

PART B: Alpha Assembly Programming (total 50 marks)

In all questions in this section, assume the Alpha computer architecture.

All values are in hexadecimal (when written as 0x). The format of Integer operate instructions is displayed below.

Operate instruction format

31	26	25	21	20	16	15	13	12	11	5	4	0
opcode		Ra		Rb		000		LF	function		Rc	

31	26	25	21	20	13	12	11	5	4	0
opcode		Ra		value		LF	function		Rc	

As well, find below an extract of the ASCII character set.

00	NUL	1B	ESC	7F	DEL
21	!	22	"	30	0
31	1	32	2	33	3
34	4	35	5	36	6
37	7	38	8	39	9
3A	:	3B	;	3C	<
41	A	42	B	43	C
44	D	45	E	46	F
61	a	62	b	63	c
64	d	65	e	66	f
76	v	77	w	78	x
79	y	7A	z	7B	{

Answer briefly to each question.

a. Provide 2 different ways to add the content of register \$T1 and register \$T2, \$T2 holding the result. [2 marks]

b. What is the main difference between the instructions **addl** and **addq**? [2 marks]

addl: Register Ra is added to register Rb or a literal and the sign-extended 32-bit sum is written to Rc. The high order 32 bits of Ra and Rb are ignored.

c. How many registers (and what is their size contents) does the alpha chip studied in the lectures contain? [3 marks]

32 integer registers, 32 float registers both 64 bits long

d. What is the purpose of the address held in the Program Counter register? [3 marks]

e. Cite 3 advantages/inconvenient of using registers (internal chip memory) instead of using main memory (external to the chip)? [3 marks]

Registers are accessed faster than chip-external memory (+). Registers latency is shorter, they can be updated more often (+). Internal chip memory size is limited compared to external chip memory (-).

f. What are the main characteristics of a 64-bit load/store RISC architecture? [4 marks]

Mainly operations between internal registers. Only external operations are access to memory via load/store instructions. RISC for Reduced Instruction Set Computer Architecture.

g. An integer operate instruction is **uniquely** identified by its Opcode and function code (see "Operate instruction format" above). How many different integer operate instructions can be theoretically handled by the integer operate instruction format? [3 marks]

Opcode – > 6 bits: 2^6 possibilities. Function code – > 7 bits: 2^7 possibilities. Overall: 2^{13} possibilities.

Assume the following section program is to be executed:

```

data {
align quad;
one: long 0x2327fffb;
align quad;
two: quad 0xaabbccdd98765432;
align quad;
message: asciiz "0x123abc";
align quad;
three: quad;
align quad;
last: word 0xff33;
}

```

Further assume that the memory address corresponding to label **one** is 0x100000 and the memory has been reset beforehand. Indicate the contents of each byte of memory (and the correct addresses labels are referring to) after the program section has executed.

Display all values in **hexadecimal**.

[10 marks]

memory address	memory contents	memory address	memory contents	memory address	memory contents
0x100000	0xfb	0x100010	30	0x100010	00
0x100001	ff	0x100011	78	0x100011	00
0x100002	27	0x100012	31	0x100012	00
0x100003	23	0x100013	32	0x100013	00
0x100004	00	0x100014	33	0x100014	00
0x100005	00	0x100015	61	0x100015	00
0x100006	00	0x100016	62	0x100016	00
0x100007	00	0x100017	63	0x100017	00
0x100008	32	0x100018	00	0x100018	33
0x100009	54	0x100019		0x100019	ff
0x10000a	76	0x10001a		0x10001a	00
0x10000b	98	0x10001b		0x10001b	00
0x10000c	dd	0x10001c		0x10001c	00
0x10000d	cc	0x10001d		0x10001d	00
0x10000e	bb	0x10001e		0x10001e	00
0x10000f	aa	0x10001f		0x10001f	00

Assume the following instructions have to be executed one after another:

```
addq $t0, 0, $t0;
addq $t0, 38, $t1;
addq $t0, 0x26, $t1;
mulq $t0, 8, $t3;
mulq $t0, 0x10, $t4;
subq $31, 26, $t5;
not $t0;
or $t0, $t6, $t7;
and $t5, $t7, $t8; xor $t0, 51, $t9;
}
```

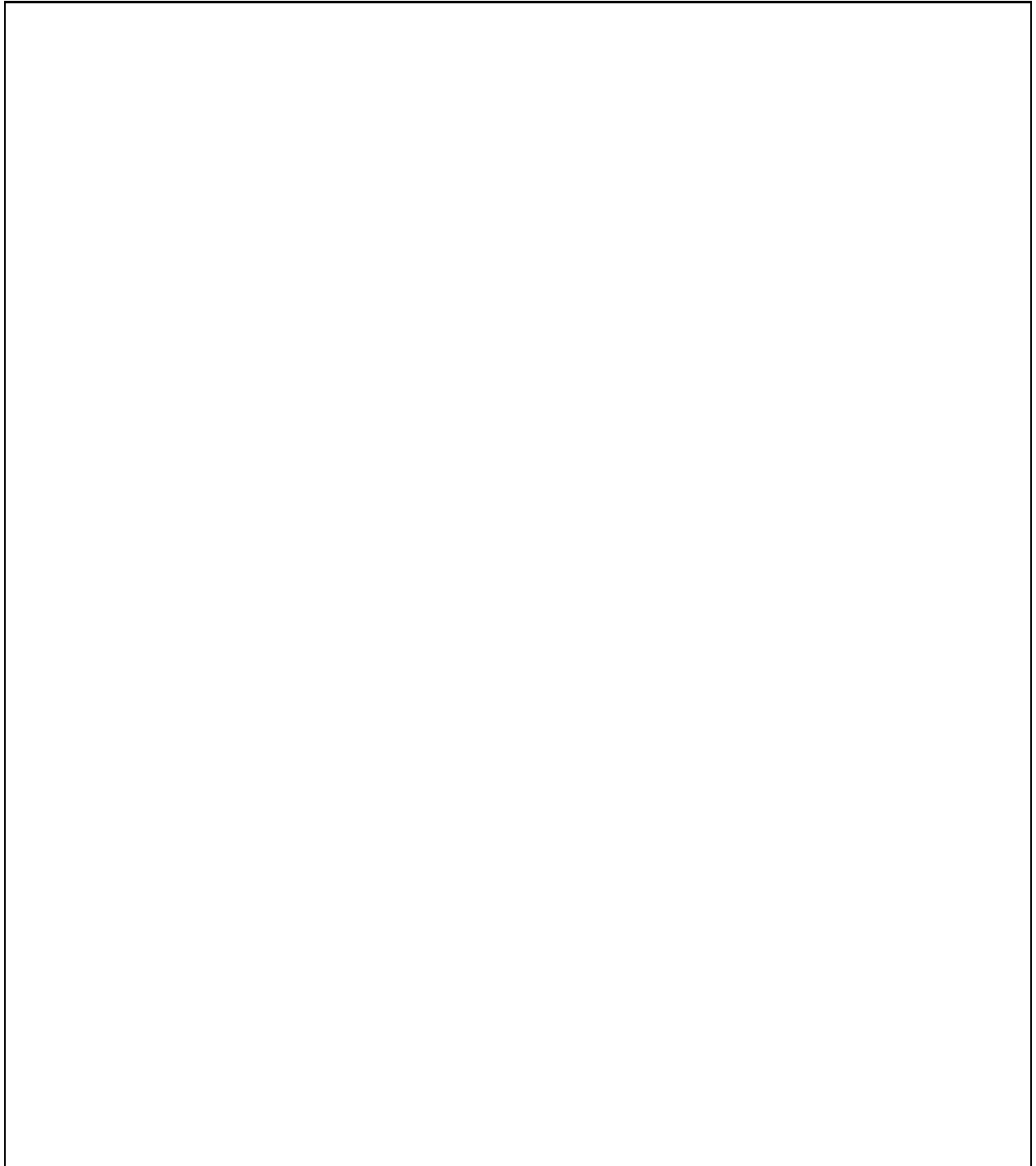
Further assume that register \$t0 **one** holds the quadword 0x34 before the above instructions. Indicate the contents of registers each time they are modified when executing the previous instructions **one after another**.

Display all values in **hexadecimal** as quadword.

[20 marks]

addq \$t0, 0, \$t0;	\$t0 = 0x34 or 0x0000000000000034
addq \$t0, 38, \$t1;	\$t1 = 0x5a
addq \$t0, 0x26, \$t1;	\$t1 = 0x5a
mulq \$t0, 8, \$t2; <=> sll \$t0, 3, \$t2;	\$t2 = 0x00...01a0
mulq \$t0, 0x10, \$t4; <=> sll \$t0, 4, \$t2;	\$t4 = 0x00...0340
subq \$31, 26, \$t5;	\$t5 = -0x1a = 0xff...ffe6
not \$t0;	\$t0 = 0xff...ffcb
or \$t0, \$t5, \$t6;	\$t6 = 0xff...fef
and \$t5, \$t6, \$t7;	\$t6 = 0xff...ffe6
xornot \$t0, 51, \$t8; 51 = 0x33	
not 51 = 0xff...fcc	
\$t7 = 0x007	

Additional area for answers:

A large, empty rectangular box with a thin black border, intended for providing answers to questions. The box is oriented vertically and occupies most of the page's width and height.