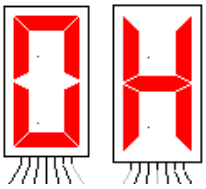
E-fólio A | Instruções para a realização do E-fólio

ARQUITETURA DE COMPUTADORES | 21010 | 2022/2023

Pretende-se uma implementação em Digital Works de um sistema digital que simule o jogo Lamberta, cujas regras se encontram em anexo a este enunciado.

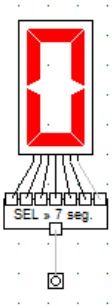
Alínea A (1 valor)

Fazer um circuito combinatório que tenha como entrada um valor binário, correspondente à marca de uma casa. O circuito deverá ter 7 saídas, para ligar a um display de 7 segmentos, que mostrará a marca presente nessa casa. Ao valor 0 deve ser mostrado o 0, e ao valor 1 ser mostrado o H:

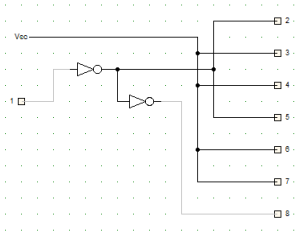


O circuito deverá ter um "*Interactive Input*" para controlar a entrada.

Dado que existe uma entrada e 7 saídas, e pretende-se controlar um display de 7 segmentos, é útil o seguinte componente:



Apenas os segmentos centrais alteram, sendo o segmento f igual à entrada P, e os segmentos a e d iguais à negação de P. Todos os outros são sempre 1.



Atendendo à simplicidade das 7 expressões, que dependem apenas de uma variável, e portanto há apenas 4 hipóteses, não seria necessário grandes justificações.

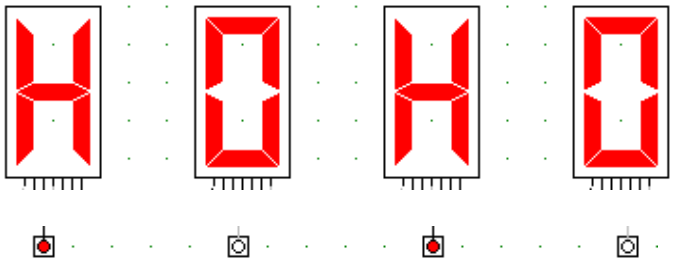
Alínea B (1 valor)

Fazer um circuito sequencial, que permite para cada casa, carregar o valor da casa no tabuleiro. Este valor é mantido em memória, até que seja substituído. Para tal deve existir a entrada LOAD, para carregar o valor caso esteja a 1, ou manter caso esteja a 0, e uma variável P com o valor a carregar. Como saída deve apresentar o valor H, correspondendo ao valor guardado em memória. Esta operação de carregamento de dados deve ser realizada de forma assíncrona.

Sugere-se que faça um componente iterativo para lidar com as casas do tabuleiro, nesta e nas restantes alíneas, de modo a ser simples adicionar ou remover casas.

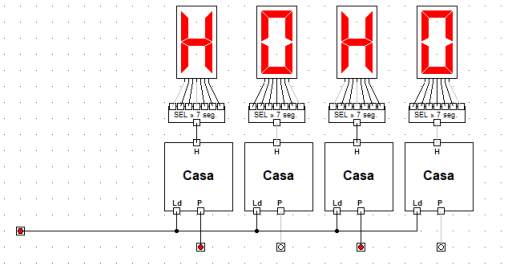
Demonstre o componente digital construído, com 4 casas, reutilizado o componente da alínea anterior para mostrar o valor da casa.

Ilustração:



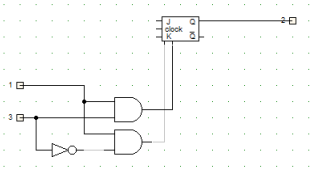
O circuito deverá ter um "*Interactive Input*" ligado ao LOAD de todas as 4 casas, e um "*Interactive Input*" para controlar o valor de P de cada casa, permitindo assim carregar qualquer posição para o tabuleiro.

Pretende-se agora ter hipótese de carregar ou manter o valor em 4 casas. Assim, é útil um segundo componente com o valor de cada casa, com duas entradas, o LOAD e o P:



Como se pretende manter um valor em cada casa, tem que se utilizar um flip-flop. A funcionalidade é não alterar se LOAD=0 e alterar se o LOAD=1, fazendo esta ação de forma assíncrona. Assim, tanto se pode utilizar um RS, ou então utilizar o set/reset assíncrono dos flip-flops com relógio. Vamos utilizar um flip-flop tipo JK:

- Set (preset): LOAD=1 e P=1
- Reset (clear): LOAD=1 e P=0



Ignoramos para já o J, K e Clock, já que estamos a utilizar o preset/clear.

Alínea C (1 valor)

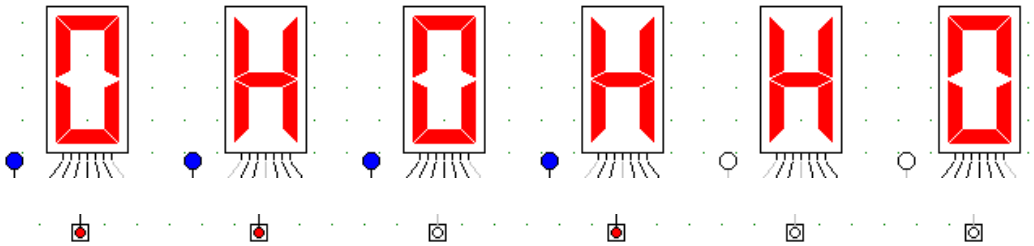
Melhorar o circuito da alínea B de modo a permitir selecionar um segmento e realizar uma jogada, sem verificar a sua validade.

Adicione duas casas ao tabuleiro, e uma nova entrada, ligada a um *"Interactive Input"*. A nova entrada será o sinal de CLOCK que no nosso caso deverá estar ligada a um botão para realizar uma jogada. Em cada sinal de CLOCK (jogada), o estado do tabuleiro deverá mudar de acordo com a seleção realizada.

A seleção do tabuleiro a considerar, deverá ser entre o primeiro e o último botão pressionado, que esteja associado a uma casa (variável P na alínea anterior). Estes botões tanto carregam um novo tabuleiro, como fornecem informação para a seleção do tabuleiro.

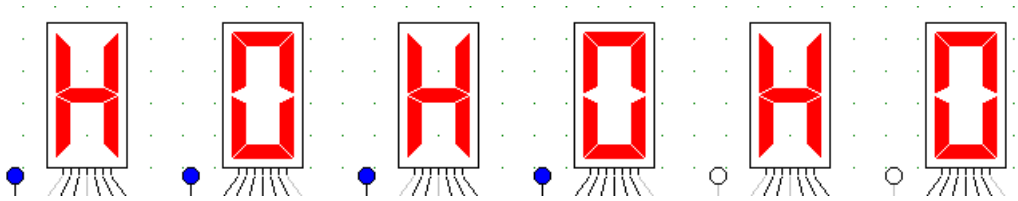
Notar que em consequência da definição de seleção, uma casa deve ficar selecionada se o botão P estiver pressionado, ou então à esquerda e à direita da casa, existir um botão P pressionado.

Ilustração:



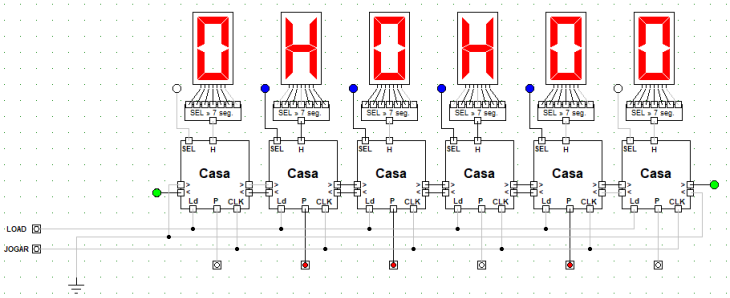
Foram acionados 3 botões, sendo os botões extremos o primeiro e o quarto. Assim, as quatro primeiras casas estão selecionadas.

Ao efetuar a jogada, acionando e libertando o botão da jogada, todas as casas na seleção devem trocar a marca:



Pretende-se a seleção do segmento, e inversão das casas selecionadas. Assim, há que atualizar o componente da casa, com um sinal de *Clock* para jogar.

Um segmento é selecionado se o botão P da casa está pressionado, ou existir uma seleção à esquerda e à direita. Assim, os componentes têm de estar interligados com as casas ao lado, indicando se estão marcados ou não:



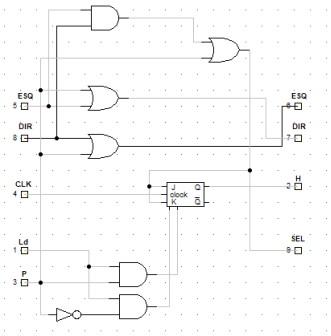
SEL = P ou ESQ e DIR

O valor da casa, quando existir uma jogada, terá de ser trocado, caso esteja selecionada. Como temos flip-flops JK, a troca da casa é quando J=1 e K=1, pelo que basta ligar J=K=SEL. Foi por este motivo que foi escolhido o flip-flop tipo JK na alínea B.

As saídas para a casa da esquerda/direita, dependem da entrada da casa direita/esquerda (para passar o valor), e do valor da casa atual o P:

ESQ(out) = P ou DIR(in)

DIR(out) = P ou ESQ(in)



Todas as expressões são muito curtas, devido a ter-se utilizado um componente iterativo. Pode-se adicionar mais ou retirar, mas o componente mantém-se igual e com a mesma simplicidade.

Alínea D (1 valor)

Melhorar o circuito da alínea C, de modo a validar uma jogada.

Mantenha as 6 casas da alínea anterior, mas adicione uma nova entrada ligada a um "*Interactive Input*" para iniciar um novo jogo.

Deve adicionar também dois displays de 7 segmentos, um de cada lado, a indicar quem joga, e o tamanho máximo do segmento que pode jogar. Este display deve ficar em branco (sem nenhum LED ligado), no caso de ser o adversário a jogar.

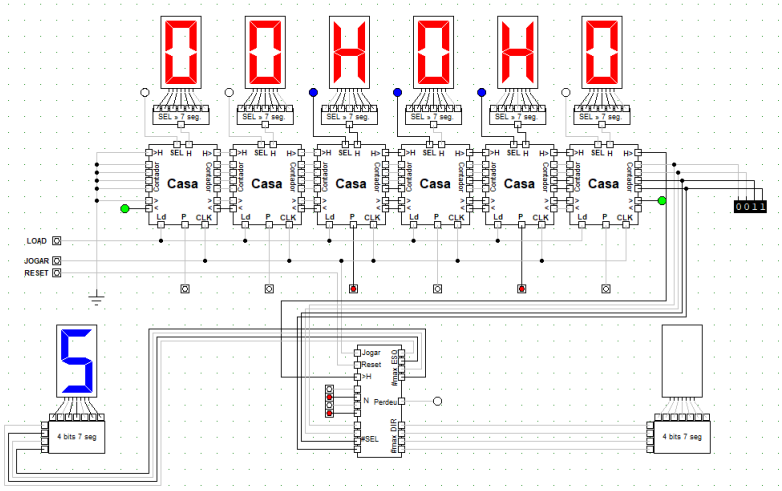
Ilustração:

Na primeira jogada aparece um 5 ao jogador da esquerda, o primeiro a jogar, dado que o tabuleiro tem 6 casas, sendo a jogada a primeira o maior segmento é 5 (6-1). Após a primeira jogada, aparece um 4 para o jogador da direita, dado que sendo na segunda jogada, o maior segmento é 4 (6-2).

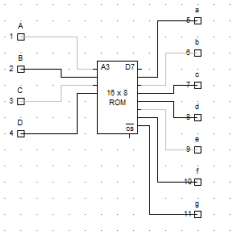
Uma jogada é inválida se a seleção não conter um X, que no nosso caso é o valor H, ou ter uma dimensão superior ao permitido. Nesse caso deverá ligar o estado de jogada inválida e não mais alterar o tabuleiro de jogo nem mover a vez a jogar, devendo permanecer ligado apenas o display do jogador que ganhou.

O botão de início de jogo permitirá sair deste estado e iniciar um novo jogo.

Precisamos de um componente para controlar o jogo, que se tem de configurar conforme o número de casas. Precisamos também de um componente para mostrar o tamanho máximo do segmento, ou ficar apagado se não for o jogador a jogar. Por outro lado, temos de saber quantas casas estão selecionadas, pelo que tem de se passar a contagem no componente iterativo:



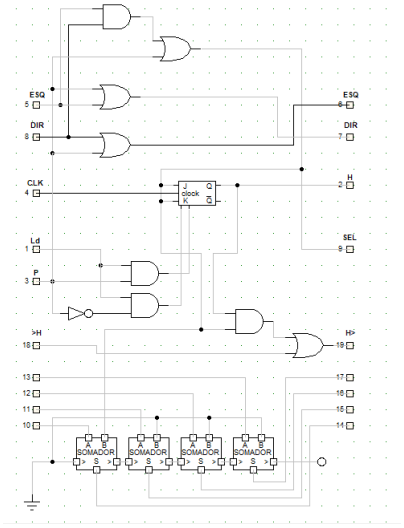
O conversor de 4 bits para 7 segmentos, bastará colocar no valor 0, os LEDs todos apagados, em vez de mostrar o 0, já que esse dígito não é um comprimento válido de um segmento. Uma solução simples, sem contas, será a utilização de uma memória:



O componente da casa, tem agora um contador de 4 bits, com o número de casas selecionadas. Em cada componente, o contador é igual a:

Contador(out) = SEL + Contador(in)

Ou seja, recebe o valor do contador até ao momento da casa vizinha, e soma 1 caso a casa atual esteja selecionada. Para somar dois números, pode-se ter um componente somador iterativo, das atividades formativas:



Notar que B é igual a SEL. Apenas o primeiro bit, já que os outros bits de B são todos 0. Em A está Contador(in), e em S fica Contador(out)

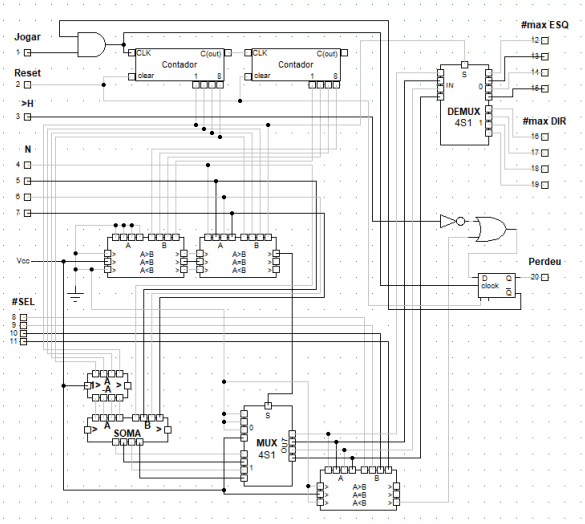
Falta agora o controlador de jogo. Tem de receber:

- #SEL - Contador(out) da última casa – para ver se é um lance válido
- Jogar – para controlar o número de jogadas
- Reset – para inicializar o jogo
- N – número de casas que existe (configuração), para saber quando o lance é válido

Controla:

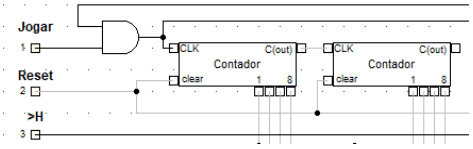
- Máximo de jogadas à esquerda – e indicação de jogador a jogar
- Máximo de jogadas à direita – e indicação de jogador a jogar
- Perdeu – indicação de jogo acabado, não sai mais deste estado

A implementação foi feita com base em componentes digitais básicos, para cada função:



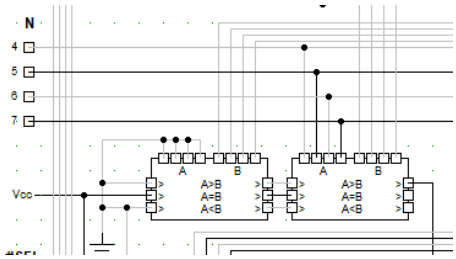
A aparente complexidade, fica simples se vista separadamente:

- Sistema para manter o número de jogada:



Número de 8 bits, com dois contadores de 4 bits cada, das atividades formativas. O sinal de jogar é que faz avançar o contador. O Reset limpa o contador de forma assíncrona. Os contadores estão interligados, pelo que o número da jogada, pode ser utilizado para calcular o segmento máximo.

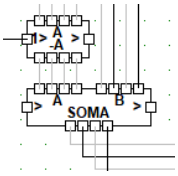
Este número é relevante apenas enquanto o número de casas, é maior que o número de jogada. É o que é realizado por dois componentes comparadores iterativos, parecidos com as atividades formativas:



A saída dos comparadores irá indicar se devemos retirar ao tabuleiro o número da jogada, para obter o segmento máximo, ou se o segmento máximo deve ser 1.

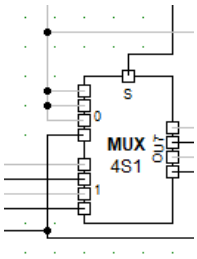
Para subtrair a N (número de casas), o valor do número da jogada (mais 1, já que o contador começa em 0), há que utilizar os circuitos de soma, iterativos, mas a necessitar apenas de 4 bits já que o número de casas é menor que 8.

Para evitar estar a somar 1, mais vale colocar logo N-1 em vez de N, na configuração:



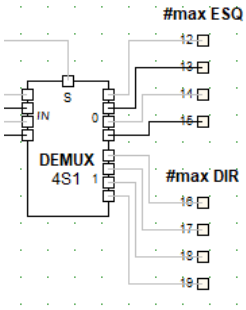
O primeiro componente, faz o complemento para 2 do contador. Ou seja, complementa todos os 4 bits, e soma 1 (o carry que vem da esquerda), colocando na entrada A do somador, ficando com o número negativo. Na entrada B chega o valor de N-1, pelo que o resultado da soma é a expressão N-1 - #jogada, que é igual ao tamanho máximo do segmento.

Há que identificar o número da jogada com base na comparação do contador com o número de jogadas, sendo esse valor enviado para um multiplexer:



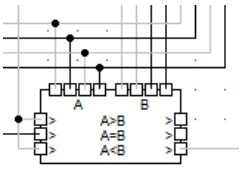
Se S=0, então estamos na situação em que o segmento máximo é 1, sendo a entrada a 0 enviada para a saída com o valor 1. Se S=1 então N é ainda maior que o número da jogada, pelo que a entrada 1 (o resultado de N-1-#jogada), é colocado na saída. Assim, o resultado final deste multiplexer é o número máximo do segmento, já a considerar todas as situações.

Este valor deve ser mostrado no display de 7 segmentos correto, e para tal é preciso um demultiplexer:



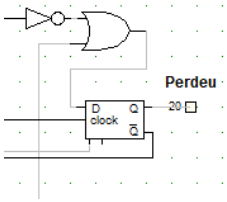
Em IN está o resultado do multiplexer anterior, com o segmento máximo. Em S está a paridade do contador, ou seja, o primeiro bit do contador, que vai alternando em cada jogada. A saída é enviado o número máixmo, ou 0 caso a saída não seja igual a S.

O teste para ver se o segmento máximo é violado, utiliza novamente o componente comparador:



Em A está o resultado do multiplexer, com o valor máximo permitido. Em B está o #SEL, com o somatório das casas selecioandas. Se A<B, então há uma violação, e o jogo deve ser interrompido com vitória do adversário.

O sistema para saber se o jogo já acabou, tem de utilizar mais um flip-flop:



O OR é ativado se não existir segmento selecionado (H=0), ou se o segmento máximo for violado. Nesse caso flip-flop é colocado a 1, significando que o jogador perdeu. O jogo não avança mais, já que o /Q deste flip-flop está em conjunção com o jogar. O Reset limpa este flip-flop permitindo que o jogo retome.