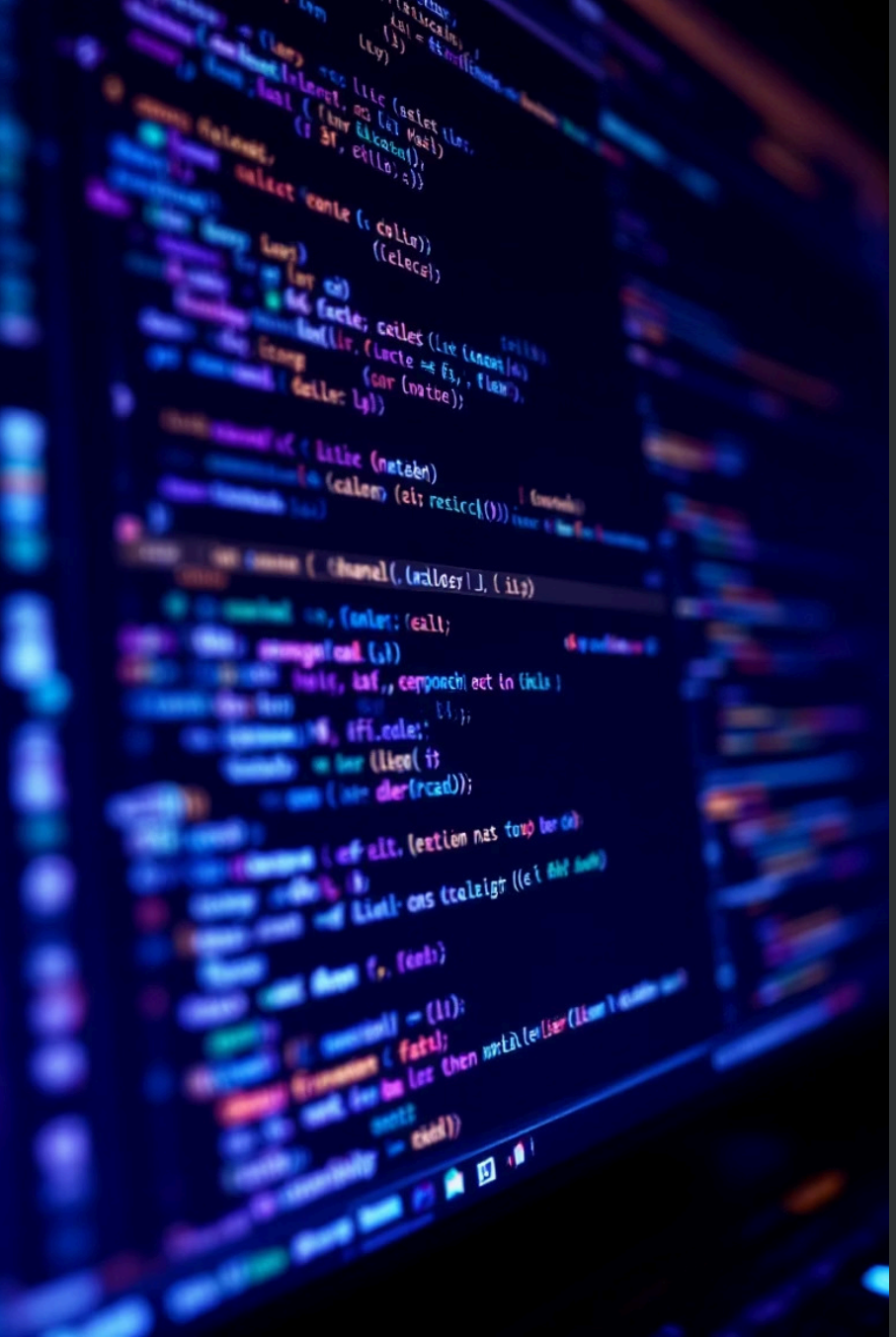


Pivota Programming Language (PPL)

Uma linguagem de domínio específico (DSL) para estratégias de negociação automatizadas com geração de código Verilog.

Objetivo do Projeto

Estrutura da Linguagem

Criar uma linguagem com variáveis, condicionais e loops.

Ferramentas

Utilizar EBNF para estruturar a linguagem e Flex/Bison para análise.

Saída

Gerar código Verilog como alternativa ao LLVM.

Motivação

Mercado Financeiro

O high frequency trading (HFT) depende de regras simples, expressivas e rápidas.

Proposta

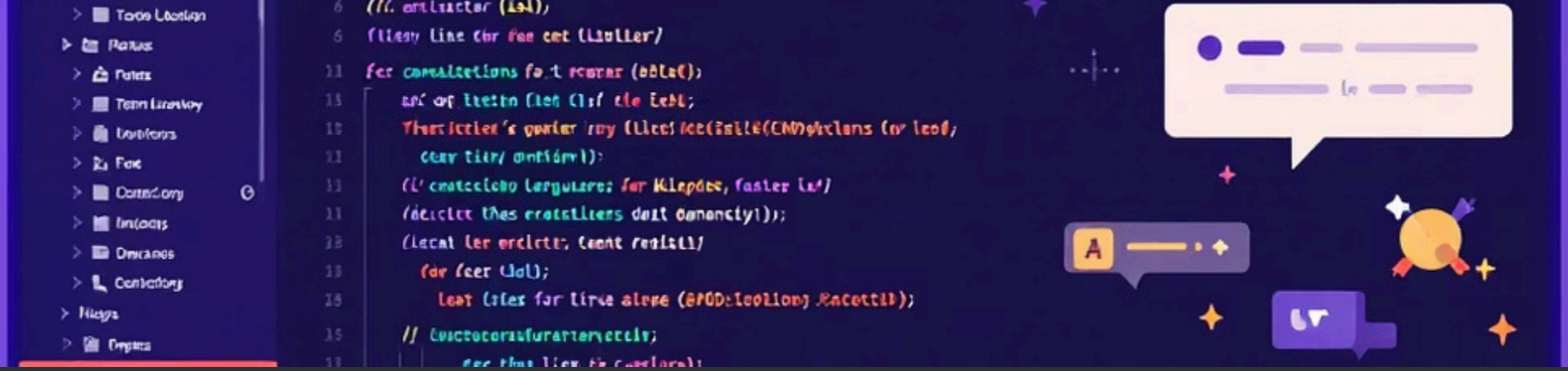
Permitir que programadores descrevam estratégias de trading em linguagem intuitiva.

Validação

Tradução automática gera código Verilog para simulação em plataformas como EDA Playground.

EBNF Final da Linguagem

```
PROGRAM = { STATEMENT } ;

STATEMENT = VAR_DECL | CONDITIONAL | LOOP | EMIT_STMT ;

VAR_DECL = "LET", IDENTIFIER, "=", EXPR, ";" ;

CONDITIONAL = "IF", "(", CONDITION, ")", "{", { STATEMENT }, "}" ;

LOOP = "LOOP", EXPR, "TIMES", "{", { STATEMENT }, "}" ;

EMIT_STMT = "EMIT", ACTION, EXPR, ";" ;

CONDITION = IDENTIFIER, COMPARISON_OPERATOR, EXPR ;

EXPR = TERM, { ("+" | "-"), TERM } ;

TERM = FACTOR, { ("*" | "/"), FACTOR } ;

FACTOR = NUMBER | IDENTIFIER | "(", EXPR, ")" ;

COMPARISON_OPERATOR = ">" | "<" | "==" ;

ACTION = "BUY" | "SELL" ;

IDENTIFIER = LETTER, { LETTER | DIGIT | "_" } ;

NUMBER = DIGIT, { DIGIT } ;

LETTER = "a" | ... | "z" | "A" | ... | "Z" ;

DIGIT = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;
```



Características da Linguagem

Declaração de variáveis

Suporte a variáveis inteiras com palavra-chave LET.

Expressões aritméticas

Operações com +, -, *, / e parênteses.

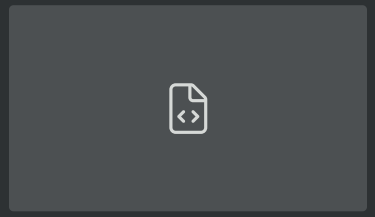
Condicionais

Comparação de variáveis com operadores >, < e ==.

Comandos especiais

EMIT BUY e EMIT SELL traduzem diretamente para Verilog.

Como funciona

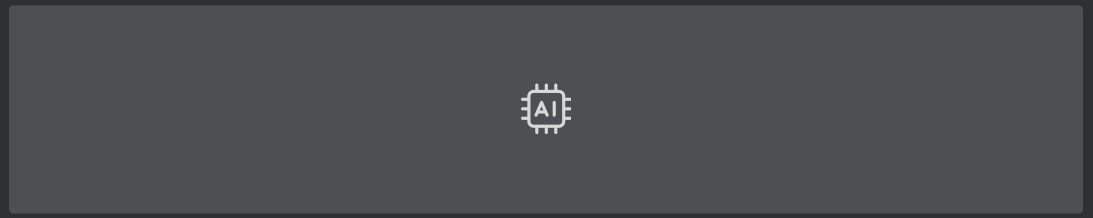


Código de entrada (.pv)

Programador escreve estratégia em Pivota

Processamento

Analisador léxico e sintático processa o código



Código Verilog

Geração automática de módulo Verilog equivalente

Código Gerado

Entrada Pivota

Código com variáveis, condicionais e loops

Tradução

Conversão automática para Verilog

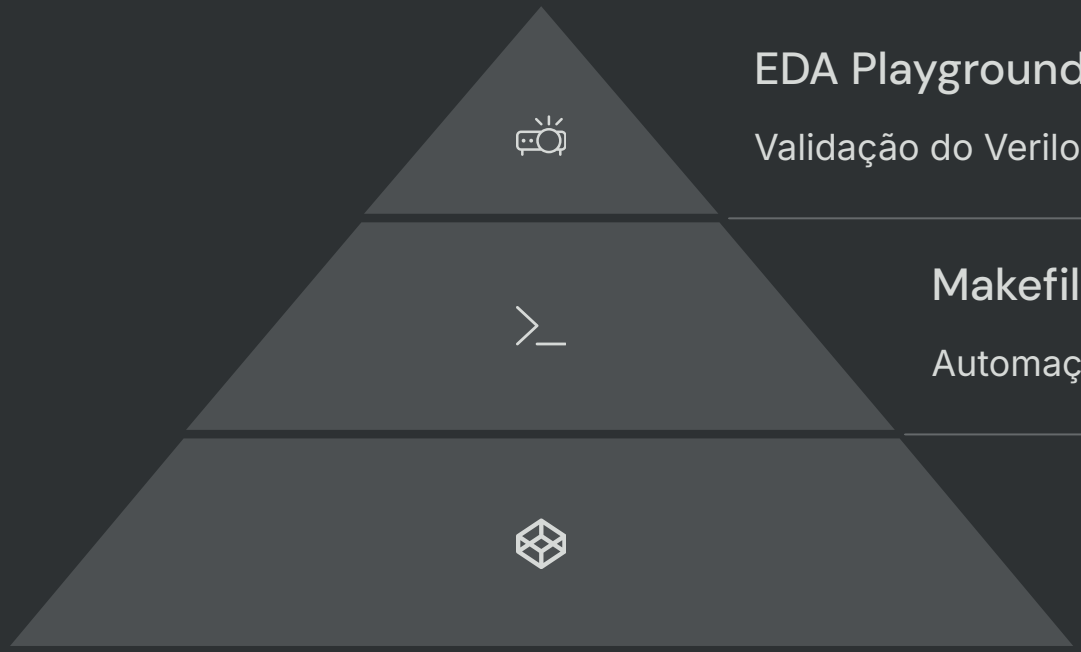
Saída Verilog

Módulo com registradores e lógica equivalente

Validação

Simulação em ambiente EDA

Ferramentas utilizadas



EDA Playground

Validação do Verilog gerado

Makefile

Automação da compilação

Flex e Bison

Analisadores léxico e sintático

Testes no EDA Playground

Interface de Simulação

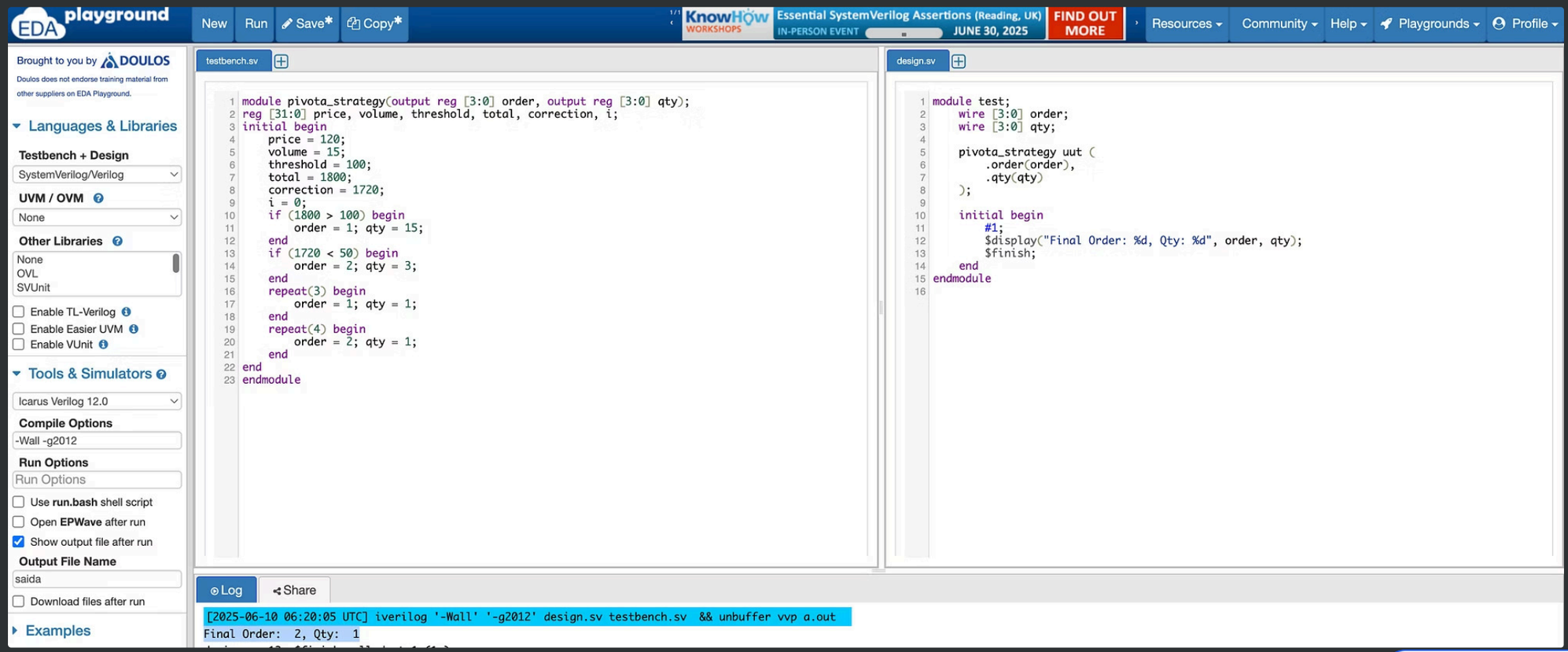
Código da esquerda: módulo Verilog gerado pela nossa linguagem.

Testbench

Código da direita: simula a execução e imprime a última ordem executada.

Resultados

Log mostra os valores finais de order e qty definidos.




```
module pivota_strategy(output reg [3:0] order, output reg [3:0] qty);
reg [31:0] price, volume, threshold, total, correction, i;
initial begin
    price = 120;
    volume = 15;
    threshold = 100;
    total = 1800;
    correction = 1720;
    i = 0;
    if (1800 > 100) begin
        order = 1; qty = 15;
    end
    if (1720 < 50) begin
        order = 2; qty = 3;
    end
    repeat(3) begin
        order = 1; qty = 1;
    end
    repeat(4) begin
        order = 2; qty = 1;
    end
end
endmodule
```

```
module test;
    wire [3:0] order;
    wire [3:0] qty;

    pivota_strategy uut (
        .order(order),
        .qty(qty)
    );

    initial begin
        #1;
        $display("Final Order: %d, Qty: %d", order, qty);
        $finish;
    end
endmodule
```