Name-surname : **Signature:**

Student no :

Final Exam (16.01.2007)

_____CSE2201(2204)- Computer Organization, SENG2201- Computing Systems _____

**************************(You can only chose one of the questions from each group : QXa or QXb)**************************

Q1a. a. Write and explain the main functions and corresponding structural components of any general-purpose computer?
b. Classify and describe registers that a CPU consists of.
c. Write the memory hierarchy in a computer system

==========

A1a. a. **Data processing**: Data processing is performed by the CPU;
Data storage: Data is stored in the memory
Data movement: I/O moves data between the computer and its peripherals
Control: A control unit manages the computer's resources and orchestrates the performance of its functional parts in response to the instructions
b. **User Visible Registers**: General Purpose, Data, Address, Condition Codes
Control & Status Registers: Program Counter, Instruction Decoding Register, Memory Address Register, Memory Buffer Register
c. Registers, L1 Cache, L2 Cache, Main memory, Disk cache, Disk, Optical, Tape

---

Q1b. Consider a computer with the following characteristic: total 1 Mbyte of main memory; the word size is one byte; block size of 16 bytes; and cache size of 64Kbytes.
a. Show the format of main memory address.
b. For the main memory addresses of F0010, 01234, and CABBE, give the corresponding tag, cache line address, and word offsets for a direct-mapped cache.

==========

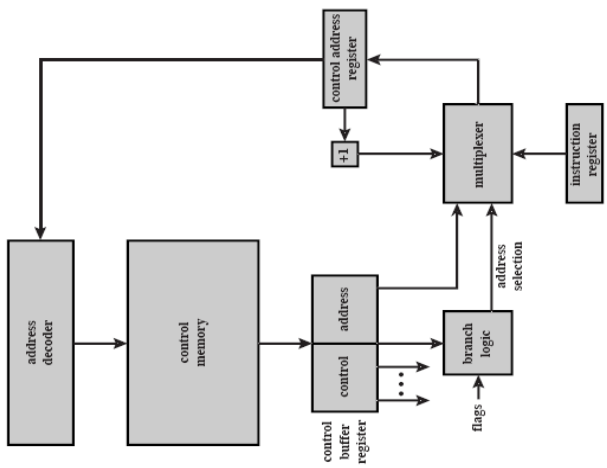
A1b. Because the block size is 16 bytes and the word size is 1 byte, this means there are 16 words per block. We will need 4 bits to indicate which word we want out of a block. Each cache line/slot matches a memory block. That means each cache slot contains 16 bytes. If the cache is 64Kbytes then 64Kbytes/16 = 4096 cache slots. To address these 4096 cache slots, we need 12 bits (2^{12} = 4096). Consequently, given a 20 bit (1 MByte) main memory address:
a. Bits 0-3 indicate the word offset (4 bits), Bits 4-15 indicate the cache slot (12 bits), Bits 16-19 indicate the tag (remaining bits)

	TAG ID	LINE ID	WORD ID
Main memory address =	4	12	4

b. F0010 = 1111 0000 0000 0001 0000
Word offset = 0000 = 0, Slot = 0000 0000 0001 = 001, Tag = 1111 = F
01234 = 0000 0001 0010 0011 0100
Word offset = 0100 = 4, Slot = 0001 0010 0011 = 123, Tag = 0000 = 0
CABBE = 1100 1010 1011 1011 1110
Word offset = 1110 = E, Slot = 1010 1011 1011 = ABB, Tag = 1100 = C

Q2a. Consider the control unit of the following figure. Asssume that the control memory is 24 bits wide. The control portion of the microinstruction format is divided into two fields. A microoperation field of 13 bits specicifies the microoperations to be performed. An address selection field specifies a condition, based on the flags, that will cause a microinstruction branch. There are 8 flags.

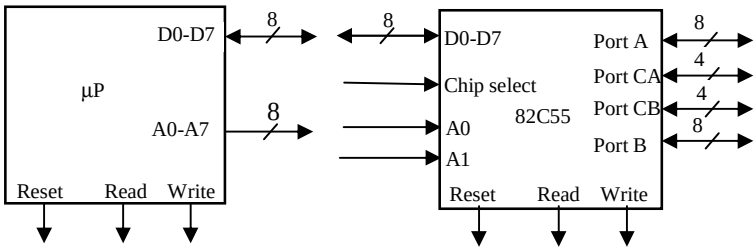
a. How many bits are in the address selection field?
b. How many bits are in the address field?
c. What is the size of the control memory?
d. How many microoperations exist in the control memory?



==========

A2a. a. Three bits are needed to specify one of 8 flags.
b. 24 – 13 – 3 = 8 bits are need in the address field
c. 2^8 = 256 words × 24 bits/word = 6144 bits.
d. 2^{13} = 8192 microoperations

Q2b. As an engineer, you are required to design a microcontroller with an 8 bit microprocessor having 8 address lines to address relevant memory locations. 82C55 PPI chip must be used to interface a keyboard and numeric display with the microprocessor. Design the required circuit for mapping PPI chip to the addresses 80, 81, 82, and 83 (addresses are in hexadecimal).

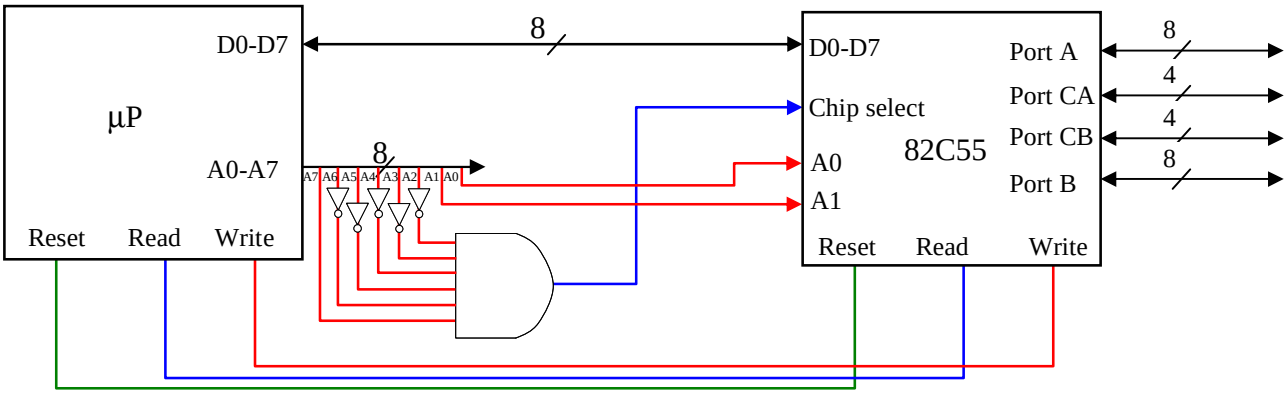


==========

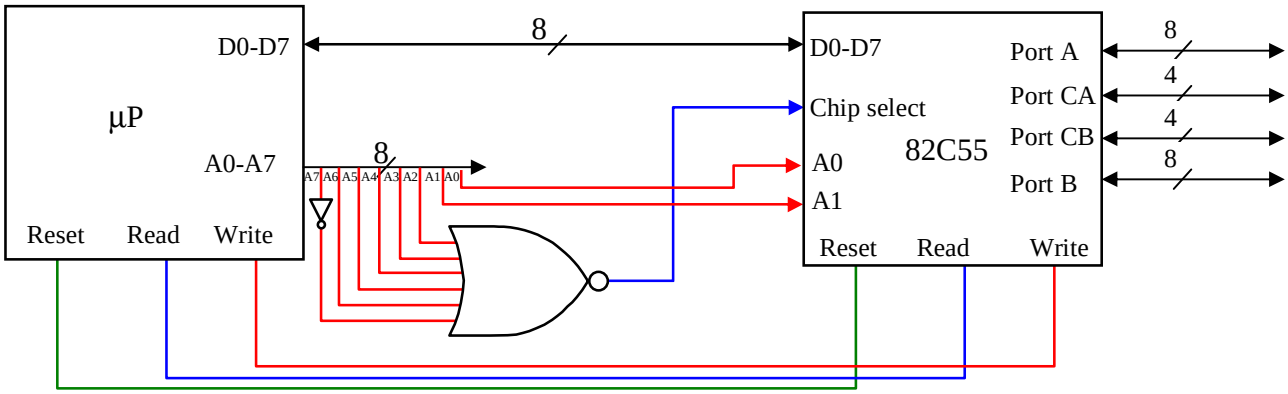
A2b.

A7	A6	A5	A4	A3	A2	A1	A0
1	0	0	0	0	0	x	x

Solution using AND gate:



Solution using NOR gate:



F

Q3a. In a microprocessor, each machine instruction cycle requires 4 clock cycles to be completed.
a. If the duration of a particular instruction is 0.8 ns, what is the microprocessor clock frequency?
b. If a wait state is inserted after each instruction, what would be the duration of a program consisting of 250 instructions?

==========

A3a. a. clock cycle = 0.8 / 4 = 0.2 ns clock frequency = 1 / clock cycle = 1 / (0.2 * 10^{-9}) = 5 * 10^9 Hz = 5GHz
b. instruction cycle with wait state = 0.8+0.2 = 1 ns, duration of a program = 1 ns * 250 = 250 ns

---

Q3b. If a computer system, in which addressing is by 8 bit unit, has the following instruction format:

31 – 27	26 – 23	22 – 20	19 – 10	9 – 0
Op-code	Addressing modes	Registers	Address1	Address2

a. How many registers this computer system might have?
b. How many bits are needed for the instruction register?
c. Total how many bits can be stored in the memory of this computer?
d. Since the addressing is by 8 bit unit, how many memory addresses this instruction will occupy?

==========

A3b. a. $2^3 = 8$ registers, b. 32 bits are needed, c. 2^{10} = 1024 Byte = 1024 * 8 = 8192 bit, d. 32 / 8 = 4

Q4a. In a computer system, a small part of memory is given in the following table. What would be the contents of accumulator after running the following assembler code? (All values are in hexadecimal)

RROTATE: Right rotate

ASHIFTR: Arithmetic shift right

```
LOAD IMMEDIATE A2
RROTATE
ADD INDIRECT A4
XOR DIRECT A3
ASHIFTR
```

Mem. Address	Data
A0	A4
A1	04
A2	04
A3	FF
A4	A1

==========

A4a.

LOAD IMMEDIATE A2	Acc = $(1010\ 0010)_2$	= $(A2)_{16}$
RROTATE	Acc = $(0101\ 0001)_2$	= $(51)_{16}$
ADD INDIRECT A4	Acc = $(0101\ 0001 + 0000\ 0100)_2 = (0101\ 0101)_2$	= $(55)_{16}$
XOR DIRECT A3	Acc = $(0101\ 0101 \text{ XOR } 1111\ 1111)_2 = (1010\ 1010)_2$	= $(AA)_{16}$
ASHIFTR	Acc = $(1101\ 0101)_2$	= $(D5)_{16}$

Q4b. In a signed (2s complement number system) addition, if the "carry in" and the "carry out" of the sign bit differ, there is an overflow.

a. Determine whether there is an overflow in the following operations or not. (use 4 bit 2s complement numbers)

4-7; -7-6; 5+7; 5+2

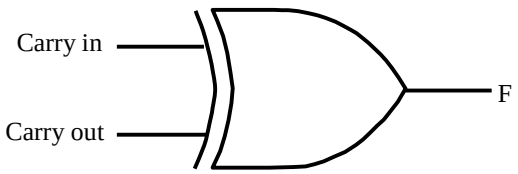
b. Design a circuit that whenever an overflow happens the output becames 1, otherwise 0.

==========

A4a. a.

operation			4-7			-7-6			5+7			5+2
Carry in			0			0			1			0
			0100			1001			0101			0101
		+	1001		+	1010		+	0111		+	0010
Result			1101			0011			1100			0111
Cary out		0			1			0			0	
overflow			no			yes			yes			no

b.



Q5a. Given $x = 110$ and $y = 011$ in unsigned integer notation, compute the product $p = x \times y$.

==========

	C	A	Q	M	Process
	0	000	110	011	Initial
Step 1	0	000	011	011	Shiftr through C,A, and Q
Step 2	0	011	011	011	A ← A + M
	0	001	101	011	Shiftr through C,A, and Q
Step 3	0	100	101	011	A ← A + M
	0	010	010	011	Shiftr through C,A, and Q

The following solution is also possible:

	C	A	Q	M	Process
	0	000	011	110	Initial
Step 1	0	110	011	110	A ← A + M
	0	011	001	110	Shiftr through C,A, and Q
Step 2	1	001	001	110	A ← A + M
	0	100	100	110	Shiftr through C,A, and Q
Step 3	0	010	010	110	Shiftr through C,A, and Q

Q5b. Write a simple VVM Assembly program to multiply two numbers entered from the keyboard. For the temporary storage use any memory location between 30 and 39. The final result will be saved in the memory location 40.

==========

A5b. The following program is one of the solutions

```
00      in
01      sto 30
02      in
03      sub 33
04      sto 31
05      lda 40
06      add 30
07      sto 40
08      lda 31
09      sub 33
10      brp 04
11      hlt
12      *33
13      dat 001
14      *40
15      dat 000
```

VVM instruction set (*nn* indicates an address location in the RAM), for example;
machine code **510** corresponds to Assembly code **LDA 10** (or **lda 10**).

Load Accumulator (5*nn*) [LDA *nn*] The content of RAM address *nn* is copied to the Accumulator Register, replacing the current content of the register. The content of RAM address *nn* remains unchanged. The Program Counter Register is incremented by one.
Store Accumulator (3*nn*) [STO *nn*] or **[STA *nn*]** The content of the Accumulator Register is copied to RAM address *nn*, replacing the current content of the address. The content of the Accumulator Register remains unchanged. The Program Counter Register is incremented by one.
Add (1*nn*) [ADD *nn*] The content of RAM address *nn* is added to the content of the Accumulator Register, replacing the current content of the register. The content of RAM address *nn* remains unchanged. The Program Counter Register is incremented by one.
Subtract (2*nn*) [SUB *nn*] The content of RAM address *nn* is subtracted from the content of the Accumulator Register, replacing the current content of the register. The content of RAM address *nn* remains unchanged. The Program Counter Register is incremented by one.
Input (901) [IN] or **[INP]** A value input by the user is stored in the Accumulator Register, replacing the current content of the register. Note that the two-digit operand does not represent an address in this instruction, but rather specifies the particulars of the I/O operation (see Output). The operand value can be omitted in the Assembly Language format. The Program Counter Register is incremented by one with this instruction.
Output (902) [OUT] or **[PRN]** The content of the Accumulator Register is output to the user. The current content of the register remains unchanged. Note that the two-digit operand does not represent an address in this instruction, but rather specifies the particulars of the I/O operation (see Input). The operand value can be omitted in the Assembly Language format. The Program Counter Register is incremented by one with this instruction.
Branch if Zero (7*nn*) [BRZ *nn*] This is a conditional branch instruction. If the value in the Accumulator Register is zero, then the current value of the Program Counter Register is replaced by the operand value *nn* (the result is that the next instruction to be executed will be taken from address *nn* rather than from the next sequential address). Otherwise (Accumulator >< 0), the Program Counter Register is incremented by one (thus the next instruction to be executed will be taken from the next sequential address).
Branch if Positive or Zero (8*nn*) [BRP *nn*] This is a conditional branch instruction. If the value in the Accumulator Register is nonnegative (i.e., >= 0), then the current value of the Program Counter Register is replaced by the operand value *nn* (the result is that the next instruction to be executed will be taken from address *nn* rather than from the next sequential address). Otherwise (Accumulator < 0), the Program Counter Register is incremented by one (thus the next instruction to be executed will be taken from the next sequential address).
Branch (6*nn*) [BR *nn*] or**[BRU *nn*]** or **[JMP *nn*]** This is an unconditional branch instruction. The current value of the Program Counter Register is replaced by the operand value *nn*. The result is that the next instruction to be executed will be taken from address *nn* rather than from the next sequential address. The value of the Program Counter Register is not incremented with this instruction.
No Operation (4*nn*) [NOP] or **[NUL]** This instruction does nothing other than increment the Program Counter Register by one. The operand value *nn* is ignored in this instruction and can be omitted in the Assembly Language format. (This instruction is unique to the VVM and is not part of the Little Man Model.)
Halt (0*nn*) [HLT] or **[COB]** Program execution is terminated. The operand value *nn* is ignored in this instruction and can be omitted in the Assembly Language format.