Departamento de Engenharia Elétrica e de Computação
SEL 455 – Laboratório de Sistemas Digitais

PRÁTICA Nº02

"Dispositivos de Lógica Programável de Complexo (CPLD- "Complex Programable Logic Devices")- Acionamento de Matriz de LEDs"

Utilizando o software QUARTUSII v.12.OSP2, escolha o dispositivo HCPLD Cyclone IV-E EP4CE30F23C7 e faça um projeto para acionamento da matriz de LEDs (item 3 da Figura 2) de 8 linhas x 5 colunas do módulo de desenvolvimento Mercúrio IV, que realize o seguinte procedimento: acenda todos os 5 LEDs da mesma linha simultaneamente, e sequencialmente a cada 5hz, de maneira a fazer a varredura nas 8 linhas de cima para baixo e, de baixo para cima.
As 5 colunas são nomeadas como LEDM_C[0] até LEDM_C[4], as 8 linhas como LEDM_R[0] a LEDM_R[7]. Tanto linhas como colunas são selecionadas com o nível '0'. O clock interno de 50MHz é nomeado como CLOCK_50MHz.
Para criar um arquivo de projeto esquemático no software QuartusII siga os passos do arquivo "Manual QUARTUS" que se encontra no Moodle disciplinas Stoa USP

1. Faça um projeto DIVISOR utilizando o componente lpm_counter que transforme a frequência de 50Mhz em 5hz. Compile e simule. Crie um símbolo para esse projeto

2 Crie um outro projeto(matriz_led_ud) para implementar o restante do circuito:

2.1 Considerando que exista um clock de 5Hz, utilize um contador lpm_counter, para implementar um contador crescente/decrescente de 0 até 7. Troque o nome lpm_counter para qualquer outro nome. Criar entrada UP/DOWN e saída Cout **(não usar o 7493!!).**

2.2 Ligar as saídas binárias do contador do item 2.1 a um decodificador 3X8, 74138.

2.3 Utilizando a saída Cout do contador do item 2.1 implemente o controle para contagem crescente e decrescente, como mostra a Figura 1, utilize o componente TFF (flip flop tipo T). Para que funcione no modo Toggle coloque na entrada T o componente Vcc. A saída do componente TFF deve ser ligada ä entrada UP/Down do contador do\item 2.1.

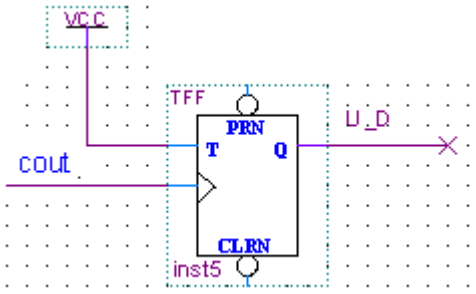


Figura 1 controle up/down

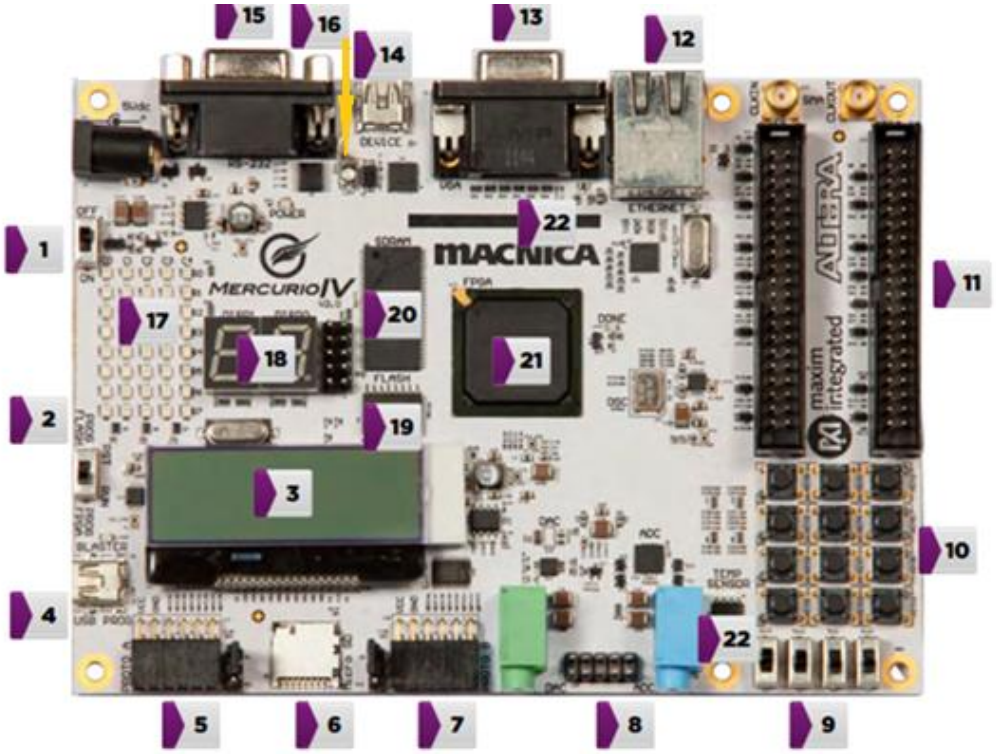
2.4 Ligar as saídas do decodificador às linhas da matriz de LEDs.

2.5 Para ativar as colunas da matriz, criar pinos de saída (output) para cada coluna da matriz LEDM_C[0] até LEDM_C[4].

2.6 Compile o projeto, gere o RTL e simule no ModelSim;
2.7 Como relatório entregue um arquivo PDF com: o printscreen do circuito esquemático documentado(explicação do circuito) incluindo nome e nº USP, Figura do RTL e as formas de ondas da simulação. Envie pelo MOODLE (não se esqueça de clicar em ENVIAR)

3. Placa mercúrio Macnica:

.



1 - Liga / Desliga;
2 - Modo de configuração;
3 - Display LCD;
4 - USB-Blaster™;
5 e 7 - Interfaces PMOD®;
6 - Cartão micro SD;
8 - ADs e DAs;
9 - Switches;
10- Teclado numérico;
11 - GPIO;

12 - Ethernet;
13 - VGA;
14 - USB;
15 - Serial;
16 - LED RGB de alto brilho;
17 - Matriz de LEDs;
18 - Displays de 7 segmentos;
19 e 20 - Memórias Flash e RAM;
21 - FPGA Cyclone® IV;
22 - Sensor de temperatura

Figura 2 Imagem do módulo de desenvolvimento com as indicações dos principais componentes.

4. INFORMAÇÕES SOBRE O PROJETO LPM_COUNTER:

O componente lpm_counter (figura 3) da biblioteca megafunction do Quartus II é um contador que possibilita que o projetista escolha o modo de seu funcionamento, up ou down assim como o tamanho da contagem ate

2^{32} bits. Apresenta as entradas e saídas descritas abaixo as quais podem ser selecionadas ou nõ com exceção da entrada clock e saídas q:

ENTRADAS:

Clock : entrada do pluso que será contado

clk_en : qdo selecionada habilita a entrada do clock no contador qdo em`1`

cnt_en: qdo selecionada habilita os valores das saídas do contador em q qdo em `1`

updown: qdo selecionada habilita contagem crescente qdo em `1` e decrescente qdo em `0`

sclear: qdo selecionada entrada reset síncrona zera as saídas qdo em `1` na subida do clock

aclear: qdo selecionada entrada reset assíncrona zera as saídas qdo em `1` independente do clock

sset: qdo selecionada entrada set síncrona e qdo em `1` força as saídas para `1` na subida do clock

aset: qdo selecionada entrada set síncrona e qdo em `1` força as saídas para `1` independente do clock;

sload:: qdo essa entrada é selecionada é criada a entrada data que possibilita que a contagem inicie a partir de um número carregado no data qdo sload em `1` e ocorrer uma subida do clock

aload: qdo essa entrada é selecionada é criada a entrada data que possibilita que a contagem inicie a partir de um número carregado no data qdo aload em `1` independente do clock

data: entrada criada qdo a entrada sload ou aload forem usadas e qdo sload ou aload forem iguais a `1`o valor de data é carregado nas sídas q

SAÍDAS:

Q: saídas dos FFs do contador

Cout: saída que é levada a `1`qdo o contador chegar a contagem final chegar ao número máximo qdo crescente e zero qdo decrescente.

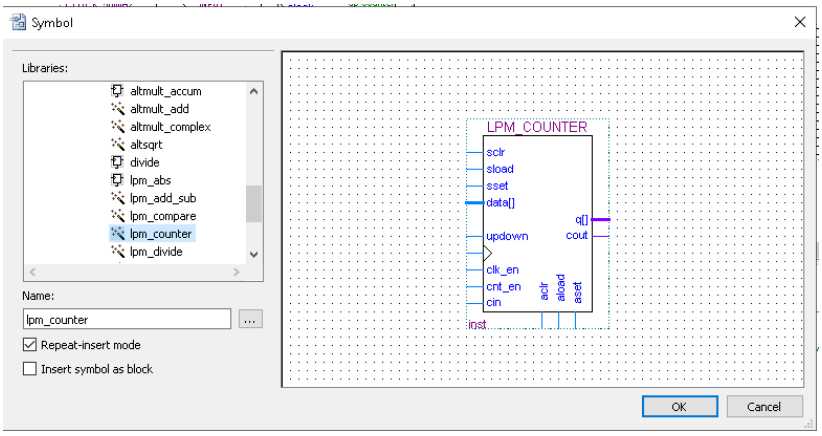


Figura 3 componente lpm_counter

INPUT PORTS

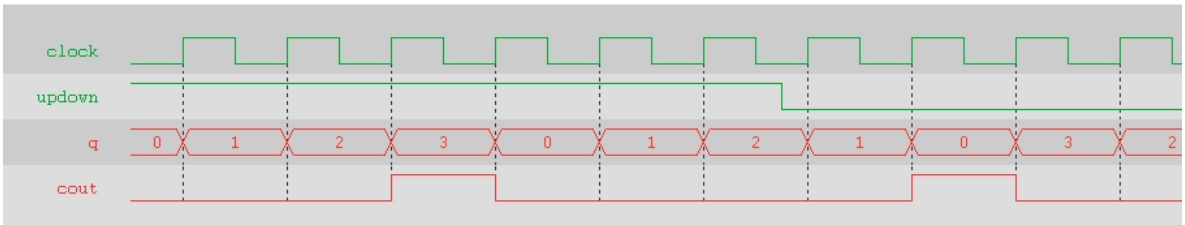
Port Name	Required	Description	Comments
clock	Yes	Positive-edge-triggered clock.	
updown	No	Controls the direction of the count. High (1) = count up. Low (0) = count down.	Default = up (1). If the LPM_DIRECTION parameter is used, the updown port cannot be connected. If LPM_DIRECTION is not used, the updown port is optional.

OUTPUT PORTS

Port Name	Required	Description	Comments
q[]	No	Data output from the counter.	Output port LPM_WIDTH wide. Either q[] or at least one of the eq[15..0] ports must be connected.
cout	No	Carry-out of the MSB.	

Truth Table/Functionality:

Inputs										Outputs	Function
aclr	aset	aload	clk_en	clock	sclr	sset	sload	cnt_en	updown	q[LPM_WIDTH-1..0]	
1	x	x	x	x	x	x	x	x	x	000...	
0	1	x	x	x	x	x	x	x	x	111...	
0	1	x	x	x	x	x	x	x	x	LPM_AVALUE	Asynchronous set to value specified for LPM_AVALUE
0	0	1	x	x	x	x	x	x	x	data[]	Asynchronous load from data[] input
0	0	0	0	x	x	x	x	x	x	q[]	Hold current count value
0	0	0	1	⌜	1	x	x	x	x	000...	Synchronous clear
0	0	0	1	⌜	0	1	x	x	x	111...	Synchronous set
0	0	0	1	⌜	0	1	x	x	x	LPM_SVALUE	Synchronous set to value specified for LPM_SVALUE
0	0	0	1	⌜	0	0	0	0	0	q[]	Hold current count value
0	0	0	1	⌜	0	0	1	x	x	data[]	Synchronous load from data[] input
0	0	0	1	⌜	0	0	0	1	1	q[]+1	Count up
0	0	0	1	⌜	0	0	0	1	0	q[]-1	Count down



Informações sobre o dispositivo FPGA EP4CE30F23C7**:**

Categoria: Circuito Integrado(CI)
Família: *Embedded - FPGAs (Field Programmable Gate Array*)
Série: Cyclone IV E
Número de Blocos Lógicos Configuráveis(CLBs ou LABs**):** 1803

Número de bits da RAM: 608256
Número de portas de Entrada/Saída(I/O): 328

Significado dos Campos do nome do dispositivo:

- EP4CE: Cyclone IV –FPGA de baixo custo
- 30 : quantidade de elementos lógicos: 28848 (aproximadamente 30 mil)
- F23: Encapsulamento: Fineline BGA de 23 x 23 mm e 484 pinos
- C: temperatura de operação, 0°C a 85°C
- 7: tempo de atraso da porta: 7ns

Tensão de Alimentação: 1,15 V ~ 1,25 V
Tipo de Montagem: Montagem de superfície

5. Bibliografia:

- Site da ALTERA
- Fregni, E. & Saraiva, A.M., " Engenharia do Projeto Lógico Digital", Ed. Edgard Blücher Ltda.
- Tocci, J. R. , "Sistemas Digitais- Princípios e Aplicações