

# Especificação do Projeto: Máquina de Vendas

## Grupo 2

Junho de 2025

### Sumário

<b>1</b>	<b>Introdução</b>	<b>3</b>
<b>2</b>	<b>Funcionamento Geral do Sistema</b>	<b>3</b>
2.1	Interação Usuário - Máquina . . . . .	3
2.2	Interação com Operador (Reposição) . . . . .	3
2.3	Diagrama da FSM . . . . .	4
<b>3</b>	<b>Estados da FSM</b>	<b>4</b>
3.1	Wait . . . . .	4
3.2	Seleção_C (Compra) . . . . .	4
3.3	Seleção_R (Reposição) . . . . .	5
3.4	Entrega . . . . .	5
<b>4</b>	<b>Entradas do Sistema</b>	<b>5</b>
<b>5</b>	<b>Saídas do Sistema</b>	<b>5</b>
<b>6</b>	<b>Registradores Internos</b>	<b>6</b>

## **7 Observações Finais**

**6**

# 1 Introdução

Este documento descreve a especificação detalhada de uma Máquina de Vendas Automática implementada em VHDL, utilizando uma abordagem de Máquina de Estados Finitos (FSM) em nível RTL. O projeto tem como objetivo controlar a lógica de seleção, pagamento, entrega e reposição de produtos.

## 2 Funcionamento Geral do Sistema

A máquina de vendas opera com quatro produtos inicialmente, aceitando apenas pagamento em dinheiro (sem troco). A entrega só ocorre após o pagamento completo. Há dois modos de operação:

- **Modo Cliente:** Compra e seleção de produtos.
- **Modo Operador:** Reposição dos produtos no sistema.

### 2.1 Interação Usuário - Máquina

1. O cliente pressiona o botão `COMPRA`.
2. Seleciona o produto desejado com `product.select.buy`.
3. Visualiza preço e quantidade (`price_display` e `quantity_display`).
4. Insere o dinheiro progressivamente (`money_in`).
5. Quando o total inserido cobre o valor, pressiona `PAG`.
6. A FSM ativa o motor e entrega o produto.

### 2.2 Interação com Operador (Reposição)

1. O operador pressiona `REP`.
2. Seleciona o produto e insere a quantidade desejada para repor.
3. Pressiona novamente `REP` para confirmar ou `ESQ` para sair.



### 3.3 Seleção\_R (Reposição)

Permite que o operador selecione um produto e defina a quantidade a ser reposta.

- REP para confirmar nova quantidade.
- ESQ para retornar ao estado inicial.

### 3.4 Entrega

Entrega o produto e ativa o sinal `motor_enable` por um ciclo. Retorna ao estado `Wait`.

## 4 Entradas do Sistema

Sinal	Tipo	Tamanho	Descrição
clk	std_logic	1 bit	Clock de 1 Hz
reset	std_logic	1 bit	Reset síncrono da FSM
money_in	integer	8 bits	Valor em dinheiro inserido
product_select_buy	std_logic_vector	3 bits	Seleção de produto (cliente)
product_select_replenish	std_logic_vector	3 bits	Seleção de produto (operador)
replenish_quantity	integer	8 bits	Quantidade a ser adicionada
COMPRA	std_logic	1 bit	Inicia processo de compra
SELECT_C	std_logic	1 bit	Confirma seleção
PAG	std_logic	1 bit	Confirma pagamento
REP	std_logic	1 bit	Inicia ou confirma reposição
ESC	std_logic	1 bit	Cancela compra
ESQ	std_logic	1 bit	Sai de qualquer processo atual
QTD	std_logic	1 bit	Indica que há produto disponível
RS	std_logic	1 bit	Indica que a reposição foi bem-sucedida

## 5 Saídas do Sistema

Sinal	Tipo	Tamanho	Descrição
quantity_display	integer	8 bits	Mostra quantidade do produto atual

price_display	integer	8 bits	Mostra o preço do produto atual
motor_enable	std_logic	1 bit	Ativa o motor de entrega

## 6 Registradores Internos

Nome	Tipo	Tamanho	Função
state	enum	2 bits	Estado atual da FSM
produto_sel_comprado	integer	2 bits	Índice de produto selecionado (cliente)
produto_sel_repor	integer	2 bits	Índice de produto selecionado (reposição)
preco[0..3]	integer array	4x8 bits	Vetor de preços
quantidade[0..3]	integer array	4x8 bits	Vetor de quantidades
valor_acumulado	integer	8 bits	Valor inserido pelo cliente
entrega_efetuada	std_logic	1 bit	Indica que a entrega ocorreu

## 7 Observações Finais

- O sistema não devolve troco. Os preços devem ser múltiplos inteiros da menor moeda permitida.
- O clock de 1 Hz simplifica o controle e a contagem de ciclos para ações temporizadas.
- A estrutura trabalha com 4 produtos fixos, definidos em vetores de tamanho 4.