Name: ____

CSE 370, Autumn, 2007, Exam 1

Please do not turn the page until instructed to do so.

Rules:

- Please remove everything from your desk area except one sheet of notes and whatever pens/pencils you want to use.
- Please stop working promptly at 11:20.
- If you rip the pages apart, please staple them back together when you are done.

Advice:

- The exam should have 11 pages; check before you start.
- Read questions carefully before you start writing.
- Write down partial solutions for partial credit.
- There are 120 points on the exam distributed unevenly; try to distribute your effort roughly according to point value.
- The questions are not necessarily ordered according to difficulty. Skip around to find parts that are easy for you.
- If you have questions, ask.
- The last two exercises are "challenge exercises". Do not work on them unless you are 100% sure you are done with the rest of the exam

Grading Summary	
1:	/ 20
2:	/ 20
3:	/ 20
4:	/ 10
5:	/ 15
6:	/ 20
7:	/ 15
Total:	/ 120

1. Arithmetic and binary numbers (20 points)

Perform the following base conversions. (8 points)

1101_2	af_{16}	25_{10}	29_{10}
to base 10	to base 10	to base 16	to base 2

Perform the following arithmetic exercises in the indicated bases. (8 points)

10101_2	$f7_{16}$	1011_2	10101_2
$+\ 11101_2$	$-\ 1ab_{16}$	$\times\ 1101_2$	$\div\ 101_2$

Translate -27 to 8-bit one's complement and 8-bit two's complement. (4 points)

2. Boolean Logic Proofs (20 points)

Prove the following theorem with the laws of Boolean algebra.

$(X + Z)(\neg X + Y) = XY + \neg XZ$

You may freely use any of the following basic laws:

1. $X + 0 = X$	1D. $X \cdot 1 = X$
2. $X + 1 = 1$	2D. $X \cdot 0 = 0$
3. $X + X = X$	3D. $X \cdot X = X$
4. $\neg(\neg X) = X$	
5. $X + \neg X = 1$	5D. $X \cdot \neg X = 0$
6. $X + Y = Y + X$	6D. $X \cdot Y = Y \cdot X$
7. $(X+Y)+Z = X+(Y+Z) = X+Y+Z$	7D. $(X \cdot Y) \cdot Z = X \cdot (Y \cdot Z) = X \cdot Y \cdot Z$
8. $X \cdot (Y+Z) = (X \cdot Y) + (X \cdot Z)$	8D. $X+(Y \cdot Z) = (X+Y) \cdot (X+Z)$

3. Truth Tables (20 points)

Use the truth table method to prove one form of DeMorgan's law for 3 variables:
$\neg A + \neg B + \neg C = \neg(ABC)$
Show at least two "intermediate" columns. (10 points)

There is a strange two player game played on a remote island. Player A holds up a card with either 4, 5 or 6 written on it. Player B holds up a card with 4, 5, 6 or 7 written on it. If the card that player B holds up has the same value or is one greater than the card that player A holds up, then player B wins. First, devise an encoding so that we can represent the cards held up by both players in one round of the card game with just four Boolean variables. Next, draw a truth table for the function that determines when player B wins. (10 points)

4. Minterms and Maxterms (10 points)

Draw K-maps for the following two functions. DO NOT go on to implement the functions.

$F(A,B,C,D) = \quad \Pi M(0,1,5,6,10,14)\Pi D(9,15)$

$G(A,B,C,D) = \quad \Sigma m(0,4,7,8,11,15) + \Sigma d(1,9,12)$

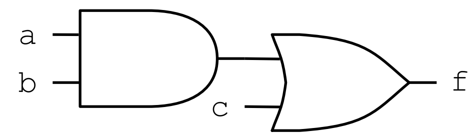
5. SoP and PoS (really just PoS) (15 points)

Draw the circuit for a minimum product-of-sums implementation of the function represented by the following K-map. Use the standard level 1 OR gates, level 2 AND gate style. You may assume both true and inverted versions of all the input variables are available.

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

6. Verilog dos and don'ts (20 points)

One of the following four Verilog modules does not correctly implement this circuit. Indicate which one it is and give a <u>very brief</u> reason why (one or two sentences). (8 points)




```
module x1(a,b,c,f);
  input a, b, c;
  output f;
  reg f, z;
  always @ (a or b or c)
  begin
    z = a & b;
    f = z | c;
  end
endmodule
```

```
module x2(a,b,c,f);
  input a, b, c;
  output f;
  wire z;
  or o1(f, z, c);
  and a1(z, a, b);
endmodule
```

```
module x3(a,b,c,f);
  input a, b, c;
  output f;
  wire z;
  assign f = z | c;
  assign z = a & b;
endmodule
```

```
module x4(a,b,c,f);
  input a, b, c;
  output f;
  reg f, z;
  always @ (a or b or c)
  begin
    f = z | c;
    z = a & b;
  end
endmodule
```

There are three substantial errors in the following Verilog code. Errors include all mistakes that would cause this code to be rejected by a simulator entirely, or that violate any of the golden rules for how to write combinational logic in Verilog. Point out each error in the code, and give a <u>very brief</u> description of it. Do not include simple style points (like "bad" is a poor module name choice). (12 points)

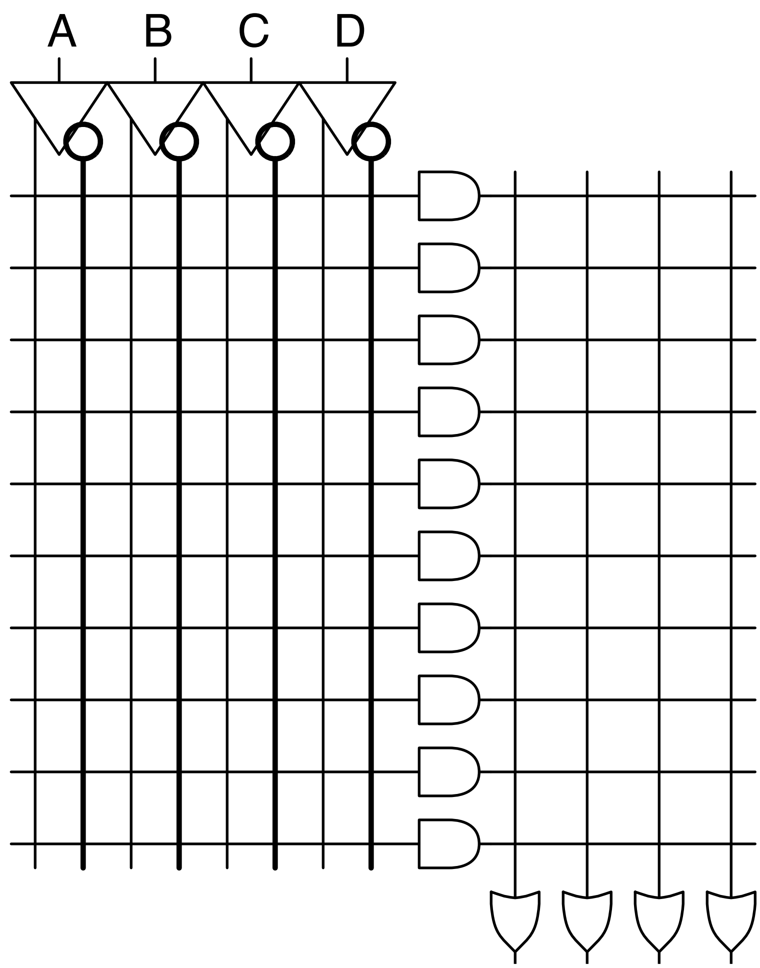
```
module bad(w, x, y, z, a, b);
  input w, x, y, z;
  output a, b;
  reg a;
  always @ (w or x)
  begin
    if (w) begin
      a = z | y;
      b = x;
    end
    else begin
      b = x & z;
    end
  end
endmodule
```

7. Everyone's PAL, the PLA (15 points)

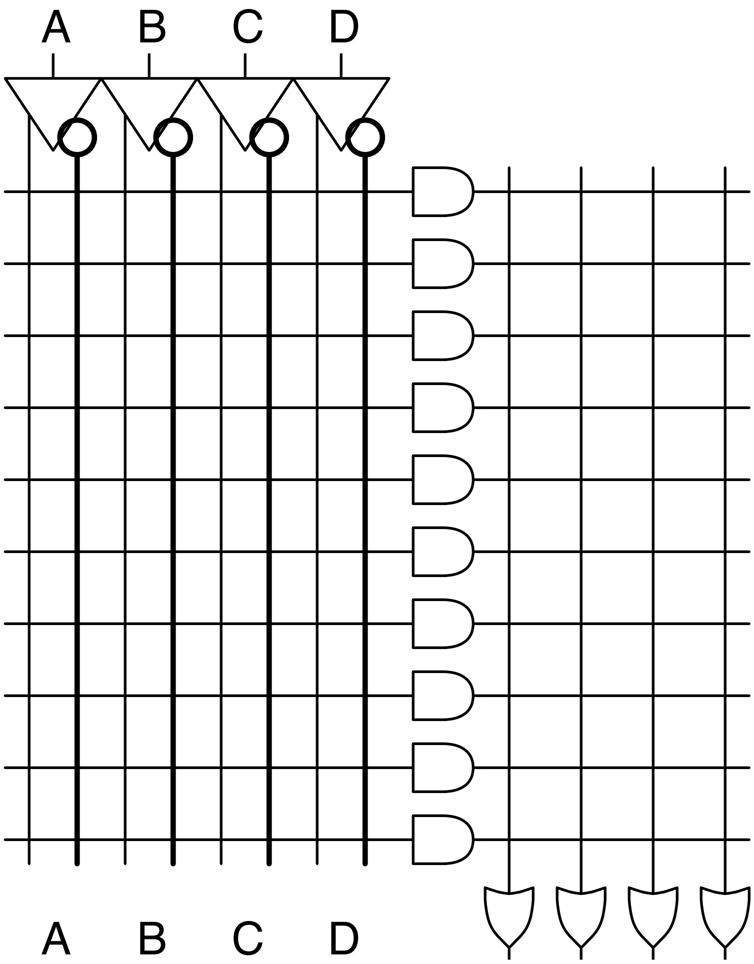
Implement the following 2 functions (F on the left and G on the right) on the PLA provided. Use as few product terms (AND gates) as you can.

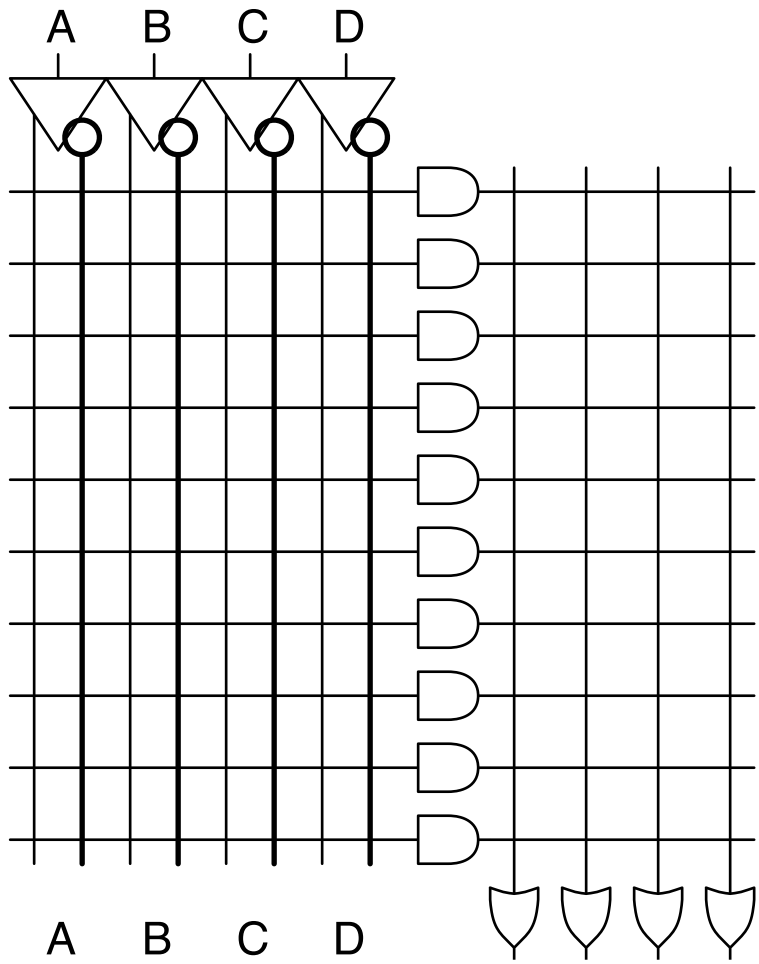
CD \ AB	00	01	11	10
00	0	0	1	0
01	1	x	x	0
11	x	1	1	x
10	0	x	0	0

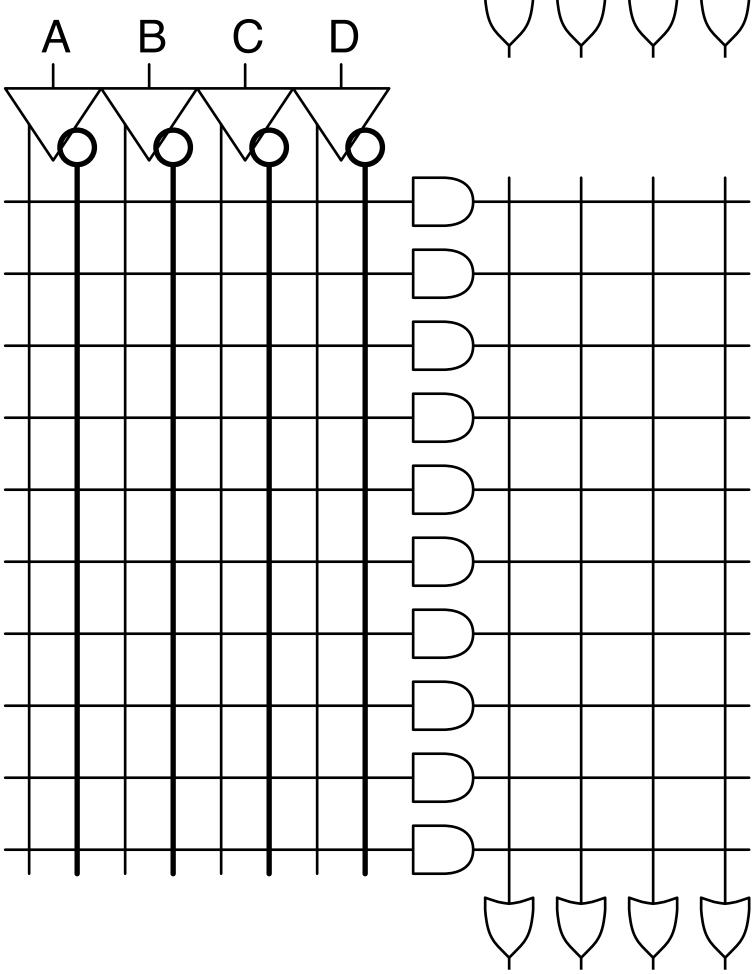
CD \ AB	00	01	11	10
00	0	0	x	0
01	x	1	1	x
11	0	x	x	1
10	0	1	0	0

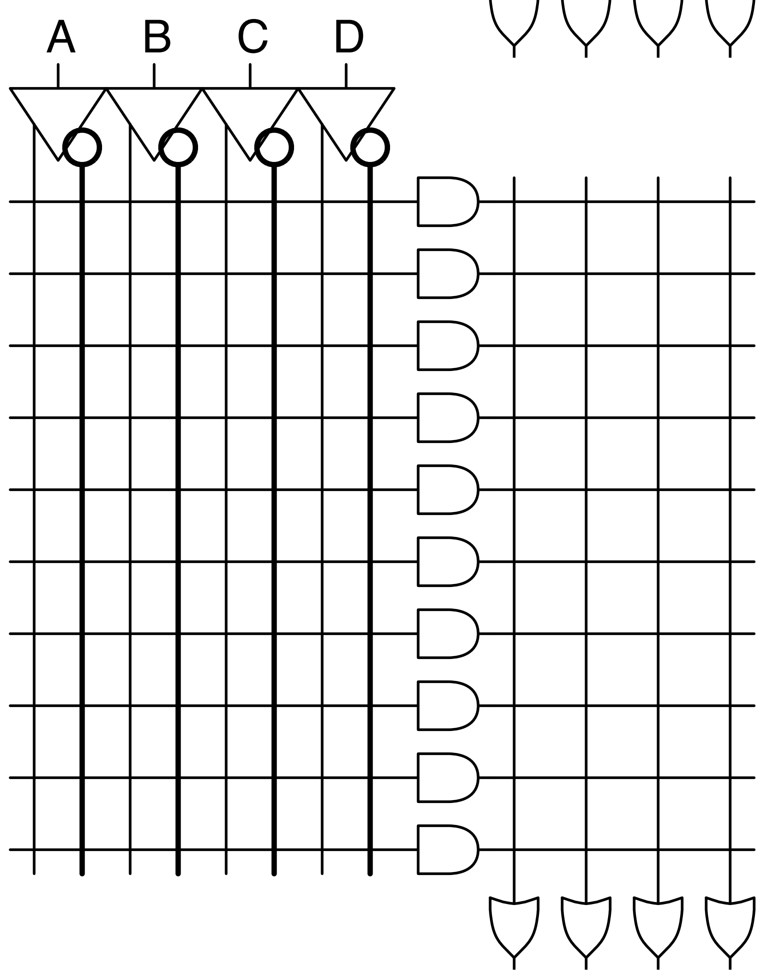


7. Scratch PLAs









Challenge 1: How Many Functions?

We say that a Boolean function F(A,B,C,…) is *self-dual*, if F(A,B,C,…) = ¬F(¬A,¬B,¬C,…). How many self-dual functions are there of N inputs?

Challenge 2 (quite hard): Logic Gymnastics

Using only the first eight theorems of Boolean algebra (and any lemmas you can prove with them), pick any form of DeMorgan's law and prove that it is true.
Hint: You may find the following lemma useful (you still have to prove it, if you want to use it).
if $A \cdot \neg B = 0$ and $A + \neg B = 1$, then $A = B$