

## 6 Máquinas de Estado Finitas em VHDL

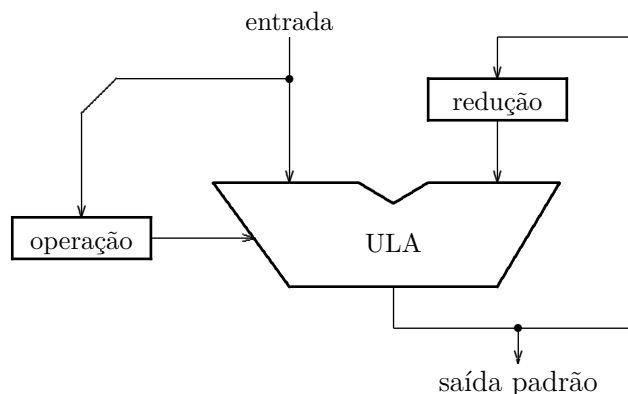
São dois os objetivos deste laboratório: (i) escrever um modelo para uma máquina de estados finita para controlar um circuito que não é trivial; e (ii) verificar, através de simulação, a corretude do seu modelo.

Sua tarefa é projetar, implementar em VHDL, e testar a ME que controla um circuito que efetua a redução de uma sequência de vetores. Por *redução* entende-se que, para a operação associativa  $\Theta \in \{\oplus, \wedge, \vee, +, \times\}$ , e entrada com 8 palavras de 32 bits  $d_i, i \in [0, 7]$ , o circuito computa

$$x = \Theta_0^7 d_i.$$

O *testbench* percorre um vetor com uma sequência de inteiros, composta por palavras de sincronismo, um comando (*oper*) e vetores de dados com 8 palavras, seguido por uma ou mais palavras de sincronismo. Após a leitura da oitava palavra, o circuito exibe a redução daquele vetor na saída padrão.

Você receberá um *testbench* com código que percorre o vetor de entradas e que escreve na saída padrão do simulador, além de um modelo completo do circuito de dados. O sistema está representado no diagrama abaixo. Sua tarefa é implementar a máquina de estados que controla o circuito de dados.



As operações para a redução, e seus respectivos códigos são:  
 0 XOR  
 1 AND  
 2 OR  
 3 SOMA  
 4 MULT  
 A palavra de sincronismo é sync=0xffff.ffff.

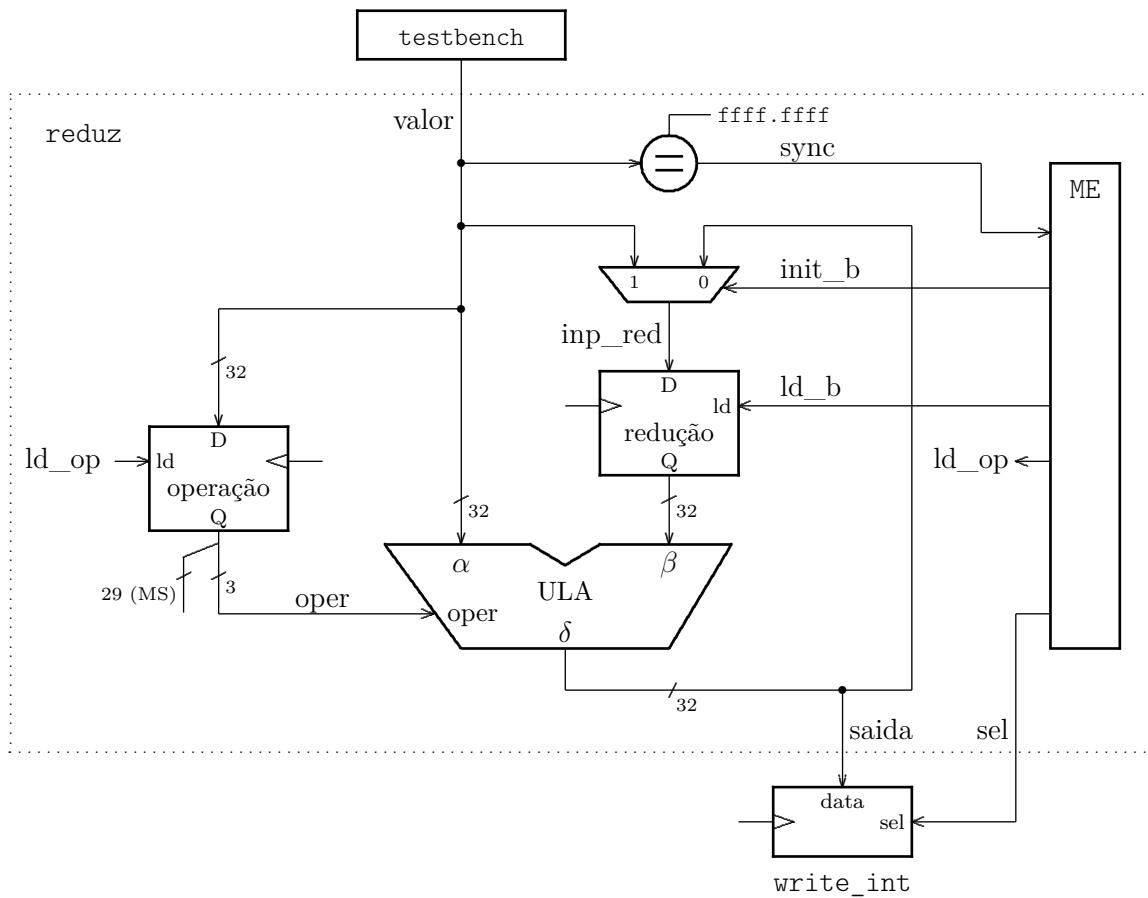
Uma sequência de entrada é algo como:

```

ffff.ffff  ffff.ffff  0000.0003  00ff.00ff  ff00.5544  ...  1122.3344  ffff.ffff  ffff.ffff
  sync      sync      oper +      d0        d1        ...    d7        sync      sync
    
```

A redução deve ser emitida durante o sync após  $d_7$ .

O diagrama abaixo mostra o circuito de dados completo, com todos os sinais de controle que devem ser gerados pela sua ME.



Os arquivos `aux.vhd`, `tb_reduz.vhd` e `packageWires.vhd` contêm os componentes necessários para o seu projeto. Você deve editar `reduz.vhd` e acrescentar o que for necessário para ‘implementar’ a máquina de estados que controla o circuito de dados.

O *script* `run.sh` compila os arquivos VHDL e dispara a simulação. Diga `./run.sh -h` para ver as opções de linha de comando.

Para o vetor de testes no TB, a saída correta da simulação é mostrada abaixo.

```
kipling:/maqEst # ./run.sh
00009d80
99999999
ffffffff
00001111
ffff1111
```

**Da tarefa:**

1. O trabalho pode ser efetuado em duplas;
2. você deve remeter por e-mail para seu professor somente o arquivo com `reduz.vhd`;
3. copie para sua área de trabalho o arquivo com o código VHDL:
  - (a) `wget http://www.inf.ufpr.br/roberto/ci210/vhdl/l_maqEst.tgz`
  - (b) expanda-o com: `tar xzf l_maqEst.tgz`  
o diretório `maqEst` será criado;
  - (c) mude para aquele diretório: `cd maqEst`
4. PLÁGIO NÃO SERÁ TOLERADO. É do interesse de todos que alunos conversem sobre o projeto mas cada dupla deve escrever seu próprio código;
5. assegure-se de que entendeu a especificação antes de iniciar o projeto da ME.

**Referências**

- [RH12] *Sistemas Digitais e Microprocessadores*, R A Hexsel, 2012, Editora da UFPR.
- [PJA08] *The Designer's Guide to VHDL*, Peter J Ashenden, 3a Ed, 2008, Morgan Kaufmann.

EOF
-----