CS1021 Tutorial #2
Machine Code and Binary Arithmetic

1 Machine Code

(a) Translate the following sequence of ARM Assembly Language instructions into machine code in hexadecimal form, using the machine code instruction templates in Figure 1.

```
ADD   R0, R5, R5
ADD   R0, R0, R5
MOV   R8, R7
ADD   R7, R7, R0
MOV   R9, R7
```

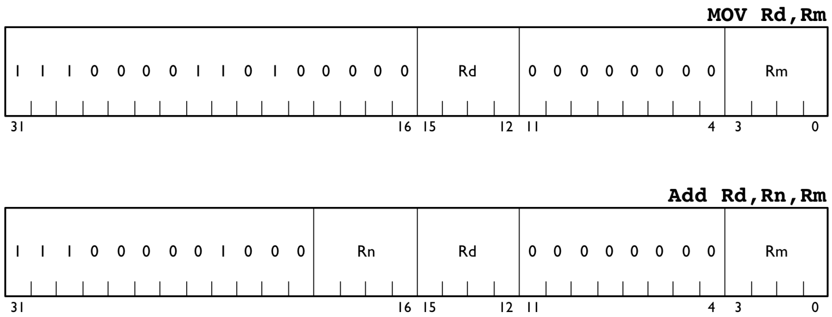

Figure 1: Instruction Templates

(b) Consider the ARM Machine Code Instruction 0xE0865008. Using the instruction templates in Figure 1, determine what what action the instruction would perform when executed (operation and operands).

(c) Modify the instruction from part (b) so that it updates the condition code flags on execution

2 Addition of Binary Numbers

Add the following binary values. Verify your solutions by converting the operands and results from binary to decimal.

(a) $0101 + 1010$

(b) $0111 + 0001$

(c) $1101 + 0101$

(d) $10011100 + 01111000$

3 Subtraction of Binary Numbers

Subtract the following binary values. Verify your solutions by converting the operands and results from binary to decimal.

(a) $10110 - 10010$

(b) $10010 - 01101$

4 Multiplication of Binary Numbers

The following is an example of four-bit binary multiplication:

$$\begin{array}{cccccccc}
 & & & 1 & 0 & 0 & 1 & \\
 & & & 0 & 1 & 0 & 1 & \times \\
\hline
 & & & 1 & 0 & 0 & 1 & \\
 & & 0 & 0 & 0 & 0 & & \\
 & 1 & 0 & 0 & 1 & & & \\
0 & 0 & 0 & 0 & & & & \\
\hline
 & 1 & 0 & 1 & 1 & 0 & 1 &
\end{array}$$

(a) Multiply the following binary values. Verify your solutions by converting the operands and results from binary to decimal.

   (i) 0101×0111

   (ii) 110100×10110

(b) How many bits are required to store the result of the multiplication of two n-bit numbers?

5 Modulo Arithmetic

Add the following values using Modulo-16 arithmetic in both **4-bit binary** and **decimal**:

(a) $2 + 3$

(b) $8 + 9$

(c) $10 + 12$

6 2's Complement

(a) State the range of integers that can be represented using a two's complement representation with the following number of bits:

   (i) 8 bits

   (ii) 32 bits

(b) Convert each of the following decimal integer values to its **8-bit** binary equivalent, assuming a two's complement representation:

(i) 0

(ii) 4

(iii) -4

(iv) -27

(c) For each of the following arithmetic operations, convert the operands into their 8-bit binary equivalents, assuming a two's complement representation. Calculate the result using binary arithmetic. Convert the result back to decimal form to verify that you obtained the correct result.

(i) $10 + (-10)$

(ii) $(-20) + 10$

(iii) $10 + (-5)$

(iv) $127 + 1$

7 64-bit and 128-bit Arithmetic

On a 32-bit processor, such as the ARM7TDMI, we are limited to performing arithmetic operations on 32-bit values, so we need to handle arithmetic on larger values differently.

The addition of two 64-bit values can be split into two 32-bit additions. The upper (most-significant) and lower (least-significant) 32-bits of each 64-bit value must be stored in two separate registers. The following example shows how two 64-bit values might be stored in R2, R3, R4 and R5.

```
start

        ; Set some test values for A and B
        LDR     r2, =0x00000E44         ; Aupr
        LDR     r3, =0x32A84FE6         ; Alwr
        LDR     r4, =0x00000000         ; Bupr
        LDR     r5, =0xF4E00322         ; Blwr

stop    B       stop
```

Write ARM Assembly Language programs to perform each of the operations listed below.

Hint: Split each operation into two (or more) 32-bit operations. Remember that the Carry flag will be set by an ADDS instruction if the addition resulted in a carry out. You may want to look up the functionality of the ADC instruction in the ARM Architecture Reference Manual.

(a) 64-bit addition.

(b) 128-bit addition.

(c) 64-bit subtraction.