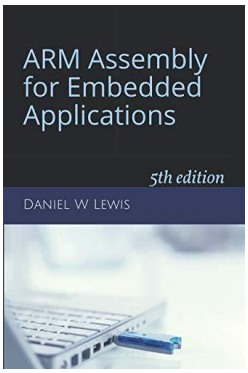

ARM Instructions Worksheet #6

Conditional Branch

Signed versus Unsigned

Prerequisite Reading: Chapter 6
Revised: March 25, 2020

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

1. Single versus unsigned conditional branch instructions.

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         R0,=0xFFFFFFFF      // *** EXECUTION STARTS HERE ***
00000004  loop:     LDR         R1,=0x11111         // Turn on all flags
00000008            CMP         R0,1
0000000C  test1:    BLO         test2               // Branch if R0 <  1 (unsigned)
00000010            SUB         R1,R1,0x10000       // Did not branch: Turn off LO flag
00000014  test2:    BHI         test3               // Branch if R0 >  1 (unsigned)
00000018            SUB         R1,R1,0x01000       // Did not branch: Turn off HI flag
0000001C  test3:    BLT         test4               // Branch if R0 < +1 (signed)
00000020            SUB         R1,R1,0x00100       // Did not branch: Turn off LT flag
00000024  test4:    BGT         test5               // Branch if R0 > +1 (signed)
00000028            SUB         R1,R1,0x00010       // Did not branch: Turn off GT flag
0000002C  test5:    BEQ         next                // Branch if R0 == 1
00000030            SUB         R1,R1,0x00001       // Did not branch: Turn off EQ flag
00000034  next:     ADD         R0,R0,1             // Increment R0
00000038            B           loop                //    and repeat.

                    .end
```

Note: The least-significant four hex digits of register `R1` will be used to indicate which conditions were satisfied according to the table shown at the right:

R1 contents	LO	HI	LT	GT	EQ
0x00010000	√				
0x00001000		√			
0x00000100			√		
0x00000010				√	
0x00000001					√

What is in `R0` the 1st time execution arrives at address 00000038_{16}?

R0 (as unsigned decimal): 00000000

R0 (as signed decimal): 0

Which conditions does `R1` indicate as true for `R0` compared to `1`?

LO ☐ EQ ☐ HI ☒ LT ☒ EQ ☐ GT ☐

What is in `R0` the 2nd time execution arrives at address 00000038_{16}? R0 (as unsigned decimal): 00000001 R0 (as signed decimal): 1

Which conditions does `R1` indicate as true for `R0` compared to `1`? LO [T] EQ [ ] HI [ ] LT [T] EQ [ ] GT [ ]

What is in `R0` the 3rd time execution arrives at address 00000038_{16}? R0 (as unsigned decimal): 00000002 R0 (as signed decimal): 2

Which conditions does `R1` indicate as true for `R0` compared to `1`? LO [ ] EQ [T] HI [ ] LT [ ] EQ [ ] GT [ ]

What is in `R0` the 4th time execution arrives at address 00000038_{16}? R0 (as unsigned decimal): 00000003 R0 (as signed decimal): 3

Which conditions does `R1` indicate as true for `R0` compared to `1`? LO [ ] EQ [ ] HI [T] LT [ ] EQ [ ] GT [T]

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.
1. In the "Disassembly" window, click in the grey area left of the `ADD` instruction. The red dot (●) is a breakpoint where the simulation will pause *before* executing this instruction.

Notes:

1. The `BLO` instruction in the "Editor" window will appear as an equivalent `BCC` instruction in the "Disassembly window.
2. You can change the number format in the "Settings" window between hex, unsigned decimal and signed decimal as needed.

Step 1: *Press F3 exactly* ***once*** *to run the simulation and stop at the breakpoint.*

What is in `R0` the 1st time execution arrives at address 00000038_{16}? R0 (as unsigned decimal): 00000000 R0 (as signed decimal): 0

Which conditions does `R1` indicate as true for `R0` compared to `1`? LO [ ] EQ [ ] HI [T] LT [T] EQ [ ] GT [ ]

Step 2: *Press F3 exactly* ***once*** *to run the simulation and stop at the breakpoint.*

What is in `R0` the 2nd time execution arrives at address 00000038_{16}? R0 (as unsigned decimal): 00000001 R0 (as signed decimal): 1

Which conditions does `R1` indicate as true for `R0` compared to `1`? LO [T] EQ [ ] HI [ ] LT [T] EQ [ ] GT [ ]

Step 3: *Press F3 exactly* ***once*** *to run the simulation and stop at the breakpoint.*

What is in `R0` the 3rd time execution arrives at address 00000038_{16}? R0 (as unsigned decimal): 00000002 R0 (as signed decimal): 2

Which conditions does `R1` indicate as true for `R0` compared to `1`? LO [ ] EQ [T] HI [ ] LT [ ] EQ [ ] GT [ ]

Step 4: *Press F3 exactly* ***once*** *to run the simulation and stop at the breakpoint.*

What is in `R0` the 4th time execution arrives at address 00000038_{16}? R0 (as unsigned decimal): 00000003 R0 (as signed decimal): 3

Which conditions does `R1` indicate as true for `R0` compared to `1`? LO [ ] EQ [ ] HI [T] LT [ ] EQ [ ] GT [T]