Módulo 6
Sistemas sequenciais

Objectivos

Pretende-se que o aluno transforme uma descrição em língua natural do funcionamento dum circuito sequencial na respectiva máquina de estados e que consiga sintetizar manualmente um circuito sequencial descrito por uma máquina de estado. É igualmente objectivo deste módulo que o aluno saiba descrever circuitos sequenciais em VHDL para posteriormente proceder à sua simulação e à sua síntese para obter uma implementação com um dispositivo de lógica programável (CPLD).

1. Controlador de Portão de Garagem Automático

A. Descrição do Controlador

Pretende-se conceber o controlador de um portão de garagem controlado remotamente. Para abrir, o portão é deslocado da esquerda para a direita por um motor. Para fechar, o portão é deslocado no outro sentido pelo mesmo motor. Para manobrar o portão, o utilizador dispõe de um controlo remoto com um único botão, designado por ***Ac***.

O motor é controlado por 2 sinais: ***Ma*** e ***Mf***, que quando activados isoladamente resultam no movimento pretendido do portão. Estes dois sinais nunca poderão estar activos simultaneamente; o comportamento do motor neste caso seria indefinido. A calha onde o portão se desloca está equipada com dois sensores de fim de curso: ***Ia*** e ***If***. O sensor *Ia* toma o valor lógico '1' quando o portão está completamente aberto, enquanto *If* toma o valor lógico '1' quando o portão está completamente fechado. O motor não deverá tentar abrir o portão quando este se encontra aberto, nem fechá-lo quando este se encontra fechado.

Para maior segurança dos utilizadores o motor está equipado com uma sirene, que deve estar acesa enquanto o portão se desloca. Esta sirene é controlada pelo sinal ***Av***.

Se o portão estiver fechado e *Ac* for premido o portão começará a abrir. Caso esteja aberto e o mesmo sinal for activado então começará a fechar.

O controlador a desenhar deve assumir que não é possível o portão parar a meio do processo de abertura ou fecho. Mas é possível que o utilizador active o sinal *Ac* enquanto o portão se está a deslocar. Neste caso se estiver a abrir deve passar a fechar e vice-versa.

B. Síntese Manual do Controlador

1. Desenhe o diagrama de blocos;
2. Desenhe o diagrama de estados, assumindo que se trata de uma máquina de Moore;
3. Construa a tabela de estados para a máquina de Moore;
4. Atribua uma combinação binária a cada estado e construa a tabela de verdade;
5. Construa a tabela de excitação para *flip-flops* D;
6. Utilizando mapas de Karnaugh, obtenha as expressões minimizadas para as saídas dos dois blocos de lógica combinatória que compõem a máquina de estados.

   TPC. Com base nas expressões minimizadas obtidas anteriormente, descrever em VHDL (estilo fluxo de dados) a arquitectura (e a entidade) de cada um dos dois blocos de lógica combinatória do controlador: o bloco que gera o próximo estado e o bloco que gera as saídas. Descrever em VHDL (estilo comportamental) a arquitectura (e a entidade) do bloco registo de estado, usando o tipo e o número de *flip-flops* considerados na síntese manual. Para terminar, descrever em VHDL estrutural a arquitectura (e a entidade) do controlador, instanciando os 3 blocos que acabou de construir.

C. Simulação em VHDL do Controlador

Fazendo uso do ambiente de simulação em VHDL proporcionado pela ferramenta Active-HDL:

1. Descrever em VHDL, de forma comportamental, o controlador do portão de garagem;
2. Obter um *testbench* em VHDL para testar o controlador;
3. Efectuar a simulação de modo a verificar se o controlador se comporta como é suposto.

D. Síntese e implementação a partir de VHDL do Controlador

Para efectuar esta tarefa pode consultar o anexo A, onde se inclui um breve manual do utilizador da placa PG04 versão 2.

Utilizando a ferramenta de desenvolvimento da Xilinx, fabricante do CPLD XC95108 incluído na placa PG04 versão 2 (v2), pretende sintetizar-se a descrição comportamental (obtida no ponto C) do controlador de portão e posteriormente implementá-la com a

CPLD do PG04 v2. Vamos supor que a descrição em VHDL comportamental se encontra no ficheiro `fsm_funcional.vhd`. Para sintetizar a descrição do controlador e programar (ou seja, configurar) a CPLD de modo a implementar esse controlador, recomendam-se os seguintes passos:

1. No ambiente de desenvolvimento da Xilinx, criar um novo projecto do tipo HDL (suponhamos que o nome escolhido é `ctl_portao`), usando as opções:

   - `Device family:` XC9500 CPLDs
   - `Device:` XC95108
   - `Package:` PC84
   - `Speed grade:` -15
   - `Top level module type:` HDL
   - `Synthesis tool:` XST (VHDL)

2. Criar o projecto vazio e adicionar *a posteriori* o ficheiro VHDL com a descrição comportamental do controlador (`fsm_funcional.vhd`);

3. Editar a localização (ou seja, os pinos) desejada para os sinais de entrada e de saída do controlador quando implementado na CPLD. Para isso utiliza-se o utilitário `PACE`, que pode ser executado através da opção `Assign Package Pins` no ambiente de desenvolvimento da Xilinx. Esta fase vai gerar um ficheiro com extensão UCF (***U**ser **C**onstraint **F**ile*). Ao executar o `PACE` opte por responder `YES` à primeira pergunta colocada e atribua graficamente a localização dos sinais do controlador aos pinos da CPLD, usando o seguinte mapeamento:

Sinal do Controlador	***Pino da CPLD***	***Acesso ao pino no PG04 v2***
reset	P74	Botão de Reset
clk	P33	*Interruptor 7*
Ac	P39	*Interruptor 2*
I_f	P40	*Interruptor 1*
Ia	P41	*Interruptor 0*
Av	P82	*LED 2*
Ma	P83	*LED 1*
Mf	P84	*LED 0*

   A atribuição processa-se arrastando cada sinal da lista `I/O pins` para o pino adequado no esquema do encapsulamento da CPLD, o qual se encontra na janela `Package pins for XC95108-PC84-15`. Guarde o ficheiro no final da atribuição e feche o utilitário `PACE`.

4. Gerar o ficheiro de configuração da CPLD através da opção `Generate Programming File`, mas para isso deve ter-se seleccionado/marcado a entidade (ficheiro) VHDL que descreve o controlador. Esta tarefa desencadeia a execução das tarefas de síntese, tradução e *fit*, antes de gerar a configuração.

5. Inspeccionar os resultados da síntese, sobre a forma de esquemático, através da opção `View RTL Schematic`. O resultado é o esperado, ou seja, a estrutura gerada pela síntese segue o esquema a 3 blocos da máquina de estados tipo Moore?

6. Simular de novo o controlador, agora com o ficheiro `ctl_portao_timesim.vhd` gerado quando se executa o comando `Generate Post-fit Simulation Model`. Deste modo, em vez de se usar `fsm_funcional.vhd` mais o ficheiro de *testbench* (suponhamos que se chama `test_fsm.vhd`), efectua-se a simulação com o mesmo *testbench* mais o ficheiro `ctl_portao_timesim.vhd`. Este ficheiro descreve o controlador de forma estrutural e contém informação temporal, de acordo com o processo de síntese. Editar o ficheiro para constatar como o controlador é descrito estruturalmente com recursos da CPLD, tais como *buffers*, *flip-flops*, inversores, ORs e ANDs.

7. Comparar as formas de onda obtidas nas duas simulações. Qual a conclusão retirada da comparação?

   TPC. Repetir os passos 1 a 7 anteriores, mas usando como descrição do controlador de portão a versão estrutural obtida no TPC do ponto B (síntese manual do controlador). Que conclui quanto às implicações que as diferenças entre uma descrição estrutural e outra comportamental podem trazer ao resultado da síntese?

8. Configurar a CPLD do PG04 v2 usando os seguintes passos:

   - Colocar os *jumpers* JP6 e JP7 do PG04 v2 no estado ON, para se poder programar a CPLD pelo cabo paralelo, e confirmar que o PG04 v2 está ligado ao PC através dum cabo paralelo;
   - Executar o utilitário `Xilinx JTAG Programmer`;
   - O `JTAG Programmer` deve detectar que existe na cadeia de JTAG, entre os sinais TDI e TDO, uma CPLD XC95108;
   - Clicar duas vezes com o rato sobre o símbolo da CPLD presente na cadeia JTAG detectada e especificar o ficheiro de configuração JEDEC (extensão .JED) a enviar para a CPLD. Escolha o ficheiro JEDEC gerado no ponto 4;
   - Clicar uma vez com o rato sobre o símbolo da CPLD presente na cadeia JTAG, de modo a que este símbolo fique seleccionado, e em seguida seleccionar o comando `Program` do menu `Operations`. Com a opção `Erase before programming` seleccionada, mandar programar a CPLD. Se não ocorrerem erros, a CPLD está apta a funcionar, implementando o controlador de portão.

   Pode finalmente verificar o correcto funcionamento do controlador de portão, implementado na CPLD do PG04 v2, usando os interruptores 0 a 2 e os LEDs 0 a 2 para gerar as entradas (Ia, I_f e Ac) e visualizar as saídas (Mf, Ma e Av), respectivamente. As saídas comportam-se de acordo com a máquina de estados inicialmente projectada? Como se comportam as saídas quando se colocam os interruptores ligados às entrada com os seguintes valores: Ia=0 (OFF), I_f=1 (ON) e Ac=1 (ON)?