B38DF: Computer Architecture and Embedded Systems 2016
Part 1 – Tutorial 2 – Weeks 4-5

For the first three questions in this tutorial (true/false, multiple choice, and terms), the answers can be easily found in the lecture slides. The solutions of the remaining questions are given in this tutorial.

Q1. Indicate whether the following statements are True (**T**) or False (**F**).

Statement	**True/False (?)**
1) Optical disks are originally developed to serve as the main memory of the computers.	False
2) If registers were cheap enough, functionality-wise they could replace all forms of primary and secondary memory forms.	False
3) Magnetic disk reader head contains no optical elements.	True
4) Magnetic disk writer head contains no optical elements.	True
5) The preamble section on a magnetic disk track includes the error-correcting codes.	False
6) CDs an DVDs are both written and read using a laser beam.	True
7) In CD-ROMs the total space spared to encode a single byte is 8-bits of memory.	False
8) There is a single continuous spiral on DVDs on which the data is encoded.	True
9) A DVD has around double size memory capacity compared to a CD-ROM.	False
10) A larger portion of the bits in a program are used to specify where operands come from rather than what operations are being performed on them.	True

Q2. Indicate (circle) all and only the correct answer(s) for the following questions. More than one correct answer is possible.

Statement	**Answer(s)**
1) Which of the following is/are the main driver(s) for the development of different forms of memory.	a) Memory size b) Cost c) Access time d) Volatility
2) Which of the following is/are among the reasons for the difference in the memory capacity of a fotmatted and unformatted disk.	a) Inefficient programming b) Inefficient memory coding c) Preamble d) Error-correcting code e) Data backup sector
3) Which modes of addressing are observed in the following instruction statement: [MOV R1, #4]	a) Immediate addressing b) Memory direct addressing c) Register direct addressing d) Register indirect addressing e) Indexed addressing
4) The abstraction that interfaces between the hardware and lowest-level software is called:	a) Assembler b) Compiler c) Instruction Set Architecture d) Bus

5) Which of the following statements is FALSE on any Register-Register instructions	a) It may fetch two operands from the register b) It may store values in registers back into memory c) It may bring the operands to the ALU input registers d) It may perform operations on the operands

Q3. Write the term that is described by the following statement.

Statement	Term
1) The term for the language that both the compilers and hardware can understand.	ISA (level)

Q4. We managed to speedup 30% of the execution time by 15 times. What is the overall speedup?

$$S_l = \frac{1}{\frac{p1}{s1} + \frac{p2}{s2} + \cdots + \frac{pn}{sn}}$$

Use this space for your work

$$\boldsymbol{S_{tot} = \frac{1}{(1-0.3) + \frac{0.3}{15}} = 1.39\ times}$$

Q5. A program executes a series of 3 tasks, corresponding to three parts, where the percentages of execution time for each part are P1 = 0.25, P2 = 0.55 and P3 = 0.20, respectively. We have budget to change one of the second and third parts which both cost the same price. With the replacement, Part 2 of the program would speed up by 1.5 times and Part 3 by 5 times. Which part is more advantageous to be replaced?

$$S_l = \frac{1}{\frac{p1}{s1} + \frac{p2}{s2} + \cdots + \frac{pn}{sn}}$$

Use this space for your work

$$\boldsymbol{S_{P2} = \frac{1}{\frac{0.25}{1} + \frac{0.55}{1.5} + \frac{0.20}{1}} = 1.22\ times}$$

$$\boldsymbol{S_{P3} = \frac{1}{\frac{0.25}{1} + \frac{0.55}{1} + \frac{0.20}{5}} = 1.19\ times}$$

Part 2 is more advantageous to be replaced.

Q6. We are given a serial task which is fulfilled by four separate components, whose percentages of execution time are

p1 = 0.09, p2 = 0.18, p3 = 0.15, p4 = 0.48, p5 = 0.10 .

Then we are told that there are up-to-date components that can replace the present ones. Updating would cost the same amount for each of the components. The new components are faster than the present ones with the following factors:

i) 1st components 1.2 times faster
ii) 2nd component 6 times faster
iii) 3rd component 3 times faster
iv) 4th components 1.2 times faster
v) 5th components 2 times faster

Make a priority list of the components for replacement according to the advantage they will bring. Explain your answer with proper computations and show your work.

Hint: $S_l = \frac{1}{\frac{p1}{s1}+\frac{p2}{s2}+\cdots+\frac{pn}{sn}}$

Use this space for your work

In case of all components being replaced the speed up would take the form:

$$S_l = \frac{1}{\frac{p1}{s1}+\frac{p2}{s2}+\frac{p3}{s3}+\frac{p4}{s4}+\frac{p5}{s5}} = \frac{1}{\frac{0.09}{1.2}+\frac{0.18}{6}+\frac{0.15}{3}+\frac{0.48}{1.2}+\frac{0.10}{2}}$$

We would like to have a large decrease in each of the sum term in the denominator due to the replacement. Accordingly the advantage of replacement (ai) can be measured as:

$$ai = pi - \frac{pi}{si}$$

$$a1 = 0.09 - \frac{0.09}{1.2} = 0.015$$

$$a2 = 0.18 - \frac{0.18}{6} = 0.15$$

$$a3 = 0.15 - \frac{0.15}{3} = 0.1$$

$$a4 = 0.48 - \frac{0.48}{1.2} = 0.08$$

$$a5 = 0.10 - \frac{0.10}{2} = 0.05$$

$a2 > a3 > a4 > a5 > a1$

Acocrdingly the priority of replacement follows: $p2, p3, p4, p5, p1$.

Q7. Describe the following terms with a few sentences.

a) Expanding opcode

Use this space for your work

a)**Expanding opcode:**

Expanding opcode is a method to address the trade-off between the number of opcodes and number of memory addresses that can be encoded in a fixed length instruction format. In this scheme the length of the opcodes variable. The shorter opcodes are allocated for the instructions that require more operands (more memory addresses), whereas the longer opcodes are allocated for the instructions that require less or no operands (memory addresses). In this way, there is a spectrum of instructions generated with varying length opcodes and hense varying number of memory addresses.

Q8. Consider the different memories given by their abbreviations in the first column of the following table. Fill in the empty cells of the table considering ht efeatures of these memories.

Use this space for your work

Type	**Category** [Read-write/ Read-only/ Read-mostly]	**Erasure** [Electrical/ UV light/ others…]	**Byte alterable** [Yes/ No]	**Volatile** [Yes/ No]
RAM				
ROM				
PROM				
EPROM				
EEPROM				
Flash memory				

Type	**Category** [Read-write/ Read-only/ Read-mostly]	**Erasure** [Electrical/ UV light/ others…]	**Byte alterable** [Yes/ No]	**Volatile** [Yes/ No]
RAM	Read-write	Electrical	Yes	Yes
ROM	Read-only	Not possible	No	No
PROM	Read-only	Not possible	No	No
EPROM	Read-mostly	UV light	No	No
EEPROM	Read-mostly	Electrical	Yes	No
Flash memory	Read-mostly	Electrical	No	No

Q9. Describe what the below program does. Give line by line descriptions for each instruction. Give also a general description of the overall program.

```
        MOV R1,#0
        MOV R2,#A
        MOV R3,#A+4096
LOOP:   ADD R1,(R2)
        ADD R2,#4
        CMP R2,R3
        BLT LOOP
```

Use this space for your work

```
        MOV R1,#0          ; accumulate the sum in R1, initially 0
        MOV R2,#A          ; R2 = address of the array A
        MOV R3,#A+4096     ; R3 = address of the first word beyond A
LOOP:   ADD R1,(R2)        ; register indirect through R2 to get operand
        ADD R2,#4          ; increment R2 by one word (4 bytes)
        CMP R2,R3          ; are we done yet?
        BLT LOOP           ; if R2 < R3, we are not done, so continue
```

This program computes the sum of the elements of a 4096 entry long array located in the main memory in between the addresses A and A+4095. The result is stored in the register R1.

Q10.

Consider the three-instruction programmable processor with the following instrcutions and their machine codes, where “r” stands for the register memory address bits and “d” stands for data memory address bits:

- **Load** instruction:	0 0 0 0 r3 r2 r1 r0 d7 d6 d5 d4 d3 d2 d1	(from memory to register)
- **Store** instruction:	0 0 0 1 r3 r2 r1 r0 d7 d6 d5 d4 d3 d2 d1	(from register to memory)
- **Add** instruction:	0 0 1 0 ra3 ra2 ra1 ra0 rb3 rb2 rb1 rb0 rc3 rc2 rc1 rc0	(c=a+b)

(i) Write a program using arithmetic operations which performs the operation D[9]=D[0]+D[1], where D (data) represents the data memory address locations. In your program use RF[i] (registry file) for the registry at the address location [i].

(ii) Write your program in assembly code using the mnemonics indicated below:

Load (data d to registry a):	MOV Ra, d	$\rightarrow$ RF[a]=D[d]
Store (registry a do data d):	MOV d, Ra	$\rightarrow$ D[d]=RF[a]
Add:	ADD Ra, Rb, Rc	$\rightarrow$ RF[a]=RF[b]+RF[c]

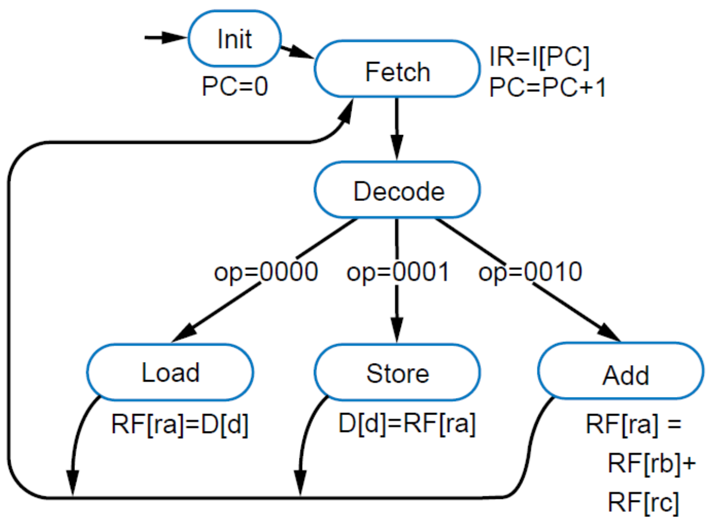
(iii) Write your program in machine code clearly indicating the order of instructions.

(iv) Plot the diagram of the high-level state machine description of the processor deribed in this question. Indicate the
-states,
-transitions,
-the transitions that correspond to the opcodes by writing the opcodes on the transition arrows,
-the updating of the PC and IR as output of the relevant states,
-and the aritnhmetic description of the output of the states that correspond to the execution of the three instrcutions.

Use this space for your work

Desired program

0: RF[0]=D[0]	0: 0000 0000 00000000	0: MOV R0, 0
1: RF[1]=D[1]	1: 0000 0001 00000001	1: MOV R1, 1
2: RF[2]=RF[0]+RF[1]	2: 0010 0010 0000 0001	2: ADD R2, R0, R1
3: D[9]=RF[2]	3: 0001 0010 00001001	3: MOV 9, R2
	machine code	<u>assembly code</u>


Init
PC=0
Fetch
IR=I[PC]
PC=PC+1
Decode
op=0000
op=0001
op=0010
Load
RF[ra]=D[d]
Store
D[d]=RF[ra]
Add
RF[ra] = RF[rb]+ RF[rc]