name:	Vince Rothenberg
course:	csci 10
assignment:	homework 10
prepared:	Tue, Apr 14, 2020 // 9:51 am

1. What size (in bits) are ARM instructions? Do ARM instructions provide access to conditional execution?

Each instruction in the ARM instruction set is 32 bits in size.

Yes, ARM instructions have access to other useful features such as conditional instructions and inline barrel shifter. Without access to conditional instructions, code needs to be handled by branching, which is more expensive. The in-line barrel shifter gives the instruction the ability to shift bits within the registers as part of the instruction itself, which eliminates the need for having separate instructions for shifting.

2. What size (in bits) are Thumb instructions? What are the advantages and disadvantages of Thumb versus ARM instructions?

In Thumb state, each instruction is 16 bits in size.

Advantage: Smaller code size.

Disadvantages: Very few instructions are conditional. Also, there is no access to the in-line barrel shifter, so separate instructions are needed for shifting bits. What this means in practice is that Thumb code would generally be slower to execute than ARM code (since more Thumb instructions might be needed to do the job than the number of ARM instructions), but it can help save code size.

3. What size (in bits) are Thumb-2 instructions? What are the advantages of Thumb-2 over ARM and Thumb instructions?

Thumb-2 has access to both 16 and 32 bit instructions.

Thumb-2 offers a â€œbest of both worldsâ€• compromise between ARM and Thumb, a balance of performance, support for conditional execution in the form of â€œIf-thenâ€• (IT) constructs, and aims to deliver the speed of ARM (32 bit) state code with the code density of Thumb (16 bit) state code.

4. Does the ARMv7-M architecture support ARM instructions? Thumb/Thumb-2 instructions?

Not ARM instructions, ARMv7-M supports Thumb/Thumb-2 only. It supports a large number of 32-bit instructions that Thumb-2 technology introduced into the Thumb instruction set. Much of the functionality available is identical to the ARM instruction set supported alongside the Thumb instruction set in ARMv6T2 and other ARMv7 profiles.

5. Describe the ARM TST instruction.

The TST instruction allows you to test either if one or more bits of a register are clear, or at least one bit is set. The internal operation is an AND of the two operands.

Operand 1 is a register, operand 2 may be a register, shifted register, or an immediate value (which may be shifted). TST is important in conditional execution and decision making.

```
TST  r0, 0x1
IT      NE // 0x1 is set

TST  r0, 0x2
IT      EQ // 0x2 not set
```

6. Write the ARM instruction to test if bit 2 is set in register r0.

```
TST    R0, #(1<<2)
IT        NE  // Bit 2 is set
```

7. Describe the ARM TEQ instruction.

The TEQ instruction allows you to test is two operands are equal. Unlike CMP, the V flag is not updated. The internal operation is a logical EOR of the two operands.

Operand 1 is a register, operand 2 may be a register, shifted register, or an immediate value (which may be shifted). TEQ is important in conditional execution and decision making.

```
TEQ  <op 1>, <op 2>
```

8. Write the ARM instruction to test if r0 and r1 are equivalent.

```
TEQ    R0, R1
IT        EQ // Yes, equivalent
```

9. Describe the Thumb-2 "it" instruction: (1) what does the instruction do, (2) what is the term used to describe instructions affected by the instruction, (3) what happens when the condition code provided as an argument to "it" evaluates to true?

The Thumb-2 it instruction conditionally executes up to four subsequent instructions.

The instructions affected by an it instruction are said to be in an it block.

If the condition code (given as an argument to the instruction) evaluates to true, then the next instruction is executed. Up to three additional t (then) or e (else) codes can be added to control the execution of the subsequent instructions.

```
CMP      R6, 127
IT           EQ
MOVEQ R7, 0
```

10. Write the Thumb-2 instructions to (1) test if r0 and r1 are equal, (2) create an if-then block to evaluate the equals (r0 == r1) condition code, and (3) add 1 to r1 if the condition code evaluates to true.

```
TEQ      R0, R1
IT          EQ
ADDEQ R1, 1
```