Study Questions (Part 2)
Friday March 1, 2019

Covering:

- Bitwise logical operations
- Arithmetic instructions (multiplication)
- Arithmetic instructions with shifts

As a good start, you need to answer the questions at the end of Chapter 3 of the textbook (pages 224-227).

1. Question 3.11 at page 224: If r1 = 11110000111000101010000011111101, and r2 = 00000000111111110000111100001111, what is the value of r3 after executing BIC **r3**,r1,r2?

2. Question 3.12 at 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).
   a. MOV **r3**,r1, LSL r2
   b. MOV **r3**,r1, LSR r2
   c. MVN **r3**,r1, LSL r2
   d. MVN **r3**,r1, LSR r2

3. Question 3.13 at 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)?
   a. ADD **r0**,r1,r1, LSL #2
   b. ADD **r0**,r1,r1, LSL #4
   c. ADD **r0**,r1,r1, ROR #4

4. Question 3.14 at page 224: What is the effect of the instruction MOV **r0**,r0, ASR #31?

5. Question 3.17 at page 224: ARM instructions have a 12-bit literal. Instead of permitting a word in the range 0 to 2^{12} - 1, the ARM uses an 8-bit format for the integer and a 4-bit alignment field that allows the integer to be shifted in steps of 2. What are the advantages and disadvantages of this mechanism in comparison with a straight 12-bit integer?

6. Question 3.18 at 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.

7. Question 3.19 at page 224: This is a classic problem of assembly language programming. Write a sequence of ARM instructions that swap the contents of registers r0 and r1 without using any additional registers or memory storage; that is, you can't move r1 to a temporary location.

8. Question 3.20 at page 224: What is the difference between the TEQ, TST, CMP and CMN comparison instructions?

9. Question 3.36 at 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:
   a. 33
   b. 1025
   c. 4095

10. Question 3.37 at page 225: A word consists of the bytes b4, b3, b2, b1. Write a function to invert the bits of b3 and clear the bits of b2 while leaving all other bits unchanged.

11. Question 3.44 at page 225: What does the following code do?
    TEQ r0,#0
    RSBMI **r0**,r0,#0

12. Question 3.45 at page 226: What is the meaning of the following mnemonics (and what do they do)?
    a. LDRB
    b. RSBLES
    c. CMPS

13. Question 3.47 at page 226: What is wrong with the following instruction?
    MLA **r0**,r0,r1,r2

14. Question 3.55 at page 226: Consider the following Java constructs. Express each in ARM assembly language. Assume all variables are single bit Booleans and are in registers r0 = A, r1 = B, r2 = C, r3 = D. *Note*. The Java operators &, |, ! are AND, OR, and NOT, respectively. The operators && and || are AND and OR operators that support short-circuit evaluation; that is, if the expression yields false (AND) or true (OR) further evaluation is halted.
    a. A = (B & C) | (!D);
    b. A = (B && C) || (!D);

15. Write only THREE ARM instructions to copy `0xFF` into `r1`, copy `0xEE` into `r2`, "and" the values in `r1` and `r2`, and finally store the result in `r0`.
    How do you enforce updating the condition flags after the anding operation?

16. Write only THREE ARM instructions to copy `0xFF` into `r1`, "and" `0xEE` with the value in `r1`, and finally store the result in `r0`.
    How do you enforce updating the condition flags after the anding operation?

17. Write only THREE ARM instructions to copy `0xFF` into `r1`, copy `0xEE` into `r2`, "exclusive or" the values in `r1` and `r2`, and finally store the result in `r0`.
    How do you enforce updating the condition flags after the exclusive oring operation?

18. Write only THREE ARM instructions to copy `0xFF` into `r1`, "exclusive or" `0xEE` with the value in `r1`, and finally store the result in `r0`.
    How do you enforce updating the condition flags after the exclusive oring operation?

19. Write only THREE ARM instructions to copy `0xFF` into `r1`, copy `0xEE` into `r2`, "inclusive or" the values in `r1` and `r2`, and finally store the result in `r0`.
    How do you enforce updating the condition flags after the inclusive oring operation?

20. Write only THREE ARM instructions to copy `0xFF` into `r1`, “inclusive or” `0xEE` with the value in `r1`, and finally store the result in `r0`.
    How do you enforce updating the condition flags after the inclusive oring operation?

21. Write only ONE ARM instruction to clear all 6 least significant bits in `r0` and to leave the other 24 bits unchanged.

22. Write only ONE ARM instruction to set all 6 least significant bits in `r0` and to leave the other 24 bits unchanged.

23. Write only ONE ARM instruction to flip all 6 least significant bits in `r0` and to leave the other 24 bits unchanged.

24. Write only ONE ARM instruction to clear all 6 most significant bits in `r0` and to leave the other 24 bits unchanged.

25. Write only ONE ARM instruction to set all 6 most significant bits in `r0` and to leave the other 24 bits unchanged.

26. Write only ONE ARM instruction to flip all 6 most significant bits in `r0` and to leave the other 24 bits unchanged.

27. Write only ONE ARM instruction to clear all 6 middle bits in `r0` and to leave the other 24 bits unchanged.

28. Write only ONE ARM instruction to set all 6 middle bits in `r0` and to leave the other 24 bits unchanged.

29. Write only ONE ARM instruction to flip all 6 middle bits in `r0` and to leave the other 24 bits unchanged.

30. Write only ONE ARM instruction to multiply the values in `r1` and `r2`, and.to store the result in `r0`.

31. Write only ONE ARM instruction to square the value in `r1` and.to store the result in `r0`.

32. Write only ONE ARM instruction to multiply the values in `r1` and `r2`, add to the result the value of `r3`, and. to store the result in `r0`.

33. Write only ONE ARM instruction to square the value in `r1`, add to the result the value of `r3`, and. to store the result in `r0`.

34. What are the main differences between AND, ANDS, BIC, and BICS?
    Provide numeric examples to demonstrate the differences.

35. What are the main differences between MUL, MULS, MLA, and MLAS?
    Provide numeric examples to demonstrate the differences.

36. After executing the following ARM instructions
    ```
            MOV   r0,#2_11001010
            MOV   r1,#2_00001111
            AND   r2,r1,r0
            ORR   r3,r1,r0
            EOR   r4,r1,r0
    ```
    What will be the values of the NZCV flags, as well as the values of r0, r1, r2, r3, and r4?

37. After executing the following ARM instructions

```
        LDR  r0,=0x87654321
        LDR  r1,=0x87654321
        ANDS r2,r1,r0
        BICS r3,r1,r0
        EORS r4,r1,r0
        ORRS r5,r1,r0
```

What will be the values of the NZCV flags, as well as the values of r0, r1, r2, r3, r4 and r5?

38. After executing the following ARM instructions

```
        LDR  r0,=0x87654321
        LDR  r1,=-0x87654321
        ANDS r2,r1,r0
        BICS r3,r1,r0
        EORS r4,r1,r0
        ORRS r5,r1,r0
```

What will be the values of the NZCV flags, as well as the values of r0, r1, r2, r3, r4 and r5?

39. After executing the following ARM instructions

```
        LDR  r0,=0xFEDCBA98
        LDR  r1,=0x87654321
        ANDS r2,r1,r0
        BICS r3,r1,r0
        EORS r4,r1,r0
        ORRS r5,r1,r0
```

What will be the values of the NZCV flags, as well as the values of r0, r1, r2, r3, r4 and r5?

40. After executing the following ARM instructions

```
        LDR  r0,=0xFEDCBA98
        LDR  r1,=-0x87654321
        ANDS r2,r1,r0
        BICS r3,r1,r0
        EORS r4,r1,r0
        ORRS r5,r1,r0
```

What will be the values of the NZCV flags, as well as the values of r0, r1, r2, r3, r4 and r5?

41. After executing the following ARM instructions

```
        LDR  r0,=0xFEDCBA98
        LDR  r1,=0xFEDCBA98
        ANDS r2,r1,r0
        BICS r3,r1,r0
        EORS r4,r1,r0
        ORRS r5,r1,r0
```

What will be the values of the NZCV flags, as well as the values of r0, r1, r2, r3, r4 and r5?

42. After executing the following ARM instructions

```
        LDR  r0,=0xFEDCBA98
        LDR  r1,=-0xFEDCBA98
        ANDS r2,r1,r0
        BICS r3,r1,r0
        EORS r4,r1,r0
        ORRS r5,r1,r0
```

What will be the values of the NZCV flags, as well as the values of r0, r1, r2, r3, r4 and r5?

43. After executing the following ARM instructions
```
LDR  r0,=0x1234567
MVN  r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

44. After executing the following ARM instructions
```
LDR  r0,=0x1234567
MVNS r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

45. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MVN  r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

46. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MVNS r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

47. After executing the following ARM instructions
```
LDR  r0,=-0x123
MVN  r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

48. After executing the following ARM instructions
```
LDR  r0,=-0x123
MVNS r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

49. After executing the following ARM instructions
```
LDR  r0,=-0xFFFFF123
MVN  r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

50. After executing the following ARM instructions
```
LDR  r0,=-0xFFFFF123
MVNS r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

51. After executing the following ARM instructions
```
LDR  r0,=-0xFFFFFFFF
MVN  r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

52. After executing the following ARM instructions
```
LDR  r0,=-0xFFFFFFFF
MVNS r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

53. Is there any difference between the ARM instructions “`MVN r0, #-0x12`” and “`MOV r0, #0x11`”?

54. Is there any difference between the ARM instructions “`MOV r0, #-0x12`” and “`MVN r0, #0x11`”?

55. After executing the following ARM instructions
```
        MOV  r1,#0x10
        MOV  r2,#0x20
        MOV  r3,#0x30
        MULS r3,r2,r1
```
What will be the values of the NZCV flags, as well as the values of r1, r2, and r3?

56. After executing the following ARM instructions
```
        LDR  r1,=0x10
        LDR  r2,=0x20
        LDR  r3,=0x30
        MULS r3,r2,r1
```
What will be the values of the NZCV flags, as well as the values of r1, r2, and r3?

57. After executing the following ARM instructions
```
        LDR  r1,=0x10
        LDR  r2,=-0x20
        LDR  r3,=0x30
        MULS r3,r2,r1
```
What will be the values of the NZCV flags, as well as the values of r1, r2, and r3?

58. After executing the following ARM instructions
```
        LDR  r1,=-0x10
        LDR  r2,=-0x20
        LDR  r3,=0x30
        MULS r3,r2,r1
```
What will be the values of the NZCV flags, as well as the values of r1, r2, and r3?

59. After executing the following ARM instructions
```
        LDR  r1,=0x10
        LDR  r2,=0x20
        LDR  r3,=0x30
        LDR  r4,=0x40
        MLAS r4,r3,r2,r1
```
What will be the values of the NZCV flags, as well as the values of r1, r2, r3, and r4?

60. After executing the following ARM instructions
```
        LDR  r1,=0x10
        LDR  r2,=-0x20
        LDR  r3,=0x30
        LDR  r4,=0x40
        MLAS r4,r3,r2,r1
```
What will be the values of the NZCV flags, as well as the values of r1, r2, r3, and r4?

61. After executing the following ARM instructions
```
        LDR  r1,=-0x10
        LDR  r2,=-0x20
        LDR  r3,=0x30
        LDR  r4,=0x40
        MLAS r4,r3,r2,r1
```
What will be the values of the NZCV flags, as well as the values of r1, r2, r3, and r4?

62. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MOVS r1,r0,LSL#7
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

63. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MOVS r1,r0,LSR#7
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

64. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MOVS r1,r0,ASR#7
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

65. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MOVS r1,r0,ROR#7
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

66. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MOVS r1,r0,RRX
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

67. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
NEGS r1,r0
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

68. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MVNS r1,r0,LSL#7
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

69. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MVNS r1,r0,LSR#7
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

70. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MVNS r1,r0,ASR#7
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

71. After executing the following ARM instructions
```
LDR  r0,=0xFEDCBA98
MVNS r1,r0,ROR#7
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

72. After executing the following ARM instructions
```
        LDR  r0,=0xFEDCBA98
        MVNS r1,r0,RRX
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

73. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        ADDS r1,r0,r0,LSL#4
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

74. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        SUBS r1,r0,r0,LSL#4
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

75. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        ADDS r1,r0,r0,LSR#4
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

76. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        SUBS r1,r0,r0,LSR#4
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

77. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        ADDS r1,r0,r0,ASR#4
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

78. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        SUBS r1,r0,r0,ASR#4
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

79. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        ADDS r1,r0,r0,ROR#4
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

80. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        SUBS r1,r0,r0,ROR#4
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

81. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        ADDS r1,r0,r0,ROR#28
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

82. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        SUBS r1,r0,r0,ROR#28
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

83. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        ADDS r1,r0,r0,RRX
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

84. After executing the following ARM instructions
```
        LDR  r0,=0xC8
        SUBS r1,r0,r0,RRX
```
What will be the values of the NZCV flags, as well as the values of r0 and r1?

85. In ARM assembly language, what is the range of the LSL steps, i.e., the allowable LSL constant values?

86. In ARM assembly language, what is the range of the LSR steps, i.e., the allowable LSR constant values?

87. In ARM assembly language, what is the range of the ASR steps, i.e., the allowable ASR constant values?

88. In ARM assembly language, what is the range of the ROR steps, i.e., the allowable ROR constant values?

89. In ARM assembly language, how the RRX shift is encoded?

90. In ARM assembly language, how is the arithmetic shift left operation synthesized? Give an example.

91. In ARM assembly language, how is the rotate shift left operation synthesized? Give an example.

92. In ARM assembly language, how is the rotate shift left through carry operation synthesized? Give an example.

93. Write only ONE ARM instruction that adds 8 times of the content of register r2 to r4 and puts the result in register r8, i.e., r8 ← 8 × r2 + r4

94. Write a suitable ARM assembly segment of code to implement the following code, where Z is an integer array (4 bytes per element) which is located at address 0x120.
```
for(r0 = 0; r0 <= 20; r0++)
  Z[r0] = r0;
```

95. Write a suitable ARM assembly segment of code to implement the following code, where Z is an integer array (4 bytes per element) which is located at address 0x120.
```
for(r0 = 0; r0 <= 20; r0++)
  Z[r0] += 0x10;
```

96. Write a suitable ARM assembly segment of code to implement the following code.
```
if ((r0 == r1) && (r2 == r3)) r4 += 16 else r5 += 32;
r6 += 64;
```

97. Write a suitable ARM assembly segment of code to implement the following code.
```
if ((r0 == r1) && (r2 == r3)) r4 /= 16 else r5 *= 32;
r6 -= 64;
```

98. Write a suitable ARM assembly segment of code to implement the following code.

```
r0 = 10;
r1 = 1;
while(r0 >0)
{ r1 += r1*65;
  r1 = r1 + r0 << 10;
  if(r1 is odd)
  THEN  r0 = r0 - 1;
  ELSE  r0 = r0 – 2;
}
```

99. Write a suitable ARM assembly segment of code to implement the following code.

```
r0 = 10;
r1 = 1;
{ r1 += r1*65;
  r1 = r1 + r0 << 10;
  if(r1 is even)
  THEN  r0 = r0 - 1;
  ELSE  r0 = r0 – 2;
}
until(r0 < 0)
```

100. Write an ARM code to implement without using any multiplication instruction.

```
if(r0 > r1)
{ r2 = 65535 * r3;
}
else
{ if(r0 == r1)
  { r2 = 65536 * r3;
  }
  else
  { r2 = 65537 * r3;
  }
}
```

101. Write a suitable ARM assembly segment of code to implement the following code.

```
if((r0 == r1) && (r2 == r3))
   r4 /= 4096;
else
   r5 *= 4096;
r6 -= 4096;
```