IMPORTANTE! Essa avaliação é on-line e por isso é permitida a consulta aos materiais da Wiki, livro texto e outras anotações do aluno. No entanto, você não está autorizado a consultar, trocar códigos e informações com outras pessoas sobre a avaliação durante o período em que está realizando a mesma. O aprendizado e avaliação é sua, e por isso peço que seja ético e correto nesta avaliação.
Ao final da prova você deverá enviar para o Moodle os arquivos solicitados em cada questão. O arquivo QAR deverá estar conter os arquivos de projeto e simulação, e deve ser nomeado de acordo com a Questão (Q1.qar, Q2.qar etc.). Todas as entradas e saída devem ser do tipo ***std_logic(_vector).***
Lembre-se que ***NÃO é permitido usar a biblioteca std_logic_arith.*** *Na hora de elaborar o projeto inicial, utilize sempre o FPGA mais simples da família Cyclone I, que permita implementar o projeto, exceto se for solicitado outro dispositivo.* ***Nesta avaliação utilize código sequencial sempre que for necessário mas é permitido o uso também do código concorrente.***
No texto BCD indica Binary Coded Decimal (0 a 9 em binário). DU significa que a dezena está em D e unidade em U.

1. Projete um circuito que realize a contagem decrescente de N até 0, quando a entrada **DIR** estiver em "1" e a contagem seja crescente de 0 até N quando a entrada **DIR** estiver em "0". Sempre que atingir o valor de N/2, a saída **MEIO** deve ir para "1" e permanecer em "0" para os demais valores. A contagem deve ser feita no momento da **borda de subida** do **CLOCK**. A entrada **INICIAR** = '1' é sincrona com o **CLOCK** e deve levar o valor do contador para 999 na contagem decrescente e 000 na contagem crescente.

   a) Escreva o código VHDL, e compile o código, para um valor N de 50.

   b) Mostre que o código funciona simulando com o MODELSIM. Mostre o funcionamento da entrada **DIR**, o funcionamento do sinal **INICIAR**, e também uma contagem completa de 50 até 0 e de 0 até 50.

   c) *Salve a imagem do código RTL, o qar contendo o projeto e a simulação como Q1RTL.png, Q1.qar, e Q1SIM.png.*

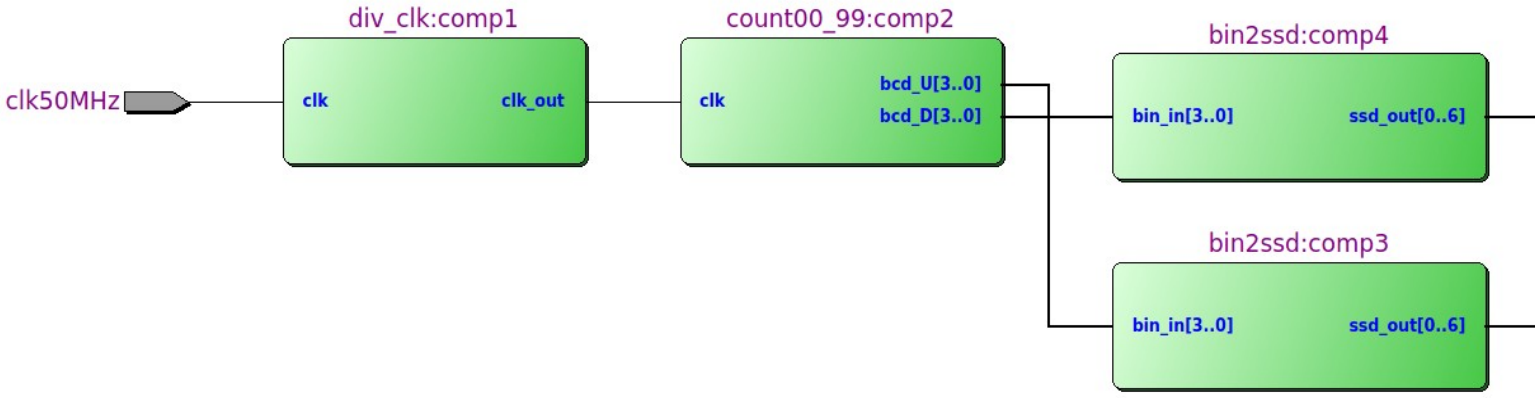
2. Conclua o projeto do Timer 00 a 59 segundos, com saída em display de 7 segmentos.

```
entity timer00_99seg IS
	generic (fclk2 : natural := 50, D : natural := 5;	U : natural := 9);
	port
	(
		clk50MHz :  in  STD_LOGIC;
		clk_1seg: out STD_LOGIC;
		ssd_D :  out  STD_LOGIC_VECTOR(0 TO 6);
		ssd_U :  out  STD_LOGIC_VECTOR(0 TO 6)
	);
end entity;
```

●O valor do fclk2 corresponde a metade do período do clock de entrada em Hz.
●Os valores D e U correspondem ao último valor do timer, depois desse valor o timer dever reiniciar a contagem.

O projeto deve ser dividido em pelo menos os seguintes componentes (ver exemplo do RTL abaixo)

Figura AE6.a - RTL do Timer 00 a 99 segundos



FONTE: Próprio autor.

●Componente 1 - Divisor de Clock, com o valor da divisão configurável pelo parâmetro **fclk2**. O sinal de saída será usado como "enable" ou "clock" a cada 1 segundo para o componente **count00_99**.

```
component div_clk is
	generic (fclk2 : natural := 50);	-- frequecia para simulacao
	port (
		clk : in std_logic;
		clk_out : out std_logic
	);
end component;
```

●Componente 2 - Contador de 00 a 99 com saída em BCD, com o valor final configurável pelos parâmetros **D e U**

```
component count00_99 is
      generic (D : natural := 9;      U : natural := 9);
      port (
            clk : in std_logic;
            clk_out : out std_logic;
            bcd_U : out std_logic_vector(3 downto 0);
            bcd_D : out std_logic_vector(3 downto 0)
      );
end component;
```

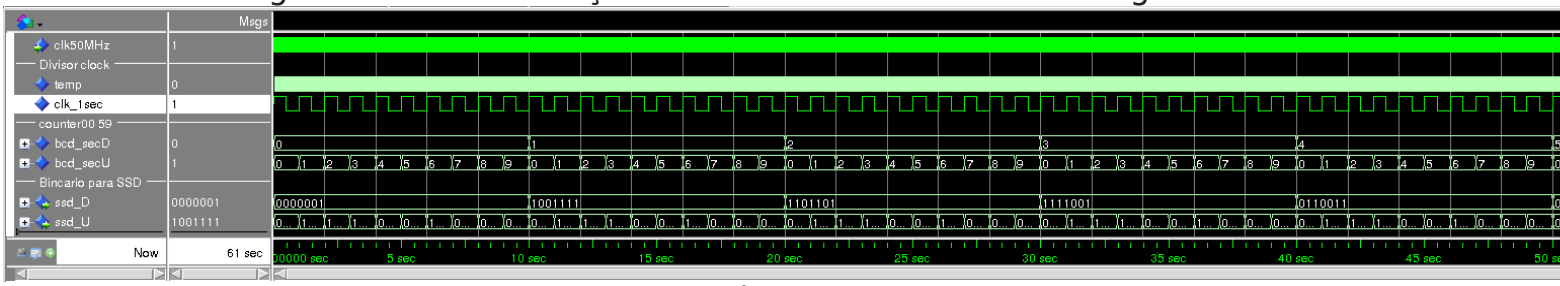
●Componente 3 - Conversor de BIN para SSD, com um parâmetro configurável **ac_ccn** para selecionar Anodo ou Catodo Comum.

```
component bin2ssd is
  generic (ac_ccn : natural := 0);
  port (
    bin_in : in std_logic_vector(3 downto 0);
    ssd_out : out std_logic_vector(0 to 6)
  );
end component;
```

OBS

●O valor ac_ccn é utilizado para configurar o circuito entre ativo alto para display de catodo comum (ac_ccn=0), ou ativo baixo para display de anodo comum (ac_ccn=1).
●Simulação do timer com ModelSim. Na simulação definir o clk50MHz com duração de 10 ms. Faça uma simulação de pelo menos 60 segundos. O projeto deve ser simulado por componente e após isso ser feita a integração dos componentes (Ver exemplo de teste de simulação abaixo)

Figura AE6.b - Simulação funcional do Timer 00 a 99 segundos



FONTE: Próprio autor.

●Após a verificação do funcionamento por simulação funcional, utilizar o lab home office para enviar o hardware para o kit e comunicar ao professor para verificar se funcionou corretamente.
●É recomendável inserir um sinal de RESET em todos os circuitos sequenciais e ao iniciar a simulação do circuito começar com RESET ativo durante 10 ps.

3. (BONUS 2) Implemente o projeto do Timer 00 a 59 no kit FPGA Mercúrio IV, fixando

   *a) Salve a imagem do **pin planner** com a configuração dos pinos no arquivo Q3PIN.png, o arquivo de programação do FPGA Q3.sof e o arquivo Q3.qar contendo o projeto escolhido com as pinagens (o arquivo que contém a pinagem e o Q3.qsf).*
   b) Faça uma filmagem do Kit funcionando realizando a contagem d e 0 a 59 segundos e mais uns 10 segundos.