Coláiste na Tríonóide, Baile Átha Cliath
Trinity College Dublin
Ollscoil Átha Cliath | The University of Dublin

Faculty of Engineering, Mathematics and Science

School of Computer Science & Statistics

Integrated Engineering **Trinity Term 2016**
Year 3 Annual Examinations

Microprocessor Systems 1

Wednesday 4th May 2016 **Exam Hall** **14:00 – 16:00**

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. You may not start this examination until you are instructed to do so by the Invigilator.

Materials permitted for this examination:

Non-programmable calculators are permitted for this examination — please indicate the make and model of your calculator on each answer book used. To be accompanied by an ARM Instruction Set and Addressing Mode Summary booklet.

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 convert an ASCII string representation of a hexadecimal number stored in memory into a 32 bit 2's complement binary version, which you will also store in memory. Negative numbers will be indicated by a '-' at the start of the number. Your solution should work with both '+' and '-' at the start of the string, so for example "-0xA34" would be stored as 0xFFFFF5CC. You should test your solution with several different strings.

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 r3?
;;
00 E3A00011         MOV   r0, #0x11
04 E3A0100F         MOV   r1, #0xF
08 E0413000         SUB   r3, r1, r0

(A) 0x1D6D3C59 (B) 0xFFFFFFFE (C) 0x343AE0FF
(D) 0x00000001 (E) 0xF9B9928B (F) OTHER (5 marks)
```

```
Question 2.2
;;
;; After execution of the following instructions
;; what value will be in register r2?
;;
00 E3A0000A         MOV   r0, #0xA
04 E3A01004         MOV   r1, #0x4
08 E0020091         MUL   r2, r1, r0

(A) 0x00000001 (B) 0x00000046 (C) 0x0000002A
(D) 0x00000618 (E) 0x00000028 (F) OTHER (5 marks)
```

```
Question 2.3
;;
;; After execution of the following instructions
;; what value will be in the condition code flags?
;;
00 E3A00103         MOV   r0, #0xC0000000
04 E3A0120F         MOV   r1, #0xF0000000
08 E0513000         SUBS  r3, r1, r0

(A) 0x3 (B) 0x9 (C) 0x7
(D) 0x8 (E) 0x2 (F) OTHER (5 marks)
```

```
Question 2.4
;;
;; After execution of the following instructions
;; what value will be in register r4?
;;
00 E59F0008         LDR   r0, =0xA9C5
04 E59F1008         LDR   r1, =0x51DE
08 E1914200         ORRS  r4, r1, r0, LSL #4


(A) 0x000CAF33 (B) 0x000E7325 (C) 0x000FDE3A
(D) 0x000ADDDE (E) 0x00000001 (F) OTHER (5 marks)
```

```
Question 2.5
;;
;; After execution of the following instructions
;; what value will be in register r1?
;;
00 E3A000EC          LDR   r0, =0xEC
04 E3A010E8          LDR   r1, =0xE8
08 E1500001          CMP   r0, r1
0c 2A000000          BCS   a_label
10 E2511013          SUBS  r1, r1, #0x13
           a_label
14 DA000000          BLE   end
18 E251104F          SUBS  r1, r1, #0x4F
           end

(A) 0x000000CC (B) 0x00000106 (C) 0x00000006
(D) 0x00000099 (E) 0x000025A7 (F) OTHER (5 marks)
```

```
Question 2.6
;;
;; After execution of the following instructions
;; what value will be in register r0?
;;
00 E3A02000          MOV    r2, #0
04 E59F1038          LDR    r1, =nums
08 E5910000          LDR    r0, [r1]
0c E3A03005          LDR    r3, =5
10 E7914002 do1      LDR    r4, [r1, r2]
14 E1500004          CMP    r0, r4
18 AA000000          BGE    next
1c E1A00004          MOV    r0, r4
20 E2822004 next     ADD    r2, r2, #4
24 E2533001          SUBS   r3, r3, #1
28 2AFFFFF8          BCS    do1
30 000000E4 nums     DCD    0xE4, 0x6A1
   000006A1
38 000000B6          DCD    0xB6, 0xB7C
   00000B7C
40 000008B6          DCD    0x8B6

(A) 0x00000003 (B) 0x007C7D24 (C) 0x00000C84
(D) 0x00000B7C (E) 0x006A746C (F) OTHER (5 marks)
```

```
Question 2.7
;;
;; After execution of the following instructions
;; what value will be in register r1?
;;
00 E59F002C          LDR    r0, =test
04 E3A01000          MOV    r1, #0
08 E5D02000 loop     LDRB   r2, [r0]
0c E352005A          CMP    r2, #'Z'
10 3A000000          BLO    skip
14 E2811001          ADD    r1, r1, #1
18 E2800001 skip     ADD    r0, #1
1c E3520000          CMP    r2, #0
20 1AFFFFF8          BNE    loop
;;BigEndian
28 664E4C42 test     DCB    "fNLBNHms",0
   4E486D73
   00

(A) 0x00000001 (B) 0x00000002 (C) 0x00000008
(D) 0x00000003 (E) 0x0000000A (F) OTHER (5 marks)
```

```
Question 2.8
;;
;; After execution of the following instructions
;; what value will be in register r0?
;;
00 E3A00000          MOV     r0, #0
04 E59F1038          LDR     r1, =nums
08 E3A02000          MOV     r2, #0
0c E7913102 do1      LDR     r3, [r1, r2, LSL #2]
10 E0800003          ADD     r0, r0, r3
14 E2822001          ADD     r2, #1
18 E3520008          CMP     r2, #8
1c 3AFFFFFA          BCC     do1
24 00000009 nums     DCD     0x9, 0xA, 0xC, 0x4
   0000000A
   0000000C
   00000004
34 00000004          DCD     0x4, 0x6, 0x4, 0x2
   00000006
   00000004
   00000002

(A) 0x00000A29 (B) 0x00000027 (C) 0x00000001
(D) 0x000005C7 (E) 0x00000033 (F) OTHER (5 marks)
```

```
Question 2.9
;;
;; After execution of the following instructions
;; what value will be in register r1?
;;
00 E59F0024          LDR   r0, =nums
04 E3A01000          MOV   r1, #0
08 E0D020D2          LDRSB r2, [r0], #2
0c E0811002          ADD   r1, r2
10 E17020D1          LDRSB r2, [r0, #-1]!
14 E0811002          ADD   r1, r2
18 E1D020D2          LDRSB r2, [r0, #2]
1c E0811002          ADD   r1, r2
;;BigEndian
24 DC6C19B4 nums     DCB   0xDC, 0x6C, 0x19, 0xB4
28 B4CEC606          DCB   0xB4, 0xCE, 0xC6, 0x6

(A) 0x305B8961 (B) 0x58E7908B (C) 0x00000001
(D) 0xFFFFFFFC (E) 0xBD8AF230 (F) OTHER (5 marks)

Question 2.10
;;
;; After execution of the following instructions
;; what value will be in register r0?
;;
00 E3A0C329          LDR   r12, =0xA4000000
04 E3A0000E          LDR   r0, =0xE
08 E24CC004          SUB   r12, r12, #4
0c E58C0000          STR   r0, [r12]
10 E3A000A4          LDR   r0, =0xA4
14 E24CC004          SUB   r12, r12, #4
18 E58C0000          STR   r0, [r12]
1c E3A0009B          LDR   r0, =0x9B
20 E24CC004          SUB   r12, r12, #4
24 E58C0000          STR   r0, [r12]
28 E59C0000          LDR   r0, [r12]
2c E28CC004          ADD   r12, r12, #4
30 E59C0000          LDR   r0, [r12]
34 E28CC004          ADD   r12, r12, #4
38 E2400010          SUB   r0, #0x10

(A) 0x000000E7 (B) 0x0000008E (C) 0x0000003C
(D) 0x000000F9 (E) 0x00000094 (F) OTHER (5 marks)
```

```
Question 2.11
;;
;; After execution of the following instructions
;; what value will be in register r0?
;;
00 E3A01063          MOV   r1, #'c'
04 EB000002          BL    vp
08 E3A01033          MOV   r1, #'3'
0c EB000000          BL    vp
10 EAFFFFFE stop     B     stop
14 E3A00000 vp       MOV   r0, #0
18 E3510061          CMP   r1, #'a'
1c 3A000000          BCC   yes
20 E12FFF1E          BX    lr
24 E3A00001 yes      MOV   r0, #1
28 E12FFF1E          BX    lr
(A) 0x00000005 (B) 0x00000001 (C) 0x00000014
(D) 0x00000019 (E) 0x00000007 (F) OTHER (5 marks)

Question 2.12
;;
;; After execution of the following instructions
;; what value will be in register r5?
;;
00 E3A0D329          LDR   sp, =0xA4000000
04 E3A000AF          LDR   r0, =0xAF
08 E3A01055          LDR   r1, =0x55
0c E52D0004          STR   r0, [sp, #-4]!
10 E52D1004          STR   r1, [sp, #-4]!
14 EB000001          BL    vp
18 E28DD008          ADD   sp, #8
1c EAFFFFFE stop     B     stop
20 E92D5000 vp       STMFD  sp!, {r12,lr}
24 E28DC008          add    r12, sp, #8
28 E92D001F          STMFD  sp!,{r0-r4}
2c E59C3004          LDR    r3, [r12, #4]
30 E59C1000          LDR    r1, [r12, #0]
34 E0810003          ADD    r0, r1, r3
38 E0415003          SUB    r5, r1, r3
3c E8BD001F          LDMFD  sp!, {r0-r4}
40 E8BD9000          LDMFD  sp!, {r12,pc}
(A) 0x00000041 (B) 0x00000008 (C) 0x152DBF3A
(D) 0xFFFFFFA6 (E) 0x00000001 (F) OTHER (5 marks)
```

Condition Code Flags

Endianness

For ease of reading machine code and integer data in the second column are displayed in little endian format, byte data and strings in big endian format.

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