PCS-3225 Sistemas Digitais II — Março de 2024 - v1

Algoritmo Double Dabble

EP 01 — Bruno Albertini, Edson Gomi e Ricardo Lera

Os números BCD (*binary-coded decimal*) são uma maneira amigável a humanos de representação numérica. Um dígito BCD consiste em 4b, correspondente direto do número em binário. Por exemplo, para representar o número 2 em binário, escreve-se `0010`, para representar o número 4 escreve-se `0100`. As mesmas regras de composição da numeração posicional se aplicam, portanto o número 42 será representado em BCD como `0100 0010`.

O algoritmo *double-dabble* é um conhecido algoritmo para converter um número binário em um número BCD. O número 42 em binário é `00101010`, mas queremos representá-lo em BCD como `0100 0010`. O algoritmo funciona da seguinte forma:

- Aloca-se um registrador R_{BIN} de W bits para o binário, zerado;
- Determina-se o tamanho máximo dos BCD (cada dígito usa 4b);
- Aloca-se um registrador R_{BCD} de N bits para os BCDs;

1 Desloca-se os registradores R_{BCD} e R_{BIN} um bit para esquerda:

   – O LSB do R_{BCD} recebe o MSB do R_{BIN};
   – O LSB do R_{BIN} recebe qualquer valor.

2 Para cada BCD empacotado no R_{BCD}:

   – Se >= 5 soma-se 3;
   – Senão mantém-se o valor.

3 Repete-se 1-2 exatamente W vezes.

- O valor do BCD estará no R_{BCD}.

EP: Codifique em Verilog o módulo `double_dabble` conforme descrição do módulo abaixo. Você deve enviar somente um arquivo contendo todos os módulos. Abaixo é fornecido um módulo `add3or0` que deve ser usado no mínimo uma vez e no máximo o número de dígitos BCD que o módulo pode suportar. Sua solução deve ser sequencial. O funcionamento do `start` e `done` é igual ao que foi usado nos demais exercícios da disciplina: espera-se o `start` para começar e mantém-se `done` com o resultado correto até que o `start` seja desabilitado.

A submissão é individual para o juiz, via e-Disciplinas. Você tem 10 submissões e vale a maior nota dentre as 10. Sugerimos fortemente que faça o *testbench* e teste sua solução antes de enviar. O nome do arquivo não importa, mas deve conter o módulo em Verilog com a sua solução.

Módulo `double_dabble`:

```
module double_dabble(clk, rst, start, binary, done, bcd);
  parameter W = 18;
  input clk, rst, start;
  input [W-1:0] binary;
  output done;
  output [4*$ceil(W/3.0)-1:0] bcd;
  // Seu codigo
endmodule
```

Módulo auxiliar `add3or0`:

```
module add3or0 (bcd_i, bcd_o);
    input  [3:0] bcd_i;
    output [3:0] bcd_o;
    assign bcd_o = (bcd_i < 5) ? bcd_i
                               : bcd_i + 4'b0011;
endmodule
```

A tabela a seguir mostra um exemplo de execução do algoritmo para o número 42, onde houve correção (soma) somente no passo 5, para a unidade.

#	BCD			BIN	Operação
0	0000	0000	0000	00101010	Alocação
1	0000	0000	0000	01010100	Deslocamento
	0000	0000	0000	01010100	Correção
2	0000	0000	0000	10101000	Deslocamento
	0000	0000	0000	10101000	Correção
3	0000	0000	0001	01010000	Deslocamento
	0000	0000	0001	01010000	Correção
4	0000	0000	0010	10100000	Deslocamento
	0000	0000	0010	10100000	Correção
5	0000	0000	0101	01000000	Deslocamento
	0000	0000	1000	01000000	Correção †
6	0000	0001	0000	10000000	Deslocamento
	0000	0001	0000	10000000	Correção
7	0000	0010	0001	00000000	Deslocamento
	0000	0010	0001	00000000	Correção
8	0000	0100	0010	00000000	Deslocamento
	0000	0100	0010	00000000	Correção