Name-surname : **Signature:**
Student no :

1st Midterm Exam (09.11.2006)

____CSE2201(2204)- Computer Organization, SENG2201- Computing Systems ___

1(10). List the main components and their functions of today's stored-program computers.

CPU: for processing data

Memory: For storing data

Input/Output: For moving data in and out of computer

System interconnection: For moving data amongst other components

---

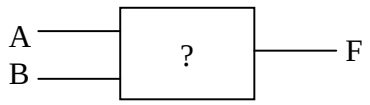
2(10). What is the corresponding decimal value of the binary 1010 when the number represents:

a. a four bit unsigned integer number?
b. a four bit signed integer number?
c. a four bit one's complement number?
d. a four bit two's complement number?

---

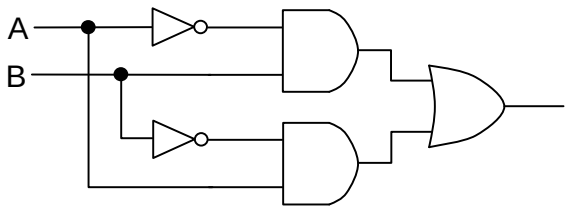
a. 10 b. -2 c. -5 d. -6

3(10). Construct a logic circuit (with two inputs and one output), in which the output becomes 1 when both inputs are different; the output becomes 0 when both inputs are the same.



A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

$F = \overline{A}B + A\overline{B}$



4(10). Why a cache memory is used in modern computers? Explain

---

Main memories are slower than the processors. This reduces the overall performance of a computer. To increase the performance, the cache, which is much faster than main memory, is

used. First, data is copied in to the cash form main memory, and then used by the processor. This will reduce accesses to the main memory during program execution resulting in a faster operation.

5(15). In the Intel 8088 microprocessor, a bus read operation requires four processor clock cycles. Assuming a processor clock rate of 8 MHz (f_c = 8 Mega Hertz);

a. what is the maximum data transfer rate? (Data transfer rate is the total number of bytes transferred in one second)

b. what is the maximum data transfer rate if a one clock cycle wait state is inserted per byte transferred? (In wait state processor does nothing)

---

a. The clock period is $T_{cc} = 1/f_c = 1/8$ MHz = 125 ns = 125×10^{-9} s.
One bus read cycle is $T_{bc} = 4\times T_{cc} = 4\times125$ = 500 ns = 0.5 µs.
If the bus cycles repeat one after another;
maximum data transfer rate is $1/(0.5\times10^{-6}) = 2\times10^{6}$ Byte per ssecond = 2 MB/s.

b. The wait state extends the bus read cycle by 125 ns, for a total duration of 0.625 µs. The corresponding data transfer rate is 1/0.625 = 1.6 MB/s.

6(10). What types of transfers must a computer's interconnection structure (or bus) support?

Memory to processor: The processor reads an instruction or a unit of data from memory.

Processor to memory: The processor writes a unit of data to memory.

I/O to processor: The processor reads data from an I/O device via an I/O module.

Processor to I/O: The processor sends data to the I/O device.

I/O to or from memory: For these two cases, an I/O module is allowed to exchange data directly with memory, without going through the processor, using direct memory access (DMA).

7(20). If one line of an assembler program (which loads contents of memory location 940 to the accumulator) for a hypothetical processor and corresponding machine code is given as:

address	assembler	machine code
300	lda 940	1940

(the values are in hexadecimal)

a. How many bits are needed for the program counter?

b. How many bits are needed for the instruction register?

c. What is the maximum directly addressable memory capacity in KBytes?

d. For the memory location 940, find the Tag, Line, and Word values in hexadecimal format for a direct-mapped cache, when *tag-id*=4 bits, *line-id*=6 bits, *word-id*=2 bits.

---

a. Because the address size is 3 nibbles (3x4=12 bits), a 12 bit Program Counter is required.

b. Instruction register size equals to *op-code size* + *address size*(4+12), which is 16 bits.

c. Maximum memory capacity = 2^{12} bytes = 2^{10+2} bytes = $2^2 \times 2^{10}$ bytes = 4 Kbytes

d. First, write the address in binary format: $(940)_{16} = (1001\ 0100\ 0000)_2$

Then define the Tag, Line, and Word values:

Tag-id← 100101000000 →Word-id

Line-id

Tag-id = 1001 = 9

Line-id = 00010000 = 10

Word-id = 0000 = 0

8(15). Write a simple VVM Assembly program to add two numbers entered from the keyboard, and save the result in address 50.

```
IN        Input 1st number to be added
STO 50    Store 1st number in address 50 temporarily
IN        Input 2nd number to be added
ADD 50    Add 1st number to the 2nd number
STO 50    Store the result in the address 50
HLT       Halt (program ends here)
```

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.

Name-surname : **Signature:**
Student no :

1st Midterm Exam (09.11.2006)

____CSE2201(2204)- Computer Organization, SENG2201- Computing Systems ___

1(10). List the main components and their functions of today's stored-program computers.

CPU: for processing data

Memory: For storing data

Input/Output: For moving data in and out of computer

System interconnection: For moving data amongst other components

---

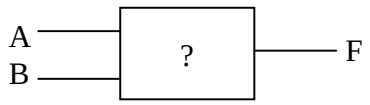
2(10). What is the corresponding decimal value of the binary 1001 when the number represents:

a. a four bit unsigned integer number?
b. a four bit signed integer number?
c. a four bit one's complement number?
d. a four bit two's complement number?

---

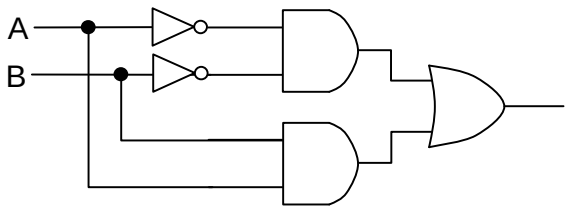
a. 9 b. -1 c. -6 d. -7

3(10). Construct a logic circuit (with two inputs and one output), in which the output becomes 0 when both inputs are different; the output becomes 1 when both inputs are the same.



A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

$F = \overline{A}\,\overline{B} + AB$



4(10). Why a cache memory is used in modern computers? Explain

---

Main memories are slower than the processors. This reduces the overall performance of a computer. To increase the performance, the cache, which is much faster than main memory, is

used. First, data is copied in to the cash form main memory, and then used by the processor. This will reduce accesses to the main memory during program execution resulting in a faster operation.

5(15). In the Intel 8088 microprocessor, a bus read operation requires four processor clock cycles. Assuming a processor clock rate of 4 MHz (f_c = 4 Mega Hertz);

a. what is the maximum data transfer rate? (Data transfer rate is the total number of bytes transferred in one second)

b. what is the maximum data transfer rate if a one clock cycle wait state is inserted per byte transferred? (In wait state processor does nothing)

---

a. The clock period is $T_{cc} = 1/f_c = 1/4$ MHz = 250 ns = 250×10^{-9} s.
One bus read cycle is $T_{bc} = 4\times T_{cc} = 4\times250 = 1000$ ns = 1 µs.
If the bus cycles repeat one after another;
maximum data transfer rate is $1/(1\times10^{-6}) = 1\times10^{6}$ Byte per ssecond = 1 MB/s.

b. The wait state extends the bus read cycle by 250 ns, for a total duration of 1.250 µs. The corresponding data transfer rate is 1/1.250 = 0.8 MB/s.

6(10). What types of transfers must a computer's interconnection structure (or bus) support?

Memory to processor: The processor reads an instruction or a unit of data from memory.

Processor to memory: The processor writes a unit of data to memory.

I/O to processor: The processor reads data from an I/O device via an I/O module.

Processor to I/O: The processor sends data to the I/O device.

I/O to or from memory: For these two cases, an I/O module is allowed to exchange data directly with memory, without going through the processor, using direct memory access (DMA).

7(20). If one line of an assembler program (which loads contents of memory location 640 to the accumulator) for a hypothetical processor and corresponding machine code is given as:

address	assembler	machine code
200	lda 640	1640

(the values are in hexadecimal)

a. How many bits are needed for the program counter?

b. How many bits are needed for the instruction register?

c. What is the maximum directly addressable memory capacity in KBytes?

d. For the memory location 640, find the Tag, Line, and Word values in hexadecimal format for a direct-mapped cache, when *tag-id*=4 bits, *line-id*=6 bits, *word-id*=2 bits.

---

a. Because the address size is 3 nibbles (3x4=12 bits), a 12 bit Program Counter is required.

b. Instruction register size equals to *op-code size* + *address size*(4+12), which is 16 bits.

c. Maximum memory capacity = 2^{12} bytes = 2^{10+2} bytes = $2^2 \times 2^{10}$ bytes = 4 Kbytes

d. First, write the address in binary format: $(640)_{16} = (0110\ 0100\ 0000)_2$

Then define the Tag, Line, and Word values:

Tag-id← 011001000000 →Word-id

Line-id

Tag-id = 1001 = 6

Line-id = 00010000 = 10

Word-id = 0000 = 0

8(15). Write a simple VVM Assembly program to subtract two numbers entered from the keyboard, and save the result in address 30.

```
IN        Input 1st number to be added
STO 30    Store 1st number in address 50 temporarily
IN        Input 2nd number to be added
SUB 30    Add 1st number to the 2nd number
STO 30    Store the result in the address 50
HLT       Halt (program ends here)
```

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.