Université d'Ottawa
Faculté de génie

École d'ingénierie et de
technologie de l'information

University of Ottawa
Faculty of Engineering

School of Information
Technology and Engineering

uOttawa

L'Université canadienne
Canada's university

COURSE: CEG2136/CEG2536
Computer Architecture I
Architecture des ordinateurs I
SEMESTER: Fall 2008

FINAL EXAMINATION

Q1.1 For 1 K memory locations you need:
- (a) 8 address lines
- (b) 10 address lines
- (c) 12 address lines
- (d) None of the above

Q1.2 Which CPU register provides the address from which the next instruction opcode is to be fetched?
- (a) Instruction register IR
- (b) Accumulator AC
- (c) Program counter PC
- (d) None of these

Q1.3 What is the difference between a direct and an indirect address instruction?

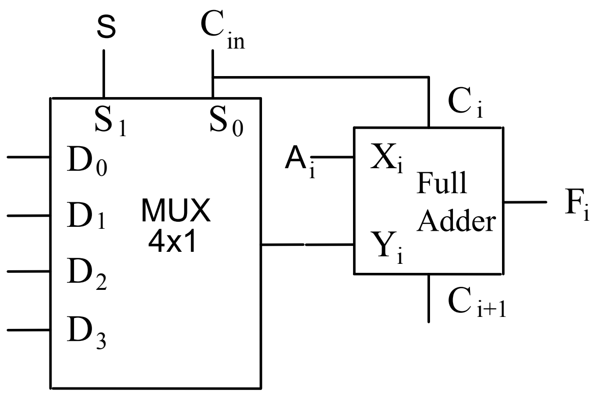
How many references to memory are needed for each type of instruction to bring an operand into a processor register, given that the instruction was already fetched into IR?

direct addressing 1
indirect addressing 2

Q1.4 An arithmetic unit performs basic operations on two numbers A and B, under the control of two bits S and C_{in}., as shown in the following table:

S	$C_{in} = 0$	$C_{in} = 1$
0	F = A + B (addition)	F = A + 1 (increment A)
1	F = A - 1 (decrement A)	F = A + B'+ 1 (subtraction)

The following combinational circuit (multiplexor and full adder) is used to implement the functions described above for bit *i*.
Select from the following combinations of multiplexor inputs which set of logic values implements correctly the above functions



(a) $D_0=B_i'$, $D_1=0$, $D_2=1$, $D_3=B_i$

(b) $D_0=0$, $D_1=B_i$, $D_2=B_i'$, $D_3=1$

(c) $D_0=1$, $D_1=B_i'$, $D_2=B_i$, $D_3=0$

(d) $D_0=B_i$, $D_1= 0$, $D_2=1$, $D_3=B_i'$

(e) $D_0=1$, $D_1=B_i$, $D_2=B_i'$, $D_3=0$

Q1.5.

The datapath of a circuit for the multiplication of two *n*-bit unsigned numbers is formed of:

- An n-bit register P ($P_{n-1} \ldots P_0$), initially set at 0.
- Two n-bit registers for multiplier Y ($Y_{n-1} \ldots Y_0$) and multiplicand X ($X_{n-1} \ldots X_0$)
- A 1-bit register C for carry, initially set to 0.
- An index initially set to *n*.

An algorithm for the multiplication of the two n-bit unsigned numbers is defined as follows:

While Index ≠ 0

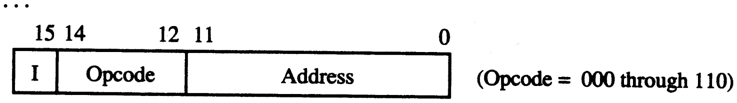
- If $Y_0 = 1$ then P = P + X
- Logic right shift of [C || P || Y].
- Index = Index - 1

Continue (While)

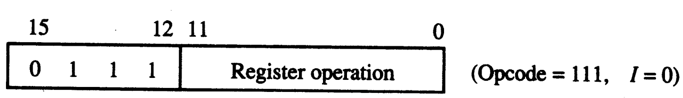
The result is provided in [P || Y].
Derive the ASM chart which describes this algorithm.

Q2

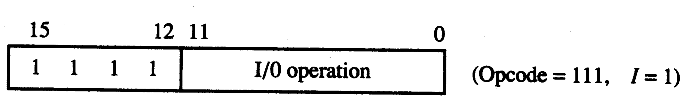
The block diagram of the basic computer that was introduced in chapter 5 of Mano's textbook is given in the annex, along with its instruction list. The instruction word is 16 bit long and has the following structure …



(a) Memory – reference instruction



(b) Register – reference instruction



(c) Input – output instruction

Q2.1 At some point, the content of **PC** of the basic computer is **3AF** (all numbers are in hexadecimal) and the content of **AC** is **2EC3**, as shown in the following table. The content of memory is partially given below, as well:

MEMORY

Address	Memory content
3AD	03B5
3AE	ABBA
3AF	93AD
3B0	DEED
3B1	7BEE
3B2	AD08
3B3	10BC
3B4	1CAA
3B5	3B9F
3B6	3BA0

BASIC COMPUTER REGISTERS

	Content before instruction execution	Content after instruction execution
PC	**3AF**	
AC	**2EC3**	
AR	0000	
DR	0000	
IR	0000	
E	0	
I	0	
SC		

a. What is the instruction that will be fetched and executed?

b. Show the operands and the binary operation that will be performed in the *AC* when the instruction is executed.

c. Fill out the last column ("Content after instruction execution") of the above table with the contents of registers PC, AR, DR, AC, and IR in hexadecimal and the values of E, I and the sequence counter SC in binary, all shown at the end of the instruction cycle.

Q2.2 What is the result, in decimal, of the operation performed by the following assembly program?

	ORG 100	
	LDA OP	
	CMA	
	INC	
	ADD OP1	
	STA OP2	
	HLT	
OP1,	0039	
OP,	FFA5	
OP2,	0	
	END	

The machine code of this program is stored in a memory of 1 kilo word of 16 bits implemented on an Altera FPGA; give the Quartus .mif file that describes this program

Quartus .mif file:

Addr	+0	+1	+2	+3	+4	+5	+6	+7
100	2107							
108	0000							
110								

Q3 Write a subroutine to subtract two numbers that are stored in memory at 2 consecutive addresses. The address of the minuend is passed to the subroutine through the accumulator AC, while the resulted difference is passed back to the calling program through AC, as well.

Q3.1. Write the subroutine in assembly language using the instruction list of the basic computer. The starting address of the subroutine is HEX 100.

Q3.2. Write an example in assembly language for a main program that calls this subroutine. This program is stored in the memory starting with address 0.

Q4.1 Examining the annexed block diagram of the Mano's basic computer, give the list of the blocks of the datapath and the list of those that form the computer's control unit.

DATAPATH: …. CONTROL UNIT: …..

Q4.2 You have to expand the instruction list of your textbook basic computer with three more instructions (ASR, DIV, TAT) as described in the last rows of the annexed instruction list **(see the ANNEX on page 7);** both circuits of the datapath and the control unit need to be extended. This basic computer is a simplified version of the one presented in the textbook as it does not have any provisions to deal with interrupts.

The datapath has to be developed to allow your DR register to support the following operations:

- $LD_{DR} = x\ T_4$: DR <- (bus) /Transfer value from bus to DR
- $INR_{DR} = y\ T_3$: DR <- DR+1 /Increment DR
- $CLR_{DR} = z\ T_3$: DR <- 0 /Clear DR
- $SHR_{DR} = v\ T_3 + w\ T_4$: DR <- shr DR /Shift right DR

Design and draw the logic diagram of DR_i (bit *i* of register DR), using any type of combinational circuits and a JK flip-flop.

Q5 Q5.1 RTL notation is used in the following table to describe the FETCH cycle of a memory reference instruction. Give short explanations of the respective microoperations in the last column

State	RTL Microoperations	Explanations
T_0	$AR \leftarrow PC$	
T_1	$IR \leftarrow M[AR]$ $PC \leftarrow PC + 1$	
T_2	$D_0, D_1, ..., D_7 \leftarrow Decode IR(14-12)$ $AR \leftarrow IR(11-0)$ $I \leftarrow IR(15)$	
$I \cdot D_7' \cdot T_3$	$AR \leftarrow M[AR]$	

Q5.2 Use RTL notation to describe the EXECUTION cycle of each of the three newly added instructions; give the corresponding control signals in terms of the instruction code and the sequence counter. Use as many lines for your table as you need; if needed more, please use the back of the page.

Solution: Use the last 3 rows of the **Instruction List** of the next page **ANNEX** to fill out the next table:

Instr.	Condition	RTL	Control signals: Bus Select S_2, S_1, S_0	SHR_{DR}	LD_{DR}	LD_{AC}	INR_{SC}	CLR_{SC}	ALU_{DR}
ASR	$D_7IT_3IR_5$	DR <- shr DR SC <- 0							
TAT									
DIV									

Derive the equations of the control signals from the above table

I=0 Direct addr. Assembly language syntax	I=0 Direct addr. Machine Code (Hex)	I=1 Indirect addr. Assembly language syntax	I=1 Indirect addr. Machine Code (Hex)	Description
AND adr	0adr	AND adr i	8(adr)	**AND** memory word M to AC
ADD adr	1(adr)	ADD adr i	9(adr)	**Add** memory word M to AC, carry to E
LDA adr	2(adr)	LDA adr i	A(adr)	**L**oad memory word from M to **A**C
STA adr	3(adr)	STA adr i	B(adr)	**St**ore content of **A**C in memory M
BUN adr	4(adr)	BUN adr i	C(adr)	**B**ranch **un**conditionally
BSA adr	5(adr)	BSA adr i	D(adr)	**S**ave return **A**ddress in m, **B**ranch to m+1
ISZ adr	6(adr)	ISZ adr i	E(adr)	**I**ncrement memory word M & **s**kip if **z**ero
CLA	7800			**Cl**ear **A**C
CLE	7400			**Cl**ear **E**
CMA	7200			**Com**plement **A**C
CME	7100			**Com**plement **E**
CIR	7080			**Ci**rculate **R**ight E and AC
CIL	7040			**Ci**rculate **L**eft E and AC
INC	7020			**Inc**rement AC
SPA	7010			**S**kip next instruction if **A**C is **p**ositive
SNA	7008			Skip next instruction if AC is negative
SZA	7004			Skip next instruction if AC is zero
SZE	7002			Skip next instruction if E is zero
HLT	7001			**Halt** computer
		INP	F800	**Inp**ut character to AC and Clear Flag
		OUT	F400	**Out**put character from AC and Clear Flag
		SKI	F200	**Sk**ip if **I**nput Flag is on
		SKO	F100	**Sk**ip if **O**utput Flag is on
		ION	F080	Turn **I**nterrupt **on**
		IOF	F040	Turn **I**nterrupt **off**
		ASR	**F020**	**Arithmetic Right Shift (DR <- DR/2)**
		DIV	**F010**	**Divide by 4 (DR <- DR/4)**
		TAT	**F008**	**Swap AC with DR (DR <-> AC)**

Basic Computer Block Diagram