

Study Questions (Chapter 03 – Part 5)

1. Question 3.10 on page 224: Why does the ARM provide a reverse subtract instruction `RSB r0, r1, r2`, that implements $[r0] = [r2] - [r1]$, when the normal subtraction instruction `SUB r0, r2, r1`, will do exactly the same job?
2. Question 3.20 on page 224: What is the difference between the `TEQ`, `TST`, `CMP` and `CMN` comparison instructions?
implementation diff.
3. Question 3.47 on page 226: What is wrong with the following instruction?
`MLA r0, r0, r1, r2`
Rd and Rn cannot be same register.
4. How to extend ARM arithmetic capabilities to make it able to add two extended precisions integer values (64-bits each). Give an example. Write down all assumptions you will make.
5. How to extend ARM arithmetic capabilities to make it able to add two extended precisions integer values (96-bits each). Give an example. Write down all assumptions you will make.
6. What are the main differences between `ADD`, `ADDS`, `ADC`, and `ADCS`?
Provide numeric examples to demonstrate the differences.
7. What are the main differences between `SUB`, `SUBS`, `SBC`, and `SBCS`?
Provide numeric examples to demonstrate the differences.
8. What are the differences between `NEG` and `MVN`?
Provide numeric examples to demonstrate the differences.
9. What are the main differences between `RSB`, `RSBS`, `RSC`, and `RSCS`?
Provide numeric examples to demonstrate the differences.
10. What are the main differences between `SUB`, `SUBS`, `RSB`, and `RSBS`?
Provide numeric examples to demonstrate the differences.
11. What are the main differences between `MUL`, `MULS`, `MLA`, and `MLAS`?
Provide numeric examples to demonstrate the differences.
12. `NEG` is a pseudo instruction. How does ARM implement it?
XOR #FFFFFF, r0
13. `LSL` is a pseudo instruction. How does ARM implement it?
14. `LSR` is a pseudo instruction. How does ARM implement it?
15. `ASR` is a pseudo instruction. How does ARM implement it?
16. `ROR` is a pseudo instruction. How does ARM implement it?
17. `RRX` is a pseudo instruction. How does ARM implement it?
18. Question 3.11 on page 224: If $r1 = 11110000111000101010000011111101$, and $r2 = 0000000011111110000111100001111$, what is the value of $r3$ after executing `BIC r3, r1, r2`?
r2 用于清除
 $r1 = 11110000111000101010000011111101$
 $r3 = 11110000111000101010000011111101$

19. Question 3.18 on page 224: Write one or more ARM instructions that will clear bits 20 to 25 inclusive in register r0. All the other bits of r0 should remain unchanged.
BIC r0, r0, #0xFFF0
20. What are the main differences between AND, ANDS, BIC, and BICS?
Provide numeric examples to demonstrate the differences.
21. Question 3.12 on page 224: If $r1 = 0FFF_{16}$, and $r2 = 4$, what is the value of r3 after each of the following instructions has been executed (assume that each instruction uses the same data).
- MOV r3, r1, LSL r2 *FFFF*
 - MOV r3, r1, LSR r2 *00FF*
 - MVN r3, r1, LSL r2 *000F*
 - MVN r3, r1, LSR r2 *7F*
22. Question 3.13 on page 224: If $r1 = 00FF_{16}$, what is the value of r0 after each of the following instructions has been executed (assume that each instruction uses the same data)?
- ADD r0, r1, r1, LSL #2 *03FC*
 - ADD r0, r1, r1, LSL #4 *0FF0*
 - ADD r0, r1, r1, ROR #4 *FFFF*
23. Question 3.14 on page 224: What is the effect of the instruction MOV r0, r0, ASR #31? *all bits equal to sign bit.*
24. Question 3.36 on page 225: Without using the ARM's multiplication instruction, write one or more instructions (using ADD, SUB, and shifting) to multiply by the following integers:
- 33
 - 1025
 - 4095
25. In ARM assembly language, how the RRX shift is encoded?
26. In ARM assembly language, how is the arithmetic shift left operation synthesized? Give an example.
LSL
27. In ARM assembly language, how is the rotate shift left operation synthesized? Give an example.
ROR
28. In ARM assembly language, how is the rotate shift left through carry operation synthesized? Give an example.