ARM Flow Control Instructions

Tom Kelliher, CS 220

1 Administrivia

Today's Objectives

1. Understand and use unconditional and conditional branch instructions, and the compare instruction.
2. Manually compile if/else blocks into ARM assembly.
3. Manually compile for and while loops into ARM assembly.

Next Up

Read 3.7–3.8. Skip 3.7.3–3.7.5.

1. Use indirect addressing to move data between registers and memory.
2. Manipulate numeric and character arrays.
3. Use subroutine call and return instructions to implement functions without a function call stack.

2 Warm-Up

1. The instruction

   ```
   add r1, r2, r3
   ```

   updates the condition codes.

   True/False.

2. The difference between the instruction

```
cmp r1, r2
```

and the instruction

```
subs r2, r1, r2
```

is

(a) The `subs` instruction changes the value of `r2`.

(b) The `cmp` instruction doesn't set the condition codes.

(c) The `subs` instruction doesn't set the condition codes.

(d) The `cmp` instruction performs `r1 - r2` whereas `subs` performs `r2 - r1`.

3. The C code

```
if (r1 <= 0)
   r2 = 1;
```

assembles to

```
        cmp r1, #0
        bgt skip
        mov r2, #1
skip
```

True/False

4. The C code

```
if (0 <= r1 && r1 <= 1)
   r2 = 1;
```

assembles to

(a)
```
         cmp r1, #0
         ble skip
         cmp r1, #1
         bge skip
         mov r2, #1
skip
```

(b)
```
         cmp r1, #0
         blt skip
         cmp r1, #1
         bgt skip
         mov r2, #1
skip
```

(c)
```
         cmp r1, #0
         bge skip
         cmp r1, #1
         ble skip
         mov r2, #1
skip
```

(d) Please, don't play your Jedi mind tricks on me today.

3 Problems

Refer to Table 3.2 on pg. 177 for the conditional branch instructions.

1. Implement the following C fragment in ARM assembly. Assume that `a` is stored in register `r0` and `b` is stored in register `r1`.

   ```
   if (a < 0)
      b = -1
   else if ( a > 0)
      b = 1;
   else
      b = 0;
   ```

2. Implement the following C fragment in ARM assembly. Assume that `k` is stored in register `r0` and `i` is stored in register `r1`.

   ```
   k = 0;
   for (i = 0; i < 32; i = i + 2)
      k = k + i;
   ```

3. Implement the following C fragment in ARM assembly. Assume that `k` is stored in register `r0` and `i` is stored in register `r1`.

   ```
   if (i < 0 || i >= 2)
      k = 0;
   else
      k = 1;
   ```

4. Implement the following C fragment in ARM assembly. Assume that `i` is stored in register `r0`, `n` is stored in register `r1`, and `fact` is stored in register `r2`.

   ```
   if (n < 0)
      fact = -1;
   else
   {
      fact = 1;
      for (i = 1; i <= n; i++)
         fact *= i;
   }
   ```

   Create a uVision project to test your solution to this problem.