

Aulas 2 e 3

Conversores de Código

Autores

Gabriel A. F. Souza, Gustavo D. Colletta, Leonardo B. Zoccal, Odilon O. Dutra

Unifei

Histórico de Revisões

13 de fevereiro de 2025	1.0	Primeira versão do documento.
-------------------------	-----	-------------------------------

Tópicos

- Introdução
- Tipos de Conversores de Código
- Implementação em Verilog
- Simulação e Verificação
- Atividades Hands-on

Introdução

Objetivos

- Compreender o conceito de circuitos conversores de códigos.
- Explorar suas aplicações na eletrônica digital.
- Apresentar uma visão geral das abordagens de descrição em Verilog.

O que são Conversores de Códigos?

- Circuitos que transformam um código de entrada em outro código de saída.
- Amplamente utilizados em sistemas digitais, comunicação e processamento de sinais.

Exemplos de Conversores de Código

- **Codificadores:** Binário para decimal, Binário para Gray, etc.
- **Decodificadores:** Decimal para binário, 7-segmentos, etc.
- **Conversores Numéricos:** Binário para Gray, BCD para Binário, Excesso-3, etc.

Aplicações Práticas

- Teclados e Displays
 - Conversores BCD para 7 segmentos usados para exibir números decimais em displays de LED ou LCD.
 - Conversores de matriz de teclado (ex: teclado de 4x4) para código binário usados para identificar tecla pressionada.

- Comunicação de Dados
 - Conversores de código Gray para binário e vice-versa usados em sistemas de posicionamento e encoders para minimizar erros na leitura de posição.
 - Conversores ASCII para binário comuns em sistemas de comunicação e armazenamento de dados.

Aplicações Práticas

- Memórias e Processadores
 - Conversores de endereçamento, como decodificadores binário-para-decimal, usados em memórias para selecionar células específicas.
 - Conversores de código excesso-3 utilizados em operações aritméticas para facilitar certos cálculos em hardware.

- Sistemas de Controle e Automação
 - Conversores de código binário para Gray usados em sistemas de controle de motores para evitar erros de transição entre estados.
 - Conversores entre diferentes padrões de codificação para integração entre dispositivos.

Aplicações Práticas

- Circuitos de Segurança e Criptografia
 - Conversores de codificação empregados em sistemas de segurança para transformar dados em representações seguras antes da transmissão.

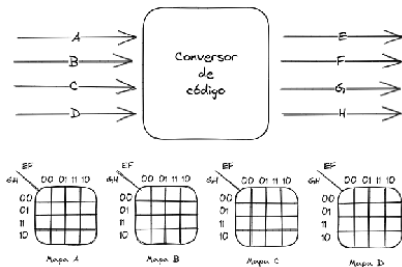
Descrição em Verilog

- **Fluxo de Dados (Dataflow)**
 - Baseado em expressões booleanas.
 - Define circuitos usando atribuições contínuas (`assign`).
- **Estrutural (Structural)**
 - Usa interconexão de módulos e portas lógicas.
 - Representa o hardware físico de forma explícita.

- **Comportamental (Behavioral)**
 - Utiliza blocos `always`, `case`, `if` para descrever o funcionamento do circuito.
 - Mais próximo da lógica algorítmica do que da estrutura física.

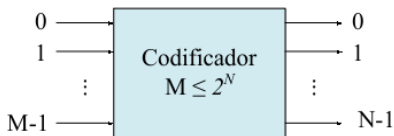
Tipos de Conversores de Código

Conversores Numéricos



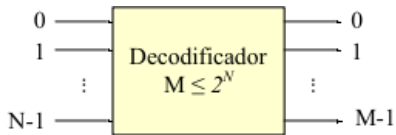
- O número de linhas de saída corresponde ao número de mapas de karnaugh.
- O número de linhas de entrada corresponde ao número de variáveis dos mapas de Karnaugh.

Codificadores



- Codifica M linhas de entrada em N linhas de saída.
- Apenas uma linha de entrada está ativa por vez.

Decodificadores



- Decodifica N linhas de entrada em M linhas de saída.
- O código presente nas entradas corresponde à linha de saída que deve estar ativa.

Implementação em Verilog

Exemplo Conversor de Código

- Tabela de Conversão BCD 5311 \rightarrow BCD 8421

Decimal	BCD 5311	BCD 8421
Variáveis	HGFE	DCBA
0	0000	0000
1	0001	0001
2	0011	0010
3	0100	0011
4	0101	0100
5	0111	0101
6	1001	0110
7	1011	0111
8	1100	1000
9	1101	1001

Expressões Lógicas

FE \ HG	00	01	11	10
	00	01	11	10
00	0	0	1	-
01	0	0	1	0
11	0	0	-	0
10	-	-	-	-

$$D = HG$$

FE \ HG	00	01	11	10
	00	01	11	10
00	0	0	0	-
01	0	1	0	1
11	0	1	-	1
10	-	-	-	-

$$C = \overline{H}GE + H\overline{G}$$

FE \ HG	00	01	11	10
	00	01	11	10
00	0	1	0	-
01	0	0	0	1
11	1	0	-	1
10	-	-	-	-

$$B = \overline{H}G\overline{E} + \overline{G}F + H\overline{G}$$

FE \ HG	00	01	11	10
	00	01	11	10
00	0	1	0	-
01	1	0	1	0
11	0	1	-	1
10	-	-	-	-

$$A = \overline{H}G\overline{F}E + \overline{H}G\overline{E} + GF + HGE + HF$$

Figura 1: Mapa de Karnaugh e expressões booleanas minimizadas

Esquemático do circuito

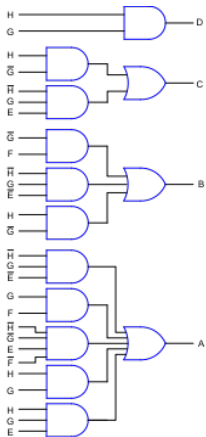


Figura 2: Circuito do conversor de códigos

Implementação em Verilog

- Descrição por Fluxo de Dados

```
1  module conversor(H,G,F,E,D,C,B,A);
2  input  H,G,F,E;
3  output D,C,B,A;
4
5  assign D = H & G;
6  assign C = (!H & G & E) | (H & !G);
7  assign B = (!H & G & !E) | (!G & F) | (H & !G);
8  assign A = (!H & !G & !F & E) | (!H & G & !E) | (G & F) |
              (H & G & E) | (H & F);
9
10 endmodule
```


Simulação e Verificação

Testbench

```
1  module conversor_tb();
2  reg H,G,F,E;
3  wire D,C,B,A;
4
5  conversor DUT(H,G,F,E,D,C,B,A);
6
7  initial begin
8      {H,G,F,E} = 4'b0000; #10;
9      {H,G,F,E} = 4'b0001; #10;
10     {H,G,F,E} = 4'b0011; #10;
11     {H,G,F,E} = 4'b0100; #10;
12     {H,G,F,E} = 4'b0101; #10;
13     {H,G,F,E} = 4'b0111; #10;
14     {H,G,F,E} = 4'b1001; #10;
15     {H,G,F,E} = 4'b1011; #10;
16     {H,G,F,E} = 4'b1100; #10;
17     {H,G,F,E} = 4'b1101; #10;
18     $stop;
19 end
20 endmodule
```

Forma de Onda

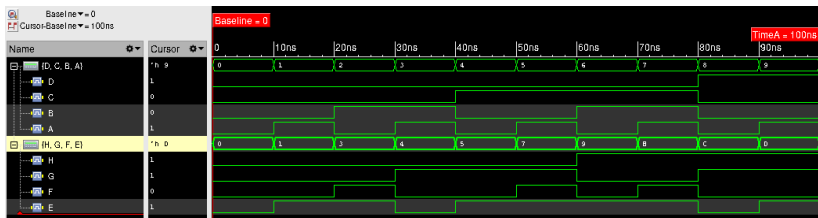


Figura 3: Resultado da simulação com testbench do conversor de código

Atividades Hands-on

Atividade 1

- Monte uma descrição estrutural do conversor de código BCD5311 para BCD8421
- Cada saída (D,C,B,A) deve ser gerada por um módulo, que internamente descreverá a função lógica usando primitivas.
- O módulo topo deverá instanciar os quatro módulos correspondentes a cada saída.
- Simule a operação do circuito por meio de Testbench.

Atividade 1

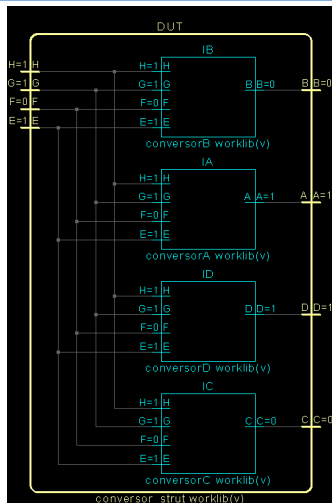


Figura 4: Descrição estrutural com 4 módulos internos

Atividade 2

- O código Excesso de 3 é útil na correção de erros devido à sua propriedade de código auto-complementar, além de simplificar operações aritméticas em processadores digitais.
- Utilizando mapas de Karnaugh, encontre as expressões lógicas necessárias para implementar um conversor de código BCD8421 para Excesso de 3.
- Implemente o circuito em verilog utilizando descrição do tipo Fluxo de Dados
- Elabore um Testbench e simule a operação do circuito.

Atividade 2

- Tabela de conversão BCD8421 para Excesso de 3

BCD (8421)	Excesso-3
0000 (0)	0011 (3)
0001 (1)	0100 (4)
0010 (2)	0101 (5)
0011 (3)	0110 (6)
0100 (4)	0111 (7)
0101 (5)	1000 (8)
0110 (6)	1001 (9)
0111 (7)	1010 (10)
1000 (8)	1011 (11)
1001 (9)	1100 (12)

Atividade 3

- O código Gray garante que, entre valores sucessivos, apenas um bit muda por vez.
- É muito utilizado em diversas aplicações, incluindo:
 - Sensores de posição de sistemas mecânicos para evitar erros de leitura
 - Contadores assíncronos e Máquinas de Estados para evitar Glitches
 - Transmissão de Dados para minimizar erros de transmissão

Atividade 3

- A tabela mostra a conversão de binário para gray com 4 bits.

Binário	Código Gray
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

Atividade 3

- Utilizando mapas de Karnaugh, encontre as expressões lógicas necessárias para implementar um conversor de código binário de 4 bits para código gray.
- Implemente o circuito em verilog utilizando descrição do tipo Fluxo de Dados
- Elabore um Testbench e simule a operação do circuito.