

# Relatório TDD: FSM com Ponteiros de Função

Projeto de Sistemas Embarcados

João Vítor Sauzem Real

22 de agosto de 2025



**Universidade Federal de Santa Maria**  
Departamento de Eletrônica e Computação

**Professor:** Carlos Henrique Barriquello

# 1 Introdução

Este relatório documenta a implementação de uma Máquina de Estados Finitos (FSM) para um protocolo de comunicação, utilizando a técnica de ponteiros de função em linguagem C. O desenvolvimento foi guiado pela metodologia TDD (*Test-Driven Development*), aproveitando a suíte de testes robusta criada para a versão anterior baseada em `switch-case`.

O objetivo foi refatorar a implementação interna do módulo, adotando uma abordagem mais modular e eficiente, enquanto se garantia que o comportamento externo permanecesse idêntico e correto, validado pelos testes preexistentes.

## 2 Método de Implementação: FSM com Ponteiros de Função

A implementação de uma FSM com ponteiros de função é uma técnica avançada que se afasta da estrutura monolítica de um `switch-case`. A lógica principal é a seguinte:

- **Cada estado é uma função:** A lógica para cada estado da FSM (ex: aguardar STX, receber dados, verificar checksum) é encapsulada na sua própria função C.
- **O estado é um ponteiro:** A estrutura da FSM (`struct`) não armazena um número ou `enum` para o estado atual, mas sim um **ponteiro para a função** que representa o estado atual.
- **Execução direta:** A função principal de processamento (ex: `protocolo_rx_processar_byte`) torna-se extremamente simples. Ela apenas chama a função para a qual o ponteiro de estado atual aponta.
- **Transição de estado:** Mudar de estado consiste em fazer o ponteiro de estado apontar para a função do próximo estado.

Esta abordagem resulta num código mais modular, mais legível e, frequentemente, mais rápido, pois elimina a necessidade de percorrer as opções de um `switch`, saltando diretamente para a lógica correta.

### 2.1 Exemplo de Implementação

A estrutura do receptor armazena o ponteiro para o estado atual:

```
1 typedef struct {  
2     // ... outras variaveis de estado ...  
3     rx_action_t estado_atual; // Ponteiro para a funcao do estado  
4 } protocolo_rx_t;
```

Listing 1: Struct da FSM do Receptor

A função de processamento principal simplesmente delega a tarefa:

```
1 void protocolo_rx_processar_byte(protocolo_rx_t *rx, uint8_t
   byte_recebido) {
2     rx->estado_atual(rx, byte_recebido);
3 }
```

Listing 2: Função principal de processamento

Cada estado é uma função, como por exemplo o estado que aguarda o checksum:

```
1 static void rx_estado_chk(void* fsm, uint8_t byte_recebido) {
2     protocolo_rx_t* rx = (protocolo_rx_t*)fsm;
3     // ... logica de validacao ...
4     if (/* checksum valido */) {
5         // Transicao para o proximo estado
6         rx->estado_atual = rx_estado_etx;
7     } else {
8         // Transicao para o estado de erro/inicial
9         protocolo_rx_iniciar(rx);
10    }
11 }
```

Listing 3: Função do estado RX\_CHK

### 3 Suíte de Testes TDD

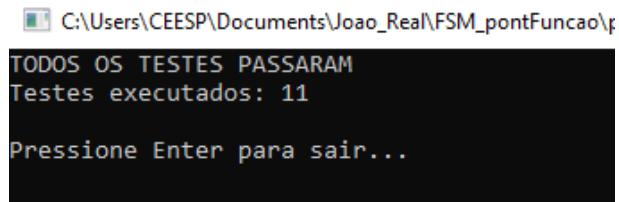
A mesma suíte de 11 testes criada para a versão `switch-case` foi utilizada para validar esta nova implementação. A capacidade de reutilizar os testes sem qualquer alteração demonstra um dos maiores benefícios do TDD: a separação entre o comportamento (o que o código faz) e a implementação (como ele faz).

Os testes garantiram a cobertura dos seguintes cenários:

- **Testes do Transmissor (3):** Verificação do estado inicial, da geração correta de um pacote completo e da geração de um pacote com zero dados.
- **Testes do Receptor (8):**
  - **Caminho Feliz:** Processamento de um pacote válido com e sem dados.
  - **Robustez:** Capacidade de ignorar lixo antes de um pacote.
  - **Tratamento de Erros:** Rejeição de pacotes com checksum inválido, com tamanho maior que o buffer, sem o byte final ETX, ou que são interrompidos a meio por um novo pacote.

### 4 Resultados e Conclusão

A execução da suíte de testes sobre a nova implementação com ponteiros de função produziu o seguinte resultado:

A screenshot of a Windows terminal window. The title bar shows the path 'C:\Users\CEESP\Documents\Joao\_Real\FSM\_pontFuncao\p'. The terminal output is in Portuguese and shows the results of a test suite. The text is as follows:

```
TODOS OS TESTES PASSARAM  
Testes executados: 11  
  
Pressione Enter para sair...
```

Figura 1: Saída no terminal após os testes.

O sucesso de todos os testes confirma que a refatoração foi bem-sucedida. A nova implementação com ponteiros de função não só é funcionalmente idêntica à versão com `switch-case`, como também apresenta um design de software superior em termos de modularidade, eficiência e manutenibilidade.

Este exercício demonstrou eficazmente como o TDD permite realizar grandes alterações na arquitetura interna de um sistema com um alto grau de confiança, garantindo que nenhuma regressão de comportamento seja introduzida.