PCS-3225 Sistemas Digitais II — 29 de maio de 2024 - v1

Códigos de Hamming

Aula 19 - Exercício

Bruno Albertini, Edson Gomi e Ricardo Lera

O objetivo deste exercício é demonstrar o funcionamento dos códigos de Hamming para detectar e corrigir erros.

Vamos utilizar o módulo de memória `ram74` que simula o efeito de perda de dados (por exemplo, devido à linha de transmissão). Esta memória escreve normalmente, porém quando esta memória é lida, ela adiciona **até 1 bit de erro** na palavra que será retornada. Isto não altera o dado salvo.

Nosso objetivo é implementar o módulo `ham74` que detecte a presença de um erro no dado, e também corrija este erro. Sabemos que a correção sempre será possível usando o algoritmo Hamming(7, 4), como o erro máximo é 1 bit. Portanto, vamos usar palavras de 7 bits no formato: $d_4d_3d_2p_3d_1p_2p_1$ (*mais* significativo à esquerda) sendo p os bits de paridade.

No edisciplinas foi disponibilizado o *template* `ham74.v`. É necessário completar:

- O bloco de detecção de erros que define um sinal de erro para cada bit de paridade.
- O bloco de correção de erros que define a saída.

  Questão auxiliar: De quais sinais este bloco depende?

Envie para o juiz somente o módulo `ham74`. Para testar seu projeto, utilize o módulo de memória ram74.v disponível no edisciplinas.

Detalhes de Sintaxe:

- Concatenação em Verilog, representada por {} (chaves), realiza a concatenação do bit *mais* significativo para o *menos* significativo. Por exemplo: `{2'b01, 2'b10}` gera o binário `4'b0110`.
- Também é possível negar um único bit ou sequência de bits em uma concatenação usando o símbolo ∼. Por exemplo, se (a, b, c) são bits com valor 1, então `{a, ~b, c}` gera o binário `3'b101`.
- Sempre utilize o atribuidor síncrono `<=` no bloco case, para assegurar que seu programa responderá adequadamente.