**Projeto Grupo: Máquina de Vendas**

**Grupo 2**

Arthur Henrique Liberato Lima

Levi Mariano Batista

William Dhomini Miranda Martins

Julho 2025

Sumário

[1 - Funcionamento geral do sistema](#_f261ril0d1fj)

[2 - Especificação do projeto](#_5reof36kk3ik)

[3 - FSM](#_7l8z7xpt0bmp)

[4 - Lista de componentes](#_biiavucw8fyy)

[5 - Caminho de Dados](#_1krlmktk4zwh)

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 1 - Funcionamento geral do sistema

A máquina de vendas opera em um ciclo contínuo. O cliente seleciona o produto desejado, e a máquina exibe o preço e a disponibilidade em estoque para verificação. Em seguida, ao inserir uma moeda, o sistema a detecta qual o valor da moeda e o acumula

A transação só é concluída se as condições necessárias forem atendidas. Se o valor inserido for igual ou superior ao preço e o produto estiver disponível, a máquina libera o item para o cliente. Caso contrário, se o crédito for insuficiente ou o produto estiver fora de estoque, a compra não é efetuada.

Além do modo de operação padrão, o sistema possui o modo de reposição, em que um operador pode reabastecer o estoque de produtos. O sistema também inclui funções de reset e escape, que permitem ao operador gerenciar e reiniciar o fluxo das operações conforme necessário.

# 2 - Especificação do projeto

**Entradas**

| **Nome** | **Tipo** | **Tamanho / Faixa** | **Descrição** |
| --- | --- | --- | --- |
| clk | std\_logic | 1 bit | Clock de 1 Hz |
| reset | std\_logic | 1 bit | Reset síncrono |
| COMPRA | std\_logic | 1 bit | Inicia processo de compra |
| REP | std\_logic | 1 bit | Inicia/confirma reposição |
| SELECT\_C | std\_logic | 1 bit | Confirma seleção de produto (cliente) |
| PAG | std\_logic | 1 bit | Confirma pagamento |
| ESC | std\_logic | 1 bit | Cancela compra |
| ESQ | std\_logic | 1 bit | Sai do processo atual |
| product\_select\_buy | std\_logic\_vector(1 downto 0) | 2 bits | Seleção de produto para compra |
| product\_select\_replenish | std\_logic\_vector(1 downto 0) | 2 bits | Seleção de produto para reposição |
| money\_in | integer range 0 to 255 | 8 bits (equiv.) | Valor inserido |
| replenish\_quantity | integer range 0 to 255 | 8 bits (equiv.) | Quantidade a ser adicionada ao estoque |

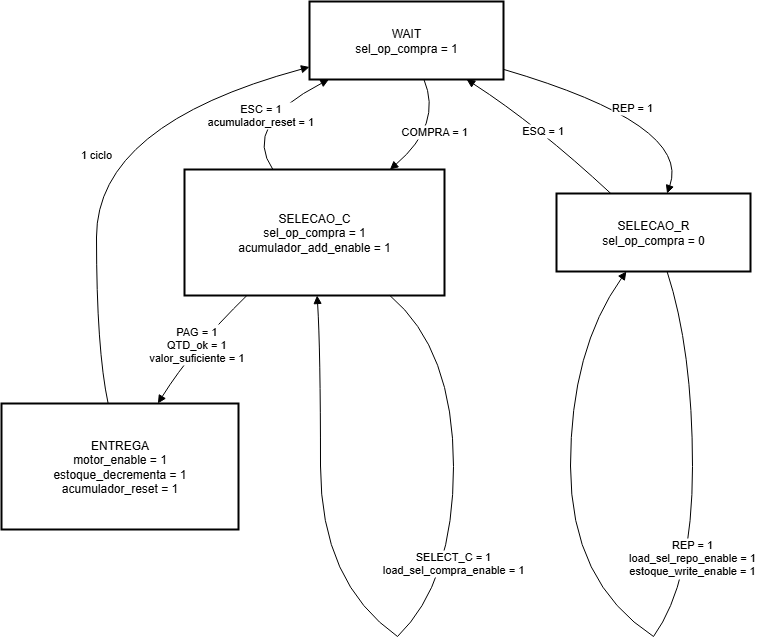
**Saídas**

| **Nome** | **Tipo** | **Tamanho / Faixa** | **Descrição** |
| --- | --- | --- | --- |
| motor\_enable | std\_logic | 1 bit | Ativa motor para entrega |
| price\_display | integer range 0 to 255 | 8 bits (equiv.) | Mostra o preço do produto |
| quantity\_display | integer range 0 to 255 | 8 bits (equiv.) | Mostra a quantidade em estoque |

**Registradores**

| **Nome** | **Tipo** | **Tamanho Faixa** | **Descrição** |
| --- | --- | --- | --- |
| current\_state | state\_type (enum) | variável | Estado atual da FSM principal |
| next\_state | state\_type | variável | Próximo estado da FSM |
| selected\_product | std\_logic\_vector(1 downto 0) | 2 bits | Produto selecionado |
| money\_received | integer range 0 to 255 | 8 bits | Valor de pagamento acumulado |
| product\_quantity | integer | 8 bits (por produto) | Estoque atual de cada produto |
| product\_price | integer | 8 bits (por produto) | Preço de cada produto |

# 3 - FSM



# 4 - Lista de componentes

| **Nome do Componente** | **Entradas** | **Saídas** | **Tipos** | **Tamanhos** | **Função** |
| --- | --- | --- | --- | --- | --- |
| **moedeiro** | habilita\_moedeiro, sinal\_moeda\_25c, sinal\_moeda\_50c, sinal\_moeda\_1real | valor\_moeda | std\_logic, integer | 1 bit (sinais), 8 bits (inteiro 0–255) | Detecta moeda inserida e gera valor correspondente |
| **reg\_2b** | clk, rst, enable, in\_D | out\_Q | std\_logic, std\_logic\_vector | 1 bit (sinais), 2 bits (vetor) | Armazena o código do produto selecionado |
| **memoria\_precos** | endereco | dado\_out | std\_logic\_vector, integer | 2 bits (endereço), 8 bits (preço) | Retorna o preço do produto com base no código |
| **memoria\_estoque** | clk, rst, wr, decrementa, endereco, dado\_in | dado\_out | std\_logic, std\_logic\_vector, integer | 1 bit (sinais), 2 bits (endereço), 8 bits (dados) | Lê e atualiza a quantidade de produtos no estoque |
| **somador** | in\_A, in\_B | out\_S | integer | 8 bits (0–255) | Soma o valor inserido ao saldo acumulado |
| **comparador** | in\_A, in\_B | out\_S | integer, std\_logic | 8 bits (valores), 1 bit (saída lógica) | Compara saldo com o preço para habilitar compra |
| **mux\_2x1\_2b** | sel, in\_A, in\_B | out\_S | std\_logic, std\_logic\_vector | 1 bit (seletor), 2 bits (vetores) | Seleciona entre dois códigos de produto diferentes |
| **subtrator** | in\_A, in\_B | out\_S | integer | 8 bits (0–255) | Calcula novo valor de estoque após venda (estoque - 1) |

# 5 - Caminho de Dados

# 