PCS3335 - Laboratório Digital A - Experiência 2

por Bruno de Carvalho Albertini

28/02/2024

Na experiência 2 você projetará com sua dupla o seu primeiro circuito digital combinatório.

Introdução

Experiência 2

Esta experiência tem planejamento. O planejamento é feito de duas maneiras:

- Arquivos enviados para o juiz eletrônico;
- Planejamento dos testes a serem realizados.

a) Funções

Os circuitos combinatórios nativamente implementam funções. As funções obviamente atuam no domínio binário, porém funcionam exatamente como as funções matemáticas: mapeiam um conjunto de entradas em um conjunto de saídas através de transformações nos dados. Nesta experiência, você implementará 6 funções combinatórias.

As operações possíveis em binário são equivalentes às portas lógicas. Os símbolos matemáticos são $\wedge$ para a função "e", $\vee$ para a função "ou", $\oplus$ para a função ou-exclusivo e $\neg$ para a função "inversora" (também conhecida como "não" ou "*not*").

As funções sobre as palavras que utilizaremos nesta experiência são: o deslocamento para a direita (*shift right*), $shr^n(x) = x \gg n$, onde a palavra é deslocada para a direita e os valores a esquerda são preenchidos com zero, e a rotação para a direita (*rotate right*), $rotr^n(x) = (x \ll n) \vee (x \gg (w - n))$, onde a palavra é rotacionada para a direita.

Para esta experiência, consideraremos todas as palavras (x, y e z) com tamanho $w = 32b$, assim como o resultado das funções. Quando uma operação em binário é aplicada em duas palavra, usamos o conceito *bitwise*, ou seja, a operação é aplicada entre os bits de mesmo índice de cada palavra, gerando uma terceira palavra.

Exemplo: `1001`$\vee$`0010`$=$`1011`

A lista de funções a serem implementadas pode ser vista abaixo. O nome que precede a função é o nome do módulo VHDL.

- `ch`: $Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z)$
- `maj`: $Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$
- `sum0`: $\sum_0^{\{256\}}(x) = rotr^2(x) \oplus rotr^{13}(x) \oplus rotr^{22}(x)$
- `sum1`: $\sum_1^{\{256\}}(x) = rotr^6(x) \oplus rotr^{11}(x) \oplus rotr^{25}(x)$
- `sigma0`: $\sigma_0^{\{256\}}(x) = rotr^7(x) \oplus rotr^{18}(x) \oplus shr^3(x)$
- `sigma1`: $\sigma_1^{\{256\}}(x) = rotr^{17}(x) \oplus rotr^{19}(x) \oplus shr^{10}(x)$

Você deve entregar em um único arquivo VHDL sua implementação combinatória para as funções acima. As entidades para as funções estão abaixo.

```
entity ch is
    port (
        x,y,z: in  bit_vector(31 downto 0);
        q: out     bit_vector(31 downto 0)
    );
end ch;
entity sumo is
    port (
        x: in  bit_vector(31 downto 0);
        q: out bit_vector(31 downto 0)
    );
end sumo;
entity sum1 is
    port (
        x: in  bit_vector(31 downto 0);
        q: out bit_vector(31 downto 0)
    );
end sum1;
```

```
entity maj is
    port (
        x,y,z: in  bit_vector(31 downto 0);
        q: out     bit_vector(31 downto 0)
    );
end maj;
entity sigmao is
    port (
        x: in  bit_vector(31 downto 0);
        q: out bit_vector(31 downto 0)
    );
end sigmao;
entity sigma1 is
    port (
        x: in  bit_vector(31 downto 0);
        q: out bit_vector(31 downto 0)
    );
end sigma1;
```

O nome do arquivo não importa, mas deve conter as entidades acima e as arquiteturas correspondentes, com a sua solução. Você deve enviar este arquivo para o juiz eletrônico individualmente antes da próxima aula. Lembre-se que este projeto é combinatório e portanto não deve usar `process`.

Depois de enviar, você pode montar um projeto no Quartus contendo sua solução. Junte com o arquivo do codificador que fez na Experiência 1 e monte um projeto para placa com as seguintes características:

A montagem do projeto pode ser feita em sala, mas você certamente terá uma experiência de laboratório melhor se preparar o projeto antes. A preparação exige o Quartus, que pode ser baixado no site da Intel, obtido com os técnicos do LabDigi ou você pode usar o laboratório nos períodos em que não há aula.

- Entrada de dados pelas chaves 7-0 (8 bits), os bits devem ser replicados e transformados para formar os 32b. Exemplo: se as chaves estiverem com `10101110`:
  - x = `10101110 10101110 10101110 10101110` (repete-se quatro vezes as chaves)
  - $y = \neg x$ = `01010001 01010001 01010001 01010001` (inverso de x)

  - $z = rev(x)$, onde $rev(x)$ espelha x, então z = `01110101 01110101 01110101 01110101`

- As chaves 9-8 escolhem qual par de funções deve ser mostrado nos *displays*: `00`: mostra o valor das chaves, `01`: `sum0` ou `sum1`, `10`: `sigma0` ou `sigma1`, e `11`: `ch` ou `maj`.

- O botão 3 (`KEY(3)`) mostra a variação da função. Exemplo: se as chaves 9-8 estiverem em `11` e o botão não estiver pressionado, mostra `ch`, mas pressionado mostra `maj`.

- Os LEDs 9-8 devem ficar apagados, os LEDs 7-0 mostram os 8b mais significativos do resultado e os *displays* mostram os 24b menos significativos do resultado, em hexadecimal.

Arquive o seu projeto, gerando um arquivo `.qar`. Planeje como pretende testar o seu circuito no dia da experiência. Descreva como pretende testar no e-Disciplinas, na tarefa correspondente (em dupla). No dia da experiência, restaure o seu `.qar`, sintetize e configure a placa FPGA e faça os testes seguindo seu roteiro do planejamento. Quando estiver satisfeito, chame o seu professor para ser avaliado.

Para a parte prática, a ser realizada no dia da experiência, é útil entender como associar as saídas do seu projeto com pinos do FPGA. Explore o *Pin Planner* do projeto base fornecido na Experiência 1 e atente-se aos manuais disponíveis no site do LabDigi, no e-Disciplinas ou no site do fabricante da placa.

Link LabDigi

O relatório desta experiência, a ser preenchido no dia da parte prática durante a aula, deve ser um relato dos seus testes.

Checklist

☐ Atividades pré-aula:

  ☐ Ler este enunciado

  ☐ Implementar as funções em VHDL

  ☐ Implementar um *testbench*

  ☐ Enviar sua solução para o juiz (individual)

  ☐ Planejar os testes a serem realizados no laboratório (dupla)

☐ Opcionalmente montar o projeto no Quartus

☐ Atividades durante a aula:

  ☐ Montar o projeto no Quartus (se não montou)

  ☐ Atribuir os pinos corretamente

  ☐ Sintetizar o projeto e configurar a placa

O teste adequado garante que sua solução funciona.

Todos os planejamentos devem ser enviados até 24h antes da sua próxima aula.

Se optar por fazer o projeto no Quartus no dia da experiência, sugerimos que revisite previamente o conceito de multiplexador em VHDL.

- ☐ Executar os testes planejados
- ☐ Corrigir eventuais erros
- ☐ Escrever o relatório da experiência