UNIVERSIDADE FEDERAL DA FRONTEIRA SUL

CURSO: CIÊNCIA DA COMPUTAÇÃO

CCR: SISTEMAS DIGITAIS

Prof. Geomar Andre Schreiner

Trabalho Final - Blackjack

Aysha Thayná Menezes de Lima[1], Erickson Giesel Muller[2] e Thais Regina Tedesco Zanella[3]

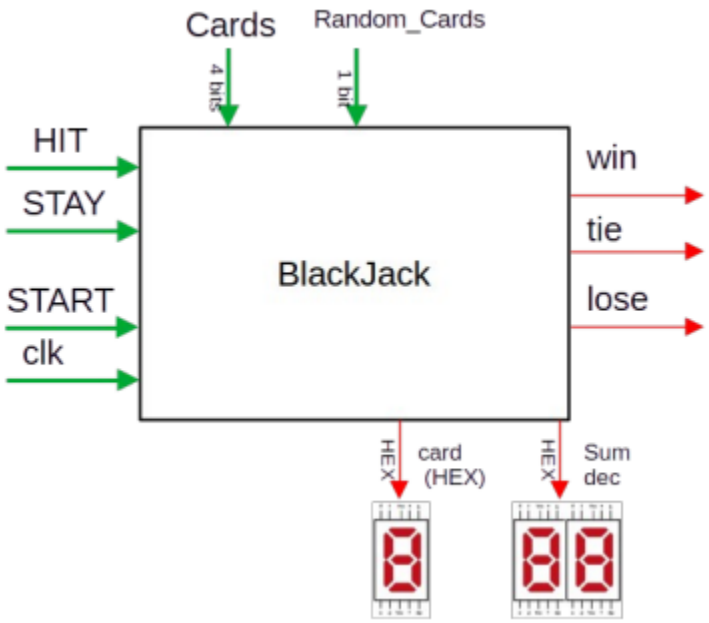
Chapecó, 5 de Dezembro de 2024

[1] Matrícula: 20230002943, aysha.lima@estudante.uffs.com.br
[2] Matrícula: 20230001178, e.rickson@live.com
[3] Matrícula: 20230000958, thais.zanella@estudante.uffs.edu.br

1. INTRODUÇÃO:

O trabalho final teve como objetivo a projetar e implementar uma Máquina de Estados Finitos para controlar um jogo de Blackjack. A implementação foi feita na FPGA DE1 - Ciclone II.

Blackjack é um jogo de cartas, jogado por um único jogador contra o dealer. O objetivo é que a soma dos valores das cartas na mão do jogador seja o mais próximo possível de 21, sem ultrapassar este valor. As cartas têm seu valor nominal, com exceção do ás, que pode valer 1 ou 11, dependendo do que for mais conveniente para o jogo. Ambos jogadores recebem 2 cartas, e é possível pedir mais cartas (“hit”) ou se manter com as cartas atuais (“stay”). Se a soma total de um jogador ultrapassar 21, ele perde automaticamente. O jogador que tiver a soma mais próxima de 21 sem ultrapassar este valor, vence. Caso ambos os jogadores possuam o mesmo valor em cartas, ocorre um empate.



Implementação proposta na atividade

2. DESENVOLVIMENTO:

O desenvolvimento do trabalho teve início pela criação do diagrama de estados, verificando os passos necessários para cada ação que deveria ser executada pela máquina de estados.

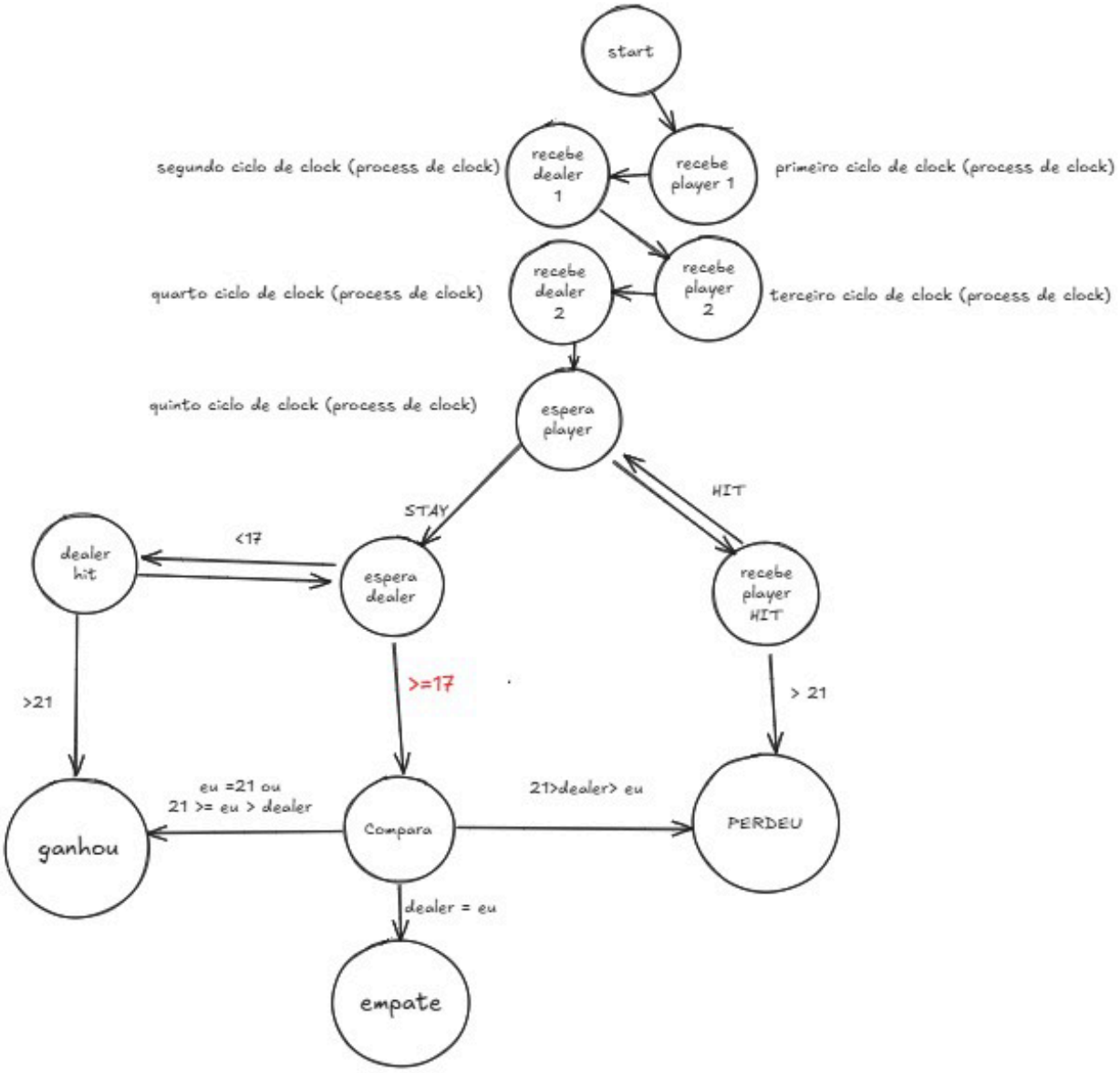


Diagrama de estados

Lógica de Controle:

- **reset/start:** Volta para o início do jogo e reseta as variáveis/signals.
- **recebePlayer1:** Player recebe a primeira carta.
- **recebeDealer1:** Dealer recebe a primeira carta.
- **recebePlayer2:** Player recebe a segunda carta.
- **recebeDealer2:** Dealer recebe a segunda carta.
- **esperaPlayer:** Espera a escolha do player (STAY/HIT)
- **hitPlayer:** Player recebe carta.
- **esperaDealer:** Espera a escolha do dealer(soma>=17 =>STAY, soma<17 =>HIT).
- **hitDealer:** Dealer recebe carta
- **compara:** Compara a soma do dealer e o player
- **perdeuP:** Player perdeu.
- **ganhouP:** Player ganhou.
- **empate:** Empate.

A partir da criação do diagrama de estados, foi criado o código VHDL para a implementação deste projeto. O código foi gerado e testado utilizando o sistema para simulações Digital, o editor de código Visual Studio Code e a FPGA.

2.1 Descrição do sistema

O sistema possui as seguintes entradas e saídas:

Entradas:

- **KEY(0) (clock)**
- **KEY(1) (Reset)**
- **SW(9) (HIT)**
- **SW(8) (STAY)**
- **SW(3 downto 0) (Valor da carta)**

Saídas:

- **HEX0 a HEX3 (Exibe as cartas e a soma total)**
- **LEDR (Indica derrota)**
- **LEDG (Indica vitória)**
- **Quando ligados LEDR e LEDG juntos indicam empate**

2.2 Funções

- **Int2Display:**

Converte valores inteiros em padrões de sete segmentos para exibição no display de 7 segmentos.

- **DysplayHand_dezena:**

Verifica e exibe o dígito da dezena de valores totais.

2.3 Tabela de Estados

A Tabela de Estados foi realizada a partir do Diagrama de Estados e o código VHDL.

Tabela de estados:

Estado Atual	Condição	Proximo Estado
Reset	KEY(0) = ‘0’	RecebePlayer1
RecebePlayer1	KEY(2) = ‘0’	Reset
RecebePlayer1	KEY(0) = ‘0’	Recebedealer1
RecebeDealer1	KEY(2) = ‘0’	Reset
RecebeDealer1	KEY(0) = ‘0’	RecebePlayer2

RecebePlayer2	KEY(2) = ‘0’	Reset
RecebePlayer2	KEY(0) = ‘0’	RecebeDealer2
RecebeDealer2	KEY(2) = ‘0’	Reset
RecebeDealer2	KEY(0) = ‘0’	EsperaPlayer
EsperaPlayer	KEY(2) = ‘0’	Reset
EsperaPlayer	KEY(0) = ‘0’, SW(9) =’1’	HitPlayer
EsperaPlayer	KEY(0) = ‘0’ , SW(8) =’1’	EsperaDealer
HitPlayer	KEY(2) = ‘0’	Reset
HitPlayer	KEY(0) = ‘0’	EsperaPlayer
EsperaDealer	KEY(2) = ‘0’	Reset
EsperaDealer	KEY(0) = ‘0’ , se somaDealer <17	HitDealer
EsperaDealer	KEY(0) = ‘0’, se somaDealer >=17	Compara
HitDealer	KEY(2) = ‘0’	Reset
HitDealer	KEY(0) = ‘0’	EsperaDealer
Compara	KEY(2) = ‘0’	Reset
Compara	KEY(0) = ‘0’, se (playerSoma = 21 and dealerSoma /= playerSoma) or (playerSoma < 21 and playerSoma > dealerSoma)	GanhouP
Compara	KEY(0) = ‘0’ , se (dealerSoma = 21 and dealerSoma /= playerSoma) or (dealerSoma < 21 and dealerSoma > playerSoma)	PerdeuP
Compara	KEY(0) = ‘0’ , se Dealer = Player.	Empate
GanhouP	KEY(0) = ‘0’	Reset
PerdeuP	KEY(0) = ‘0’	Reset
Empate	KEY(0) = ‘0’	Reset

3. CONCLUSÃO

O código implementa com sucesso um jogo de blackjack funcional e interativo. Ele é bem estruturado e faz uso eficiente de estados e sinais digitais para controlar o jogo e exibir os resultados.