University of Waterloo
Department of Electrical & Computer Engineering
E&CE 223 Digital Circuits and Systems
Midterm Examination
Feb 12, 2004

Total Time = 90 Minutes, Total Marks = 100

Student Name:			**Student ID:**	
1.	**2.**	**3.**	**4.**	**Total**

Attempt all problems. Show all work. If information appears to be missing make a reasonable assumption, state it, and proceed. Calculators are not needed and are not allowed.

Problem 1 [28 marks]

A) Convert the following numbers from one radix to another radix **[6+4]**:
$(1E.F)_{16}$ to radix 8, and 10.

$(1E.F)_{16} = (0001,1110.1111)_2 = (36.74)_8$

$(1E.F)_{16} = (1*16^1) + (14*16^0) + (15*16^{-1}) = 30 + 15/16 = (30.9375)_{10}$

B) Convert the following number from “excess-3” code to “84-2-1” code. **[8]**
(0111 1001 1011) $_{\text{excess-3}}$ to (………..) $_{84\text{-}2\text{-}1}$

0111: 0111 – 11 = 0100 = 4 (BCD) = 0100 $_{(84\text{-}2\text{-}1)}$
1001: 1001 – 11 = 0110 = 6 (BCD) = 1010 $_{(84\text{-}2\text{-}1)}$
1011: 1011 – 11 = 1000 = 8 (BCD) = 1000 $_{(84\text{-}2\text{-}1)}$

=> (0111 1001 1011) $_{\text{excess-3}}$ = (4 6 8) $_{\text{BCD}}$ = (0100 1010 1000) $_{84\text{-}2\text{-}1}$

C) Assuming that we use 8 bits to represent binary numbers. For given numbers *A* and *B*, perform *(B-A)* operation using 2’s complement arithmetic. Show all steps. **[5+5]**

i) A = $(101001)_2$ and B = $(11001011)_2$

ii) A = $(0000)_2$ and B = $(10111001)_2$

i)

B = $(11001011)_2$

2’s Comp. A = $(11010111)_2$ +

$(10100010)_2$

ii)

B = $(10111001)_2$

2’s Comp. A = $(00000000)_2$ +

$(10111001)_2$

Problem 2 [12 marks]

A) An operator # is defined by $A \# B = A' + B$ [6]
Is this operation commutative and associative? Explain

i) $A' + B \neq B' + A$ => not commutative

ii) $A\#(B\#C) \neq (A\#B) \# C$

$A' + B\#C \neq (A\#B)' + C$
$A'+B'+C \neq (A'+B)'+C$
$A'+B'+C \neq A.B' + C$

=> not associative

B) Given a function, $F = (A+B'+C')'. (A+B') + A'C$ Find its complement in **simplest** "sum of products" form. [6]

$F = (A+B'+C')'. (A+B') + A'C$
$= A'BC.(A + B') + A'C = A'BCA + A'BCB' + A'C = A'C$

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

$F' = [(A+B'+C')'. (A+B') + A'C]'$
$= [(A+B'+C')']' + (A+B')' . (A'C)'$
$= [(A+B'+C') + A'B] . (A + C')$
$= (A + B' + C' + A'B) . (A + C')$
$= (A +A'B + B' + C') . (A + C')$
$= (A +B + B' + C') . (A + C')$
$= (A +1 + C') . (A + C')$
$= 1 . (A + C')$
$= A + C'$

Problem 3 [30 marks]
Simplify the following Boolean function *F*, together with the don't-care conditions *d* using four-variable map method into:

i) Sum of the products **[15]**
ii) Product of sums **[15]**

F (A, B, C, D) = ∑ (0, 7, 8, 12) + *d* (A, B, C, D) = ∑ (2, 3, 4, 10, 13)

i)

C

AB\CD	00	01	11	10
00	1	0	d	d
01	d	0	1	0
11	1	d	0	0
10	1	0	0	d

F = C′D′ + ACD

ii)

C

AB\CD	00	01	11	10
00	1	0	d	d
01	d	0	1	0
11	1	d	0	0
10	1	0	0	d

F′ = CD′ + C′D + AD => F = (C + D′) . (C′ + D) . (A′ + D′)

Problem 4 [30marks]

A transmitter is transmitting a total of three bits D (data), P (even parity bit) and Q (odd parity bit). Assuming only 1 bit may be corrupted during transmission, we are able to detect and correct the erroneous bit.

- i) Draw a truth table for transmitted codes. [5]
- ii) Draw a logic circuit for transmitted parity bits. [5]
- iii) Draw a truth table containing all possible combinations of received bits, Parity Error Check bit (C) and Recovered (corrected) data bit (R). [10]
- iv) Derive the circuits for Parity Error Check bit (C), and Recovered data bit (R) in any "two level" form. [10]

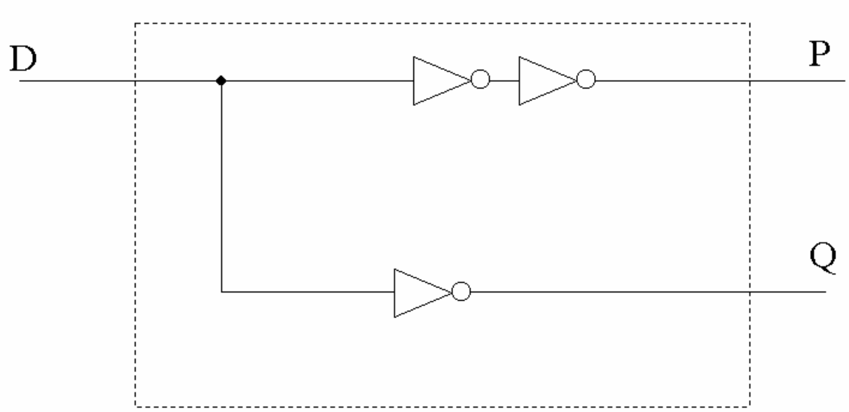
Example: if D is 0, transmitter sends 001 (DPQ). For this transmission, if the receiver receives 101, it's able to detect the error as well as recover data which is 0)

i)

Transmitted Data

D	P	Q
0	0	1
1	1	0

ii) Logic circuit for generating transmitted parity bits



iii)

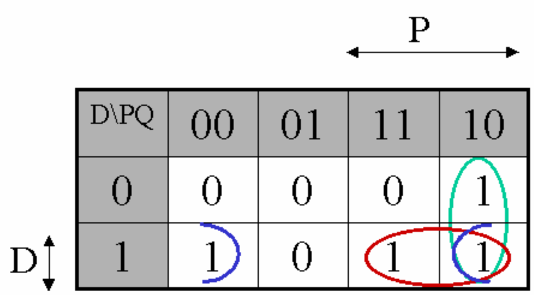
Received Bit at Receiver				Parity Check	Recovered or Corrected Data
D	P	Q		C	R
0	0	0*		0	0
0	0	1		1	0
0*	1	0		0	1
0	1*	1		0	0
1	0*	0		0	1
1*	0	1		0	0
1	1	0		1	1
1	1	1*		0	1

* Corrupted data

C = D′P′Q + DPQ′

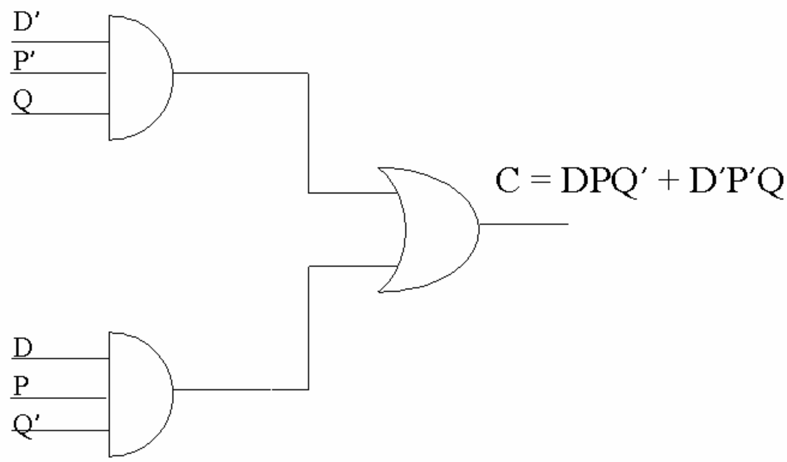
R = D′PQ′ + DP′Q′ + DPQ′ + DPQ

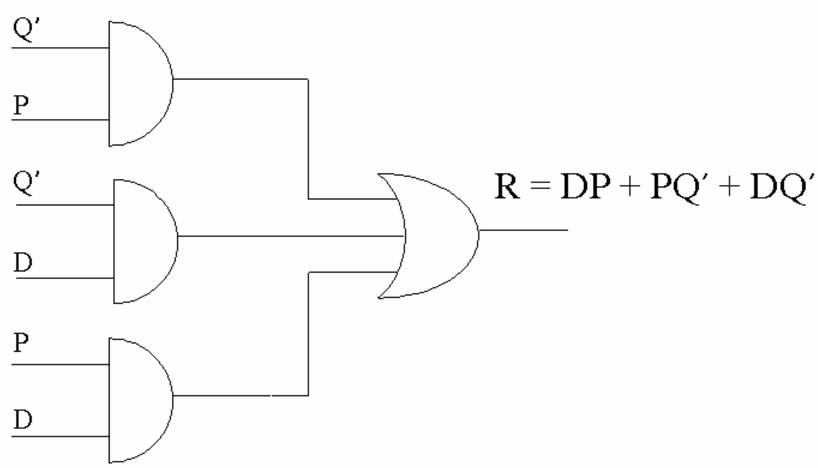
iv)



R = DQ′ + DP + PQ′

C = D′P′Q + DPQ′


D′
P′
Q
D
P
Q′
C = DPQ′ + D′P′Q


Q′
P
Q′
D
P
D
R = DP + PQ′ + DQ′