EE 648 – VLSI Design

Binary Coded Hexadecimal to Seven-Segment Display Converter

Ryker Dial

Cody Gossel

Zach Krehlik

March 10, 2017

Contents

1 Introduction

The objective of this project is to design and fabricate an integrated circuit that takes four input bits representing a hexadecimal number and displays that number on a seven-segment display. Such a circuit will be useful for a microcontroller because it reduces the number of pins required to use the display from seven to four, freeing up GPIO pins for other tasks.

2 Top Level Design

To facilitate the design of this circuit, the logic for each segment is separated into its own module. This converter is being designed for an active-low seven-segment display, so the output of each module will be used to switch a FET that pulls the segment to ground when switched on, turning on the segment. The top-level design for the converter circuit with these modules included is shown in Figure 1.

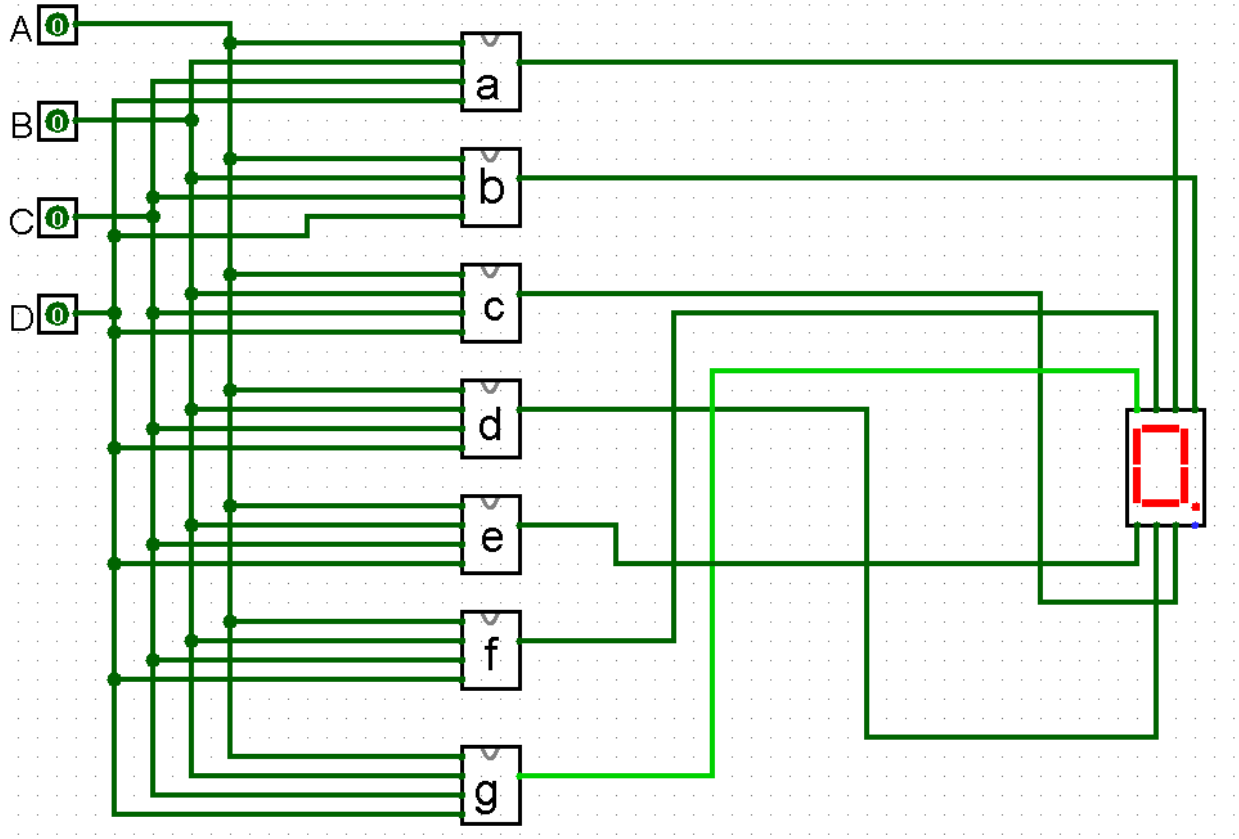

Figure 1: Converter Top Level layout

3 Detailed Design

3.1 Gate Level

The truth table for the converter, which includes its four inputs and seven outputs, is shown in Table 1. A value of zero corresponds to the segment being on, and vice-versa. From this, truth tables for each individual output were obtained and transferred to a program called Logisim, which is a free tool for designing and simulating logic circuits. This tool was used to generate minimized NAND-only Boolean expressions for each module, as using only inverting logic will reduce the number of inverters required to realize the converter circuit. Logisim was also used to generate the gate-level schematics for each module; the gate-level schematics for each module can be found in Appendix A, and the corresponding logic functions can be found in Appendix B.

Table 1: Converter Truth Table

Inputs				**Outputs**						
A	**B**	**C**	**D**	**a**	**b**	**c**	**d**	**e**	**f**	**g**
0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	1	1	1	1
0	0	1	0	0	0	1	0	0	1	0
0	0	1	1	0	0	0	0	1	1	0
0	1	0	0	1	0	0	1	1	0	0
0	1	0	1	0	1	0	0	1	0	0
0	1	1	0	0	1	0	0	0	0	0
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	1	1	0	0
1	0	1	0	0	0	0	1	0	0	0
1	0	1	1	1	1	0	0	0	0	0
1	1	0	0	0	1	1	0	0	0	1
1	1	0	1	1	0	0	0	0	1	0
1	1	1	0	0	1	1	0	0	0	0
1	1	1	1	0	1	1	1	0	0	0

From the logic equations and gate-level schematics, note that the complement of any particular input is used many times. In the actual implementation of the circuit, as opposed to what is shown in the gate-level schematics, the complements of the input signals will be produced only once and reused as needed, greatly reducing the number of inverters in the circuit. Additionally, there are several terms in the logic equations that appear more than once: specifically $\overline{(\bar{A}B\bar{C}\bar{D})}$, $\overline{(AB\bar{C}D)}$, $\overline{(AB\bar{D})}$, and $\overline{(\bar{B}\bar{C}D)}$; the outputs of these gates will be reused as well.

To verify the functionality of the circuit, Logisim was used to simulate the output for each of the sixteen possible inputs. The input and output waveforms were then plotted and are shown in Figure 2. Comparing the simulated output to the converter circuit's truth table verifies that the design is valid.

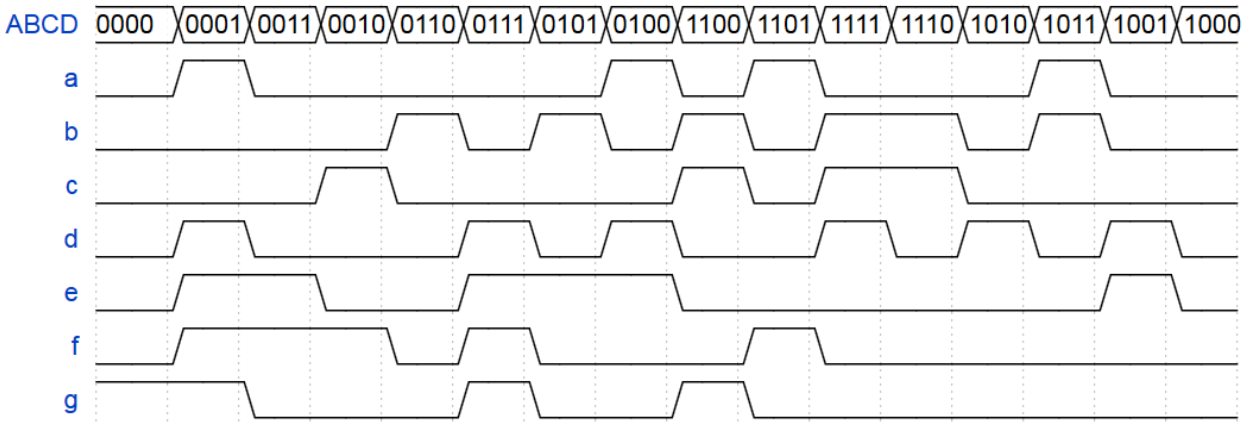

Figure 2: Simulated Input and Output of Converter

3.2 Transistor Level

3.2.1 Transistor Level Schematics

By using Logisim to implement each module with only NAND and inverter gates, the converter circuit can be realized with only a few gates: specifically two, three, and four input NAND gates, and an inverter. Efficient implementations of these gates can be designed once and reused as needed, though different sizes of these gates will need to be made to optimize for delay and power consumption. The transistor level design of these gates is shown in Figures 3, 4, 5, and 6.

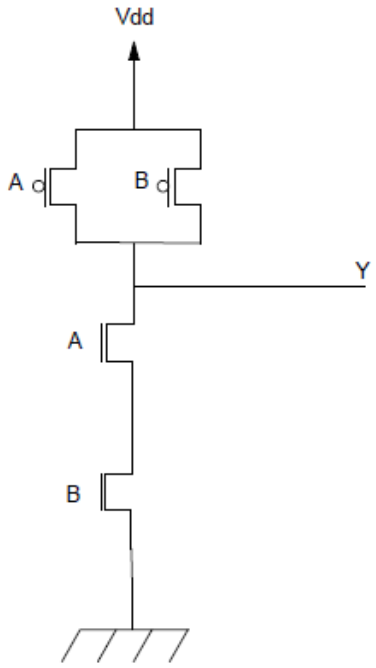

Figure 3: Transistor Level Schematic for Two-Input NAND

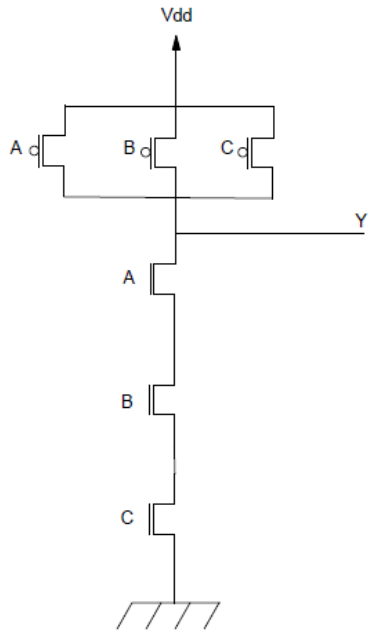

Figure 4: Transistor Level Schematic for Three-Input NAND

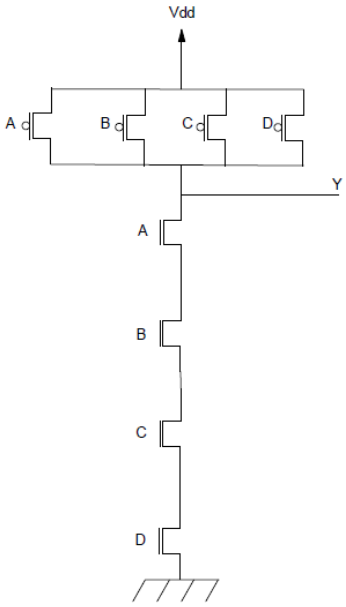

Figure 5: Transistor Level Schematic for Four-Input NAND

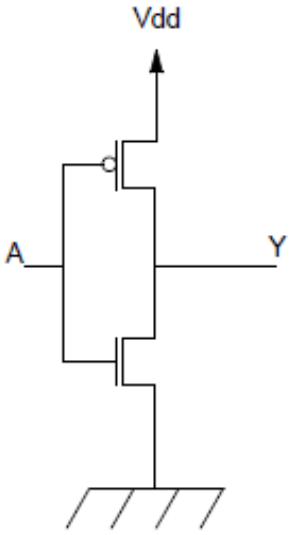

Figure 6: Transistor Level Schematic for Inverter

3.2.2 Transistor Sizing

Sizing the transistors to optimize for delay first requires knowing the input and output capacitance for each module of the converter circuit. Since the circuit only generates inverted

Output capac-itance

inputs once and reuses them, the input capacitance depends upon the largest fanout required for an inverted input. The required fanout for each inverted input is determined by counting the number of times each inverted term appears in the logic equations in Appendix B, while accounting for terms appearing in the duplicate subcircuits mentioned in Section 3.1. Doing so yields fanouts for $\bar{A}$, $\bar{B}$, $\bar{C}$, and $\bar{D}$ of 11, 8, 8, and 6, respectively. Each inverter will be sized so that it has the same driving strength as a unit-sized inverter with a fanout of one.

When optimizing for delay, the input capacitance of interest is that of the critical path. Since the inverter for $\bar{A}$ has the largest fanout it will have the largest input and output capacitance. Since each module contains an $\bar{A}$ term, the critical path for each module will start with this inverter.

(relate fanout and transistor size)

Inverter driving a NAND, so load cap is h*(kPMOS + kNMOS)*C, but what are the k values?

A Segment Logic Diagrams

A.1 Segment a

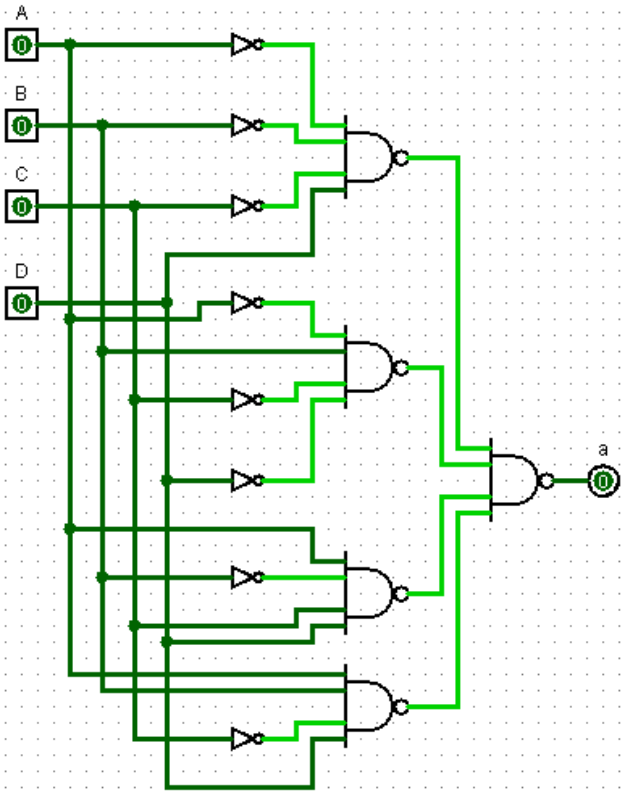

Figure 7: Block a Gate Level Schematic

A.2 Segment b

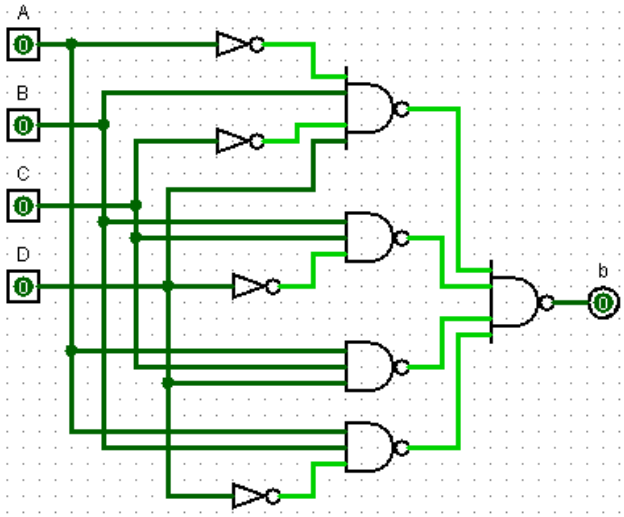

Figure 8: Block b Gate Level Schematic

A.3 Segment c

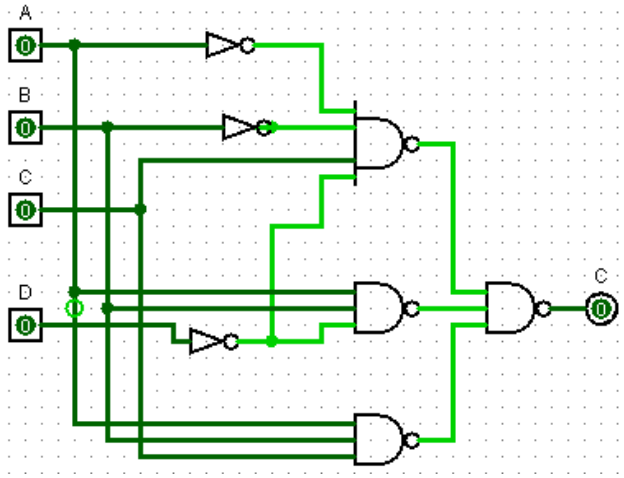

Figure 9: Block c Gate Level Schematic

A.4 Segment d

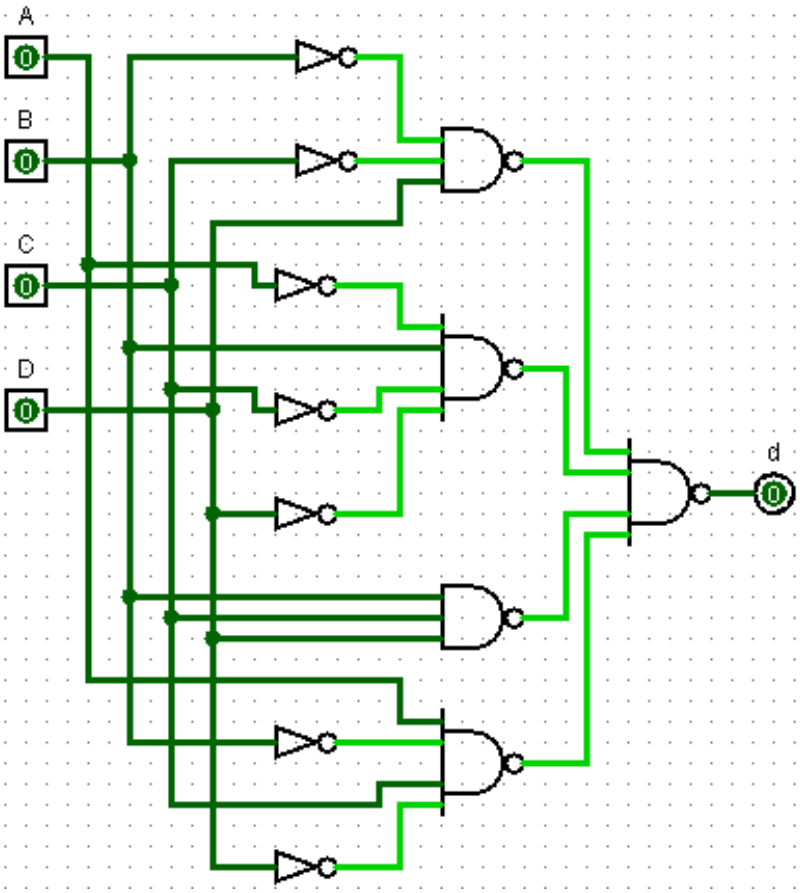

Figure 10: Block d Gate Level Schematic

A.5 Segment e

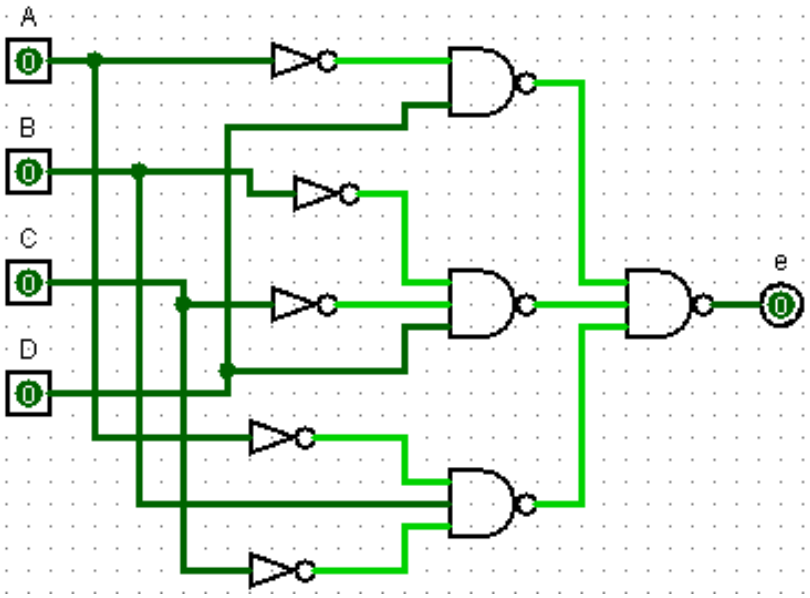

Figure 11: Block e Gate Level Schematic

A.6 Segment f

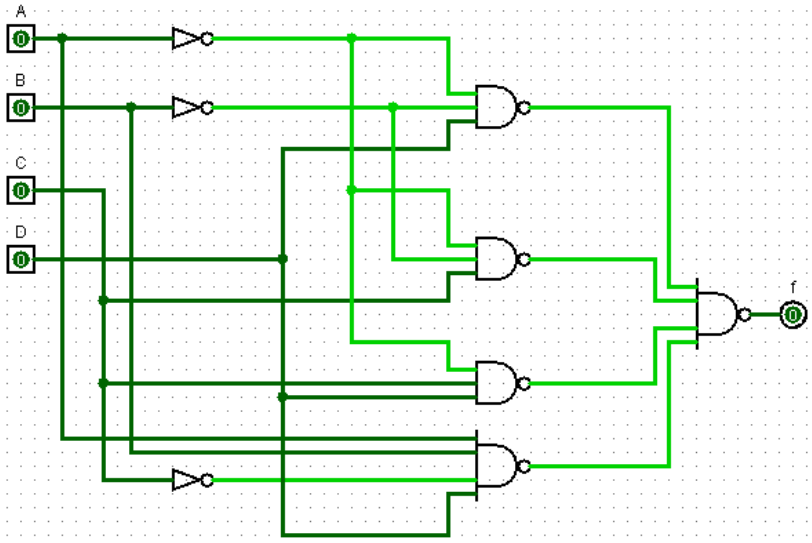

Figure 12: Block f Gate Level Schematic

A.7 Segment g

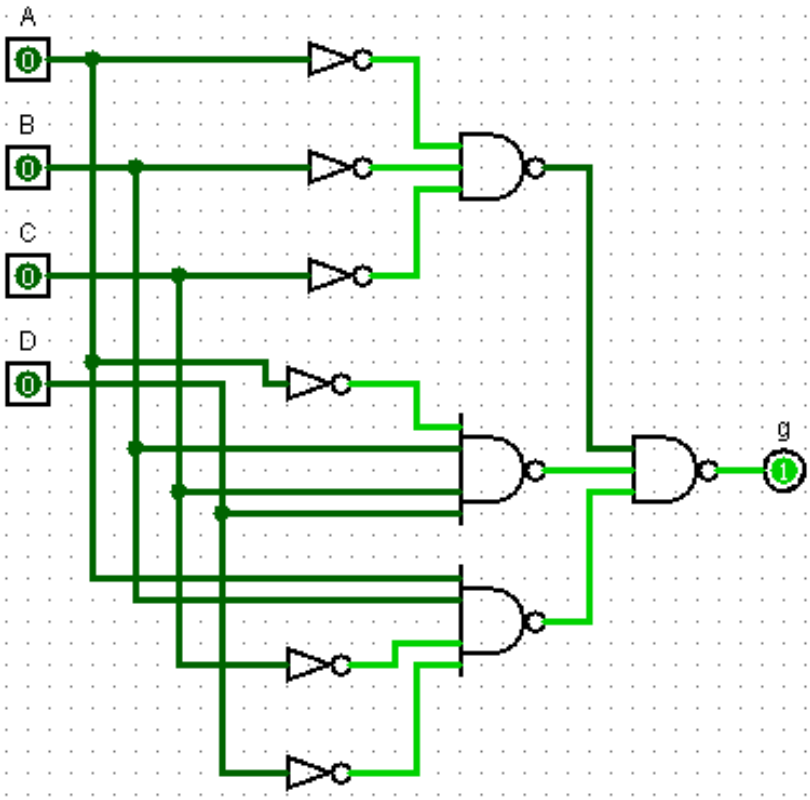

Figure 13: Block g Gate Level Schematic

B Logic Equations

$$a = \overline{\overline{(\bar{A}\bar{B}\bar{C}D)}\,\overline{(\bar{A}B\bar{C}\bar{D})}\,\overline{(A\bar{B}CD)}\,\overline{(AB\bar{C}D)}} \tag{1}$$

$$b = \overline{\overline{(\bar{A}B\bar{C}D)}\,\overline{(BC\bar{D})}\,\overline{(ACD)}\,\overline{(AB\bar{D})}} \tag{2}$$

$$c = \overline{\overline{(\bar{A}\bar{B}CD)}\,\overline{(AB\bar{D})}\,\overline{(ABC)}} \tag{3}$$

$$d = \overline{\overline{(\bar{B}\bar{C}D)}\,\overline{(\bar{A}B\bar{C}\bar{D})}\,\overline{(BCD)}\,A\bar{B}C\bar{D}} \tag{4}$$

$$e = \overline{\overline{(\bar{A}D)}\,\overline{(\bar{B}\bar{C}D)}\,\overline{(\bar{A}B\bar{C})}} \tag{5}$$

$$f = \overline{\overline{(\bar{A}\bar{B}D)}\,\overline{(\bar{A}\bar{B}C)}\,\overline{(\bar{A}CD)}\,\overline{(AB\bar{C}D)}} \tag{6}$$

$$g = \overline{\overline{(\bar{A}\bar{B}\bar{C})}\,\overline{(\bar{A}BCD)}\,\overline{(AB\bar{C}\bar{D})}} \tag{7}$$