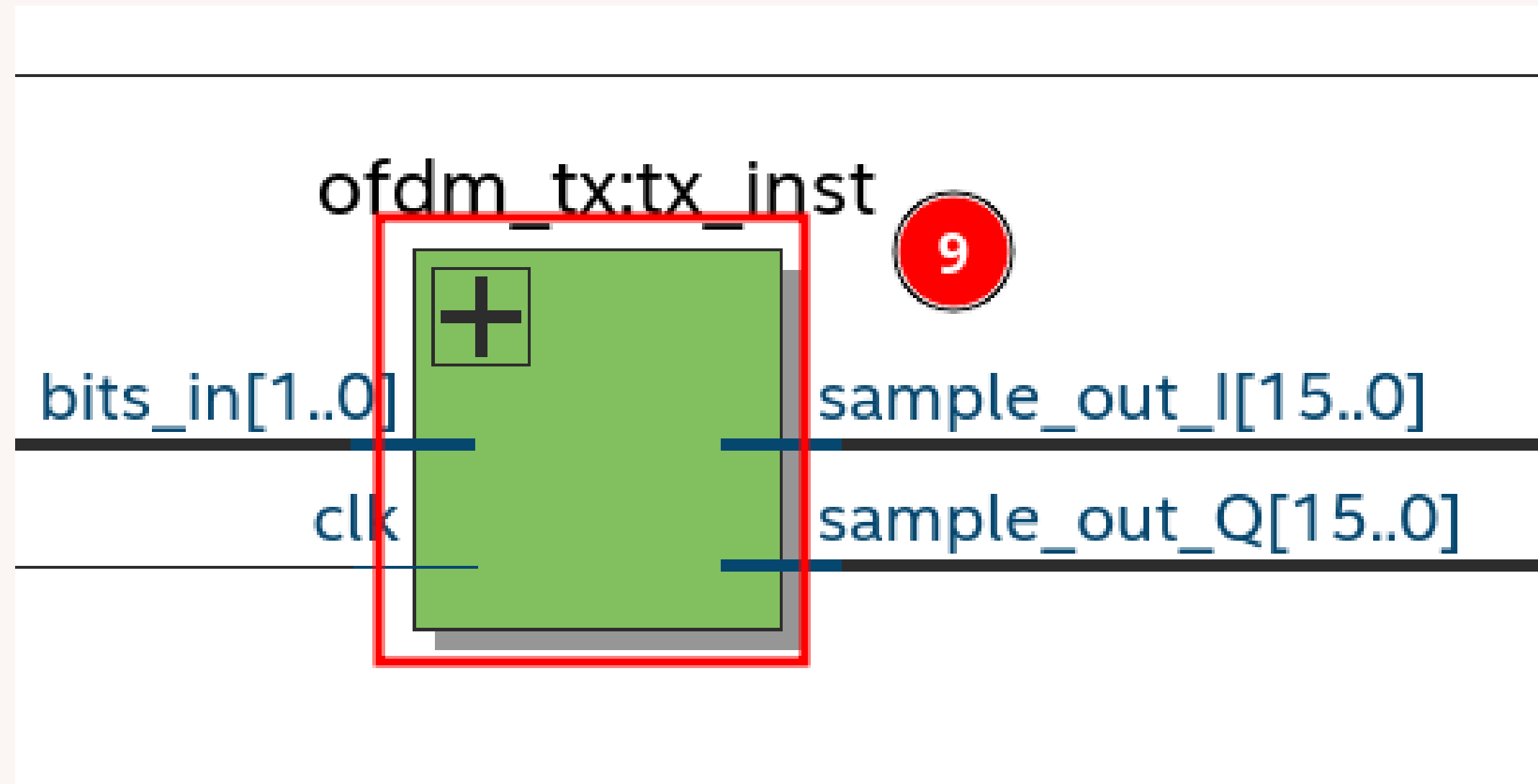
8 - Cliente em: Netlist Viewrs → RTL Viewer

Desenvolvimento



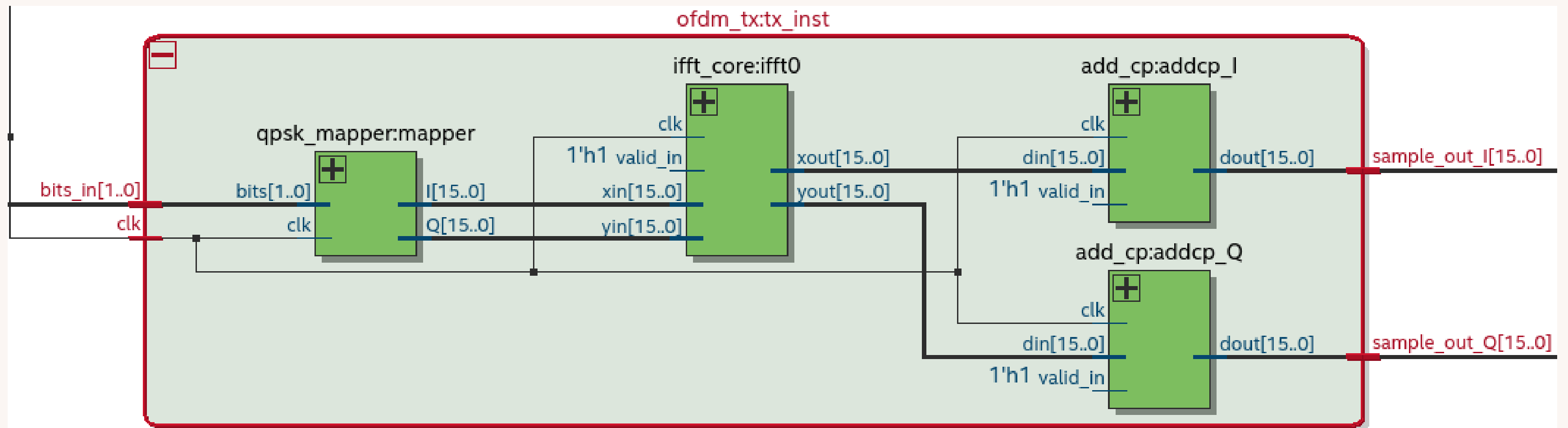
O diagrama mostra a validação lógica do sistema em "loopback". O transmissor (`ofdm_tx`) gera o sinal OFDM a partir dos dados de entrada, que é passado diretamente (sem ruído) para o receptor (`ofdm_rx`) através de uma conexão ideal. O objetivo é provar que os dados que entram à esquerda são corretamente recuperados na saída à direita após todo o processamento de sinal.

Desenvolvimento



9 - Dê um duplo clique no **ofdm_tx**

Desenvolvimento

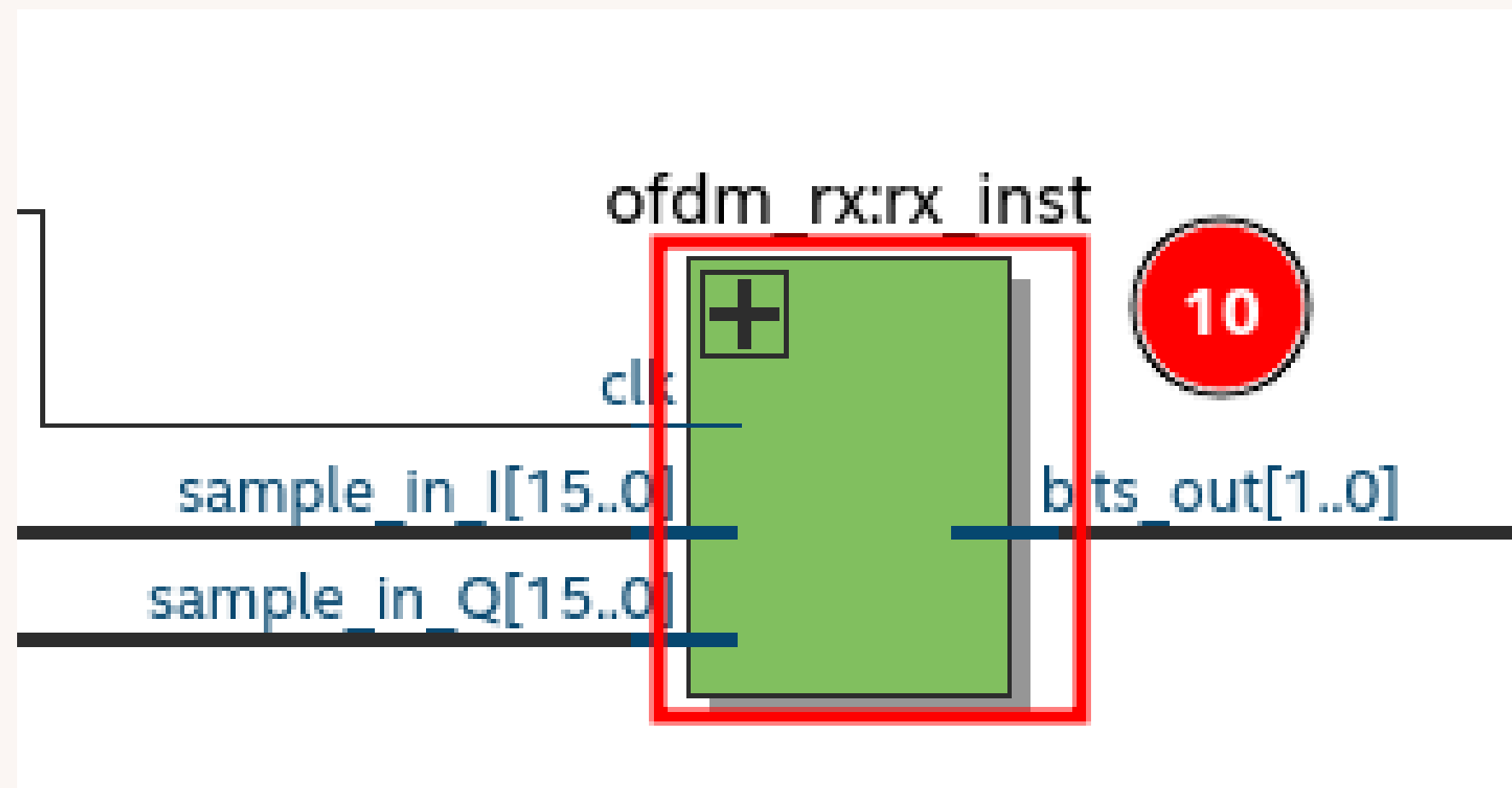




Desenvolvimento

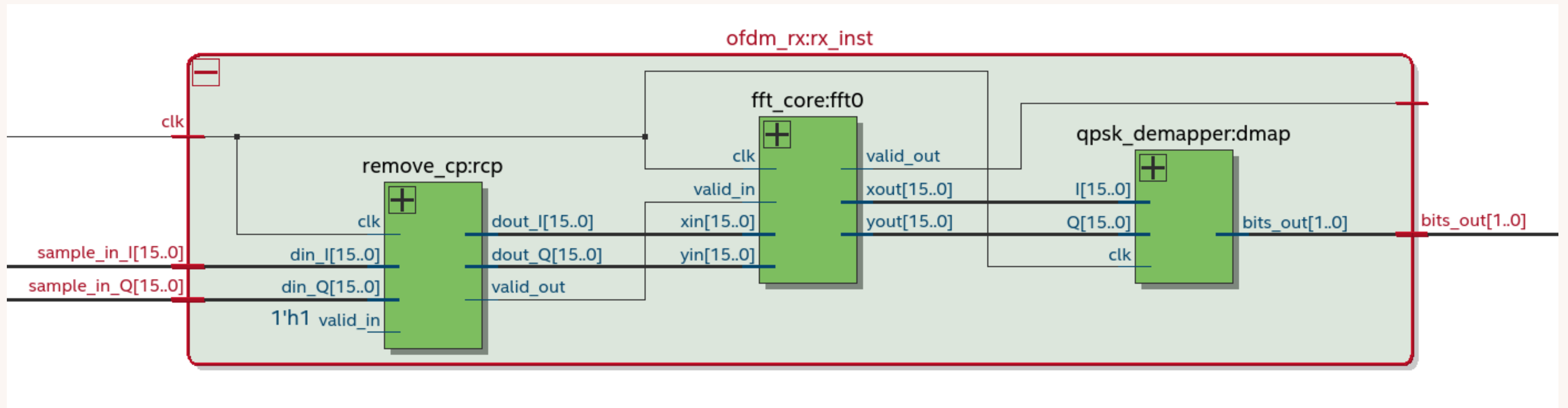
- **Mapeamento (qpsk_mapper):** O bloco recebe a entrada `bits[1..Q]` e a converte em dois sinais de 16 bits: I (In-phase) e Q (Quadrature). Isso transforma os dados digitais em pontos de uma constelação QPSK.
- **Modulação (ifft_core):** Os sinais I e Q entram no bloco da IFFT. Aqui, o sinal é convertido do domínio da frequência (subportadoras) para o domínio do tempo. Observe que as saídas deste bloco são `xout` e `yout`.
- **Inserção de Prefixo Cíclico (add_cp):** O sinal no tempo é dividido em dois caminhos paralelos. Existem dois blocos idênticos:
 - **addcp_I:** Processa a parte real do sinal.
 - **addcp_Q:** Processa a parte imaginária. Ambos pegam o final da sequência de dados e a copiam para o início (o prefixo cíclico), gerando as saídas finais `sample_out_I` e `sample_out_Q` prontas para transmissão.

Desenvolvimento



10 - Dê um duplo clique no **ofdm_rx**

Desenvolvimento





Desenvolvimento

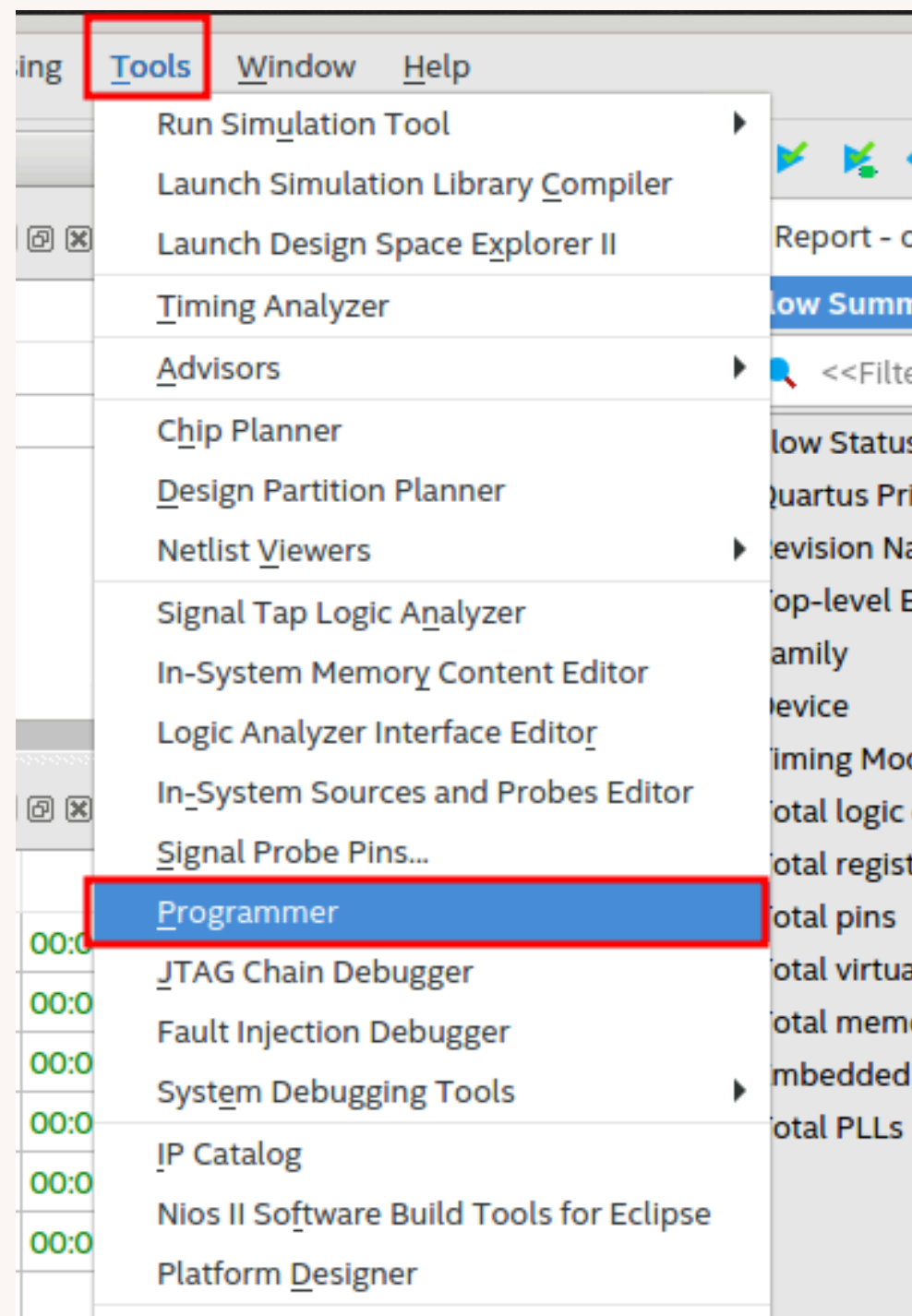
- **Remoção do Prefixo Cíclico (remove_cp):** O bloco recebe as amostras de entrada (sample_in) e descarta as primeiras 16 amostras de cada símbolo (que eram apenas cópias de proteção), deixando passar apenas a parte útil do sinal para a próxima etapa.
- **Demodulação (fft_core):** O sinal "limpo" entra no núcleo da FFT via xin e yin. Aqui, ocorre a transformação matemática inversa à do transmissor: o sinal volta do domínio do tempo para o domínio da frequência (subportadoras).
- **Desmapeamento (qpsk_demapper):** As saídas da FFT (xout, yout) entram neste bloco final. Ele analisa a polaridade (sinal positivo ou negativo) dos componentes I e Q para decidir se o bit original era 0 ou 1, reconstruindo o vetor de saída bits_out[1..N].

Para transferir o seu projeto de modulador OFDM do software Quartus para a placa DE2-115, siga este roteiro. Este processo é chamado de programação ou gravação da FPGA:

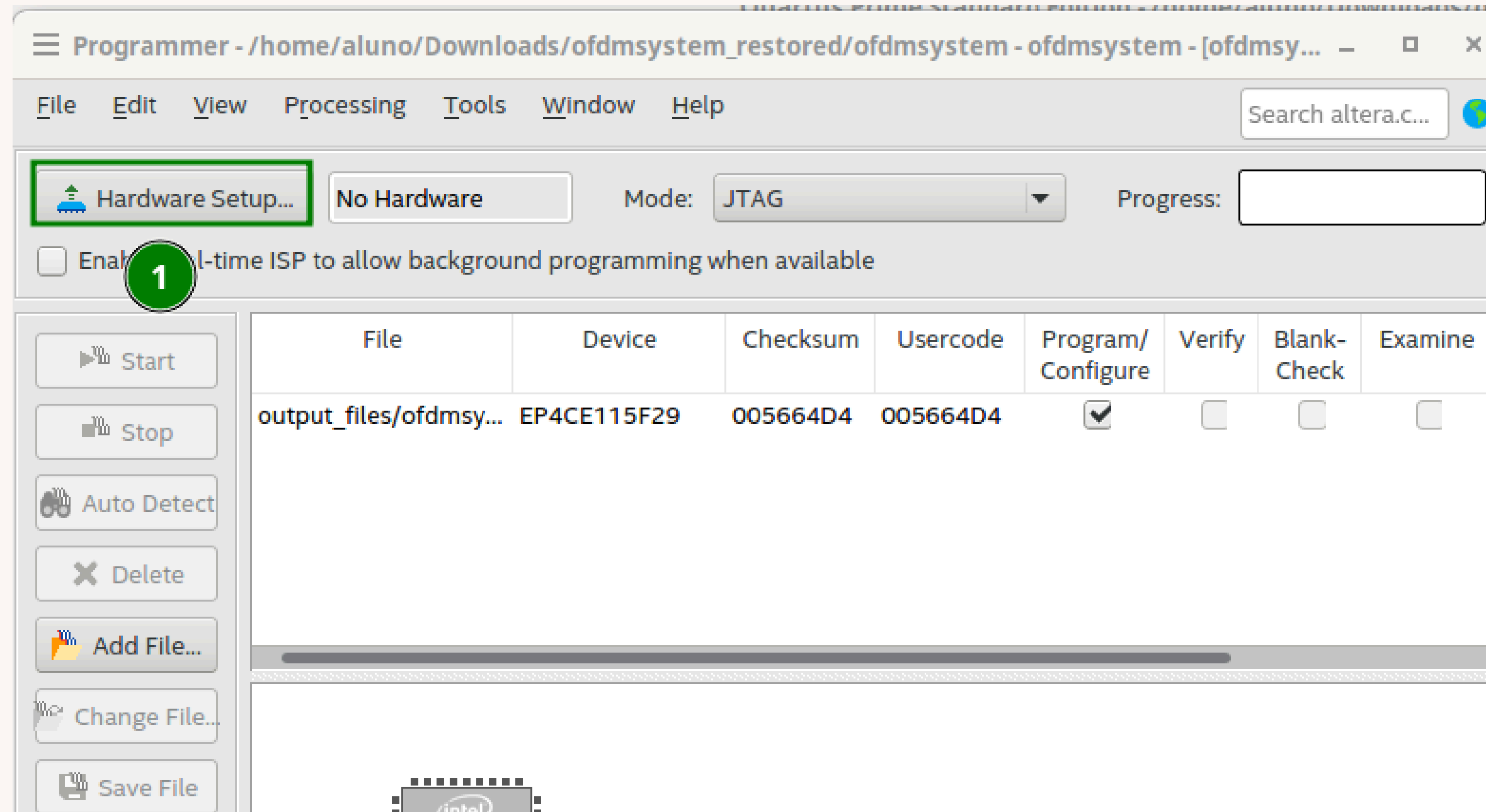
1. **Conexão USB:** Conecte o cabo USB da porta **USB BLASTER** da placa DE2-115 ao seu computador.
2. **Alimentação:** Ligue a fonte de energia da placa e acione o botão de Power.

Desenvolvimento

Gravar Código na Placa



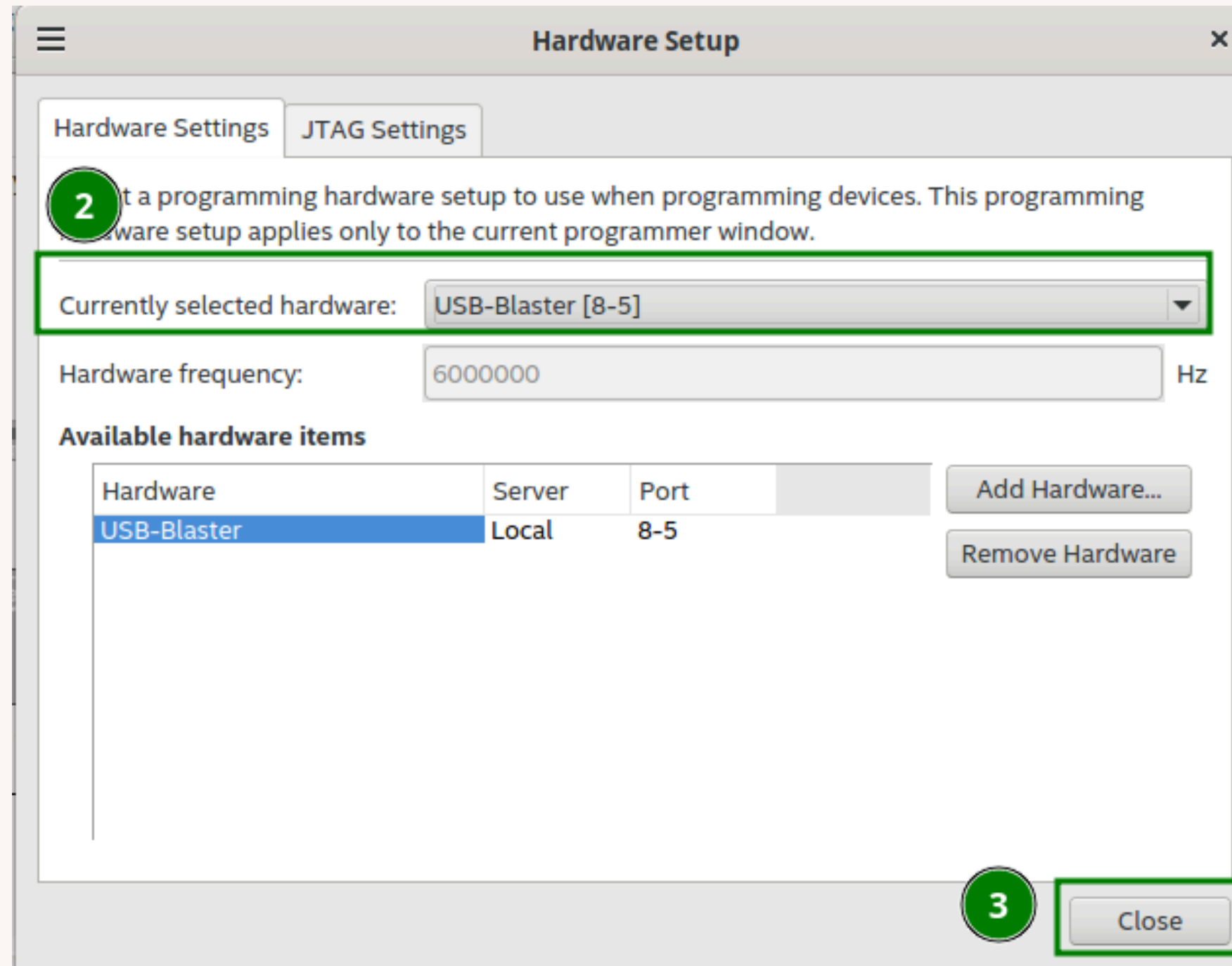
- Vá em **Tools** → **Programmer**



1 - Clique em **Hardware Setup**

Desenvolvimento

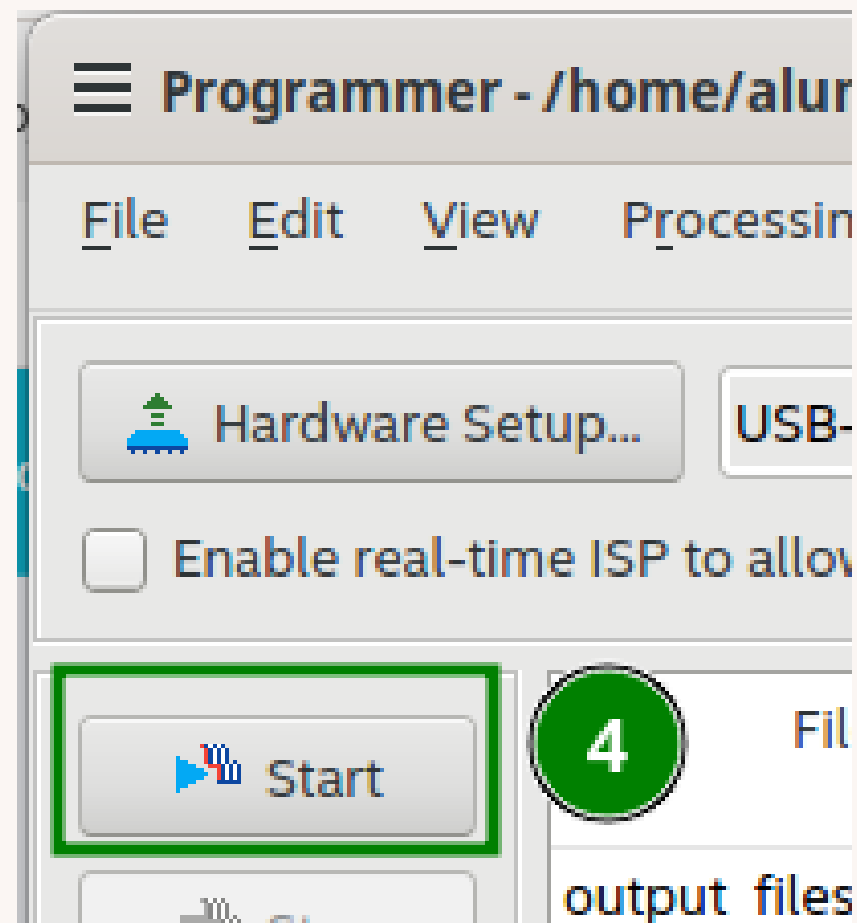
Gravar Código na Placa



Em Currently selected hardware:
2 - Selecione **USB-Blaster**

3 - Clique em **Close**

Gravar Código na Placa



4 - Cliente em **Start**

Fazendo esses passos o código será gravado na placa



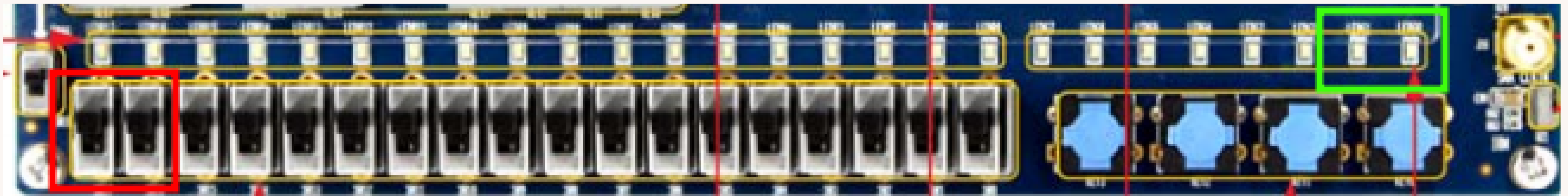
Desenvolvimento

Observação do funcionamento...

Desenvolvimento

Funcionamento

Mapeamento



- **Switch:** Os Switchs 0 e 1 são as entradas do sinal

- **LED GREEN:** Os Leds verdes 0 e 1 são as saídas do sinal

Entrada de Dados do Sistema OFDM:

Neste tutorial, o sistema OFDM foi projetado com uma entrada de 2 bits. Essa escolha está diretamente relacionada ao esquema de modulação utilizado antes do bloco OFDM, que é a QPSK.

Na modulação QPSK, cada símbolo representa 2 bits de informação, pois existem 4 estados possíveis, associados a diferentes fases do sinal. Assim, a cada par de bits de entrada, é gerado um símbolo QPSK que pode ser representado pelas componentes I (em fase) e Q (em quadratura).

Esses símbolos QPSK são então utilizados como entrada para o bloco OFDM, onde são distribuídos entre as subportadoras e combinados por meio da IFFT, formando o sinal OFDM no domínio do tempo.



1. Entrada:

- **Ação:** Posicionamos os switches SW1 e SW⊙ para baixo (nível lógico ⊙).
- **Dado:** Isso envia o vetor binário ⊙⊙ para a entrada do transmissor OFDM.
- **Significado:** Representa a transmissão de um símbolo de valor "zero".

2. O Processamento:

- O sistema opera em loopback: o sinal viaja internamente no FPGA sem perdas.

3. Saída:

- **Resultado:** O receptor entrega o vetor ⊙⊙ na saída.
- **Visualização:** Os LEDs Verdes 1 e ⊙ permanecem apagados.
- **Conclusão:** O fato dos LEDs refletirem exatamente o estado dos switches confirma que o dado percorreu todo o caminho complexo do OFDM sem sofrer erros ou inversões.

Desenvolvimento

Validação Prática em Hardware

O mesmo ocorre nos demais caso:

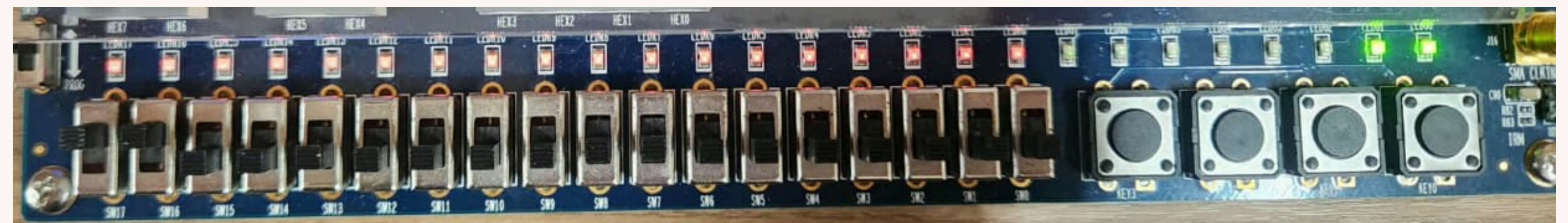
- Caso 01(Transmite o valor 1):



- Caso 10(Transmite o valor 2):



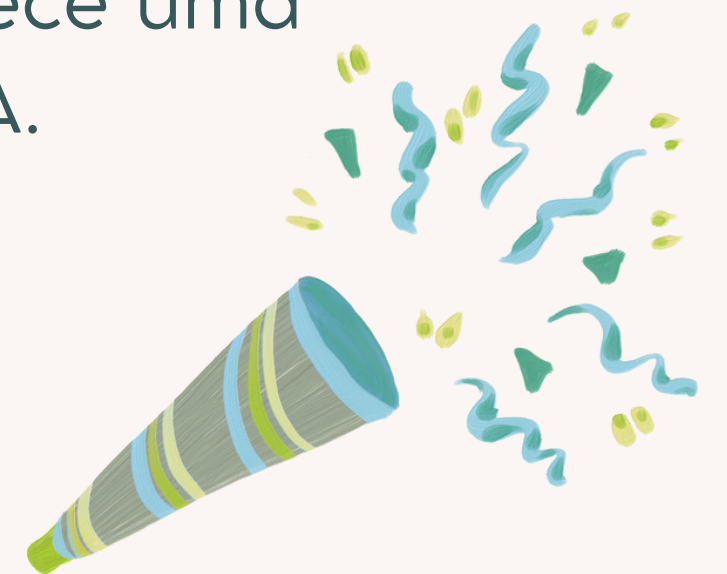
- Caso 11(Transmite o valor 3):



Interpretação dos Resultados

Para concluir, o projeto atingiu seu objetivo principal: implementar um modem OFDM funcional em hardware.

Conseguimos provar que a matemática complexa da IFFT e a lógica de proteção do Prefixo Cíclico estão funcionando em sincronia. O teste de loopback foi decisivo: ao ver os LEDs respondendo corretamente aos comandos dos switches, validamos que o sistema não apenas processa dados, mas preserva a integridade da informação através de todo o ciclo de transmissão e recepção. Isso estabelece uma arquitetura confiável para futuros avanços em comunicação digital na FPGA.





**Muito
obrigado!**