Universidade Federal de Santa Catarina
Centro Tecnológico - CTC
Departamento de Engenharia Elétrica - EEL

Marcos Filipe de Liz Rodrigues (15200613)
Mariany Ferreira (15200614)

01028D

<marcsfilipe.liz16@gmail.com >
<fsilvamariany@gmail.com >

Relatório de Projeto Final EEL5105
2015.2

Florianópolis, 1 de Dezembro de 2015.

Conteúdo

Introdução

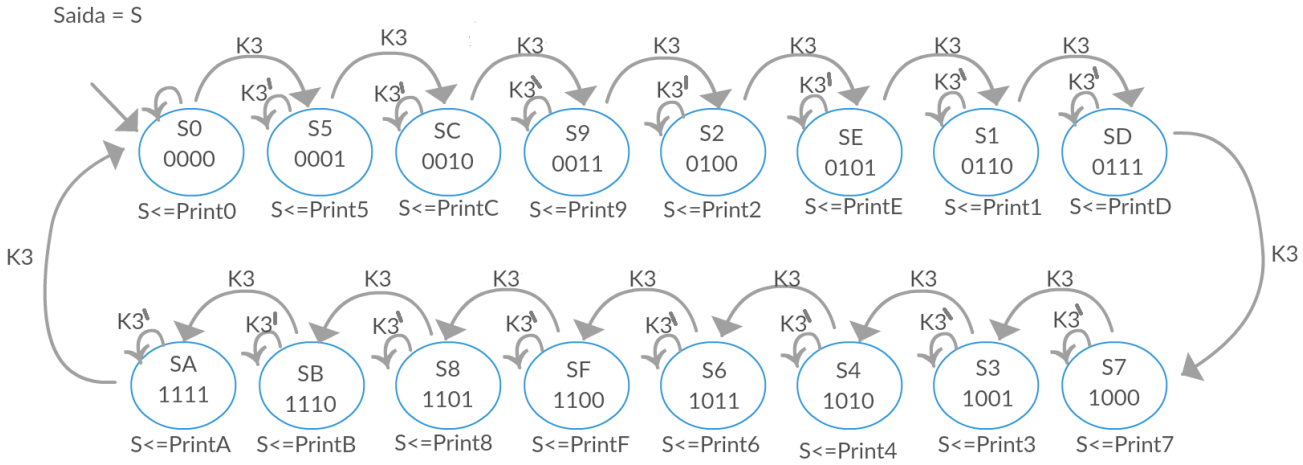
O projeto tem como ideia a construção de uma maquina de jogo do estilo Slot Machine. Para a descrição de toda a estrutura da máquina foram usados ao todo 20 arquivos VHDL, separados em 5 blocos (Sequenciadores, contador, controle, seletor de nível e comparador) esses 5 blocos foram unidos e interligados através de um bloco superior, nomeado como **SLOTMACHINE.** Cada bloco possui sua função exclusiva. O bloco dos Sequenciadores cuida de gerar a sequência da máquina que vai para o bloco comparador onde compara a sequencia vinda do bloco dos sequenciadores e define o valor do prêmio que o jogador recebe por rodada que é enviado ao bloco somador, que gerencia o credito do jogador, seletor de nível seleciona o nível de jogo e está conectado ao bloco comparador afim de definir qual valor de premio o jogador recebe, pois cada nível possui uma pontuação e uma velocidade (obtida através do clock) diferenciada . O bloco de controle está ligado com todos os demais blocos, com exceção do seletor de níveis. O objetivo do jogo é obter um credito maior que o inicial no fim de cada jogo.

1.1 Sequenciadores

Os sequenciadores da slot Machine foram organizados em um bloco contendo seis arquivos VHDLs, sendo eles: **Topo_sequenciadores, Sequenciadores_FSM_C1, Sequenciadores_FSM_C2, Sequenciadores_FSM_C3 e Sequenciadores_decod.**

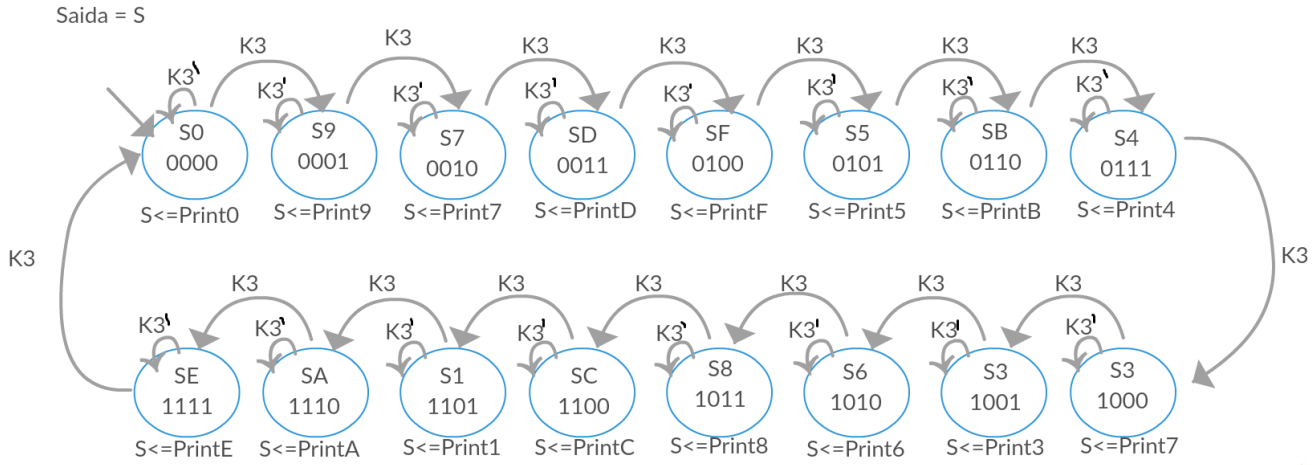
- **Sequenciadores_FSM_C1** é uma maquina de estados comportamental com 16 estados para gerenciar a primeira sequencia da Slot Machine. Iniciada pelo signal '**K3**' que está ligado no componente 'topo' a porta '**C1'**, mudando de estado a cada borda de subida do clock e enviando para a saída o respectivo '**Printn**' (constantes que tem o valor a ser atribuído a saída).

(FSM C1)

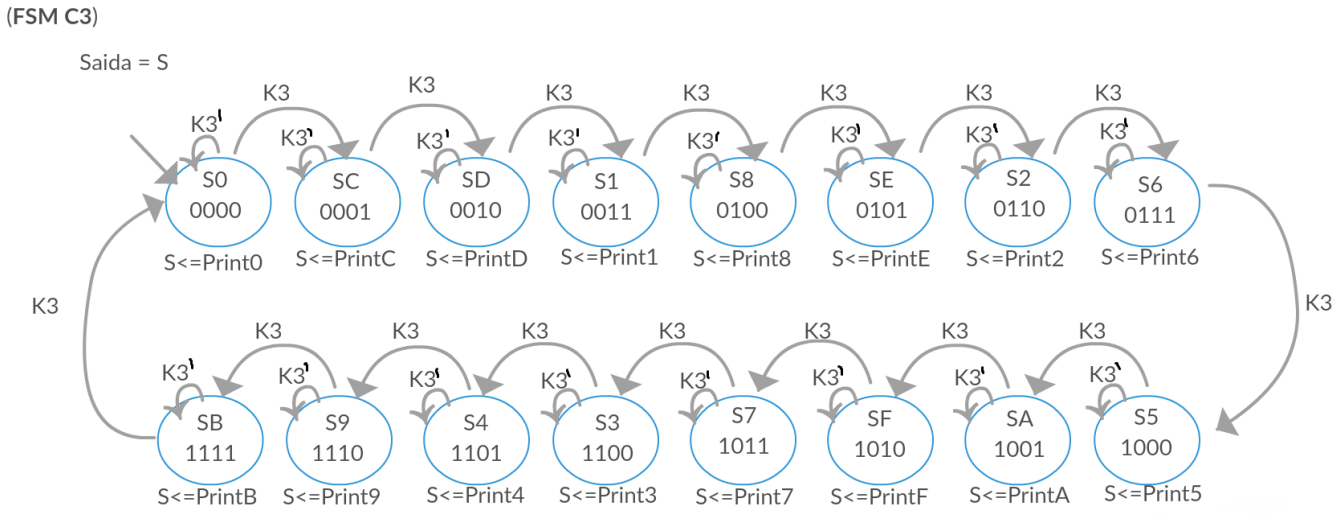


- **Sequenciadores_FSM_C2** comportamental funciona de forma idêntica a **Sequenciadores_FSM_C1**, com a mesma quantidade de estados. No entanto, sua **K3** recebe o valor lógico da porta **C2**, e ela é responsável por gerenciar a segunda sequência da slot Machine. Sua sequência é diferenciada da sequência de **Sequenciadores_FSM_C1** .

(FSM C2)

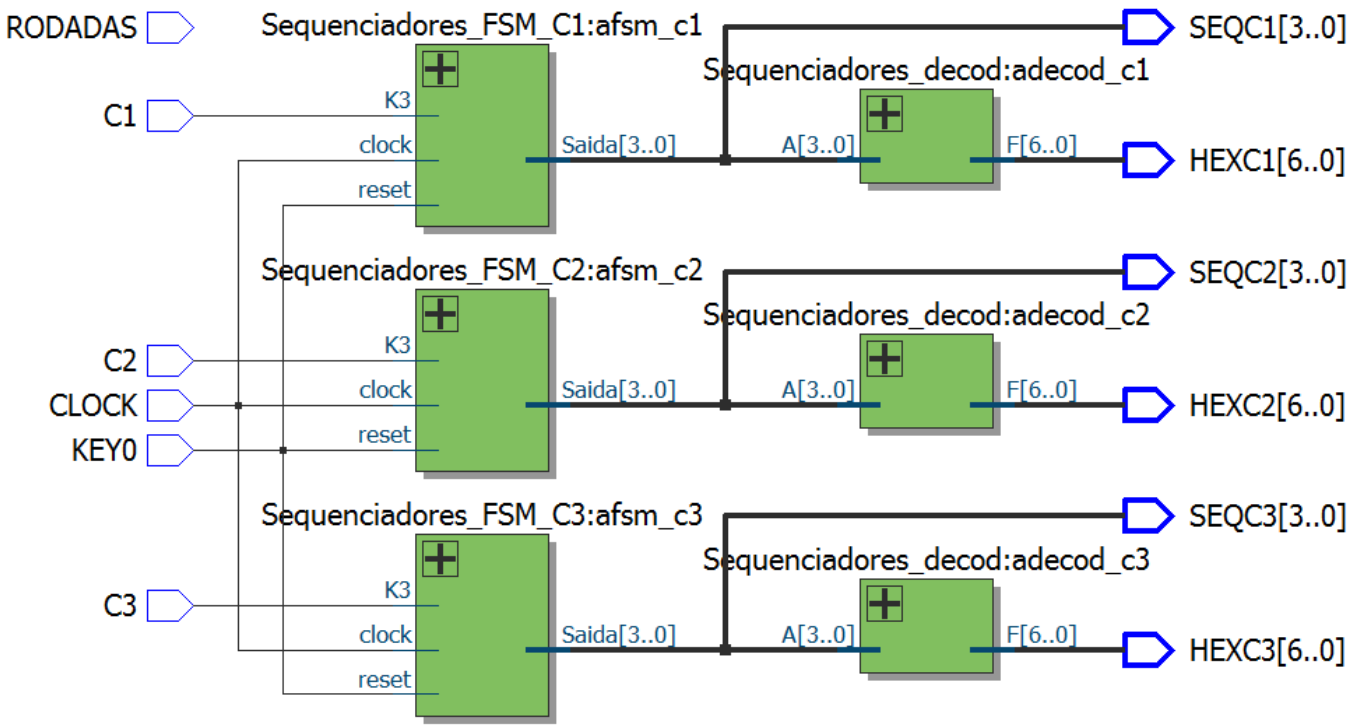


- **A Sequenciadores_FSM_C3** comportamental funciona coma a **Sequenciadores_FSM_C1** e **Sequenciadores_FSM_C1**, novamente com o mesmo número de estados a única alteração é a ligação de sua **K3** com a porta **C3** e sua sequência diferente das demais.

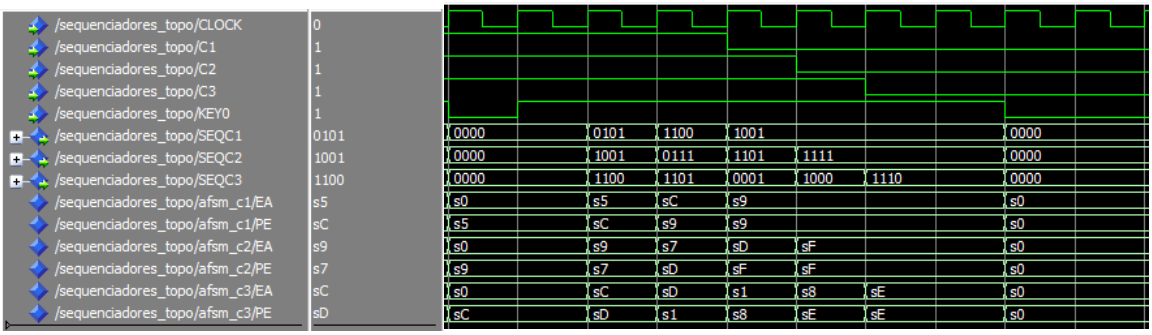


- O **Sequenciadores_decod** transforma a saída de '4' bits vinda das FSMs para serem printadas nos displays de 7 segmentos (**HEX1, HEX2, HEX3**).

- O **Topo_sequenciadores** recebe as entradas **C1, C2, C3, CLOCK e KEY0** , que são conectadas as entradas **k3** (sinal que inicia ou para cada FSM)**, Clock e reset** de cada FSM, que possuem também saídas para os outros blocos e para os displays.



SIMULAÇÃO:



- Na simulação do bloco dos sequenciadores vê-se a funcionalidade de cada sinal de entrada e de cada componente do bloco dos sequenciadores. As FSMs responsáveis por gerenciar as sequencias mudam seu estado e informam o próximo estado a cada subida de clock. No entanto quando seu respectivo sinal que determina seu funcionamento (**C1, C2 e C3**) estão em nível baixo, o estado atual permanece em um loop enquanto a saída (**SEQ1, SEQ2** ou **SEQ3**) recebe constantemente o valor do estado atual. O reset assíncrono da máquina (**KEY0**) ativa-se quando está em nível lógico baixo e quando ativo, todos os estados e saídas voltam ao estado inicial onde esperam para serem iniciadas novamente caso KEY0 e seus sinais de inicio de funcionamento sejam igual a '1'. (EA = estado atual, PE = próximo estado).

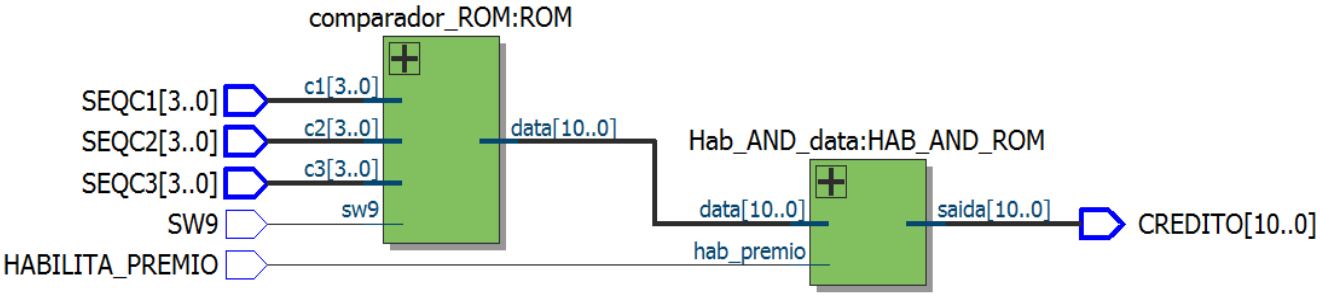
1.2 Comparador

O bloco comparador de sequencias está organizado em um bloco contendo três arquivos VHDLs , **Comparador_ROM, Comparador_AND e Comparador_Topo.**

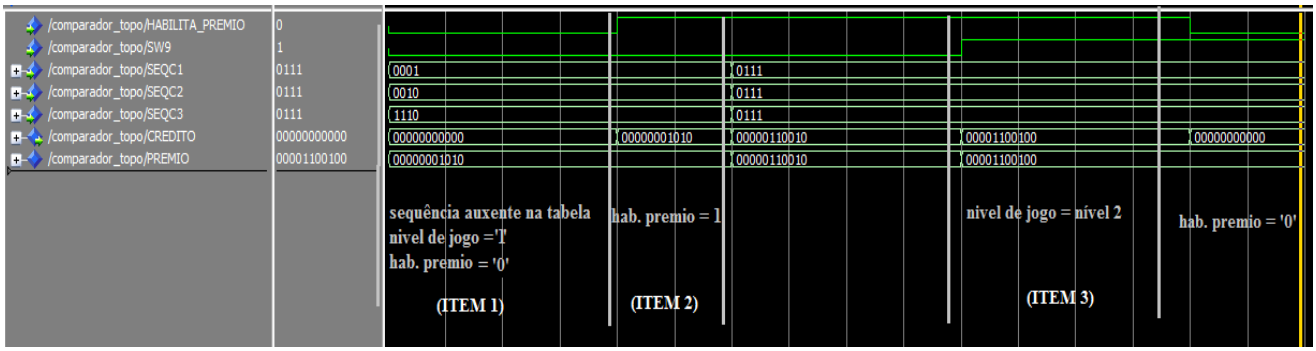
- **Comparador_ROM** é o VHDL comportamental contendo uma memória ROM com um vetor de 15 posições, cada posição contém um valor de prêmio em binário que é atribuindo a uma variável de saída (**data**) de 11 bits. O prêmio a ser liberado depende do nível de jogo escolhido e da sequencia de combinações que são **SW9** (conectado a SW(9), funciona como seletora do nível de jogo), entradas **C1, C2, C3** (ligados respectivamente a SEQ1, SEQ2 E SEQ3) que são concatenadas em um signal '**address'** que é comparado com as posições da memória ROM.

- **Comparador_AND** tem sua forma comportamental onde **saída** recebe **data**, caso o sinal **HABILITA_PREMIO** (sinal lógico usado para definir se o premio será enviado ao somador) seja positivo. Caso não seja, o valor '00000000000' é atribuido a 'saída'.

- **Comparador_Topo** recebe as saídas SEQC1, SEQC2, SEQC3 do bloco dos sequenciadores (**Topo_sequenciadores)** como entrada e os sinais **HABILITA_PREMIO** e **SW9** do bloco de controle Fazendo as ligações com os componentes Comparador_ROM e Comparador_AND. Possui uma saida de 11 bits chamada '**CREDITO**' que recebe o valor da saida do componente Comparador_AND, que é um multiplexador 2:1.



SIMULAÇÃO:



- No início desta simulação a sequencia vinda das FSMs (a concatenação de SEQ1, SEQ2, SEQ3) é uma sequencia ausente na memória rom de prêmios, e como **HABILITA_PREMIO** também inicia em nível lógico baixo a saída '**CREDITO**' recebe a saída '00000000000' do mux. Quando HABILITA_PREMIO recebe nível alto, automaticamente a saída **'PREMIO'** recebe o valor referente na tabela conforme a sequência gerada, e por ser uma sequencia ausente na tabela , como todas as sequencias que não estão presentes, o valor a ser atribuído a **'PREMIO'** é "00000001010" (definido na linha 18 de Comparador_AND). Quando uma sequencia presente na tabela é inserida (item 2) o valor de PREMIO muda, assim como também muda no item 3 mesmo com sequencia sendo a mesma, mas no entanto o nível de jogo é o nível 2 e os prêmios deste nível são 2x o premio gerado no nível 1.

1.3 Contador de Credito

No bloco onde é realizada a contagem e a soma do crédito, cinco VHDLs foram criados: **ContadorCredito_Topo, ContadorCredito_Mux, ContadorCredito_Comb, ContadorCredito_Registrador** e **ContadorCredito_Somador.**

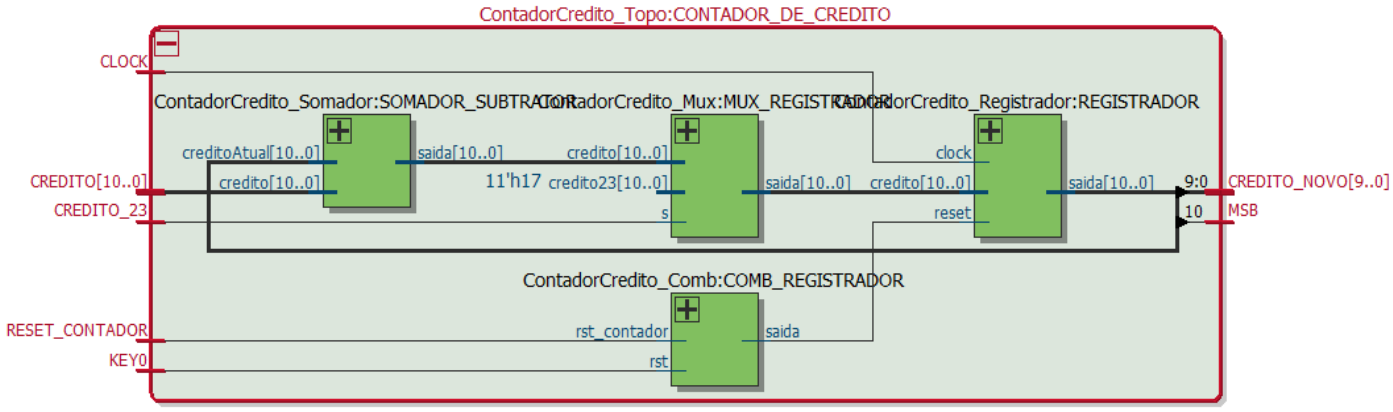
- **ContadorCredito_Somador** é responsável por somar o crédito atual do jogador (variável **creditoAtual,** vinda do registrador de pontos), com o novo crédito que usuário recebe, disponibilizando assim a soma em uma variável de 11 bits, '**saida'.** (A library `ieee.std_logic_signed.all` foi utilizada para ser possível somar os valores).

- **ContadorCredito_Mux** é um multiplexador 2:1 que depende apenas da entrada '**s**' que recebe **credito_23**(variável de saida do VHDL **Controle_FSM**). Quando **s** = '0' a saída do Mux (variável **saida)** recebe **credito** (variável de 11 bits que recebe o valor de saída do **ContadorCredito_Somador**) e quando '**s'** diferente de '0' (isso ocorre por diversos motivos, gerenciados em **Controle_FSM**, um deles é o reset da máquina) a saída do ContadorCredito_mux recebe credito23, uma constante que tem um valor de 23 em binário, definindo o inicio da máquina sempre com um saldo de crédito de 23.

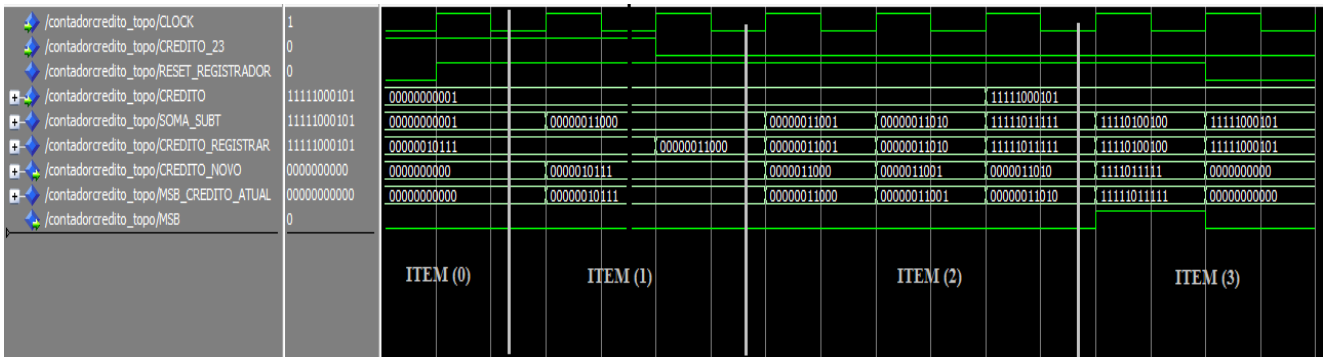
- O VHDL comportamental, **ContadorCredito_Registrador,** executa sua função exclusivamente quando a entrada **reset** sofre alteração em seu nível lógico e/ou quando há borda de subida do **clock**. Esse registrador de 11 bits armazena e fornece em sua variável de saída **'saida'** o valor vindo da variável **'saida'** do Multiplexador enquanto **reset** está em nível lógico baixo. Quando **reset** recebe nível alto, automaticamente o valor '**00000000000**' é registrado e fornecido á saída.

- **ContadorCredito_Comb** é uma estrutura com lógica combinacional onde é utilizado uma porta **'or'** entre **rst** (entrada que recebe a **KEY0**) e **rst_contador** (sinal vindo de **Controle_FSM**). O resultado dessa combinação define o nível logico do reset presente no **ContadorCredito_Registrador** que é ativo em nível lógico baixo**.**

- O **ContadorCredito_Topo** recebe como entrada a variável **CREDITO** de 11 bits (variável de saída do bloco **Comparador_Topo** que possui o valor de credito a ser somado com o valor de **creditoAtual,** vinda do registrador de pontos, que possui a quantia de credito já obtido pelo jogador . **CREDITO_23** e **RESET_CONTADOR** (ligada ao **rst_contador** de **ContadorCredito_Comb**) são entradas de um bit vindas da **Controle_FSM** (pertencente ao bloco de controle do Slot Machine). Possui uma saída de 1 bit nomeada de **MSB** que é um sinal que identifica se o credito é positivo ou negativo para ser enviado ao bloco de controle e a saída **Credito_novo** de 10 bits, que é o valor de créditos que o jogador possui que retorna para o somador e também é conectada aos Leds **LEDR** da placa Cyclone V.



SIMULAÇÃO:



- Nesta simulação do bloco gerenciador do crédito, inicialmente (ITEM (0)) o valor de pontos das sequencias é '1', e como não havia nada no registrador a soma (**SOMA_SUB**) fica igual a 1. Mas como o sinal **CREDITO_23** está em nível lógico alto, a saída do mux, (**CREDITO_REGISTRAR**) que define qual valor será armazenado no registrador é o valor 23 ("00000010111"). No entanto o registrador armazena "00000000000", pois o reset **RESET_REGISTRADOR** está ativo (RESET_REGISTRADOR ativo quando em nível lógico baixo). No ITEM(1) quando o reset está desativado o registrador recebe o valor 23, pois **CREDITO_23** ainda está ativo. A soma acontece levando em conta o novo valor do registrador e **CREDITO_REGISTRAR** será enviado ao registrador no próximo sinal de clock, pois o registrador se altera apenas com a subida do **CLOCK** ou com o reset. No ITEM(2) as somas continuam sendo executadas sem nunhuma modificação pois nenhum sinal além do clock sofre alteração, apenas na terceira subida de clock deste item, **CREDITO** recebe um valor negativo a ser subtraído do Credito atual do jogador, por isso no ITEM(3) o sinal **MSB** torna-se igual a '1', para sinalizar ao controle da Slot Machine que o valor de pontos do jogador é negativo. Nesse mesmo item, o reset é ativo na segunda borda de subida do clock. Resetando todos os entradas e o sinal MSB volta a ser '0'.

1.4 Seletor de nível

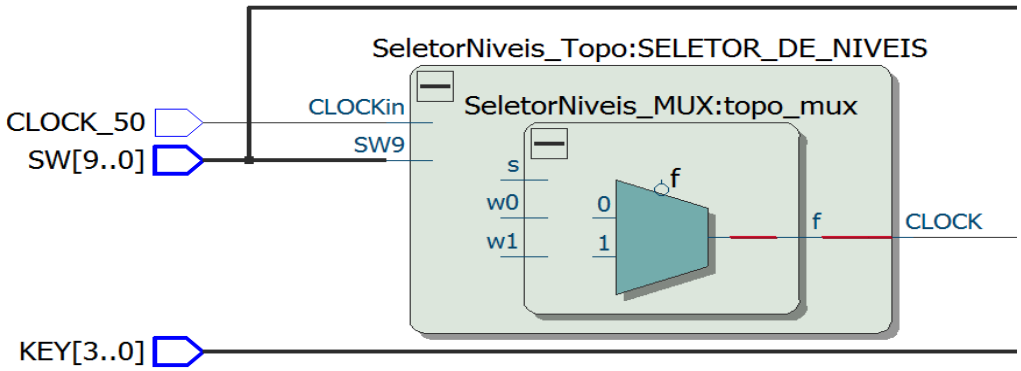
O bloco do seletor de níveis conta com três VHDLs, que fazem a troca de níveis de jogo com alterações no frequência do clock. **SeletorNiveis_Clock, SeletorNiveis_Mux e SeletorNiveis_Topo.**

- **SeletorNiveis_Clock** é um Componente comportamental que possui como entrada de um bit, **IN_50MHz** (conectado ao CLOCK_50 da máquina), e **OUT_1Hz** (W0 do mux)**, OUT_2Hz** (W1 do mux) como saídas também de um bit. Esse Componente tem a função de converter o clock padrão de 50MHz para 1MHz e 2MHz. Essa conversão é feita através de dois contadores (signals **count_a** e **count_b**) que são acrescentado '1' a cada borda de subida do clock de 50MHz. No caso de count_a Enquanto contador é menor que que '1200000', **OUT_2Hz** recebe nível lógico baixo, e quando ele é maior que '1200000 OUT_1Hz recebe nível lógico alto. Quando count_a é igual a 2499999', o mesmo recebe '0'. Já count_b enquanto menor que '25000000' seu nível permanece baixo, caso seja maior ele passa a ter nível alto. Quando count_b é igual a 50000000 o mesmo recebe '0'.

- **SeletorNiveis_Mux** é um multiplexador 2:1 com a função de definir qual a saída de clock será usada (OUT_1Hz ou OUT_2Hz) a partir do nível de jogo selecionado que é definido pela porta de entrada **SW9** que

é ligada a entrada **'s'** do Mux (se seu nível lógico for igual a '0' o nível de jogo é "nível 1", caso contrário o nível de jogo é "nível 2").

- **SeletorNiveis_Topo** possuí a mesma função dos demais Topos. Contendo um multiplexador e Possuindo as entradas **CLOCK_50** (clock contido na placa Cyclone V)**, SW9** e a saída **CLOCK**.

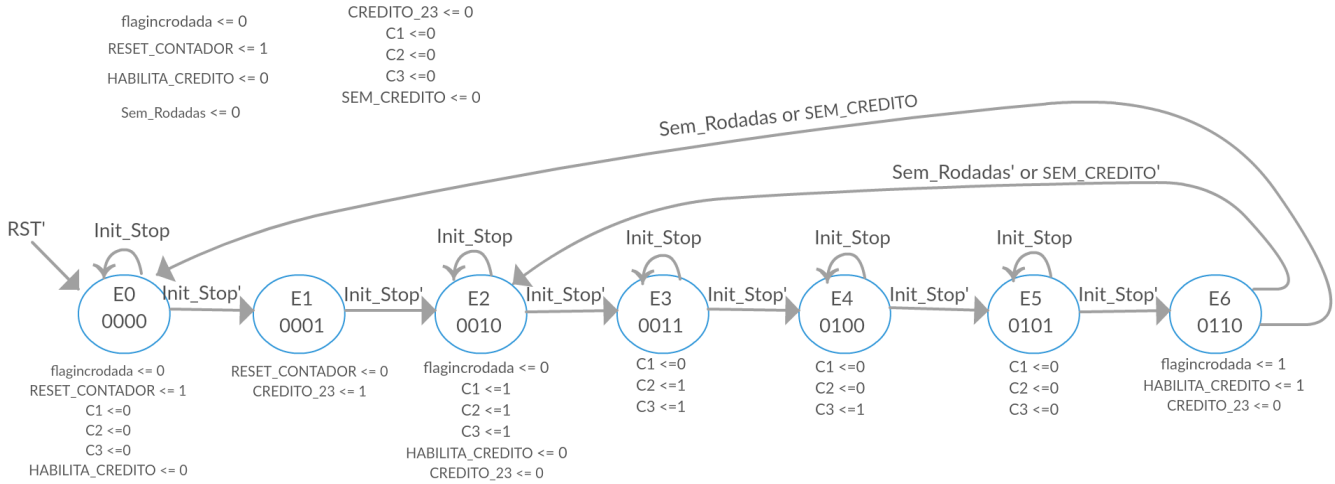


2 Controlador

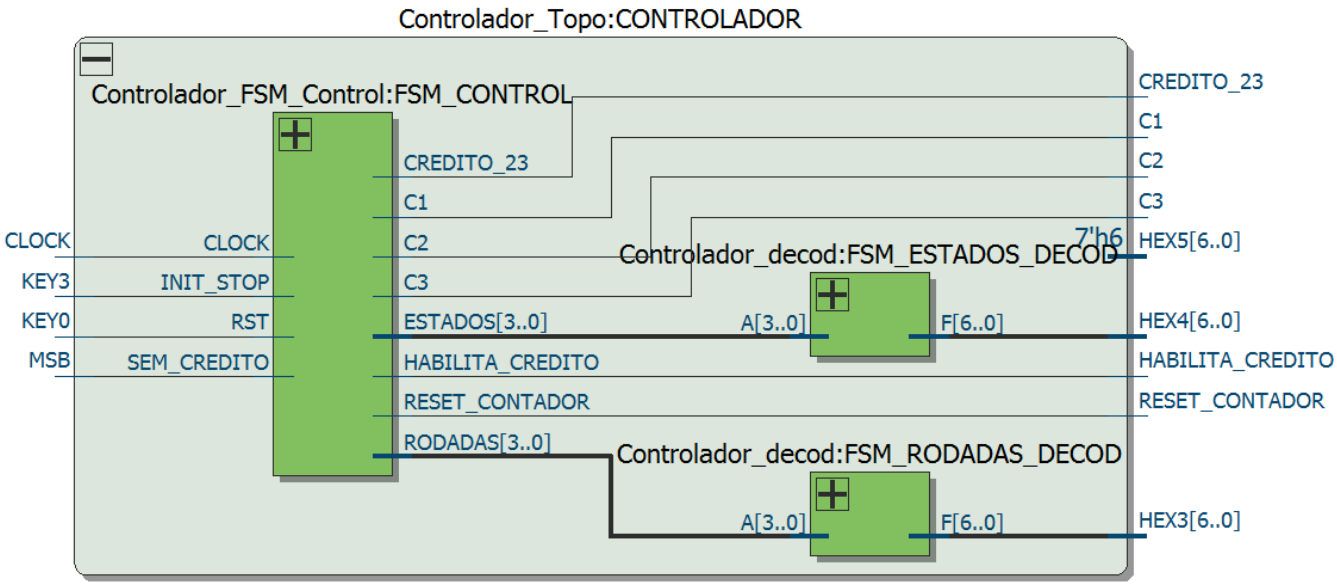
O bloco do controlador é o bloco responsável por controlar todos os outros, através de uma FSM. Possui 3 VHDLs, **Controlador_decod, Controlador_FSM_control** e **Controlador_Topo.**

- **Controlador_decod** é um decodificador de binários para Hexadecimal. Utilizado em dois casos, ele envia a decodificação do estado atual da **Controlador_FSM_control** para o display HEX4 e para o display HEX3 ele envia a rodada atual.

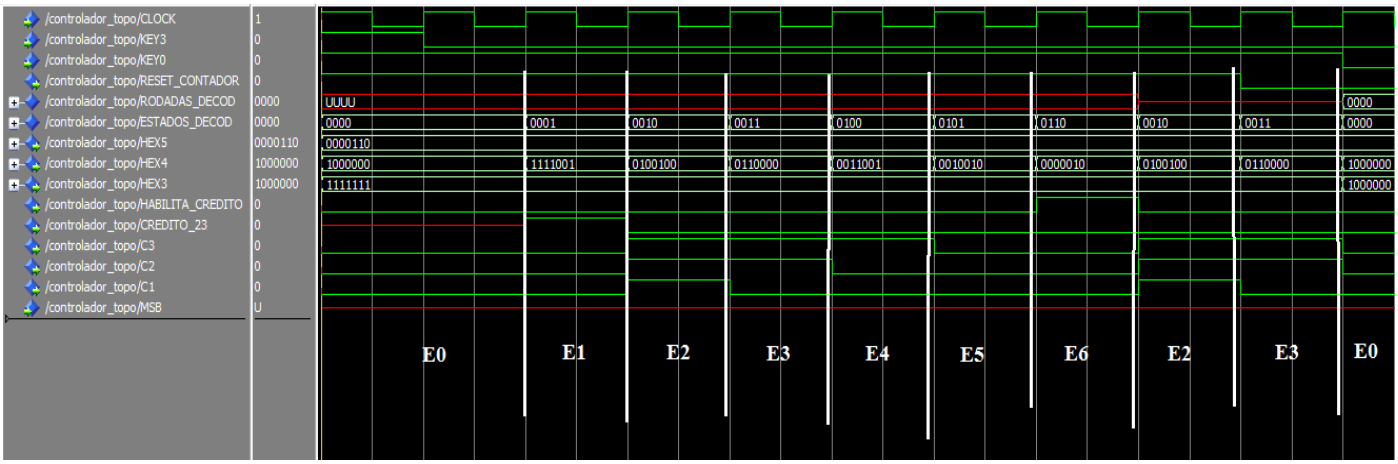
- **Controlador_FSM_control** é a FSM responsável por gerenciar todos os outros blocos, com exceção do bloco Seletor de Níveis, enviando os sinais e dados necessários para os mesmos. Essa FSM possui dos process. O primeiro deles determina o funcionamento de **RST** (conectado a KEY(0) da máquina) e a transição de estados conforme a subida do CLOCK. O segundo define as transições de estados. Essa FSM possui 7 estados que vão de E0 á E6. No E0 a **flagincrodada** (flag usada para sinalizar quando rodadas deve ser incrementado recebe '0'. HABILITA_CREDITO e os sinais que iniciam as FSMs responsáveis pelas sequências são desativados. No estado E1 **CREDITO_23** é ativo, enviando assim os 23 pontos iniciais para o registrador. No estado E3 a **Sequenciadores_FSM_C1** é parada, mantendo o primeiro digito da sequencia. O mesmo acontece com **Sequenciadores_FSM_C2** e **Sequenciadores_FSM_C3** nos estados E4 e E5, respectivamente. No estado E6 a flag **flagincrodada** é ativa e **RODADAS** (variável que registra a rodada de jogo atual) é incrementada. HABILITA_CREDITO também é ativo permitindo que o registrador recebo a nova soma de premio. Nesse estado, é verificado se Rodadas é menor que 10. Caso seja, ela continua o jogo, voltando para o estado E2. Caso seja 10 ela é resetada, e volta para o estado E3. Todos as transições desses estados acontecem por meio do sinal **Init_Stop**, conectado a **KEY(3)** com exceção dos estados E1 que vai para o estado E2 automaticamente e E6 que vai para o estado E2 caso haja credito (determinado pelo sinal **MSB** vinda do registrador que informa se o credito atual é positivo (nível logico baixo) ou negativo) e a quantidade de rodadas jogadas seja menor que '1001' se não vai para o E0 e o jogo é reiniciado.



- **Controlador_Topo** recebe as entradas **CLOCK, KEY3, KEY0 e MSB,** as quais conecta nas devidas entradas de Controlador_FSM e os decodificadores utilizados para imprimir o valor de Rodadas (HEX4), **Estados** (valor do estado atual, HEX5) e Rodadas (HEX6). Possui as saídas CREDITO_23, HABILITA_CREDITO e RESET_CONTADOR para os devidos blocos.



SIMULAÇÃO:

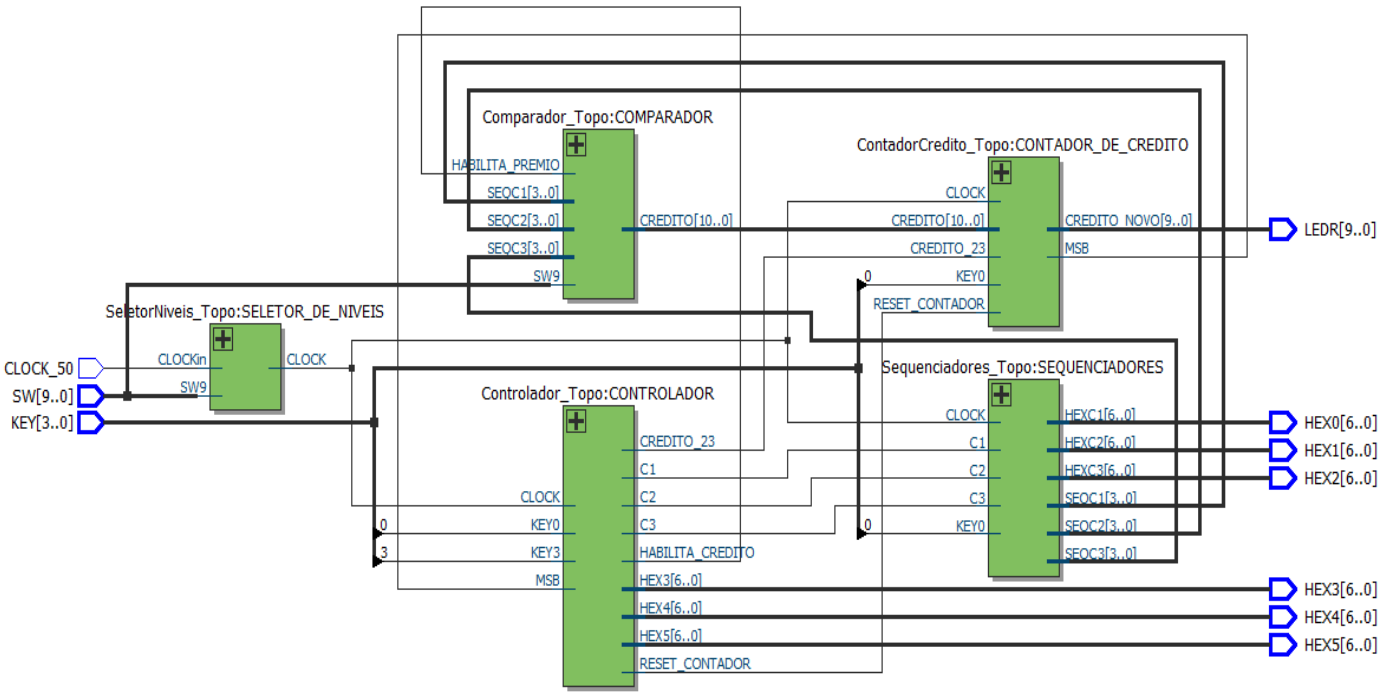


- Na simulação do bloco de controle, no E0 a maquina permanece por dois ciclos de clock no mesmo estado, pois o sinal **KEY3,** que inicia a FSM do controlador, está em nível logico alto (desativada). Neste estado os sinais **HABILITA_CREDITO**, **C1, C2** E **C3** recebem nível logico baixo. Nota-se também que rodadas não foi ativa, pois primeiramente deve ser resetada. No estado E1 o sinal **CREDITO_23** é ativo, gravando no registrador do bloco contador o valor inicial de 23 pontos (“00000010111”) que todo jogador inicia. No Estado E2 CREDITO_23 recebe nível baixo e C1, C2 e C3 passam a ter nível alto, iniciando assim as FSMs que gerenciam as sequências no bloco dos sequenciadores. Nos estados E3, E4 e E5, respectivamente, acontecem apenas as interrupções dos sinais C1, C2 e C3 pois o sinal KEY3 responsável por ativá-las, está em nível ativo. No estado E6 o sinal **HABILITA_CREDITO** é ativo e enviado para o bloco comparador para que possa ser liberado um premio para ir ao bloco contador. E por rodadas ser menor que 10 (neste caso, por estar indefinida) o jogo prossegue e o próximo estado é o estado E2. Os estão vão realizando novamente suas funções até o estado E3, pois nesse momento o reset da máquina é ativo, voltando então para o estado E0.

3 Slot Machine

A SLOTMACHINE é o topo de todos os demais blocos da máquina . Ele é Responsável por fazer a junção de todos os Blocos montados para a realização das operações necesárias para que o Jogo funcione corretamente. Esse VHDL liga todos os sinais necessários entre os Blocos e recebe do Usuário da Máquina os Sinais de reset (KEY(0)), de Inicio e de parada dos sequenciadores (KEY(3)) e de seleção do Nível de Dificuldade (SW(9)).

(topo completo (RTL))



4 Conclusão

Ao final deste projeto concluímos que elaborar um projeto de descrição de hardware requer muito mais que apenas conhecer o método de descrição dessa linguagem (VHDL). Exige concentração, estudo, planejamento, simulação e esforço para buscar métodos de resolver os problemas a serem enfrentados.

Durante este projeto, vários erros e duvidas foram esclarecidos. Alguns momentos onde parecia não haver solução para o problema, na verdade era apenas um pequeno erro, mas que impedia totalmente ou parcialmente o funcionamento correto da máquina.

Logo no inicio nos deparamos com um erro onde as sequências e os dados da máquina se quer apareciam nos displays e nos leds, mas bastou apenas alguns ajustes na forma correta como seriam declarados alguns atributos padrões da máquina (como o nome definido para o clock presente na placa, CLOCK_50). A partir daí tudo começou a se encaixar e a funcionar aos poucos. Encontramos mais Alguns erros durante o processo, uns resolvidos como o problema relacionado ao envio do crédito a ser gravado no registrador de pontos que bastou apenas uma alteração em um sinal qual habilitava o recebimento ou não do premio. Outro problema, no entanto, não pode ser resolvido perfeitamente. O problema é que rodadas ao ser igual a ‘9’ no estado atual deveria receber ‘0’ e resetar a máquina. Os créditos e demais sinais são resetados, mas o numero de Rodadas não recebe ‘0’ como deveria , mas sim continua a contar até atingir o valor 15 (F, em Hexadecimal).

o projeto desta máquina nos concedeu pratica e maior aprendizado com a área de circuitos e técnicas digitais, pois tivemos que correr atrás de conteúdos ou formas de uso do VHDL que ainda não tínhamos conhecimento suficiente.