UNIVERSIDADE DE PERNAMBUCO
ESCOLA POLITÉCNICA DE PERNAMBUCO
PROJETO MODELO PARA O CURSO DE AUTOMAÇÃO DE PROJETOS DE CIRCUITOS INTEGRADOS

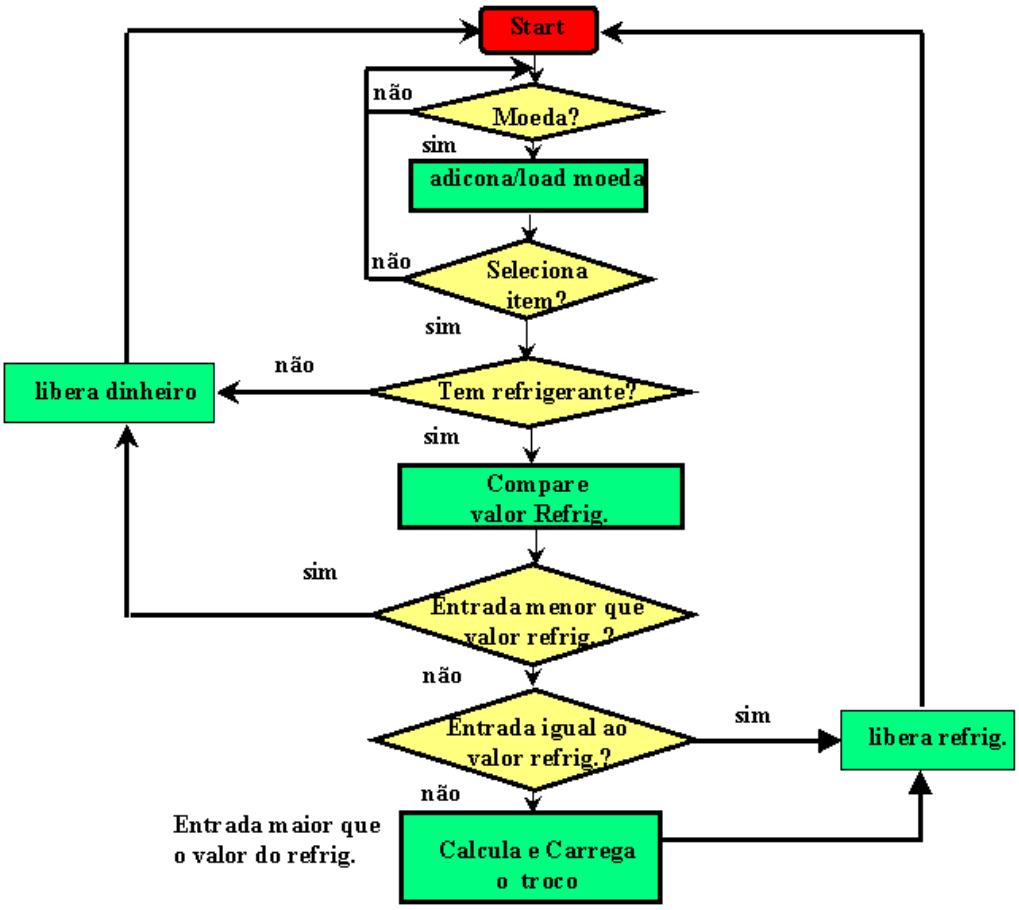
PROJETO DE UMA MÁQUINA DE VENDER REFRIGERANTES

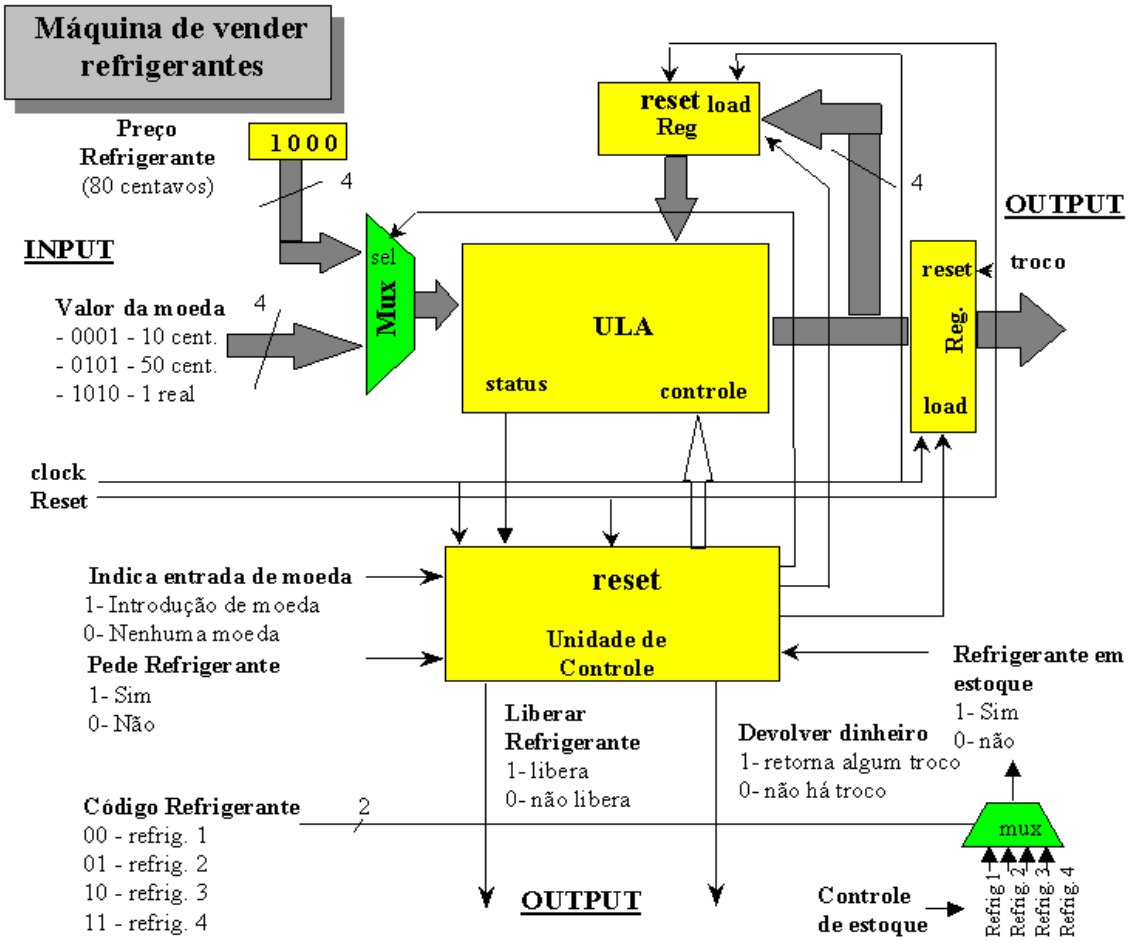
Elaborado por:

JENER TOSCANO LINS E SILVA

Recife, novembro de 2008.

FUNCIONALIDADE DO PROJETO

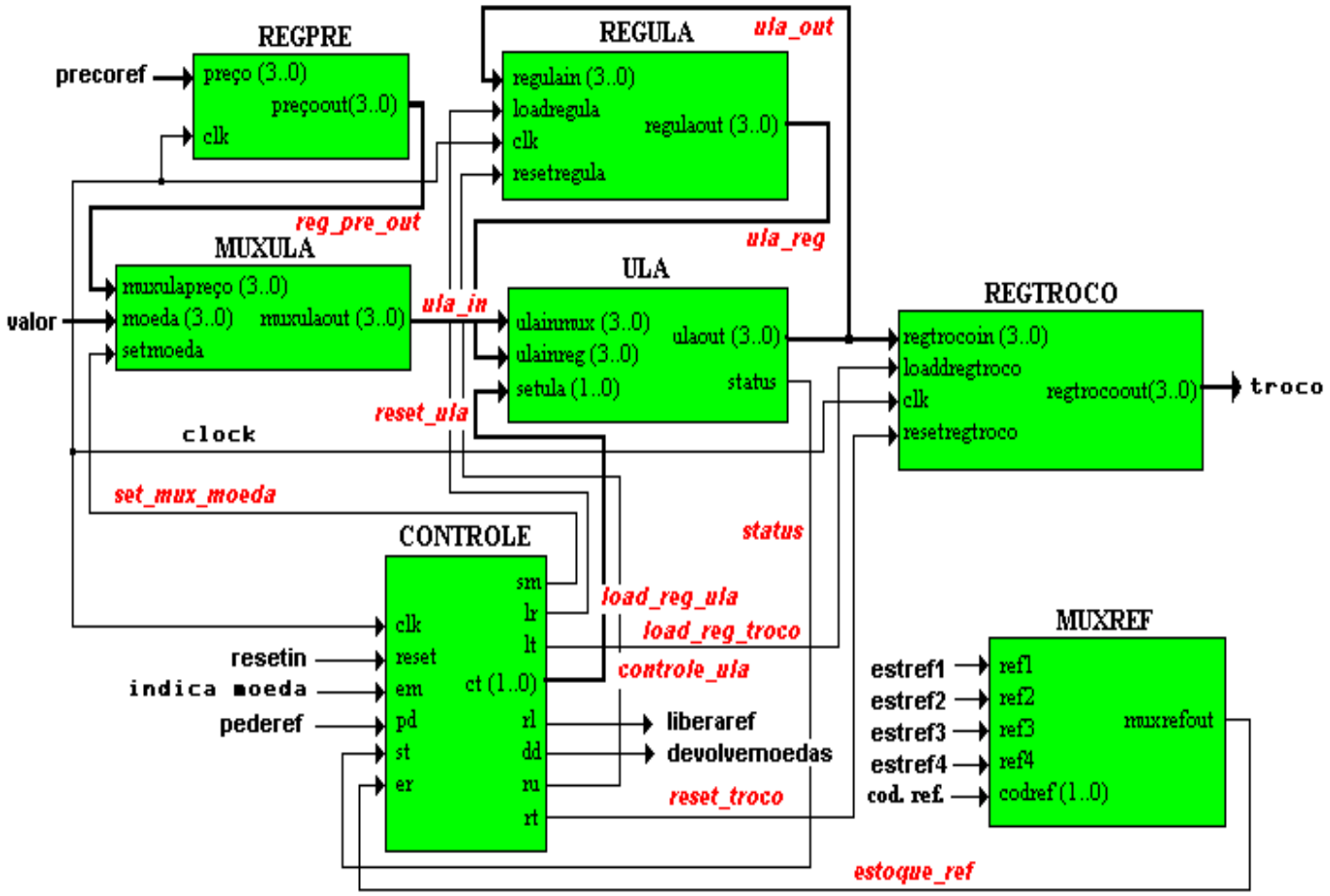




Foi implementado um circuito controlador de uma máquina de vender refrigerante em VHDL, utilizando a plataforma MAXPLUS II da Altera. A máquina trabalha com quatro tipos de refrigerante de mesmo preço (R$ 0,80) no qual aceita moedas de R$ 0,10; R$ 0,50 e R$ 1,00. Um sinal de entrada indica a entrada de moedas, enquanto a outra entrada indica a solicitação do tipo de refrigerante, o qual deve ser previamente escolhido através da entrada de seu código. Embora as entradas de estoque dos refrigerantes sejam fornecidas por um circuito que não depende da máquina, esta será considerada entrada acessível pelo usuário.

A máquina esta preparada para devolução do troco, caso o valor da(s) moeda(s) exceda o valor de R$ 0,80 (preço do refrigerante), ou caso valor seja inferior ao seu preço. Para isso foram implementados os sinais de saída: “libera troco”, “troco” e “libera refrigerante”.

O projeto baseado na hierarquia foi dividido em módulos de modo a facilitar a implementação. O diagrama em blocos a seguir mostra estes aspectos:

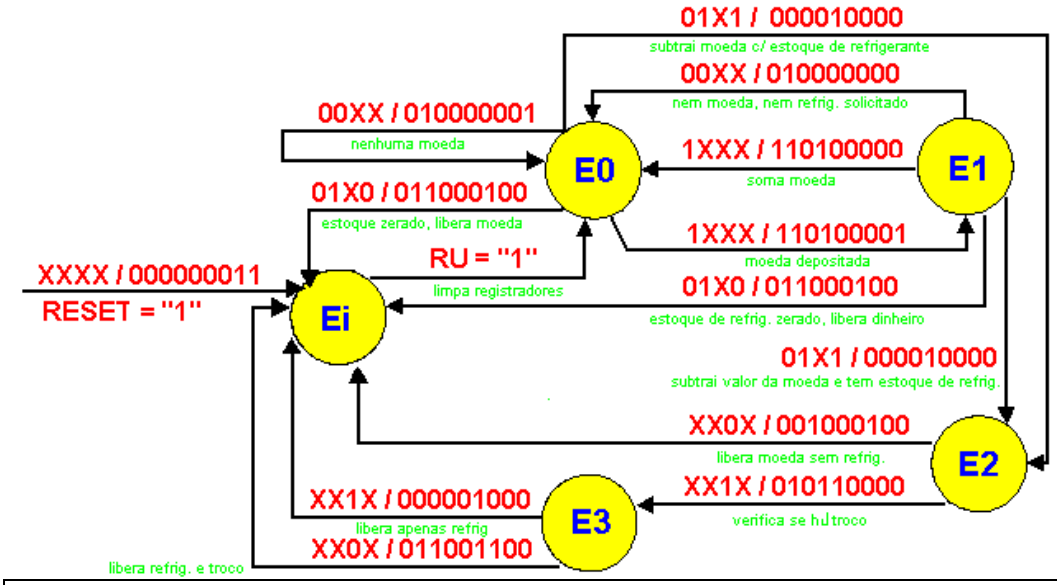


A ULA possui quatro operações que variam como o valor colocado na “setula”:

[00] A saída corresponde ao valor do registrador REGULA;
[01] Subtrai o valor de entrada da ULA do valor do registrador REGULA (ulaout = ulainreg – ulainmux);
[10] Soma o valor do registrador REGULA ao valor de entrada da ULA
(ulaout = ulainreg + ulainmux);
[11] Verifica se o valor do registrador REGULA é igual ao valor de entrada da ULA (se a igualdade acontecer, status = 1).

No bloco de controle foram incrementados sinais de reset específicos para o registrador da ULA e para o registrador do troco para que a mesma tivesse um funcionamento contínuo.

O modulo da unidade de controle foi implementada usando o *diagrama de estados*, ilustrado a seguir:



EM, PD, ST, ER / SM, LR, LT, CT, RL, DD, RU, RT

Entradas:	**Saídas**	
EM – Indica moeda	SM - Seta mux de moedas	RL – Libera refrigerante
PD – Pede refrigerante	LR – Load do reg. da ULA	DD – Devolve dinheiro
ST – Status	LT – Load do reg. do troco	RU – Reset do reg. da ULA
ER – Estoque do refrigerante	CT – Controle da ULA	RT – Reset do reg. do troco

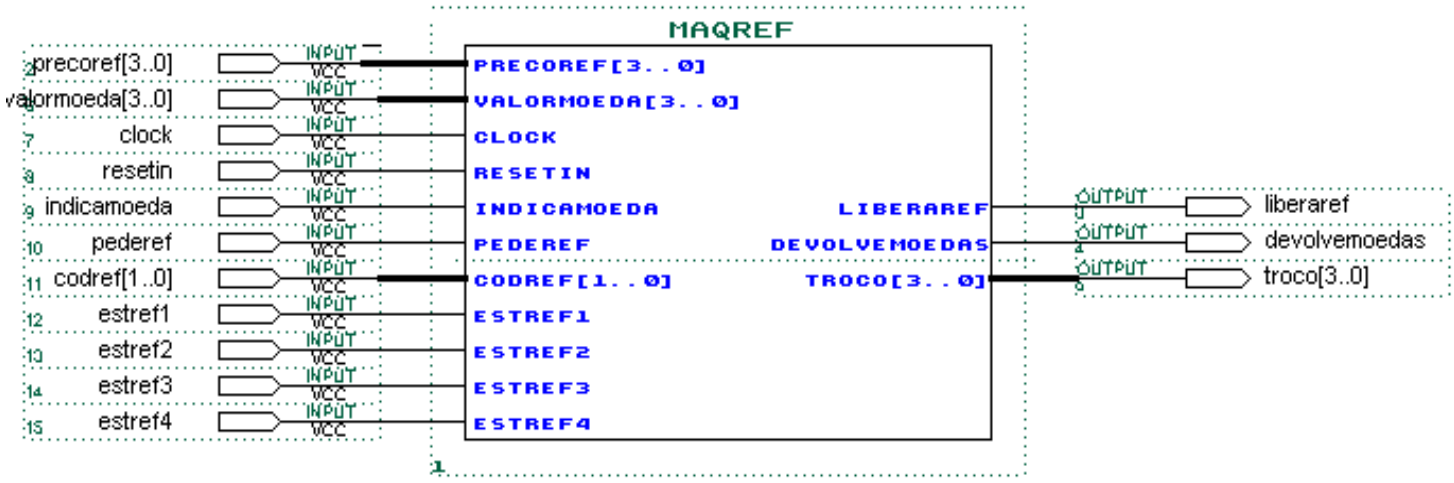
A máquina de vender refrigerante funciona de seguinte maneira:

Resetada a máquina o controle gera os sinais de reset para os registradores. No próximo pulso de clock o controle irá para o estado E0, enquanto o sinal “indica moedas” estiver alto os valores de entrada das moedas serão somados, ficando o diagrama de estados nos estados E0 e E1. Quando o sinal “pede refrigerante” for acionado, a máquina irá verificar se existe estoque do refrigerante solicitado, caso não haja, a máquina irá para o estado Ei, enviando o sinal “libera dinheiro”. No pulso seguinte irá para o estado E0 devolvendo o valor depositado. Caso haja estoque de refrigerante a máquina irá para o estado E2, onde verificará se o valor depositado das moedas é menor que o preço do refrigerante. Se o valor depositado for menor que o valor do refrigerante, a máquina vai para o estado Ei e no próximo pulso de clock para o estado E0 liberando o valor depositado. Caso contrário, se o valor do deposito for maior ou igual que o preço do refrigerante, a máquina irá para o estado E3, onde fará a comparação, caso seja igual irá para o estado Ei, liberando o refrigerante e indo para o estado E0 no próximo pulso de clock. Caso o valor depositado seja maior do que o valor do refrigerante a máquina irá para o estado Ei, liberando o refrigerante e o troco no próximo pulso de clock, ou seja, estado E0.

O projeto foi compilado para implementação em um FPGA da família FLEX 10K da Altera. Para facilitar a montagem e verificação padronizou-se seus pinos conforme tabela a seguir:

HOLE (PLACA)	PINO (FPGA)	SINAL	DESCRIÇÃO
27	193	precoref0	valor do refrigerante
28	194	precoref1	
30	196	precoref2	
29	195	precoref3	
18	183	valormoeda0	valor da moeda
17	182	valormoeda1	
16	181	valormoeda2	
15	175	valormoeda3	
22	187	indicamoeda	detecta moeda
21	186	pederef	pede refrigerante
24	190	clock	clock
23	188	resetin	reset
25	191	codref0	código do refrigerante
26	192	codref1	
31	198	estref1	estoque de refrigerantes
32	199	estref2	
33	200	estref3	
34	201	estref4	
35	202	troco0	troco
36	203	troco1	
37	204	troco2	
38	206	troco3	
40	208	liberaref	libera refrigerante
39	207	devolvemoedas	libera troco

A representação esquemática do FPGA é a seguinte:



OBS.: ANEXAR AQUI O **SUMÁRO DO DISPOSITIVO** E **PINAGEM ALTERADAS DO PLD**, CAPTURADOS NO ARQUIVO DE EXTENSÃO ".rpt" DA MÁQUINA DE REFRIGERANTES.

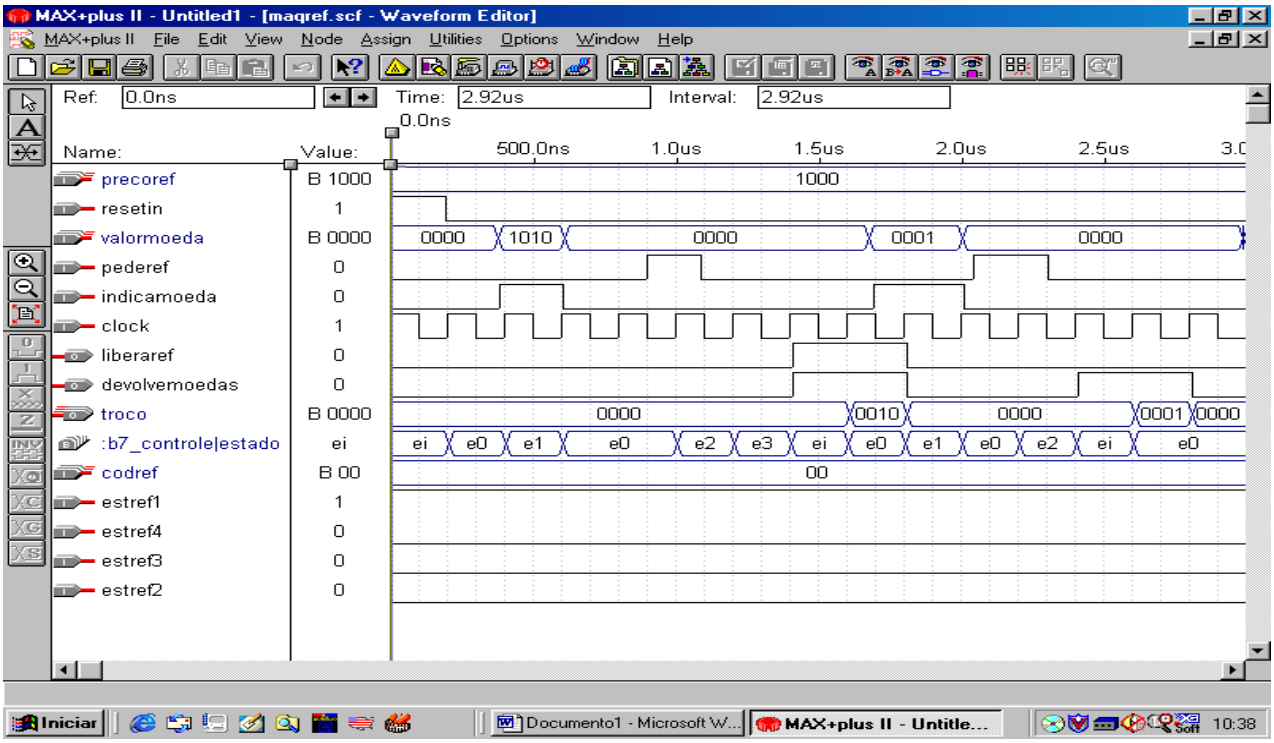
Listagem da entidade *maqref.vhd* implementando a máquina de vender refrigerantes.

```
-- Projeto Máquina de Refrigerantes
-- maqref.vhd - Implementação da Máquina de Refrigerantes
-- Versão 0.0 - 03/08/2000
-- Autor: Jener Toscano

entity maqref is
port  (
  precoref        : in bit_vector (3 downto 0); --preco do refrigerante
  valormoeda      : in bit_vector (3 downto 0);   --valor da moeda
  clock           : in bit;                    --clock externo
  resetin         : in bit;                    --reset de entrada
  indicamoeda     : in bit;                    --indica entrada de moeda
  pederef         : in bit;                    --pede refrigerante
  codref          : in bit_vector (1 downto 0);   --codigo refrigerante
  estref1         : in bit;               --estoque do refrigerante 1
  estref2         : in bit;               --estoque do refrigerante 2
  estref3         : in bit;               --estoque do refrigerante 3
  estref4         : in bit;               --estoque do refrigerante 4
  liberaref       : out bit;                     --libera refrigerante
  devolvemoedas   : out bit;                     --devolver dinheiro
  troco           : out bit_vector (3 downto 0)  --valor do troco

  -- sinais utilizados apenas para visualizacao na simulacao
  --ula_reg_teste       : out bit_vector (3 downto 0);
  --load_reg_ula_teste  : out bit;
  --ula_out_teste       : out bit_vector (3 downto 0);
  --load_reg_troco_teste: out bit;
  --status_teste        : out bit;
  --ula_in_teste        : out bit_vector (3 downto 0);
  --reset_teste         : out bit
  ------------------------------------------------------------
  );
end maqref;


architecture arc_maqref of maqref is

component regpre             -- Registrador do preco do refrigerante
port (
   preco    : in bit_vector (3 downto 0);
   clk      : in bit;
   precoout : out bit_vector (3 downto 0)
 );
end component;

component regula             -- Registrador da ULA
port (
   regulain       : in bit_vector (3 downto 0);
   loadregula     : in bit;
   clk            : in bit;
   resetregula    : in bit;
   regulaout      : out bit_vector (3 downto 0)
 );
end component;

component regtroco           -- Registrador do troco
port (
   regtrocoin     : in bit_vector (3 downto 0);
   loadregtroco   : in bit;
   clk            : in bit;
   resetregtroco  : in bit;
   regtrocoout    : out bit_vector (3 downto 0)
```

```
 );
end component;

component muxula          -- Multiplexador da entrada da ULA
port (
      muxulapreco : in bit_vector (3 downto 0);
      moeda       : in bit_vector (3 downto 0);
      setmoeda    : in bit;
      muxulaout   : out bit_vector (3 downto 0)
  );
end component;

component muxref           -- Multiplexador do estoque de refrigerante
 port (
      ref1        : in bit;
      ref2        : in bit;
      ref3        : in bit;
      ref4        : in bit;
      codref      : in bit_vector (1 downto 0);
      muxrefout   : out bit
  );
end component;

component ula              -- Unidade Logica Aritmetica
port (
      ulainmux    : in bit_vector (3 downto 0);
      ulainreg    : in bit_vector (3 downto 0);
      setula      : in bit_vector (1 downto 0);
      ulaout      : out bit_vector (3 downto 0);
      status      : out bit
  );
end component;
component controle         -- controle da maquina
 port (
  clk       : in bit;
  reset : in bit;
  em    : in bit;
  pd    : in bit;
  st    : in bit;
  er    : in bit;
  sm    : out bit;
  lr    : out bit;
  lt    : out bit;
  ct    : out bit_vector (1 downto 0);
  rl    : out bit;
  dd    : out bit;
  ru    : out bit;
  rt    : out bit
 );
end component;

        -- declaracao dos sinais utilizados para a montagem da maquina

 signal ula_reg         : bit_vector (3 downto 0);
 signal ula_out         : bit_vector (3 downto 0);
 signal load_reg_ula    : bit;
 signal load_reg_troco  : bit;
 signal reg_pre_out     : bit_vector (3 downto 0);
 signal set_mux_moeda   : bit;
 signal ula_in          : bit_vector (3 downto 0);
 signal estoque_ref     : bit;
 signal controle_ula    : bit_vector (1 downto 0);
 signal status          : bit;
 signal reset_ula       : bit;
 signal reset_troco     : bit;
```

```
begin
      --preco,clk,precoout
 b1_regpre        : regpre    port map (precoref, clock, reg_pre_out);

      --regulain,loadregula,clk,resetregula,regulaout
 b2_regula        : regula    port map (ula_out, load_reg_ula, clock,
reset_ula, ula_reg );

      --regtrocoin,loadregtroco,clk,resetregtroco,regtrocoout
 b3_regtroco      : regtroco  port map (ula_out, load_reg_troco, clock,
reset_troco, troco);

      --muxulapreco,moeda,setmoeda,muxulaout
 b4_muxula        : muxula    port map (reg_pre_out, valormoeda, set_mux_moeda,
ula_in);

      --ref1,ref2,ref3,ref4,codref,muxrefout
 b5_muxref        : muxref    port map (estref1, estref2, estref3, estref4,
codref, estoque_ref);

      --ulainmux,ulainreg,setula,ulaout,status
 b6_ula           : ula             port map (ula_in, ula_reg, controle_ula,
ula_out, status);

      --clk,reset,em,pd,st,er,sm,lr,lt,ct,rl,dd,ru,rt
 b7_controle      : controle  port map (clock, resetin, indicamoeda, pederef,
status, estoque_ref, set_mux_moeda, load_reg_ula, load_reg_troco, controle_ula,
liberaref, devolvemoedas, reset_ula, reset_troco);

 -- sinais utilizados apenas para visualizacao na simulacao
 --ula_reg_teste        <= ula_reg;
 --load_reg_ula_teste   <= load_reg_ula;
 --ula_out_teste        <= ula_out;
 --load_reg_troco_teste <= load_reg_troco;
 --status_teste         <= status;
 --ula_in_teste         <= ula_in;
 --reset_teste          <= reset_ula;
end arc_maqref;
```

Simulação da entidade maqref.



CONCLUSÃO

A experiência do uso da linguagem VHDL na prototipação de circuitos digitais utilizando PLD's ficou bastante vivenciada, tendo como meta alcançada, o funcionamento da máquina de vender refrigerantes.

Assim, o curso de Prototipação Rápida de Circuitos Integrados Digitais mostrou diante a evolução da microeletrônica as modernas técnicas utilizadas nos projetos de Sistemas Digitais de Engenharia, tendo o computador como uma ferramenta de auxílio na programação de PLDs (Ferramentas de CAD), junto a linguagem de programação de hardware VHDL, de maneira a acelerar a execução dos projetos desses dispositivos.