

## Midterm Exam

27.03.2019

Group A ---- A ----- A -----A  
Name:

Sign:

Grades

1. (20 pnt) Consider the program given in Table 1. Content of Stack pointer (SP, R13) is 0x00FFFF00 before the program is executed. Write the content of R0, R1, R2 and SP after each PUSH and POP instruction, into the shaded table cells. Be consistent: stack may grow upwards or downwards.

Table 1: Program memory

Content of general purpose registers:

Fill this part:

ADDRESS	INSTRUCTION	R0	R1	R2	SP
0x00000000	MOV R0, #0x18	0x00000018	-----	-----	0x00FFFF00
0x00000004	MOV R1, #0x21	0x00000018	0x00000021	-----	
0x00000008	MOV R2, #0x37	0x00000018	0x00000021	0x00000037	
0x0000000C	PUSH R2				
0x00000010	POP R1				
0x00000014	PUSH R0				
0x00000018	POP R2				

**2. (20 pnt) For loop.**

**a.** Translate the following C code to ARM assembly.

```
accu = 0;
for (i = 0; i < 6; i++) {
    accu += a[i];
}
```

---

```
MOV r0, #_____ // r0 will hold accumulation, initialize!
MOV r1, #_____ // first element's index
____ r2, _____ // load the address of array
```

Loop:

```
LDR r3, [_____, _____, _____] // load value (a[i]) from memory
ADD r0, r3, _____ // sum += a[i]
____ r1, r1, #_____ // update the index i= ...
CMP r1, #_____ // check the index, i
B__ Loop // loop only if i...
```

done: B done

```
arraya: .word 12, 28, 43, 46, 23, 876
```

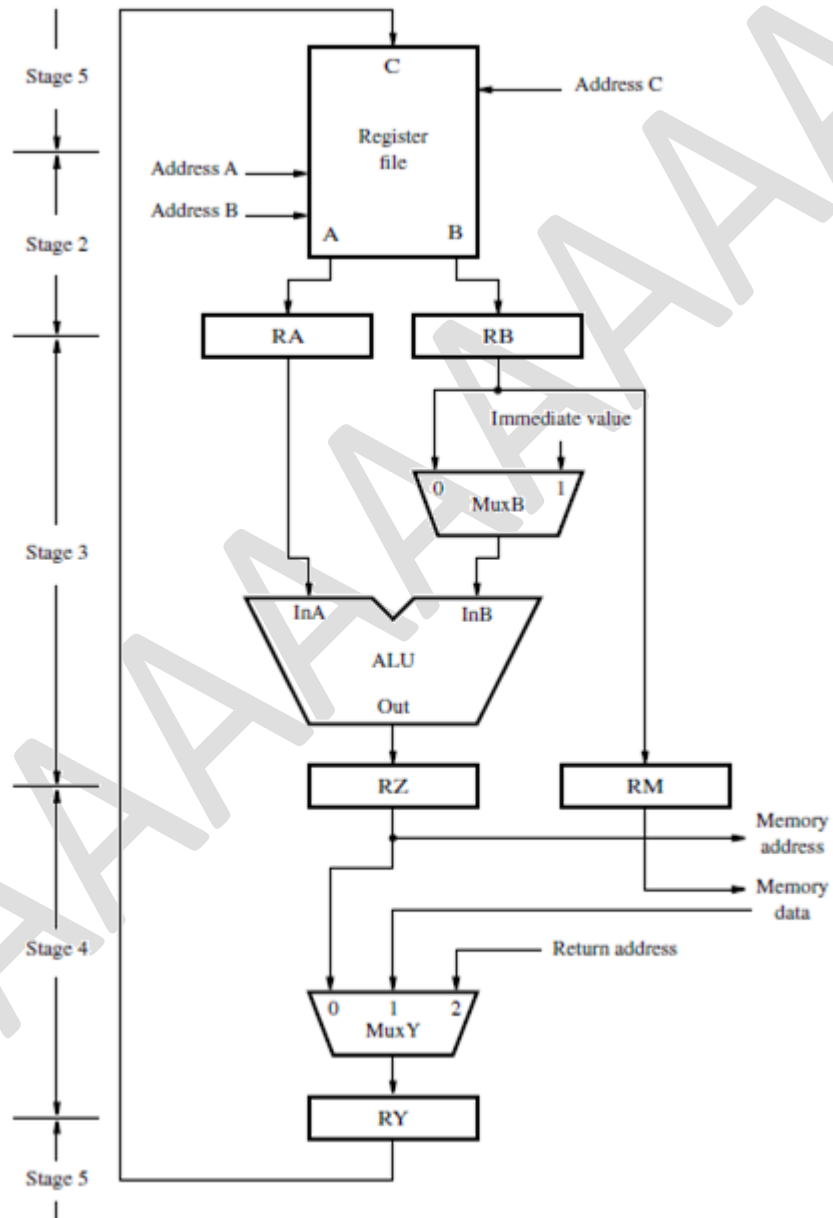
---

**b.** Draw the flow chart.

Group A ----- A ----- A ----- A  
 Name:  
 Sign:

### 3. (20 pnt) The instruction

SUBS R4, R5, #1  
 is stored in location 0x37C0 in the memory. At the time this instruction is fetched, registers R4 and R5 contain the values 0x1000 and 0x1004, respectively. The memory locations 0x1000 and 0x1004 contain the values 0x3214 and 0xA105. Write the 5 execution steps for this instruction. Give the values in registers R4, RA, RB, RM, RZ, and RY whenever they change as this instruction is executed. Note: Datapath flow is given in the figure. SUBS instruction performs as ARM instruction set.





4. (40 pnt) ASCII codes of “.Quiz .Report .Exam .Final” text contains 26 characters: letters, dots and whitespace. We write the ARM assembly program in Fig. 2.1 to make a list out of this text. After this program is executed, the dots will be replaced by numbers: “ 1Quiz 2Report 3Exam 4Final” and all characters will occupy their old locations, as shown in Fig. 2.2. ASCII code for dot and numbers are given in Fig. 2.3.

- Draw the flowchart of your program.
- Write LOOP subroutine
- Explain each step of your program.

```
.global _start
_start:

MOV R0, #26 // NUMBER OF CHARACTERS
LDR R1, =CHAR // FIRST CHAR ADDRESS
MOV R3, #_____ // Ascii for 0.
LOOP: .....
    (intentionally left blank)
    B LOOP
END: B END

CHAR: .byte 0x2e, 0x51, 0x75, 0x69, 0x7a, 0x20, 0x2e, 0x52,
0x65, 0x70, 0x6f, 0x72, 0x74, 0x20, 0x2e, 0x45, 0x78, 0x61,
0x6d, 0x20, 0x2e, 0x46, 0x69, 0x6e, 0x61, 0x6c
.end
```

Fig. 2.1: Program

<b>Before:</b>					
00000040	1769296174	1378754682	1919905893	1160650868	.Quiz .Report .E
00000050	544039288	1852393006	538995809	538976288	xam .Final
<b>After:</b>					
00000040	1769296177	1379016826	1919905893	1160978548	1Quiz 2Report 3E
00000050	544039288	1852393012	538995809	538976288	xam 4Final

Fig 2.2: Memory content before and after program execution.

Numbers	ASCII Codes
0	0x30
1	0x31
2	0x32
3	0x33
...	...
...	...
9	0x39
.	0x2e

Fig. 2.3: ASCII codes for uppercase and lowercase letters