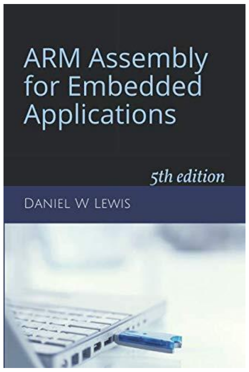

ARM Instructions Worksheet #5

Multiplication

Single/Double-Length, Signed/Unsigned

Prerequisite Reading: Chapter 5

Revised: March 26, 2020

Objectives: *To use the web-based simulator ("CPULator") to better understand …*

1. The `MUL`, `SMULL`, and `UMULL` instructions
2. Single versus double-length products.
3. Signed versus unsigned multiplication.

To do offline: Answer the questions that follow the listing below. (Numbers at far left are memory addresses.)

```
                    .syntax         unified
                    .global         _start

00000000  _start:   LDR             R2,=+3          // *** EXECUTION STARTS HERE ***
00000004            LDR             R3,=-5
00000008            MUL             R0,R2,R3
0000000C            SMULL           R0,R1,R2,R3
00000010            LDR             R2,=3
00000014            LDR             R3,=0x80000000
00000018            MUL             R0,R2,R3
0000001C            UMULL           R0,R1,R2,R3

00000020  done:     B               done
                    .end
```

Note: Use this hex to decimal converter to convert 64-bit products to decimal.

What is left in `R2` by the `LDR` pseudo-instruction at 00000000_{16}? R2 (8 hex digits): ____ R2 (as signed decimal): ____

What is left in `R3` by the `LDR` pseudo-instruction at 00000004_{16}? R3 (8 hex digits): ____ R3 (as signed decimal): ____

What product is left in `R0` by the `MUL` instruction at 00000008_{16}? R0 (8 hex digits): ____ R0 (as signed decimal): ____

What is left in `R1.R0` by the `SMULL` instruction at $0000000C_{16}$? R1 (8 hex digits): ____ R0 (8 hex digits): ____ R1.R0 (as signed decimal): ____

Did the single-length signed product produced by the previous `MUL` overflow? Yes: ☐ No: ☐

What is left in `R2` by the `LDR` pseudo-instruction at 00000010_{16}? R2 (8 hex digits): ____ R2 (as unsigned decimal): ____

What is left in `R4` by the `LDR` pseudo-instruction at 00000014_{16}? R3 (8 hex digits)): ____ R3 (as unsigned decimal): ____

What product is left in `R0` by the `MUL` instruction at 00000018_{16}? R0 (8 hex digits): ____ R0 (as unsigned decimal): ____

What is left in `R1.R0` by the `UMULL` instruction at $0000001C_{16}$?

R1 (8 hex digits)	R0 (8 hex digits)	R1.R0 (as unsigned decimal)

Did the single-length unsigned product produced by the previous `MUL` overflow? Yes: ☐ No: ☐

Getting ready: Now use the simulator to collect the following information and compare to your earlier answers.

1. Click here to open a browser for the ARM instruction simulator with pre-loaded code.

Note: You can change the number format in the "Settings" window between hex, unsigned decimal and signed decimal as needed. For 64-bit products, use this hex to decimal converter.

***Step 1:** Press F2 exactly **2** times to execute the two `LDR` pseudo-instructions (`MOV`, `MVN`) to provide the operands*

What is left in `R2` by the `LDR` pseudo-instruction at 00000000_{16}?

R2 (8 hex digits)	R2 (as signed decimal)

What is left in `R3` by the `LDR` pseudo-instruction at 00000004_{16}?

R3 (8 hex digits)	R3 (as signed decimal)

***Step 2:** Press F2 exactly **once** to execute the `MUL R0,R2,R3` instruction.*

What product is left in `R0` by the `MUL` instruction at 00000008_{16}?

R0 (8 hex digits)	R0 (as signed decimal)

***Step 3:** Press F2 exactly **once** to execute the `SMULL R0,R1,R2,R3` instruction.*

What is left in `R1.R0` by the `SMULL` instruction at $0000000C_{16}$?

R1 (8 hex digits)	R0 (8 hex digits)	R1.R0 (as signed decimal)

Did the single-length signed product produced by the previous `MUL` overflow? Yes: ☐ No: ☐

***Step 4:** Press F2 exactly **2** times to execute the two `LDR` pseudo-instructions (`MOV`, `MOV`) to provide the operands*

What is left in `R2` by the `LDR` pseudo-instruction at 00000010_{16}?

R2 (8 hex digits)	R2 (as unsigned decimal)

What is left in `R4` by the `LDR` pseudo-instruction at 00000014_{16}?

R3 (8 hex digits))	R3 (as unsigned decimal)

***Step 5:** Press F2 exactly **once** to execute the `MUL R0,R2,R3` instruction.*

What product is left in `R0` by the `MUL` instruction at 00000018_{16}?

R0 (8 hex digits)	R0 (as unsigned decimal)

***Step 6:** Press F2 exactly **once** to execute the `UMULL R0,R1,R2,R3` instruction.*

What is left in `R1.R0` by the `UMULL` instruction at $0000001C_{16}$?

R1 (8 hex digits)	R0 (8 hex digits)	R1.R0 (as unsigned decimal)

Did the single-length unsigned product produced by the previous `MUL` overflow? Yes: ☐ No: ☐