Name Key - 1 GT#

**ECE 2030h, Intro. To Computer Eng., Final Exam**

**Dec. 12, 2001**
**Prof. John A. Copeland**
**School of Electrical and Computer Engineering**

RULES.

i This quiz is closed book.

ii. Non-programmable calculators may be used.

iii Answer all questions and show all work to receive full credit.

iv All questions have the same weight. (10 Points). All sub-questions within a question are weighted equally.

v Please do not ask the proctors any questions during the exam about exam questions. Part of the test is understanding the question as written, without supplemental information. If you feel additional data is needed to solve the problem, make (and state) an assumption and then work the problem.

## Question 1 – Minterm and Maxterm Indices

For the truth tables below, show the minterm sum of products, and the maxterm product of sums:

| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

sum of products $A'B'C' + A'BC + AB'C' + ABC'$

product of sums $(A+B+C')(A+B'+C)(A'+B+C')(A'+B'+C')$

Name Key - 1

**Question 2 – Karnaugh Map** For the Karnaugh map below, circle the Prime Implicants and label the Essential Prime Implicants with "EPI".

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 0 | 1 | 1 |
| 01 | 1 | 0 | 0 | 1 |
| 11 | 1 | 0 | 0 | 0 |
| 10 | 0 | 0 | 1 | 1 |

EPI A'C'D'

A''B D''

- B'C

EPI

Write the reduced logic expression: $B'C + A''CD'' + BC''D''$ (or $A''BD''$)

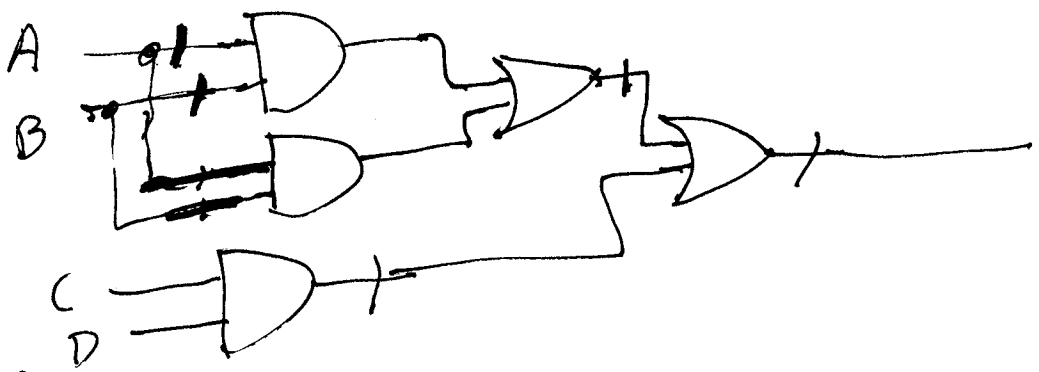
minterm indices (decimal) 2, 3, 4, 6, 10, 11, 12

**Question 3 – Mixed Logic.** Design a logic circuit using NOR gates for the logic function

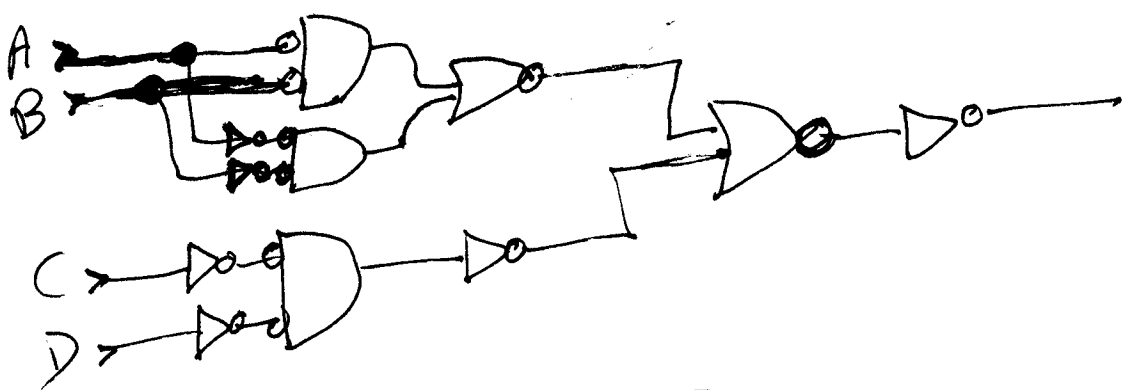
X = (A'B' + A B)' + (CD)'

. using the mixed logic technique. Assume only signals A, B, C, and D are available. A' = NOT A..

**Step 1.**

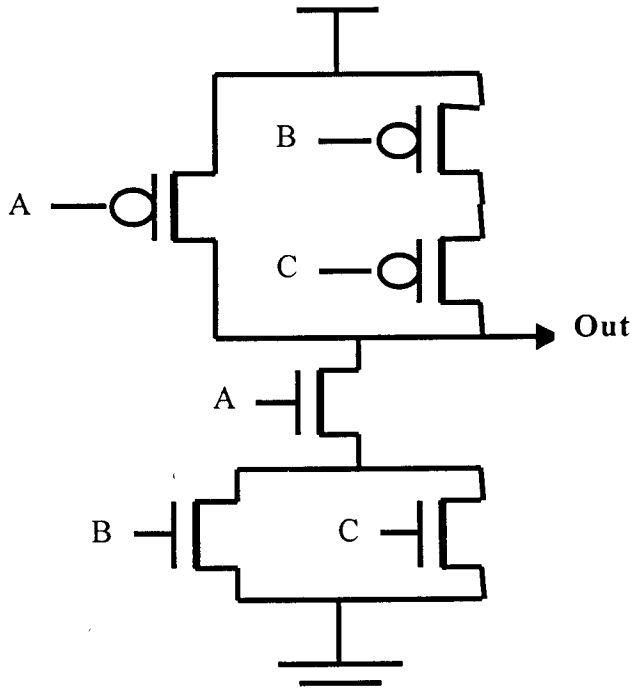



**Step 2.**



Name Key-1

**Question 4 – CMOS Logic Gates.** For the following switch level circuit, complete the truth table computed. If a floating or shorted output is detected, indicate that in the truth table. If no floats or shorts are detected, write the Boolean expression computed by the circuit.




| A | B | C | Out |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

Write the Boolean expression for this function, Out = $\overline{A} + \overline{B}\,\overline{C}$

**Question 5.** Write the missing unsigned integer numbers in binary, hex, and decimal representations.

| Decimal | Hex | Binary |
|---|---|---|
| 318 | 13E | 1 0011 1110 |
| 249 | F9 | 1111 1001 |
| 166 | A6 | 10100110 |

Name Key -1

## Question 6 – Binary Arithmetic in Two's Complement Notation

Do the arithmetic below in two's compliment binary arithmetic (8-bit integers, -128 to +127).

| | | |
|---|---|---|
| +51 | 0011 0011 | $83 = 0101 0011 |
| +(-83) | 1010 1101 | $-83 = 1010 1101 |
| = -32 | 1110 0000 | |

**Question 7.** Our MIPS architecture has only "Branch on Equal" (BEQ $X, $Y) and "Branch on Not Equal" (BNE $X, $Y) commands. Show how to use the "Set on Less Than" (SLT $1, $X, $Y) and BEQ or BNE to make as two-command equivalent of the following (note: $0 always = 0 ):

BLE $2, $3

```
SLT $1, $3, $2
BEQ $1, $0, xxx
```

$1 = 1 if $3 < $2
$2 ≥ $3
$1 = 0 if $3 ≥ $2

BGE $2, $3

```
SLT $1, $2, $3
BEQ $1, $0, xxx
```

BLT $2, $3

```
SLT $1, $2, $3
BNE $1, $0, xxx
```

set if $2 < $3

BGT $2, $3

```
SLT $1, $3, $2
BNE $1, $0, xxx
```

set if $3 < $2

Show how to make a branch comand that always jumps ahead to the label xxx.

```
B   BEQ $0, $0, xxx
    ADD $4, $3, $5
xxx: ADDI $4, $4, 56
```

**Question 8.** Branch and Jump Ranges

Branching is relative to the present "next instruction."

a. What is the range of a BNE or BEQ instruction? −128 K to +128 K ±($2^{17}$)

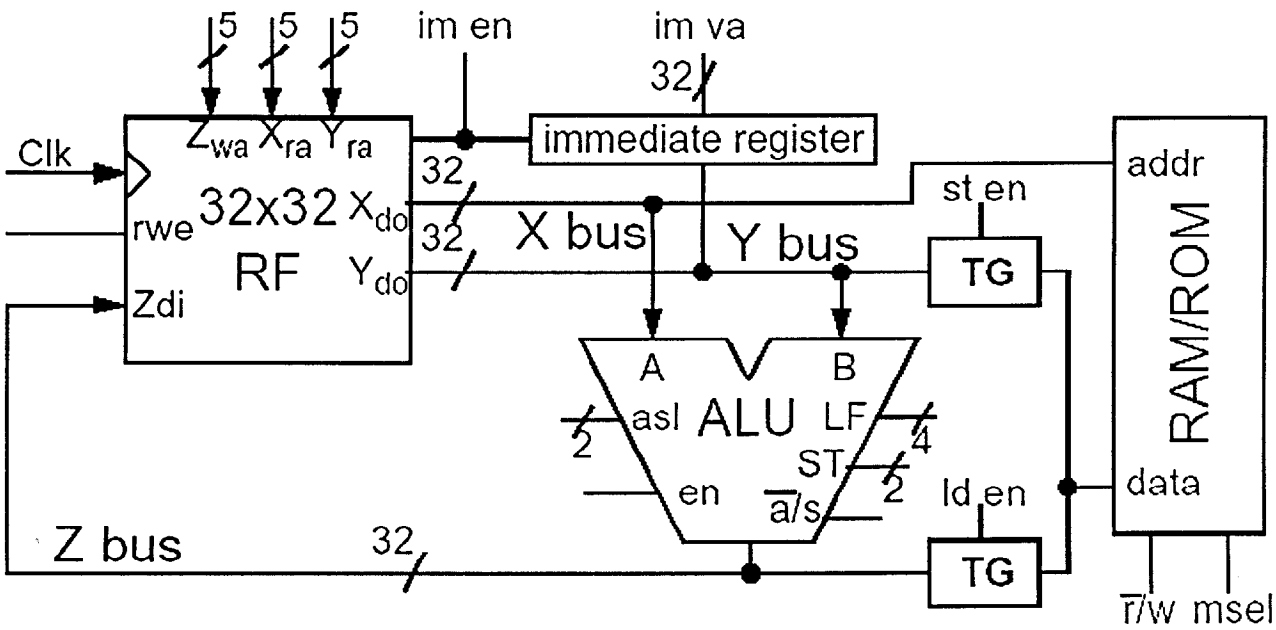
b. What is the range of a "jr" jump instruction? 0 to 4G

c. What is the range of a "j" jump instruction? 0 to 256 M ($2^{28}$)

note: you can use G to mean 2^30, (~ 1 billion), M to mean 2^20 (~ 1 million), K to mean 2^10 (1024)

Name Key-1

**Question 9** – CPU Control Lines




Show how the control lines are set to achieve the operations below. Assume msel = 1;

asl: 0=AU, 1=LU, 2=SU, 3 = invalid; ST: 0=arithmetic, 1=logical, 2=rotate, 3 = invalid;
LF: 0=AND, 1=OR, 2=XOR, 3 = invalid; a'/s: 0=add, 1= subtract rwe: 1=write, 0= do not write

Add $5 to the value 82 and put the result into memory (address in $9) M[9] = $5 + 82
(note: loading or storing data requires a separate CPU clock cycle)

| Mem r'/w | X | Y | Z | rwe | asl | a'/s | en | ld en | st en | im en | im va (Immediate Value) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 5 | X | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 82 |
| 1 | 9 | 1 | X | 0 | X | X | 0 | 0 | 0 | 0 | X |

same. any # but 9 (or 0).

**Question 10.** When is a "jal" jump used? What value is put into register $ra? Why?

For a jump to a subroutine
PC + 4 → $ra
New PC value when subroutine
is done (jr, $ra).