

## Study Questions (Chapter 03 – Part 6)

1. Question 3.29 on page 225: Some machines have a find-first-one instruction that counts the location of the first bit set to 1 within the word. Write an ARM sequence of instructions that takes the word in r0 and puts the locations of the first bit set to 1 in r1. Count from the left so that if bit 31 is set, the value returned should be 0. If bit 0 is set, the value returned is 31. If no bit is set, the value returned should be 32.
2. Write a suitable ARM assembly segment of code to implement the following code.

```
r0 = 255;
r1 = 1;
while(r0 >=0)
{ r1 += r1*8;
  r1 = r1 + r0 >> 2;
  if(r1 is odd)
  THEN  r0 = r0 - 64;
  ELSE  r0 = r0 - 96;
}
```

3. Explain what this fragment of code does. What are the values of registers at the end of the execution?

```
                MOV    R0, #255
                MOV    R1, #1
loop            CMP    R0, #0
                BLT    whileExit
                ADD    R1, R1, R1, LSL#3
                ADD    R1, R1, R0, ASR#2
                TST    R1, #1
odd             SUBNE  R0, #64
even           SUBEQ  R0, #96
                B      loop
whileExit
```

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

```
r0 = 255;
r1 = 1;
{
  r1 = r1*9;
  r1 = r1 + r0 >> 2;
  if(r1 is even)
  THEN  r0 = r0 - 64;
  ELSE  r0 = r0 - 96;
} until(r0 <=0)
```

5. Explain what this fragment of code does. What are the values of registers at the end of the execution?

```
                MOV    R0, #255
                MOV    R1, #1
repeat          ADD    R1, R1, R1, LSL#3
                ADD    R1, R1, R0, ASR#2
                TST    R1, #1
even           SUBEQ  R0, #64
odd            SUBNE  R0, #96
                CMP    R0, #0
                BGT    repeat
```

6. Write a suitable ARM code to implement the following code segment.

```
int total;
int i;

total = 0;
for (i = 10; i > 0; i--)
{
    total += i;
}
```

7. Explain what this fragment of code does. What are the values of registers at the end of the execution?

```
        MOV    R0, #0           ; R0 accumulates total
        MOV    R1, #10          ; R1 counts from 10 down to 1
for      ADD    R0, R0, R1
        SUBS   R1, R1, #1
        BNE    for
```

8. Write a suitable ARM code to implement the following code segment.

```
a = 40;
b = 25;
while (a != b) {
    if (a > b) a -= b;
    else      b -= a;
}
```

9. Explain what this fragment of code does. What are the values of registers at the end of the execution?

```
        MOV    R0, #40          ; R0 is a
        MOV    R1, #25          ; R1 is b
while    CMP    R0, R1
        SUBGT   R0, R0, R1
        SUBLT   R1, R1, R0
        BNE     while
halt     B       halt
```

10. Write a suitable ARM code to implement the following code segment.

```
iters ← 0
while n ≠ 1:
    iters ← iters + 1
    if n is odd:
        n ← 3 × n + 1
    else:
        n ← n / 2
```

11. Explain what this fragment of code does. What are the values of registers at the end of the execution?

```

MOV  R0, #5           ; R0 is the current number
MOV  R1, #0           ; R1 is a count of the number of iterations
while ADD R1, R1, #1   ; increment number of iterations
TST  R0, #1           ; test whether R0 is odd
BEQ  even
ADD  R0, R0, R0, LSL #1 ; if odd, set R0 = R0 + (R0 << 1) + 1
ADD  R0, R0, #1       ; and repeat (guaranteed R0 > 1)
B    while
even  MOV R0, R0, ASR #1 ; if even, set R0 = R0 >> 1
SUBS R7, R0, #1       ; and repeat if R0 != 1
BNE  again
halt  B    halt       ; infinite loop to stop the computation
```