

# Relatório de Testes - Decodificador de Protocolo com FSM e TDD

Nícolas Jordani  
UFSM

2 de setembro de 2025

## 1 Introdução

Este relatório descreve a implementação de um decodificador para um protocolo de comunicação utilizando uma Máquina de Estados Finita (FSM). O protocolo segue o formato:

STX (1B) | QTD (1B) | DADOS (N B) | CHK (1B) | ETX (1B)

A FSM foi projetada de forma modular, utilizando uma tabela de estados para controlar as transições. A metodologia TDD (Test Driven Development) foi aplicada para validar cada etapa do desenvolvimento. Neste trabalho, os testes foram realizados manualmente, através de um programa em C que imprime no terminal o resultado de cada caso de teste.

## 2 Metodologia

O desenvolvimento seguiu os seguintes passos:

1. Definição dos estados e transições da FSM.
2. Implementação das funções em `decoder.c` e `decoder.h`.
3. Criação de um programa de testes (`test_decoder.c`) com `main()` que envia diferentes sequências de bytes para a FSM.
4. Verificação manual da saída impressa no terminal para confirmar o comportamento esperado.

### 3 Implementação

A FSM foi implementada com uma tabela de transições baseada nos estados:

- `WAIT_STX` – espera pelo byte de início (STX).
- `WAIT_QTD` – espera pela quantidade de dados.
- `WAIT_DATA` – recebe os bytes de dados.
- `WAIT_CHK` – espera pelo byte de checksum.
- `WAIT_ETX` – espera pelo byte de término (ETX).
- `DONE` – mensagem válida concluída.
- `ERROR` – erro detectado no fluxo.

O arquivo `decoder.c` contém a implementação da FSM, enquanto o arquivo `test_decoder.c` contém os testes.

### 4 Testes

Foram realizados três testes principais:

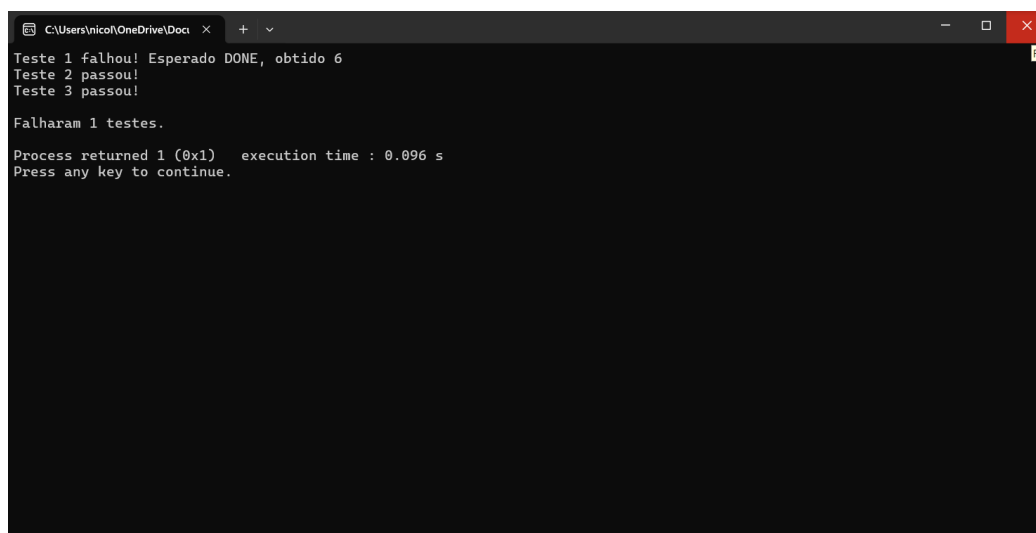
1. **Mensagem válida:** A FSM recebeu uma sequência correta contendo STX, QTD, dados, checksum válido e ETX. O estado final esperado foi `DONE`.
2. **Checksum incorreto:** A FSM recebeu uma sequência com dados válidos, mas checksum errado. O estado final esperado foi `ERROR`.
3. **Mensagem sem ETX:** A FSM recebeu uma sequência sem o byte final ETX. O estado final esperado foi `ERROR`.

### 5 Resultados

Os testes demonstraram que a FSM implementada responde corretamente às condições previstas:

- Reconhece mensagens válidas e finaliza em `DONE`.
- Detecta erros de checksum e ausência de ETX, terminando em `ERROR`.

Isso comprova a conformidade da FSM com as especificações do protocolo.



A screenshot of a Windows command prompt window. The title bar shows the file path 'C:\Users\nico\OneDrive\Doc...' and standard window controls. The command prompt displays the following text: 'Teste 1 falhou! Esperado DONE, obtido 6', 'Teste 2 passou!', 'Teste 3 passou!', 'Falharam 1 testes.', 'Process returned 1 (0x1) execution time : 0.096 s', and 'Press any key to continue.'.

```
C:\Users\nico\OneDrive\Doc...
Teste 1 falhou! Esperado DONE, obtido 6
Teste 2 passou!
Teste 3 passou!

Falharam 1 testes.

Process returned 1 (0x1) execution time : 0.096 s
Press any key to continue.
```

Figura 1: Print dos testes realizados