

**CEG2136: COMPUTER ARCHITECTURE I**  
**CEG2536: ARCHITECTURE DES ORDINATEURS I**

**FINAL EXAM**

Length of Examination: 3 hrs

**Problem 1. (22 points) Short questions.**

1. Which of the following choice(s) is/are correct? A penalty will be applied for every wrong answer.

- (a) (4 pts) All sequential circuits can be implemented using only JK flip-flops and NOT gates

**only T flip-flops and NAND gates**

only D flip-flops and AND gates

**only D flip-flops and NAND gates**

- (b) (4 pts) An overflow can occur after

**the subtraction of two signed numbers**

the addition of two unsigned numbers

a logic shift left

an arithmetic shift right

2. Perform the following conversions.

(a) (2 pts)  $(AB.2AD)_{16} = ( \mathbf{171.1672} )_{10}$

(b) (2 pts)  $(10010.0)_2 = ( \mathbf{20} )_9$

3. A digital computer represents its floating point numbers using a signed 6-bit exponent and a signed normalized 10-bit mantissa. Negative exponent and mantissa values are expressed in 2's complement. Show the binary values of the exponent and mantissa to represent  $(21.1)_8$ .

10-bit mantissa= - ---'**0100010010**-----

6-bit exponent= **..00101**\_\_

4. Using Table 6, translate the following machine codes to their equivalent assembly instructions.

Machine code (in Hex.)	Equivalent in assembly
ecce	<b>BUN CCC I</b>
1234	<b>ADD 234</b>
DCBA	<b>BSA CBA I</b>
7002	<b>SIZE</b>

**Problem 2.** (20 points) A JK flip-flop  $A$  is used by a CPU to perform the following micro-operations (in RTL):

$xT_1: A \leftarrow 0$                       Reset  $A$  to 0  
 $yT_2: A \leftarrow 1$                       Set  $A$  to 1  
 $vT_3: A \leftarrow 1$                       Set  $A$  to 1

Otherwise the content of  $A$  remains the same. The signals  $T_1, T_2, \dots$  are outputs of a decoder.

1. Draw the logic diagram of the control circuit governing the flip-flop  $A$ .
2. The above micro-operations are replaced by the following ones:

$xT_1: A \leftarrow 0$                       Reset  $A$  to 0  
 $yT_2: A \leftarrow 1$                       Set  $A$  to 1  
 $zT_3: A \leftarrow A'$                       Complement  $A$   
 $wT_4: A \leftarrow G$                       Transfer external bit  $G$  to  $A$

Without using excitation tables and K-maps, draw the new logic diagram of the control circuit governing the flip-flop  $A$ .

**Problem 3.** (26 points) Consider Program 1.

1. Use Table 6 to translate Program 1 in its equivalent machine code by specifying the machine code of each instruction/operand (in hexadecimal), and its address in the memory (in hexadecimal)

Program 1: Assembly program

X,    ORG 0  
       HEX 0  
  
 Y,    ORG 17  
       LDA A  
       SZA  
       LDA B  
       INC  
       STA C  
       HLT  
  
 A,    ORG 100  
       0  
 B,    DEC 14  
 C,    HEX 000A  
       END

| Address in Hex | Content in Hex |

0000	0000
0017	2100
0018	7004
0019	2101
001A	7020
001B	3102
001C	7001
0100	0000
0101	000E
0102	000A

2. After the execution of the program, what is the content (in hexadecimal) of the word with the symbolic address C?                      **0001**
3. After the execution of the program, what is the content (in hexadecimal) of the register AC?  
 $AC =$            **0001**
4. After the execution of the program, what is the content (in hexadecimal) of the word with address zero?  
 $M[0] =$                       **0000**

**Problem 4.** (20 points)

<p><b>1.</b> Write an assembly subroutine (not a service routine) whose symbolic base address is SRT=200 (Hex.) to increment the content of address 161 (Hex.) if address 160 (Hex.) contains an even number. Do not exceed 12 lines of code.</p>	<p>This part is independent of part 1. Assume that 16 operands are stored in the memory starting from address 100 (Hex.) Write an assembly program to count the number of even operands and store it at address 161 (Hex.) The program should start at address 50 (Hex.) and use the subroutine SRT (of part 1). Do not exceed 14 lines of code.</p>
---	--

**Solution:**

```

SRT, 0
  IDA 160
  CIR
  SZE
  BUN EXT
  LDA 161
  INC
  STA 161
  EXT, BUN SRT I
    
```

```

          ORG 50
          CLA
          STA 161
LOP,     LDA PTR I
          STA 160
          BSA SRT
          ISZ PTR
          ISZ CTR
          BUN LOP
          HLT
PTR,     100
CTR,     FF00 /DEC -16
    
```

**Problem 5.** (36 points) Consider the computer of Lab 3, the architecture of which is described in Figure 1 and Tables 1, 2, 3, 4, and 5.. The instruction type is determined by the 2 most significant bits of the 8-bit register IR, as follows:

- $X_0 = IR'(7) IR'(6)$  denotes a memory-referenced instruction (MRI) in direct addressing mode;
- $X_1 = IR'(7) IR(6)$  denotes a register-referenced instruction (RRI); and
- $X_2 = IR(7) IR'(6)$  denotes a memory-referenced instruction (MRI) in indirect addressing mode.

The flip-flop S is a STOP register which prevents PC from being incremented if  $S = 1$ .

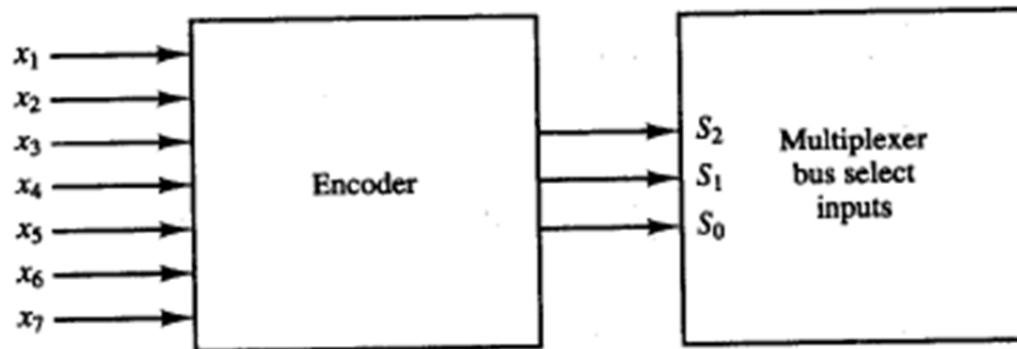
Assume that all registers are equipped with 3 control bits for loading the register, increment it by 1, and reset it to zero.

1. Find the list of all the micro-operations which use the bus and group them according to the register to be placed on the bus (IR, AR, PC, etc.)

- Copy Table 5 of annex to columns  $S_2 S_1 S_0$ , Register
- Inspect Tables 1, 2 and 3 of the annex and pick up the micro-operations that have corresponding registers on the right hand of RTL expressions

$S_2$	$S_1$	$S_0$	Register	Control functions
0	0	0	Memory	$T_1 : IR \leftarrow M [AR]$ $T_4 IR(6)' : AR \leftarrow M [AR]$ $T_5 X_2 : AR \leftarrow M [AR]$ $T_6 Y_0 : DR \leftarrow M [AR]$ $T_6 Y_1 : DR \leftarrow M [AR]$ $T_6 Y_4 : DR \leftarrow M [AR]$ $T_6 Y_5 : DR \leftarrow M [AR]$ $T_6 Y_6 : DR \leftarrow M [AR]$
0	0	1	AR	$T_6 Y_3 : PC \leftarrow AR$
0	1	0	PC	$T_0 : AR \leftarrow PC$ $T_3 IR(6)' : AR \leftarrow PC$
0	1	1	DR	$T_7 Y_1 : AC \leftarrow DR$ $T_8 Y_4 : M [AR] \leftarrow DR$
1	0	0	IR	
1	0	1	AC	$T_7 Y_2 : M [AR] \leftarrow AC$
1	1	0	OUTA	
1	1	1	None!	

2. Draw the logic diagram of the control circuit which governs the bus.



From the RTL of the last column of the above table derive

$$M: \quad x_M = x_0 = T_1 + T_4 IR(6)' + T_5 X_2 + T_5 (Y_0 + Y_1 + Y_4 + Y_5 + Y_6)$$

$$AR: \quad x_{AR} = x_1 = T_6 Y_3$$

$$PC: \quad x_{PC} = x_2 = T_0 + T_3 IR(6)'$$

$$DR: \quad x_{DR} = x_3 = T_7 Y_1 + T_8 Y_4$$

$$IR: \quad x_{IR} = x_4 =$$

$$S_0 = x_1 + x_3 + x_5 + x_7$$

$$AC: \quad x_{AC} = x_5 = T_7 Y_2$$

$$S_1 = x_2 + x_3 + x_6 + x_7$$

$$OUTA: \quad x_{OUTA} = x_6 = 0$$

$$S_2 = x_4 + x_5 + x_6 + x_7$$

3. Find the list of all the micro-operation which change the value of register PC.

From Tables 1, 2, 3 select all RTL's where PC is on the left side of the equations:

$$T_0 S': PC \leftarrow PC + 1$$

$$T_3 IR(6)' S': PC \leftarrow PC + 1$$

$$T_6 Y_3: PC \leftarrow AR$$

$$T_9 Y_4: \text{if } (DR = 0) S' \text{ then } (PC \leftarrow PC + 1)$$

$$T_{10} Y_4: \text{if } (DR = 0) S' \text{ then } (PC \leftarrow PC + 1)$$

4. Draw the logic diagram of the control circuit of PC.

$$LD_{PC} = T_6 Y_3$$

$$INC_{PC} = T_0 S' + T_3 IR(6)' S' + (T_9 Y_4 + T_{10} Y_4) (DR_7 DR_6 DR_5 DR_4 DR_3 DR_2 DR_1 DR_0)' S'$$

## Appendix

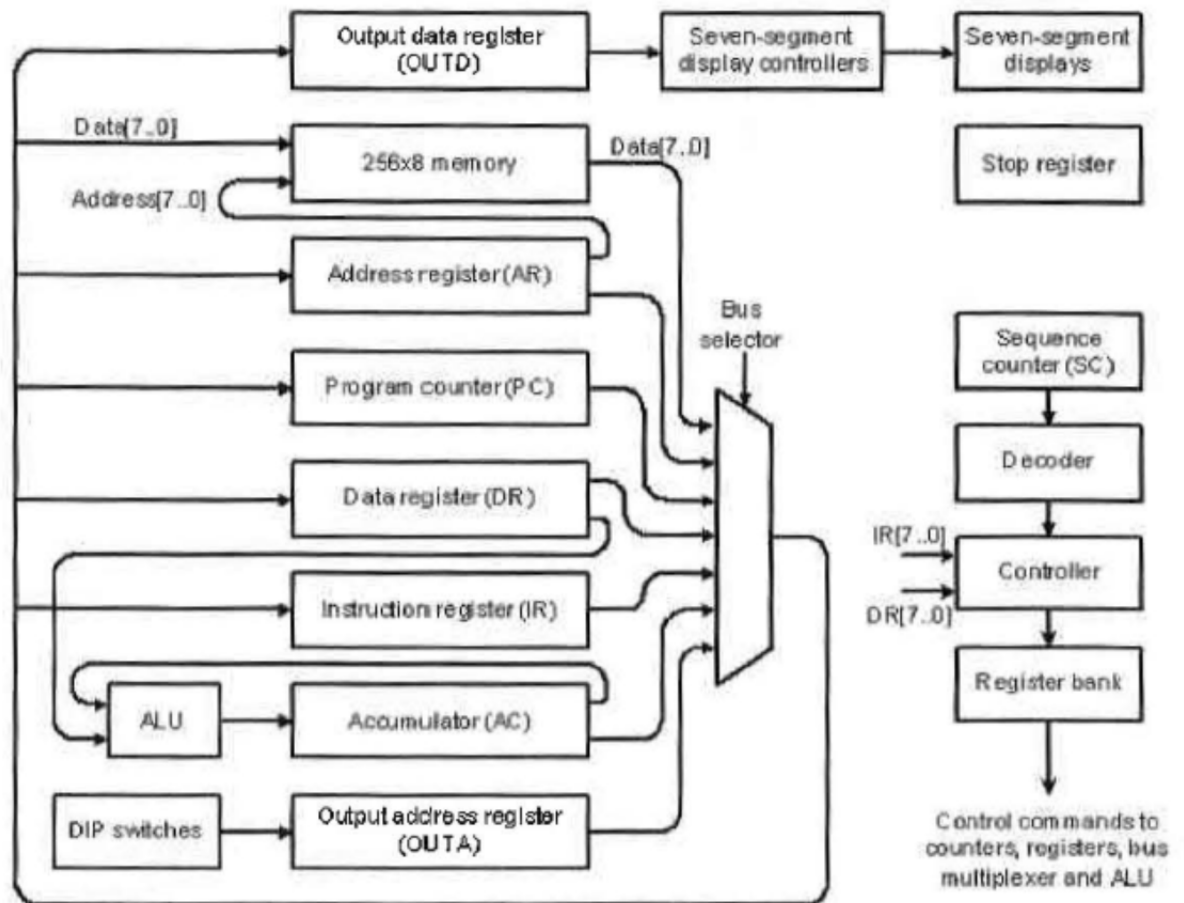


Table 1: Starting micro-operations in the instruction cycle of the 8-bit mini computer

Instant	Description	Notation (RTL)
$T_0$	Load the program counter PC in AR, and increment PC	$T_0 : AR \leftarrow PC$ $T_0S' : PC \leftarrow PC + 1$
$T_1$	Read instruction from memory and put it in the instruction register IR	$T_1 : IR \leftarrow M[AR]$
$T_2$	This cycle is not used to allow for the new value of the instruction to propagate in the controller	(nothing)
$T_3$	If $X_1$ , execute a register-referenced instruction (RRI)	$T_3X_1 : \text{execute an RRI instruction (Table 4)}$ $T_3X_1 : SC \leftarrow 0$
$T_3$	If $X_0$ or $X_2$ , read the memory address of the operand and place it in AR, and increment PC. Recall that $(X_0 + X_2) = IR(6)$	$T_3\overline{IR(6)} : AR \leftarrow PC$ $T_3\overline{IR(6)}S : PC \leftarrow PC + 1$
$T_4$	Read memory address into AR	$T_4\overline{IR(6)} : AR \leftarrow M[AR]$
$T_5$	If the reference to the memory is indirect, then read memory address again into AR	$T_5X_2 : AR \leftarrow M[AR]$
$T_5$	Si la référence à la mémoire est directe, aucune action supplémentaire n'est nécessaire	$T_5X_0 : (\text{rien})$
starting from $T_6$	Execute the memory-referenced instructions (MRI) described in Table 2	(see Table 2)

Table 2: Execution of a memory-referenced instruction in the 8-bit mini computer

Symbol		Notation (RTL)
ADD	$Y_0 = \overline{IR(6)} IR(1) \overline{IR(0)}$	$T_6 Y_0 : DR \leftarrow M[AR]$ $T_7 Y_0 : AC \leftarrow AC + DR, SC \leftarrow 0$
LDA	$Y_1 = \overline{IR(6)} IR(2)$	$T_6 Y_1 : DR \leftarrow M[AR]$ $T_7 Y_1 : AC \leftarrow DR, SC \leftarrow 0$
STA	$Y_2 = \overline{IR(6)} IR(3)$	$T_8 :$ (cycle not used to allow for the address bus to stabilize) $T_7 Y_2 : M[AR] \leftarrow AC, SC \leftarrow 0$
BUN	$Y_3 = \overline{IR(6)} IR(4)$	$T_6 Y_3 : PC \leftarrow AR, SC \leftarrow 0$
ISZ (assuming that the following instruction is a memory-referenced instruction (MRI), two addresses further)	$Y_4 = \overline{IR(6)} IR(5)$	$T_6 Y_4 : DR \leftarrow M[AR]$ $T_7 Y_4 : DR \leftarrow DR + 1$ $T_8 Y_4 : M[AR] \leftarrow DR$ $T_9 Y_4 (DR = 0) S' : PC \leftarrow PC + 1$ $T_{10} Y_4 (DR = 0) S' : PC \leftarrow PC + 1$ $T_{10} Y_4 : SC \leftarrow 0$
AND	$Y_5 = \overline{IR(6)} \overline{IR(1)} IR(0)$	$T_6 Y_5 : DR \leftarrow M[AR]$ $T_7 Y_5 : AC \leftarrow AC \wedge DR, SC \leftarrow 0$
SUB	$Y_6 = \overline{IR(6)} IR(1) IR(0)$	$T_6 Y_6 : DR \leftarrow M[AR]$ $T_7 Y_6 : AC \leftarrow AC - DR, SC \leftarrow 0$

Table 3: Function table of the 8-bit ALU

$S_2$	$S_1$	$S_0$	Operation
0	0	0	$F = AC + DR$
0	0	1	$F = AC - DR$
0	1	0	$F = \text{ashl } AC$
0	1	1	$F = \text{ashr } AC$
1	0	0	$F = AC \wedge DR$
1	0	1	$F = AC \vee DR$
1	1	0	$F = DR$ (transfer DR)
1	1	1	$F = \overline{AC}$

Table 4: Execution of a register-referenced instruction of the 8-bit mini computer

Symbol	Notation (RTL)
CLA	$T_3 X_1 IR(0) : AC \leftarrow 0$
CMA	$T_3 X_1 IR(1) : AC \leftarrow \overline{AC}$
ASL	$T_3 X_1 IR(2) : AC \leftarrow \text{ashl } AC$
ASR	$T_3 X_1 IR(3) : AC \leftarrow \text{ashr } AC$
INC	$T_3 X_1 IR(4) : AC \leftarrow AC + 1$
HLT	$T_3 X_1 IR(5) : S \leftarrow 1$

Table 5: Function table of the 8-bit bus

$S_2$	$S_1$	$S_0$	Register placed on the bus
0	0	0	Memory
0	0	1	AR
0	1	0	PC
0	1	1	DR
1	0	0	IR
1	0	1	AC
1	1	0	OUTA
1	1	1	None

Table 6: Instruction list of the 16-bit mini computer of the textbook (Mano'93)

Symbol	Hexadecimal code		Description
	<i>I</i> = 0	<i>I</i> = 1	
AND	0xxx	8xxx	AND memory word to AC
ADD	1xxx	9xxx	Add memory word to AC
LDA	2xxx	Axxx	Load memory word to AC
STA	3xxx	Bxxx	Store content of AC in memory
BUN	4xxx	Cxxx	Branch unconditionally
BSA	5xxx	Dxxx	Branch and save return address
ISZ	6xxx	Exxx	Increment and skip if zero
CLA	7800		Clear AC
CLE	7400		Clear E
CMA	7200		Complement AC
CME	7100		Complement E
CIR	7080		Circulate right AC and E
CIL	7040		Circulate left AC and E
INC	7020		Increment AC
SPA	7010		Skip next instruction if AC positive
SNA	7008		Skip next instruction if AC negative
SZA	7004		Skip next instruction if AC zero
SZE	7002		Skip next instruction if E is 0
HLT	7001		Halt computer
INP	F800		Input character to AC
OUT	F400		Output character from AC
SKI	F200		Skip on input flag
SKO	F100		Skip on output flag
ION	F080		Interrupt on
IOF	F040		Interrupt off