QUESTÕES COLOCADAS PELOS ALUNOS ENTRE 6 E 7 DE DEZEMBRO RELATIVAMENTE AO TRABALHO 1

Questão 1

Ao falarmos com colegas da cadeira sobre o trabalho prático foi-nos dito que a implementação VHDL tinha que ser feita recorrendo apenas a portas lógicas AND e/ou OR. Nós fizemos a nossa implementação em VHDL recorrendo apenas ao uso do operador XOR. Temos que criar uma entidade para o XOR com AND e OR e instância-la ou podemos utilizar o operador 'XOR' do VHDL?

Podem efectuar a implementação em termos apenas de gates X-OR 2.
A gate X-OR 2 pode ser considerada como implementando uma operação booleana elementar.
Contudo, como pretendo uma descrição estrutural, as gates X-OR 2 têm que ser explicitamente implementadas por vocês.
Vejam como o fiz para o código do *population counter* que vos forneci como exemplo.

Questão 2

Quero esclarecer algumas questões sobre o projeto de ACA:
1) Se por exemplo tivermos o encoder em modo bit serial e o checker em modo paralelo, o encoder terá 1 bit de entrada, mas depois poderá ter os 24 bits de saída para ligar ao checker? Ou terá apenas 1 bit de saída e o checker é que vai ter que guardar cada bit recebido antes de processar.
2) A entrega dos documentos do projeto será feita pelo elearning?

1 - Terá apenas um bit de saída. Leia com atenção as especificações colocadas no documento tutorial que disponibilizei. O *encoder* e o *checker* são projectos separados. As versões bit serial são para a transmissão/recepção de mensagens e as versões paralelo para o armazenamento de informação. Não faz parte das especificações do trabalho fazer o acoplamento dos dois dispositivos.
2 - Não, será num email dirigido a mim. Indicarei mais tarde as regras do procedimento.

Questão 3

Possuo a seguinte dúvida sobre a implementação em série para o Assignment 1: Podemos utilizar um LFSR (Linear-Feedback Shift Register) para a implementação em série do CRC-8? Ou o professor preferia outros modos de implementação em série? A nossa ideia era implementar o checker na versão em série através de um LFSR.

Pode usar o que quiser, mas o LFSR tem que ser implementado em termos de flip-flops D e lógica adicional.

Questão 4

Obrigado.
Ainda sobre essa implementação, poderemos usar "rising_edge" juntamente com o "process" para a implementação do flip-flop D e da top-level Entity? Ou só podemos utilizar puramente lógica adicional?

O flip flop D a usar é aquele que eu forneci com o exemplo do bit serial *population counter*. Para a sua implementação do bit serial *encoder* em VHDL, só o flip flop D e a ROM de controlo podem ser descritos comportalmente, tudo o resto tem que ser descrito em termos de combinações de flip flops e de gates que implementem a lógica elementar.

Questão 5

Tenho uma questão relativamente à aula de reposição de quarta feira, 9 de dezembro. A que horas será a aula para a turma P2? E a sala será a mesma das aulas práticas?

As turmas P1 e P2 serão entre as 16h e as 18h. As salas são as mesmas das aulas práticas.
Não se trata, no entanto, de uma aula de reposição, mas de uma sessão tutorial para tirar dúvidas sobre o trabalho.

Questão 6

Nós estamos a implementar o checker-crc em série. Dado a entrada de 24 bits(16 + 8), conseguimos calcular o resto(8 bits). No entanto, a saida do checker apenas deve ter 1 bit (bit error), para isso pensamos em fazer um comparador do resto com "00000000". A nossa questão é se podemos usar lógica combinacional para o representar ou o mais correto seria usar um OR com 8 entradas em que a saída corresponde ao bit error?

O recurso a uma gate OR 8 é a solução mais barata.

Questão 7

No desenvolvimento do trabalho, conseguimos 2 simplificações: uma com 38 portas xor e 6 de atraso; e uma outra com 39 portas xor e 5 de atraso.
Qual das duas será a melhor opção, ou seja, o que é mais revelante, o número de portas xor ou o atraso?

A melhor solução será um compromisso entre as duas, função da situação de aplicação.
Descrevam ambas na apresentação e implementem uma delas.

Questão 8

Nós estamos a tentar fazer o encoder em série, mas o resto está a dar mal pode dizer o que estamos a fazer mal?
Em anexo vai o código do encoder e o ficheiro da simulação que estamos a usar.

Lamento, mas é vossa responsabilidade resolver esse tipo de questões.

Questão 9

Segundo o enunciado, para a entrega o professor pede a especificação do encoder e do checker mais a prova do seu funcionamento através do simulador do quartus. Quanto à prova de funcionamento pretende que nós entreguemos os ficheiros gerados (ficheiros .vwf)?

Não, só quero os ficheiros VHDL. A prova do funcionamento será feita por vocês durante a apresentação do trabalho.

Questão 10

No pdf que colocou no elearning com as questões dos alunos mencionou que está a pensar calenderizar as apresentações para a próxima 5a e 6a feira, dias 10 e 11. Pode confirmar se vão mesmo ser nesses dias ou se será depois noutra altura?
Dava-nos jeito saber com antecedência para organizarmos a nossa semana e sabermos se temos que nos deslocar a Aveiro nesses dias.

Lamento, mas não posso confirmar ainda as datas definitivas em que a avaliação terá lugar.
6.ª feira será com uma forte probabilidade, mas 5.ª feira não é certo.

Questão 11

Encontramo-nos com o Encoder feito e cremos que está completo. A discutir como realizar o checker, tivemos uma ideia mas que não temos a certeza absoluta se cumpre o objetivo.
A ideia seria:
Quando são recebidos os 24 bits (16 da data, mais 8 do resto), reutilizávamos o encoder, o qual nos daria o resto correto da data recebida no checker, e comparavámos com o resto recebido inicialmente. Basicamente, confirmávamos que o resto recebido no checker era igual ao resto real calculado no momento, e fazendo um xor de ambos, obteriamos 0 se for exatamente igual, confirmando que nem a data nem o crc foram corrompidos a meio da transmissão. As nossas dúvidas surgem devido ao facto que na realidade existe uma certa repetição do processo e pode um pouco redundante. Pode-nos dar a sua opinião sobre a ideia?

A vossa ideia tem razão de ser, mas a forma como a exprimem é confusa e está errada.
Se um dos componentes do checker for um encoder, duas abordagens são possíveis:
1 - O encoder (diferente do anterior) vai processar 24 bits, com origem presumível na informação original (16 bits) e no resto anteriormernte calculado (8 bits). Neste caso, o resto que é agora calculado, se for diferente de zero, sinaliza um erro.
2 - O encoder (semelhante ao anterior) vai processar 16 bits, com origem presumível na informação original. Neste caso, o resto que é agora calculado, é comparado com o resto presumivelmente anteriormente calculado, se forem diferentes, um erro é sinalizado.
A repetição no processo de verificação é inevitável porque é sempre necessário o cálculo de um novo resto para que se possa concluir algo.