*SPECIAL MAKEUP - FINAL EXAMINATION*

# COMPUTER ARCHITECTURE AND DIGITAL DESIGN
# 03-60-265-01

**SCHOOL OF COMPUTER SCIENCE - UNIVERSITY OF WINDSOR**
*Fall 2008*

**Last Name:** .....

**First Name:** .....

**Student ID:** .....

## PLEASE READ CAREFULLY BEFORE YOU START

1. This is a CLOSED book test; no notes, textbooks, calculators or computer aids are allowed.
2. PRINT your name legibly and clearly with your Student ID in the space indicated above.
3. You will be asked to sign your name, once during the exam (sign-in) and once before leaving the exam room (sign-out).
4. Answer all the questions in the space provided. DO NOT REMOVE any pages or attach any papers to this test or you will void your test and receive a mark of zero. If you need more space please use the reverse side of any page, or request an additional exam booklet which MUST be returned with this exam paper with your name and ID clearly written on it.
5. You are not allowed to give or receive unauthorized help with your test. Any misconduct, as outlined by the Senate bylaw 31 article I, will be reported accordingly.
6. You have 3 hours to complete this test.
7. Final examinations papers are not returned to students.

*Good Luck!*

**I AGREE TO THE ABOVE TERMS AND WILL NEITHER RECEIVE NOR GIVE UNAUTHORIZED HELP ON THIS EXAM**

..... .....

**SIGNATURE** **DATE**

**TOTAL MARK (MAX 54): _____**

**Question 1. [ 10 marks ]**
Answer all parts of this question.

A. Assume that X is a Boolean variable. Simplify the following Boolean expression. [1 mark]

**X + 0 = X**

B. Assume that X and Y are Boolean variables. Simplify the following Boolean expression. [1 mark]

**X ( Y + 1 ) = X**

C. Assume that X, Y and Z are Boolean variables. Simplify the following Boolean expression. [2 marks]

**X Z + Y ( X′ Z + Y′ ) = XZ + YX′Z + YY′ = XZ + X′YZ**

**= (XZ+X′)(XZ+Y)(XZ+Z)**

**= (X+Y)Z**

D. Assume that W, X, Y and Z are Boolean variables. Simplify the following expression for F(W,X,Y,Z) using only Boolean algebra and axioms (the axioms are provided on the last page of the examination paper) You **must** show your full derivation to get full marks. [2 marks]

**F(W,X,Y,Z) = WXYZ + WXYZ′ + WXY′Z + WXY′Z′ + W′XYZ′**

**+ WX′YZ′ + W′X′YZ′ + W′XY′Z**

**= WXYZ + WXYZ′ + WXY′Z + WXY′Z′ + W′XYZ′ + WX′YZ′**
**+ W′X′YZ′ + W′XY′Z**

**= WX(Y+Y′)(Z+Z′) + (W+W′)(X+X′)YZ′ + (W+W′)XY′Z**

**= WX + YZ′ + XY′Z**

**Question 1 – continued**

E. Assuming the following Truth Table for a three input variable function F(A,B,C), determine the full Minterm expression for F. [ 2 marks ]

| 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 |

**Answer:** `F(A,B,C) = A'B'C' + A'BC + AB'C' + ABC'`

F. Using a Karnaugh map, simplify the expression for F(A,B,C) defined in Part E above. [ 2 marks ]

| K-map | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| A 0 | 1 | | 1 | |
| 1 | 1 | | | 1 |

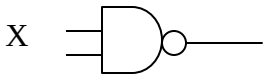
**Answer:** `F(A,B,C) = A'BC + B'C' + AC'`

**Question 2. [ 8 marks ]**
Answer all parts of this question.

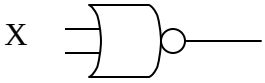
A. Draw a logic circuit for an Inverter (ie. equivalent to the complementer gate) using only a NAND gate. [1 mark]

X' = (XX)'




B. Draw a logic circuit for an Inverter (ie. equivalent to the complementer gate) using only a NOR gate. [1 mark]

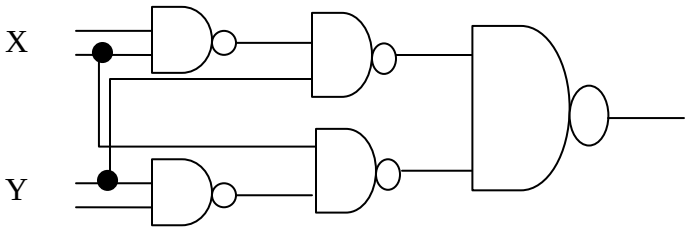
X' = (X+X)'




C. Draw the circuit diagram for the Boolean expression below. Use NAND gates only. (ie. you may not use inverter gates). [2 marks]

**F(X,Y) = XY' + X'Y**

**F = F'' = (XY' + X'Y)'' = ((X(YY)')'((XX)'Y)')'**




D. Express the base-10 value $\mathbf{185}_{10}$ as an 8-bit unsigned binary value. [2 marks]

**185 = 128 + 32 + 16 + 8 + 1 = 10111001**

E. Express the base-16 value $\mathbf{1A2}_{16}$ in **base-7** representation. [2 marks]

**1A2 = 256 + 160 + 2 = 418**

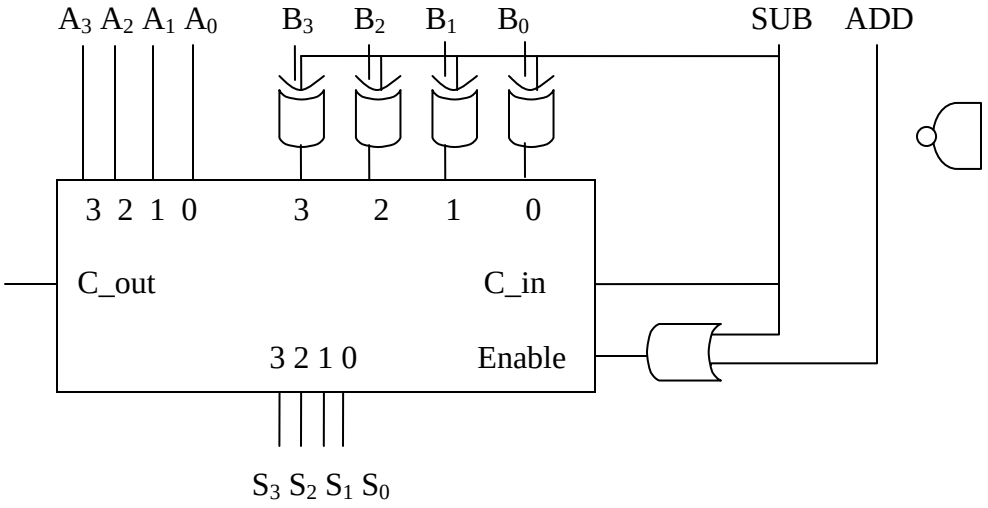
**418/7 = 59 + 5 ; 59/7 = 8 + 3 ; 8/7 = 1 + 1**

**Question 3. [ 10 marks ]**
Answer both parts of this question.

A. A 4-bit adder unit consists of two sets of 4-bits representing inputs A and B, and also an input Carry_In bit. It produces the sum S as a 4-bit output, and also an output Carry_Out bit.

Using the 4-bit adder as an MSI component (ie. do not define the internal logic) expressed as a block diagram, draw a complete circuit for a 4-bit Adder/Subtracter unit showing all necessary control logic on the inputs, and also including control inputs ADD and SUB. When ADD and SUB are both 0, the Adder/Subtracter circuit is disabled. When ADD=1 and SUB=0, the output S = A+B. When ADD=0 and SUB=1, the output S = A-B. Assume that ADD and SUB are mutually exclusive (ie. they are never both 1 at the same time). [6 marks]




B. Using any, or all, input and output bits specified in Part A above, draw the circuit for an output overflow bit V, where V has value 1 if S=A+B overflows, or if S=A-B overflows, depending on whether addition or subtraction is selected. [4 marks]

NOTE: Overflow occurs for ADD if the signs of the two operands are the same and this sign differs from the sign of S. For SUB, overflow occurs if A and B have different signs and the sign of S (the difference) is the different than the sign of A. For example, if A=7 and B=-1, then A-B=8=-8! (overflow). Alternatively, if A=-8 and B=1, then A-B=-9=7! (overflow).

Hence, **V = A3.B3.S3'.ADD + A3.B3'.S3.SUB**

**Markers will watch out for equivalent solutions, stated differently.**

**Question 4. [ 10 marks ]**

Answer all parts of this question based on Mano's simplified computer architecture and instruction set. The following mnemonics are used below: PC – program counter; AR – address register; AC – accumulator; IR – instruction register; DR – data bus register; SC – sequence counter register; M[X] – data at RAM (M) location X (either AR, or a value).

A. The instruction LDA performs loading of the AC register from a memory location whose address is located in the low order 12 bits of the 16-bit instruction. If the address is $100_{16}$, the complete machine (binary) code, in hexadecimal, is 2100. Assume that all RAM-to-CPU bus transfers are made using the DR register first, and that loading of the AC register from DR must be performed in a separate time step.

Using Register Transfer Language statements, state the complete logic necessary to perform the full instruction cycle for the LDA instruction. Use T0, T1, etc. to denote the time step values and for each time step state the RTL logic. It is not necessary to state additional condition logic – the time steps are sufficient. [5 marks]

| Time Step | Micro-operation(s) |
|---|---|
| T0 | AR = PC, SC++ |
| T1 | IR = M[AR] , PC = PC + 1, SC++ |
| T2 | I = IR(15) , OpCode = IR(12..14) , AR = IR(0..11) , SC++ |
| T3 | DR = M[AR] , SC++ |
| T4 | AC = DR, SC = 0 |

B. Based on Mano's simplified computer architecture and instruction set, the instruction ADD performs 2's complement addition of a value in memory to the AC register. The indirect addressing form of the operation code is **9**. For instance, if the machine code for the instruction is **9100**$_{16}$, and the 16-bit value **0200**$_{16}$ is stored at location **100**$_{16}$, and the 16-bit value **0F3A**$_{16}$ is stored at location **0200**$_{16}$, then the value **0F3A**$_{16}$ is added to the AC register.

Using Register Transfer Language statements, state the complete logic necessary to perform the full instruction cycle for the ADD (Indirect) instruction. Use T0, T1, etc. to denote the time step values and for each time step state the RTL logic. It is not necessary to state additional condition logic – the time steps are sufficient. [5 marks]

| Time Step | Micro-operation(s) |
|---|---|
| T0 | AR = PC, SC++ |
| T1 | IR = M[AR] , PC = PC + 1, SC++ |
| T2 | I = IR(15), OpCode = IR(12..14), AR = IR(0..11), SC++ |
| T3 | DR = M[AR] , SC++ |
| T4 | AR = DR(0..11) , SC++ |
| T5 | DR = M[AR] , SC++ |
| T6 | AC = AC + DR, SC = 0 |

**Question 5. [ 10 marks ]**
Answer all parts of this question.

A. Below is provided a sequence of memory address locations, each containing 16 bits of data, expressed as a hexadecimal value (the corresponding operation mnemonic is provided in parentheses). At a certain time when the SC (sequence counter) register is 0, the value stored in the PC register is $\mathbf{100}_{16}$, the AC register contains 0 and the E register contains 0. Immediately following the execution of **each** executable instruction, state the values stored in the PC, AR and AC registers. Stop if a HALT (7001) instruction is executed (do not state values after this instruction). [5 marks]

| Address | Content (Stored value) | PC | AR | AC |
|---|---|---|---|---|
| 100 | 2109 (LDA) | 101 | 109 | 1F06 |
| 101 | 010A (AND) | 102 | 10A | 1104 |
| 102 | 7200 (CMA) | 103 | 102 | EEFB |
| 103 | B10B (STA) | 104 | 108 | EEFB |
| 104 | 7008 (SNA) | 106 | 104 | EEFB |
| 105 | 7100 (CME) | | | |
| 106 | 7080 (CIR) | 107 | 106 | 777D |
| 107 | 7001 (HLT) | | | |
| 108 | 210A | | | |
| 109 | 1F06 | | | |
| 10A | 11E4 | | | |
| 10B | 0108 | | | |

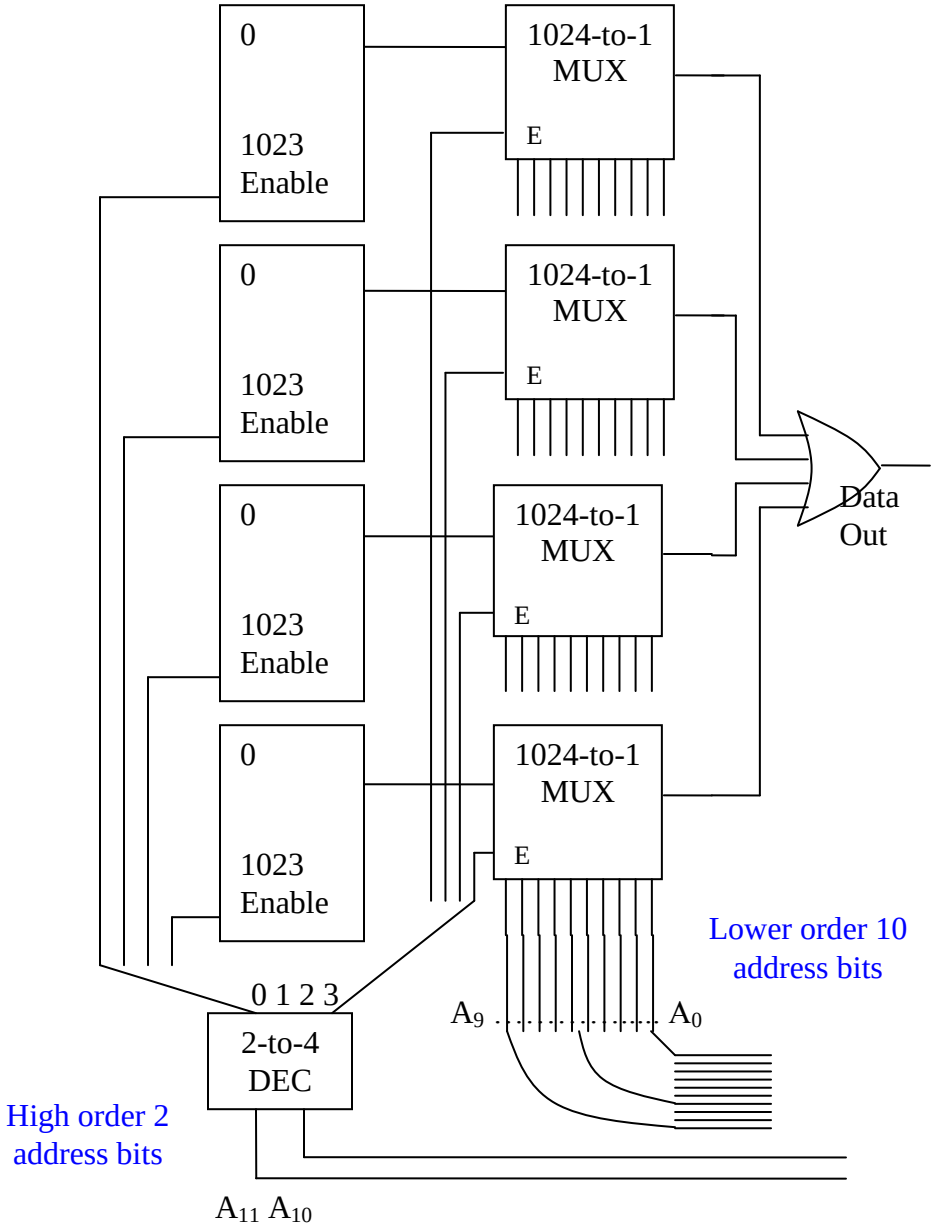
B. Below is provided a sequence of memory address locations, each containing 16 bits of data, expressed as a hexadecimal value (the corresponding operation mnemonic is provided in parentheses). At a certain time when the SC (sequence counter) register is 0, the value stored in the PC register is $\mathbf{100}_{16}$, the AC register contains 0 and the E register contains 0. Following the execution of each executable instruction, state the values stored in the PC, AR and AC registers. Stop if a HALT (7001) instruction is executed (do not state values after this instruction). [5 marks]

| Address | Content (Stored value) | PC | AR | AC |
|---|---|---|---|---|
| 100 | A108 (LDA) | 101 | 108 | F10A |
| 101 | 7020 (INC) | 102 | 101 | F10B |
| 102 | B10B (STA) | 103 | 108 | F10B |
| 103 | 6109 (ISZ) | 105 | 109 | 0000 |
| 104 | 210B (LDA) | | | |
| 105 | 7020 (INC) | 106 | 105 | 0001 |
| 106 | B10B (STA) | 107 | 108 | 0001 |
| 107 | 7001 (HLT) | | | |
| 108 | F10A | | | |
| 109 | FFFF | | | |
| 10A | 0002 | | | |
| 10B | 0108 | | | |

**Question 6. [ 6 marks ]**

Design a memory (RAM) unit of 4096 address locations, each location specifying 16 bits of storage. The unit is to be constructed using basic RAM components, each consisting of 1024 address locations, each location specifying 16 bits of storage. Additional decoders, multiplexers and other logic gates or components may be required and, if used, must be completely labeled. The complete RAM circuit, once it is enabled, must be capable of supporting access at a specified address (stored in the AR register) and delivery of the data stored at that address M[AR] to the CPU DR register. Draw the complete circuit using block diagrams.



**This page contains various definitions and information provided freely for each student to use for the examination, if required. You may detach this page.**

**Operation Codes and Mnemonics for Mano's machine:**

***Memory Access Opcodes (4 bits) – Direct (Indirect)***

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 (8) | AND | 1 (9) | ADD | 2 (A) | LDA | 3 (B) | STA |
| 4 (C) | BUN | 5 (D) | BSA | 6 (E) | ISZ | | |

***CPU based Opcodes (16 bits)***

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7800 | CLA | 7400 | CLE | 7200 | CMA | 7100 | CME |
| 7080 | CIR | 7040 | CIL | 7020 | INC | 7010 | SPA |
| 7008 | SNA | 7004 | SZA | 7002 | SZE | 7001 | HLT |

***I/O based Opcodes (16 bits)***

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| F800 | INP | F400 | OUT | F200 | SKI | F100 | SKO |
| F080 | ION | F040 | IOF | | | | |

**Boolean Postulates:**

**P0:** Existence: There exist at least two elements x,y in B such that $x \neq y$

**P1:** Closure: For every x,y in B there exist two combinational operators + and . where x+y is in B and x.y is in B

**P2:** Identity: There exist identity elements 0,1 in B relative to the operations + and ., such that for every x in B: 0+x = x+0 = x and 1.x = x.1 = x.

**P3:** Commutativity: The operations + and . are commutative for all x,y in B: x+y = y+x and x.y = y.x

**P4:** Distributivity: Each operation + and . is distributive over the other; that is, for all x,y,z in B: x.(y+z) = x.y+x.Z and x+(y.z) = (x+y).(x+z)

**P5:** Complementation: For every element x in B there exists an element ~x, called the complement of x, satisfying: x+~x = 1 and x.~x = 0

**Fundamental Logic Micro-operations:**

| | | | |
|---|---|---|---|
| F = 0 | F = AB | F = AB′ | F = A |
| F = A′B | F = B | F = AB′+A′B | F = A+B |
| F = A′B′ | F = AB+A′B′ | F = B′ | F = A+B′ |
| F = A′ | F = A′+B | F = A′+B′ | F = 1 |