Imperial College London

Introduction to Computer Architecture – EEE2&EIE1 labs

Lab 4 – Condition codes

Tom Clarke & Max Cattafi

(t.clarke@imperial.ac.uk, m.cattafi@imperial.ac.uk)

(For worked examples, pen and paper exercises and self-test questions, see the 'Coursework' page of the course, accessible from the link on "Introduction to Computer Architecture" on Blackboard and on `https://intranet.ee.ic.ac.uk/t.clarke/arch/contents.html`)

Carry on

Write an ARM assembly program which implements 64 bit (unsigned) int addition in both versions: with and without making use of `ADC` (see also the "64 bit addition" slides on the lecture notes). Test your program on suitable operands, for instance `0xCFFFFFFFF + 0x1`.

V for (signed) oVerflow

The V flag is set when signed overflow occurs: for instance when the result of the sum of two positive signed integers could be interpreted as a negative signed integer. Write an ARM assembly program which sets the V condition code (e.g. by performing an addition with a suitable version of the instruction and on suitable operands).

Exercises revisited

Try to write alternative versions of the exercises proposed in the last session making use (whenever suitable), instead of `CMP`, of instructions setting condition codes (if you already know or want to learn about it, you can also try using, whenever suitable, conditional execution instead of branches).

Labels are for memory

Test the following code (pay attention to the 'memory' visualization area available in the bottom right of *µVision* during the debug/run mode):

```
    AREA        ArmLabels, CODE
    ENTRY
        MOV r0, #0
LOADME  ADD r0, r0, #1
        LDR r1, LOADME
        LDR r2, =LOADME
STOP    B       STOP
    END
```

What operations are performed by the program? Edit the program in order to achieve (without making use of `B` or `B`-derived instructions) an infinite loop which, at each iteration, increases by 1 the value in `r0`.

Conjectures

Wikipedia describes the Collatz conjecture algorithm as follows:

> Take any natural number n. If n is even, divide it by 2 to get n/2. If n is odd, multiply it by 3 and add 1 to obtain 3n+1. Repeat the process (which has been called "Half Or Triple Plus One", or HOTPO) indefinitely. The conjecture is that no matter what number you start with, you shall always eventually reach 1.

Write an ARM assembly program which tests the Collatz conjecture for the number initially stored in `r0` and keeps in `r1` the number of iterations and in `r2` the maximum of the numbers generated in the sequence (for instance if `r0` initially contains 6, at the end of the program execution `r1` should contain 8 and `r2` 16).

Remember that $3n = n + n + n$.

For the sake of this exercise you can obtain n/2 as follows: if e.g. n is initially contained in `r3`, after the execution of
`ASR r3, #1`
`r3` contains n/2.

In order to test if n is even or odd, have a look at slide "Data Processing Instructions (3) - Logical Register Operations" from the lecture notes.