Study Questions (Chapter 03 – Part 7)

1. Consider the following assembly program

```
   AREA branching, CODE, READONLY
   ENTRY
A  ANDEQ r0,r0,r0
B  ANDEQ r0,r0,r0
C  ANDEQ r0,r0,r0
   BEQ   F
D  ANDEQ r0,r0,r0
E  ANDEQ r0,r0,r0
F  ANDEQ r0,r0,r0
   END
```

   a) Encode these instructions to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instructions to generate the original assembly instruction.

2. Write an ARM data-processing instruction with all operands as registers.
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

3. Write an ARM data-processing instruction with a literal operand.
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

4. Write an ARM move instruction with all operands as registers.
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

5. Write an ARM negative instruction with all operands as registers.
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

6. Write an ARM comparison instruction with all operands as registers.
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

7. Write an ARM data-processing instruction, where operand 2 has static LSL
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

8. Write an ARM data-processing instruction, where operand 2 has dynamic LSL
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

9. Write an ARM data-processing instruction, where operand 2 has static LSR
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

10. Write an ARM data-processing instruction, where operand 2 has dynamic LSR
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

11. Write an ARM data-processing instruction, where operand 2 has static ASR
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

12. Write an ARM data-processing instruction, where operand 2 has dynamic ASR
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

13. Write an ARM data-processing instruction, where operand 2 has static ROR
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

14. Write an ARM data-processing instruction, where operand 2 has dynamic ROR
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

15. Write an ARM data-processing instruction, where operand 2 has RRX
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

16. Write an ARM data-processing instruction with a conditional execution code and all operands as registers.
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

17. Write an ARM data-processing instruction with a conditional execution code and a literal operand.
   a) Encode this instruction to machine language.
   b) Use the ARM simulator to verify that your answer is correct.
   c) Decode the generated machine language instruction to generate the original assembly instruction.

18. Write an ARM data-processing instruction with a conditional execution code and shift operation.
    a) Encode this instruction to machine language.
    b) Use the ARM simulator to verify that your answer is correct.
    c) Decode the generated machine language instruction to generate the original assembly instruction.

19. Question 3.17 on 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?

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

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

22. 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;
  }
}
```

23. Explain what this fragment of code does.

```
CMP   R0,R1
RSBGT R2,R3,R3,LSL#16
MOVEQ R2,R3,LSL#16
ADDLT R2,R3,R3,LSL#16
```

24. 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;
```

25. Explain what this fragment of code does.

```
TEQ   R0,R1
TEQEQ R2,R3
ADDEQ R4,R4,#16
ADDNE R5,r5, #32
ADD   R6,r6,#64
```

26. 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;
```

27. Explain what this fragment of code does.

```
TEQ   R0,R1
TEQEQ R2,R3
MOVEQ R4,R4,ASR#12
MOVNE R5,r5,LSL#12
SUB   R6,r6,#4096
```