Arquitetura de Computadores I Ano Letivo 2011/12 - 1º Semestre

Teste Prático I

Nº Mec.:	Nome:		Data: 19-Nov-2011
----------	-------	--	-------------------

NOTE BEM: Leia atentamente todas as questões, comente o código usando a linguagem C e respeite a convenção de passagem de parâmetros e salvaguarda de registos que estudou. Na tradução para o *Assembly* do MIPS respeite rigorosamente os aspetos estruturais e a sequência de instruções indicadas no código original fornecido. O código em C apresentado pode não estar funcionalmente correto, pelo que **não deve ser interpretado**.

1) Observe o seguinte screen shot do MARS e responda às perguntas que se seguem.

Tev	kt Segment							□	7		Regi	isters
	at ooginont again	***************************************		000000000000		400000000000000000000000000000000000000		000000000000000000000000000000000000000	_	Name I	Num	Value
Progra	m Arguments:	Exame Prati	co de AC1							\$zero	0	0x00000000
Dient	Address	Code	Basic			Course				\$at	1	0x80000000
Bkpt	0x00400000		lw \$4,0x0000(\$29)	171:	lu ¢a/	Source O O(\$sp)		argc		\$v0	2	0x00000004
			addiu \$5,\$29,0x0004			\$a1 \$sp		argv		\$v1	3	0x00000000
	0x00400004		addiu \$6,\$5,0x0004	173:		\$a2 \$a1		envp		\$a0	4	0x7fffffef
	0x00400008		s11 \$2,\$4,0x0002	174:		v0 \$a0 2	- 1	CIIVD		\$a1	5	0x7fffefec
	0x00400000		addu \$6,\$6,\$2	175:		\$a2 \$a2 \$'	vr0			\$a2	6	0x7ffff000
	0x00400010		jal 0x00400024	176:	jal ma		***			\$a3	7	0x0000000
	0x00400014	0x00000000		177:	nop					\$t0	8	0x12345678
	0x00400018		addiu \$2,\$0,0x000a	179:	li \$v(0 10				\$t1	9	0x8000ffff
	0x0040001C	0x0000000c		180:	syscal		4	syscal		\$t2	10	0x00000000
	0x00400024		lw \$4,0x0004(\$5)	11:	lw	\$a0,4(5,5542111		\$t3	11	0x0000000
	0x00400024		addiu \$2,\$0,0x0004	12:	li	\$v0,4	, uz,			\$t4	12	0x0000000
	0x0040002c	0x0000000c		13:	syscal					\$t5	13	0x0000000
	0x0040002C		lui \$1,0x1234	14:	li		x12345678			\$t6	14	0x0000000
	0x00400034		ori \$8,\$1,0x5678			,,,,				\$t7	15	0x0000000
	0x00400034		lui \$1,0x8000	15:	li	\$t.1. 0:	x8000FFFF			\$30	16	0x0000000
			ori \$9,\$1,0xffff			,,,,,				\$81	17	0x0000000
	0x0040003C		or \$11,\$8,\$9	16:	or	\$t.3. \$	t0, \$t1			\$32	18	0x0000000
			xor \$12,\$8,\$9	17:	xor		t0, \$t1			\$83	19	0x0000000
	0x00400044		and \$13,\$8,\$9	18:	and		t0, \$t1			\$34	20	0x0000000
			sll \$14,\$9,0x0002	19:	sll	\$t6, \$				\$85	21	0x0000000
	0x0040004C		sra \$15,\$9,0x0002	20:	sra	\$t7, \$				\$36	22	0x0000000
			srl \$24,\$9,0x0002	21:	srl	\$t8, \$				\$87	23	0x0000000
		0x00090002		23:	jr	\$ra	01, L		Ţ.	\$t8	24	0x0000000
	0.00400030	0x03e00000	J2 402	201	J-	724			- III	\$t9	25	0x0000000
	ra koamont							000000000000		\$k0	26	0x0000000
	ta Segment 💮									\$k1	27	0x0000000
Addre				e (+c)		alue (+14)		Value (+1c)		\$gp	28	0x1000800
			206f 0x78657400 0x32							\$sp	29	0x7fffefe
			0000 0x00000000 0x00						Ţ	\$fp	30	0x0000000
v1001	10040 0x00000	00000 0x0000	0000 0x000000000 0x00	000000	0x00000000000000	KUUUUUUUU	0x00000000	0x00000000		\$ra	31	0x0040001
_										pc		0x0040004
	♦	0x100100	00 (.data) 🔻 🔽 Hex	adecim	al Addresses	✓ Hexadeo	imal Values	■ ASCII		hi		0x0000000
										10		0x00000000

Pergunta:	Resposta:
a) Admitindo que o programa começou a ser executado no endereço 0x00400000 qual	Pratico
o resultado do syscall na instrução presente no endereço 0x0040002c?	
b) Qual o valor actual do Stack Pointer?	0x7FFFEFE8
c) Qual a próxima instrução a ser executada?	or \$t3, \$t0, \$t1 (or \$11, \$8, \$9)
d) Indique qual o código ascii do caracter armazenado no endereço 0x1001000e	0x20
e) Indique o código máquina da instrução and \$t5, \$t0, \$t1	0x01096824

f) Determine e apresente na tabela, <u>em hexadecimal</u>, os valores armazenados nos registos \$\mathbf{t3} - \mathbf{t8}, após a execução de todas as instruções a partir do endereço atual do PC até ao endereço 0x00400054.

\$t3	0x9234FFFF	\$t5	0x00005678	\$t7	0xE0003FFF
\$t4	0x9234A987	\$t6	0x0003FFFC	\$t8	0x20003FFF

Zona de rascunho

DETI-UA; 2011 Cotações: 1-15%; 2-30%; 3-25%; 4-30%

2) Codifique em assembly do MIPS a seguinte função main: int proc_string(char *); int main(int argc, char *argv[]) static int array[10]; int *p; int res = 0; if((argc >= 10) || (argc < 2))return -1; for(p = array; p < (array + argc); p++)</pre> *p = proc string(*argv); res += *p; argv++; print_int(res); return 0; } 3) Codifique em *assembly* do MIPS a seguinte função **proc1**: int proc2(char *, char); int proc1(char *p, char c, int n) int res=0; do if(*p == c)res += proc2(p, c + *p); *p = *p + 'A' - 'a';n--; } while (n > 0); return res; } 4) Codifique em assembly do MIPS a seguinte função array proc: int *array proc(int *list, int num, int *count) { int i; int j = num-1, k = 0; for(i = 0; i < num; i++) if((list[i] < 200) || (list[i] >= list[j])) list[j] = list[i] ^ list[j]; // exclusive or k++;

*count = k;
return list;

}