universidade de aveiro
theoria poiesis praxis

Relatório - Master Mind

Turma P17 - Diogo Daniel Soares Ferreira e Eduardo Reis Silva
Universidade de Aveiro - Laboratórios de Sistemas Digitais

25 de Maio de 2015

Conteúdo

Especificações do Sistema

Para este sistema, é pedido que o sistema gere um número aleatório de quatro algarismos decimais(de 0 a 9), não o mostrando ao utilizador, que o deve tentar adivinhar. Através dos botões disponíveis na FPGA (*keys*), o utilizador deve tentar adivinhar o número gerado, incrementando ou subtraindo o valor de cada algarismo, mostrado através dos *displays* hexadecimais. Quando estiver certo da sua aposta, carrega noutro botão disponível para confirmar a sua escolha.

O número máximo de tentativas será 99, o que será mostrado também nos *displays* hexadecimais quando o utilizador acertar no número gerado ou quando chegar ao número máximo de tentativas. Em cada escolha, caso o algarismo escolhido esteja igual ao algarismo no mesmo local do número gerado, o número de dígitos certos incrementa um valor, e o de dígitos errados subtrai também um valor. Estes dois sinais também serão mostrados nos *displays*, para o utilizador saber quando acerta em mais um número, ou quando erra. Quando acerta em todos os números, os *displays* que mostram os algarismos inseridos devem piscar a uma frequência de 2 Hz.

Arquitetura detalhada do sistema

Tal como demonstrado no diagrama de blocos do anexo, o sistema possui quatro entradas, três nos botões($Keys$) e um no interruptor(SW) e oito saídas (todas através dos displays hexadecimais). Os blocos preenchidos com a mesma cor são semelhantes (não necessariamente iguais). Analisando o diagrama começando pela esquerda, podemos ver ligado a cada botão de entrada, um *Debouncer*, para evitar possíveis conflitos mecânicos de leituras múltiplas. Também podemos ver um *clock*, que pode ser interpretado como um sinal de entrada, mas que na prática

já vem implementado na *FPGA*. Esse *clock* , ligado a dois *Frequency Dividers* (a *FPGA* tem um *clock* que executa a uma frequência de 50 Mhz, o que é demasiado rápido algumas entidades), irá sincronizar todo o sistema. Também o *reset*, implementado no interruptor *(SW(0))* quando pressionado, irá levar o sistema à sua posição inicial.

Mais acima, temos uma máquina de estados, *Random Number*, com duas saídas. A saída *count*, deverá indicar ao contador (acima de *Random Number*) quando contar. Quando a entrada *stop_signal* for ativada, o *count* fica desativo até o sinal de *Reset* estar ativo. Quando é ativada a entrada *Reset*, para além de count ficar ativo novamente, se não estiver, a saída *ResetOut* irá também ficar ativa, indicando ao contador para reiniciar a contagem. Esse contador (a vermelho), deverá ser instanciado com o número máximo sendo 9999, voltando depois a zero e reiniciando a contagem. Também deverá ter como entradas *enable*, *clock* e *reset*, para assegurar o seu bom funcionamento.

A entrada *Key(0)* deverá estar ligada a um contador de 0 a 3, sendo incrementado sempre que *Key(0)* for ativo. A saída do contador deverá ligar-se a um *Demultiplexer*, que sinalizará a saída onde o utilizador pretende alterar o dígito.

Para cada dígito, teremos contadores de 0 a 9, com *reset* sendo *Key(3)*, o botão de confirmação do número desejado. Para alterarmos o número, podemos somar uma unidade, no botão *Key(1)*, ou subtrair, no botão *Key(2)*. Logo, os contadores têm entradas para somar ou subtrair a contagem atual. Essas entradas estão ligadas com uma porta *and*, entre as entradas *Key(1)* e *Key(2)*, e as saídas do *Demultiplexer*, para garantir que apenas somamos ou subtraímos um dígito de cada vez.

Ligados à saída dos contadores teremos um *Register*, sendo o *clock* o botão *Key(3)* e sendo *reset* o interruptor *SW(0)*. Assim, sempre que o utilizador confirmar a sua escolha, o número atual escolhido vai ser a saída do *Register*. Caso contrário, o número anterior permanecerá na sua saída.

A entidade *Compare* (a cor-de-rosa) irá comparar bit a bit as suas entradas, que neste caso serão as saídas do *Register* e as saídas do *Counter* que gera o número aleatório. A entidade tem como saída quatro sinais ("1"se cada *bit* dos números *"Random"* e do utilizador são iguais, "0"se contrário), entre os dois vetores de entrada. Também há um contador ligado ao *Key(3)*, para contar o número de tentativas que o utilizador introduz.

A entidade *checkEnd* irá receber como entrada o número de tentativas do utilizador, o *clock* e o botão de *reset (SW(0))*. A entidade deverá enviar o número de dígitos certos e o número de dígitos errados. Caso o utilizador acerte no número antes das 99 tentativas a entidade deverá mostrar o valor de dígitos corretos sendo 4, o de errados 0, e deverá ativar a entrada *isBlink*, que fará piscar a saída, indicando o número correto. Também deverá mostrar o número de tentativas. Caso o utilizador chegue às 99 tentativas, a entidade apenas deverá mostrar o número de tentativas igual a 99 e esperar pelo *reset*, para o sistema ser reiniciado.

Finalmente, temos os módulos *Blink* e *Binary to Decimal*. Os primeiros fazem piscar o seu resultado a uma frequência recebida pelo *clock*. Caso o *enable*

esteja desligado, apenas mostram o resultado sem piscar O *enable* é um *or* entre o *isblink*, que indica se o utilizador acertou, e a entrada correspondente do *demultiplexer*, para piscar sempre que estiver no modo *set*. A entidade *Binary to Decimal* converte a sua entrada binária para ser mostrada corretamente nos *displays* hexadecimais.

Nota: O diagrama de blocos encontra-se anexado no final deste relatório.

Arquitetura faseada do desenvolvimento e validação

No nosso planeamento, os módulos precisam de estar todos individualmente feitos até dia 8 de Maio. Desde esse dia, até ao dia 10, procederemos com todos os testes e validações necessários para simular o funcionamento do sistema. No máximo, dia 15 de Maio, o sistema tem de estar completamente funcional, com todos os testes realizados e a funcionar corretamente na *FPGA*, para podermos ter tempo para nos preparar caso algo corra fora do previsto.

Divisão do trabalho entre os dois elementos do grupo

Para não estar a beneficiar ninguém, ou a prejudicar, decidimos dividir o trabalho de forma aleatória. O Eduardo Silva vai ficar com as entidades:

- *Debouncer*
- *RandomNumber*
- *Counter99,Counter3,Counter9999*
- *Demultiplexer*

O Diogo Ferreira vai ficar com as entidades:

- *Shift_Register*
- *Compare*
- *checkEnd*
- *Blink*
- *Binary to Decimal*

De notar que a maior parte destas entidades já foram construídas, ou parcialmente construídas em aulas, pelo que só serão necessários alguns ajustes de forma a se adaptar ao sistema. Cada aluno deverá realizar as suas *TestBenches* às entidades nucleares de forma a certificar-se de que estão a funcionar corretamente. Não deverão apenas fazer *TestBenches* à entidade criada, mas também às ligações entre as entidades, de forma a garantir a sanidade do sistema.

Manual do utilizador

O utilizador tem à sua disposição cinco botões onde pode modificar o comportamento do circuito. No primeiro botão(KEY(0)), pode selecionar o dígito que deseja modificar, para com o os dois botões seguintes incrementar(KEY(1)) ou diminuir(KEY(2)) o valor desse dígito, de 0 até 9. Quando estiver confortável com a sua escolha, o utilizador pode premir o botão KEY(3) para confirmar a sua escolha e verificar quantos dígitos acertou nos *displays* hexadecimais. Caso queira voltar ao início , gerando um novo número aleatório e repondo as tentativas para zero, pode ligar o interruptor SW(0) e carregar no botão *KEY(3)*, repondo o sistema no seu estado inicial.