

PROJETO 6: CONTADOR BINÁRIO DE QUATRO BITS NO MODO AHDL.

Deseja-se projetar um contador binário síncrono de quatro bits, baseado em flip flops do tipo D, e com as seguintes operações:

- Possui uma entrada de carregamento paralelo, operando no nível positivo (load). Isto significa que quando load = 1 os valores das quatro entradas paralelas serão transferidos para as correspondentes saídas dos flip flops. Esta entrada deverá ser prioritária sobre as demais.
- Possui uma entrada de habilitação de contagem (ena), que quando em nível alto, libera o contador para contar crescentemente.
- Possui entrada clear para zerar o contador.

A figura 6.1 resume o projeto

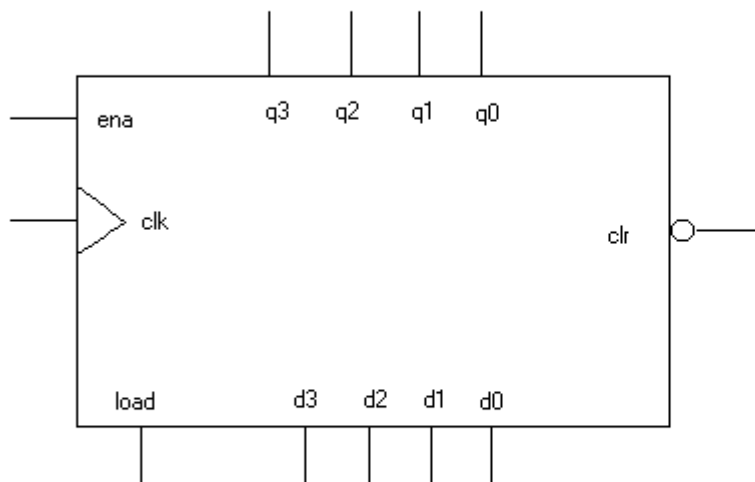


Fig.6.1 Contador binário decrescente com carregamento paralelo

O desenvolvimento deste projeto poderá ser realizado no modo AHDL, através dos seguintes procedimentos:

- 1) Execute o software MAX + PLUS II e inicialize o projeto com o nome **proj6**, dentro do subdiretório max2work.

Para tal, selecione a opção Project Name dentro do menu File.

Habilite o subdiretório max2work no campo Directories.

Entre com o nome proj6 no campo Project Name.

Selecione OK.

Verifique que na parte superior da tela está o caminho de desenvolvimento do projeto, por exemplo, c:\max2work\proj6.

- 2) Abra a área de desenvolvimento do projeto usando New do menu File.

Habilite a opção Text Editor File que terá extensão .Tdf.

Selecione OK.

- 3) Entre com o texto em AHDL correspondente para o projeto, conforme apresentado na figura 6.2.

Title “Contador binário de quatro bits”

```

SUBDESIGN proj6
(
    clk, load, ena, clr, d[3..0]      :INPUT;
    q[3..0]                          :OUTPUT;
)
VARIABLE
    count[3..0]      :DFF;

BEGIN
    count[ ].clk = clk;      % define que todas as entradas clk dos      %
                             % flip flop D estão ligadas na entrada clk      %
    count[ ].clrn = !clr;    % idem para os pinos clrn – ativo por nível negativo %
    IF load THEN
        count [ ].d = d[ ];
    ELSIF ena THEN
        count[ ].d = count [ ].q + 1;
    ELSE
        count[ ].d = count [ ].q;
    END IF;

    q[ ] = count [ ];

END;

```

Fig. 6.2 Projeto de um contador binário crescente no modo AHDL

3) Comentário sobre o projeto

As declarações Title, Subdesign, Begin e End fazem parte da estrutura de um projeto no modo ADHL e já foram comentadas em projetos anteriores.

A declaração **Variable** foi usada para definir quatro registradores implementados com flip flops do tipo D.

Escolheu-se a lógica condicional da estrutura **IF** para a implementação da operação do contador.

A estrutura **IF** avalia uma, ou mais, expressão Booleana para definir o comportamento de diversas variáveis condicionadas a tal, ou tais, expressão. Esta declaração poderá estar na forma **IF THEN** ou nas variações das formas compostas por **IF THEN ELIF... ELSE**, como usado no projeto em questão.

Nesta estrutura definiu-se que:

- Se load = 1 então o contador carrega as entradas paralelas.
- Senão se ena = 1 então o contador conta crescentemente.
- Caso contrário o contador permanece constante.

O comando q[] = count[] conecta as saídas dos flip flops nas saídas dos circuitos.

5) Salvar o projeto usando a opção **Save As** no **menu File**.

Verifique que o projeto esteja com o mesmo nome no campo **File Name**, que o diretório esteja no caminho correto, por exemplo c:\ **MAX2WORK** e que a extensão seja **.tdf**.

Selecione OK.

6) Siga os passos seguintes descritos no Projeto 1 para compilar, programar e testar o dispositivo.

Se não houver erro, serão criados os arquivos:

proj6.cnf – que contém informações da lógica e conexões do projeto.

proj6.rpt – que contém informações gerais de implementação.

proj6.snf – que contém base de dados para simulações funcionais.

proj6.pof – que contém as informações para programação do dispositivo.

7) Para testar o projeto efetue as conexões necessárias de entradas e saídas, conforme mostrado e complete a tabela seguinte. Use o arquivo proj6.rpt para verificar as pinagens correspondentes.

ENTRADAS / CHAVES								SAÍDAS / LED'S			
ena	clk	clr	load	d3	d2	d1	d0	q3	q2	q1	q0
A	C	D	E	F	G	H	I	L3	L2	L1	L0
x	x	0	1	0	1	0	1	0	1	0	1
x	x	1	x	x	x	x	x	0	0	0	0
0	x	0	0	x	x	x	x	Fica constante			
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				
1	↑	0	0	x	x	x	x				

x - significa não interessa o nível lógico

8) Verifique os resultados.

This document was created with Win2PDF available at <http://www.daneprairie.com>.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.