



Laboratório de Circuitos Lógicos - 6º Experimento

CIRCUITOS COMBINACIONAIS: CODIFICADOR E DECODIFICADOR

OBJETIVO: Elaboração de um codificador e de um decodificador usando-se circuitos combinacionais e aplicando-se as técnicas de minimização de funções lógicas. Verificação da possibilidade de conversão de um número decimal em um número binário de código qualquer e sua posterior decodificação.

1. INTRODUÇÃO TEÓRICA

O sistema de numeração binário apresenta várias vantagens no seu uso em sistemas digitais, principalmente devido à simplicidade dos circuitos utilizados que precisam distinguir somente dois níveis de tensão. Isto possibilita o aumento da confiabilidade dos sistema, a baixo custo.

Concernente à comunicação homem-máquina, entretanto, o sistema binário nem sempre é o mais adequado. O homem é acostumado desde sua infância a raciocinar e trabalhar no sistema decimal (base 10) e a maioria das pessoas não têm muita facilidade de manusear dígitos binários (*bits*) diretamente.

Assim sendo, convém incorporar aos equipamentos de entrada e saída de computadores um conversor de código decimal-binário ou vice-versa. Chamaremos de **codificador** um conversor de códigos em que o código de entrada é o decimal e de **decodificador** aquele em que o código de saída é o decimal. A rigor, o uso dos termos codificador e decodificador é mais geral. Eles são reservados a conversores de códigos em que se tenha na entrada ou saída, respectivamente, um código onde somente um *bit* é ativado num dado instante. A **Figura 1** mostra o esquema geral de um codificador e de um decodificador em que um dos códigos envolvidos é o decimal.

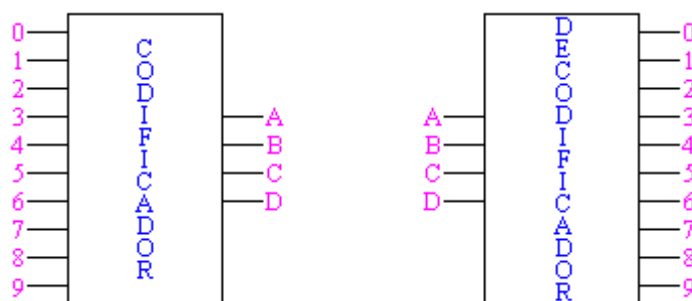


Figura 1 – Codificador e decodificador envolvendo o código decimal

No codificador da **Figura 1** teremos *sempre uma e somente uma entrada ativada* em cada instante e a saída (A, B, C e D) apresentará uma combinação de 0's e 1's correspondente ao dígito decimal ativado. O código usado na saída dependerá da finalidade específica em cada caso.

No caso do decodificador da **Figura 1** tudo se passa analogamente ao caso do codificador, mudando-se as entradas pelas saídas.



Lembre-mos que a codificação de 10 símbolos decimais requer pelo menos 4 *bits*, pois existem $2^4 = 16$ formas diferentes de arranjar 4 *bits* enquanto com 3 *bits* conseguiríamos apenas $2^3 = 8$ códigos distintos. Sendo disponíveis 16 arranjos de 0's e 1's, pode-se atribuir a cada arranjo um valor entre 0 e 9. Portanto, temos ao todo:

$$C(16,10) \times P(10,10) = \frac{16!}{10!6!} \cdot 10! = \frac{16!}{6!} = 29 \cdot 10^9$$

ou seja, cerca de 29 bilhões de códigos diferentes em que cada dígito 0-9 é representado por um arranjo de quatro 0's e 1's.

Na prática apenas alguns desses códigos têm sido usados, pois com a maioria deles as operações aritméticas tornam-se excessivamente dificultosas. Os mais conhecidos são o código BCD, o 2421, o Gray, o excesso-3, etc.

1.1. PLATAFORMA DE DESENVOLVIMENTO INTEL DE2

Nesta disciplina utilizaremos o software Quartus-II, da Intel, para realizar simulações e síntese dos circuitos digitais em FPGA. O Quartus-II é a ferramenta de desenvolvimento desenvolvido pela empresa Intel para seus produtos FPGA. Esta ferramenta permite a síntese de circuitos descritos em diagramas esquemáticos ou nas Linguagens de Descrição de Hardware Verilog, System Verilog ou VHDL, para fins de implementação no dispositivo FPGA ou para simulação.

Um chip FPGA (*Field Programmable Gate Array*) é um circuito eletrônico integrado capaz de sintetizar qualquer sistema digital através apenas das interconexões de seus componentes elementares (Elementos Lógicos). Neste, e nos próximos experimentos, iremos utilizar kits de desenvolvimento FPGA DE2 da Intel.

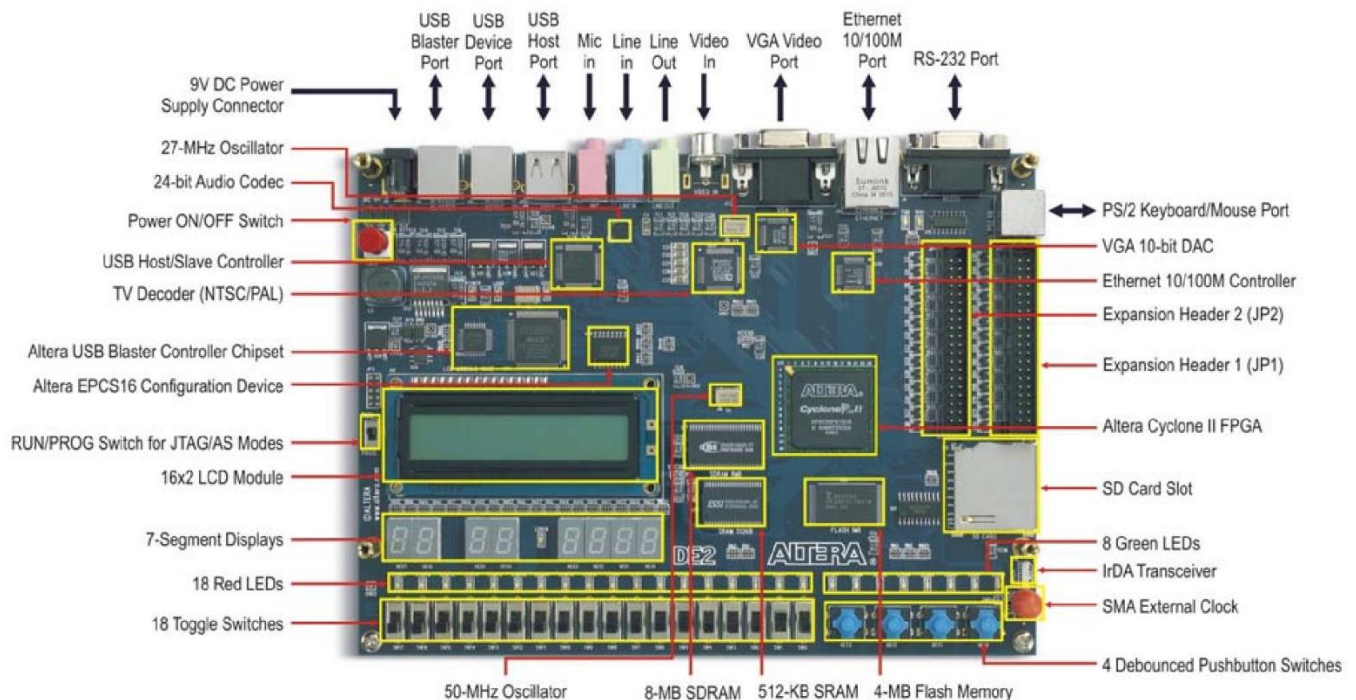


Figura 2 – Kit de desenvolvimento Altera DE2

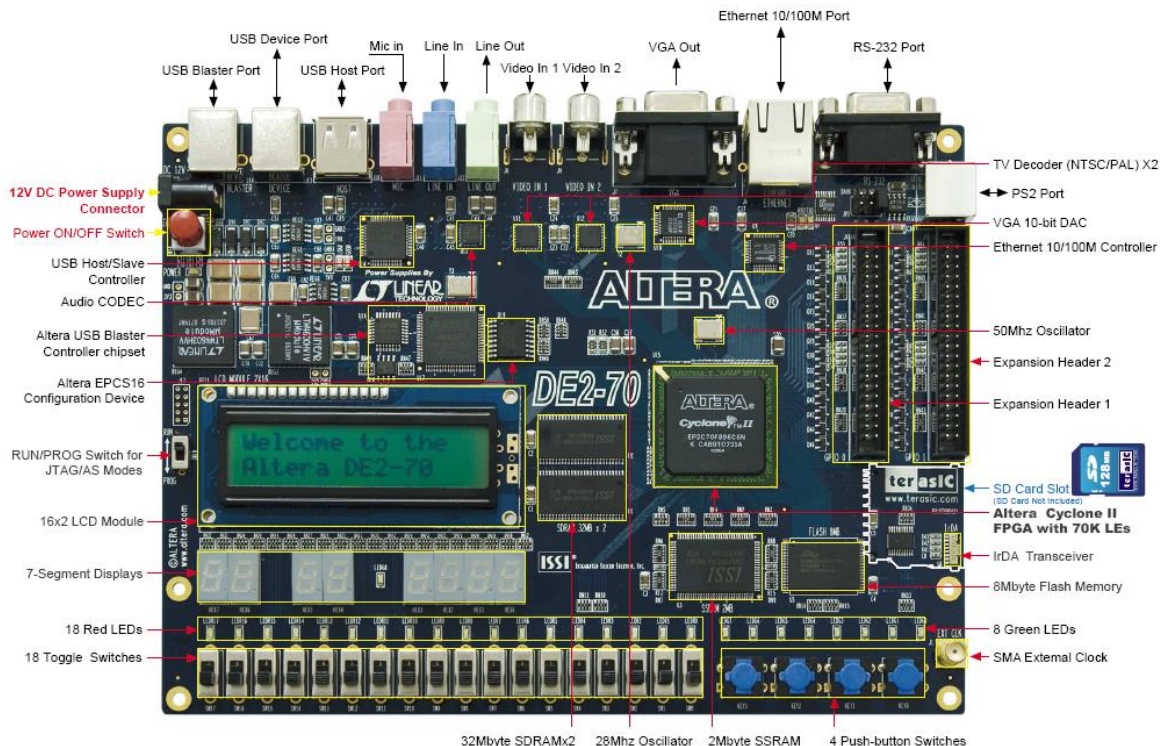


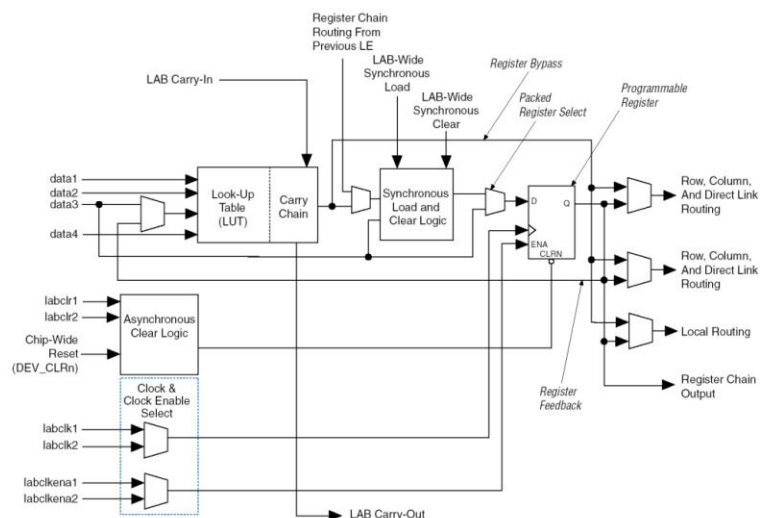
Figura 3 – Kit de desenvolvimento Altera DE2-70

O LINF dispõe de **dois** modelos deste kit: a DE2, mostrada na **Figura 2**, e a DE2-70, mostrada na **Figura 3**. Ambos os kits são equivalentes aos nossos propósitos, diferenciando-se apenas no modelo do chip FPGA.

O kit DE2 é composto de um chip FPGA da família Cyclone II modelo EP2C35F672C6, que possui 484 pinos e contém 33.216 Elementos Lógicos. O kit DE2-70 possui um chip FPGA também da família Cyclone II, porém modelo EP2C70F896C6, que possui 672 pinos e contém 68.416 Elementos Lógicos.

Figure 2-2. Cyclone II LE

Cada Elemento Lógico, mostrado na figura ao lado, é capaz de implementar uma função lógica booleana de 4 entradas e 1 saída, possui um registrador programável (*Flip-Flop*), são encadeáveis (*carry*) e podem ser usados para roteamento (ligação de um ponto a outro no chip FPGA).





1) Ligando o kit FPGA:

- Coloque a fonte de alimentação na tomada e conecte-a a placa.
- Conecte o cabo USB, uma extremidade no computador que possui o Quartus-II instalado, e a outra extremidade na porta USB denominada BLASTER (escrito na placa).
- Ligue a placa pressionando, *cuidadosamente*, o botão vermelho.
- Não mexa **NUNCA** na chave RUN↔PROG que está ao lado do display LCD, que deverá permanecer sempre na posição RUN.

Se os passos forem seguidos de maneira correta, a placa deve ligar, os LEDs devem piscar, os displays mostram uma contagem em hexadecimal e a mensagem “Welcome to the Altera DE2” deve ser mostrada no display LCD.

2) Implementando o seu projeto no chip FPGA da DE2:

Após o projeto a ser implementado ter sido desenhado e corretamente simulado por forma de onda (funcional e temporal) no Quartus-II, conforme indicado no roteiro do Experimento 4, devemos sintetizá-lo no chip FPGA do kit de desenvolvimento.

Para tanto:

- Confira se o chip FPGA definido no seu projeto está correto com Assignments > Device > Family Cyclone II e device de acordo com o kit que vc está usando EP2C35F672C6 ou EP2C70F896C6

- Configure o pino nCEO como pino de IO clicando no botão Device and Pin Options... / Dual-Purpose Pins e selecione nCEO como Use as regular IO.

- Clique em Ok e Ok novamente

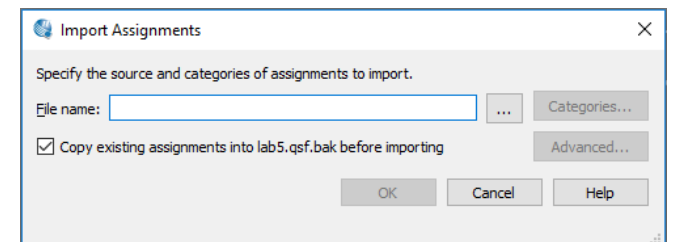
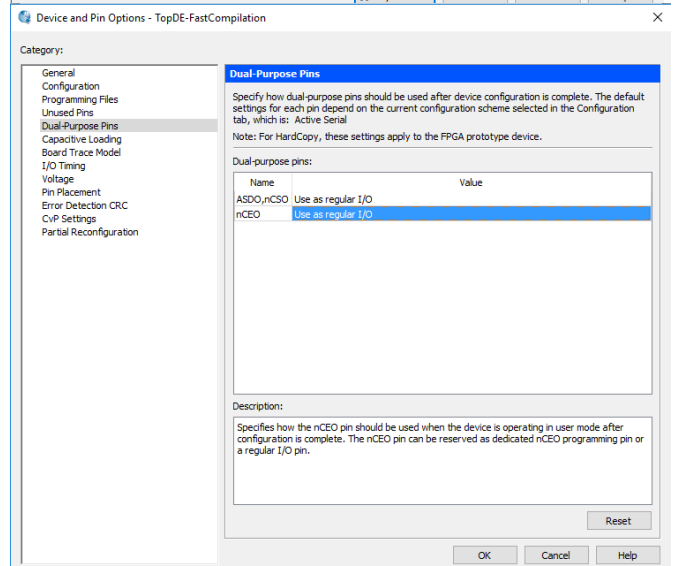
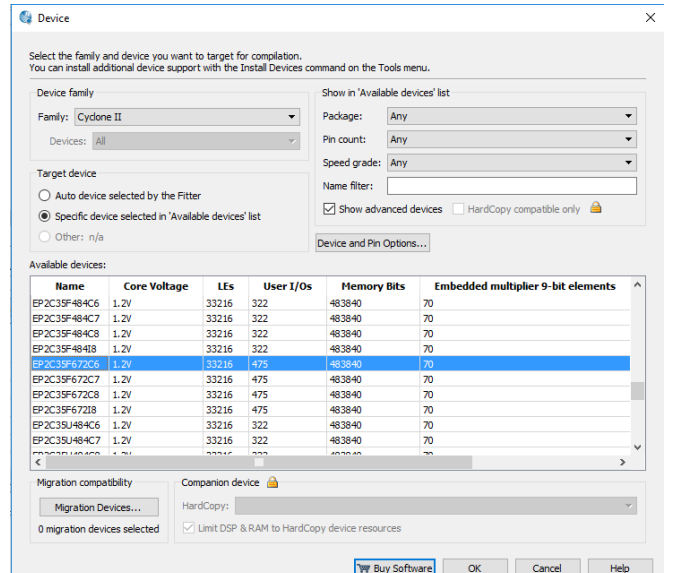
- Faça o download do arquivo de definição dos pinos da FPGA:

- DE2_pin_assignments.csv
 - DE2_70_pin_assignments.csv
- OU

do Moodle.

- Importe o arquivo de definição dos pinos: Assignments > Import Assignments e selecione o arquivo correto .csv clicando no ícone ...

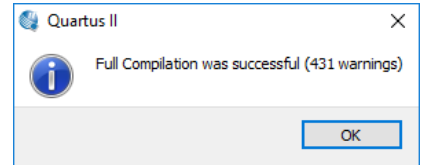
Nestes arquivos .csv estão as definições dos nomes de todos os dispositivos ligados aos pinos da FPGA e que devem ser usados no seu projeto. Você **NÃO** deve alterar estes arquivos!
Ex.: SW[0], LEDR[0], KEY[0], etc.






- Compile novamente seu projeto

- Se tudo estiver Ok deve aparecer uma mensagem indicando que ocorreram mais de 420 *warnings*. Se não aparecer esta quantidade é provável que algum passo tenha sido esquecido.

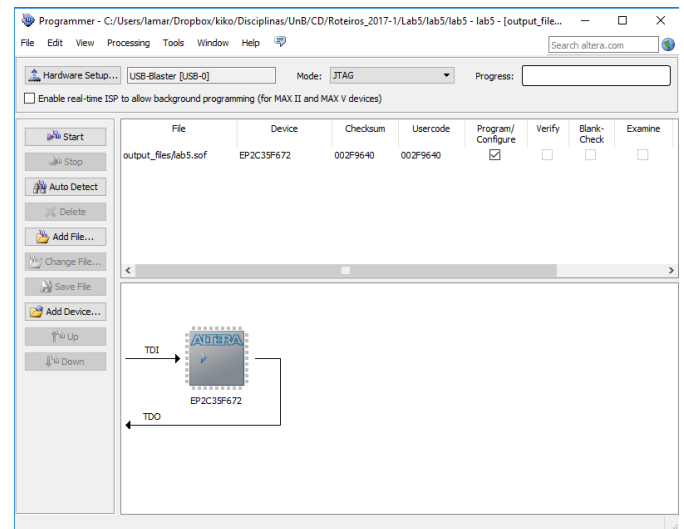


- Abra a janela do programador do chip FPGA através do ícone Programmer 

- Verifique se o cabo USB está corretamente conectado na placa DE2 na entrada Blaster

- Verifique se o driver USB Blaster está corretamente instalado: Hardware Setup > Currently selected hardware > USB Blaster.

- Se for a primeira vez a usar o kit DE2 no seu computador você deverá instalar o driver. Vá ao Gerenciador de Dispositivos do Windows, identifique o dispositivo Blaster não reconhecido, clique com o botão direito e selecione Atualizar Driver..., clique em Procurar Software de driver no Computador, e selecione o diretório c:/altera/13.0/quartus/drivers, clique em Ok e Avançar.



- Caso não apareça automaticamente o arquivo de programação do FPGA (.sof), selecione manualmente o arquivo a ser carregado na FPGA: Add File... > diretório output_files > arquivo.sof

- Clique em Start. O kit DE2 deverá apagar e instantes depois o seu circuito deverá estar carregado no FPGA.

- Teste o circuito implementado e faça um vídeo do circuito em funcionamento, coloque no YouTube e o link no relatório.

3) Terminando o Experimento

Ao finalizar o experimento, guarde **corretamente** os cabos, a fonte e a placa na caixa e devolva ao professor.

Lembre-se de reconectar na tomada os computadores do LINF!!!



2. PARTE EXPERIMENTAL

Projetar e testar um codificador e um decodificador segundo o código de Gray (**Tabela I**).

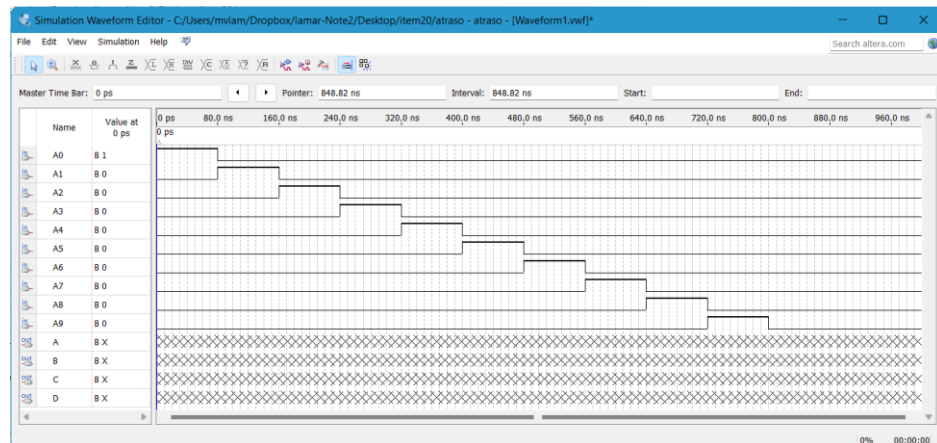
Tabela I – Código de Gray

Decimal	A	B	C	D
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	1	1
6	0	1	0	1
7	0	1	0	0
8	1	1	0	0
9	1	1	0	1

Obs.: O código decimal é representado por 10 símbolos diferentes. O símbolo ‘0’ é codificado como ‘0000000001’, o símbolo ‘1’ como ‘0000000010’, ... e o símbolo ‘9’ como ‘1000000000’.

2.1. Obtenha as funções booleanas para o codificador considerando as condições optativas (*don't care*) e faça um diagrama lógico total.

2.2. Desenhe no Quartus-II (**Pré-Projeto 1**) e faça as simulações funcional e temporal do diagrama esquemático total do codificador. Dica: defina os sinais de entrada como abaixo.



2.3. Obtenha as funções booleanas para o decodificador considerando as condições optativas (*don't care*) e faça um diagrama lógico total.

2.4. Desenhe no Quartus-II (**Pré-Projeto 2**) e faça as simulações funcional e temporal desse diagrama lógico total do decodificador. Apresente a tabela verdade completa (com todas as



possíveis combinações das entradas) obtida. Dica: Para gerar as entradas peça para fazer um contador de 4 bits em código Gray (ao invés de contagem em binário).

2.5. **(Pós-Experimento)** Desenhe o circuito da **Figura 4**, onde coder e decoder são os circuitos do codificador e do decodificador projetados nos itens 2.2 e 2.4, sintetize e teste no kit FPGA DE2 (ou DE2-70) filmando seu funcionamento.

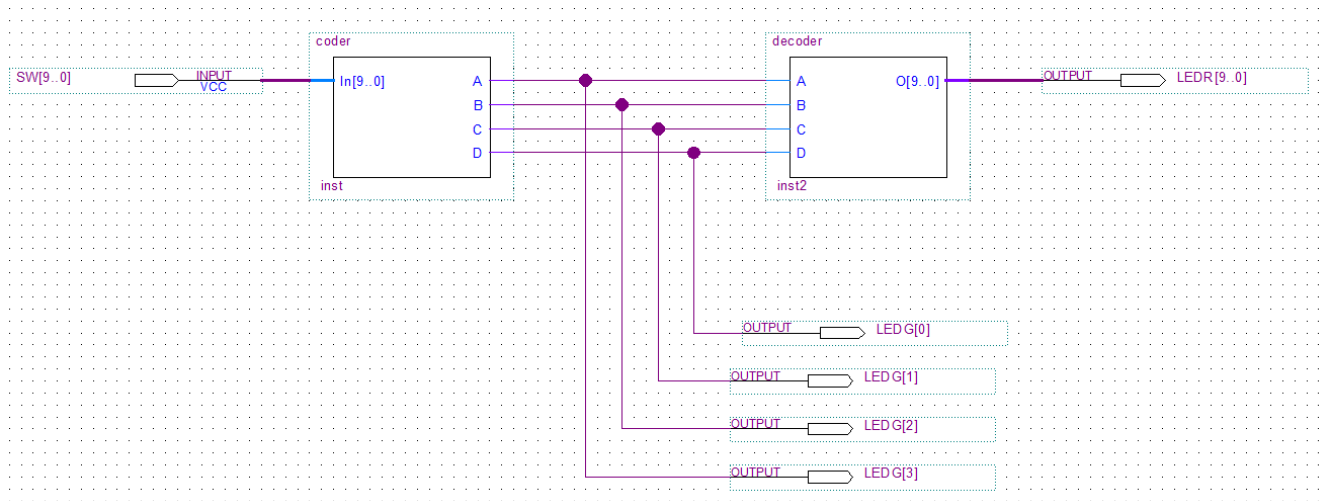


Figura 4 – Interface com o kit FPGA DE2

3. SUMÁRIO

Chama-se codificador um conversor de códigos em que o código de entrada é o decimal e o decodificador é aquele em que o código de saída é o decimal. Estudou-se um codificador e um decodificador em que o código binário utilizado é o de Gray.

4. EQUIPAMENTOS E MATERIAL

- Software Quartus-II versão 13.0 SP1
- Kit de desenvolvimento em FPGA DE2 ou DE2-70 Intel
- Pendrive 4GiB



5. TESTE DE AUTOAVALIAÇÃO

1. A saída em código BCD de um codificador, em que a chave do dígito 6 está acionada, será:
 - a) 0101
 - b) 1010
 - c) 0110
 - d) 1001
2. O código BCD **0111** corresponde ao seguinte número decimal:
 - a) 3
 - b) 6
 - c) 7
 - d) 9
3. A saída do codificador da **Figura 2** está no código BCD. Foi preenchida uma tabela da verdade e notou-se que tudo estava perfeito, exceto que em uma certa saída foi obtido DCBA = 0100 em vez de DCBA = 0101. Qual dos seguintes defeitos seria o mais provável?
 - a) Porta A defeituosa.
 - b) Ligação interrompida entre chave 8 e porta A.
 - c) Ligação interrompida entre chave 7 e porta A.
 - d) Ligação interrompida entre chave 7 e porta C.
4. Ao se implementar o codificador da **Figura 2**, notou-se que tudo estava correto, exceto que:

DCBA = 0000 foi obtido em vez de DCBA = 0010.
DCBA = 0001 foi obtido em vez de DCBA = 0011.
DCBA = 0100 foi obtido em vez de DCBA = 0110.
DCBA = 0101 foi obtido em vez de DCBA = 0111.

Qual dos seguintes defeitos seria o mais provável?

 - a) Porta A defeituosa.
 - b) Porta B defeituosa.
 - c) Porta D defeituosa.
 - d) Ligação interrompida entre a chave 5 e a porta B.

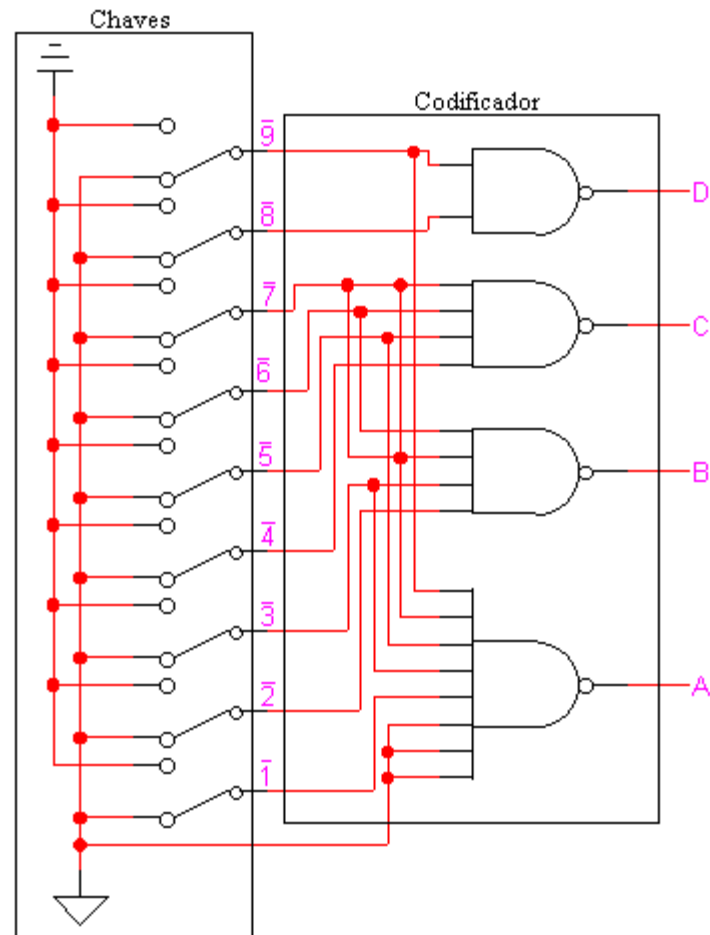


Figura 2 – Codificador BCD
A barra sobre os números das entradas nos diz que uma entrada está ativa quando em 0.

5. Ao se implementar o decodificador da **Figura 3**, notou-se que tudo estava perfeito, exceto que a saída correspondente ao decimal 7 permanecia sempre no nível lógico 1. Qual dos seguintes defeitos seria o mais provável?
 - a) Porta NOT D defeituosa, sempre com nível 0 na saída.
 - b) Porta NOT D defeituosa, sempre com nível 1 na saída.
 - c) Porta AND 7 defeituosa, sempre com nível 1 na saída.
 - d) Porta AND 7 defeituosa, sempre com nível 0 na saída.
 - e) Porta NOT A defeituosa, sempre com nível 0 na saída.
 - f) Porta NOT A defeituosa, sempre com nível 1 na saída.
6. Qual seria o defeito mais provável entre as opções da questão **5**, se tudo estivesse funcionando perfeito, exceto que as saídas correspondentes aos decimais pares sempre permanecem no nível lógico 0?
7. Qual seria o defeito mais provável entre as opções da questão **5**, se tudo estivesse funcionando perfeito, exceto que ao acender a saída correspondente ao dígito 8, também acende a saída correspondente ao dígito 0, e que ao acender a 9, também acende a 1?

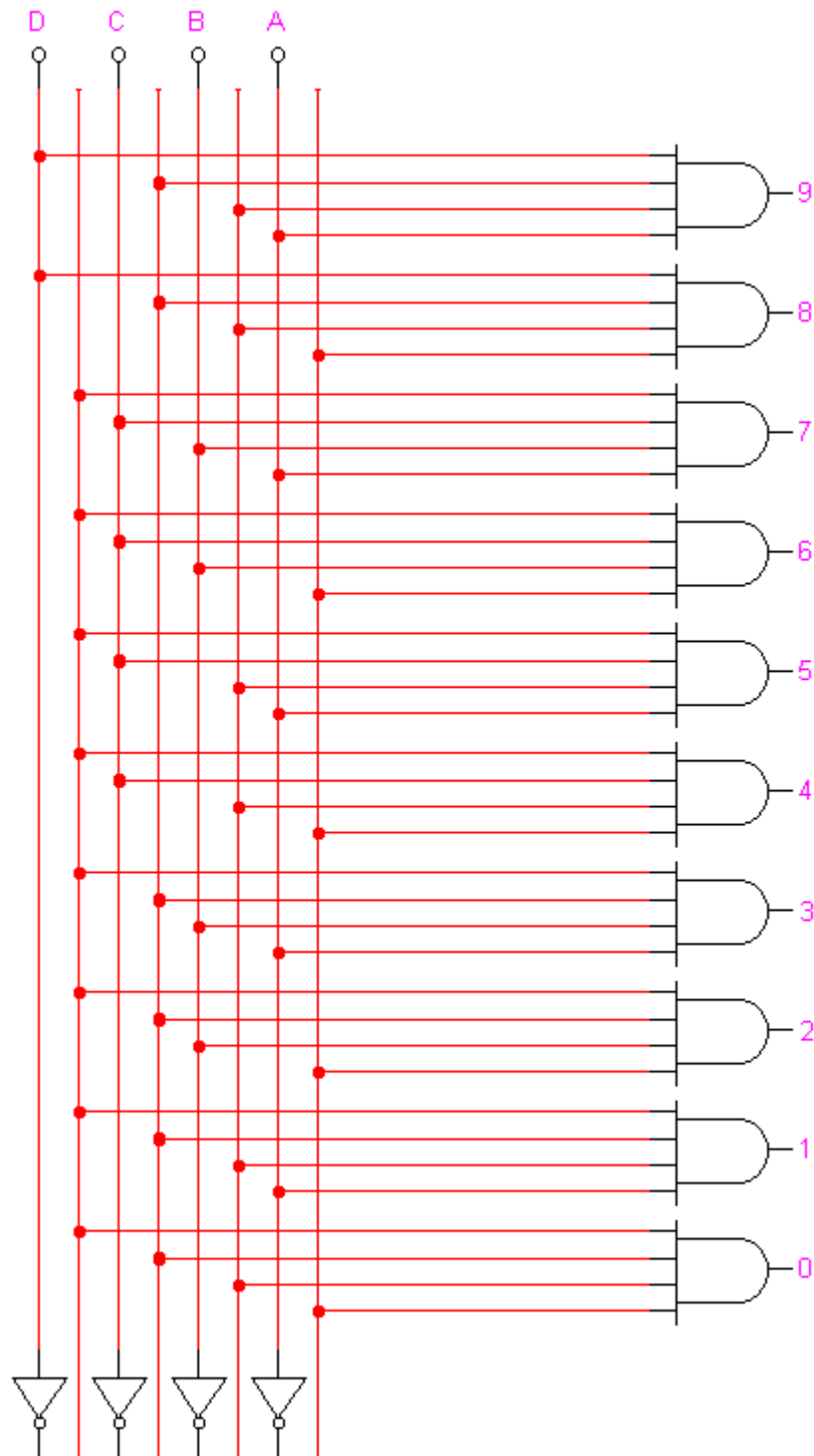


Figura 3 – Decodificador BCD

8. Qual seria o defeito mais provável entre as opções da questão 5, se tudo estivesse funcionando perfeito, exceto que ao acender a saída correspondente a um dígito ímpar, a saída do dígito par imediatamente inferior também acende?