

Departamento de Engenharia Elétrica e de Computação
SEL 384 – Laboratório de Sistemas Digitais I

PRÁTICA Nº10

“Dispositivos de Lógica Programável de Complexo (FPGA- “Field Programmable Gate Array”)- Aplicação de Display de Cristal Líquido

”

1. Objetivos:

- Projeto e *síntese* de circuitos em dispositivo FPGA utilizando o esquemático do Quartus II;
- Aplicação de *display* LCD e memória ROM para escrita de caracteres

2. Material utilizado:

- Configuração de Dispositivo Programável de Alta Complexidade HCPLD do tipo FPGA Cyclone IV-E da Altera
- Módulo de desenvolvimento Mercúrio IV – Macnica DWH
- Multímetro

3. Procedimento Experimental:

3.1 Utilizando o software QUARTUSII v.12.OSP2, escolha o dispositivo HCPLD Cyclone IV-E EP4CE30F23C7 e projete um circuito que apresente no display de cristal líquido (LCD) do módulo de desenvolvimento Mercúrio IV, um conjunto de caracteres de duas maneiras:

- Display configurado para apresentação de caracteres em duas linhas de 16 caracteres cada linha;
- Display configurado para apresentação de caracteres em um única linha com 16 caracteres, altura dupla.

• **Sobre o LCD:**

O display LCD, cuja referência na placa é J18, contém 2 linhas de 16 caracteres (2x16), fabricante NewHaven e código é NHD-CO216CU-FSW-GBW-3V3 . Esse display possui um mapa de caracteres, Tabela III, que pode ser utilizado para escrever no display. Além do mapa de caracteres, existe uma memória DDRAM utilizada para o mapeamento do display diretamente, ou seja, existem 32 endereços da DDRAM (Tabela I), cujos valores são exibidos no display, segundo o mapa de caracteres. **Observe que a linha 2 inicia no endereço 40H da DDRAM. (ref. manual_mercurioiv_v2)**

Tabela I

| | | Posição no display | | | | | | | | | | | | | | | |
|-------------------|---------|--------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| Endereço da DDRAM | Linha 1 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |
| | Linha 2 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 4A | 4B | 4C | 4D | 4E | 4F |

Por exemplo, no mapa de caracteres o valor, em binário, 01000001 (41H) corresponde ao caractere **A** maiúsculo, então, se o dado 41H for escrito no endereço 0 da memória DDRAM, o caractere **A** será desenhado na primeira posição, superior esquerda do display.

A configuração do display e a escrita dos caracteres é feita pelo barramento LCD_DATA[7..0], como mostra a Tabela II e pelos sinais LCD_EN, LCD_RS e LCD_RW.

LCD_RW: em nível alto indica que será feita uma leitura e em nível baixo uma escrita;

LCD_RS: em nível alto indica que o acesso será feito na memória do display e em nível baixo indica que o acesso será feito num registrador de configuração do display;

LCD_EN: deve receber um pulso positivo quando se deseja realizar uma operação de escrita ou leitura (sensível à borda de descida);

LCD_BACKLIGHT: é utilizado para ligar ou desligar o *backlight* do display (nível '1' aceso).

Tabela II

| NOME DO SINAL | PINO DO FPGA | DESCRIÇÃO |
|---------------|--------------|---|
| LCD_BACKLIGHT | V10 | Controlador do backlight |
| LCD_EN | V9 | Operation Enable, inicia uma operação com o LCD |
| LCD_RS | U9 | Register Select, seleciona se o acesso é em registrador ('0') ou em memória ('1') |
| LCD_RW | U8 | Read Write, indica se é uma leitura ('1') ou escrita ('0') |
| LCD_D[0] | V8 | Bit 0 do barramento de comando/dado - LSB |
| LCD_D[1] | V7 | Bit 1 do barramento de comando/dado |
| LCD_D[2] | V6 | Bit 2 do barramento de comando/dado |
| LCD_D[3] | V5 | Bit 3 do barramento de comando/dado |
| LCD_D[4] | V4 | Bit 4 do barramento de comando/dado |
| LCD_D[5] | Y4 | Bit 5 do barramento de comando/dado |
| LCD_D[6] | V3 | Bit 6 do barramento de comando/dado |
| LCD_D[7] | Y3 | Bit 7 do barramento de comando/dado - MSB |

Tabela III Mapa de caracteres do controlador ST7032 utilizado no LCD do kit mercúrio

ST7032-0D (ITO option OPR1=1, OPR2=1)

| b7-b4 b3-b0 | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
|----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 0000 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | - | . | , | ; | : |
| 0001 | ! | + | ! | 1 | 9 | 0 | 3 | 4 | 0 | 3 | = | P | 7 | 4 | 3 | 7 |
| 0010 | @ | 5 | " | 2 | B | R | b | n | é | E | r | 7 | W | X | 3 | * |
| 0011 | 7 | 7 | * | 3 | 0 | S | o | 3 | 3 | 3 | U | 7 | 7 | 0 | 0 | 7 |
| 0100 | 4 | 7 | * | 4 | D | T | d | t | 3 | 3 | 7 | E | t | 7 | 0 | 7 |
| 0101 | ↑ | Δ | Σ | 5 | E | U | e | u | 3 | 3 | . | 7 | 7 | 7 | 0 | 7 |
| 0110 | ↓ | Θ | Σ | 6 | F | V | f | v | 3 | 0 | 7 | 7 | 7 | 7 | * | 7 |
| 0111 | → | Δ | 7 | 7 | G | W | g | w | 3 | 0 | 7 | 7 | 7 | 7 | 7 | 7 |
| 1000 | ← | 3 | 0 | 0 | H | X | h | x | 3 | 0 | 7 | 7 | 7 | 7 | 7 | 7 |
| 1001 | 7 | 7 | > | 9 | I | Y | i | y | 3 | 0 | 7 | 7 | 7 | 7 | 7 | 7 |
| 1010 | 7 | 7 | * | * | J | Z | j | z | 3 | 0 | 7 | 7 | 7 | 7 | 7 | 7 |
| 1011 | L | 7 | + | 3 | K | C | k | c | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 1100 | 7 | 7 | . | < | L | 7 | l | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 1101 | . | ψ | - | = | M | I | m | > | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 1110 | 0 | 0 | . | > | N | ^ | n | * | 3 | 0 | 7 | 7 | 7 | 7 | 7 | 7 |
| 1111 | 0 | 0 | / | ? | 0 | L | o | ← | 3 | 0 | 7 | 7 | 7 | 7 | 7 | 7 |

- **Funcionamento do Display:**

Para configuração do *display* é necessário executar algumas instruções de controle como mostra a tabela IV.

Tabela IV - Tabela de Comandos (*data sheet* do NHD-CO216CU-FSW-GBW-3V3 controlador do *display*)

| Instruction | Instruction Code | | | | | | | | | | Description | Instruction Execution Time | | |
|----------------------------|------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|--|----------------------------|-------------|-------------|
| | RS | R/W | DB7 | DB6 | DB5 | DB4 | DB3 | DB2 | DB1 | DB0 | | OSC= 380KHz | OSC= 540kHz | OSC= 700KHz |
| Clear Display | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Write "20H" to DDRAM, and set DDRAM address to "00H" from AC | 1.08 ms | 0.76 ms | 0.59 ms |
| Return Home | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | x | Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed. | 1.08 ms | 0.76 ms | 0.59 ms |
| Entry Mode Set | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | UD | S | Sets cursor move direction and specifies display shift. These operations are performed during data write and read. | 26.3 us | 18.5 us | 14.3 us |
| Display ON/OFF | 0 | 0 | 0 | 0 | 0 | 0 | 1 | D | C | B | D=1:entire display on C=1:cursor on B=1:cursor position on | 26.3 us | 18.5 us | 14.3 us |
| Function Set | 0 | 0 | 0 | 0 | 1 | DL | N | DH | *0 | IS | DL: interface data is 8/4 bits N: number of line is 2/1 DH: double height font IS: instruction table select | 26.3 us | 18.5 us | 14.3 us |
| Set DDRAM address | 0 | 0 | 1 | AC8 | AC5 | AC4 | AC3 | AC2 | AC1 | AC0 | Set DDRAM address in address counter | 26.3 us | 18.5 us | 14.3 us |
| Read Busy flag and address | 0 | 1 | BF | AC8 | AC5 | AC4 | AC3 | AC2 | AC1 | AC0 | Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read. | 0 | 0 | 0 |
| Write data to RAM | 1 | 0 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Write data into internal RAM (DDRAM/CGRAM/CONRAM) | 26.3 us | 18.5 us | 14.3 us |
| Read data from RAM | 1 | 1 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Read data from internal RAM (DDRAM/CGRAM/CONRAM) | 26.3 us | 18.5 us | 14.3 us |

Note *: this bit is for test command, and must always set to "0"

1º Passo: Executar a função **display ON/OFF** para ligar o display e o cursor setando os bits : D, C,B (ver tabela V);

2º Passo: Executar o **CLEAR display** enviar "00000001" (ver tabela V);

3º Passo: Executar a função **SET DDRAM address** (ver tabela V):

'10000000' para a 1ª. linha endereço 00H
'11000000' para a 2ª. linha endereço 40H

4º Passo: Executar a **Function Set** (ver tabela V);







5º Passo: Enviar os códigos ASCII da mensagem a ser escrita no display com a instrução **Write DATA** (ver tabela V);

Para enviar essas funções para o *display*, pode-se utilizar uma memória que armazena os sinais de controle (EN, RS e R/W) e os códigos das funções (para configuração) ou o código ASCII (para a mensagem de texto).

É necessário a geração de um pulso de Enable (EN) para o *display* receber a configuração ou o dado. Isso é feito utilizando duas posições da memória, uma com EN='1' e outra com EN='0', mantendo os outros bits iguais.

Para a execução das instruções é necessário que os níveis dos sinais RS(seleciona instrução) e R/W(seleciona gravação) sejam '0', conforme exemplo da tabela V para a função *display* ON/OFF:

Tabela V

| Instrução | EN | RS | R/W | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Descrição |
|-------------------|---|----|-----|----|----|----|----|----|----|----|----|--|
| Display ON/OFF |  | 0 | 0 | 0 | 0 | 0 | 0 | 1 | D | C | B | EN = pulso de habilitação RS='1' seleciona dado ='0'seleciona instrução R/W ='1' leitura ='0' gravação D = '1' display ligado ='0' display desligado C = '1' cursor ligado '0' cursor desligado B = '1' cursor piscando ='0' cursor fixo |
| Clear Display |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Limpa o display |
| Set DDRAM address |  | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Define endereço 00H da DDRAM |
| |  | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Define endereço 40H da DDRAM |
| Function Set |  | 0 | 0 | 0 | 0 | 1 | DL | N | DH | 0 | 0 | DL= '1' dado de 8 bits '0' dado de 4 bits N = '1' LCD com duas linhas '0' LCD com uma linha DH ='1' se altura da letra é Dupla ='0'se altura for simples |
| Write data RAM |  | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | Envia a letra A maiúscula (41H) na DDRAM |

Para todas as funções da tabela V devem ser enviados duas palavras uma com EN = '1' e outra com EN = '0' sendo os demais bits com valores iguais. Por exemplo, para o comando **Write data**

RAM: palavra 1 : 110 0100 0001 (envia a letra A que é 41H)
palavra 2 : 010 0100 0001 (envia a letra A que é 41H)

- **Implementação do Circuito para transmissão de caracteres para o LCD:**

- ✓ Implementar um divisor de frequência que gere a partir do clock de 50MHz da placa (CLOCK_50MHz), um clock de 1ms para fazer a varredura em uma memória ROM (lpm_ROM), onde deverão ser armazenadas as palavras de instrução e dados, as quais serão transmitidas ao LCD. Observação: o período de 1ms foi definido de acordo com a instrução mais lenta do LCD (data sheet da Sitronix ST7032, controlador do display).
- ✓ Determine a quantidade de palavras da ROM considerando que serão armazenados instruções e os dados que serão visualizados, e cada instrução e cada dado irão ocupar duas posições de memória. Observe que o tamanho da palavra é de 11 bits.
- ✓ Para gerar os endereços da lpm_ROM, implemente um contador de tal maneira a varrer o conteúdo da memória que será enviado para o display uma única vez. O contador (lpm_counter) deve parar de contar quando gerar o último endereço e em seguida deve ser zerado. Para isso, utilize as entradas **clk_en**, e **sclr** e a saída **cout** (funcionamento na tabela VI). O módulo do contador deve ser igual ao número de palavras da ROM acrescido de uma unidade.
- ✓ Utilize a chave Push- Buttom KEY[2] para reiniciar a varredura da memória ROM, ou seja, para reiniciar a contagem do contador. Para tal, implemente o circuito da Figura 1, o qual gera o sinal de *enable* para o contador de endereço.

Obs: A chave KEY[2] quando **não** pressionada gera nível '0'.

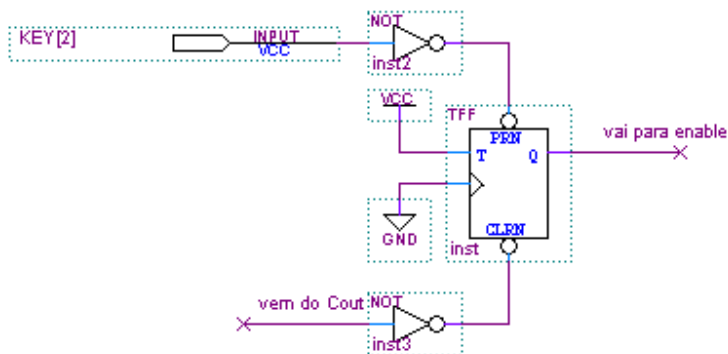


Figura1 Circuito de controle do enable do contador de endereço.

.Crie um arquivo mensagem.mif para armazenar na memória ROM, com os controles e os códigos ASCII da mensagem a ser escrita no display LCD(ver prática nº9).

Para implementação da memória ROM deve-se utilizar o seguinte procedimento:

- i. Instanciação do componente `lpm_rom` no diagrama esquemático;
- ii. Selecione com o botão da direita do *mouse* o componente `lpm_rom` e entre em **PROPERTIES/PORTS** e selecione o **STATUS** para **UNUSED** para os sinais “**memenab**” e “**outclock**”. A seguir selecione **PARAMETER** e programe os seguintes parâmetros:
LPM_ADDRESS_CONTROL : REGISTERED
LPM_WIDTH: 11 (tamanho do dado);
LPM_WIDTHAD: XX (número de linhas de endereços depende da quantidade de palavras a serem armazenadas Ex: 5 resulta em 32 palavras com largura de 11 bits);
LPM_FILE : mensagem.mif.
- iii. Para criar o arquivo que vai ser inserido na memória (`mensagem.mif`), seleciona-se no menu FILE/NEW/MEMORY FILE/ Memory Initialization File. Escolha :**Number of word** = 76 (número de palavras de 11 bits para escrever em duas linhas do *display*) e **Word Size** = 11 (tamanho do dado). Finalize com FILE/SAVE AS/ `mensagem.mif` , salvando o arquivo na mesma pasta do projeto.

a. **Display configurado para apresentação de caracteres em duas linhas de 16 caracteres cada linha:**

Gravar a função **clear** seguida das funções **display ON/OFF** ('000000111X' com X =1 cursor fixo e X=0 cursor piscando) e **function set** ('000001DLN DH 00' com DL ='1', N='1' e DH ='0'). Preencher a 1ª. linha da memória ROM com até 16 caracteres. Em seguida defina o endereço da segunda linha (endereço 40H) e preencha a 2ª. linha da memória ROM com até 16 caracteres.

b. **Display configurado para apresentação de caracteres em um única linha com 16 caracteres, altura dupla:**

Gravar a função **clear**, seguida da função **display ON/OFF** ('000000111X' com X =1 cursor fixo e X=0 cursor piscando). Em seguida use o comando **function set** '1000110100' e '0000110100' para definir linha única no display, e preencha a memória com até 16 caracteres.

3.2 Configure o dispositivo no módulo de desenvolvimento Mercúrio IV e teste os circuitos projetados nos itens anteriores. Para programar, a **chave 1 do Kit** deve estar na posição **PROG FPGA** e a **chave 2** na posição **ON**. (como mostrado no Guia Rápido Mercúrio IV).

3.3 Mostre o funcionamento do circuito sintetizado no FPGA após a programação.

3.4 Como relatório entregue o circuito esquemático documentado e os arquivos `mensagem.mif` impressos.

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.

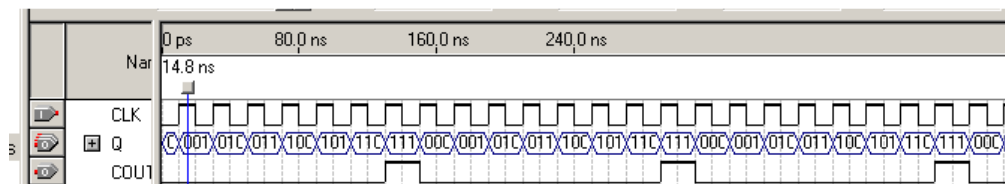
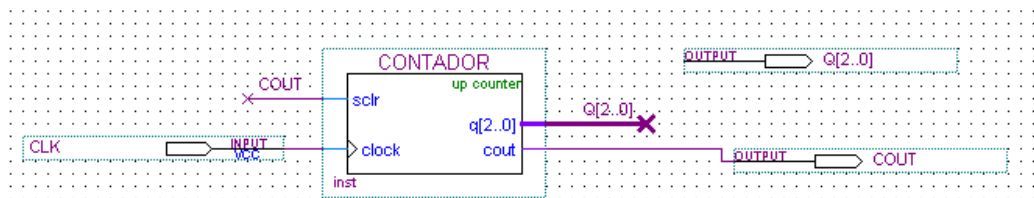
Tabela VI sinais de contador lpm_counter

Truth Table/Functionality:

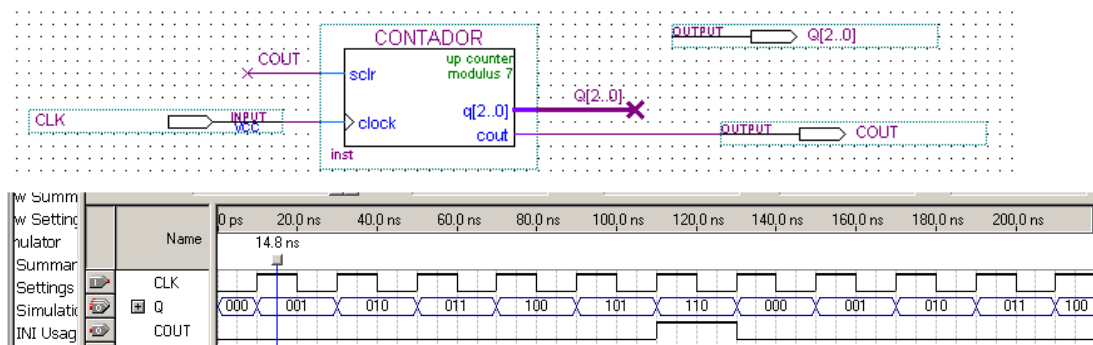
| Inputs | | | | | | | | | | Outputs | Function |
|--------|------|-------|--------|-------|------|------|-------|--------|--------|-------------------|--|
| aclr | aset | aload | clk_en | clock | sclr | sset | sload | cnt_en | updown | q[LPM_WIDTH-1..0] | |
| 1 | x | x | x | x | x | x | x | x | x | 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 |

Exemplo do funcionamento do projeto lpm_counter utilizando entradas **clock** e **sclr** e saídas **Qi** e **Cout**

Ex1: lpm_counter como contador binário de 3 bits



Ex2: lpm_counter como contador de módulo 7 e de 3 bits



4. 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