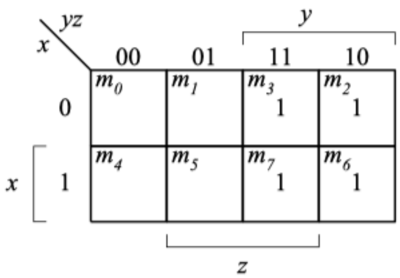
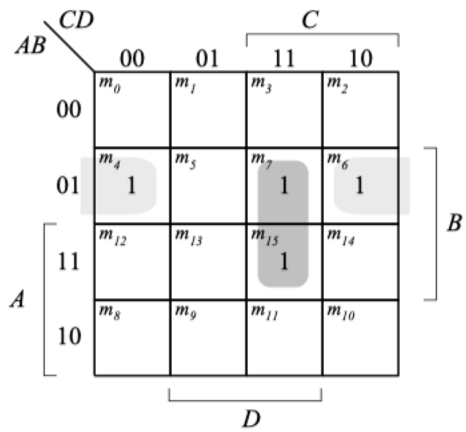
Gate-Level Minimization

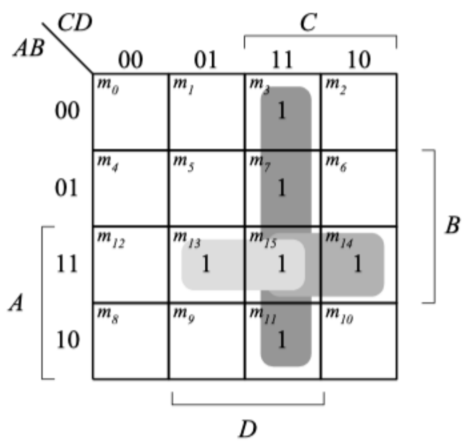
1) Simplify the following Boolean functions, using *Karnaugh* maps:



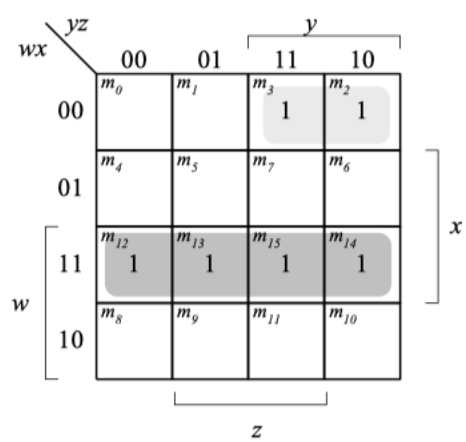
(a) $F = y$



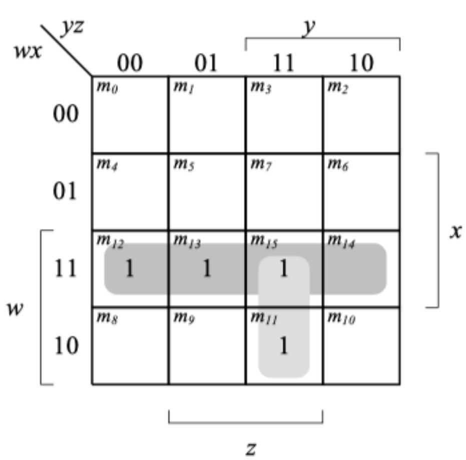
(b) $F = BCD + A'BD'$



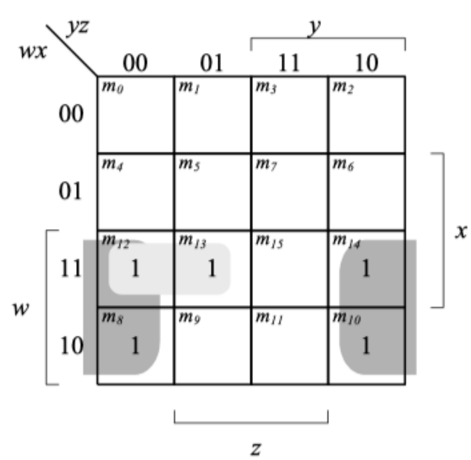
(c) $F = CD + ABD + ABC$



(d) $F = w'x'y + wx$



$F = wx + wyz$



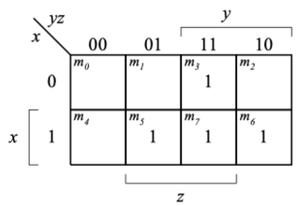
$F = wz' + xy'w$

(e)

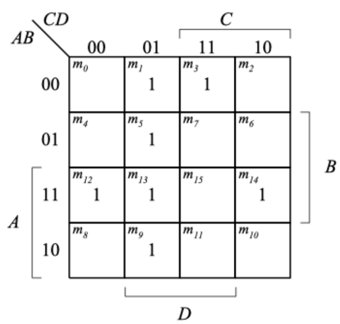
(f)

2) Find the minterms of the following Boolean expressions by first plotting each function in a map:

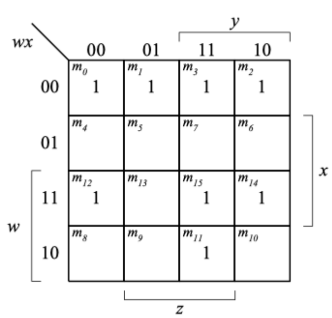
(a) $F(x, y, z) = \Sigma(3, 5, 6, 7)$



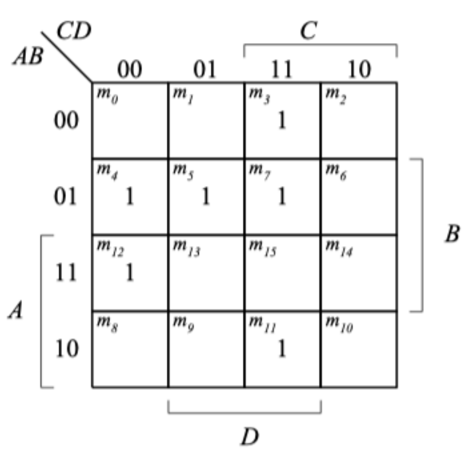
(b) $F = \Sigma(1, 3, 5, 9, 12, 13, 14)$



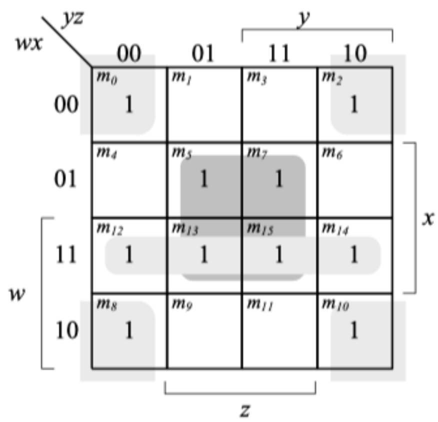
(c) $F = \Sigma(0, 1, 2, 3, 11, 12, 14, 15)$



(d) $F = \Sigma(3, 4, 5, 7, 11, 12)$

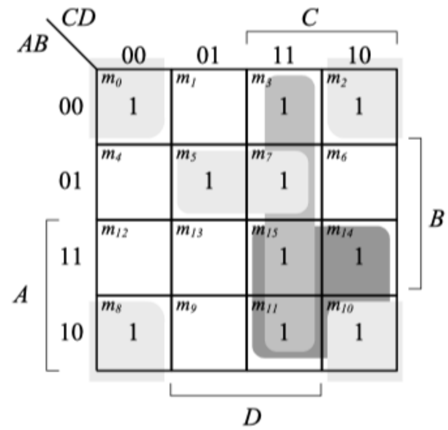


3) Simplify the following Boolean functions by first finding the essential prime implicants:



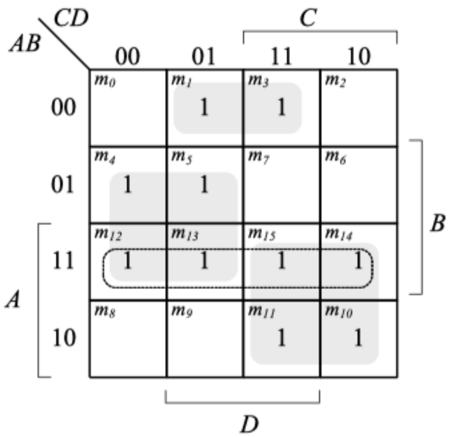
F = Σ(0, 2, 5, 7, 8, 10, 12, 13, 14, 15)
Essential: *xz, wx, x′z′*
$F = xz + wx + x'z'$

(a)



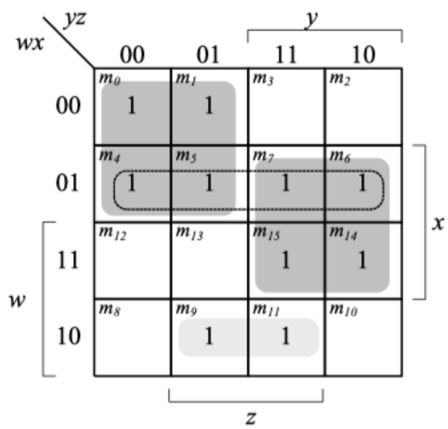
F = Σ(0, 2, 3, 5, 7, 8, 10, 11, 14, 15)
Essential: *AC, B′D′, CD, A′BD*
$F = AC + B'D' + CD + A'BD$

(b)



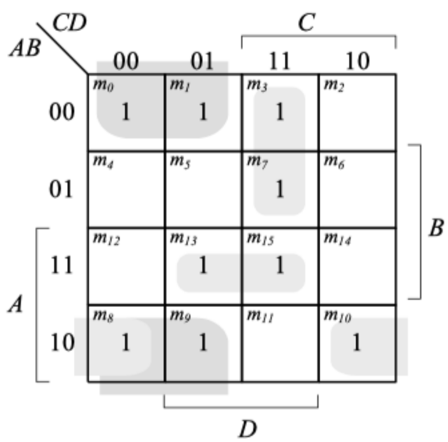
F = Σ(1, 3, 4, 5, 10, 11, 12, 13, 14, 15)
Essential: AC, BC', A'B'D
Non-essential: *AB, A'B'D, B'CD, A'C'D*
F = AC + BC' + A'B'D

(c)



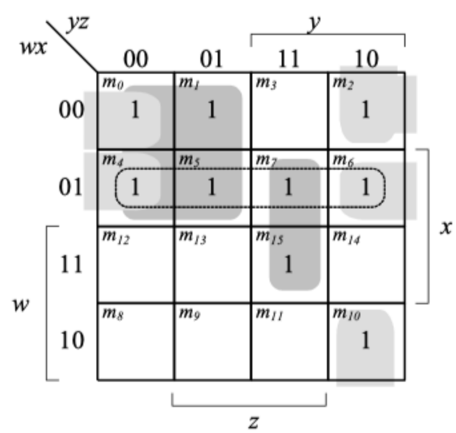
F = Σ(0, 1, 4, 5, 6, 7, 9, 11, 14, 15)
Essential: *w'y', xy, wx'z*
Non-essential: *wx, x'y'z, w'wz, w'x'z*
F = *w'y' + xy + wx'z*

(d)



F(A, B, C, D) = S(0, 1, 3, 7 8, 9, 10,13,15)
Essential: *B'C', AB'D'*
Non-essential: *ABD, A'CD, BCD*
F = B'C' + AB'D' +A'CD +ABD

(e)

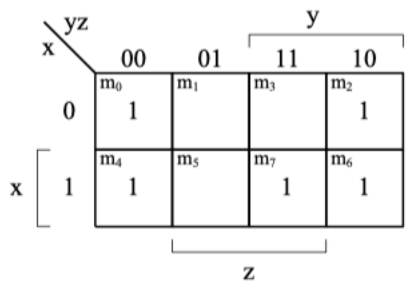


F = S(0, 1, 2, 4, 5, 6, 7, 10, 15)
Essential: *w'y', w'z', xyz, x'yz'*
Non-Essential: *w'x*
F = w'y' + w'z' + xyz + x'yz'

(f)

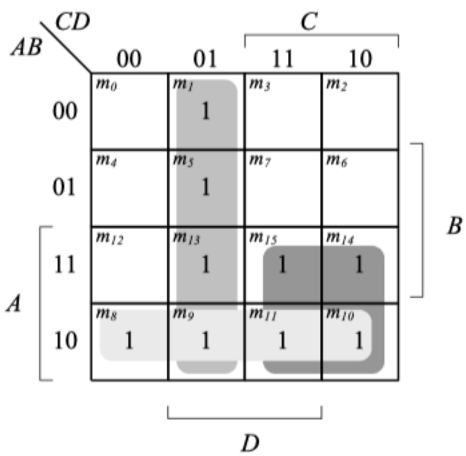
4) Simplify the following expressions to (1) sum-of-products and (2) products-of-sums:

(a) $F = x'z' + y'z' + yz' + xy = x'z' + z' + xy = z' + xy$



$F' = x'z + y'z$
$F = (x + z')(y + z')$

(b) $F = ACD' + C'D + AB' + ABCD$



$F = AC + AB' + C'D$
$F' = A'C + A'D' + BC'D'$
$F = (A + C')(A + D)(B'+C + D)$

(c)

$F = (A' + B + D')(A' + B' + C')(A' + B' + C)(B' + C + D')$
$F' = AB'D + ABC + ABC' + BC'D$

$AB \backslash CD$	00	01	11	10
00	m_0	m_1 0	m_3 0	m_2
01	m_4	m_5 0	m_7	m_6
11	m_{12} 0	m_{13} 0	m_{15} 0	m_{14} 0
10	m_8	m_9	m_{11}	m_{10}

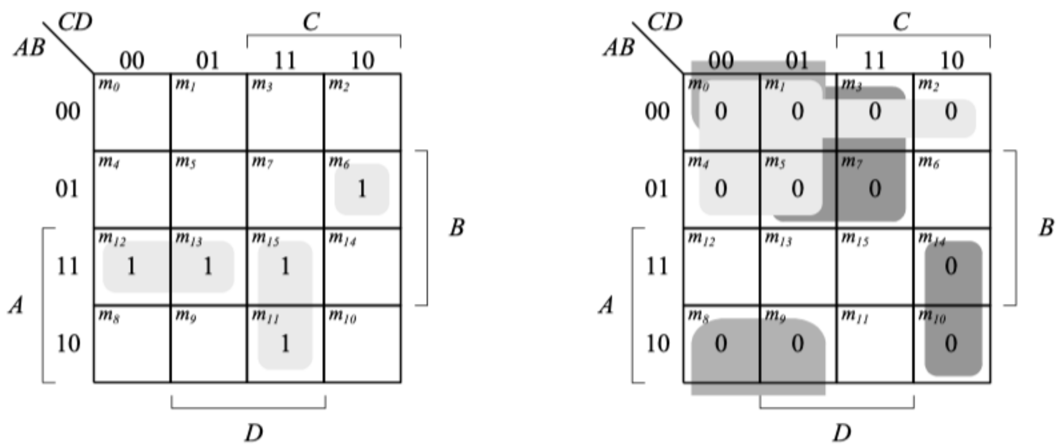
$F' = AB + BC'D$
$F = (A' + B')(B' + C + D')$

$F = A'D' + A'BC + AB'$

$AB \backslash CD$	00	01	11	10
00	m_0 1	m_1 0	m_3 0	m_2 1
01	m_4 1	m_5 0	m_7 1	m_6 1
11	m_{12} 0	m_{13} 0	m_{15} 0	m_{14} 0
10	m_8 1	m_9 1	m_{11} 1	m_{10} 1

(d)

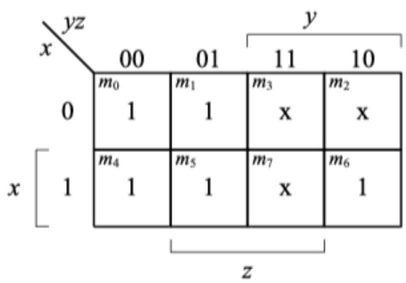
$F = BCD' + ABC' + ACD$



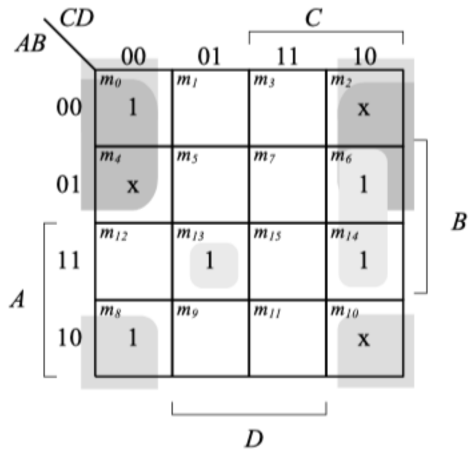
$F' = A'C' + A'D + B'C' + A'B' + ACD'$\

$F = (A + C)(A + D')\ (B + C)(A + B)(A' + C' + D)$

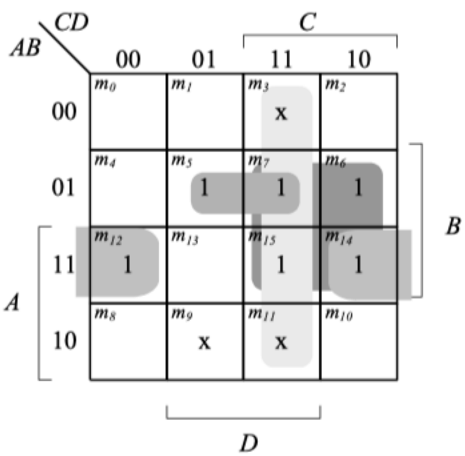
5) Simplify the following Boolean function F, together with the don't-care conditions d, and then express the simplified function in sum-of-minterms form:



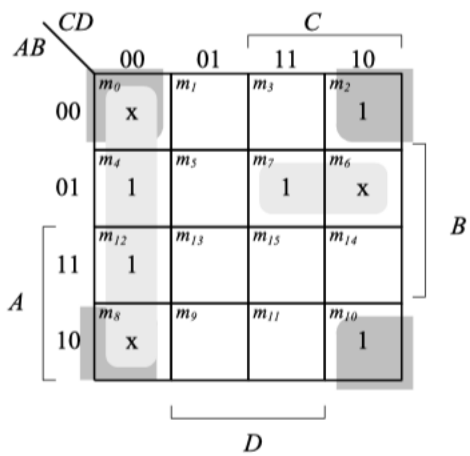
$F = 1$
$F = \Sigma(0, 1, 2, 3, 4, 5, 6, 7)$



$F = A'D' + B'D' + BCD' + ABC'D$
$F = \Sigma(0, 2, 4, 6, 8, 10, 13, 14)$



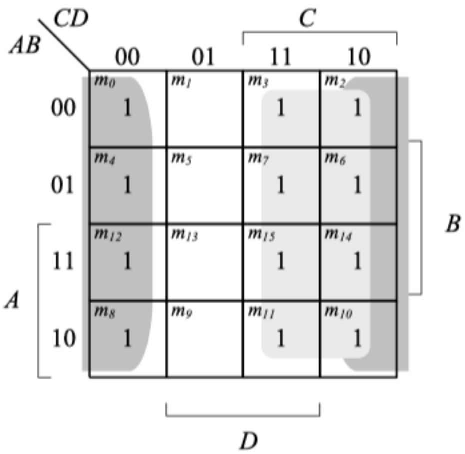
$F = BC + CD + ABD' + A'BD$
$F = \Sigma(3, 5, 6, 7, 11, 12, 14, 15)$



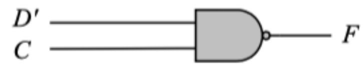
$F = B'D' + C'D' + A'BC$
$F =$ $F = \Sigma(0, 2, 4, 6, 7, 8, 10, 12)$

6) Simplify the following functions, and implement them with two-level NAND gate circuits:

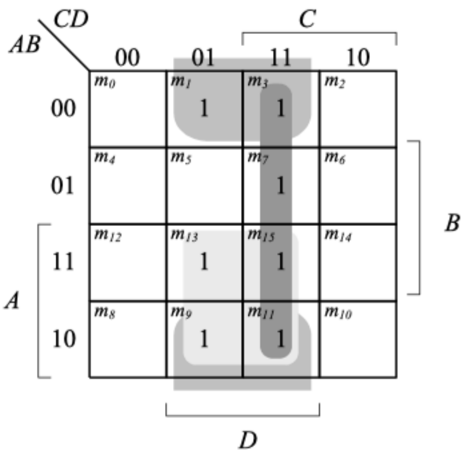
(a)



$$F = C + D'$$
$$F = (C'D)'$$

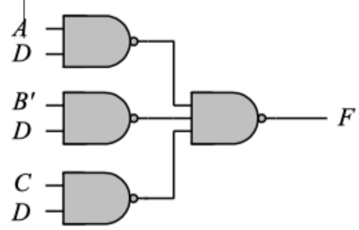


(b)



$F = AD + B'D + CD$

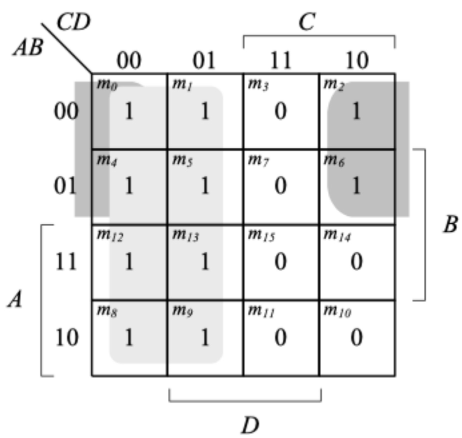
$F = ((AD)'(B'D)'(CD)')'$



(c) $F = (A' + C' + D')(A' + C')(C' + D')$

$F' = (A' + C' + D')' + (A' + C')' + (C' + D')'$

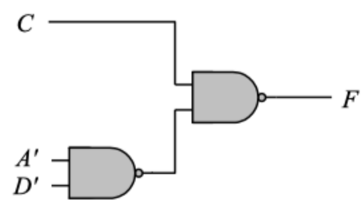
$F' = ACD + AC + CD$



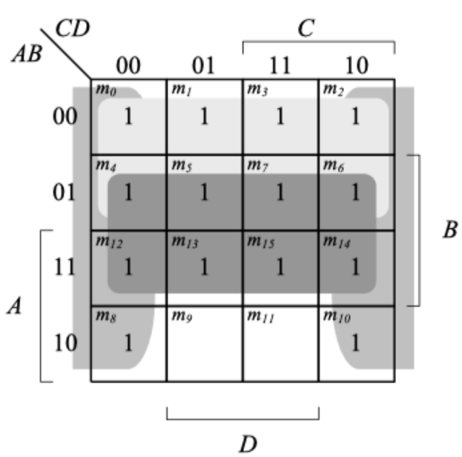
$F = C' + A'D'$

$F = (C(A + D))'$

$F = (C(A'D')')'$

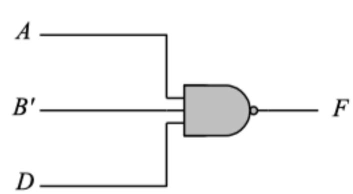


(d)



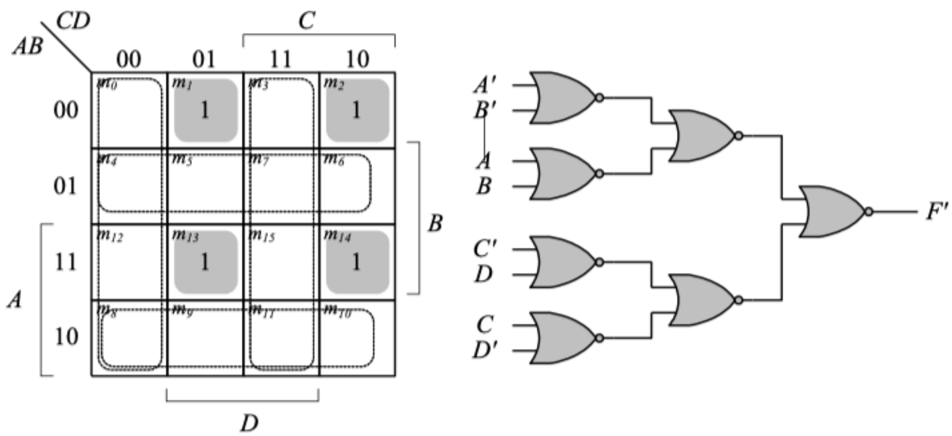
$F = A' + B + D'$

$F = (A(B')D)'$



7) Draw a logic diagram using only two-input NOR gates to implement the following function:
$F(A,B,C,D) = (A \oplus B)'(C \oplus D)$

$$F = (A \oplus B)'(C \oplus D) = (AB' + A'B)'(CD' + C'D)$$
$$= (AB + A'B')(CD' + C'D) = ABCD' + ABC'D + A'B'CD' + A'B'C'D$$
$$F' = (AB + A'B')' + (CD' + C'D)'$$
$$F' = ((A' + B')' + (A + B)')' + ((C' + D)' + (C + D')')'$$



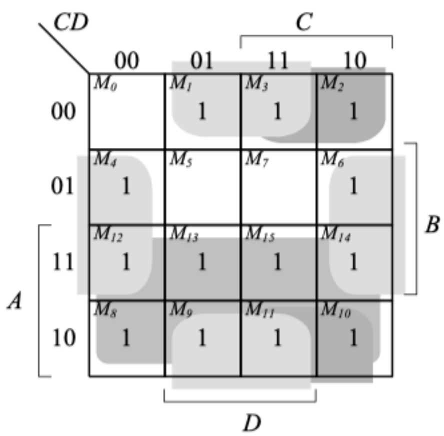
Combinational Logic

8) Consider the combinational circuit shown in the following figure:

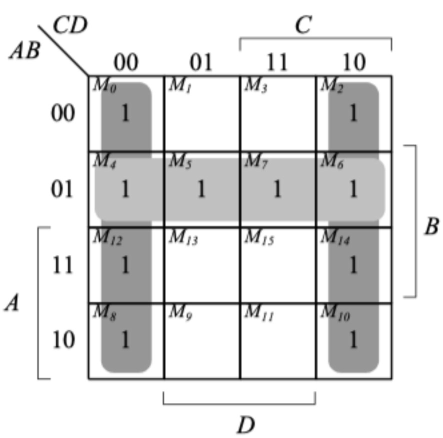
$T_1 = B'C$, $T_2 = A'B$, $T_3 = A + T_1 = A + B'C$,
$T_4 = D \oplus T_2 = D \oplus (A'B) = A'BD' + D(A + B') = A'BD' + AD + B'D$
$F_1 = T_3 + T_4 = A + B'C + A'BD' + AD + B'D$
With $A + AD = A$ and $A + A'BD' = A + BD'$:
$F_1 = A + B'C + BD' + B'D$
Alternative cover: $F_1 = A + CD' + BD' + B'D$

$F_2 = T_2 + D' = A'B + D'$

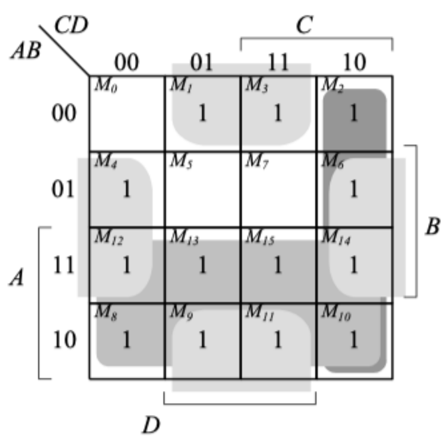
$ABCD$	T_1	T_2	T_3	T_4	F_1	F_2
0000	0	0	0	0	0	1
0001	0	0	0	1	1	0
0010	1	0	1	0	1	1
0011	1	0	1	1	1	0
0100	0	1	0	1	1	1
0101	0	1	0	0	0	1
0110	0	1	0	1	1	1
0111	0	1	0	0	0	1
1000	0	0	1	0	1	1
1001	0	0	1	1	1	0
1010	1	0	1	0	1	1
1011	1	0	1	1	1	0
1100	0	0	1	0	1	1
1101	0	0	1	1	1	0
1110	0	0	1	0	1	1
1111	0	0	1	1	1	0



$F_1 = A + B'C + B'D + BD'$



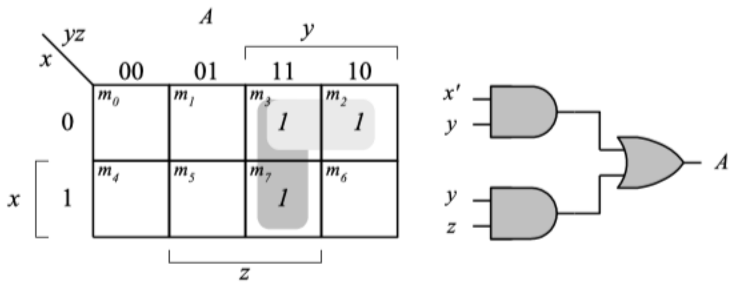
$F_2 = A'B + D'$



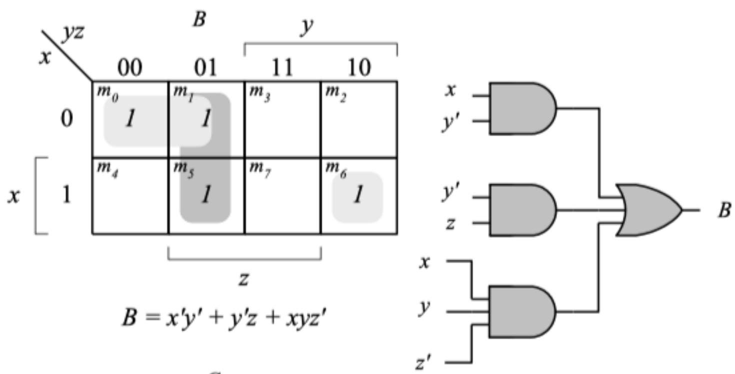
$F_1 = A + CD' + B'D + BD'$

9) Design a combinational circuit with three inputs, x, y, and z, and three outputs, A, B, and C. When the binary input is 0, 1, 2, or 3, the binary output is one greater than the input. When the binary input is 4, 5, 6, or 7, the binary output is two less than the input.

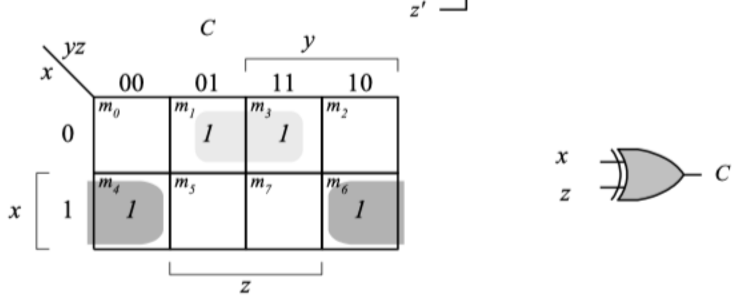
xyz	ABC
000	010
001	011
010	100
011	101
100	001
101	010
110	011
111	100



$A = x'y + yz$



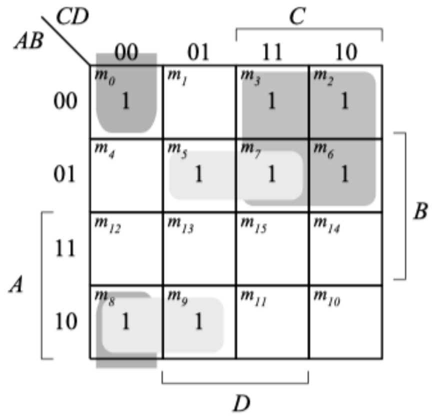
$B = x'y' + y'z + xyz'$



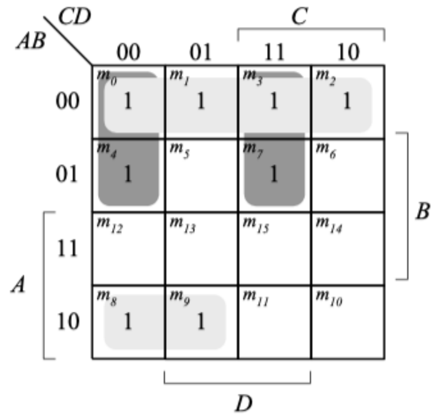
$C = x'z + xz'$

10) An BCD-to-seven-segment decoder is a combinational circuit that converts a decimal digit in BCD to an appropriate code for the selection of segments in an indicator used to display the decimal digit in a familiar form. The seven outputs of the decoder (a, b, c, d, e, f, g) select the corresponding segments in the display, as shown in Fig. (a). The numeric display chosen to represent the decimal digit is shown in Fig. (b). Using a truth table and Karnaugh maps, design the BCD-to-seven-segment decoder using a minimum number of gates. The six invalid combinations should result in a blank display. The following figure shows the segment designation and numbering scheme.

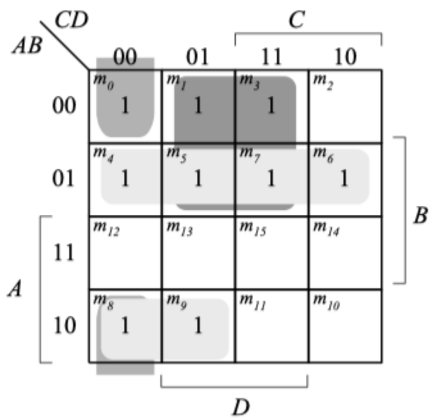
ABCD	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0000	1	1	1	1	1	1	0
0001	0	1	1	0	0	0	0
0010	1	1	0	1	1	0	1
0011	1	1	1	1	0	0	1
0100	0	1	1	0	0	1	1
0101	1	0	1	1	0	1	1
0110	1	0	1	1	1	1	1
0111	1	1	1	0	0	0	0
1000	1	1	1	1	1	1	1
1001	1	1	1	1	0	1	1



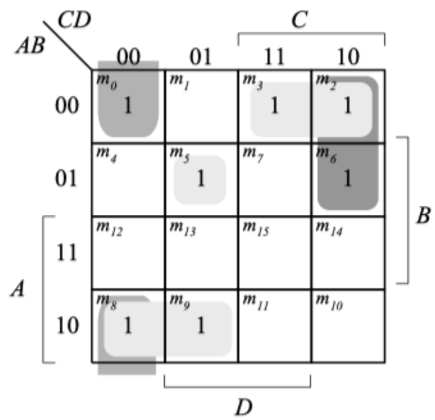
a = A'C + A'BD + B'C'D' + AB'C'



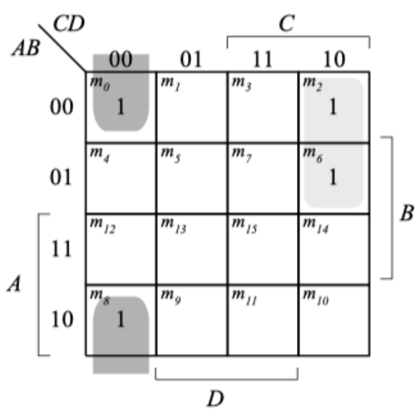
b = A'B' + A'C'D' + A'CD + AB'C'



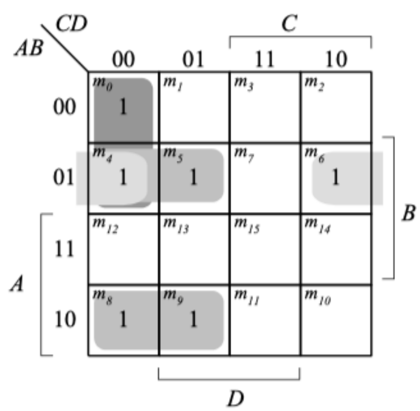
c = A'B + A'D + B'C'D' + AB'C'



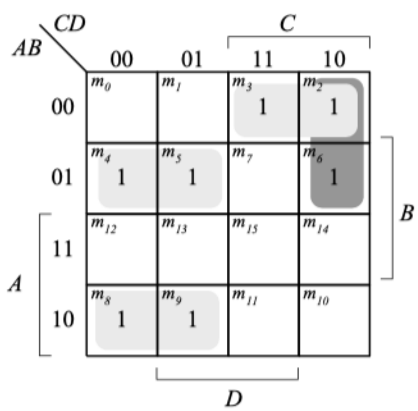
d = A'CD' + A'B' C+ B'C'D' + AB'C' + A'BC'D



e = A'CD' + B'C'D'



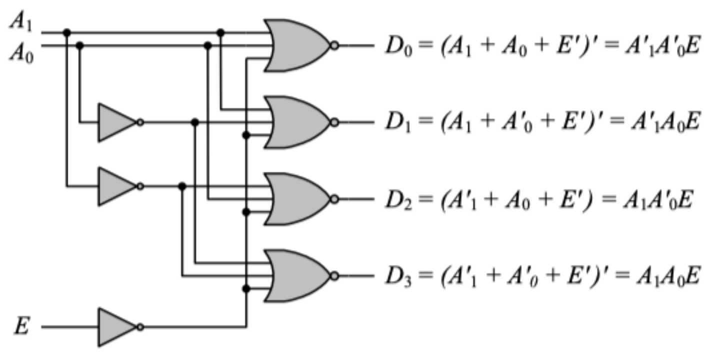
f = A'BC' + A'C'D' + A'BD + AB'C'

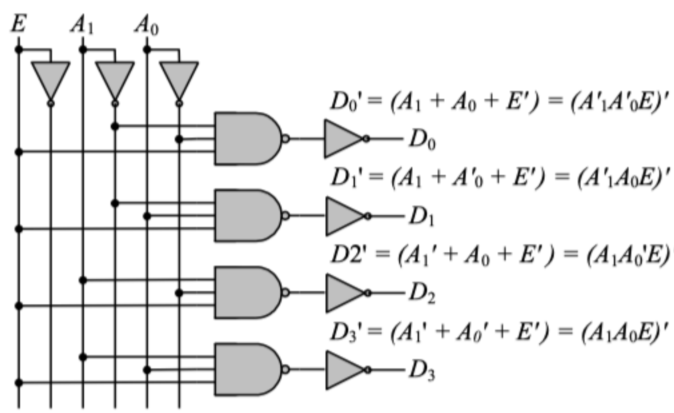


g = A'CD' + A'B'C + A'BC' + AB'C'

11) Draw the logic diagram of a 2-to-4-line decoder using (a) NOR gates only and (b) NAND gates only. Include an enable input.

$D0 = A1'A0' = (A1 + A0)'$ (NOR) $D0' = (A1'A0')'$ (NAND)
$D1 = A1'A0 = (A1 + A0')'$ (NOR) $D1' = (A1'A0)'$ (NAND)
$D2 = A1A0' = (A1' + A0)'$ (NOR) $D2' = (A1A0')'$ (NAND)
$D3 = A1A0 = (A1' + A0)'$ (NOR) $D0' = (A1A0)'$ (NAND)





12) A combinational circuit is specified by the following three Boolean functions: