CS3D1-1

TRINITY COLLEGE DUBLIN
THE UNIVERSITY OF DUBLIN

Faculty of Engineering, Mathematics and Science

School of Computer Science & Statistics

Integrated Engineering
Year 3 Annual Examinations

Trinity Term 2015

Microprocessor Systems I

Saturday, 16th May 2015 **Sports Centre** **09:30 - 11:30**

Prof. John Waldron

Instructions to Candidates

Question 1 is worth 50 marks. Each part of Question 2 is worth 5 marks, your best ten answers are counted. Answer both questions. Please detach the last page of the exam booklet and mark your answers on this and include with your answer book.

To be accompanied by an ARM Instruction Set and Addressing Mode Summary booklet.

Permitted Materials

Non-programmable calculators are permitted for this examination.

Section A

In this section marks are awarded for neatness, organisation, spelling, ability to communicate technical information and results, as well as assembly programming syntax, commenting and skill.

Design and write an ARM Assembly Language program that will determine the cardinality of a set of word values stored in memory. The result (cardinality) should be stored in r0. e.g. if the values stored in memory are ... 4, 9, 3, 4, 7, 9, 12, 10, 4, 7, 3, 12, 5, 5, 7 then the program should store 7 in r0.

1. (a) Describe what you are attempting to do in English. (10 marks)

   (b) Outline your algorithm using diagrams and pseudo code as appropriate. (15 marks)

   (c) Write down the actual ARM assembly code you would use, including comments. (15 marks)

   (d) Explain the test cases you would use, why you would chose them and the results expected. (10 marks)

Section B

Question 2.1

```
;;
;; After execution of the following instructions
;; what value will be in register r0?
;;
00 0410A0E3         MOV     r1, #0x4
04 910100E0         MUL     r0, r1, r1
08 0420A0E3         LDR     r2, =4
0c 920000E0         MUL     r0, r2, r0
10 0320A0E3         LDR     r2, =3
14 910202E0         MUL     r2, r1, r2
18 020080E0         ADD     r0, r0, r2
```

(A) 0x00000026 (B) 0x00000079 (C) 0x0000004C
(D) 0x0000050C (E) 0x00000214 (F) OTHER (5 marks)

Question 2.2

```
;;
;; After execution of the following instructions
;; what value will be in the condition code flags?
;;
00 0201A0E3         MOV   r0, #0x80000000
04 0111A0E3         MOV   r1, #0x40000000
08 007051E0         SUBS  r7, r1, r0
```

(A) 0x3 (B) 0x7 (C) 0x1
(D) 0xB (E) 0x9 (F) OTHER (5 marks)

Question 2.3

```
;;
;; After execution of the following instructions
;; what value will be in the condition code flags?
;;
00 0201A0E3         MOV   r0, #0x80000000
04 0211A0E3         MOV   r1, #0x80000000
08 002051E0         SUBS  r2, r1, r0
```

(A) 0xA (B) 0xC (C) 0x9
(D) 0x1 (E) 0x6 (F) OTHER (5 marks)

Question 2.4

```
;;
;; After execution of the following instructions
;; what value will be in register r2?
;;
00 0020A0E3         MOV     r2, #0x0
04 1610A0E3         LDR     r1, =0x16
08 A110B0E1         MOVS    r1, r1, LSR #1
0c 0020A2E2         ADC     r2, r2, #0
10 A110B0E1         MOVS    r1, r1, LSR #1
14 0020A2E2         ADC     r2, r2, #0
```

(A) 0x00000001 (B) 0x00000004 (C) 0x00000009
(D) 0x00000000 (E) 0x0000000A (F) OTHER (5 marks)

Question 2.5

```
;;
;; After execution of the following instructions
;; what value will be in register r0?
;;
00 0100A0E3          MOV     r0, #1
04 0410A0E3          MOV     r1, #0x4
08 0720A0E3          MOV     r2, #0x7
0c 0220B0E1          MOVS    r2, r2
           while
10 0200000A          BEQ     end
14 910000E0          MUL     r0, r1, r0
18 012052E2          SUBS    r2, r2, #1
1c FBFFFFEA          B       while
           end
```

(A) 0x00004000 (B) 0x0F6DC000 (C) 0x0000239E
(D) 0x00007E38 (E) 0x00006D62 (F) OTHER (5 marks)

Question 2.6

```
;;
;; After execution of the following instructions
;; what value will be in register r1?
;;
00 28009FE5          LDR    r0, =testcase
04 0010A0E3          MOV    r1, #0
08 0020D0E5 loop     LDRB   r2, [r0]
0c 5A0052E3          CMP    r2, #'Z'
10 0000003A          BLO    skip
14 011081E2          ADD    r1, r1, #1
18 010080E2 skip     ADD    r0, #1
1c 000052E3          CMP    r2, #0
20 F8FFFF1A          BNE    loop
           testcase
28 626B6E5A          DCB    "bknZhl2",0
   686C3200
```

(A) 0x00000006 (B) 0x00000012 (C) 0x0000000A
(D) 0x00000003 (E) 0x00000007 (F) OTHER (5 marks)

Question 2.7

```
;;
;; After execution of the following instructions
;; what value will be in register r0?
;;
00 0000A0E3          MOV     r0, #0
04 38109FE5          LDR     r1, =nums
08 0020A0E3          MOV     r2, #0
0c 023191E7 do1      LDR     r3, [r1, r2, LSL #2]
10 030080E0          ADD     r0, r0, r3
14 012082E2          ADD     r2, #1
18 080052E3          CMP     r2, #8
1c FAFFFF3A          BCC     do1

24 01000000 nums     DCD     0x1, 0x6, 0xC, 0xB
   06000000
   0C000000
   0B000000
34 04000000          DCD     0x4, 0x7, 0xD, 0x1
   07000000
   0D000000
   01000000
```

(A) 0x00000B9A (B) 0x000001B8 (C) 0x00000010
(D) 0x00000001 (E) 0x00000037 (F) OTHER (5 marks)

Question 2.8

```
;;
;; After execution of the following instructions
;; what value will be in register r4?
;;
00 28009FE5          LDR    r0, =str
04 0A32A0E3          LDR    r3, =0xA0000000
08 0120D0E4 loop     LDRB   r2, [r0], #1
0c 0120C3E4          STRB   r2, [r3], #1
10 000052E3          CMP    r2, #0
14 FBFFFF1A          BNE    loop
18 0A32A0E3          LDR    r3, =0xA0000000
1c 0340D3E5          LDRB   r4, [r3, #3]

24 38484550 str      DCB    "8HEPs4OpbhN",0
   73344F70
   62684E00
```

(A) 0x00000050 (B) 0x0000009C (C) 0x00001540
(D) 0x00000048 (E) 0x00000038 (F) OTHER (5 marks)

Question 2.9

```
;;
;; After execution of the following instructions
;; what value will be in register r2?
;;
00 29C3A0E3          LDR    r12, =0xA4000000
04 7200A0E3          LDR    r0, =0x72
08 04002CE5          STR    r0, [r12, #-4]!
0c 0F00A0E3          LDR    r0, =0xF
10 04002CE5          STR    r0, [r12, #-4]!
14 9E00A0E3          LDR    r0, =0x9E
18 04002CE5          STR    r0, [r12, #-4]!
1c 0700BCE8          LDMIA  r12!, {r0-r2}
20 482042E2          SUB    r2, #0x48
```

(A) 0x0000003A (B) 0x00000048 (C) 0x0000002F
(D) 0x0000002A (E) 0x00000001 (F) OTHER (5 marks)

Question 2.10

```
;;
;; After execution of the following instructions
;; what value will be in register r3?
;;
00 29D3A0E3          LDR   sp, =0xA4000000
04 50209FE5          LDR   r2, =str
08 060000EB          BL    vpc
0c FEFFFFEA stop     B     stop
10 0000A0E3 vp       MOV    r0, #0
14 610051E3          CMP    r1, #'a'
18 0000003A          BCC    yes
1c 1EFF2FE1          BX     lr
20 0100A0E3 yes      MOV    r0, #1
24 1EFF2FE1          BX     lr
28 00402DE9 vpc      STMFD  sp!,{lr}
2c 0030A0E3          MOV    r3, #0
30 0110D2E4 lop      LDRB   r1, [r2], #1
34 000051E3          CMP    r1, #0
38 0200000A          BEQ    edw
3c F3FFFFEB          BL     vp
40 003083E0          add    r3, r3, r0
44 F9FFFFEA          B      lop
48 0040BDE8 edw      LDMFD  sp!,{lr}
4c 1EFF2FE1          BX     lr

50 5A706441 str      DCB    "ZpdA4wli",0
   34776C69
   00
```

(A) 0x00000019 (B) 0x00000003 (C) 0x00000001
(D) 0x00000009 (E) 0x00000007 (F) OTHER (5 marks)

Question 2.11

```
;;
;; After execution of the following instructions
;; what value will be in register r12?
;;
00 2903A0E3          LDR    r0, =0xA4000000
04 28109FE5          LDR    r1, =str
08 0120D1E4 wh1      LDRB   r2, [r1], #1
0c 000052E3          CMP    r2, #0
10 0100000A          BEQ    endwh1
14 042020E5          STR    r2, [r0, #-4]!
18 FAFFFFEA          B      wh1
           endwh1
1c 08C090E5          LDR    r12, [r0, #8]

24 70643437 str      DCB    "pd47xcEWEOG8",0
   78634557
   454F4738
   00
```

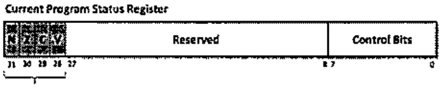
(A) 0x00000001 (B) 0x00000019 (C) 0x0000004F
(D) 0x00000229 (E) 0x0000009A (F) OTHER (5 marks)

Question 2.12

```
;;
;; After execution of the following instructions
;; what value will be in register r5?
;;
00 29D3A0E3          LDR   sp, =0xA4000000
04 E400A0E3          LDR   r0, =0xE4
08 9B10A0E3          LDR   r1, =0x9B
0c 04002DE5          STR   r0, [sp, #-4]!
10 04102DE5          STR   r1, [sp, #-4]!
14 010000EB          BL    vp
18 08D08DE2          ADD   sp, #8
1c FEFFFFEA stop     B     stop
20 00502DE9 vp       STMFD  sp!, {r12,lr}
24 08C08DE2          add    r12, sp, #8
28 1F002DE9          STMFD  sp!,{r0-r4}
2c 04309CE5          LDR    r3, [r12, #4]
30 00109CE5          LDR    r1, [r12, #0]
34 030081E0          ADD    r0, r1, r3
38 035041E0          SUB    r5, r1, r3
3c 1F00BDE8          LDMFD  sp!, {r0-r4}
40 0090BDE8          LDMFD  sp!, {r12,pc}
```

(A) 0xC31EC3E3 (B) 0xD7DA30CA (C) 0xFFFFFFB7
(D) 0x0700803C (E) 0x0000000A (F) OTHER (5 marks)

Condition Code Flags



ASCII Table

	0	1	2	3	4	5	6	7
0	NUL	DLE	SPACE	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

Conditional Branch Instructions

Branch Instruction	Condition Code Flag Evaluation	Description
B (or BAL)	don't care	unconditional (branch always)
BEQ	Z	equal
BNE	$\bar{Z}$	not equal
BCS / BHS	C	unsigned ≥
BCC / BLO	$\bar{C}$	unsigned <
BMI	N	negative
BPL	$\bar{N}$	positive or zero
BVS	V	overflow
BVC	$\bar{V}$	no overflow
BHI	$C\bar{Z}$	unsigned >
BLS	$\bar{C} + Z$	unsigned ≤
BGE	$NV + \bar{N}\bar{V}$	signed ≥
BLT	$N\bar{V} + \bar{N}V$	signed <
BGT	$\bar{Z}(NV + \bar{N}\bar{V})$	signed >
BLE	$Z + N\bar{V} + \bar{N}V$	signed ≤

Summary of LDR/STR Addressing Modes

Addressing mode	Syntax	W, B	H, SH, SB	Operation
Immediate Offset	`[<Rn>, #+/-<offset>]`	✓	✓	`address ← Rn +/- offset`
Register Offset	`[<Rn>, +/-<Rm>]`	✓	✓	`address ← Rn +/- Rm`
Scaled Register Offset	`[<Rn, +/-<Rm>, <shift> #<count>]`	✓		`address ← Rn +/- (Rm <shift> <count>)`

Immediate Pre-Indexed	`[<Rn>, #+/-<offset>]!`	✓	✓	`Rn ← Rn +/- offset`
				`address ← Rn`

Register Pre-Indexed	`[<Rn>, +/-<Rm>]!`	✓	✓	`Rn ← Rn +/- Rm`
				`address ← Rn`

Scaled Register Pre-Indexed	`[<Rn, +/-<Rm>, <shift> #<count>]!`	✓		`Rn ← Rn +/- (Rm <shift> <count>)`
				`address ← Rn`

Immediate Post-Indexed	`[<Rn>], #+/-<offset>`	✓	✓	`address ← Rn`
				`Rn ← Rn +/- offset`

Register Post-Indexed	`[<Rn>], +/-<Rm>`	✓	✓	`address ← Rn`
				`Rn ← Rn +/- Rm`

Scaled Register Post-Indexed	`[<Rn], +/-<Rm>, <shift> #<count>`	✓		`address ← Rn`
				`Rn ← Rn +/- (Rm <shift> <count>)`

Exam Number	Seat Number		
Question 2.1	Question 2.2	Question 2.3	Question 2.4
Question 2.5	Question 2.6	Question 2.7	Question 2.8
Question 2.9	Question 2.10	Question 2.11	Question 2.12